COHERENS

A Coupled Hydrodynamical-Ecological Model for Regional and Shelf Seas

User Documentation

Release 8.4, September 1999

Patrick J. Luyten, John Eric Jones, Roger Proctor, Andy Tabor, Paul Tett and Karen Wild-Allen

MAS3-CT97-0088

COHERENS

Addresses

Patrick Luyten
Management Unit of the Mathematical Models of the North Sea (MUMM)
Gulledelle 100
B-1200 Brussels
Belgium

John Eric Jones and Roger Proctor
Proudman Oceanographic Laboratory (POL)
Bidston Observatory
Birkenhead, Merseyside L43 7RA
United Kingdom

Andy Tabor
British Oceanographic Data Centre (BODC)
Bidston Observatory
Birkenhead, Merseyside L43 7RA
United Kingdom

Paul Tett and Karen Wild-Allen
Napier University (NUE)
Department of Biological Sciences
10 Collinton Road
Edinburgh EH10 5DT
United Kingdom

License Agreement

Acknowledgements

The authors wish to acknowledge the following persons who made an important contribution to the development of the program: Judith Wolf (general concept of the program), Kevin Ruddick (circulation and transport modules), Eric Deleersnijder (numerical methods), Claire Smith (biology), Sarah Jones (resuspension and sediment module) and Kajo Mirbach (Lagrangian tracers). Tom De Mulder, Ian James and other colleagues of the PROFILE group are thanked for their advice. Special thanks are for Jean-Pierre Malisse and the members of the CAMME team for computer support. We wish to thank the colleagues who were prepared to review the manuscript: Job Baretta (RIKZ), Ian James and John Huthnance (POL), Rob Uittenbogaard and Robert Vos (Delft Hydraulics), Boris Kelly-Gerreyn (Southampton Oceanography Centre), Alain Norro and Kevin Ruddick (MUMM). Finally, this work, performed by a multinational group, would not have been possible without the financial support by the European Union under contracts MAST-0050-C (Profile I), MAST-900064, MAS2-CT93-0054 (Profile II) and MAS3-CT97-0088 (Coherens).

Contents

List of Figures

List of Tables

Preface

COHERENS is a three-dimensional multi-purpose numerical model for coastal and shelf seas. The hydrodynamic model is coupled to biological, resuspension and contaminant models, and resolves mesoscale to seasonal processes. The code has been developed over the period 1990-1998 by a multinational group, as part of the MAST projects PROFILE, NOMADS and COHERENS funded by the European Union.

The COHERENS model is now available for the scientific community and can be considered as a tool for a better understanding of the physical and ecological processes and for the prediction and monitoring of waste material in shelf seas and coastal areas. The design, testing and validation of a three-dimensional integrated model requires years of efforts. Its ease of implementation across a range of computing platforms means that it will be attractive to groups with a sufficient modelling expertise who have need of sophisticated model products. Important advantages of the model are its transparency due to its modular structure and its flexibility because of the possibility of selecting different processes, specific schemes or different types of forcing for a particular application. This allows its use for process studies as well as for predictive or operational purposes without prior knowledge of its detailed structure. Future developments can easily be implemented without affecting the core of the program.

Part I

Introductory Manual

Chapter 1

General Overview of COHERENS

1.1 Introduction

Concern over water quality in European shelf seas during recent years has given rise to the development of coupled physical-biological numerical models as tools for understanding the relevant physical and biological processes and the influence of human activity on the ecological conditions (North Sea Task Force, 1993). The aim is to predict the effect of the changing conditions on the biota and to simulate the input and dispersion of contaminants. This is an ambitious and multidisciplinary task which requires the collaboration of a variety of specialists and consumes a substantial number of years of scientific and programming effort for designing, developing and testing the model code and to validate the model against observational data.

During the last decades a series of three-dimensional hydrodynamic models for shelf seas have been developed at different research institutes. An overview can be found in e.g. Nihoul and Jamart (1987), Dyke and Davies (1992), Lynch and Davies (1995). This includes the well known Backhaus (1985) model and the Princeton Ocean Model (Blumberg and Mellor, 1987). In recent years some of these models have been coupled with a biological module (Aksnes et al., 1995; van den Bergh et al., 1996) or with a sediment or contaminant transport model. While some of the model equations and numerical schemes have become standard (e.g. Navier-Stokes equations, mode splitting technique), the parameterisations of some processes (turbulence, surface waves, biological interactions, ...) have to be improved. Many turbulence schemes are, for example, available from the literature. Further progress may be expected in forthcoming years. The coupling of physical and biological modules is often limited by memory and CPU time if the module involves a large number of state variables. An essential requirement is therefore that biological processes are parameterised in a way that makes the numerical integration more efficient.

COHERENS is a three-dimensional hydrodynamic multi-purpose model for coastal and shelf seas, which is coupled to biological, resuspension and contaminant models, and resolves mesoscale to seasonal scale processes. The program has been developed over the period of 1990–1998 by a multinational European group, as part of the MAST projects

PROFILE, NOMADS and COHERENS funded by the European Union. It has primarily been used to simulate conditions in the North Sea and stratified coastal areas (Huthnance, 1997; Proctor, 1997; Luyten, 1999).

The COHERENS model is now available for the scientific community and can be considered as a tool for a better understanding of the physical and ecological processes and for the prediction, monitoring of waste material in shelf seas and coastal areas. Its ease of implementation across a range of computing platforms means that it will be attractive to groups with a sufficient expertise in modelling, who have need of sophisticated model products. Important advantages of the model are its transparency due to its modular structure and its flexibility because of the possibility of selecting different processes, specific schemes or different types of forcing for a particular application. This allows its use for process studies as well as for predictive or operational purposes without prior knowledge of its detailed structure. Future developments can be implemented without affecting the core of the program.

1.2 Scientific background

An important feature of COHERENS is that the time step and the horizontal and vertical resolution can be defined by the user in relation to the relevant time scale and the horizontal and vertical length scales which are imposed mainly by the physical forcing conditions. These are associated with processes on the mesoscale (wind, tides, inertial oscillations), the synoptic scale (frontal structures), the seasonal scale (thermocline formation, plankton blooms) and the global scale (climatic processes). The corresponding time scales are of the order of a few hours, a week, a few months or several years. The time scales, inherent to the biology, are mostly of the order of the seasonal scale and therefore much larger than the physical ones. The coupling of the biology with the physics — mainly through advection and diffusion — imposes the necessary constraint that both parts of the program are solved numerically using the same time step. To limit the amount of CPU time the number of state variables in the biology, which are updated in time by solving a transport equation for each variable, has been kept at a minimum.

Shelf seas are often characterised by strong horizontal density gradients, e.g. haline fronts occur in coastal areas with an important river outflow and may have horizontal scales of the order of a few km or less. A two-layer structure is frequently observed in river plume fronts with a layer of fresh water floating above a tidally mixed bottom layer. The halocline separating the two layers has a thickness of only a few meters. The transition area between the shallow and deeper parts of a shelf sea is characterised by thermal fronts in the summer season. The water column in the shallow part is well mixed vertically, whereas a thermocline develops in the latter part during the spring and summer periods. The thermocline and halocline layers must be adequately resolved by the model since high chlorophyll concentrations are often observed in these layers during summer.

The vertical exchange of physical quantities, biological substances and contaminant concentrations are strongly controlled by turbulent processes. A wide range of turbulence schemes are available in the COHERENS model. In the bottom or benthic boundary layer, turbulence, generated by tidal friction, determines the vertical structure of the tidal currents. Sediment and organic material are deposited in an unconsolidated "fluff" layer and are easily resuspended by the action of the bottom stress. Remineralisation of this material in the seabed and the benthic boundary layer enriches the lower water column with nutrients. Turbulent diffusion into the thermocline, entrainment of thermocline water into the surface mixed layer, and lateral isopycnal transport from vertically mixed waters, provide mechanisms for the resupply of nutrients to the euphotic zone. These processes thus control the productivity of the sea in summer and early autumn, when phytoplankton growth in the surface mixed layer is restricted by low nutrient concentrations. Resuspension from the fluff layer depends on the bottom shear stress usually given as a function of the bottom roughness length. In shallow areas with large tides its value may be strongly enhanced by the non-linear interactions between currents and surface waves. The effect of current-wave interaction on the bed shear stress can be included in the simulations. The strong vertical stratification which prevails in the thermocline or halocline regions inhibits turbulence and impedes the penetration of resuspended material into the surface layer. The structure of the surface layer is governed by wind mixing, surface cooling and solar heating. The forcing is provided by the interactions at the air-sea interface using formulations for the surface fluxes of momentum, heat and salinity, as a function of meteorological forcing parameters.

In addition to physical transports (including sinking), the cycling of organic matter and nutrients is controlled by four groups of processes.

1. The rates of production of organic material, and its rapid remineralisation, at rates of the order of 0.1 per day, depends on the dynamics of phytoplankton and pelagic microheterotrophs (including bacteria and protozoa), controlled by the availability of light and nutrients to the phytoplankton, and thus by the turbulent regime. These microbiological processes are parameterised in COHERENS by a microplankton component.

2. The slower remineralisation, at rates of the order of 0.01 per day, of detrital organic particles and refractory dissolved organic material, replaces nutrients more slowly, and draws on a larger, but more refractory, reservoir than the first group. COHERENS includes a detritus but not a dissolved organic matter compartment.

3. The penetration of light into the sea has consequences for photosynthesis and for heating and stratification of the water column, and depends in part on light absorption and scattering by organic substances, especially the pigments contained in phytoplankton. Mineral suspended solids also play a major role in light attenuation in many well mixed waters. COHERENS thus pays attention to water-column optics.

4. Finally, mortality due to predation of phytoplankton or microheterotrophs by higher trophic levels is important in cases when light and nutrients are adequately supplied. Such predation is not modelled dynamically but is supplied as a forcing time series.

An important prerequisite of contemporary models is their ability to simulate the advection and dispersion of waste material, oil spills and other dissolved substances (e.g. radioactive nucleides). The numerical method depends on the scale of the pollutant material. An approach using Langrangian particle tracking is preferred for small scale events whereas the Eulerian method using advection-diffusion transport equations is more efficient on larger scales. Both methods are available in COHERENS.

A useful model must not only be able to describe adequately the physical and biological processes but also take account of additional constraints by the numerics. A selection can be made between different forms of open sea and river boundary conditions depending on the type of model application and the availability of input data. The program has the capacity to resolve sharp frontal structures, such as river plumes and thermal fronts, and eddy structures with a strong vorticity. Sophisticated schemes are available for the advection of scalars (salinity, temperature, biological variables, contaminant concentrations) and for momentum. As an alternative, it is possible to use more diffusive schemes which have the advantage of consuming less CPU time.

Finally, the following two features are briefly mentioned. Firstly, the program allows to select either a Cartesian or a spherical computational grid. The former has the advantage that the horizontal grid axes can be rotated arbitrarily in the horizontal plane, whereas the latter takes account of the effects of the Earth's curvature. Secondly, the possibility is foreseen to run the program either as a full three-dimensional model or a point model in the vertical.

1.3 Description of the program

The program is written in FORTRAN 77 and has four major components :

1. A physical part with a general module for solving advection-diffusion equations.
2. A microbiological module which deals with the dynamics of microplankton, detritus, dissolved inorganic nitrogen and oxygen.
3. An Eulerian sediment module which deals with deposition and resuspension of inorganic as well as organic particles.
4. A component with both an Eulerian and a Lagrangian transport model for contaminant distributions.

The design of the program consists of a "core" part and of a series of modules. This modular design allows easy updating of any particular process, or the inclusion of an alternative solution method, or the addition of new processes. The core updates the current field by solving the Navier-Stokes equations and contains the advection-diffusion module. A series of switches (marked by a * below) allows the user to select whichever processes are required, for a particular numerical simulation.

The characteristics of the program can be summarised as follows :

1. General
   - Cartesian or spherical grid (*) defined by the user
   - the possibility to run the program as a one-dimensional point model in the vertical (*)
   - σ-coordinates in the vertical with the possibility to use non-uniform grid sizes
   - one time step for the update of all 3-D quantities
   - different schemes for the advection of scalars (*) and momentum (*) which can be updated in future versions (upwind, TVD, Lax-Wendroff, ...) and for horizontal diffusion (*)
   - the possibility to perform an harmonic analysis on user-defined variables (*)
   - various forms of data input with the possibility to use different time intervals for different components of the program (physics, biology, suspended material)
   - type of model output specified by the user (format, time step, grid locations, variables; time series, harmonic or time-averaged output; particle trajectories)
   - error traps which stop execution of the program after improper initialisation and in some other cases, and which provide an explanatory message
   - a graphics interface program which converts model output into the portable "netCDF" format, supported by several graphical software platforms (PV-Wave, IDL, Matlab, FERRET, ...)
2. Physics
   - The mode-splitting technique is used to solve the 2-D and 3-D momentum and continuity equations as in the Princeton Ocean Model.
   - It is possible to include temperature (*), salinity(*) or both in the simulation.
   - The absorption of solar radiation in the upper part of the water column is implemented by an optical module (*).
   - Density effects in the momentum and turbulence equations are included via an equation of state (*).
   - The program incorporates a variety of turbulence schemes ranging from simple algebraic formulations to one- or two-equation turbulence energy models, such as the Mellor-Yamada (1982) and $k-\varepsilon$ turbulence energy schemes (Rodi, 1984; Luyten et al., 1996) (*).
   - Various types of radiation conditions can be used at the open sea and river boundaries (*).
   - The effect of wave-current interaction on the bottom shear stress can be included using the formulations described in Signell et al. (1990), Davies and Lawrence (1994) (*).

   - Different formulations for the wind stress and surface heat fluxes are available (e.g. Geernaert, 1990) (*).

3. Biology (*)
   - Water-column biology and nutrient cycling are described by a microplankton-detritus module (Tett and Walne, 1995; Tett, 1998) with associated optical equations which take account of light attenuation by organic and inorganic material.
   - The module describes the cycling of carbon and nitrogen through microplankton and detrital compartments, with corresponding changes in dissolved concentrations of nitrate, ammonium and oxygen.
   - The microplankton compartment provides an efficient parameterisation of fast autotrophic and heterotrophic processes, effectively including most of the types of organisms that are specified separately in "microbiological" loop models (e.g. Baretta-Bekker et al., 1995).
   - Microplankton dynamics are mainly those suggested by the "cell-quota, threshold-limitation" theory described in Droop (1983).
   - The effects of mesozooplankton are imposed as a grazing pressure, which removes some microplankton and converts the rest to detritus, with a slower rate of decay. A portion of grazed microplankton nitrogen is immediately returned to the water column as ammonium.

4. Sediments (*)
   - All particulate compartments of the biological model have a sinking rate, and can deposit to the fluff layer.
   - The sediment model also describes suspended inorganic particle concentrations as one or two state variables, which may carry contaminants (in future versions) and which influence light attenuation.
   - Deposition and resuspension of particulates involves a "fluff" layer (Jago et al., 1993) of finite capacity. Organic particles decay in this layer and may be actively drawn into the sediment by benthic animals.
   - Resuspension increases as a power function of bed stress (Jones et al., 1995), and the absence of the usual threshold for the resuspension of cohesive sediments can be seen as a simple parameterisation for a heterogeneous population of particles and bed types.

5. Contaminants (*)
   - Two transport models are available (*).
   - The first one uses an Eulerian approach and solves advection-diffusion type equations for a number of user-defined dissolved substances.

- The second is a Lagrangian type model using passive tracers. Vertical and horizontal diffusion of suspended particles is determined by a random walk method (Maier-Reimer, 1980).
- In both models new input of suspended material either by river discharge or via open sea boundaries can be defined by the user.

A number of default values are adopted for switches and model parameters (hydrodynamics only, default turbulence scheme, ...), making it easier for a less experienced user to run the model. All switches, model parameters and arrays, including their default settings, are discussed in Chapter IV-2 and presented in tabular form at the end of that chapter. A set of test cases has already been developed. They are intended to check the portability of the code, to verify the model with analytical solutions, to test a particular scheme or specific module, or to show some realistic responses (e.g. seasonal cycles in the North Sea, idealised river outflow).

1.4 User experience

The COHERENS program has been designed such that it can be applied by users with a “low” or a “high” level of model experience. In the former case the program is run with default values of model setup variables and only a limited number of program variables (scalars and arrays) need to be defined by the user such as the location of the grid, bathymetry, open boundary and initial conditions. To facilitate its use for experienced modellers, the program is supplied in modular form. This allows to replace a module by a user-defined version without affecting the core of the program. It is clear that this practice is not without danger especially when the module is linked with other program units. Changing the default settings of switches and model parameters is easily performed but not always without risk. This is for example the case when a non-default scheme for turbulence or advection is selected or when one or more default parameter values for the biological module are changed.

For a user it is important to know which specific experience is required to run COHERENS for a particular type of application. There are two basic criteria: programming experience and scientific background. They are further discussed below.

Programming experience

A basic knowledge of FORTRAN 77 and UNIX is required. The shell scripts, used to run the test cases (see Chapter I-2), assume that the operating system works under the UNIX C Shell, but can easily be adapted to other Shell environments. As explained in Chapters I-2 and I-3 a series of FORTRAN source code files need to be created for a user application. Generic example files have been made available to facilitate this. In most cases the user only has to replace or add a number of assignment statements or to insert READ instructions for data input. Changing the source code of the program is a more difficult task and is in principle not recommended unless the user has a good experience with FORTRAN programming and numerical modelling. For a better understanding how the different modules are interconnected and constructed and how the program is written it is advised to read Chapters IV-1 (program structure and programming techniques) and Chapter V-1 (module descriptions). The replacement of a module by an alternative version is briefly discussed in Section IV-5.5.

Scientific background

The default settings allow to run the model without a specific knowledge of the program's scientific basis. Only a limited number of default values need to be changed for most applications. Exceptions are the selection of open boundary and initial conditions for which no default values can be provided. Detailed instructions are given in this documentation together with a large number of examples (see Part II). A more specific scientific background is required if for example the user intends to perform experiments with different turbulence or numerical schemes or with alternative model values for the biology, turbulence, optical module, ... It is then recommended to read first the appropriate chapters of Part III (Model Description).

1.5 Contents of the documentation

1.5.1 Structure

This documentation is composed of five parts:

- Part I is considered as an introductory manual. Instructions for preparing and running a pre-defined test case or a user application are given in Chapter I-2. Model setup with the aid of the generic code examples is explained in Chapter I-3. The graphical interface program and visualisation of model output with the freeware package FERRET are discussed in Chapter I-4.

- Ten test cases and two more realistic applications are presented in Part II which contains the following seven chapters: introduction (II-1), test cases for advection (II-2), test cases for turbulence (II-3), plume test cases (II-4), biological test cases (II-5), the NOMADS tracer experiments (II-6) and the simulation of a yearly cycle in the North Sea using the integrated model (II-7). Each test case is provided with a number of experiments testing the role of program switches, model parameters or different types of boundary conditions. A total number of 51 experiments can be performed by the user. The aim is to test the portability of the code, to illustrate how model setup is performed and to show different types of applications.

- The scientific aspects of COHERENS are fully explained in Part III (Model Description) where all model equations and numerical schemes are written out in detail and the scientific background is discussed. This part is further divided in five chapters: physics (III-1), biology and sediments (III-2), particle module and contaminants

(III-3), numerical methods (III-4) and a final chapter (III-5) with suggestions for improvement and future updates of COHERENS.

- Part IV is the User Manual covering all programming aspects of COHERENS, model setup, compiling and running the program, input/output files, ... The structure of COHERENS and technical features of the code are analysed in Chapter IV-1. The remaining chapters deal with the preparation of model input (IV-2), model output (IV-3), the I/O structure (IV-4) and additional issues such as compilation, error messages and splitting a simulation into several runs (IV-5).

- Part V is the Reference Manual. The purpose of each module is described in Chapter V-1 with the aid of a few flow charts. A listing of program variables with their FORTRAN name and purpose is presented in Chapter V-2. Bibliographic references are given at the end.

1.5.2 Suggestions for reading

The documentation is written, as much as possible, in a self-contained form. Appropriate links are made, where necessary, to chapters and sections where a specific topic is explained in more detail. First users whose prime intentions are to perform simulations with the model and have less interest in learning all the features of COHERENS, should proceed as follows:

1. Read Part I.
2. Select a few test cases from Part II to test the portability of the model. Have a look at the source code files used to set up the selected test cases.
3. Read those sections of Part III which are of interest for the intended application.
4. Create the source files for the application using the instructions given in Chapters I-2 and I-3. For more information about the model setup or the name of certain variables for model output consult the appropriate sections of Part IV and V.

For a full understanding of all the aspects of COHERENS it is obviously recommended to read the whole documentation.

The text is typesetted in LaTeX. The following numbering conventions are adopted:

- Chapters and pages are numbered within each Part.
- Sections are numbered in a two-number format (chapter-section).
- Subsections are numbered in a three-number format (chapter-section-subsection).
- Equations, figures and tables are numbered in a three-number format (part-chapter-equation/figure/table).

References to chapters, sections or subsections are given by their number preceded by a Roman number denoting the Part (e.g. Chapter II-3 for Chapter 3 of Part II).

Specific names and keywords are outlined in different LaTeX fonts:

- *slanted* style for the names of files
- **bold** type for UNIX command names, shell scripts, directory names and the names of test cases
- SANS SERIF style for the FORTRAN names of program variables and specific FORTRAN keywords
- `type-writer` style to emphasize specific instructions to be performed by the user or to outline text.

Chapter 2

Getting Started

The aim of this chapter is to provide the necessary information to install the program, to run one of the built-in test cases and to summarise the different steps for setting up a user-defined application. It is assumed that the operating system is either UNIX or LINUX, working under the C Shell.

This chapter is organised as follows:

- Installation instructions and a description of the program's file system are given in Section I-2.1.
- The different steps needed to run a test case are explained in Section I-2.2. For a detailed description of all test cases the reader is referred to Part II.
- A general outline explaining how to prepare and run a realistic application of the model, is presented in Section I-2.3. More detailed instructions and guidelines for model setup can be found in Chapter I-3. A complete description is given in the User Manual (Part IV).

2.1 Installing the program

In the following it is assumed that the file *release84.tar* is downloaded on the user's home directory. The current version of the program is Release 8.4. To retrieve the files use:

```
tar -xvf release84.tar
```

The following directories are created:

- **~/release_8.4/source** containing the general source code in the form of **.f* and **.inc* files and the *Makefile* needed to compile the program
- **~/release_8.4/utilities** containing the startup shell scripts **csetup** and **vsetup**, and where other utility programs may be installed by the user

- **~/release_8.4/tests** containing a series of subdirectories where the code is installed for running a specific test case and the subdirectory **/examples** with a series of example files further discussed in Chapter I-3 and the subdirectories **/nomads** and **/consp** for setting up the applications discussed in Chapters II-6 and II-7 and the subdirectory **/ptests** with a series of files containing values of output parameters for all test cases (see Section II-1.1 for details).

The source files located in the **source** directory are all listed in Tables 4.1.1 and 4.1.2 in the User Manual together with a description of their purpose. The test case subdirectories contain the following files:

- the shell scripts **Prepare** and **Run**
- the FORTRAN files *param.inc* and *defmod.f* [1] used to prepare model input
- the FORTRAN file *defout.f* where model output is defined
- the FORTRAN file *print.f* where output parameters specific for the test case are defined
- the parameter input file *defmod.par*
- the FORTRAN file *defavr.f* (in **/csnsp**) for time-averaged output and *defanal.f* (in **/river** and **/plume**) for harmonic analysis
- the file *defpar.f* (in **/nomads**) for particle output
- data files used for model input (in **/csnsp**)
- alternative source files (e.g. *metin.f* in **/csbio**)

The meaning of these files will be explained in this and the following two chapters. Instructions for preparing and running a test case or a user application are given in the next two sections. Before making any application the user must first ensure that the program is compiled with the appropriate command. On a UNIX machine this is performed as follows:

- Edit the *Makefile* in the **SOURCE** directory.
- Replace f77 in the line FC = f77 by the appropriate command for compiling FORTRAN programs.

More detailed instructions for implementing COHERENS on different computer platforms (including PCs) are discussed in Section II-1.2.

[1] In **/river** and **/plume** *defmod.f* is replaced by the two files *defmod_0.f* and *defmod_1.f*.

2.2 Preparing and running a test case

There are currently 10 test cases installed. Their names are listed in Table 1.2.1. Before running a test case the following 4 preparatory steps are to be made:

1. Change to a C shell (if necessary) by typing

   ```
   sh
   ```

2. Select a working directory, e.g. **/tmp/work** for running the program:

   ```
   cd /tmp/work
   ```

3. Define the environment variable COHERENS as the full pathname of the directory where the current version of the program is installed:

   ```
   setenv COHERENS ~/release_8.4
   ```

4. Execute the startup shell script **csetup**:

   ```
   $COHERENS/utilities/csetup test_name
   ```

   where **test_name** is the name of the test case (see Table 1.2.1 for a complete list) which is the same as the name of the corresponding subdirectory in **/tests**. Executing the shell script **csetup** links the **/source**, **/tests/test_name** and the **/utilities** directories to **SOURCE, PROJECT** and **UTIL** and copies all the necessary source files to the working directory.

5. Execute the shell script **Prepare**:

   ```
   Prepare arg
   ```

   The argument arg is an upper case letter denoting the type of experiment performed with the test case. Allowed values for each test case are given in Table 1.2.1. For a description of the experiments see Part II. The following actions are performed by **Prepare**:

   - The preprocessor **preproc** and the main program **coherens** are compiled with the **make** utility using the instructions given in *Makefile*.
   - The preprocessor program is run to generate all the input files for the main program (e.g. settings of switches and model parameters, model grid, initial and boundary conditions).

   The preprocessor writes a *log_file*. Its name is given by *prelog* followed by *arg*. If no run-time errors occurred, the last line of the *prelog*-file should read:

Table 1.2.1: Test case names and experiments (indicated by the argument in the **Prepare** and **Run** commands).

Test case	Experiment							
cones	A	B	C	D				
front	A	B	C	D				
seich	A	B	C	D				
fredy	A	B	C	D				
pycno	A	B	C	D	E	F	G	
csnsp	A	B	C	D	E	F	G	H
river	A	B	C	D				
plume	A	B	C	D	E	F	G	
batch	A	B	C					
csbio	A	B	C	D	E	F		

```
Preprocessor terminated
```

If no problems occurred during execution of the preceding steps, the program is run using the simple command

```
Run arg
```

where arg must obviously the same as the argument used with **Prepare**. The main program writes a new *log*-file whose name is given by *runlog* followed by *arg*. If no run-time errors occurred, the last line should read:

```
Main program terminated
```

To check whether the program ran successfully, a file with values for a series of output parameters is written by the main program. Its name is given by the name of the test case, followed by the experiment's name (arg) and followed by the suffix *.tst* (e.g. *conesA.tst* for experiment A of the test case **cones**). The file is sufficiently documented so that the parameter values can be compared with the corresponding table in Part II where all test cases are discussed in detail.

Model results from a test case simulation can be visualised on different graphical platforms with the aid of the interface program **netcdfint**. This is further explained in Chapter I-4.

2.3 Preparing and running a user application

Setting up and running a user application proceeds along a series of steps similar to the ones explained in the previous section for a test case simulation. The main difference is obviously that a series of files needs to be created by the user whereas these files are already available for the test runs. The proceedings are explained below in a step-wise form.

1. Create a new directory, say **/mytest** in the **/tests** directory:

   ```
   cd ~/release_8.4/tests
   mkdir mytest
   ```

   A series of files, described in steps 2 to 5 must or may optionally be created by the user within this subdirectory. Most of these files must be written as FORTRAN source code files. Generic examples are found in **/examples**.

2. The file *defmod.f* must be created. This a FORTRAN source code file and is composed of an ensemble of subroutines where model parameters, grid arrays, open boundary conditions and data, initial conditions together with meteorological and/or wave input data are defined or read from input files supplied by the user.

3. A file *param.inc* must be created. This has the form of a PARAMETER statement which is used in the program for the dimensioning of arrays.

4. A parameter file must be supplied containing the TITLE of the simulation and a set of parameters for time series output, harmonic analysis, time-averaged output and particle output. The default name of this file is *defmod.par*. Its contents are read by the preprocessor **preproc** from standard input.

5. The following FORTRAN files can optionally be created depending on the type of output required by the user:

   - *defout.f* : ensemble of subroutines where the quantities for time series output are defined. This is only necessary if the switch IOUTS is set to 1.
   - *defanal.f* : ensemble of subroutines where the quantities for which harmonic analysis will be applied, are defined. This is only necessary if the switch IOUTT is set to 1 or 2.
   - *defavr.f* : ensemble of subroutines where the quantities for time-averaged output are defined. This is only necessary if the switch IOUTF is set to 1.
   - *defpar.f* : a subroutine where the positions are stored of the particles selected for output. This is only necessary if the switch IOUTP is set to 1.
   - *print.f* : contains the subroutine PRINT where any other user-defined output is defined.

   If these files are not supplied by the user, the program automatically compiles with the default versions given in the **SOURCE** directory. Note that the default files are only provided to avoid compiling errors and do not provide any significant output.

6. Select a working directory, e.g. **/tmp/work** for running the program:

   ```
   cd /tmp/work
   ```

7. Define the environment variable COHERENS as the full pathname of the directory where the current version of the program is installed, e.g.

   ```
   setenv COHERENS ~/release_8.4
   ```

8. Execute the startup shell script **csetup**:

   ```
   $COHERENS/utilities/csetup test_name
   ```

   where **test_name** is the name of the subdirectory in **/tests** (e.g. **mytest**) where the user files mentioned in steps 2 to 5 are located. Executing the shell script **csetup** links the **source**, **/tests/mytest** and the **utilities** directories to **SOURCE**, **PROJECT** and **UTIL** and copies all the necessary source files to the working directory.

9. Compile the preprocessor **preproc** and the main program **coherens** with the **make** utility:

   ```
   make
   ```

10. Run the preprocessor program:

    ```
    preproc < param_file > log_file
    ```

    where *param_file* is the name of the parameter file described in step 4 and *log_file* a file where "run-time" information is written by the program. When the default name *defmod.par* is used, input redirection can be omitted. The "log"-file is of no use for the program but can be helpfull to detect errors during execution of the program. If no run-time errors occur, the last line of the "log-file" should read:

    ```
    Preprocessor terminated
    ```

    The preprocessor generates a series of files (settings of switches and model parameters, model grid, initial and boundary conditions, ...) which serve as input for the main program **coherens**.

11. Run the main program

    ```
    coherens < input_file > log_file
    ```

where the prefix of *input_file* is the TITLE of the run (6 characters) defined in the parameter file mentioned in step 4 and the suffix is *.conA*. If for example TITLE equals mytest, the name of the input file is *mytest.conA*. The "log"-file only contains "run-time" information and is not used by the program. In the absence of run-time errors, the last line should read:

```
Main program terminated
```

12. A series of output files is created by the program which can be used for plotting or to restart the program. The number of files, their contents and formats depends on the values of the output specifiers defined by the user (see steps 4 and 5).

As explained in Chapter I-3, most of the files mentioned in steps 2–5 can be constructed from a generic file in the **/examples** subdirectory. All aspects of model setup are fully discussed in the following chapters of the User Manual:

- Chapter IV-2: preparing model input with the files *param.inc* and *defmod.f*
- Chapter IV-3: preparing model output with the files *defmod.par*, *defout.f*, *defanal.f*, *defavr.f*, *defpar.f*, *print.f*
- Chapter IV-4: input/output structure of the program including a description of all files written by the program and the modules for reading model output
- Chapter IV-5: startup shell scripts, compilation, error and warning messages, splitting a simulation into several parts, making changes to the source code.

Chapter 3

User Instructions

As already explained in Section I-2.3, model input and model output are defined through a series of files, mostly written in the form of FORTRAN source code. The aim of this chapter is to explain in a general way how these files are created. No prior knowledge is assumed concerning the model equations and numerical schemes, which are fully described in Part III of the documentation. No attempt is made to explain in detail each model switch, parameter or array as a complete description of the model setup is reserved for Part IV.

The chapter is organised as follows:

- Instructions for making up the model setup files are given in Section I-3.1.
- General guidelines and useful hints for designing a user application are described in Section I-3.2.

3.1 Creating the files for model setup

The files can be divided into two categories:

- files which prepare model input: *param.inc*, *defmod.f*
- files which prepare model output: *defmod.par*, *defout.f*, *defanal.f*, *defavr.f*, *defpar.f*, *print.f*.

The file *defmod.par* contains a series of parameters read by the preprocessor **preproc**. All other files are to be written in FORTRAN 77. Generic examples are found in the **/examples** subdirectory except for *defmod.par* and *print.f*. The user should note that, although these generic files are written in FORTRAN, they cannot be compiled without specifying the user-defined values indicated by a "?", as discussed below.

3.1.1 *param.inc*

The file may be generated as follows:

1. Copy the generic file from the **/examples** to the **PROJECT** directory, e.g.:

   ```
   cp $COHERENS/tests/examples/param.inc $COHERENS/tests/mytest/param.inc
   ```

2. Replace the "?" in the assignments of NC, NR, NZ by the appropriate values. These parameters represent the resolution of the grid and denote respectively the number of grid cells in the X-, Y- and vertical direction.

3. The other parameters are given by their default values. The user should change their values where necessary.

The meaning of the other parameters which are also documented internally in the form of comment lines, can be summarised as follows:

- NOBU and NOBV are the number of open boundaries in respectively the X- and Y-direction. A zero value indicates that all boundaries in the corresponding direction are solid land boundaries.

- NVPROF represents the number of vertical profile arrays at open boundaries (see Section IV-2.5). Its value must be non-zero if either NOBU or NOBV is non-zero.

- NCON and NCONTO are the number of frequencies for respectively the tidal forcing and harmonic analysis. The latter has a minimum value of 1 but is only relevant for the program if harmonic analysis is enabled with the switch IOUTT.

- NCONC and MAXNOP are the number of contaminants in the Eulerian contaminant module and the number of particles in the Lagrangian particle module. Their values are only of importance if these modules are enabled with the switches IOPTC or IOPTP.

- NOUTMAX, NANALMAX and NAVRMAX must be greater than or equal to the number of 2-D or 3-D fields defined in respectively the files *defout.f*, *defanal.f* and *defavr.f*. They have a minimum value of 1. Their values are important only when the corresponding modules are activated using the switches IOUTS, IOUTT or IOUTF.

- MAXOUT and MAXAVR must be greater than or equal to the number of output files for respectively time series and time-averaged output as defined in *defmod.par* and have a minimum value of 1. Their values are only relevant if time series or time-averaged output is enabled with the switch IOUTS or IOUTF.

It is recommended not to change the default values of parameters which have no relevance for the simulation. More detailed information concerning *param.inc* can be found in Section IV-2.1. Examples of *param.inc* can be found in all the test case subdirectories.

3.1.2 *defmod.f*

In this file all model parameters and arrays, needed to initialise the program, are defined (e.g. switches, date/time parameters, counters, model parameters for a specific module, bathymetry, open boundary and initial conditions, surface forcing, ...). The procedure to generate this file, is the following:

1. Copy the generic file from the **/examples** to the **PROJECT** directory, e.g.:

   ```
   cp $COHERENS/tests/examples/defmod.f $COHERENS/tests/mytest/defmod.f
   ```

2. The file is composed of 7 FORTRAN subroutines, which are described below. Each routine contains a series of assignment statements for FORTRAN scalars or array elements either with a default value or an undetermined value, denoted by a "?" on the right hand side. The unit of each dimensional variable is indicated in a comment line above the statement line. If the user changes a default value, this new value becomes effective throughout the program. No changes have to be made if the user is satisfied with the default. In the case of an undetermined variable for which no default is available, the "?" must be replaced by an appropriate value. This is for example the case for the start and end date of the simulation and the time step. In many cases the assignments are embedded within an IF- or ELSEIF-block or within a DO-block. It is clear that when the IF or ELSEIF test evaluates as .FALSE. (depending on the value of a switch or counter) or the upper limit of a DO-loop is smaller than the lower one, the value of a scalar or array element variable becomes unimportant. In that case it is sufficient to replace the "?" by a zero value.

3. To set up the program for a realistic application additional code has to inserted. This is usually the case when arrays are initialised from a data input file. Opening files for data input is discussed in Section IV-4.1.

4. It is recommended to check the file for any remaining "?" before compiling the program.

5. The preprocessor and main program check whether invalid or inappropriate values are selected for a series of parameters and arrays. This occurs most often for the arrays defined in subroutine DEFGRID. If this is the case, an error or warning message is issued. A further discussion is given in Section IV-5.3.

The subroutines within *defmod.f* are called by the preprocessor program **preproc** and have the following purpose:

DEFCON
: Defines all values for model switches, date/time parameters, counters, file parameters and specific parameters for different modules (e.g. optics, turbulence, biology, ...).

DEFGRID
: Defines the location of the grid, the bathymetry, a series of pointer arrays, the grid indices of the open boundary points and (possibly) the bottom roughness length.

DEFOB2D
: Defines the type of open boundary input for the depth-integrated current, the tidal frequencies, and provides the amplitudes and phases for time-independent open boundary forcing.

DEFOB3D
: Defines the type of open boundary conditions for all 3-D quantities (salinity, temperature, currents, biological state variables, ...) and all time-independent input data at the open boundaries for 3-D quantities and the particle module.

DEFICS
: Defines the initial conditions.

WRFCDAT
: Provides all the meteorological and/or wave input data.

WROBDAT
: Provides all time-dependent open boundary input.

A few useful comments can be given.

- Although it is not explicitly required, several source lines can be deleted from the code of *defmod.f* making the code more transparent and easier to read. This procedure has been followed in the setup of the test cases. Code reductions can be performed in the following cases:

  1. When the test condition within an IF- or ELSEIF-block evaluates as .FALSE., the statements within the block can be omitted. The test mostly depends on the known value of a switch or a counter.
  2. When the upper bound in a DO-loop is smaller than the lower one, the loop is not executed and can be deleted. The end value is always given by a parameter already defined in *param.inc* and therefore known to the program. It is remarked that some FORTRAN compilers do execute at least one iteration of a DO-loop even when the upper limit is smaller than the lower one. This is mostly provided in the form of a separate compiler option and is non-standard FORTRAN. It may induce serious problems when running the program and must be disabled by the user. For more information about compiling the program see Section IV-5.2.
  3. Subroutine WRFCDAT is only called by the program if one of the switches IOPTM or IOPTW is larger than 1 in which case all code within the routine can be removed.

4. Subroutine WROBDAT is only called by the program if one of the counters IC2BC, ICHBC, ICBBC, ICCBC or ICPBC is non-zero.
5. It is obvious that the SUBROUTINE and END statements can never be removed even when the routine does not contain any other statement lines.
6. It is highly recommended not to remove any of the INCLUDE statements. These INCLUDE files contain the COMMON and PARAMETER statements which allow access to the model variables in different program units.

- Selecting proper values for setup variables in *defmod.f* depends on the type of model application and may require some knowledge about the structure of the program. A further discussion and some general guidelines are given in Section I-3.2 below. For a detailed description of the model setup in *defmod.f* the user is referred to Chapter IV-2. All setup variables with a short description of their purpose are listed in a series of tables at the end of that chapter.

- Example *defmod.f* files can be found in all test case subdirectories. The "setup" tables in Part II contain a listing for each test case of all variables except those whose default values have not been changed.

3.1.3 *defmod.par*

This is the default name of a parameter file read by the preprocessor from standard input. No generic example is available. The aim is to provide the following information to the program:

- the TITLE of the run (given in the form of a CHARACTER variable on the first line)
- the number of output files (except those which are automatically written by the program)
- the number of output data fields for each different type of output (time series, harmonic, time-averaged or particle trajectories) as defined in the user files *defout.f*, *defanal.f*, *defavr.f*, *defpar.f* (see Section I-3.1.4 below)
- the format of the output files (ASCII or binary)
- the specification and resolution of the grid area and time steps for which output is requested
- the time periods for harmonic analysis and time averaging.

The number of data lines depend on the values of the output switches IOUTS, IOUTT, IOUTF and IOUTP used to select respectively time series, harmonic, time-averaged and particle output. Examples of *defmod.par* files can be found in all test case subdirectories.

The contents of *defmod.par* and a description of the output specifiers are discussed in Section IV-3.1.

As an example the file *defmod.parA* with the output specifiers for the **plume** experiment A is listed below.

```
Record  1:plumeA
Record  2:4
Record  3:1 3 1 1
Record  4:U - surface plots
Record  5:1 120 1
Record  6:1 40 1
Record  7:20 20  1
Record  8:0  8640 360
Record  9:U - transect 1
Record 10:30 30 1
Record 11:1 40 1
Record 12:1 20 1
Record 13:0 8640 360
Record 14:U - transect 2
Record 15:1 120 1
Record 16:5 5 1
Record 17:1 20 1
Record 18:0 8640 360
Record 19:U - times series
Record 20:30 30 1
Record 21:1 1 1
Record 22:1 1 1
Record 23:0 8640 12
Record 24:--- harmonic analysis
Record 25:1 1 1 1
Record 26:U
Record 27:1 120 1
Record 28:1 40 1
Record 29:1 20 1
Record 30:0 8640 1440
```

The records provide the following information for the program.

Record 1: The title of the run is plumeA.

Record 2: Number of (2-D and/or 3-D) time series output files equals 4.

Record 3: The number of 2-D vectors, 2-D scalars, 3-D vectors and 3-D scalars for time series output, defined in *defout.f*, is given by respectively 1, 3, 1, and 1.

Record 4: The first data file will be written in 'U'-format (binary). Alternative is 'A' which means ASCII-formatted data.

Record 5: Column data for the first output file will be written with a IFIRST value of 1, ILAST value of 120 and ISTRIDE value of 1.

Record 6: Row data for the first output file will be written with a JFIRST value of 1, JLAST value of 40 and JSTRIDE value of 1.

Record 7: Level data for the first output file will be written with a KFIRST value of 20, KLAST value of 20 and KSTRIDE value of 1.

Record 8: Data for the first output file will be written in time with a LFIRST value of 0, LLAST value of 8640 and LSTRIDE value of 360.

Records 9–13: The same meaning as Records 4–8 now for the second time series file.

Records 14–18: The same meaning as Records 4–8 now for the third time series file.

Records 19–23: The same meaning as Records 4–8 now for the fourth time series file.

Record 24: Header line.

Record 25: The number of 2-D vectors, 2-D scalars, 3-D vectors and 3-D scalars for harmonic analysis, defined in *defanal.f*, is given by 1.

Record 26: The harmonic data file will be written in 'U'-format (binary).

Record 27: Column data will be written to the harmonic data files with a IFIRST value of 1, ILAST value of 120 and ISTRIDE value of 1.

Record 28: Row data will be written to the harmonic data files with a JFIRST value of 1, JLAST value of 40 and JSTRIDE value of 1.

Record 29: Level data will be written to the harmonic data files with a KFIRST value of 1, KLAST value of 20 and KSTRIDE value of 1.

Record 30: Harmonic analysis starts at time index 0 and ends at time index 8640. The harmonic period is given by 1440 (2-D) time steps.

The **plume** test case uses 120 cells in the X-direction, 40 cells in the Y-direction, 20 levels in the vertical and a time step of 30 s so that the following type of output is defined:

- First time series data file: output values at all surface grid points starting at the initial time, ending after 8640 time steps (3 days) and at intervals of 360 time steps (3 hours).
- Second time series data file: output values at all grid points along the vertical transect I=30 parallel to the Y-axis using the same output times.

- Third time series data file: output values at all grid points along the vertical transect J=5 parallel to the X-axis using the same output times.
- Fourth time series data file: output values at the grid point with indices (30,1,1) starting at the initial time, ending after 3 days and at intervals of 12 time steps (6 minutes).
- Harmonic data files: output values at all grid points. Harmonic aanalysis starts at the initial time, ends after 3 days and has a period of 1440 time steps (12 hours). Output will then be written for a total number of 6 periods.

3.1.4 Files for setting up model output

A series of five files can optionally be created where the user defines the data fields for model output: *defout.f*, *defanal.f*, *defavr.f*, *defpar.f*, *print.f*. For the first four, generic example files are provided in the **/examples** subdirectory. Since no default values can be provided, the user is responsible for filling in the code. In most common applications however, the user only needs to make a few assignment statements and to insert the necessary INCLUDE statements.

- *defout.f*

  This file is intended for output at selected time intervals (time series output) which is enabled with the switch IOUTS. This is the most common type of output and is switched on by default. The file is composed of five subroutines. The following procedure is recommended:

  1. Copy the generic file from the **/examples** to the **PROJECT** directory, e.g.:

     ```
     cp $COHERENS/tests/examples/defout.f $COHERENS/tests/mytest/defout.f
     ```

  2. Define the 3-D and 2-D output data fields and insert the necessary INCLUDE statements in the first two subroutines (OUTPT3D and OUTPT2D).
  3. Define a series of string arrays with the names, descriptions and units of the output data in the last two subroutines (OUT3DVAR and OUT2DVAR).
  4. Remove the unnecessary code including the lines containing a "?".

  Examples can be found in all test case subdirectories.

- *defanal.f*

  This is used to define the fields for harmonic analysis. A mathematical description of the method is given in Section III-1.7. Harmonic analysis is selected with the switch IOUTT and is disabled by default. Note that, if IOUTT = 2, the tidal ellipse parameters are determined in addition. The file contains five routines. The procedure is similar to the previous case.

1. Copy the generic file from the **/examples** to the **PROJECT** directory, e.g.:

   ```
   cp $COHERENS/tests/examples/defanal.f $COHERENS/tests/mytest/defanal.f
   ```

2. Define the 3-D and 2-D output data fields and insert the necessary INCLUDE statements in the first two subroutines (ANALPT3D and ANALPT2D). Note that the current must be the first output field when the calculation of tidal ellipse parameters is requested.
3. Define a series of string arrays with the names, descriptions, units of the data output and with the harmonic frequencies in the last three routines (ANAL3DVAR, ANAL2DVAR, ANALFR).
4. Remove the unnecessary code including the lines containing a "?".

Examples can be found in the **/river** and **/plume** directories.

- *defavr.f*

This file is used to define time-averaged output. This is selected with the switch IOUTF and disabled by default. The file contains four subroutines and is constructed similar to the four steps described above for *defout.f*. A example can be found in the **/csnsp** directory.

- *defpar.f*

This file contains one routine where the coordinates of the particles, determined from the Lagrangian particle module, are specified for output. The procedure is the following:

1. Copy the generic file from the **/examples** to the **PROJECT** directory, e.g.:

   ```
   cp $COHERENS/tests/examples/defpar.f $COHERENS/tests/mytest/defpar.f
   ```

2. Define the coordinates of the particles, whose trajectories are to be plotted.
3. Remove the lines containing a "?".

- *print.f*

This routine is foreseen for defining other types of output. The user is completely responsible for its contents.

Note that all the preceding files, except *print.f*, only need to be created if the corresponding output switch is set with a non-zero value. If *print.f* is not supplied by the user, the program automatically compiles with an empty default version in the **SOURCE** directory. For a more complete description of the setup files for model output see Chapter IV-3.

3.2 General guidelines for *defmod.f*

Of all the setup files discussed in the previous section the file *defmod.f* is the most important one and probably the most difficult to construct. The aim of this section is therefore to provide some general guidelines for setting up a model application with *defmod.f*. The analysis, presented here, is not complete and should only be considered as a first introduction. A complete description of the file *defmod.f* and of all model input parameters and arrays is given in Chapter IV-2 of the User Manual.

The routines of *defmod.f* are organised in what one may consider the most logical sequence: switches, start/end of the simulation, time step, counters, model parameters (divided in different subcategories), bathymetric and grid arrays, type of open boundary conditions, initial conditions, input data for the surface forcing, open boundary input data. Each item is discussed below in a separate subsection. The name of the corresponding routine in *defmod.f* is given in each section heading.

3.2.1 Switches (DEFCON)

The intention of the switches is to enable or disable a particular module or to select a particular process. A zero value in the former case means that the module is disabled. Negative values are not allowed. Some switches have only off/on values (0/1) while others may have additional (positive) values (see Section IV-2.2.1). The allowed values of all switches are indicated in the generic *defmod.f* file.

- A 3-D (default) or a 1-D application is selected with IGRDIM.
- The model grid is either Cartesian (default) or spherical selected with IGTRH.
- The program is composed of a core physical module, a biological module, a sediment module, an Eulerian contaminant module and a Lagrangian particle module. The physics links the other parts of the program and cannot be disabled while the others are selected by respectively IOPTB, IOPTS, IOPTC and IOPTP.
- Several numerical schemes are available to evaluate the advection and diffusion terms in the model equations: IADVC for the advection of momentum, IADVS for the advection of all scalar quantities, IODIF for horizontal diffusion and IOPTK to select the general type of scheme for vertical diffusion. In the first three cases the corresponding terms in the transport equations are disabled when the switch is set to zero.

- The physical part contains a series of circulation modules and two modules for salinity and temperature. The momentum equations are solved using the mode splitting technique which means that they are split into a set of equations for the depth-integrated current and surface elevation (2-D mode) and the full 3-D momentum equations (3-D mode). The two sets are selected using respectively IOPT2 and IOPT3. The 2-D and 3-D current modules are switched on by default and it is recommended not to change these values[1]. The salinity and temperature modules are enabled by respectively IOPTSA and IOPTHE. They are disabled by default. Note that the optical module is included in the update of the temperature field if IOPTHE = 2.

- The form of the equation of state is selected with IOPTD. It is recommended not to set this switch to 0 except when IOPTSA = IOPTHE = 0 in which case all stratification effects are removed in the momentum and turbulence modules.

- Several switches are available to select the surface and bottom boundary conditions for currents, temperature and salinity. The bottom stress formulation is selected with IBSTR. (The default value of 1 corresponds to the usual quadratic law and should normally not be changed). Several formulations for the surface drag coefficient can be selected with IDRAG. The meaning of ITDIF is explained in Section III-1.6.4. Keeping the default value is mostly sufficient. Surface fluxes of momentum, heat and salinity can be disabled with IOPTM which also selects the type of meteorological data for the surface forcing.

- The program contains a wave-current interaction module which includes surface wave effects in the calculation of the bottom stress (see Section III-1.6.6). This is enabled with IOPTW which also determines the type of wave data supplied to the program.

- Two switches are available for the biological module. Resuspension, deposition and bottom losses of microplankton and detritus are enabled with IFLUFF. The switch IADVWB is usually only relevant for 1-D applications where it sets the type of advection scheme for vertical sinking.

- Four switches are available to select the type of model output: IOUTS (time series), IOUTT (harmonic analysis), IOUTF (time-averaged output) and IOUTP (particle trajectories).

- A whole series of additional switches are available to select different types of turbulence schemes. For a better understanding it is advised to read first Section III-1.2. Keeping the default values can often be considered as sufficient.

[1]IOPT2 is automatically set to 0 only if the program is used in a 1-D model application (see Section IV-2.9).

3.2.2 Time parameters and counters (DEFCON)

- The period of the simulation is determined by fixing the date and the year at the start and the end of the simulation, determined by the four parameters IBDATE, IBYEAR, IEDATE and IEYEAR. Note that the date is given in the form month-day-hour-minute in FORTRAN I8.8 format.

- The program uses several time steps. The smallest one is DELT which is the time step for the 2-D mode in the circulation modules (see Section IV-2.2.3). All other time steps in the program are evaluated by a time counter multiplied by DELT.

- The counter IC3D determines the time step in all 3-D transport equations and the particle module (see Section IV-2.2.4).

- All other counters determine the time step for the inclusion of different types of input: meteorological (ICMET); surface waves (ICWAV); residuals (including river discharges), amplitudes and phases for the depth-integrated current and/or sea surface elevation (IC2BC); open boundary input of salinity, temperature and horizontal current (ICHBC); open boundary input of biological and inorganic sediment concentrations (ICBBC); open boundary input of suspended matter for the Lagrangian module (ICPBC); open boundary input of contaminant concentrations (ICCBC). A zero value for one of these counters implies that no input is provided or that the input remains fixed during the entire simulation.

3.2.3 Model parameters (DEFCON)

- A number of parameters determine the form of the model output. The parameter HEADERS allows the program to put additional header information in all ASCII output files. A series of parameters inform the program to write certain types of output data files either in ASCII ('A') or in binary ('U') format. For large applications it is recommended to use the latter form to save disk space. MAXFILES is a machine-dependent parameter representing the maximum number of files which can be connected to the program and OUTTIT allows the names of the output data files to be changed.

- DLAREF and DLOREF are a reference latitude and longitude (decimal degrees) of the domain in the case of a Cartesian grid (IGTRH $= 0$) in which case they are used to determine the Coriolis frequency and solar irradiance. In the case of a spherical grid (IGTRH $= 1$) longitude and latitude are spatially dependent and are obtained from the grid coordinates.

- DEPUN is the (uniform) mean water depth in the case of a 1-D application (IGRDIM $=$ 1).

- The reference values R0REF (density), SREF (salinity) and TREF (temperature) should take values considered as typical for the simulation. SREF and TREF are used to initialise the salinity and temperature fields by default.
- The expansion coefficients are needed in the linear equation of state (IOPTD $= 1$) and for the evaluation of the buoyancy frequency in the turbulence module. They are evaluated by the program as a function of space and time if IOPTD $= 2$.
- The optical parameters are used in the optical module (see Section III-1.4).
- VISMOL and DIFMOL are either the uniform (IOPTK $= 0$) or the background values (IOPTK $= 1$ or 2) of the vertical diffusion coefficients for momentum and scalars.
- HEDVUN and HEDDUN are the uniform horizontal diffusion coefficients for momentum and scalars if IODIF $= 1$. CM0 and CS0 are parameters used in the Smagorinsky (1963) formulation if IODIF $= 2$ (see Section III-1.3).
- CDLIN is the linear bottom friction coefficient used only when IBSTR $= 2$ (linear bottom friction law).
- CDZ0UN represents the bottom roughness length used only in the case of a quadratic bottom stress law (IBSTR $= 1$) and when the same bed type is used at all grid points.
- FSUN and GSUN are the X- and Y-component of the surface stress used only when IOPTM $= 1$ (uniform surface stress).
- HSUN and TWUN are the significant wave height and period used in the wave-current interaction module if IOPTW $= 1$.
- Different turbulence schemes are implemented in the program giving a large number of adjustable parameters. Before changing the defaults, set up by the program, the user should consult first Section III-1.2 where all turbulence schemes are discussed in detail.
- The parameters of the microplankton and sediment modules are discussed in Chapter III-2 and presented schematically in Table 4.2.3.
- The parameters STLSOL and STSSOL are the masses of the particles entering at river or open sea boundaries, and are only needed in the particle module in the case that input is supplied at the open boundaries. HEDPUN is the horizontal diffusion coefficient used only in the particle module.

3.2.4 Grid arrays (DEFGRID)

- GX0 and GY0 are the X- and Y-coordinates of the cell corners expressed either in m (Cartesian grid) or in decimal degrees (spherical grid).

- GZ0 are the σ-coordinates of the vertical nodes of the model grid (starting from the bottom up to the surface). A uniform vertical grid is selected by default.

- DEP is an horizontal array representing the mean water depth at the centres of the grid cells.

- NWD, NPIX and NPIY are horizontal pointer arrays. The default values of NWD should not be changed. The arrays NPIX and NPIY inform the program whether a cell face perpendicular to the X- or Y-direction is dry (0), an internal wet boundary (1), an open sea boundary (2) or a river boundary (3). For more details see Section IV-2.3.

- The arrays IOBU, JOBU, IOBV and JOBV are used to locate the open boundaries of the domain. For more details see Section IV-2.3.

- CDZ0 is an horizontal array representing the bottom roughness length (at the cell centres) used to determine the quadratic bottom friction coefficient (see Section III-1.6.2).

3.2.5 Type of open boundary conditions (DEFOB2D, DEFOB3D)

The program offers the choice between different kinds of open boundary conditions. Radiation conditions, based upon the theory of Riemann characteristics, are used for the 2-D mode. In the case of a 3-D quantity either a zero gradient condition or the specification of a value external to the grid can be selected. Several arrays have to be defined by the user to inform the program which type of condition is used at each open boundary. Since no default values are available in most cases, it is strongly advised to read first Section III-1.6.3 where the general theory and the different forms are discussed, and Sections III-4.3.11b, Sections IV-2.4 and IV-2.5 where the implementation of the theory in the program is explained.

- The arrays ITYPOBU and ITYPOBV determine the type of condition for the depth-integrated current and/or sea surface elevation.

- SIGMA and PHASE0 are the frequencies and initial phases of the tidal forcing.

- If harmonic analysis is enabled with the switch IOUTT, the frequencies for the analysis must be supplied using the array ANALSIG.

- The open boundary data for the 3-D quantities are given in the form of a series of vertical arrays. The arrays IVPOBU and IVPOBV inform the program which array is applied at which boundary.

- The arrays LSTOBU and LSTOBV are used in the Lagrangian module for assigning labels to the particles which are created by the input of suspended material at the open boundaries.

3.2.6 Initial conditions (DEFICS)

The following arrays can be initialised:

- physics: salinity and temperature. They are set to their reference values SREF and TREF by default.
- biology: initial concentrations of all state variables (microplankton carbon and nitrogen, detrital carbon and nitrogen, dissolved ammonium, nitrate and oxygen) and mesozooplankton grazing pressures
- sediment module: inorganic sediment concentrations
- contaminant concentrations
- particle module: initial particle positions (6 arrays), particle labels and masses.

Initial conditions are further discussed in Section IV-2.6.

3.2.7 Surface forcing (WRFCDAT)

There are two kinds of surface forcing data: meteorological data for the surface fluxes of momentum, heat, salinity and surface solar irradiance, and wave data for the wave-current interaction module. Note that data input is only enabled if the corresponding switches IOPTM and IOPTW are larger than 1. The intervals for new data input are given by the values of the counters ICMET and ICWAV. The data can be supplied in different forms. For more details see Section IV-2.7.

The source code of subroutine WRFCDAT serves two purposes. Firstly, the forcing data are supplied by the user for each time step. A common practice is to read the data from an input data file which requires a few extra code lines. Secondly, the data are written by the preprocessor to the appropriate data files at each time step in the format, required by the program. This is performed by the WRITE statements for scalar data or by calling the program routine WRARR for array data. It is therefore recommended not to change these statements and to remove them only if they form part of an IF- or ELSEIF-block not executed by the program.

3.2.8 Open boundary data (DEFOB2D, DEFOB3D, WROBDAT)

There are three different kinds of open boundary data. Those for the depth-integrated current and/or surface elevation consist of two arrays AMPOBU, AMPOBV representing the amplitudes (including residuals and possibly river discharge values) and two arrays PHAOBU and PHAOBV representing the phases of tidal forcing. Boundary data related to 3-D quantities have the form of (mostly) a two-dimensional array. The name of these arrays is given by the FORTRAN name of the quantity followed by the letters "VP" (for example "TVP" for temperature). The meaning of the arrays is further explained in Sections IV-2.4

and IV-2.5. The third type of open boundary data are those for the particle module given by an ambient concentration at open sea boundaries or by a river discharge of suspended matter at river boundaries. The following arrays can be defined:

- 2-D mode: AMPOBU, PHAOBU, AMPOBV, PHAOBV
- 3-D physics: arrays for salinity, temperature and horizontal current
- biology: arrays for microplankton carbon and nitrogen, detrital carbon and nitrogen, ammonium and nitrate
- sediments: one array for inorganic sediment concentrations
- contaminants: a 3-D array representing all contaminant concentrations
- particle module: the arrays QSTOBU and QSTOBV.

Defining 3-D open boundary data is often a difficult task in view of the absence of reliable data, either obtained from observations or from a larger area model. The program therefore allows use of a simpler zero gradient condition which does not require any data. This is also the default and denoted by the numbers “-99.0” and “-100.0” in the generic source code.

The time interval for data input is selected with the counters IC2BC, ICHBC, ICBBC, ICCBC and ICPBC. If a counter is zero, the corresponding data are fixed in time and defined in subroutine DEFOB2D (2-D mode) and DEFOB3D (all other arrays). When a counter is non-zero, the corresponding data are defined and written to the appropriate data files at the selected time interval in subroutine WROBDAT.

Similar to WRFCDAT (see Section I-3.2.7 above) the source code in subroutine WROBDAT consists of sections where the data are defined (eventually by reading from a data input file) and sections where the data are written to the appropriate files in the format required by the program. This is performed by calling the program routine WRARR for array data. It is therefore recommended not to change these statements and to remove them only if they form part of an IF- or ELSEIF-block not executed by the program.

Chapter 4

Graphics Interface

The aim of this chapter is to:

- summarise model output from COHERENS
- introduce the netCDF file format and the graphical software package FERRET
- explain the use of the interface program which converts model output into files in netCDF format
- illustrate how model output in netCDF format can be visualised with FERRET

4.1 Introduction

The ability to view the results from COHERENS, and to subsequently manipulate the output data, is an important requirement of the model system. Many methods of doing this are available, since there are as many plotting packages around as there are models. The aim is to provide access to the model data in an internationally recognised format, allowing ease of transfer from one platform to another, and access to a wide range of software for data display/manipulation.

The data format selected for standardisation of COHERENS model data is netCDF. This is widely accepted in the meteorological/oceanographic community as a standard data format, both for observational data and model data, and many software packages support a netCDF interface. NetCDF is freely available via anonymous ftp (direct or web-based).

In selecting a data visualisation and display package for COHERENS, several factors have been taken into consideration: proprietary use, ease of use, user support, output products. Many of the commercial packages, e.g. PV-Wave, IDL, Matlab, support a netCDF interface, i.e. data in netCDF format can be input to the package and the package "knows" what the data looks like. Whilst commercial packages are widely used in the marine science community, it is apparent that no single package is preferred to the others. It therefore makes more sense to recommend the use of software which is freely available and which can easily be obtained. It might be argued that commercial software is a better option in so much as it is (usually) well documented, well maintained and with regular upgrades, and covers a wide range of general use. Freely available software is often the product of small groups, designed for a specific purpose, poorly documented, and often containing bugs. However, there are exceptions to this. One of these is the visualisation package FERRET, produced and made available by PMEL in the United States. This package has been constructed over many years to visualise four-dimensional (three space and time) meteorological and oceanographic data, is freely available over the WWW, has good documentation, a user group and help facilities. It primarily expects datasets to be input in netCDF form but also allows data to be input in ASCII or binary form. FERRET is a recommended package for the display of COHERENS data. One advantage for COHERENS users (apart from being simple to use) is that the package is primarily oriented towards "real world" datasets and installs with global coastlines. As it is envisaged that most COHERENS users will develop real world applications, a visualisation package which supports this makes model development much easier. However it is a simple matter within FERRET to display data in Cartesian coordinates.

COHERENS provides the user with a FORTRAN program (called **netcdfint**) to convert the model generated datasets (in binary or ASCII form) into netCDF format. Examples of the use of these datasets in FERRET are given. Of course, the user is not restricted to the use of FERRET for visualisation or required to write their data into netCDF format, they are free to use any software they have available. The advantages of using a standard data format like netCDF is the range of software available (both commercial, shareware and freeware) which automatically recognises the data format and the ease with which data can migrate between platforms and users. FERRET is found to be the best of the available freeware packages, it fulfils many of the requirements of the COHERENS user: time series (single point, space/time), contours (line and fill), vectors, surfaces (wire frame), overlays and movie generation, both in real world" (longitude/latitude) as in Cartesian coordinates.

The following items are discussed in this chapter:

- a review of the COHERENS model output data structure (Section I-4.2)
- a brief description of netCDF and how to install it (Section I-4.3.1)
- a description of the interface program to convert COHERENS data sets in netCDF format (Sections I-4.3.2 and I-4.3.4)
- a brief description of FERRET and how to install it (Section I-4.4.1)
- displaying model results with FERRET (Section I-4.4.2)

4.2 Review of COHERENS model output

4.2.1 Output files

The properties of the output data are defined by the user following the instructions given in Section I-3.1.3 and I-3.1.4. Model output is written by COHERENS in the form of three files:

- an Information file with specifications about the output data (location of the output in space and time; number of 2-D and 3-D vector and scalar fields; names, descriptions and units of the output data; ASCII or binary output)
- a data file with values of 2-D output data
- a data file with values of 3-D output data

As an example, the names of these output files for the **cones** experiment A are given by *conesA_1.outIA, conesA_1.out2U* and *conesA_1.out3U*. The general form of an output file name is a 14-character string:

- characters 1–6: output title. In the case of a test case run this is given by the name of the test case (5 characters, e.g. *cones*) followed by the name of the experiment (1 character, e.g. *A*).
- character 7: "_" (underscore)
- character 8: file number
- character 9: "." (dot)
- characters 10–12: describes the kind of output. Allowed values are:
  - *out*: time series output
  - *avr*: time-averaged output
  - *res, amp, pha, ell*: harmonic output
  - *par*: particle output
- characters 13: "I" for an Information file, "2" for a 2-D output data file, "3" for a 3-D output data file
- character 14: type of data format, either 'A' for ASCII or 'U' for binary data. Note that the Information file is always in 'A'-format.

4.2.2 Information files

An example of an Information file is listed below for the **cones** experiment A.

```
Record  1:COHERENS header information :
Record  2:Run title : conesA
Record  3:Results file format : U
Record  4:Header information : F
Record  5:Grid file :
Record  6:conesA.grdA
Record  7:File format
Record  8:A
Record  9:NC NR NZ NSTEP :     40     40      2       1008
Record 10:Grid type : 0
Record 11:XRES :       1     40      1
Record 12:YRES :       1     40      1
Record 13:ZRES :       2      2      1
Record 14:TRES :          0       1008          42
Record 15:Number of 2D vectors :  0
Record 16:Number of 2D scalars :  1
Record 17:CONCN   :Contaminant                  :g/m3
Record 18:Number of 3D vectors :  1
Record 19:U2      :U-velocity                   :m/s
Record 20:V2      :V-velocity                   :m/s
Record 21:W2PHYS  :W-velocity                   :m/s
Record 22:Number of 3D scalars :  1
Record 23:CONCN   :Contaminant                  :g/m3
Record 24:2D results file :
Record 25:conesA_1.out2U
Record 26:3D results file :
Record 27:conesA_1.out3U
Record 28:DELX =  1.0000000E+00
Record 29:DELY =  1.0000000E+00
Record 30:DELT =  1.5000000E+01
```

Records 1, 5, 7, 24 and 26 are header lines. The other records provide the following information:

Record 2: The output title is conesA (first 6 characters in the output file name).

Record 3: The data are written in 'U'-format (binary). Alternative is 'A' which means ASCII-formatted data.

Record 4: The output data are written without extra header lines ('F'). Alternative is 'T' which informs that extra header lines are written in the output data.

Record 6: Name of the output file with the grid data. (See Section IV-2.3 and Table 4.2.4 for a description of the grid arrays.)

Record 8: The grid file is written in 'A'-format.

Record 9: The simulation uses 40 grid points in the X-direction, 40 grid points in the Y-direction, 2 vertical levels and 1008 (2-D) time steps. The FORTRAN names of these parameters are NC, NR, NZ and NSTEP.

Record 10: A Cartesian grid is selected (value "0"). Alternative is "1" for a spherical grid.

Record 11: Column data are written with a IFIRST value of 1, ILAST value of 40 and ISTRIDE value of 1.

Record 12: Row data are written with a JFIRST value of 1, JLAST value of 40 and JSTRIDE value of 1.

Record 13: Level data are written with a KFIRST value of 2, KLAST value of 2 and KSTRIDE value of 1.

Record 14: Data are written in time using a LFIRST value of 0, LLAST value of 1008 and LSTRIDE value of 42.

Record 15: Number of 2-D vectors is 0.

Record 16: Number of 2-D scalars is 1.

Record 17: CONCN is the FORTRAN name of the first 2-D scalar, Contaminant its description and g/m3 its unit.

Record 18: Number of 3-D vectors is 1.

Record 19: U2 is the FORTRAN name of the X-component of the first 3-D vector, U-velocity its description and m/s its unit.

Record 20: V2 is the FORTRAN name of the Y-component of the first 3-D vector, V-velocity its description and m/s its unit.

Record 21: W2PHYS is the FORTRAN name of the Z-component of the first 3-D vector, W-velocity its description and m/s its unit.

Record 22: Number of 3-D scalars is 1.

Record 23: CONCN is the FORTRAN name of the first 3-D scalar, Contaminant its description and g/m3 its unit.

Record 25: Name of the 2-D output data file.

Record 27: Name of the 3-D output data file.

Record 28: The model grid has a (uniform) grid spacing of 1 m in the X-direction.

Record 29: The model grid has a (uniform) grid spacing of 1 m in the Y-direction.

Record 30: The model uses a time step of 15 seconds represented by the FORTRAN parameter DELT.

From this Information File it is found that there are 2 associated data files written in binary format:

- *conesA_1.out2U*: containing zero 2-D vector arrays and one 2-D scalar array
- *conesA_1.out3U*: containing one 3-D vector array and one 3-D scalar array

The FORTRAN names, descriptions and units are obtained from records 17, 19–21 and 23.

Two- and three-dimensional arrays have the following dimensions in COHERENS:

- 2-D arrays: X(NR,NC) where NR equals the number of rows (number of grid points in the Y-direction) and NC the number of columns (number of grid points in the X-direction)
- 3-D arrays: X(NZ,NR,NC) where NZ equals the number of vertical levels

The array dimensions NC, NR, NZ are stored in record 9. The total simulated time is obtained by multiplying the time step DELT (record 30) with the total number of time steps NSTEP (obtained from record 9)[1]. Records 11–14 inform that that the arrays are written to the data files using the column indices IFIRST to ILAST with stride ISTRIDE, row indices JFIRST to JLAST with stride JSTRIDE, level indices KFIRST to KLAST with stride KSTRIDE and at time levels LFIRST to LLAST with stride LSTRIDE. The first and last output time (measured with respect to the startup time of the simulation) and the output time interval in seconds are determined by multiplying LFIRST, LLAST and LSTRIDE with the time step DELT.

The type of the grid (Cartesian or spherical) is given by record 10. The parameters DELX and DELY are the grid spacings, measured in meters (decimal degrees), in the X-direction (longitude) and the Y-direction (latitude) for the case of a Cartesian (spherical) grid. These parameters are only meaningfull for a regular grid. More information concerning the model grid is obtained by reading the following 1-D and 2-D grid arrays, stored in the file whose name is given by record 6:

- GX0(NC+1): X-coordinates or longitude of the cell corners

[1]Note that DELT is the 2-D mode time step which is usually smaller than the time step used in the 3-D calculations. See Section III-4.2.4 for details.

- GY0(NR+1): Y-coordinates or latitude of the cell corners
- GZ0(NZ+1): σ-coordinate of the vertical level surfaces
- DEP(NR,NC): array of mean water depths
- NWD(NR,NC): cell pointer array which takes the value of 0 on dry and 1 on a wet cells

For more information about the grid arrays and the contents of the Information file see Sections IV-4.2.2 and IV-2.3.

The example Information file, listed above, is a typical file for time series output. Similar files are written for harmonic or time-averaged output[2]. The Information and data files, related to particle output, have, however, a different form. This will not be discussed in this chapter. For more information see Section IV-4.2.2b.

4.2.3 Reading model output

The sizes of the 2-D and 3-D output arrays are determined using the first, last and stride values given in the Information file. Denoting these dimensions by NCMAX, NRMAX and NZMAX one has

```
NCMAX = (ILAST-IFIRST)/ISTRIDE + 1
NRMAX = (JLAST-JFIRST)/JSTRIDE + 1
```

and

```
NZMAX = (KLAST-KFIRST)/KSTRIDE + 1
```

for the 3-D case while NZMAX is set to 1 in the 2-D case. The arrays are stored in the data files at each of the time levels, selected with LFIRST, LLAST and LSTRIDE and in the same order as they are defined in the Information file. For example, in the case of the **cones** test case A, the array CONCN (the only 2-D array) is stored in the 2-D output file while the array U2 is first stored into the 3-D output file, followed by V2, W2PHYS and CONCN. The total number of output times is then given by

```
NTMAX = (LLAST-LFIRST)/LSTRIDE + 1
```

Binary data are read using

```
READ(IUNIT) (((X(K,J,I),K=1,NZMAX),J=1,NRMAX),I=1,NCMAX)
```

In the case of ASCII-data without headers the READ instructions are

[2]Note that, in the case of harmonic or time-averaged output, the period for harmonic analysis or time-averaging is given by the value of LSTRIDE multiplied by DELT.

```
      READ(IUNIT,*)
      READ(IUNIT,'(100(1PE14.7,1X))') (((X(K,J,I),K=1,NZMAX),
     1                                  J=1,NRMAX),I=1,NCMAX)
```

Extra header lines are inserted in the ASCII-formatted data files with extra header information (value 'T' in record 4 of the Information file). This gives

```
      READ(IUNIT,*)
      DO 110 I=1,NCMAX
         READ (IUNIT,*)
         DO 111 J=1,NRMAX
            READ (IUNIT,*)
            READ (IUNIT,'(100(1PE14.7,1X))') (X(K,J,I),K=1,NZMAX)
  111    CONTINUE
  110 CONTINUE
```

4.3 Creating netCDF output with COHERENS

4.3.1 Installing netCDF

Much of the following text is taken from the netCDF home pages which can be accessed through

`http://www.unidata.ucar.edu/packages/netcdf/`

The netCDF software was developed at the Unidata Program Center in Boulder, Colorado, USA. NetCDF (network Common Data Form) is an interface for array-oriented data access and a freely-distributed collection of software libraries for C, Fortran, C++, and perl that provide implementations of the interface. The netCDF software was developed by Russ Rew, Glenn Davis, Steve Emmerson and Harvey Davies (Rew et al., 1997) at the Unidata Program Center in Boulder, Colorado, and augmented by contributions from other netCDF users. The netCDF libraries define a machine-independent format for representing scientific data. Together, the interface, libraries, and format support the creation, access, and sharing of scientific data.

NetCDF data is:

- Self-Describing. A netCDF file includes information about the data it contains.
- Network-transparent. A netCDF file is represented in a form that can be accessed by computers with different ways of storing integers, characters, and floating-point numbers.
- Direct-access. A small subset of a large dataset may be accessed efficiently, without first reading through all the preceding data.

- Appendable. Data can be appended to a netCDF dataset along one dimension without copying the dataset or redefining its structure. The structure of a netCDF dataset can be changed, though this sometimes causes the dataset to be copied.
- Sharable. One writer and multiple readers may simultaneously access the same netCDF file.

The netCDF mailing list has over 500 addresses (some of which are aliases to more addresses) in thirty countries. Several groups have adopted netCDF as a standard way to represent some forms of scientific data.

A description of some of the projects and groups that have used netCDF is available from

`http://www.unidata.ucar.edu/packages/netcdf/usage.html`

The 3.4 release of netCDF is supported on the following platforms:

- AIX-4.3
- BSD/OS 3.0
- HPUX-10.20, HPUX-11.00
- IRIX-5.3, IRIX-6.5, IRIX64-6.5
- Linux 2.0.31
- OSF1-4.0
- SunOS-4.1.4, SunOS-5.6 (Sparc and i386)
- SUPER-UX 7.2 (NEC SX-4)
- ULTRIX-4.5
- UNICOS-9.0 (C90, T3D), 9.1 (T90), 10.0 (J90), UNICOS/mk 2.0 (T3E)

NetCDF is also available for Windows 95 or Windows NT via user contributions (see netCDF FAQ page)

Software revisions and upgrades to the netCDF package are provided by Unidata. NetCDF source distributions are available via anonymous FTP from the directory

`ftp://ftp.unidata.ucar.edu/pub/netcdf/`

or from other mirror sites.

Files in that directory include:

netcdf-3.4.tar.Z
: compressed "tar" file of source code for the latest general release, Version 3.4, March 1998

guidec.ps.Z
: NetCDF User's Guide for C (compressed PostScript)

guidec.html.tar.Z
: NetCDF User's Guide for C (compressed tar HTML files)

guidef.ps.Z
: NetCDF User's Guide for FORTRAN (compressed PostScript)

guidef.html.tar.Z
: NetCDF User's Guide for FORTRAN (compressed tar HTML files)

Binary distributions for some platforms are available from the directory

`ftp://ftp.unidata.ucar.edu/pub/binary/`

Once you have downloaded the most recent Version 3.4 of netCDF from the web site it will be necessary to install it.

1. First uncompress the downloaded file e.g.

   ```
   uncompress netcdf-3.4.tar.Z
   ```

2. Then "untar" it

   ```
   tar xvf netcdf-3.4.tar
   ```

3. You will need 1.2 Mbytes to download the tarred compress file and 4.3 Mbytes for the uncompressed, untarred directory and 2.8 Mbytes for the installed library.

4. Follow the installation instructions found in INSTALL, located in the **netcdf-3.4/src** directory.

5. Don't forget to add the location of the netCDF library to your path.

4.3.2 Creating netCDF output for a test case application

A FORTRAN 77 program, called **netcdfint**, is provided which converts model data into a portable netCDF format. The interface program uses Version 3.4 of the netCDF library.

Before running **netcdfint** the user must ensure that the program is compiled with the appropriate instructions. This is performed as follows:

- Edit the file *MakePost* in the **SOURCE** directory

- Replace f77 in the line FC = f77 by the appropriate command for compiling FORTRAN programs.
- Replace path in the line FLIB = -Lpath -lnetcdf by the full pathname of the directory containing the netCDF library *libnetcdf.a* (usually in **. . . /netcdf-3.4/src/libsrc**)

NetCDF output for a particular test case is then performed as follows:

1. Prepare and run the test case application using the shell scripts **Prepare** and **Run** along the steps outlined in Section I-2.2.
2. Execute the shell script **Runpost**:

   ```
   Runpost arg
   ```

   where arg must be the same as the argument used with **Prepare** and **Run**. The following actions are performed by **Runpost**:

   - The interface program **netcdfint** is compiled with the **make** utility using the instructions given in *MakePost*.
   - The program is run and creates a series of binary files in netCDF format (see below).
3. The interface program writes a *log_file*. Its name is given by *'postlog'* followed by *'arg'*. If no run-time errors occurred, the last line of the *postlog*-file should read:

   ```
   Interface program terminated
   ```

A netCDF file is created for each 2-D and 3-D output file. The name of each file is given by the name of the corresponding data file appended by *'.cdf'*. For example, executing the main and interface program for the **cones** experiment A produces the model data files *conesA_1.out2U* and *conesA_1.out2U* and the netCDF files *conesA_1.out2U.cdf* and *conesA_1.out3U.cdf*. Examples how to display netCDF output using FERRET are described in Section I-4.4.2.

Although written in binary form, a netCDF file can be converted into an ASCII-format using the netCDF utility program **ncdump**. Invoking the command

```
ncdump input_file
```

produces an ASCII version of the binary netCDF file *input_file* in CDL-format (network Common Data form Language). The **ncdump** command has several options. For example, the -h option shows only the header information in the output, i.e. declarations for the netCDF dimensions, variables, and the attributes of the input file, but without data values for any variable. As an illustration, typing

```
ncdump -h conesA_1.out3U.cdf
```

gives the following output

```
netcdf conesA_1.out3U {
dimensions:
x = 40 ;
y = 40 ;
z = 1 ;
time = UNLIMITED ; // (25 currently)
variables:
float DEP(y, x) ;
DEP:long_name = "water_depth" ;
DEP:units = "m" ;
DEP:_FillValue = 1.e+34f ;
float U2(time, z, y, x) ;
U2:long_name = "U3D                     " ;
U2:units = "m/s       " ;
U2:_FillValue = 1.e+34f ;
float V2(time, z, y, x) ;
V2:long_name = "V3D                     " ;
V2:units = "m/s       " ;
V2:_FillValue = 1.e+34f ;
float W2PHYS(time, z, y, x) ;
W2PHYS:long_name = "W3D                     " ;
W2PHYS:units = "m/s       " ;
W2PHYS:_FillValue = 1.e+34f ;
float CONCN(time, z, y, x) ;
CONCN:long_name = "Contaminant             " ;
CONCN:units = "g/m3      " ;
CONCN:_FillValue = 1.e+34f ;
double x(x) ;
x:units = "meters" ;
double y(y) ;
y:units = "meters" ;
double z(z) ;
z:units = "sigma_level" ;
z:positive = "up" ;
double time(time) ;
time:time_origin = "zero" ;
time:units = "hours" ;

// global attributes:
:title = "conesA_1.out3U" ;
}
```

The section with the heading "dimensions" shows that two (horizontal) dimensions with names x and y, one (vertical) dimension z and one ("unlimited") time dimension time have been defined. The values of the dimensions are equal to the spatial and time dimensions NCMAX, NRMAX, NZMAX, NTMAX of the corresponding output data file (see Section I-4.2.3).

The "variables" part of the listing contains a number of array decalarations:

- The first variable is always the bathymetric array with the mean water depths, which, obviously, has no time dimension.
- U2, V2, W2PHYS: the first (and only) 3-D vector field in the output data file
- CONCN: the first (and only) 3-D scalar field in the output data file
- x, y, z, time: the coordinate arrays in the X-, Y-, Z- and time directions

Note that the coordinate arrays have the same name as the corresponding dimensions. The names of the other variables are obtained from the Information file.

A number of "attributes" are associated with each data variable:

- long_name: the description of the variables as obtained from the Information File (for the reasons explained below the long_name values of vector components are replaced by (U3D, V3D, W3D)
- units: unit of the variable derived from the Information File
- _Fillvalue: "fill" value used by netCDF to pre-fill disk space allocated to the variable (see netCDF User's Guide for details)

The following attributes are associated with the coordinate variables:

- units: An appropriate length or time unit is selected by the interface program depending on the type of the grid (Cartesian or spherical), the horizontal scales of the computational grid (as measured by the parameters DELX, DELY) and the time period covered by the output data. Horizontal lengths are either measured in "m", "km" or "degrees", the time unit is either "seconds, "hours" or "days". The appropriate conversions are performed by the interface program.
- positive: Attribute with value "up" to indicate the preferred plotting orientation in the vertical (attribute used by FERRET).
- time_origin: Attribute with value "zero" to indicate that time is measured with respect to the start of the simulation.

The last line in the listing shows that one global atribute “title” is defined. “Global” means that it is not associated with any data or coordinate value. Its value is given by the name of the cooresponding model output file and can be used by the graphical software to produce a plot title.

Executing ncdump without any option would show an additional data section with the values of each data array.

For more information about the ncdump utility, the CDL-language and netCDF in general, the user is referred to the netCDF User's Guide.

4.3.3 Creating netCDF output for a user application

Creating netCDF output for a user application[3] is easily performed along the following steps:

1. If the interface program is compiled for the first time, the user must first check the file *MakePost* in the **SOURCE** directory using the instructions given in the beginning of Section I-4.3.2.

2. Proceed along steps 1–11 described in Section I-2.3 to set up, compile and run the preprocessor and main program.

3. A parameter file, either in the **PROJECT** or in the “work” directory where the simulation is performed, must be supplied to inform the interface program which data files are to be converted into equivalent netCDF files. The default name of this file is *defpost.par*. Its contents are read by **netcdfint** from standard input. The file is composed of the following lines:

```
Header line
NOFILES
OUTFIL(1)
...
OUTFIL(NOFILES)
```

   - The first line is only a comment line but need to be there.
   - The parameter NOFILES is of type INTEGER and equals the number of subsequent lines.
   - OUTFIL(1) to OUTFIL(NOFILES) are of type CHARACTER*12 and are equal to the first 12 characters of the model data files (i.e. without the ending characters ‘2A’, ‘3A’, ‘2U’ or ‘3U’) for which netCDF output conversion is requested.

   Assume, for example, that the user defined an application with the title mytest with with 2 time series output files and 1 time-averaged output file. To convert all output data into netCDF form the file *defpost.par* should (e.g.) read:

[3]Particle output in netCDF format is not included in the current version of the program.

```
# netCDF output for mytest
3
mytest_1.out
mytest_2.out
mytest_1.avr
```

4. Compile the interface program with the **make** utility:

   ```
   make -f MakePost
   ```

5. Run the interface program

   ```
   netcdfint < param_file > log_file
   ```

   where *param_file* is the name of the parameter file described in step 3 and *log_file* a file where "run-time" information is written by the program. When the default name *defpost.par* is used, input redirection can be omitted. The "log"-file is of no use for the program but can be helpfull to detect errors during execution of the program. If no run-time error occurred, the last line of the "log-file" should read:

   ```
   Interface program terminated
   ```

6. A number of binary netCDF files is produced. Their names are given by the corresponding 2-D and 3-D model data file appended by *.cdf* [4]. As an example, the following netCDF files are written for the mytest application above:

   mytest_1.out2U.cdf, mytest_2.out2U.cdf, mytest_1.avr2U.cdf

   for 2-D data, and

   mytest_1.out3U.cdf, mytest_2.out3U.cdf, mytest_1.avr3U.cdf

   for 3-D data.

The output created with **netcdfint** is portable and can be used by FERRET or any other graphical package which understands the netCDF file format to visualise model results.

[4] No netCDF file is written if the corresponding 2-D or 3-D model output file appears to be empty.

4.3.4 The interface program netcdfint

The aim of this section is to describe the structure of **netcdfint** and to provide a few recommendations for defining names of variables.

The structure of the interface program is as follows:

- Read the value of NOFILES and the first output file name from the input parameter file (*defpost.par* by default).
- Construct the name of the Information File by appending '*IA*' to the output name.
- Read the Information file.
- Read the model file with the bathymetric data (its name, including the directory path, is given by record 6 in the Information file).
- Read the 2-D model data and create/write the associated netCDF file by calling CDF2D (except when the total number of 2-D fields equals zero).
- Read the 3-D model data and create/write the associated netCDF file by calling CDF3D (except when the total number of 3-D fields equals zero).
- Read the next output file name (if any) from the input parameter file and proceed along the same steps.
- The whole procedure is repeated up to the last file in the input parameter file.

Subroutine CDF3D performs the following:

- Enter "define" mode.
- Define dimension names and values (using an "unlimited" time dimension).
- Define names of the (3-D) variables using the "names" string obtained from the Information file.
- Define names for the coordinate variables (space and time).
- Define the "long_name" and "units" attributes for the 3-D data variables using the "description" and "units" strings obtained from the Information file.
- Define the attributes for coordinate variables (as discussed in Section I-4.3.2 the units for the horizontal and time coordinates are determined by the program).
- Leave "define" mode.
- Write the spatial coordinate arrays and the bathymetric array to the netCDF file.

- Read the 3-D model output for each time output time step.
- Store the 3-D model output in the corresponding netCDF file for each output time step.
- Write the time coordinate at each output time step.

Subroutine CDF2D has the same purpose now for 2-D data. Additional attributes may be defined by the user in CDF3D and CDF2D. The two routines call a number of routines from the netCDF library. For more information see the netCDF User's Guide for FORTRAN (Version 3).

The descriptions and units of the 3-D output fields are defined by the user through the CHARACTER arrays DESC3D and UNIT3D in subroutine OUTPT3D for time series output (file *defout.f*), ANALPT3D for harmonic output (file *defanal.f*) and AVRPT3D for time-averaged output (file *defavr.f*). The descriptions and units of the 2-D output fields are defined in the same way by the arrays DESC2D and UNIT2D in subroutines OUTPT2D, ANALPT2D and AVRPT2D.

A number of restrictions and conventions should or may be adopted.

- Descriptions must start with an alphabetic character followed by zero or more alphanumeric characters including the underscore '_'. Note that names starting with '_' are reserved for internal use in the netCDF library.
- Description and unit names are case sensitive.
- If netCDF output is used for graphical display with FERRET, reserved names must be avoided (I, J, K, L, X, Y, Z, T, XBOX, YBOX, ZBOX, TBOX, ...).
- A series of netCDF conventions, sponsored by the "Cooperative Ocean-Atmosphere Research Data Service" (COARDS), are adopted by some graphical packages, such as FERRET. A document, describing these documents, is available at

  `http://www.unidata.ucar.edu/packages/netcdf/conventions.html`

The following additional constraints have been implemented in **netcdfint** for compatibility with FERRET:

- The COHERENS program allows to use up to 30 characters in the "descriptions" arrays DESC3D and DESC2D (17 characters in the case of harmonic output). The strings are cut off after character 21 in the case of time series and time-averaged output and after character 8 in the case of harmonic output.
- In the same way, only the first 10 characters in the "units" arrays UNIT2D and UNIT3D are taken into account whereas COHERENS allows to use up to 16 characters.

- Blanks between words are replaced by '_'.
- Invalid characters, such as '-' and '%' are replaced by a '_'.
- The interface program replaces the names of all 3-D vector components by U3D, V3D, W3D and the names of all 2-D vector components by U2D, V2D. This description helps FERRET to associate velocity components with vector quantities. The disadvantage is that only one 3-D vector and one 2-D vector, usually representing the 3-D current vector and its depth-averaged or depth-integrated counterpart, are allowed in the current version of **netcdfint**. This problem can be remedied in future versions.

4.4 Visualisation of model results with FERRET

4.4.1 Installing FERRET

Much of the following text is taken from the FERRET WWW site

`http://ferret.wrc.noaa.gov/Ferret/`

FERRET is an interactive computer visualisation and analysis environment designed to meet the needs of oceanographers and meteorologists analyzing large and complex gridded data sets. It is well suited to the analysis and display of remote data sets via X-windowing as well as being portable to a number of modern computer systems: SunOS, Solaris, DECstation, IBM/AIX, SGI, Hewlett-Packard HP-UX and Linux.

FERRET was developed by the Thermal Modeling and Analysis Project (TMAP) at PMEL (Pacific Marine Environmental Laboratory) in Seattle, USA to analyze the outputs of its numerical ocean models and compare them with gridded, observational data (Hankin and Denham, 1996). The model data sets are generally multi-gigabyte in size with mixed 3 and 4-dimensional variables defined on staggered grids. FERRET offers a Mathematica-like approach to analysis; new variables may be defined interactively as mathematical expressions involving data set variables. Calculations may be applied over arbitrarily shaped regions. Fully documented graphics are produced with a single command.

Many excellent software packages have been developed recently for scientific visualisation. The features that make FERRET distinctive among these packages are Mathematica-like flexibility, geophysical formatting, "intelligent" connection to its data base, memory management for very large calculations, and symmetrical processing in 4 dimensions.

FERRET is widely used in the oceanographics community to analyze data and create publication quality graphics. It is available in binary format for each of the systems listed below.

- DEC Alpha/OSF-1
- DEC MIPS/Ultrix
- Sun (SunOS 4.x)

- Sun (Solaris)
- IBM (RS6000/AIX)
- SGI IRIX 4.x, 5.x and 6.x
- HP-UX
- LINUX

FERRET can be obtained via anonymous ftp from

`http://ferret.wrc.noaa.gov/Ferret/Downloads/`

Approximately 30 Mbytes of disk space are required to install FERRET, preferably on your **/usr/local** file system. Another 30 Mbytes are needed on a disk device to install the demonstration data sets. If disk space is a problem, the section Reducing Disk Space Required in the installation guide provides assistance with which files are safe to delete.

The FERRET distribution is made up of three compressed *tar*-files:

- *fer_executables.tar.Z* (Ferret and utilities)
- *fer_environment.tar.Z* (support files)
- *fer_dsets.tar.Z* (sample data sets)

All three of these files need to be placed on your system. After acquiring them, follow the steps below to install the application.

The installation of FERRET should be as follows:

1. Go to the directory for your operating system.
2. Read the README file which describes which files you need.
3. Look for files called "release_notes.xxx" that summarize recent changes to the FERRET program.
4. Although it is somewhat large (30 megabytes), you are strongly encouraged to pick up the file "fer_dsets.tar.Z" which contains sample gridded data sets used in the FERRET demonstrations and tutorials.
5. The Installation and Update Guide will lead you through the installation procedure.

Documentation is installed as part of the installation procedure. Examine the files located in **$FER_DIR/doc** for a number of online documents about FERRET. Files ending in *'.ps'* are in PostScript format and are intended for printing on PostScript printers. Other documents are not intended for hard copy but are ASCII text versions of the PostScript documents. They are suitable for on-line review.

The script **print_ferret_docs** presents a menu from which you may choose documents to be printed.

4.4.2 Using FERRET to display model results

FERRET can be used in one of two ways. For the command line entry mode type

```
ferret
```

for the graphical user interface mode type

```
ferret -gui
```

For a productive and smooth start using FERRET, review the Demonstration Files section of the Ferret User's Guide (Type 'Fhelp "demonstration files"' at the shell prompt for a list). Tutorial scripts demonstrating Ferret data sets and usage are documented there. Users are recommended to try them out first.

In addition, the tutorial scripts (journal files) in **$FER_DIR/examples** are excellent source material for your own scripts. Copy any or all to your own area and print out, study and modify them to suit your needs.

In the remainder of this section it is explained how results from a test case simulation can be viewed with FERRET (Version 4.4) using the **cones** test as an example. Output from all the other test cases (except for the **front** case) can be visualised in a similar way[5] and compared with the corresponding figures in Part II.

Firstly, it is assumed that the **cones** experiment A has been run and that the interface program **netcdfint** produced the two netCDF files:

conesA_1.out2U.cdf, *conesA_1.out3U.cdf*

Viewing the results with FERRET is performed as follows.

- Invoke FERRET with the graphical user interface

```
ferret -gui
```

[5]A drawback of the currently used Version 4.4 of FERRET is its inability to represent data on a σ-coordinate grid, as used in the test case **front**. This appears to be remedied in the new Version 5.0, announced in June 1999.

- The main FERRET interface will be displayed. Select FILE then Open Data Set and choose conesA_1.out2U or conesA_1.out3U from the list (one may have to select SEARCH to change to the correct directory).
- Once the file is selected, clicking on the SELECT button will reveal all the files opened. Select a file and all the variables stored in the data file will be displayed. Use the mouse to select a variable. FERRET is an "intelligent" package and will try and associate scalar quantities to make vectors. If velocity components are given names like (U3D, V3D, W3D), FERRET will attempt to make vector quantities. In addition to the basic U3D, V3D, and W3D one sees, in the SELECT list, quantities like U3D, V3D or U3D, W3D (guidelines on this are given in the FERRET User Guide).
- Clicking on a SELECT variable highlights the plotting options at the bottom of the frame. For example, for scalars options SHADE, CONTOUR and FILL are emboldened. For vectors, only the option VECTOR is emboldened.
- Select a preference and click on PLOT. A separate window will appear containing the plotted variable. To obtain, for example, a plot with colour filled contours of CONCN in cones.out3U, choose the data file from the list, go to DATA, select CONCN, select FILL and then PLOT. The output from these commands is shown in Figure 1.4.1.
- Note from the slider bars for X, Y, Z and T that FERRET has selected the full domain in X and Y, level 1 in Z and level 1 in T. The user can easily change these values or switch from X, Y, Z and T (in coordinate units) to use array indices by changing X to X Index etc[6].
- To add the horizontal current vector field in Figure 1.4.1, click on DATA and select U3D ,V3D (if it has been created) and click on OVERLAY. The output from this combined scalar and vector plot is shown in Figure 1.4.2.
- Plots of data along different axis combinations can be made by selecting the axis set from the PLANE drop down menu and then adjusting the sliders of X, Y, Z and T to produce the required output. For example, to produce a timeseries of Contaminant at location I=10,J=10,K=1, select T LINE and adjust the slider bars in X, Y and Z to the required values, then click on PLOT. The result is given in Figure 1.4.3.
- It may be that the vector quantity U3D, V3D has not been recognised by FERRET at load time. It is a simple matter to create the vector field using the COMMAND LINE window found in the OPTIONS drop down menu. Click on COMMAND LINE and a small window will open which will contain the FERRET commands you have already made by clicking on DATA and PLOT etc. (This is the way to input commands to

[6]Note that these indices refer to the data arrays stored in the output files. If only a limited number of grid points has been selected for model output, the sizes of these arrays are smaller than the ones used in the simulation which means that the same index value may point to a different coordinate value in the data and the model grid array.

FERRET without using the option -gui). To create the overlayed vector plot using the COMMAND LINE window, type into the window

```
VECTOR/I=1:40/J=1:40/K=1/L=1/OVERLAY  U3D,V3D
```

Highlight this line then click on the RUN button. The vectors should appear on the contour plot as shown in Figure 1.4.2. To produce a vector plot only, omit the /OVERLAY option. Note that the domain to be plotted is given by the user (i.e. I=1:40 etc.).

- All different plots can be made using the commands in the COMMAND LINE window. The command

  ```
  FILL/I=1:40/J=1:40/K=1/L=1 CONCN
  ```

  produces the plot using the default colour palette. To produce the greyscale image in Figures 1.4.1 the colour palette must be changed. This is easily done as follows

  ```
  FILL/PAL=inverse\_greyscale/I=1:40/J=1:40/K=1/L=1 CONCN
  ```

- A list of colour palettes can be obtained with the UNIX command Fpalette '*' (see Chapter 6.5 of the FERRET User Guide)
- Useful information on creating commands is given in the User Guide and specific examples can be obtained from the tutorial demonstration scripts provided with FERRET (see above).
- Plots can be saved and subsequently printed as postscript files. Instructions on this are given in Chapter 8.4 of the FERRET User Guide. Briefly, before making a plot the issue the command

  ```
  set mode metafile
  ```

  in the COMMAND LINE window. After this all plots will be written to disk files named (e.g.)

  metafile.plt, metafile.plt.˜1˜, . . . , metafile.plt.˜n˜ etc.

  where *metafile.plt* is the current plot and the most recent plot is the highest number *˜n˜*. To stop making plot files enter

  ```
  cancel mode metafile
  ```

  in the COMMAND LINE.

- Plots can be converted to PostScript files using the UNIX command Fprint. For example,

  ```
  Fprint -o cones_contour.ps -l ps metafile.plt.~6~
  ```

  Chapter 5 of the FERRET User Guide shows how to make animations of your data with a simple few commands. However, to view the animation it is necessary to download additional (free) software. Information on this is provided in Chapter 5 of the FERRET User Guide.

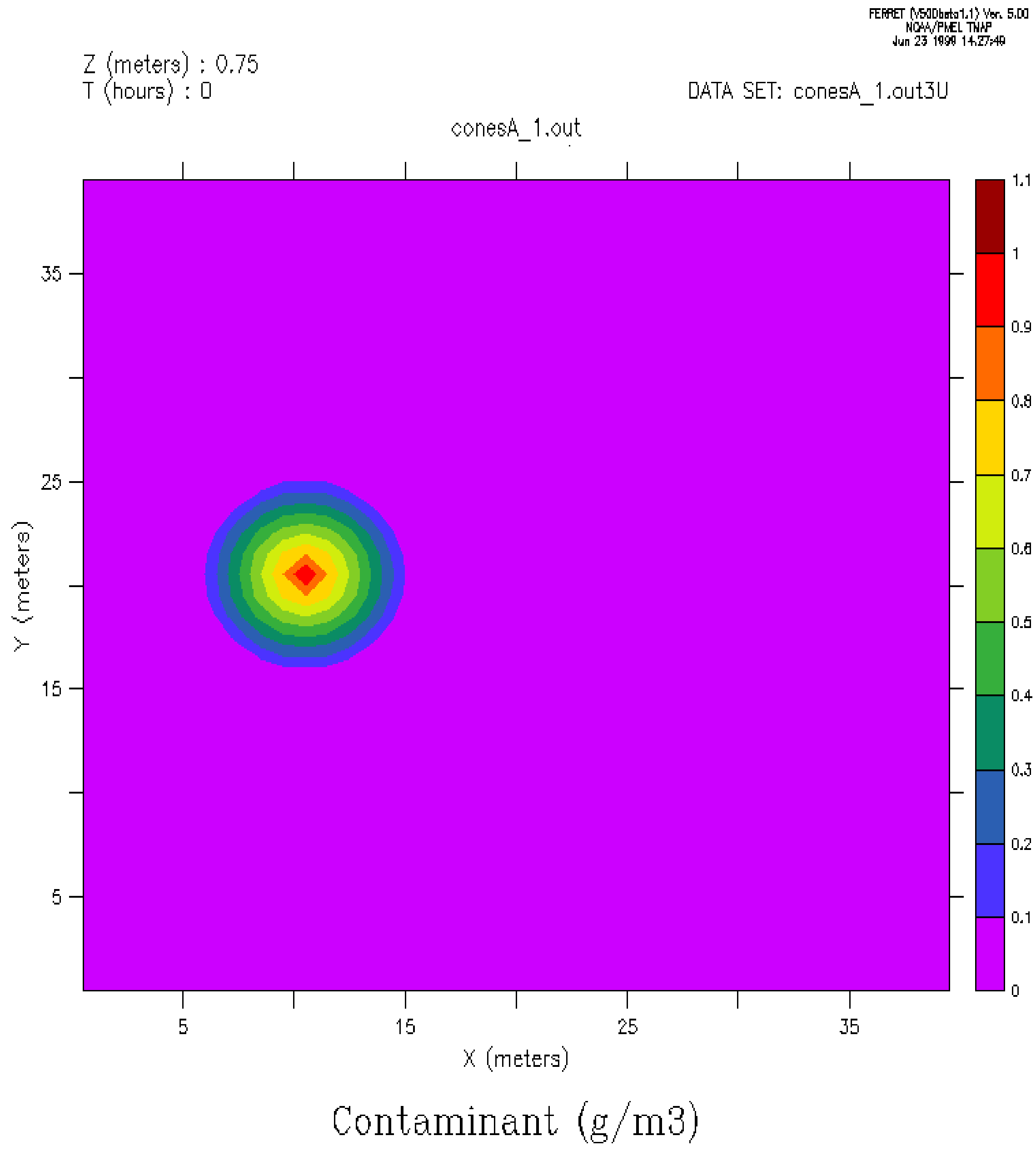


Figure 1.4.1: Initial surface distribution of the contaminant concentration in the **cones** experiment A (plot made using FERRET).

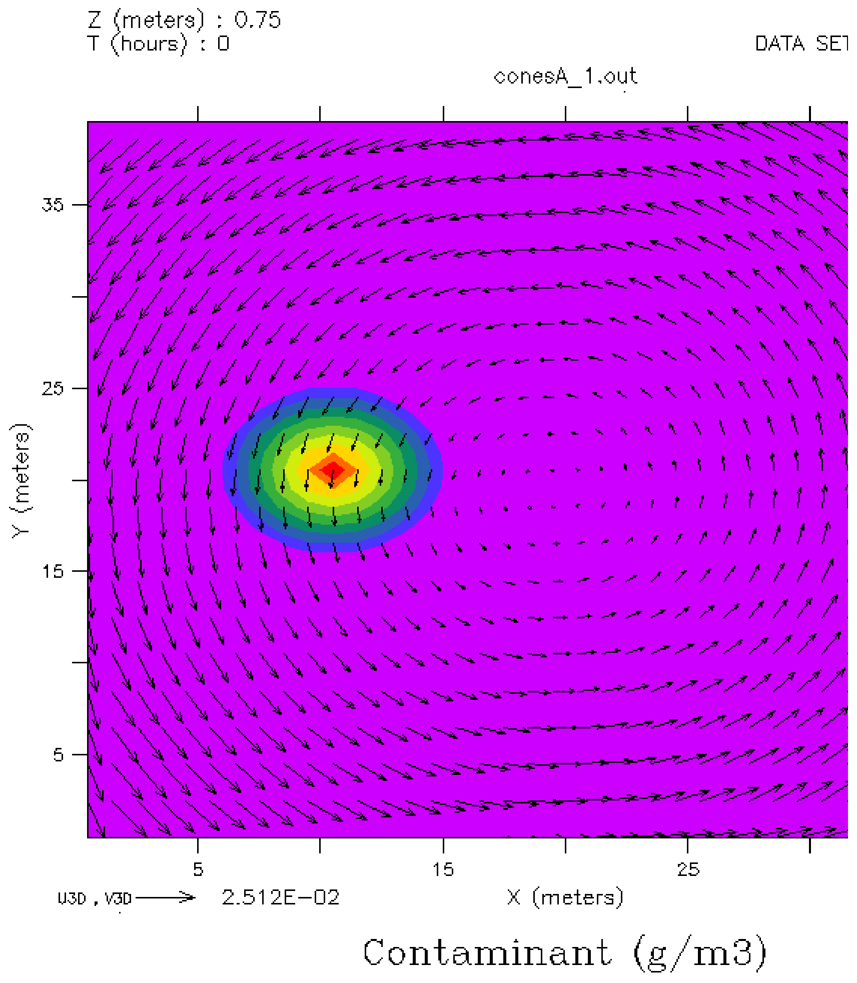


Figure 1.4.2: As Figure 1.4.1 now with the current vector added to the plot (made using FERRET).

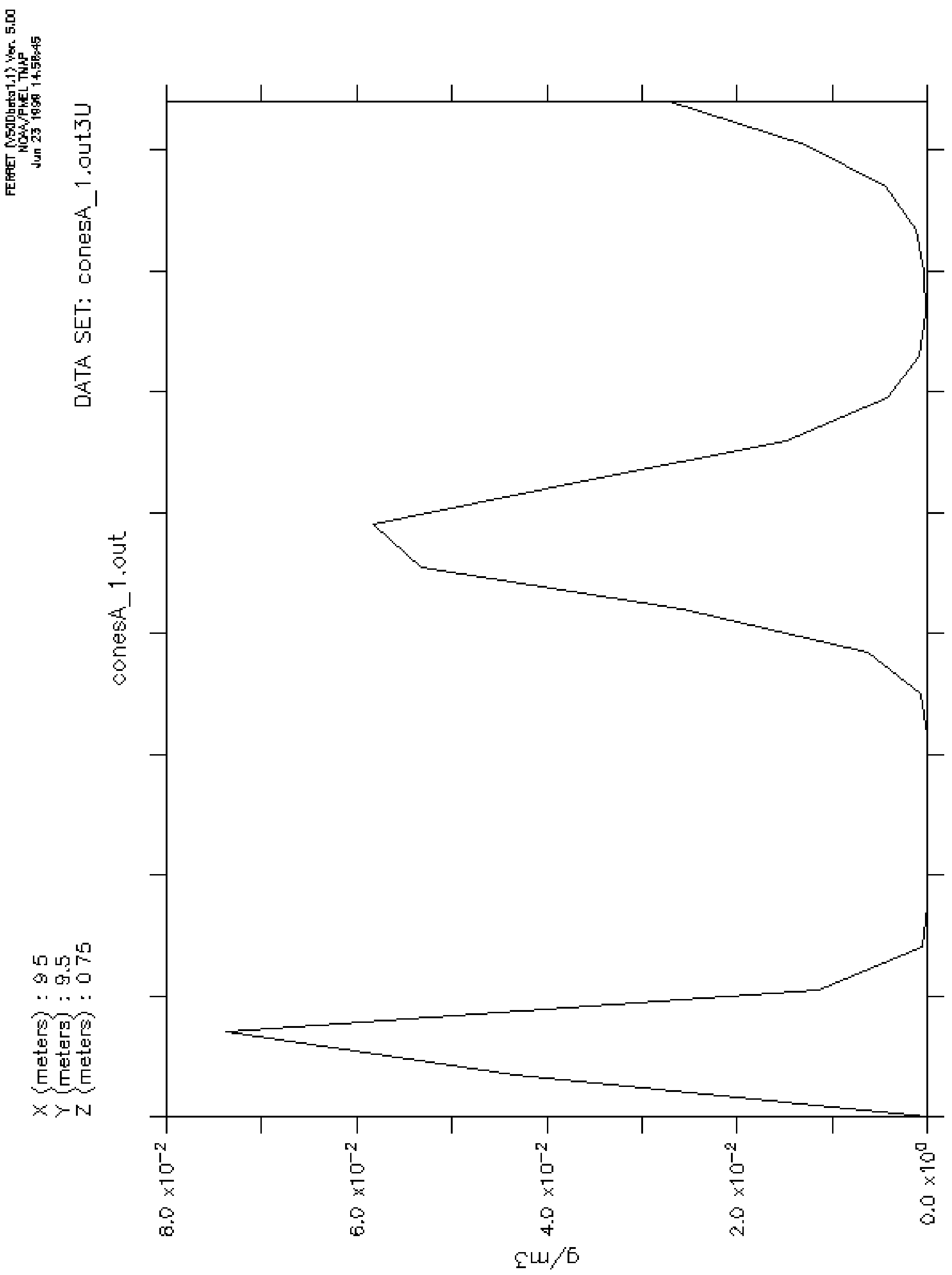


Figure 1.4.3: Time series of the contaminant concentration in the **cones** experiment A at the grid point I=10, J=10, K=1 (plot made using FERRET).

Part II

Model Applications

Chapter 1

Computer Platforms and Portability

1.1 Test cases

The aim of this part of the documentation is to give an overview of the most recent simulations which have been conducted with COHERENS. The first ten are test cases which are described in Chapters II-2 to II-5 and discussed more extensively, the last two are selected applications, performed in the frame of European MAST projects: advection-dispersion study in the North Sea (NOMADS) and the annual physical-biological cycle in the North Sea (COHERENS). The source code used to setup the simulations is available on the CD-ROM. The user is recommended to run one or more of these test cases. This is not possible for the last two applications since this would require the storage of a large amount of input data on the CD-ROM.

The aim of the test cases is to:

- test the portability of the program
- verify the model results against exact or approximate analytical solutions (test cases **cones, front, seich and pycno**) or to validate the model against measured data (test case **csnsp**)
- show how model results are affected by different choices of model switches and parameters
- illustrate how the model is set up for an application
- present an overview of the different processes which can be simulated with COHERENS

The following test cases have been implemented:

1. Test cases for advection.

cones
: advection in a closed basin of an initially coned-shaped contaminant distribution rotating around the basin's centre. Four experiments are defined using different values of the model switch for scalar advection.

front
: advection of a front by a tidal current over a sloping bottom (in σ-coordinates) in an open channel without "physical" diffusion (James, 1996). Four experiments are defined using different values of the model switch for scalar advection.

seich
: adjustment to equilibrium of an initial horizontal density gradient in a closed channel, through the propagation of internal waves. The numerical results are compared with an analytical solution. Four experiments are defined using different values of the model switches for scalar and momentum advection.

fredy
: development of 3-D baroclinic eddies in a closed basin. This test case was previously considered in the NOMADS project (Proctor, 1997). Four experiments are defined using different values of the model switches for scalar and momentum advection. Results are compared with those described in Tartinville et al. (1998).

2. Test cases for turbulence

pycno
: 1-D test describing the evolution of a wind-driven surface layer without rotation. An initial linear stratification is assumed and advection is neglected. The results are compared with analytical theory (Kundu, 1981; Luyten et al., 1996). Seven experiments are defined using different values of the turbulent switches.

csnsp
: seasonal evolution of thermal stratification at station CS in the North Sea. This is also a 1-D problem. The results are compared with the data of the North Sea Project. Eight experiments are defined using different values of the turbulence switches, different surface boundary conditions and different values of the optical parameters.

3. Plume test cases

river
: evolution of an estuarine salinity front advected by a tidal current and the corresponding estuarine circulation in an open non-rotating channel. Four experiments are defined using different values of the switches for turbulence and momentum advection.

plume
: formation and evolution of a tidally modulated river plume front. Some comparison is made with the theory of tidal plumes presented in Visser et al. (1994). Seven experiments are defined testing the role and form of horizontal diffusion, the scheme for momentum advection and the open boundary conditions.

4. Biological test cases

batch
: idealised "batch" culture experiment where populations of microplankton are grown in a mesocosm under controlled conditions. Users may employ this test case to investigate the role of biological model parameters. Three experiments are defined using different initial conditions for nitrate and ammonium.

csbio
: seasonal cycle of biological processes (e.g. spring bloom) and SPM resuspension. The setup is similar as for test case **csnsp** except that a simplified tidal forcing is taken and mean climatological data are considered. Six experiments are defined testing the role of the fluff layer in the model, the sinking processes, the turbulence scheme and wave-current interaction at the seabed.

The instructions for preparing and running a test case are already explained in Chapter I-2. The most important commands are the shell scripts **Prepare** and **Run** executed with the argument `arg`. The argument informs the program which experiment is performed, and takes the value of a capital letter `A` to `H` (see Table 1.2.1). In this way a total of 51 simulations can be conducted with the model. A list of all test cases with the allowed values of `arg` is given in Table 1.2.1.

Each test case presentation in the following chapters is accompanied by three tables. Model setup parameters are listed in the first one with their FORTRAN name, value and purpose. Not included are the parameters whose default values have been retained. Further information about the setup of a test case can be obtained by inspecting the FORTRAN source files in the corresponding **/tests** subdirectory. All these files are constructed from the corresponding generic example files in the **/tests/examples** subdirectory after elimination of all unnecessary source code. The experiments are described in a second table. A series of output parameters, described in the text, are defined for each test case. The source code, where the parameters are defined, is contained in the file *print.f* located in the corresponding **/tests** subdirectory. Their values, obtained on a DEC Alpha machine, are listed in a third table[1]. If no run-time error occurred during the execution of an experiment, a file is produced at the end of the simulation. Its name is composed of 10 characters:

- characters 1–5: name of the test case (as given in the first column of Table 1.2.1)
- character 6: name of the experiment presented by one of the capital letters `A` to `H`

[1]A "*" in one of the output tables indicates that no significant value could be assigned to an output parameter. They are given a flagged value of "-99.00" in the corresponding output file.

- character 7: "." (dot)
- character 8–10: *"tst"*

The portability of the program can be checked by comparing the output values given in *.tst*-file with the ones given in the corresponding table or by comparing the file with the corresponding one in the **/tests/ptests** subdirectory, obtained from a simulation with the DEC Alpha. The results of the test case simulations are illustrated with a large series of figures, which can be compared by the user with the output produced on his/her machine.

1.2 Implementation on different platforms

A total of 7 different platforms were used to test some or all of the test cases with no or moderate optimisation options. The CPU times are compared in Table 2.1.1. The simulations were performed on the following machines[2].

1. A Digital DEC Alpha with 4 Alpha 400 Mhz processors and 1Gb memory (DEC).
2. A Silicon Graphics Indigo with a R4000 processor and 32 Mb memory (R4000).
3. A Silicon Graphics Origin 2000 with 16 processors and 10 Gb memory (Origin).
4. A Cray Y-MP EL with 6 processors and 256 Mwd memory (Cray).
5. A Fujitsu VPP300 CMOS vector machine with 3 processors (each with a 2.2 Gflop performance) and 2 Gb memory (Fujitsu).
6. A Pentium 233 with 128 Mb RAM using an operating system under LINUX (LINUX).
7. A Dell Optiplex GX1 PC with a Pentium 450 Mhz processor and 256 Mb memory (PC).

The FORTRAN 77 source code proved to be portable on all these machines[3]. General aspects of the FORTRAN code and comparison with the ANSI Standard for FORTRAN 77 are discussed in Section IV-1.3.

Non-portable aspects of COHERENS, related to the installation and compilation of the code, and to the use of shell scripts for preparing and running the program, are summarised below for the different types of machines used in the intercomparison study.

[2]The name in parentheses refers to the corresponding column in Table 2.1.1.

[3]A few SAVE instructions have been inserted in the code to ensure portability on machines using STATIC allocation of scalar and array variables.

1.2.1 UNIX machines

COHERENS has been developed on this type of machine so that only a few minor points need to be considered for implementation:

1. Ensure that a C shell is selected which is performed with the UNIX command csh.
2. Edit the *Makefile* for the appropriate way of invoking the FORTRAN 77 compiler. This is performed by replacing f77 on the line FC = f77 by the appropriate command, e.g. cf77 on the Cray or frt on the Fujitsu machine.
3. A further shell script may be required for machines working with a batch queue operating system.

1.2.2 PCs with LINUX

The LINUX PC tests were performed using the LINUX version "Caldera Open Linux Base 1996". The following specific instructions apply:

1. Invoke the C shell instead of the default bash shell with the command csh.
2. Edit the *Makefile* to comment out the line where IFLAGS is defined.
3. Precompile all FORTRAN modules without linking by inserting

   ```
   f77 -c *.f
   ```

   before the make instruction in the **Prepare** script. Note that this line has to be inserted twice in the **Prepare** scripts of the **river** and **plume** test cases before the lines containing the **make** command.

1.2.3 PCs with Microsoft Fortran

The model was run on the Dell Optiplex GX1 PC with the Pentium processor. The operating system was Windows 95 and the Fortran compiler was Microsoft Fortran Powerstation, Professional edition, Version 4.0. The instructions, given below, only serve as an example (using the test case **cones**) and assume that the code was previously installed on a UNIX machine in the form of a *tar*-file.

1. On the UNIX system issue the command

   ```
   tar -xvf release84.tar
   ```

   to create the three **source, tests** and **utilities** directories under the directory name **release_8.4**.

2. A temporary directory, e.g. **cohdir**, is set up

```
mkdir cohdir
cd cohdir
```

3. The next two stages set up the particular test case, e.g. **cones**

```
setenv COHERENS .../release_8.4
$COHERENS/utilities/csetup cones
```

   All the files ready to run the test case are now assembled in the UNIX directory **cohdir**.

4. The files in **cohdir** are transferred to a directory on the PC, for example a partition of the hard disk labelled `E:` with subdirectory `cones`, i.e. `E:\cones`. The transfer used the LAN Workplace rapid filer facility. One file name *biolgy_in.f* and one directory name **utilities** did not conform to the DOS filename conventions. There were two options either (1) allow the automatic renaming or (2) re-expand them once they were on the PC. The latter course was allowed by Microsoft Fortran under Windows 95. Either procedure was acceptable as the file names themselves are not used, merely the routines contained within each file.

5. Microsoft Fortran Developer uses project folders to run programs. These folders can contain program files themselves or links to files in other directories. To allow for the preprocessing stage two project folders were created `C:\msdev\Projects\precones` and `C:\msdev\Projects\cones`.

6. Within these two folders, links were established to all the files in `E:\cones`. In the former cases the link to **preproc** was removed and in the latter case the link to **mainprog**.

7. The file *defmod.par* was transferred from `E:\cones` to `C:\msdev\Projects\precones` and the first line edited to include the appropriate test case letter A, B, C etc.

8. Microsoft Fortran Developer has a **build** command which effectively supplants "Makefiles". The **build** command was then invoked in the precones folder and by default an executable file precone.exe was created. This was then run and the input files created by the preprocessor for the main run itself automatically appeared in the precones folder. These input files were transferred to the `C:\msdev\Projects\cones` directory. Within this directory a **build** was performed resulting in the executable mainprog.exe. This was then run and the output files appeared automatically in the same directory.

9. The above procedure was applicable to the **cones, front, seich, fredy, pycno, csnsp** and **batch** cases. In **river** and **plume** the **Prepare** stage includes an initial preprocessing and initialisation run stage. The above method was extended by, for example, creating four project folders :

```
C:\msdev\Projects\preriv1 C:\msdev\Projects\river1
C:\msdev\Projects\preriv2 C:\msdev\Projects\river2
```

10. The initialisation is then treated as a completely separate run, the output files from each stage being transferred from folder to folder for the next stage in the proceedings.

Table 2.1.1: CPU time table (in seconds) of all test case experiments, performed on different machines.

Experiment		DEC	R4000	Origin	Cray	Fujitsu	LINUX	PC
cones	A	14	179	60	404		118	28
	B	23	347	120	768		223	52
	C	25	449	127	894		236	61
	D	26	433	128	895		237	61
front	A	8	114	35	294		75	19
	B	10	182	61	394		122	30
	C	11	189	63	422		126	30
	D	11	184	63	424		126	30
seich	A	5	62	20	141		43	16
	B	6	77	25	187		53	21
	C	8	116	39	285		81	24
	D	8	101	34	258		72	24
fredy	A	2736	37146	12003	100555		24840	6426
	B	4826	69854	21351	181389		45139	11669
	C	3237	46103	14876	116277		30299	7834
	D	5281	75743	24311	201233		50637	13375
pycno	A	14	198	61	516		142	60
	B	19	300	82	779		193	76
	C	21	319	87	869		204	78
	D	15	249	74	570		176	63
	E	16	250	75	586		175	64
	F	16	265	84	562		193	67
	G	15	245	78	479		182	63
csnsp	A	57	1074	273	1838		653	352
	B	50	792	253	1488		608	346
	C	68	1104	307	2651		726	390
	D	44	783	266	1789		499	340
	E	56	877	273	1839		652	357
	F	56	926	272	1834		652	357
	G	56	893	273	1839		653	355
	H	54	862	266	1819		577	333

Table 2.1.1: continued.

Experiment		DEC	R4000	Origin	Cray	Fujitsu	LINUX	PC
river	A	282	4669	1315	15215	1258	1776	794
	B	238	3608	1132	12489	1027	1543	700
	C	256	6732	1193	14402	1210	1628	718
	D	205	2878	901	10416	896	1266	567
plume	A	6755	65026	21633	257332		39485	17026
	B	4463	44764	12707	164819		26484	11115
	C	5758	54721	15291	227135		31918	14875
	D	5327	50692	14578	211383		30417	12549
	E	5296	50854	14573	205657		30436	12686
	F	5285	50650	14790	208713		30432	13068
	G	6566	61978	21326	257483		37780	17364
batch	A	8	102	27	276		55	254
	B	8	97	27	276		55	253
	C	8	101	28	276		56	254
csbio	A	92	1776	556	2911		1243	691
	B	91	1620	548	2736		1226	716
	C	87	1568	515	2761		1158	692
	D	89	1635	551	2760		1235	698
	E	75	1426	491	2295		1113	677
	F	94	1696	558	2941		1249	726

Chapter 2

Test Cases for Advection

2.1 Test case cones

2.1.1 Description of the problem and model setup

The **cones** problem is a well known problem for testing advection schemes (e.g. Takacs, 1985; Ruddick, 1995). A squared-shaped basin is considered bounded by solid walls. The width of the basin is taken as 40 m and the mesh size of the grid is 1 m. A current field is imposed in the form of a solid body rotation with angular velocity Ω around the point (x_{10},x_{20}) located near the centre of the basin

$$u = -\Omega(x_2 - x_{20}) \ , \ v = \Omega(x_1 - x_{10}) \ , \ w = 0 \tag{2.2.1}$$

where (x_1,x_2) are the horizontal Cartesian coordinates, (u,v) the horizontal components of the current, w the vertical velocity, $\Omega = 1/1200\ \mathrm{s}^{-1}$ and $(x_{10},x_{20}) = (19.5,19.5)$. An initial contaminant distribution has a cone shaped form with radius R and centered at (x_{1c},x_{2c}):

$$C = \max\Big[1 - \frac{\sqrt{(x_1 - x_{1c})^2 + (x_2 - x_{2c})^2}}{R}, 0\Big] \tag{2.2.2}$$

where $R = 5$, $(x_{1c},x_{2c}) = (10.5,20.5)$ and C is the contaminant concentration in arbitrary units. The initial current and contaminant field distributions are plotted in Figure 2.2.1. The program only needs to solve one transport equation. This takes the following simple form in the absence of diffusion, vertical currents, surface elevation and variations in bottom depth

$$\frac{\partial C}{\partial t} + \frac{\partial}{\partial x_1}(Cu) + \frac{\partial}{\partial x_2}(Cv) = 0 \tag{2.2.3}$$

A list of parameters used to setup the program is given in Table 2.2.1.

2.1.2 Experiments and output parameters

The **cones** test intercompares the four scalar advection schemes implemented in the program. The following experiments, also listed in Table 2.2.2, have been set up:

A : upwind scheme

B : Lax-Wendroff scheme

C : TVD scheme using the superbee limiting function

D : TVD scheme using the monotonic limiting function

In each experiment the program is run for two rotation periods. The contaminant distributions at the end of the simulation are shown in Figures 2.2.2a–d for the four schemes. The contour lines are drawn at intervals of 0.05. Figures 2.2.3a–b display the surface distributions after two periods of revolution along two transects through the cone centre parallel to respectively the X- and Y-axis. The exact case is plotted by the solid line for comparison.

A series of parameters are evaluated by the program and written to the appropriate output files at $t = 0.5T$, T, $1.5T$, $2T$ where T is the rotation period.

XMIN
: The cone radius measured in the negative X-direction (m). This is taken as the distance between the cone centre and the point to the left where c becomes less than 0.01 along a line parallel to the X-axis.

XPLUS
: The cone radius measured in the positive X-direction (m). This is taken as the distance between the cone centre and the point to the right where c becomes less than 0.01 along a line parallel to the X-axis.

YMIN
: The cone radius measured in the negative Y-direction (m). This is taken as the distance between the cone centre and the point above where c becomes less than 0.01 along a line parallel to the Y-axis.

YPLUS
: The cone radius measured in the positive Y-direction (m). This is taken as the distance between the cone centre and the point below where c becomes less than 0.01 along a line parallel to the Y-axis.

CMIN
: Minimum value of the concentration over the entire domain.

CMAX
: Maximum value of the concentration over the entire domain.

Exact values are 5 for the four radii, 0 for CMIN and 1 for CMAX. Output values of the parameters are listed in Table 2.2.3.

Table 2.2.1: Model setup for the test case **cones**.

parameter	value	purpose
param.inc		
NC	40	number of grid points in the X-direction
NR	40	number of grid points in the Y-direction
NZ[a]	2	number of grid points in the vertical
NOBU	0	number of open boundary points in the X-direction (a zero value means that the boundaries are closed)
NOBV	0	number of open boundary points in the Y-direction (a zero value means that the boundaries are closed)
NCONC	1	number of contaminants
DEFCON		
IOPTC	1	contaminant module switched on
IADVC	0	advection of momentum disabled
IOPTK	0	vertical diffusion coefficients set to a uniform (zero) value
IOPT2	0	mode splitting disabled
IOPT3	0	solution of the momentum equations disabled (the current field prescribed by equation (2.2.1) is time independent and is therefore not updated in time by the program).
IOPTD	0	uniform density
IBSTR	0	bottom stress set to zero
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01010412	end date (MMDDHHMM)
DELT	15	time step (s)
VISMOL	0.0	vertical diffusion coefficient for momentum (m^2/s)
DIFMOL	0.0	vertical diffusion coefficient for contaminant transport (m^2/s)
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the basin (m)
GX0(NC+1)	40	X-coordinate of the upper right corner of the basin (m)
GY0(1)	0	Y-coordinate of the lower left corner of the basin (m)
GY0(NR+1)	40	Y-coordinate of the upper right corner of the basin (m)
DEP(NR,NC)	6	array of uniform water depths (m) (its exact value is irrelevant for the **cones** test)

[a]Although all vertical dependence is eliminated in the present application, a minimum value of 2 is required by the program structure.

Table 2.2.1: continued.

parameter	value	purpose
DEFICS		
CONCN(NZ,NR,NC,1)		initial contaminant concentration as prescribed by (2.2.2) (g/m^3)
U2(NZ,NR,NC+1)		X-component of the current as prescribed by (2.2.1) (m/s)
V2(NZ,NR+1,NC)		Y-component of the current as prescribed by (2.2.1) (m/s)

Table 2.2.2: Definition of the **cones** experiments.

Test	IADVS[a]	purpose
A	1	upwind scheme
B	2	Lax-Wendroff scheme
C	3	TVD scheme using the superbee limiting function
D	4	TVD scheme using the monotonic limiting function

[a]Switch to select the type of advection scheme for scalars.

Table 2.2.3: Output values for the parameters of the test case **cones**.

parameter	exact	time (T)	A[a]	B	C	D
XMIN	5	0.5	8.000	6.000	6.000	7.000
		1.0	*	8.000	6.000	8.000
		1.5	12.00	7.000	6.000	7.000
		2.0	*	9.000	6.000	8.000
XPLUS	5	0.5	*	7.000	6.000	7.000
		1.0	10.00	7.000	6.000	7.000
		1.5	*	9.000	6.000	8.000
		2.0	13.00	7.000	6.000	8.000
YMIN	5	0.5	12.00	7.000	6.000	8.000
		1.0	15.00	9.000	6.000	8.000
		1.5	*	8.000	6.000	9.000
		2.0	20.00	11.00	7.000	9.000
YPLUS	5	0.5	13.00	8.000	6.000	7.000
		1.0	15.00	8.000	6.000	9.000
		1.5	18.00	10.00	6.000	9.000
		2.0	*	9.000	6.000	10.00
CMIN	0	0.5	0.0	-0.1226	0.0	0.0
		1.0	0.0	-0.1758	0.0	0.0
		1.5	0.0	-0.1955	0.0	0.0
		2.0	0.0	-0.2021	0.0	0.0
CMAX	1	0.5	0.2092	0.7831	0.6993	0.5589
		1.0	0.1162	0.6772	0.6467	0.4474
		1.5	0.0808	0.5958	0.6111	0.3820
		2.0	0.0618	0.5277	0.5837	0.3379

[a] A “*” indicates that the radius is larger than the distance of the cone centre to the corresponding basin boundary.

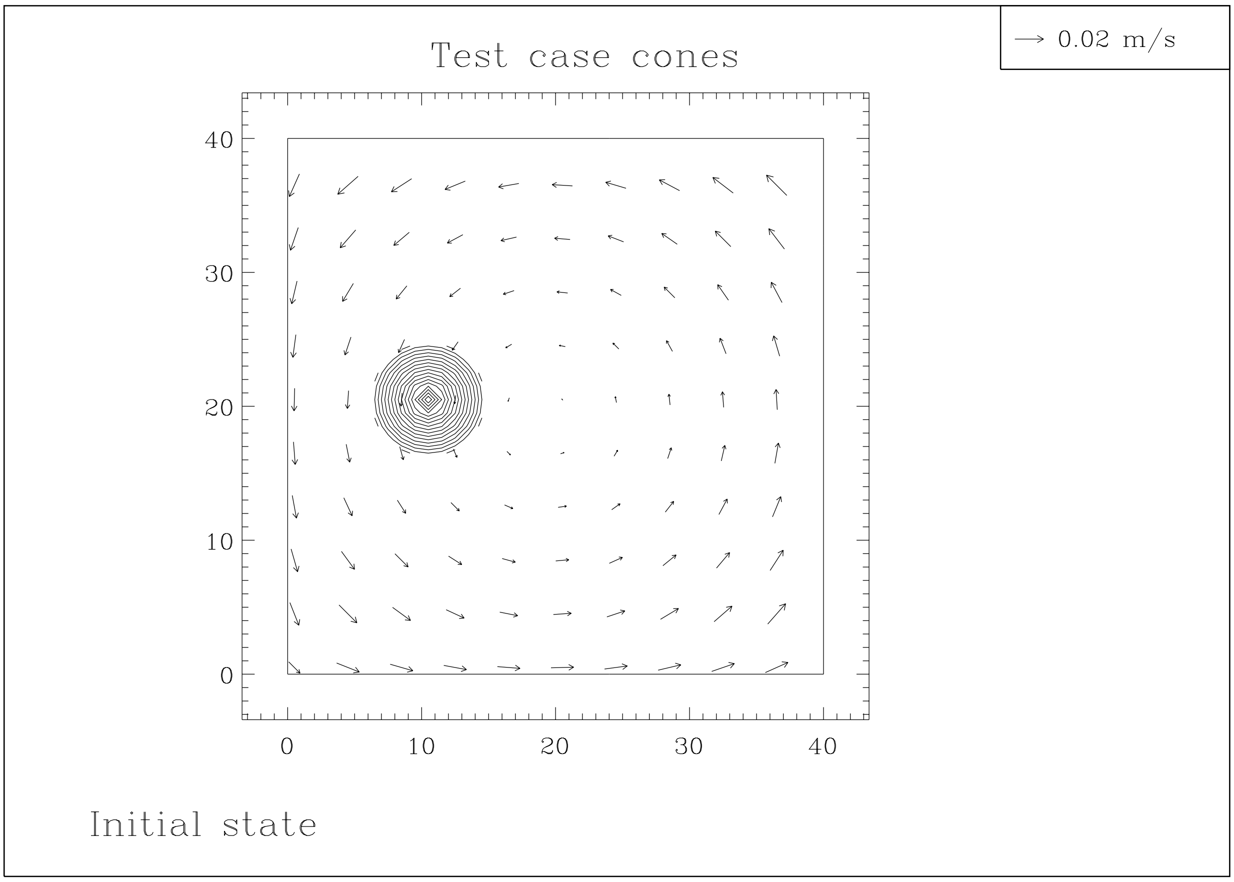


Figure 2.2.1: The initial configuration of the **cones** test case. Contour lines are at 0.05 intervals.

2.1.3 Results

- It is clear that scheme C which is also the default scheme of the program, gives the best performance both in preserving the original shape as in maintaining the gradient of the contaminant concentration.
- Scheme D produces a more asymmetric distribution and less sharp gradients.
- The upwind scheme A is only first order accurate and therefore highly diffusive. This is clearly observed in Figure 2.2.2a where the original distribution has been smeared out over the entire basin.
- Although scheme B can preserve sharp gradients, the original shape is highly distorted. A further problem is the non-preservation of extremal values yielding even negative concentrations (see Table 2.2.3).

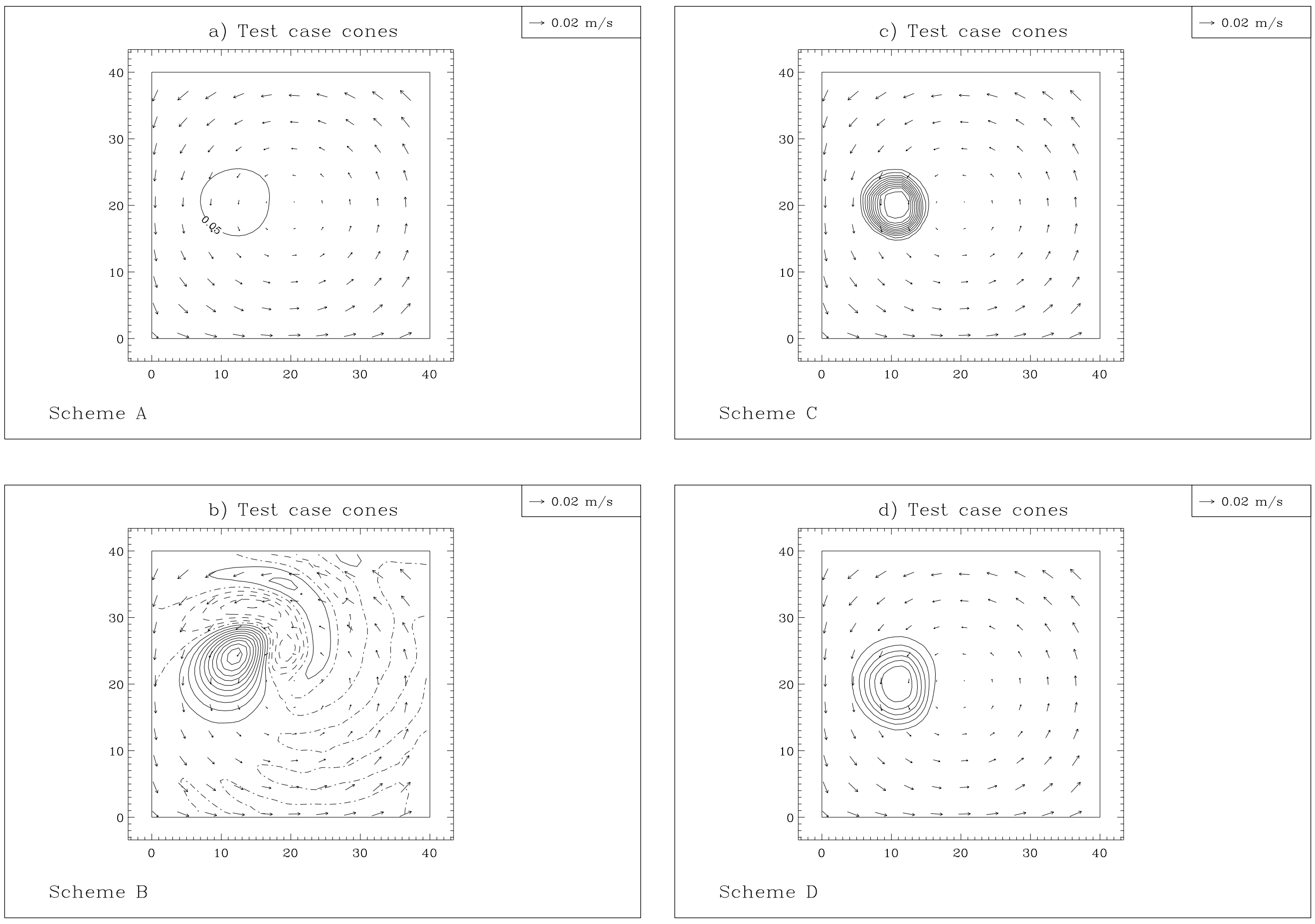
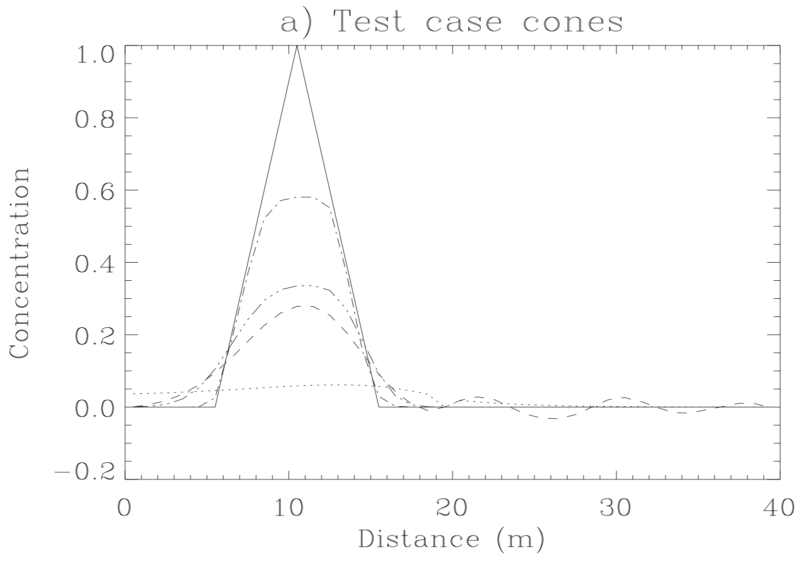


Figure 2.2.2: Surface concentrations after two periods of revolution for scheme A (a), B (b), C (c), D (d). Contour lines are at 0.05 intervals. Dashed curves indicate negative values.



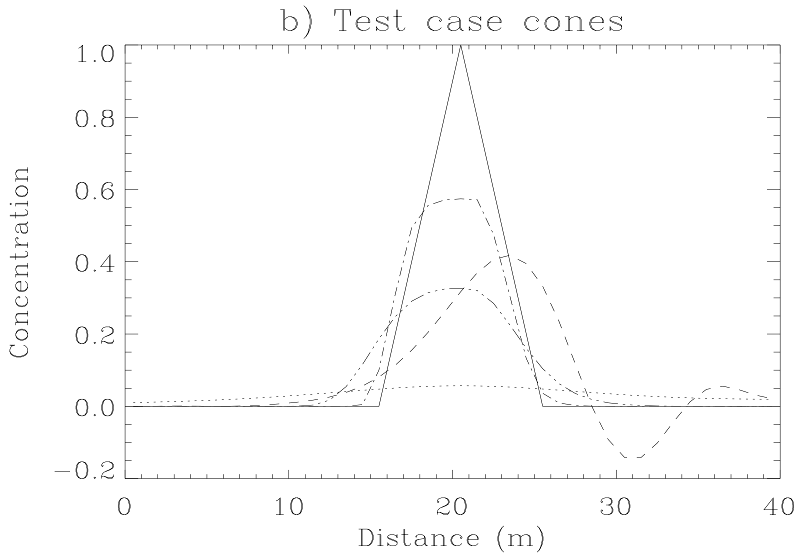


Figure 2.2.3: Surface concentrations after two periods of revolution along a transect through the cone centre parallel to the X-axis (a) and Y-axis (b): solid (exact case), dots (scheme A), dashes (scheme B), dash-dots (scheme C), dash plus 2 dots (scheme D).

2.2 Test case front

2.2.1 Description of the problem and model setup

This test case has been designed to verify the horizontal and vertical schemes for the advection of scalar quantities. The initial configuration, shown in Figure 2.2.4a, consists of a non-rotating channel with open boundaries at the two ends. The water depth is determined by a piecewise linear profile of the form

$$\begin{array}{lll} H = 50 & \text{for} & 0 < x_1 < 25 \\ H = 150 - 4x_1 & \text{for} & 25 \leq x_1 \leq 30 \\ H = 30 & \text{for} & x_1 > 30 \end{array} \tag{2.2.4}$$

where H is the water depth in m and x_1 the along-channel distance in km. An initial two-layer distribution is specified by

$$\begin{array}{lll} C = 2 & \text{if} & z_s < 20\ ,\ 0 < x_1 < 25 \\ C = 1 & \text{if} & x_1 \geq 25 \\ C = 0 & \text{if} & z_s \geq 20\ ,\ 0 < x_1 < 25 \end{array} \tag{2.2.5}$$

where z_s is the distance to the surface in m and C the contaminant concentration in arbitrary units. During the simulation the lateral and horizontal front are advected by an imposed tidal current with a semi-diurnal period:

$$u = u_a(x_1, z_s)\cos(2\pi t/T) \tag{2.2.6}$$

where $T = 12$ h. The amplitude u_a in m/s is prescribed by

$$\begin{array}{lll} u_a = 2 & \text{if} & z_s + \Delta x_3/2 < 30 \\ u_a = 0 & \text{if} & z_s - \Delta x_3/2 > 30 \\ u_a = 2(30 - z_s + \Delta x_3/2)/\Delta x_3 & \text{otherwise} & \end{array} \tag{2.2.7}$$

where Δx_3 is the vertical grid spacing. The model uses a σ-coordinate system in the vertical so that the vertical grid spacing Δx_3 varies horizontally in analogy with (2.2.4).

The program only solves two equations: the transport equation for the contaminant concentration without diffusion and the continuity equation for the vertical velocity (see equations (3.3.9), (3.1.46) and (3.1.47)). The latter is non-zero only along the slope where the prescribed velocity field has a non-zero horizontal gradient. The channel length is set to 50 km. All cross-channel variations are neglected. More detailed specifications about the model setup are given in Table 2.2.4[1].

[1]The current field, prescribed by (2.2.7), is updated at each time step in a modified version of the program module CRRNT3P (file *crrnt3.f* in the **PROJECT** directory).

2.2.2 Experiments and output parameters

In analogy with the **cones** test the **front** test case is an idealised problem enabling to intercompare the different advection schemes implemented in the program using simple forcing and initial conditions. The same four tests (see Table 2.2.5) have been set up:

A : upwind scheme

B : Lax-Wendroff scheme

C : TVD scheme using the superbee limiting function

D : TVD scheme using the monotonic limiting function

The simulations are performed for 36 hours, i.e. 3 semi-diurnal tidal cycles. The contaminant distributions and current field are displayed in Figures 2.2.4a–f at the initial time and at 3 h intervals during the last cycle for scheme C which is also the default scheme of the program. The distributions after 27 hours are compared for the four experiments in Figures 2.2.5a–d. Finally, the concentrations at 5 m below the surface and a vertical profile at 30 km from the left boundary are shown in Figures 2.2.6a and b after 27 hours of simulation.

The following output test parameters are defined.

GRADH
: The maximum horizontal gradient (km^{-1}) along an horizontal transect taken at 5 m depth below the surface.

GRADV
: The maximum vertical gradient (m^{-1}) along a moving vertical transect taken at the left edge of the lateral front.

HLENG
: The width of the lateral front measured in km at 5 m below the surface. This is defined as the horizontal width of the area where the horizontal gradient is larger than 0.01 km^{-1}.

CMIN
: Minimum concentration along the moving vertical transect.

CMAX
: Maximum concentration along the moving vertical transect.

Values for the exact case and the four experiments are given in Table 2.2.6.

Table 2.2.4: Model setup for the test case **front**.

parameter	value	purpose
param.inc		
NC	50	number of grid points in the X-direction
NR	2	number of grid points in the Y-direction
NZ	20	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NOBV	0	number of open boundary points in the Y-direction
NVPROF	1	number of profiles at open boundaries
NCON	1	number of tidal frequencies
NCONC	1	number of contaminants
DEFCON		
IOPTC	1	contaminant module switched on
IADVC	0	advection of momentum disabled
IOPTK	0	vertical diffusion coefficients set to a uniform (zero) value
IOPT2	0	mode splitting disabled
IOPTD	0	uniform density
IBSTR	0	bottom stress set to zero
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01021200	end date (MMDDHHMM)
DELT	180	time step (s)
VISMOL	0.0	vertical diffusion coefficient for momentum (m^2/s)
DIFMOL	0.0	vertical diffusion coefficient for scalars (m^2/s)
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the horizontal grid (m)
GX0(NC+1)	50000	X-coordinate of the upper right corner of the horizontal grid (m)
GY0(1)	0	Y-coordinate of the lower left corner of the horizontal grid (m)
GY0(NR+1)	2000	Y-coordinate of the upper right corner of the horizontal grid (m)
DEP(NR,NC)		depth array as specified by (2.2.4) (m)

Table 2.2.4: continued.

parameter	value	purpose
DEFOB2D		
SIGMA(1)	1.4544E-04	frequency of the semi-diurnal tide (rad/s)
DEFOB3D		
IVPOBU(NOBU)	1	profile number used at X-open boundaries (zero gradient condition for contaminants)
DEFICS		
CONCN(NZ,NR,NC,1)		initial contaminant concentration as prescribed by (2.2.5)

Table 2.2.5: Definition of the **front** experiments.

Test	IADVS[a]	purpose
A	1	upwind scheme
B	2	Lax-Wendroff scheme
C	3	TVD scheme using the superbee limiting function
D	4	TVD scheme using the monotonic limiting function

[a]Switch to select the type of advection scheme for scalars.

Table 2.2.6: Output values for the parameters of the test case **front**.

parameter	time (h)	exact	A	B	C	D
GRADH	24	1	0.0855	0.2231	0.3167	0.1710
	27	1	0.0511	0.2263	0.2960	0.1621
	30	1	0.1001	0.2043	0.3069	0.1583
	33	1	0.0362	0.1955	0.3070	0.1549
	36	1	0.0891	0.2010	0.3080	0.1542
GRADV	24	0.8	0.1091	0.7014	0.2941	0.1892
	27	0.8	0.1104	0.6123	0.2838	0.1970
	30	0.8	0.1038	0.5310	0.2519	0.1712
	33	0.8	0.0570	0.6774	0.2269	0.1723
	36	0.8	0.0552	0.5433	0.2648	0.1715
HLENG	24	1	27.00	14.00	6.000	12.00
	27	1	26.00	15.00	6.000	13.00
	30	1	29.00	10.00	6.000	12.00
	33	1	24.00	19.00	5.000	13.00
	36	1	26.00	11.00	6.000	13.00
CMIN	24	0	0.5985	-0.0311	0.1292	0.1615
	27	0	0.5957	-0.2704	0.1415	0.3037
	30	0	0.6181	0.1267	0.1610	0.2700
	33	0	0.7702	-0.0112	0.2001	0.2572
	36	0	0.7762	0.1077	0.1475	0.2552
CMAX	24	2	1.870	2.017	1.997	1.995
	27	2	1.871	2.514	1.998	1.996
	30	2	1.868	2.027	1.998	1.990
	33	2	1.793	2.018	1.998	1.994
	36	2	1.786	2.041	1.996	1.991

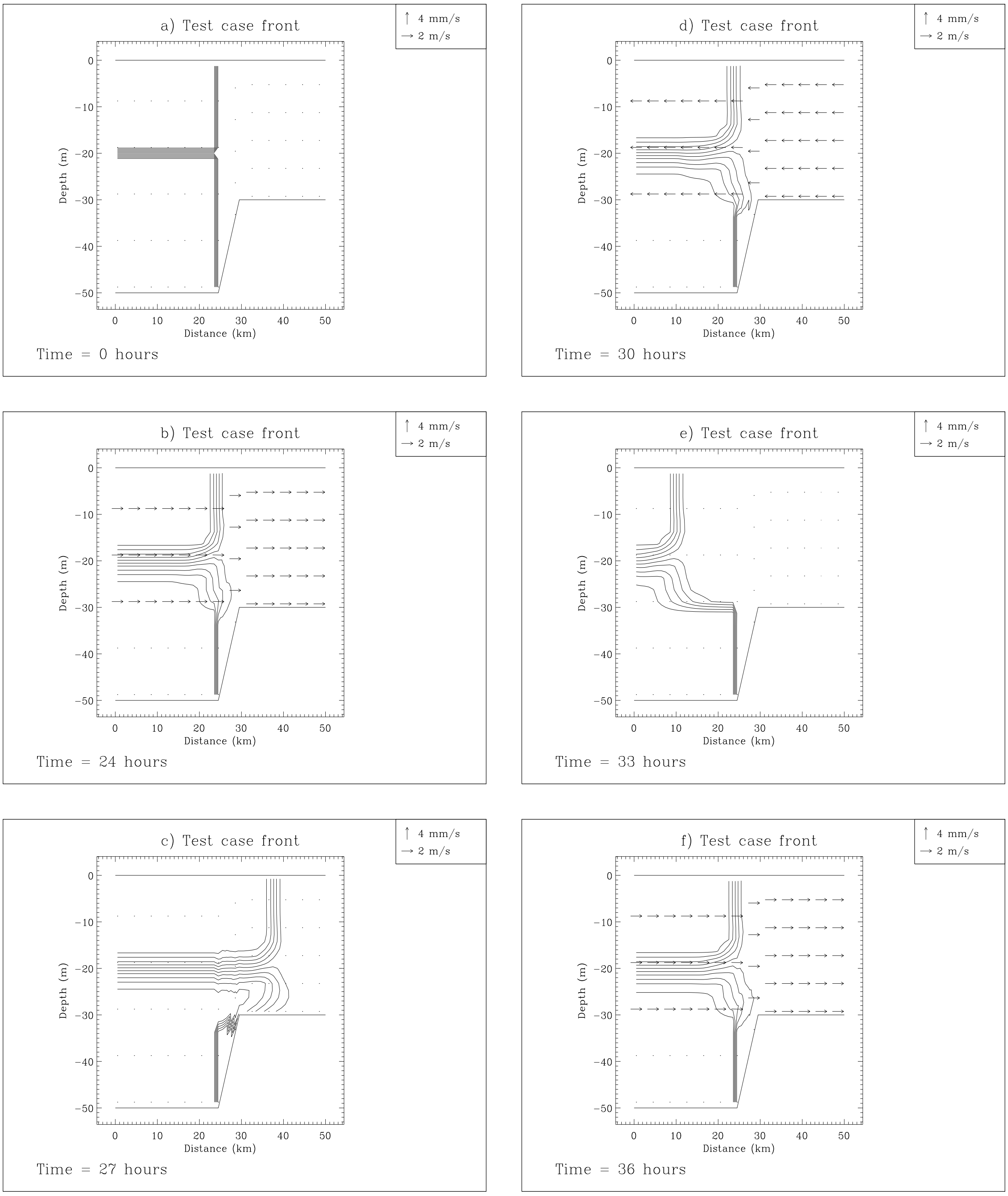


Figure 2.2.4: Contaminant distribution and current field for experiment C at $t = 0$ h (a), $t = 24$ h (b), $t = 27$ h (c), $t = 30$ h (d), $t = 33$ h (e), $t = 36$ h (f). Contour lines are at 0.2 intervals.

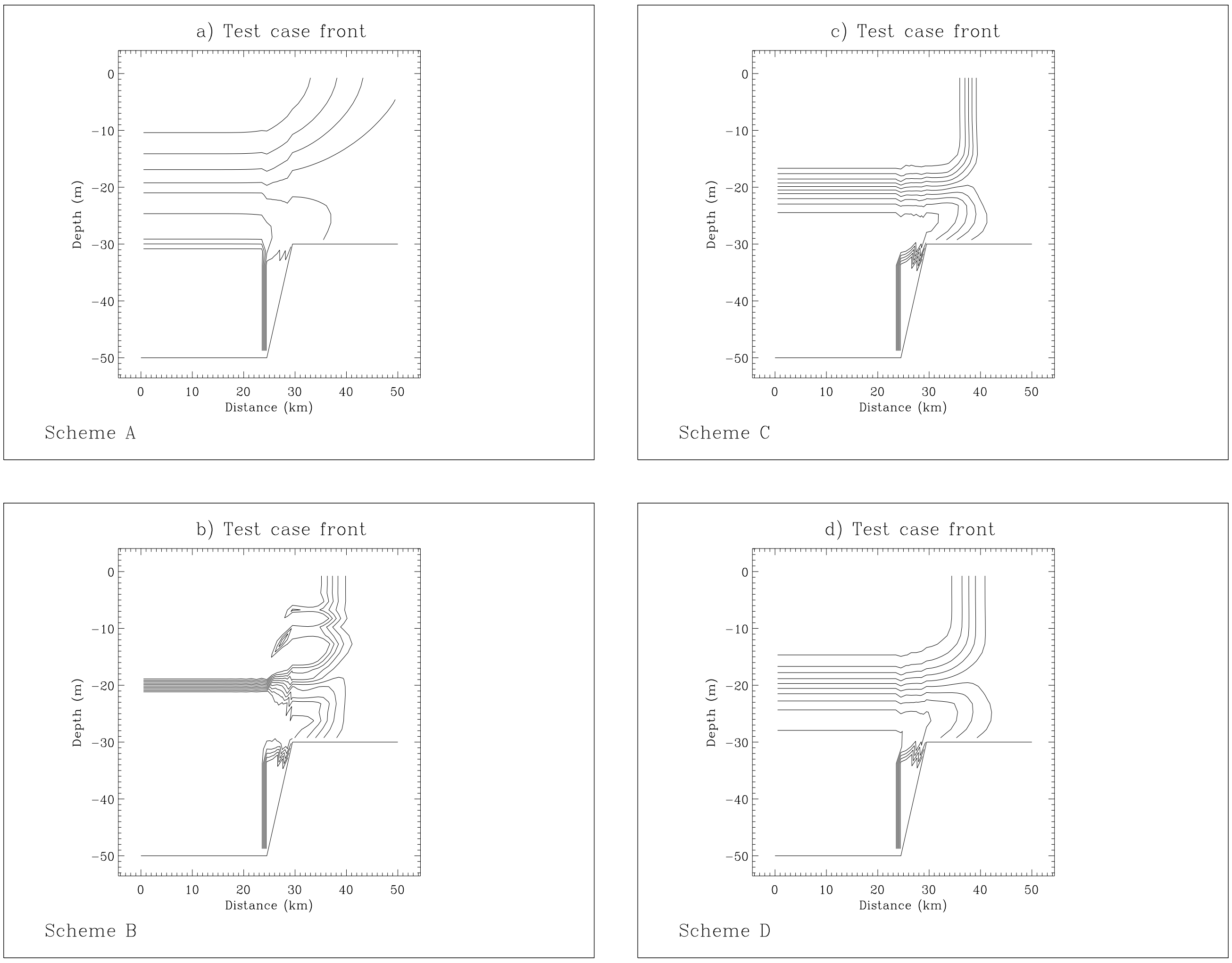


Figure 2.2.5: Contaminant concentration after 27 hours for scheme A (a), B (b), C (c), D (d). Contour lines are at 0.2 intervals.

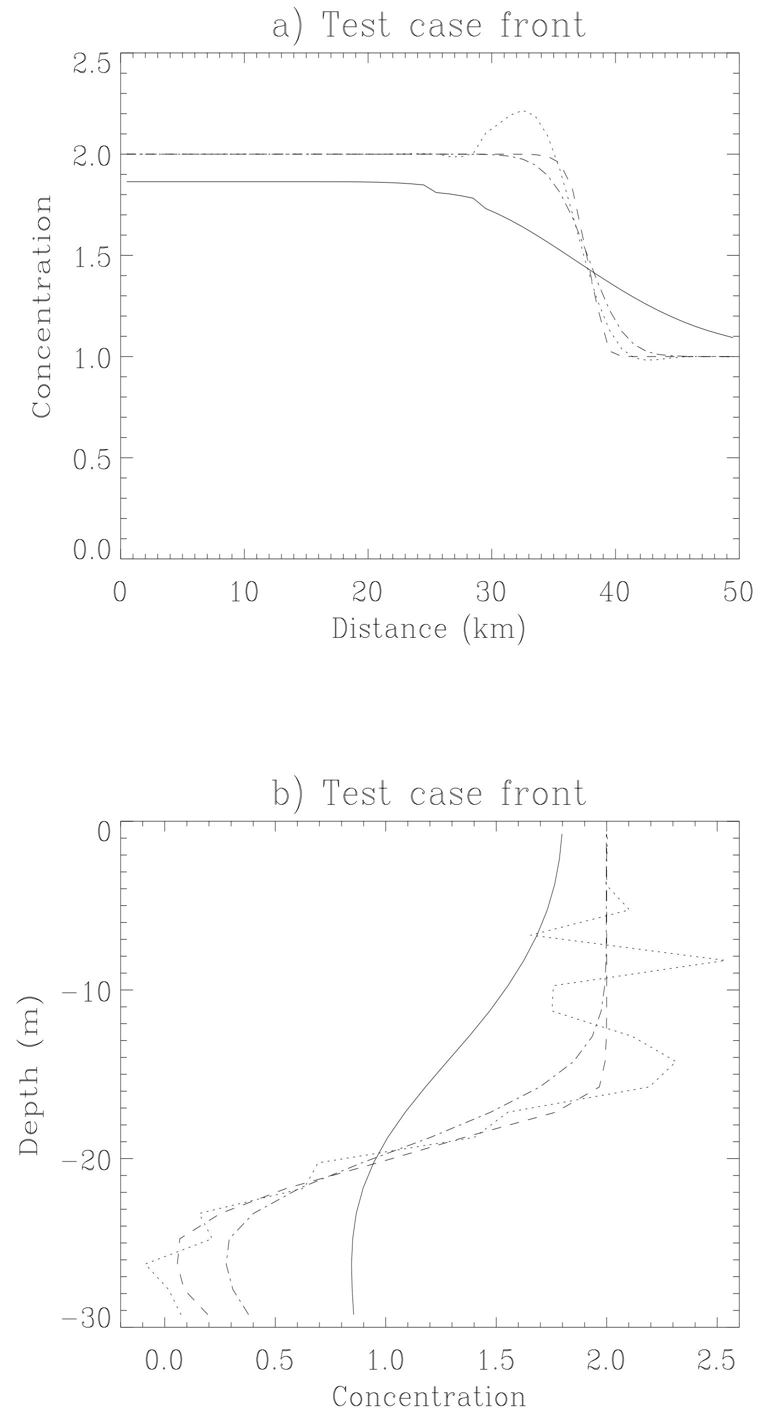


Figure 2.2.6: Contaminant concentration after 27 hours for scheme A (solid), B (dots), C (dashes), D (dash-dots): horizontal profile at 5 m below the surface (a), vertical profile at 30 km from the left boundary (b).

2.2.3 Results

- After an initial broadening due to numerical diffusion, the shape of the fronts remains unchanged after a tidal cycle as can be seen by comparing Figures 2.2.4b and f.
- Table 2.2.6 shows that the default scheme C not only produces the sharpest (lateral) front (parameter GRADH) but also the smallest frontal width (parameter HLENG).
- The importance of the type of advection scheme for the simulation of frontal systems can be further observed in Figures 2.2.5a–d. The upwind scheme which is only first order accurate, is obviously the most diffusive. Schemes C and D are similar although the latter is somewhat more diffusive. The sharpest horizontal front is produced by the second order Lax-Wendroff scheme. A clear disadvantage is that this scheme does not preserve monotonicity resulting in spurious over- and undershooting along the lateral front (Figure 2.2.5b) and negative concentrations (Table 2.2.6). The same features can also be inferred from Figure 2.2.6a and b. The results are clearly in agreement with the previous **cones** test.

2.3 Test case seich

2.3.1 Description of the problem and model setup

The previous two problems served to test the advection schemes for scalar quantities. They are idealistic in the sense that the velocity is not updated by the program but prescribed externally. A more realistic test case is the **seich** problem where a two-dimensional circulation along a vertical plane is generated through an horizontal pressure gradient. The main aim is to verify the program's circulation module.

A non-rotating channel, 30 km long, with a uniform depth of 20 m is considered bounded by a solid lateral wall at the two ends. All cross-channel variations are neglected. The initial surface elevation is set to zero. A salinity stratification is specified initially in a two-layer form, as depicted in Figure 2.2.7 with a fresh water value of $S_1 = 25$ PSU in the upper and a sea water value of $S_2 = 35$ PSU in the lower layer. The interface is located at a depth of 7.5 m in the left and 12.5 m in the right portion of the channel. In the middle of the channel the interface decreases with a uniform slope of 0.0025 m^{-1}. The upper layer density is related to salinity via a linear equation of state of the form

$$\rho_1 = \rho_2(1 + \beta_S(S_1 - S_2)) \tag{2.2.8}$$

where ρ_1, ρ_2 are the densities in the upper, respectively lower layer and β_S is the uniform salinity expansion coefficient. The resulting horizontal (baroclinic) pressure gradient creates, in the absence of vertical and horizontal diffusion, an horizontal motion through a balance with the inertial force in the momentum equation and a vertical current via the continuity equation. The program solves all momentum (2-D and 3-D) equations, the equation of continuity (2-D and 3-D) and the transport equation for salinity. For more details see Section III-1.1.1b. The setup of model parameters for this test case is given in Table 2.2.7.

2.3.2 Analytical solution

The problem can be solved analytically provided that all non-linear terms are neglected in the momentum and continuity equation. This assumption is validated by the numerical solutions presented below. The general solution consists of a combination of four modes: two internal (baroclinic) modes propagating in opposite along-channel directions with a wave speed $c_i = 2.2$ km/h and two external (barotropic) modes propagating in the same opposite directions with a larger speed $c_e = 50$ km/h. Since the initial surface elevation is set to zero, the latter modes only have a negligible impact. As a consequence the slope front in the middle of the channel splits in two equal parts. The upper (lower) part moves with speed $-c_i$ (c_i) to the left (right). Vertical motions are generated above and below the slope front which are downwards directed along the leftward moving and upward along the rightward moving front. The circulation is closed between the two fronts by an horizontal current towards the left in the upper and towards the right in the lower layer. This evolution is illustrated in Figures 2.2.8a–c showing the circulation and the front after 1, 3, and 6 hours. The frontal system reaches the boundaries of the domain after nearly 7 hours.

2.3.3 Experiments and output parameters

To test firstly the influence of the advection of momentum and secondly the type of advection scheme the following four experiments have been defined (Table 2.2.8).

A : advection of momentum switched off, TVD scheme for salinity

B : upwind scheme for momentum, TVD scheme for salinity

C : TVD scheme for momentum and salinity

D : TVD scheme for momentum, upwind scheme for salinity

Experiment C uses the default scheme implemented in the program. The simulations are performed for a period of 7 hours. The results obtained with scheme C are compared with the analytical solution in Figures 2.2.8a–f. The four experiments are compared in Figures 2.2.9a–d showing the distributions after 4 hours.

The following output parameters are defined.

HLEFT
: The distance (km) of the leftward moving front with respect to the centre of the channel. The position of the front is taken at the location where the vertical current in the upper layer attains its largest negative value.

HRIGHT
: The distance (km) of the rightward moving front with respect to the centre of the channel. The position of the front is taken at the location where the vertical current in the lower layer attains its largest positive value.

V2UP
: The value of the horizontal current (m/s) in the upper layer averaged between the two fronts.

V2LO
: The value of the horizontal current (m/s) in the lower layer averaged between the two fronts.

WMAXUP
: Maximum value of the vertical current (mm/s) in the upper layer.

WMAXLO
: Maximum value of the vertical current (mm/s) in the lower layer.

WMINUP
: Minimum value of the vertical current (mm/s) in the upper layer.

WMINLO
: Minimum value of the vertical current (mm/s) in the lower layer.

SDEV
: The relative difference between the volume averaged salinity and its initial value, defined by

$$\mathsf{SDEV} = 10^7(\langle S(t)\rangle - \langle S(0)\rangle)/\langle S(0)\rangle \tag{2.2.9}$$

where a $\langle\ \rangle$ denotes the averaged value over the whole width and depth of the channel. Since the salinity fluxes are zero at the surface, the bottom and along the solid side walls, the exact value of SDEV is zero. Since the numerical schemes for the advection of salinity are conservative, non-zero values are only produced by rounding errors. This parameter is therefore useful to test the machine accuracy.

In the definition of the above parameters the upper and lower layer are conveniently taken at respectively 5.5 below the surface and 4.5 above the bottom. Values for the the analytical solution and the four experiments are listed in Table 2.2.9 at hourly intervals.

2.3.4 Results

- It should be remarked first that contrary to the numerical setup the analytical derivation assumes that the channel has open ends allowing the perturbation to propagate towards infinity. This explains the small perturbation in the current field outside the fronts in Figure 2.2.8d caused by the barotropic mode which already attains the edges

Table 2.2.7: Model setup for the test case **seich**.

parameter	value	purpose
param.inc		
NC	2	number of grid points in the X-direction
NR	60	number of grid points in the Y-direction
NZ	20	number of grid points in the vertical
NOBU	0	number of open boundary points in the X-direction (a zero value means that the boundaries are closed)
NOBV	0	number of open boundary points in the Y-direction (a zero value means that the boundaries are closed)
DEFCON		
IOPTK	0	vertical diffusion coefficients set to a uniform (zero) value
IOPTSA	1	update of the salinity field enabled
IBSTR	0	bottom stress set to zero
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01010700	end date (MMDDHHMM)
DELT	15	time step for the external (2-D) mode (s)
IC3D	16	time step for the internal (3-D) mode divided by DELT
SREF	35.0	reference salinity (PSU) set equal to the value S_2 of the salinity in the lower layer
VISMOL	0.0	vertical diffusion coefficient for momentum (m^2/s)
DIFMOL	0.0	vertical diffusion coefficient for salinity (m^2/s)
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the horizontal grid (m)
GX0(NC+1)	1000	X-coordinate of the upper right corner of the horizontal grid (m)
GY0(1)	0	Y-coordinate of the lower left corner of the horizontal grid (m)
GY0(NR+1)	30000	Y-coordinate of the upper right corner of the horizotnal grid (m)
DEP(NR,NC)	20	array of uniform water depths (m)
DEFICS		
S(NZ,NR,NC)		initial salinity field given by 25 PSU in the upper and 35 PSU in the lower layer

Table 2.2.8: Definition of the **seich** experiments.

Test	IADVC[a]	IADVS[b]	purpose
A	0	3	no advection of momentum, TVD scheme for salinity
B	1	3	upwind scheme for momentum, TVD scheme for salinity
C	3	3	TVD scheme for momentum and salinity
D	3	1	TVD scheme for momentum, upwind scheme for salinity

[a]Switch to select the type of advection scheme for momentum.
[b]Switch to select the type of advection scheme for scalars.

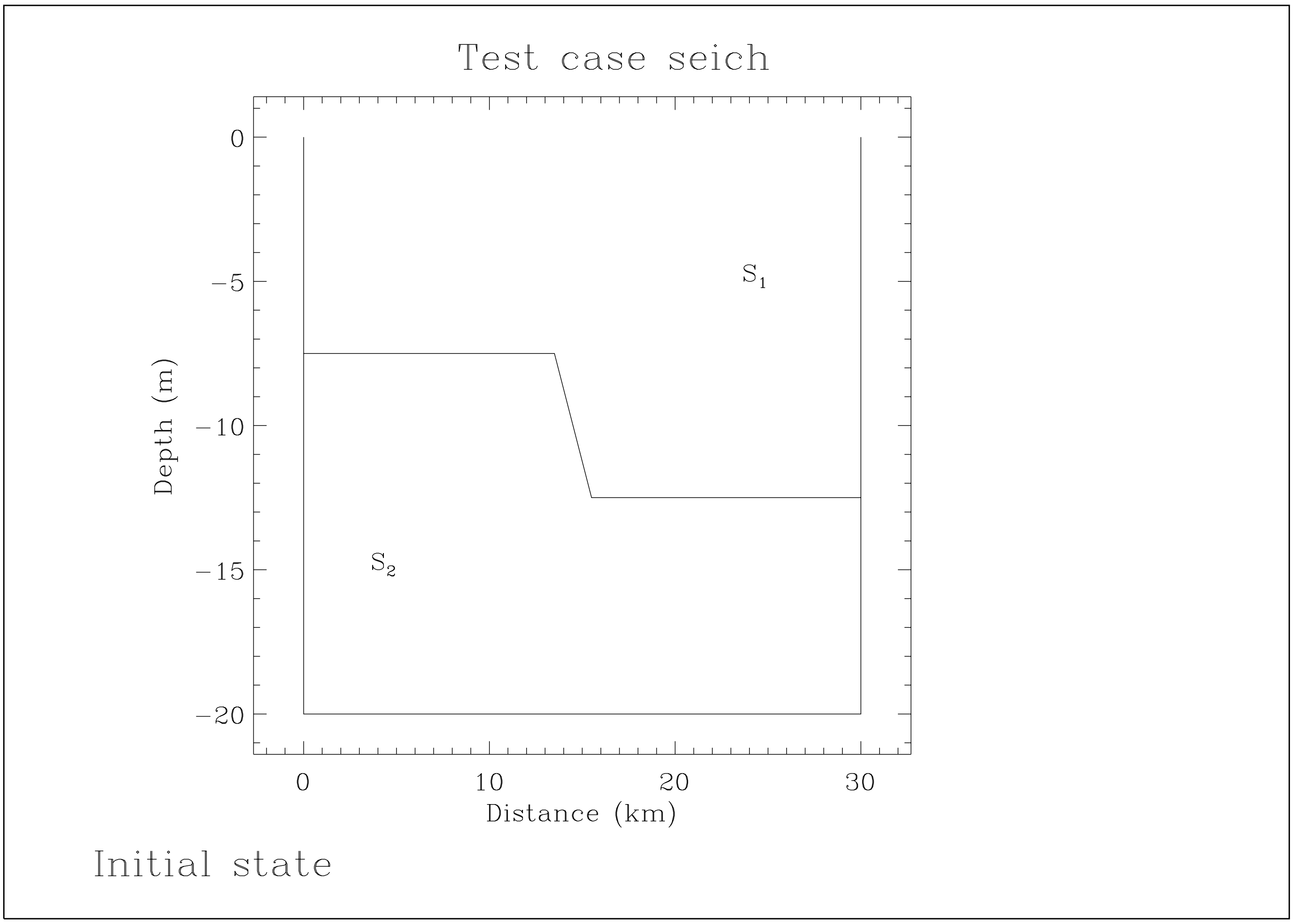


Figure 2.2.7: The initial configuration of the **seich** test case.

Table 2.2.9: Output values for the parameters of the test case **seich**.

parameter	time (h)	analytical	A	B	C	D
HLEFT	1	2.750	2.250	2.250	2.250	2.250
	2	5.250	4.250	4.250	4.250	4.250
	3	7.250	6.250	6.750	6.250	6.250
	4	9.750	8.250	8.750	8.750	8.250
	5	11.75	10.25	10.75	10.75	10.25
	6	13.75	12.25	12.75	12.75	12.75
HRIGHT	1	2.750	1.750	2.250	2.250	2.250
	2	5.250	3.750	4.250	4.250	3.750
	3	7.250	5.750	6.250	6.250	6.250
	4	9.750	7.750	8.250	8.250	8.250
	5	11.75	10.25	10.25	10.25	10.25
	6	13.75	12.25	12.25	12.25	12.25
V2UP	1	-0.1188	-0.1524	-0.1507	-0.1515	-0.1517
	2	-0.1273	-0.1653	-0.1633	-0.1644	-0.1651
	3	-0.1358	-0.1639	-0.1569	-0.1629	-0.1639
	4	-0.1355	-0.1588	-0.1509	-0.1542	-0.1588
	5	-0.1392	-0.1600	-0.1532	-0.1561	-0.1601
	6	-0.1417	-0.1668	-0.1609	-0.1621	-0.1646
V2LO	1	0.1316	0.1215	0.1156	0.1159	0.1160
	2	0.1401	0.1342	0.1299	0.1304	0.1347
	3	0.1486	0.1449	0.1417	0.1422	0.1427
	4	0.1483	0.1548	0.1521	0.1526	0.1533
	5	0.1520	0.1551	0.1541	0.1547	0.1555
	6	0.1545	0.1524	0.1473	0.1477	0.1509
WMAXUP	1	0.4180	0.5875	0.6154	0.6302	0.6719
	2	0.4180	0.5209	0.5235	0.5722	0.5608
	3	0.4180	0.4583	0.4495	0.5165	0.4781
	4	0.4179	0.4207	0.4232	0.4848	0.4357
	5	0.4180	0.3835	0.4044	0.4623	0.4109
	6	0.4180	0.3393	0.3547	0.4178	0.3395

Table 2.2.9: continued.

parameter	time (h)	analytical	A	B	C	D
WMAXLO	1	0.3432	0.5208	0.4809	0.4812	0.4451
	2	0.3433	0.4742	0.3951	0.4092	0.3755
	3	0.3432	0.4395	0.4049	0.4317	0.3587
	4	0.3433	0.3895	0.3458	0.3862	0.3087
	5	0.3433	0.3448	0.3177	0.3590	0.2833
	6	0.3432	0.3261	0.2805	0.3220	0.2533
WMINUP	1	-0.4185	-0.6525	-0.5993	-0.6151	-0.6048
	2	-0.4185	-0.6326	-0.5696	-0.6287	-0.5745
	3	-0.4185	-0.6027	-0.5201	-0.6142	-0.5345
	4	-0.4186	-0.5712	-0.5315	-0.6023	-0.5042
	5	-0.4185	-0.5219	-0.5122	-0.6165	-0.4553
	6	-0.4185	-0.4663	-0.4680	-0.5921	-0.3993
WMINLO	1	-0.3431	-0.3832	-0.4453	-0.4632	-0.4838
	2	-0.3431	-0.3638	-0.4232	-0.4753	-0.4442
	3	-0.3431	-0.3890	-0.4278	-0.5161	-0.4417
	4	-0.3431	-0.3549	-0.3974	-0.4927	-0.4116
	5	-0.3431	-0.3248	-0.3570	-0.4662	-0.3821
	6	-0.3431	-0.2876	-0.3426	-0.4297	-0.3155
SDEV	1	0.0	22.08	5.047	6.940	-4.416
	2	0.0	-11.99	-11.36	-5.047	-21.45
	3	0.0	22.71	-66.87	-27.13	-37.85
	4	0.0	-27.13	7.571	-12.62	6.940
	5	0.0	13.25	-17.03	-7.571	20.82
	6	0.0	-33.44	-17.03	-10.09	13.25

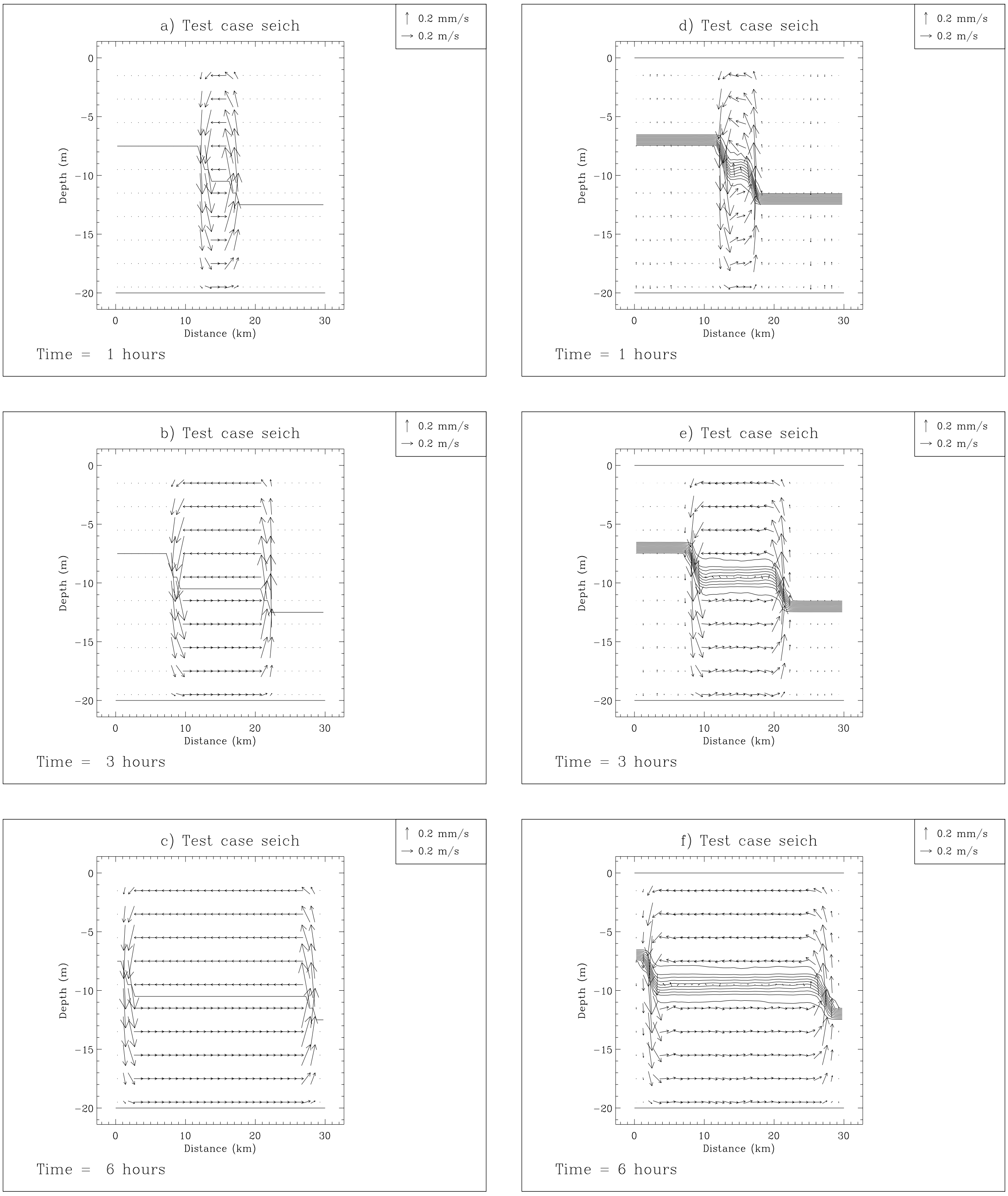


Figure 2.2.8: Evolution of the current field, salinity and the frontal system for the **seich** test case after 1, 3, and 6 hours according to the analytical theory (a, b, c) and experiment C (d, e, f). Contour lines in (d, e, f) are at intervals of 1 PSU.

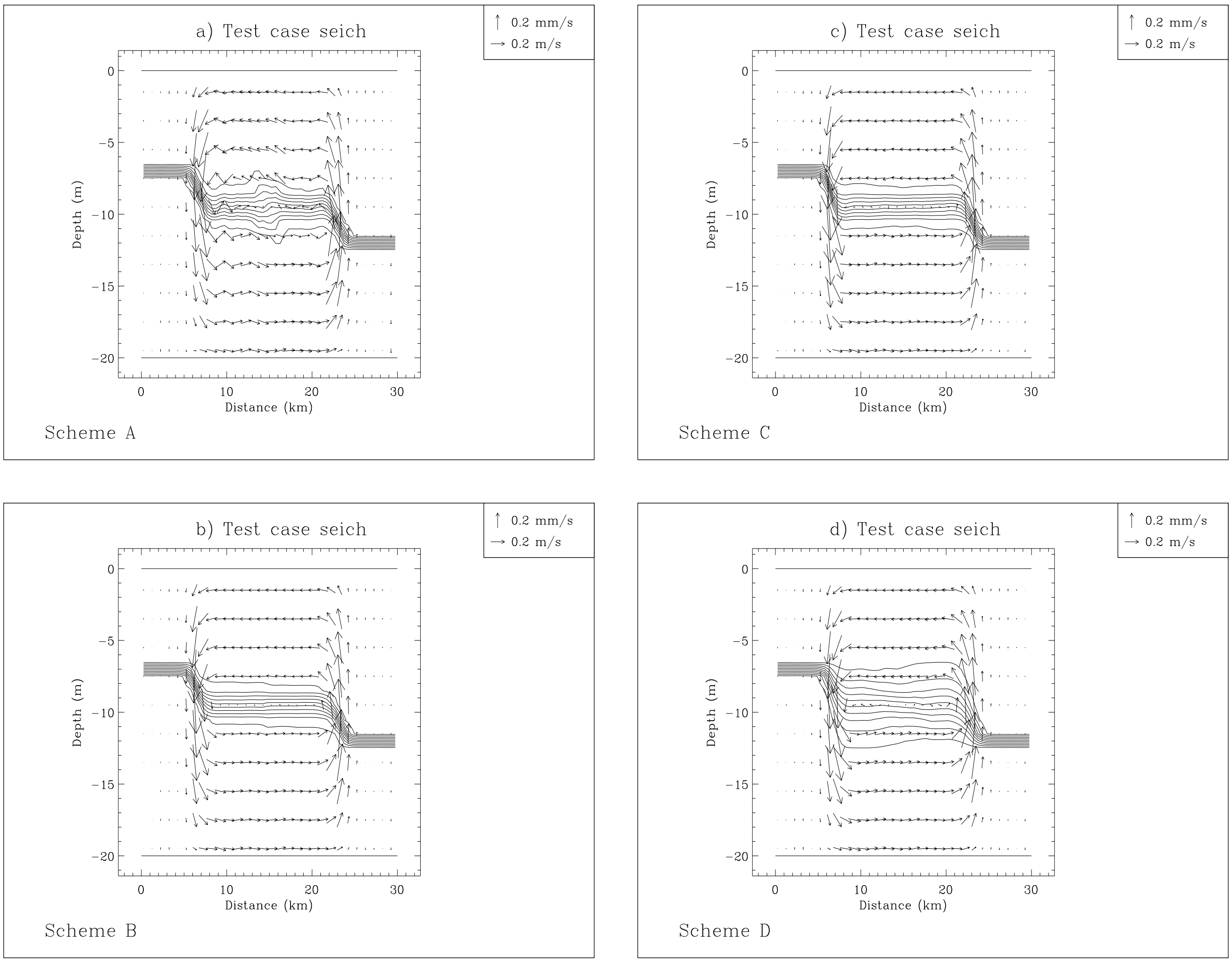


Figure 2.2.9: Current field and salinity distribution for the **seich** test case after 4 hours using scheme A (a), B (b), C (c), D (d). Contour lines are at intervals of 1 PSU.

of the channel after 17 min and bounces off the solid walls. The model appears in good agreement with the analytical theory. A notable difference is the initial occurrence of spurious oscillations of the current field gradually attenuating in time. Note also that the interface which separates the upper and lower layer between the two fronts is spread over 3–4 m (or grid cells). The interface thickness remains however uniform in time.

- The numerical oscillations of the current field are more clearly observed in the case of scheme A (Figure 2.2.9a) which does not advect momentum, whereas they are almost absent in the other runs due to numerical diffusion by the advection scheme. Note also the larger thickness of the interface when the salinity is advected by the upwind scheme.

- Comparing the values of the parameters HLEFT and HRIGHT for the analytical case and the four experiments it is seen that the fronts advance more slowly to the left and the right in the model results. This indicates that the internal wave speed c_i is somewhat underestimated by the model. Note also that the modelled front moving to the left has a larger speed compared to the one moving to the right.

- The magnitude of the horizontal current is overestimated by the model in the upper and mostly underestimated in the lower layer. However, the horizontal kinetic energy has the tendency to increase in all cases. This occurs at the expense of potential energy which decreases while the fronts are moving.

- The vertical velocity is non-zero only above and below the points where the interface has a non-zero gradient. The maximum and minimum values within the upper and lower layer are time-independent in the analytical solution which is not the case in the model results where the magnitude of the vertical current decreases with time both in the upper and the lower layer. This may be due to numerical diffusion.

- All scalar advection schemes are discretised in a conservative form within the program. This means that SDEV should be zero in the absence of rounding errors. Table 2.2.9 shows that the magnitude of this parameter has a maximum value of mostly below 4×10^{-6} which may considered as sufficiently accurate.

2.4 Test case fredy

2.4.1 Description of the problem and model setup

The test case **fredy** is a 3-D problem intended to compare the advection schemes of momentum and scalars and was used previously in the MAST-NOMADS project (Proctor, 1997) as an intercomparison study for shelf sea models. The results of the exercise are described in Tartinville et al. (1998). The initial setup of the problem is based upon an earlier numerical study of James (1996). An analogous series of laboratory experiments has been conducted by Griffiths and Linden (1981) whereby a bottomless cylinder of fresh water is immersed into a rotating tank containing a fluid of higher density. After removal of the cylinder the fresh water spreads radially outwards until a geostrophic quasi-equilibrium state is reached. During the next stage non-axisymmetric instabilities develop in the form of growing vortices at the edge of the cylindrical area. The number of vortices is determined by the growth rate of the most unstable mode which depends on the initial parameters of the experiment. The laboratory results bear resemblance to those obtained by Madec et al. (1991) who investigated numerically the process of deep water formation in the Mediterranean Sea and found similar eddy-like structures.

The setup used in test case **fredy** is almost the same as in Tartinville et al. (1998). A 20 m deep squared basin is considered. Contrary to the previous investigation the domain is closed by four solid boundaries and the basin has a width of 60 km[2] instead of 30 km to minimise the influence of the side walls. The grid resolution is 1 km in the horizontal and 1 m in the vertical. The initial salinity distribution in PSU, shown in Figures 2.2.10a–b, is given by

$$\begin{aligned} S &= 1.1(D/3)^8 + 33.75 \quad \text{if} \quad D \leq 3 \quad \text{and} \quad d \leq 10 \\ &= 34.85 \quad \text{otherwise} \end{aligned} \tag{2.2.10}$$

where D is the distance in km to the axis of the cylinder, located at (14.5,14.5) in Figure 2.2.10a and d denotes the depth in m. The density is determined with the aid of the equation of state

$$\rho = \rho_0(1 + \beta_S(S - S_0)) \tag{2.2.11}$$

The simulated area is located at $52^0 3'$N with a corresponding Coriolis frequency of 1.15×10^{-4} s^{-1}.

The initial current and sea surface elevation are set to zero. The problem is further simplified by neglecting all diffusion terms and setting the surface and bottom stress equal to zero. The simulation is performed for a period of 6 days. The program solves the following equations: the 2-D and 3-D momentum equations, the 2-D and 3-D forms of the continuity equation and the salinity equation. For more details see Section III-1.1.1b. A list of model setup parameters is given in Table 2.2.10.

2.4.2 Experiments and output parameters

The results, presented in Tartinville et al. (1998), clearly show that the type of advection scheme, used for momentum and salinity, has an important influence on the development and structure of the baroclinic eddies. Only the upwind and TVD scheme with the superbee limiter are considered in the present study. The following four experiments, also listed in Table 2.2.11, are defined:

A : upwind scheme for momentum and salinity

[2]For convenience, the outer 15 km of the basin are not displayed in Figures 2.2.10–2.2.13.

B : TVD scheme for momentum, upwind for salinity

C : upwind scheme for momentum, TVD for salinity

D : TVD scheme for momentum and salinity (default schemes of the program)

A series of figures is produced showing the state at the end of the simulation for the four experiments: surface distributions of current and salinity (Figures 2.2.11a–d), surface distributions of the vertical vorticity component (2.2.12a–d), current and salinity along a lateral transect parallel to the X-axis and through the axis of the cylinder (Figures 2.2.13a–d). For convenience, only the upper 10 m are displayed in the latter figures.

In analogy with Tartinville et al. (1998) the volume-integrated kinetic energy, available potential energy and enstrophy are defined by

$$E_{kin} = \frac{1}{2}\rho_0 \int_{V(t)} (u^2 + v^2)\, dV \tag{2.2.12}$$

$$\begin{aligned} A_{pot} &= \int_{V(t)} \rho g x_3\, dV - \rho_0 \int_{V_0} g x_3\, dV_0 \\ &= \frac{1}{2} g\rho_0 \int_{S(t)} \zeta^2\, dS - \rho_0 \int_{V(t)} b x_3\, dV \end{aligned} \tag{2.2.13}$$

$$E_{str} = \int_{V(t)} \Big(\frac{\partial v}{\partial x_1} - \frac{\partial u}{\partial x_2}\Big)^2\, dV \tag{2.2.14}$$

where $V(t)$ is the volume of the basin, V_0 its initial value, $S(t)$ the surface, ζ the surface elevation and $b = -g(\rho - \rho_0)/\rho_0$ the buoyancy. A fourth quantity A1% is defined as the surface area bounded by the contour line where the salinity is 0.01 PSU less than the ambient value of 34.85 PSU. This determined by the number of surface grid cells where the salinity is less than 34.839 PSU multiplied by the area of a surface grid cell. Time series of the four quantities are plotted in Figures 2.2.14.

The following output parameters are defined for this test case:

EKIN
: Volume integrated kinetic energy (10^9J) as defined by (2.2.12).

APOT
: Volume integrated available potential energy (10^9J) as defined by (2.2.13).

ENSTR
: Volume integrated vorticity ($\mathrm{m}^3/\mathrm{s}^2$) as defined by (2.2.14).

A1PT
: Value of A1% ($10^8\mathrm{m}^2$).

SALMIN
: The minimum salinity value (PSU) inside the domain.

SALMAX
: The maximum salinity value (PSU) inside the domain.

SDEV
: In analogy with the test case **seich** this parameter represents the relative difference between the volume averaged salinity and its initial value and is, defined by

$$\mathsf{SDEV} = 10^6(\langle S(t)\rangle - \langle S(0)\rangle)/\langle S(0)\rangle \qquad (2.2.15)$$

where a $\langle\ \rangle$ represents an averaged value over the entire domain. This parameter is useful for testing the machine accuracy.

THETA
: The value of the ratio E_{kin}/A_{pot} averaged over an inertial period, determined by applying an harmonic analysis for the last 3 days (4.7 inertial periods).

Values of the parameters are listed in Table 2.2.12 for the four experiments at daily intervals (except THETA).

2.4.3 Results

The following conclusions can be made:

- A four-lobed structure is obtained. This is clearly observed in Figure 2.2.11d for scheme D and in a much weaker form for the other experiments.
- In analogy with the experimental results of Griffiths and Linden (1981) four cyclonic vortex rings develop at the points where the unstable displacements are directed inwards (Figures 2.2.12). The vortices are separated by areas with a weaker anticyclonic vorticity. The phenomenon is again most clearly observed for scheme D. This shows the ability of the TVD scheme to simulate highly sheared density fronts.
- The main circulation pattern, as seen from Figures 2.2.13, consists of an outward (inward) displacement of the central area above (below) the interface, upwelling inside and below the fresh water patch with a maximum at the edge of core region and downwelling motions a few km outside the front. Note also the different depths of the fresh water layer in the four experiments. In the case of schemes C and D the interface depth decreases from 10 to 5–6 m giving a much shallower surface layer compared to A and B.
- During the first hours of the simulation the kinetic energy increases at the expense of potential energy representing the outward expansion of the core region (Figures 2.2.14a–b). As soon as a quasi-geostrophic balance is achieved and the eddies start

to grow, both E_{kin} and A_{pot} oscillate with the inertial period and opposite phases. This quasi-equilibrium state is most clearly attained for experiment D, whereas A_{pot} gradually increases in the case of experiments A and B. Note also that the vorticity, measured by the parameter E_{str}, is stronger in experiments B and D which use the TVD scheme for the advection of momentum (Figure 2.2.14c). On the other hand, the surface area of the patch increases more rapidly for schemes A and B (Figure 2.2.14d). This is due to the larger numerical diffusion produced by the upwind scheme for salinity.

- According to Tartinville et al. (1998) the form and growth of the eddies is related to the ratio Θ of the kinetic to the available potential energy (represented here by the output parameter THETA). Firstly, the number of baroclinic eddies given by the wavenumber n scales as $n \sim \Theta^{-1/2}$. Simulations using the PPM advection scheme both for salinity and currents (James, 1996) yield a value of Θ of $\sim$0.5 and a 2-lobed instability. This is higher than the largest value obtained with scheme D in which case a 4-lobed instability is found. Secondly, the growth rate of the instability increases with $\Theta^{1/2}$. This is, at least qualitatively, in agreement with the present results, as can be seen by comparing the development of the eddies in Figures 2.2.11 and 2.2.12 for experiments A, B, C and D which corresponds to increasing values of Θ.

Table 2.2.10: Model setup for test case **fredy**.

parameter	value	purpose
param.inc		
NC	60	number of grid points in the X-direction
NR	60	number of grid points in the Y-direction
NZ	20	number of grid points in the vertical
DEFCON		
IOPTK	0	vertical diffusion coefficients set to a uniform (zero) value
IOPTSA	1	update of the salinity field enabled
IBSTR	0	bottom stress set to zero
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01070000	end date (MMDDHHMM)
DELT	30	time step for the external (2-D) mode (s)
IC3D	10	time step for the internal (3-D) mode divided by DELT
ICRFORM	'U'	file with initial/final conditions written in binary format
DLAREF	52.05	reference latitude (degrees)
R0REF	1025.858	reference density ρ_0 (kg/m^3)
SREF	34.85	reference salinity S_0 (PSU)
SBETUN	7.6×10^{-4}	uniform expansion coefficient β_S for salinity (PSU^{-1})
VISMOL	0.0	vertical diffusion coefficient for momentum (m^2/s)
DIFMOL	0.0	vertical diffusion coefficient for salinity (m^2/s)
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the basin (m)
GX0(NC+1)	60000	X-coordinate of the upper right corner of the basin (m)
GY0(1)	0	Y-coordinate of the lower left corner of the basin (m)
GY0(NR+1)	60000	Y-coordinate of the upper right corner of the basin (m)
DEP(NR,NC)	20	array of uniform water depths (m)
DEFICS		
S(NZ,NR,NC)		initial salinity field (PSU) as prescribed by (2.2.10)

Table 2.2.11: Definition of the **fredy** experiments.

Test	IADVC[a]	IADVS[b]	purpose
A	1	1	upwind scheme for momentum and salinity
B	3	1	TVD scheme for momentum, upwind for salinity
C	1	3	upwind scheme for momentum, TVD for salinity
D	3	3	TVD scheme for momentum and salinity

[a]Switch to select the type of advection scheme for momentum.

[b]Switch to select the type of advection scheme for scalars.

Table 2.2.12: Output values for the parameters of the test case **fredy**.

parameter	time (days)	A	B	C	D
EKIN	1	1.417	2.617	1.313	2.702
	2	0.2990	0.9868	0.3804	0.8235
	3	0.5139	1.105	0.5116	1.472
	4	0.2896	0.8445	0.3497	0.9046
	5	0.2929	0.7763	0.3185	1.163
	6	0.3034	0.7820	0.3376	0.9836
APOT	1	5.263	5.968	3.119	3.401
	2	6.369	7.771	3.428	4.609
	3	6.229	7.807	2.993	3.677
	4	6.515	8.189	2.943	4.021
	5	6.566	8.378	2.797	3.571
	6	6.596	8.471	2.619	3.531
ENSTR	1	0.7366	1.364	0.7621	1.575
	2	0.2386	0.7680	0.3913	0.9802
	3	0.2243	0.5632	0.2802	0.8763
	4	0.1589	0.5279	0.2546	0.7968
	5	0.1554	0.5009	0.2135	0.9255
	6	0.1489	0.5254	0.2183	0.8663
A1PT	1	1.810	1.890	1.490	1.610
	2	2.090	2.210	1.530	1.690
	3	2.450	2.650	1.730	1.840
	4	2.610	3.010	1.770	1.810
	5	2.890	3.130	1.810	1.880
	6	3.010	3.250	1.850	2.050

Table 2.2.12: continued.

parameter	time (days)	A	B	C	D
SALMIN	1	33.91	34.09	33.75	33.75
	2	34.05	34.23	33.75	33.75
	3	34.11	34.31	33.75	33.75
	4	34.17	34.36	33.75	33.76
	5	34.22	34.39	33.75	33.76
	6	34.25	34.42	33.75	33.76
SALMAX	1	34.85	34.85	34.85	34.85
	2	34.85	34.85	34.85	34.85
	3	34.85	34.85	34.85	34.85
	4	34.85	34.85	34.85	34.85
	5	34.85	34.85	34.85	34.85
	6	34.85	34.85	34.85	34.85
SDEV	1	-6.240	-7.444	-1.204	-2.627
	2	-9.415	-11.93	-3.065	-2.956
	3	-10.29	-15.55	-3.065	-4.051
	4	-11.82	-17.95	-2.408	-4.270
	5	-12.37	-18.83	-2.518	-3.394
	6	-14.23	-20.47	-2.627	-4.489
THETA		0.0458	0.0957	0.1206	0.2948

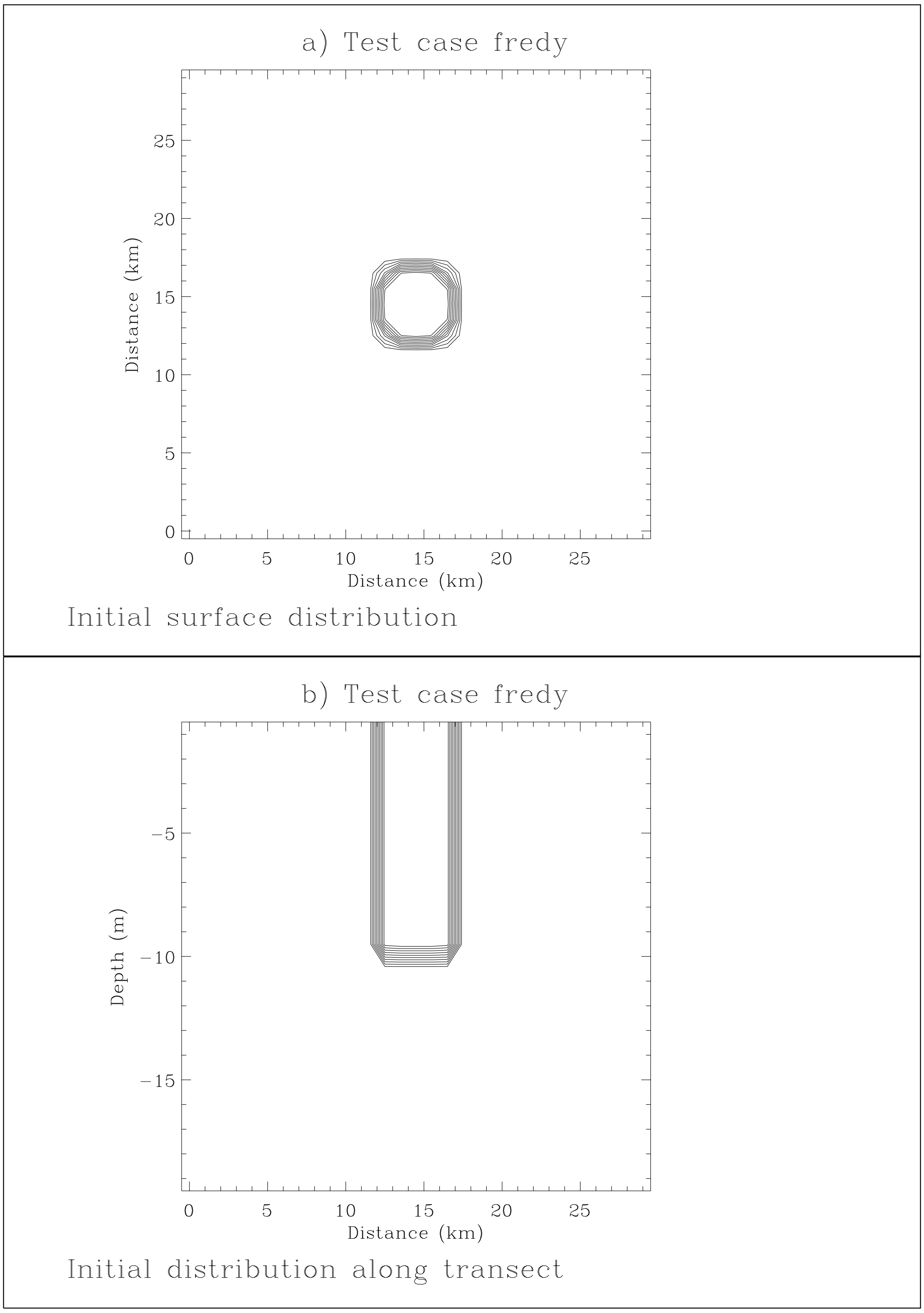


Figure 2.2.10: Initial salinity distribution for the **fredy** test case at the surface (a) and along a transect parallel to the X-axis and through the centre of the patch (b). Contour lines are at intervals of 0.1 PSU.

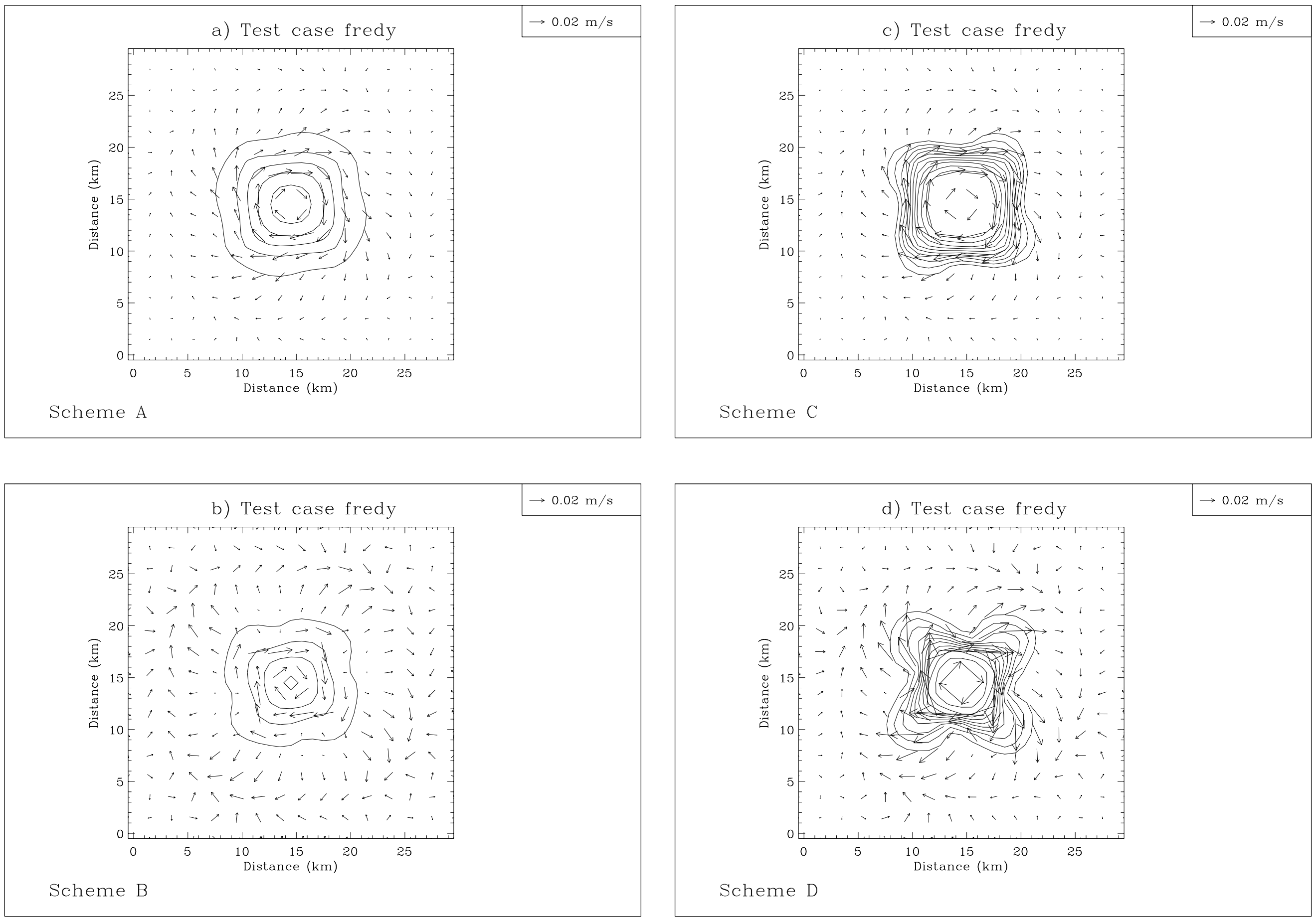


Figure 2.2.11: Surface current and salinity after 6 days for experiment A (a), B (b), C (c), D (d). Contour lines are at intervals of 0.1 PSU.

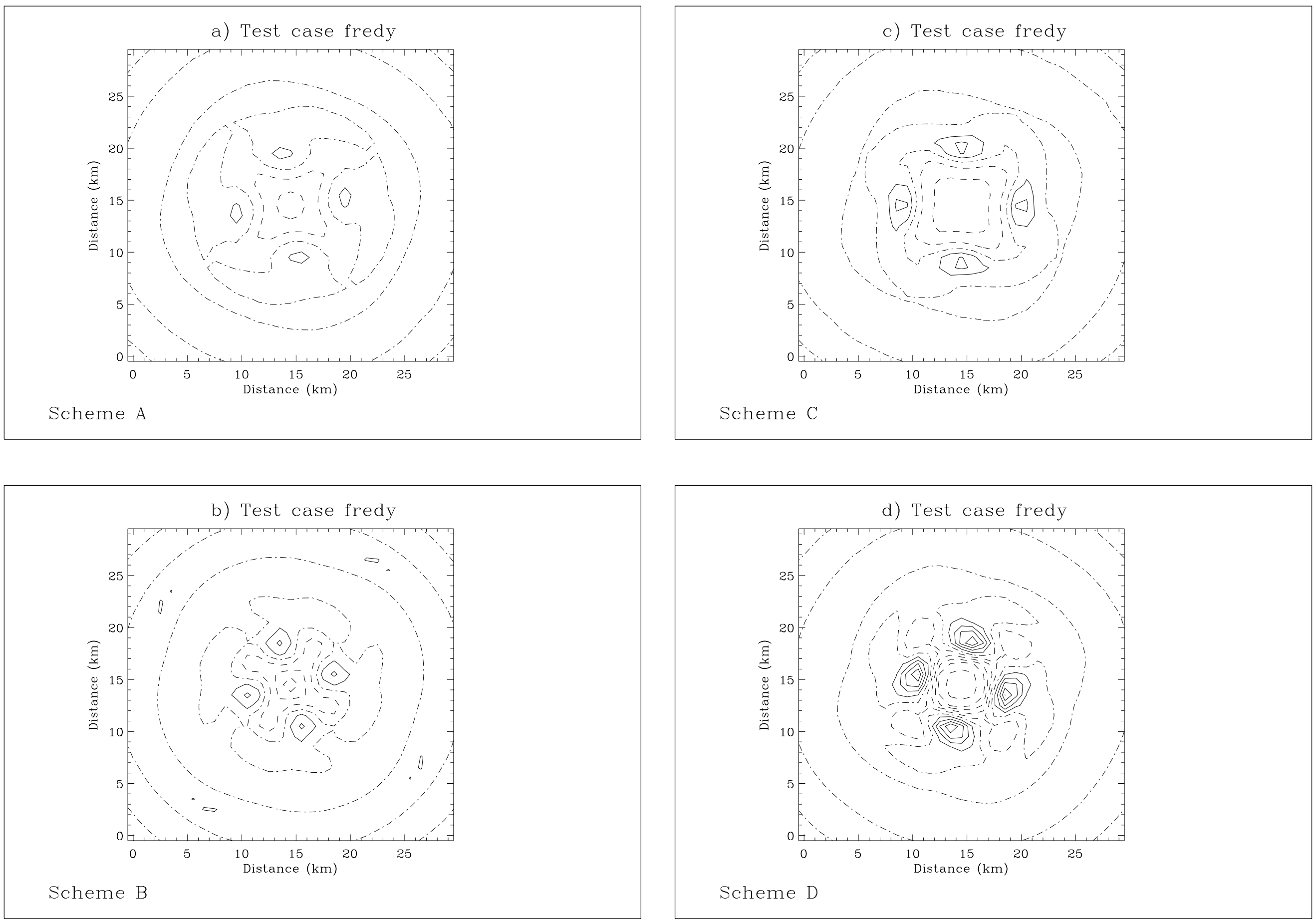


Figure 2.2.12: Surface vorticity divided by the Coriolis frequency for experiment A (a), B (b), C (c), D (d). Solid and dash-dotted contour lines represent cyclonic, respectively anticyclonic vorticity. Contour lines are at 0.1 intervals.

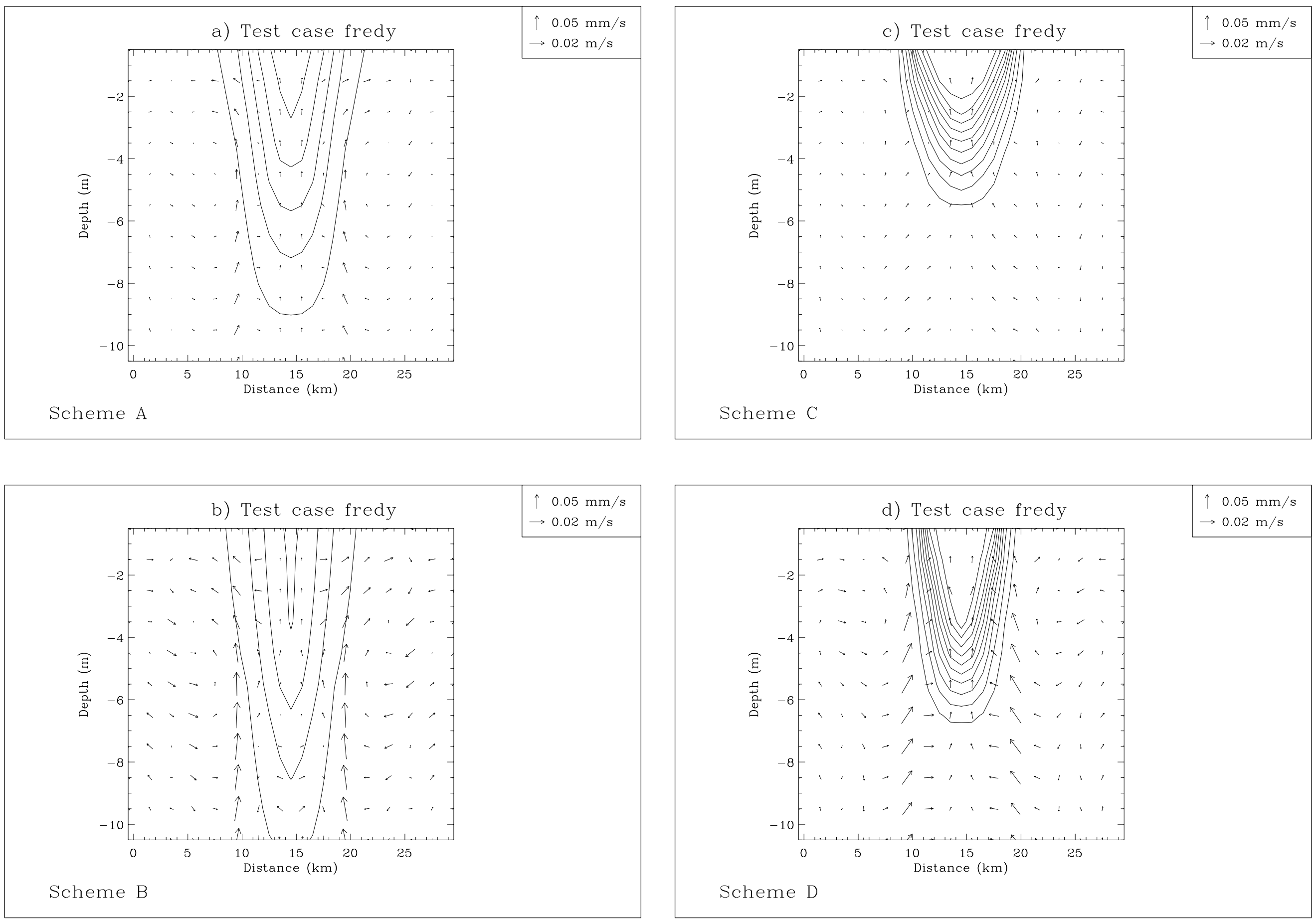


Figure 2.2.13: Current and salinity along a transect parallel to the X-axis and through the centre of the patch after 6 days for experiment A (a), B (b), C (c), D (d). Contour lines are at intervals of 0.1 PSU.

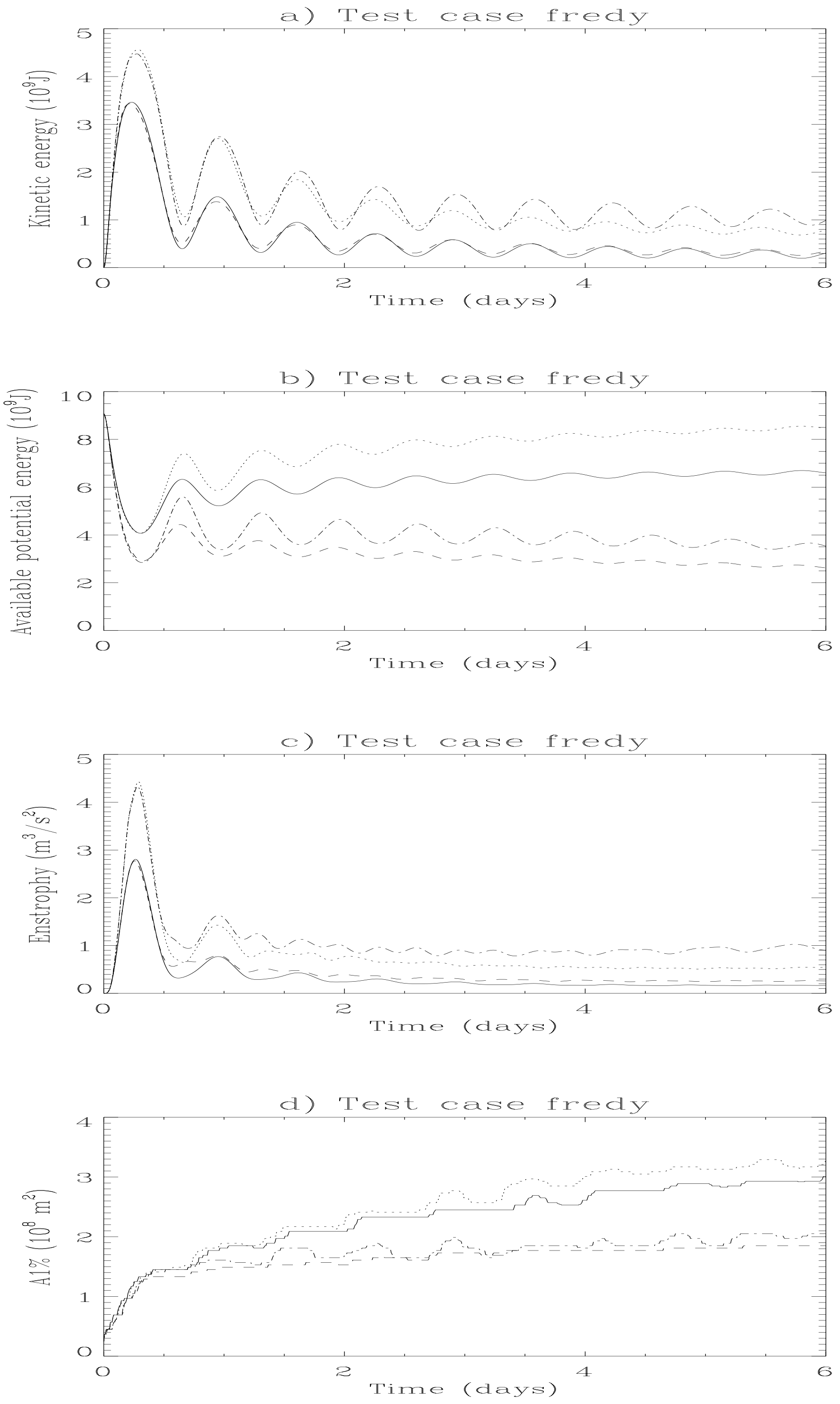


Figure 2.2.14: Time series of the kinetic energy E_{kin} in 10^9J (a), the available potential energy A_{pot} in 10^9J (b), the enstrophy E_{str} in m^3/s^2 (c) and the surface area A1% in units of 10^8m^2 (d) for experiment A (solid), B (dots), C (dashes), D (dash-dots).

Chapter 3

Test Cases for Turbulence

3.1 Test case pycno

3.1.1 Theory

This test case is a 1-D problem describing the evolution of a wind-driven surface mixed layer in the absence of rotation. Although simple in form the problem is of considerable oceanographic interest. Moreover it allows to intercompare in a simple way the different turbulence formulations implemented in the program. The initial state consists of a water column at rest with a stable stratification in the vertical using a constant density gradient. A surface stress is applied initially remaining uniform in space and time. A surface mixed layer develops which grows in time. The deepening is governed by the entrainment of denser water at the base of the mixed layer. As will be shown below the process critically depends on some parameters of the turbulence formulation while others are of less importance.

The basic equations in the absence of advection and horizontal diffusion are given by

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x_3}\Big(\nu_T\frac{\partial u}{\partial x_3}\Big) \tag{2.3.1}$$

$$\frac{\partial \rho}{\partial t} = \frac{\partial}{\partial x_3}\Big(\lambda_T\frac{\partial \rho}{\partial x_3}\Big) \tag{2.3.2}$$

where x_3 is the vertical coordinate (increasing upwards and zero at the surface), u the current, ρ the density and ν_T, λ_T the vertical eddy coefficients[1]. The initial and boundary conditions are given by

$$u = 0\ ,\ -\frac{g}{\rho}\frac{\partial\rho}{\partial x_3} = N_0^2 \quad \text{at} \quad t = 0 \tag{2.3.3}$$

$$\nu_T\frac{\partial u}{\partial x_3} = u_{*s}^2\ ,\ \lambda_T\frac{\partial\rho}{\partial x_3} = 0 \quad \text{at} \quad x_3 = 0 \tag{2.3.4}$$

where N_0 is the uniform initial buoyancy frequency and ρ_0 a reference density.

[1]As discussed in Chapter III-1.2 the eddy coefficients reduce to their molecular values in the absence of turbulence.

Kundu (1981) explored the possibility to cast the solutions of (2.3.1) and (2.3.2) into a self-similar form

$$u(x_3,t) = \hat{U}(t)F(\xi) \tag{2.3.5}$$

$$\rho(x_3,t) = \rho_h(t) + (\hat{\rho}(t) - \rho_h(t))G(\xi) \tag{2.3.6}$$

where $\xi = -x_3/h(t)$, $h(t)$ is the depth of the turbulent layer, ρ_h the density at the base of the layer and $\hat{U}, \hat{\rho}$ the values of respectively the current and density averaged over the turbulent layer. He showed that (2.3.5)–(2.3.6) are acceptable forms provided that the following additional assumptions are made:

- $u = 0$ and $Ri = -\rho_0(\partial u/\partial x_3)^2/(g(\partial\rho/\partial x_3)) = Ri_c$ at $x_3 = -h$ where Ri_c is a constant critical Richardson number
- the eddy coefficients can be cast into the form

$$\nu_T = f_\nu(\xi)h^2/t\ ,\ \lambda_T = f_\lambda(\xi)h^2/t \tag{2.3.7}$$

The solutions of (2.3.1)–(2.3.2) can then be written as

$$u(x_3,t) = \frac{u_{*s}}{(2\overline{Ri})^{1/4}}F(\xi)(tN_0)^{1/2} \tag{2.3.8}$$

$$\begin{aligned}\Delta\rho(x_3,t) &= \rho(x_3,t) - \rho(x_3,0)\\ &= \frac{\rho_0 N_0^2}{2g}(2 - 2\xi - G(\xi))h(t)\end{aligned} \tag{2.3.9}$$

and

$$h(t) = (2\overline{Ri})^{1/4}u_{*s}(t/N_0)^{1/2} \tag{2.3.10}$$

The bulk Richardson number $\overline{Ri}$ is defined by

$$\overline{Ri} = \frac{N_0^2 h^2(t)}{2\hat{U}^2(t)} = -\frac{F'^2(1)}{G'(1)}Ri_c = \text{constant} \tag{2.3.11}$$

where a $'$ denotes a derivative with respect to ξ. For a more profound theoretical discussion of this idealised mixed layer problem the reader is referred to the papers by e.g. Kundu (1981), Kranenburg (1983), Luyten et al. (1996).

3.1.2 Model setup

Equations (2.3.1)–(2.3.4) are solved numerically with the following parameters

$$u_{*s} = 0.01\text{m/s}\ ,\ N_0 = 0.01\text{s}^{-1} \tag{2.3.12}$$

for a simulated period of 1 day. The total water depth is set to 100 m which is more than 2 times larger than the simulated mixed layer depths. The density equation (2.3.2) is transformed into an equivalent salinity equation using a linear equation of state. The initial salinity field is given by

$$S = S_0 - \frac{N_0^2}{g\beta_S}(x_3 + H) \tag{2.3.13}$$

where S_0 is a reference salinity and H the water depth. Model setup parameters are listed in Table 2.3.1.

3.1.3 Experiments and output parameters

Seven numerical experiments have been defined by setting a number of turbulence switches listed in Table 2.3.2. They belong to the class of turbulence closure schemes, further discussed in Section III-1.2.2 and selected in the program by setting the switch IOPTK equal to 2. The reader is referred to that section for more details. The following schemes are used in the experiments:

A : zero-equation model using the Mellor-Yamada expressions (3.1.118)–(3.1.119) for the stability functions and the “Blackadar” mixing length formulation (3.1.134)–(3.1.135)

B : one-equation model using the Munk-Anderson formulation (3.1.122) for the stability functions and the “Blackadar” mixing length (3.1.134)–(3.1.135) without limiting conditions

C : as scheme B now with limiting conditions enabled

D : one-equation model using the $k - \varepsilon$ formulation (3.1.120)–(3.1.121) for the stability functions and the parabolic mixing length (3.1.131) with limiting conditions enabled

E : as scheme D now with the “Blackadar” mixing length formulation (3.1.134)–(3.1.135)

F : two-equation $k-\varepsilon$ model using the $k-\varepsilon$ formulation (3.1.120)–(3.1.121) for the stability functions with limiting conditions enabled

G : two-equation Mellor-Yamada model using the formulation (3.1.118)–(3.1.119) for the stability functions and without limiting conditions

Experiment E is performed with the default scheme of the program. All schemes, except B, yield a critical Richardson number so that the first criterion for self-similarity is satisfied. It can be demonstrated that none of the schemes complies with the second criterion. As shown by Luyten et al. (1996) the latter condition is of minor importance.

Results for all seven test runs are given in Figures 2.3.1, Figures 2.3.2 and Table 2.3.3. Time series of the mixed layer depth $h(t)$, the surface current $u(0,t)$ and the surface density difference $\Delta\rho(0,t)$ are shown in Figures 2.3.1a–f. The mixed layer depth is determined as the surface distance of the first grid point below the surface where the current is lower than 1% of its surface value.Vertical profiles of the current u, the density ρ minus the initial surface value $\rho(0,0)$ and the Richardson number Ri after one day are plotted in Figures 2.3.2a–f.

A number of output parameters are defined. Their intention is primarily to compare the results with the self-similarity theory. Equations (2.3.8)–(2.3.11) for the surface current, the surface value of $\Delta\rho$, $h(t)$ and $\overline{Ri}$ are then written in the form

$$\ln h = \alpha_h + \beta_h \ln(tN_0) \tag{2.3.14}$$

$$\ln u(0,t) = \alpha_u + \beta_u \ln(tN_0) \tag{2.3.15}$$

$$\ln \Delta\rho(0,t) = \alpha_d + \beta_d \ln(tN_0) \tag{2.3.16}$$

$$\ln \overline{Ri} = \alpha_r + \beta_r \ln(tN_0) \tag{2.3.17}$$

where, according to the self-similarity theory,

$$\beta_h = \beta_u = \beta_d = 1/2 \ , \ \beta_r = 0 \tag{2.3.18}$$

and

$$\ln 2\overline{Ri} = 4(\alpha_h - \ln u_{*s}/N_0) = 4\alpha_h \tag{2.3.19}$$

The eight coefficients (α_h,α_u,α_d,α_r), (β_h,β_u,β_d,β_r) in (2.3.14)–(2.3.17) are determined from the model results by applying a linear regression analysis. To remove the influence of initial conditions and adjustments the analysis is restricted to the last 14 hours. A description of all output parameters is given below. Values for each experiment are listed in Table 2.3.3.

RVMEAN
: The value of $\overline{Ri}$ defined by (2.3.11) and averaged over the last 14 hours.

BRV
: The coefficient β_r in the linear relation (2.3.17) for $\overline{Ri}$.

CORRV
: The squared correlation coefficient r^2 for the linear relation (2.3.17). A zero value indicates that $\overline{Ri}$ is uncorrelated with time.

RVMEAN2
: The value of $\overline{Ri}$ as determined form (2.3.19).

BH
: The coefficient β_h in the linear relation (2.3.14) for $h(t)$ (log(m)).

CORRH
: The squared correlation coefficient r^2 for the linear relation (2.3.14).

BU2
: The coefficient β_u in the linear relation (2.3.15) for $u(0,t)$ (log(m/s)).

CORRU2
: The squared correlation coefficient r^2 for the linear relation (2.3.15).

BDR0
: The coefficient β_d in the linear relation (2.3.16) for $\Delta\rho(0,t)$ (log(kg/m^3)).

CORRDR0
: The squared correlation coefficient r^2 for the linear relation (2.3.16).

VEDMAX
: The maximum value of the eddy viscosity coefficient ν_T over the whole water depth (m^2/s) at the end of the simulation.

DIFMAX
: The maximum value of the eddy diffusion coefficient λ_T over the whole water depth (m^2/s) at the end of the simulation.

TKEMAX
: The maximum value of the turbulence energy k over the whole water depth (J/kg) at the end of the simulation .

TMLD
: The value of h at the end of the simulation (m).

USUR
: The value of the surface velocity at the end of the simulation (m/s).

DROSUR
: The value of the surface density difference $\Delta\rho$ at the end of the simulation (kg/m^3).

SDEV
: The relative difference between the depth-averaged salinity and its initial value, defined by

$$\mathsf{SDEV} = 10^5(\overline{S}(t) - \overline{S}(0))/\overline{S}(0) \tag{2.3.20}$$

where an overbar denotes a depth-averaged value. In the absence of bottom and surface salinity fluxes, the exact value of SDEV is zero. A conservative scheme is applied in the model for the vertical diffusion of salinity so that non-zero values are only due to rounding errors. This parameter is therefore useful to test the machine accuracy.

3.1.4 Results

The results can be summarised as follows:

Table 2.3.1: Model setup for the test case **pycno**.

parameter	value	purpose
param.inc		
NC[a]	2	number of grid points in the X-direction
NR[a]	2	number of grid points in the Y-direction
NZ	200	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NOBV	4	number of open boundary points in the Y-direction
DEFCON		
IGRDIM	1	1-D application of the program
IADVC	0	advection of momentum disabled
IADVS	0	advection of salinity disabled
IOPT2	0	mode splitting disabled
IOPTSA	1	update of the salinity field enabled
IBSTR	0	bottom stress set to zero
IOPTM	1	uniform surface stress
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01020000	end date (MMDDHHMM)
DELT	60	time step (s)
DEPUN	100.0	uniform water depth (m) (used in 1-D applications only)
FSUN	0.0001	uniform surface stress in the X-direction (m^2/s^2)
GSUN	0.0	uniform surface stress in the Y-direction (m^2/s^2)
DEFICS		
S(NZ,NR,NC)		initial salinity field as given by (2.3.13) (PSU)

[a] Although the program is used as a 1-D model in this application, a minimum value of 2 is required by the program structure.

Table 2.3.2: Settings of the turbulence switches for the **pycno** experiments.

Test	NTRANS	ITCPAR	ISTPAR	ILENG	ILIM
A	0	1	1	4	0
B	1	1	2	4	0
C	1	1	2	4	1
D	1	2	1	1	1
E	1	2	1	4	1
F	2	2	1	4	1
G	2	1	1	4	0

Table 2.3.3: Values of the output parameters for the test case **pycno**.

parameter	A	B	C	D	E	F	G
RVMEAN	0.3312	3.900	1.104	0.7074	0.6715	0.6491	0.4769
BRV	0.1215	0.1985	0.0716	-0.0110	0.0472	-0.0041	0.0456
CORRV	0.6443	0.8146	0.5542	0.0185	0.3082	0.0033	0.1972
RVMEAN2	0.1527	1.097	0.6989	0.7593	0.4967	0.6667	0.3566
BH	0.5303	0.5496	0.5179	0.4972	0.5118	0.4989	0.5114
CORRH	0.9982	0.9981	0.9990	0.9984	0.9988	0.9987	0.9980
BU2	0.5036	0.3840	0.4076	0.4268	0.4240	0.4128	0.4202
CORRU2	0.9999	0.9991	1.000	1.000	1.000	1.000	1.000
BDRO	0.4905	0.5487	0.5378	0.5229	0.5265	0.5309	0.5402
CORRDRO	0.9999	0.9999	1.000	1.000	1.000	1.000	1.000
VEDMAX	10.02	1.221	1.188	1.907	1.263	1.428	1.255
DIFMAX	12.56	1.399	1.379	2.606	1.837	2.092	1.539
TKEMAX	230.2	3.210	3.202	3.000	3.001	3.002	7.181
TMLD	26.75	50.25	36.25	32.25	31.75	31.25	29.25
USUR	0.6507	0.4030	0.4335	0.4331	0.4560	0.4506	0.4770
DROSUR	0.0987	0.1523	0.1434	0.1375	0.1358	0.1354	0.1257
SDEV	-2.218	-2.159	-12.09	-0.1416	-0.1770	-0.1652	2.584

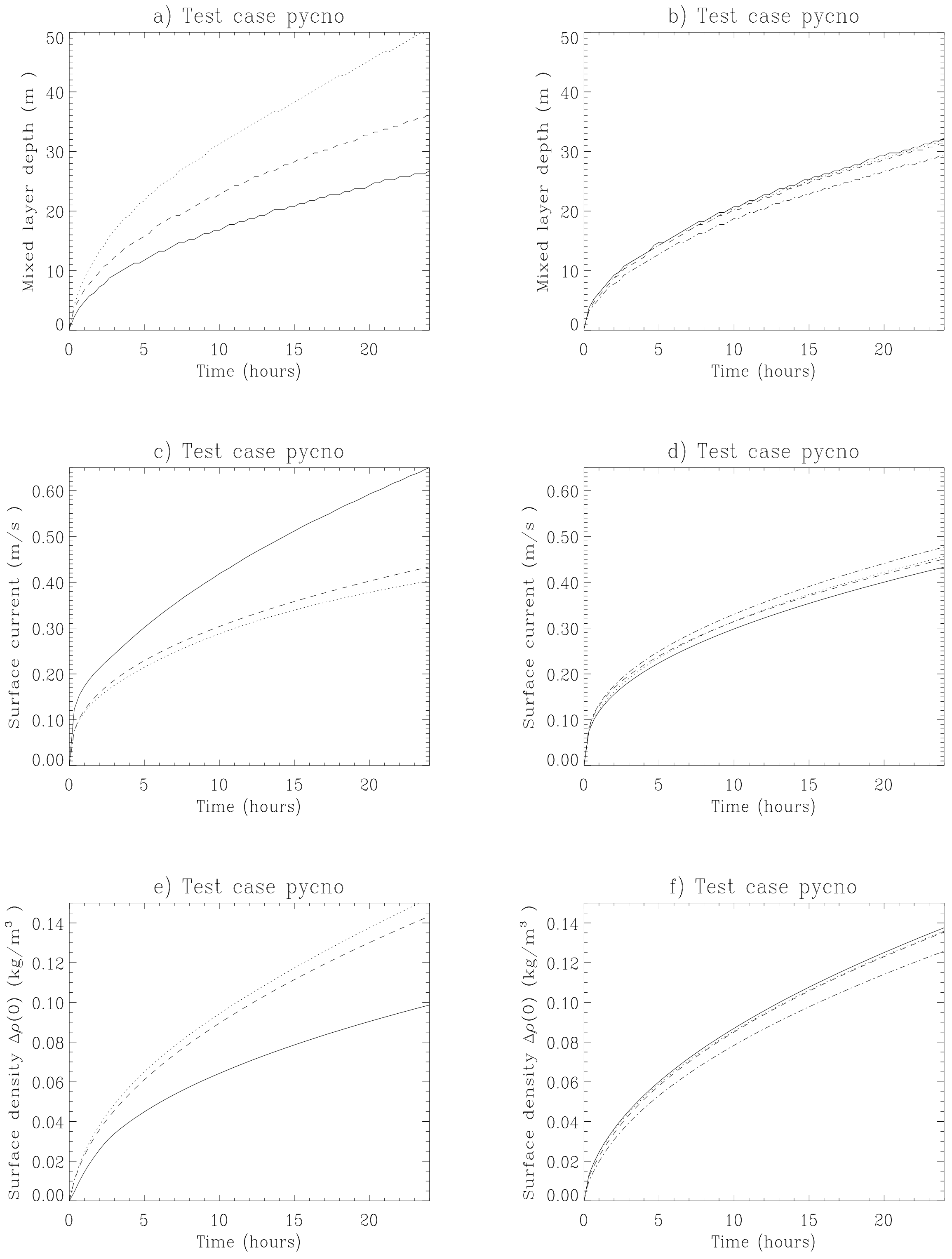


Figure 2.3.1: Time series of the mixed layer depth (a–b), the surface current (c–d) and the surface density difference $\Delta\rho(0,t)$ (e–f). Test runs A, B, C, are represented in Figures a, c, e by respectively the solid, dotted and dashed curves. Test runs D, E, F, G are shown in Figures b, d, f by respectively the solid, dotted, dashed and dash-dotted curves.

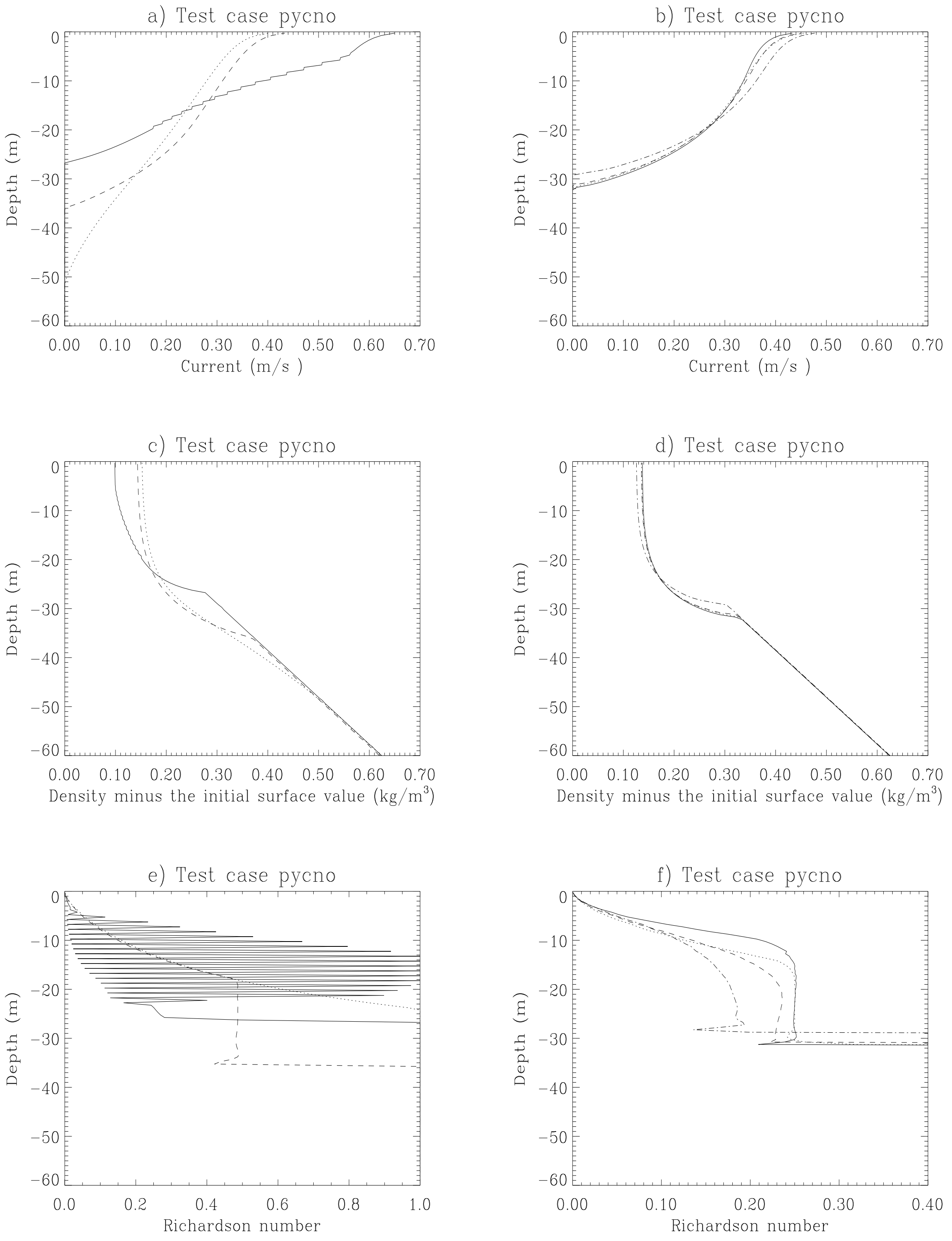
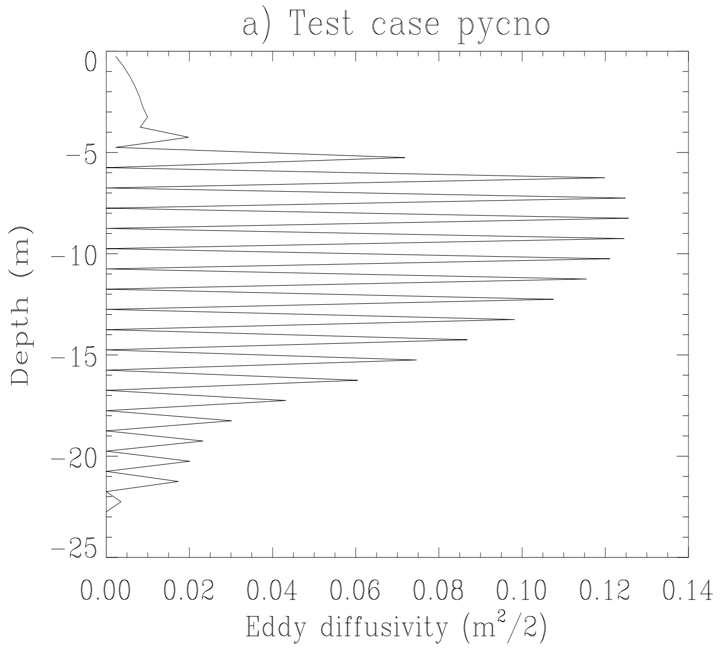


Figure 2.3.2: Vertical profiles of the current (a–b), the density minus $\rho(0,0)$ (c–d) and the Richardson number (e–f) after one day. Test runs A, B, C, are represented in Figures a, c, e by respectively the solid, dotted and dashed curves. Test runs D, E, F, G are shown in Figures b, d, f by respectively the solid, dotted, dashed and dash-dotted curves.



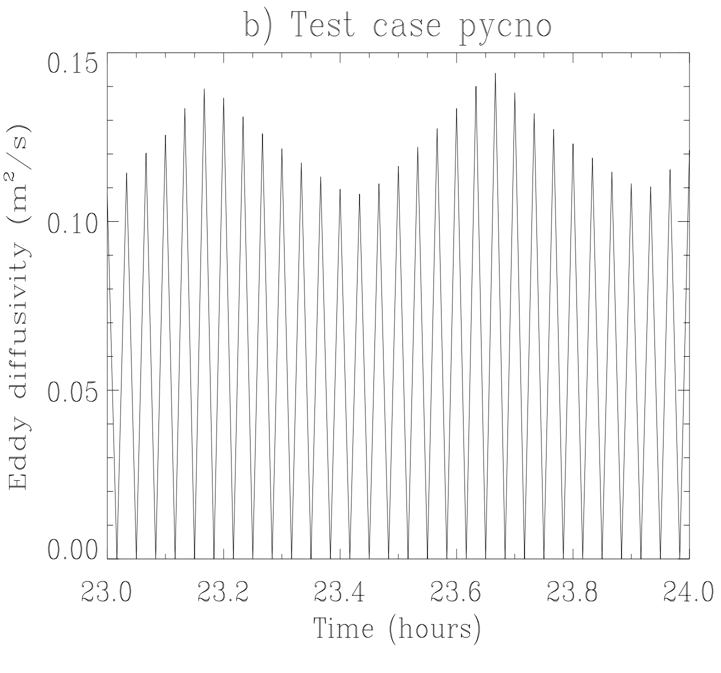


Figure 2.3.3: Vertical profile of the vertical diffusion coefficient λ_T after one day (a) and time series of λ_T at 10 m depth during the last hour of the simulation (b) for test run A.

- The time-averaged bulk Richardson number $\overline{Ri}$ (parameter RVMEAN) is much higher for the schemes B and C which use the Munk-Anderson formulation for the stability functions compared to the other schemes and is larger for the $k-\varepsilon$ (D, E, F) than for the $k-l$ (A, G) schemes.

- The general time dependence of $\overline{Ri}$ (not shown) consists of an oscillation around a mean value which increases in time. The amplitude of the oscillations and the growth rate of the mean value of $\overline{Ri}$ remain small in the case of schemes C, D, E, F and G. This explains why BRV is smaller in those cases whereas CORRV is not always close to zero in experiments A and B.

- The parameter RVMEAN2 is mostly lower than RVMEAN. The former is obtained from the linear regression analysis and (2.3.19) whereas the latter is evaluated using the definition (2.3.11). The difference is larger for the schemes A–C indicating that the self-similarity hypothesis is less valid for this type of schemes. The laboratory data of Kato and Phillips (1969) calibrated by Price (1979) yield a value of $\overline{Ri}$ close to 0.6 which agrees more with the values predicted by the $k-\varepsilon$ schemes E, F than those obtained from the $k-l$ schemes A and G.

- The parameters β_h and β_d are close to 0.5 in agreement with the value predicted by the self-similarity theory whereas $\beta_u \simeq 0.4$ (except for case A) which is lower than the theoretical value of 0.5.

- For larger values of $\overline{Ri}$ (parameter RVMEAN) the surface current tends to decrease whereas the surface density and the mixed layer depth tend to increase in agreement with equations (2.3.8)–(2.3.10).

- The schemes D to G produce larger shears and much shallower mixed layer depths compared to the B and C schemes. Note the remarked difference between the zero-equation and two-equation models A and G which are physically nearly the same. The discrepancy is most clearly observed in the profiles of the current (Figures 2.3.2a–b) and the density (Figures 2.3.2c–d).

- From Figures 2.3.2e–f one observes that the Richardson number approaches its critical value in the stratified zone just below the surface mixed layer (except in the runs A and B). Note that $Ri \simeq Ri_c$ at the base of the turbulent layer for the schemes C, D, E, F which use a limiting condition (ILIM $= 1$). This is in agreement with the theory discussed in Section III-1.2.2c. The highly oscillating curve for run A is due to a numerical instability (see below).

- It is clear that the results obtained with the zero-equation model A should not be considered as realistic. As explained in Section III-1.2.2b this type of schemes becomes numerically unstable either when the time step Δt is too large or the vertical grid spacing Δx_3 is too small. The problem is illustrated in Figures 2.3.3a–b showing the vertical profile of λ_T at the end of the simulation and a time series of λ_T for the last hour of the run at 10 m depth. An unrealistic "jittering" is observed oscillating over 2 vertical grid spacings and 2 time steps. The problem can be remedied by reducing the time step or increasing the vertical resolution. A more elaborate discussion can be found in Frey (1991) and Luyten et al. (1996). Since there exists no clear criterion for the stability of the level 2 schemes, it is recommended to avoid the zero-equation scheme in a realistic model application.

3.2 Test case csnsp

3.2.1 Description of the problem and model setup

The intention of the previous test cases is to verify the different advection and turbulence schemes implemented in the program against analytical solutions. The setup of these applications is however far from the realistic conditions prevailing in coastal and shelf seas. A more realistic study should take account of the effects of e.g. wind, tides and seasonal stratification. Test case **csnsp** has been developed for this purpose. This simulates the annual cycle of thermal stratification at station CS (55^0 $30'$N, 0^0 $55'$E) located in the deeper parts of the North Sea where a thermocline forms during the summer. The model is forced using realistic meteorological and tidal data. To limit CPU time the simulation is performed with the 1-D version of the program so that advective effects are ignored. A full 3-D application for the North Sea is described in Chapter II-7. The results are validated with the observed data sets of the UK North Sea Project (Charnock et al., 1994). This test case allows to compare the influence of different formulations available in the program (turbulence, boundary conditions, optics, equation of state).

Using the notations of Section II-3.1.1 the basic equations are

$$\frac{\partial u}{\partial t} - fv = -g\frac{\partial \zeta}{\partial x_1} + \frac{\partial}{\partial x_3}\Big(\nu_T\frac{\partial u}{\partial x_3}\Big) \tag{2.3.21}$$

$$\frac{\partial v}{\partial t} + fu = -g\frac{\partial \zeta}{\partial x_2} + \frac{\partial}{\partial x_3}\Big(\nu_T\frac{\partial v}{\partial x_3}\Big) \tag{2.3.22}$$

$$\frac{\partial T}{\partial t} = \frac{1}{\rho_0 c_p}\frac{\partial I}{\partial x_3} + \frac{\partial}{\partial x_3}\Big(\lambda_T\frac{\partial T}{\partial x_3}\Big) \tag{2.3.23}$$

where $f = 2\Omega\sin\phi$ is the Coriolis frequency, ϕ the latitude, ζ the surface elevation, T the temperature, c_p the specific heat of seawater at constant pressure and $I(x_3,t)$ solar irradiance. Neglecting the baroclinic component of the pressure gradient the surface slope terms are written as

$$g\frac{\partial \zeta}{\partial x_1} = F\cos(\omega_2 t - \varphi_x)\ ,\ g\frac{\partial \zeta}{\partial x_2} = G\cos(\omega_2 t - \varphi_y) \tag{2.3.24}$$

where ω_2 is the frequency of the dominant M_2-tide.

The boundary conditions are

$$\rho_0\nu_T\Big(\frac{\partial u}{\partial x_3},\frac{\partial v}{\partial x_3}\Big) = (\tau_{s1},\tau_{s2}) \ , \ \rho_0 c_p\lambda_T\frac{\partial T}{\partial x_3} = -Q_{nsol} \tag{2.3.25}$$

at the surface, and

$$\rho_0\nu_T\Big(\frac{\partial u}{\partial x_3},\frac{\partial v}{\partial x_3}\Big) = \rho_0 C_D^b(u_b^2+v_b^2)^{1/2}(u_b,v_b) \ , \ \rho_0 c_p\lambda_T\frac{\partial T}{\partial x_3} = 0 \tag{2.3.26}$$

at the bottom where (u_b,v_b) are the velocities at the grid point nearest to the bottom. The surface stress is related to the wind speed and direction using the empirical formula (3.1.221) after Geernaert et al. (1986). The non-solar heat flux $-Q_{nsol}$ is calculated as a function of wind speed, air temperature, relative humidity and cloud coverage with the aid of the standard relations described in Section III-1.6.1b and the Monin-Obukhov similarity formulation discussed in Section III-1.6.4. The bottom drag coefficient C_D^b is expressed as a function of the roughness length z_0 using (3.1.187).

Solar irradiance is decomposed into a near-infrared part absorbed in a shallow surface layer, and a non-infrared part which attenuates exponentially within a larger surface layer. An extra surface attenuation is taken into account. Further details of the optical module are described in Section III-1.4. The relations which determine the solar radiation incident on the surface, are discussed in Section III-1.6.5.

The simulation is performed from the beginning of January until the end of October 1989. The initial temperature is taken from the observed vertical profile from the North Sea Project. The meteorological data are obtained from the British Meteorological Office. Model setup parameters are listed in Table 2.3.4.

3.2.2 Experiments and output parameters

Eight experiments are defined (see Table 2.3.5). The first four (A, B, C, D) are intended to examine the role of the turbulence scheme whereas the last four analyse the influence of the surface boundary condition for temperature (E, F), the formulation of the optics (G) and the equation of state used to evaluate the density gradient (H). Unless stated otherwise all experiments use the setup of experiment A.

A : Uses the default turbulence scheme and model setup listed in Table 2.3.4.

B : The limiting conditions for the mixing length and turbulence energy are disabled.

C : The formulation (3.1.123) instead of (3.1.120)–(3.1.121) is used for the stability functions and limiting conditions are disabled.

D : The turbulence scheme is replaced by the simpler empirical relations (3.1.95)–(3.1.100) of Munk and Anderson.

E : The surface exchange coefficients C_E and C_H take the uniform values (3.1.217) instead of the Monin-Obukhov formulation.

F : All radiant heat is absorbed at the sea surface, i.e. $I = 0$ below the surface and equal to the incident solar radiation at the surface.

G : The effect of the extra surface attenuation is not taken into account, i.e. $R_2 = 1$ (see Section III-1.4).

H : The buoyancy frequency in the turbulence expressions is determined assuming a uniform value for the thermal exchange coefficient.

The evolution of the surface and bottom temperatures for experiments A, C, D, F is illustrated in Figure 2.3.4. Time-depth contour plots are shown in Figures 2.3.5a–h for all experiments. The results in Figures 2.3.4 and 2.3.5 are displayed using averaged values over a 3-day period.

The following output parameters are defined:

TSUR
: Surface temperature (^{0}C) measured one-half grid distance below the surface.

TBOT
: Bottom temperature (^{0}C) measured one-half grid distance above the bottom.

TMEAN
: Depth-averaged temperature (^{0}C).

TDEP
: The thermocline depth (m) measured as the distance to the surface where the temperature is 1^0C higher than the bottom temperature.

TGRAD
: Maximum value of the vertical temperature gradient (^{0}C/m).

TWIDTH
: This parameter measures the sharpness of the thermocline and is defined as the vertical distance (m) between the two points where the temperature is equal to respectively 8.5 ^{0}C and 12 ^{0}C.

Values are listed in Table 2.3.6 for all experiments together with the corresponding observed values at the times when the measured vertical profiles were taken. The measurement dates are indicated by the number in the second column: March 6 (1), April 8 (2), May 4 (3), June 2 (4), July 1 (5), August 4 (6), September 1 (7) and September 29 (8) corresponding to intervals of approximatively one month.

3.2.3 Results

The results can be summarised as follows:

- The effect of using a limiting condition for the mixing length is a somewhat higher surface temperature while there is no appreciable effect on the temperature within the tidally mixed bottom layer. Thermocline depths are larger in the late summer and autumn with scheme B.

- Schemes C and D produce a much more diffusive thermocline. This is due to the absence of a critical Richardson number in these formulations. In consequence, surface temperatures are up to 2^0C lower while bottom temperatures are 2–3^0C higher during the summer. Compared to all other experiments and the data the two schemes yield less sharp temperature gradients and a much wider thermocline.

- An important difference between the schemes C and D which make use of of the Munk-Anderson formulation and the turbulence energy models is that the latter produce an almost vertically mixed water column by day 280 (beginning of October) whereas a significant surface-bottom temperature difference remains in the latter experiments even at the end of October. This is clearly observed in Figure 2.3.4 where schemes C, D are represented by the dotted and dashed curves and the turbulence energy schemes by the solid and dash-dotted curves.

- Experiments E, G and H do not produce significantly different results compared to the "standard" scheme A except that surface temperatures are slightly higher (0–0.5^0C) in case E.

- As can be observed from Figure 2.3.4 the surface temperature rapidly increases during periods of strong surface heating in the case of experiment F, attaining values above 20^0C. This is due to the absorption of all radiant heat at the surface and results in a large surface stratification inhibiting the downwards diffusion of heat by the turbulence.

Results of 1-D numerical simulations at station CS are further discussed in Luyten (1996) and Warrach (1998).

Table 2.3.4: Model setup for test case **csnsp**.

parameter	value	purpose
param.inc		
NC[a]	2	number of grid points in the X-direction
NR[a]	2	number of grid points in the Y-direction
NZ	50	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NOBV	4	number of open boundary points in the Y-direction
NCON	1	number of tidal frequencies
DEFCON		
IGRDIM	1	1-D application of the program
IADVC	0	advection of momentum disabled
IADVS	0	advection of temperature disabled
IOPT2	0	mode splitting disabled
IOPTHE	2	module for the update of temperature switched on, optical module switched on (except for experiment F)
IOPTD	2	thermal expansion coefficients calculated using the general equation of state of seawater (see Section III-1.5) except for experiment H
IOPTM	2	program reads time-dependent meteorological input data
IDRAG	3	surface stress evaluated using the Geernaert et al. (1986) formulation
ITDIF	1	surface exchange coefficients evaluated using the Monin-Obukhov formulation (see Section III-1.6.4) except for experiment E
IOUTF	1	time-averaged output enabled
IBDATE	01030000	start date (MMDDHHMM)
IEDATE	10312100	end date (MMDDHHMM)
IBYEAR	1989	start year
IEYEAR	1989	end year
DELT	1200	time step (s)
ICMET	9	time interval for meteorological input divided by DELT
ICRFORM	'U'	files with initial/final conditions written in binary format

[a]Although the program is used as a 1-D model in this application, a minimum value of 2 is required by the program structure.

Table 2.3.4: continued.

parameter	value	purpose
BCSFORM	'U'	file with open boundary conditions written in binary format
DLAREF	55.515	reference latitude (degrees)
DLOREF	0.90833	reference longitude (degrees)
DEPUN	83	uniform water depth (m) (used in 1-D applications only)
R0REF	1026.8	reference density (kg/m^3)
SREF	34.8	reference salinity used to initialise the salinity field (PSU)
TREF	10.0	reference temperature used to initialise the temperature field (^{0}C)
CDZ0UN	0.0024	uniform roughness length used for the evaluation of the bottom drag coefficient (m)
DEFOB2D		
SIGMA(1)	1.4056E-04	frequency of the M_2-tide (rad/s)
AMPOBU(1-4,1)	4.2228E-05	the amplitude F of the surface slope term (2.3.24) in the X-direction (m/s^2)
PHAOBU(1-4,1)	4.3320	phase of the surface slope term (2.3.24) in X-direction (rad)
AMPOBV(1-4,1)	4.6299E-05	the amplitude G of the surface slope term (2.3.24) in the Y-direction (m/s^2)
PHAOBV(1-4,1)	5.8581	phase of the surface slope term (2.3.24) in Y-direction (rad)
DEFICS		
T(NZ,NR,NC)[a]		initial temperature field (^{0}C)
WRFCDAT		
WINDU2UN[b]		wind component in the X-direction (m/s)
WINDV2UN[b]		wind component in the Y-direction (m/s)
SAT2UN[b]		air temperature (^{0}C)
HUM2UN[b]		relative humidity
CLOUD2UN[b]		fractional cloud coverage
EVAPRUN	0.0	zero salinity flux at the surface (kg m^{-2} s^{-1})

[a]Initial temperature field is obtained from the input file *tin.cs*.
[b]Meteorological data are obtained from the input file *metin.cs*.

Table 2.3.5: Definition of the **csnsp** experiments.

Test	Purpose
A	uses the model setup listed in Table 2.3.4
B	as experiment A but without limiting conditions (ILIM = 0)
C	as experiment A but without using limiting conditions (ILIM = 0) and using the Munk-Anderson formulation (3.1.123) for the stability functions (ISTPAR = 2)
D	as experiment A but using the Munk-Anderson formulation (3.1.95)–(3.1.100) instead of the default turbulence scheme (IOPTK = 1, ITFORM = 2)
E	as experiment A but using constant values for the surface exchange coefficients instead of the Monin-Obukhov formulation of Section III-1.6.4 (ITDIF = 0)
F	as experiment A but with all radiant heat absorbed at the sea surface (IOPTHE = 1)
G	as experiment A but without the extra surface attenuation (R2OPTUN = 1)
H	as experiment A but using uniform values for the thermal expansion coefficient (IOPTD = 1)

Table 2.3.6: Output values for the parameters of the test case **csnsp**.

parameter	date	data	A	B	C	D	E	F	G	H
TSUR	1	7.088	7.206	7.231	7.158	7.199	7.227	7.649	7.100	7.210
	2	6.944	6.841	6.839	6.831	6.818	6.874	6.846	6.831	6.853
	3	9.158	9.926	9.842	9.125	9.387	10.22	11.42	9.080	10.01
	4	10.78	11.97	11.65	10.46	10.35	12.60	12.92	11.23	12.04
	5	13.43	14.20	13.82	12.15	12.50	14.51	15.52	13.70	14.17
	6	15.16	14.53	14.08	12.53	12.21	14.71	14.06	14.53	14.42
	7	15.47	15.32	15.08	13.56	13.19	15.53	15.09	15.25	15.21
	8	14.64	14.15	13.93	12.84	12.46	14.33	13.91	14.25	14.14
TBOT	1	7.053	6.803	6.803	6.802	6.797	6.823	6.798	6.805	6.802
	2	6.891	6.766	6.767	6.769	6.771	6.795	6.752	6.774	6.761
	3	7.029	6.895	6.929	7.063	7.125	6.924	6.841	6.957	6.862
	4	7.162	7.013	7.049	7.476	7.780	7.045	6.895	7.152	6.968
	5	7.271	7.139	7.177	8.041	8.571	7.171	6.954	7.356	7.078
	6	8.004	7.284	7.370	9.044	9.531	7.315	7.045	7.573	7.201
	7	8.030	7.414	7.597	9.876	10.54	7.445	7.149	7.754	7.315
	8	8.298	7.562	7.915	10.80	11.39	7.587	7.275	7.934	7.433
TMEAN	1	7.086	6.820	6.820	6.819	6.813	6.840	6.818	6.820	6.820
	2	6.902	6.795	6.796	6.797	6.790	6.826	6.786	6.799	6.794
	3	7.494	7.294	7.300	7.328	7.320	7.343	7.222	7.328	7.284
	4	8.527	8.091	8.137	8.319	8.324	8.155	7.896	8.237	8.067
	5	9.417	8.936	9.026	9.431	9.437	9.007	8.571	9.235	8.916
	6	10.69	9.698	9.872	10.65	10.67	9.742	9.279	10.12	9.702
	7	12.04	10.17	10.40	11.44	11.49	10.21	9.747	10.62	10.20
	8	13.48	10.15	10.43	11.68	11.80	10.20	9.761	10.60	10.19

Table 2.3.6: continued.

parameter	date	data	A	B	C	D	E	F	G	H
TDEP[a]	1	*	*	*	*	*	*	*	*	*
	2	*	*	*	*	*	*	*	*	*
	3	16.03	9.901	8.992	10.74	7.188	10.03	7.221	11.54	10.58
	4	34.40	23.04	23.98	28.35	22.86	22.60	20.31	27.15	22.81
	5	33.88	27.49	31.51	41.92	34.91	27.06	26.38	30.71	27.84
	6	34.38	31.67	36.00	40.30	36.79	31.21	30.68	33.86	32.51
	7	53.82	33.96	37.94	40.22	32.97	33.75	32.89	36.26	35.32
	8	70.00	37.59	41.41	35.65	23.43	37.31	36.69	39.32	39.06
TGRAD	1	0.0170	0.1570	0.1820	0.1005	0.1620	0.1566	0.4543	0.1320	0.1601
	2	0.0090	0.0121	0.0061	0.0046	0.0025	0.0099	0.0125	0.0070	0.0126
	3	0.2410	0.5217	0.4908	0.2160	0.3114	0.6270	0.9918	0.1555	0.5641
	4	0.6880	0.4809	0.5622	0.2707	0.3227	0.4438	0.8100	0.3808	0.5172
	5	0.9790	0.8613	0.5511	0.3212	0.6045	0.8153	1.213	0.7070	0.9067
	6	1.903	0.8357	0.9951	0.3811	0.2231	0.9203	0.8136	0.8365	0.9426
	7	1.149	0.6661	0.4996	0.2078	0.1281	0.6488	0.5747	0.7082	0.6714
	8	1.316	0.5024	0.5695	0.1765	0.0759	0.4802	0.4398	0.5328	0.4927
TWIDTH[b]	1	*	*	*	*	*	*	*	*	*
	2	*	*	*	*	*	*	*	*	*
	3	*	*	*	*	*	*	*	*	*
	4	*	*	*	*	*	18.58	16.76	*	19.55
	5	7.160	4.296	6.793	43.96	80.52	4.648	5.022	5.260	4.172
	6	14.28	5.414	7.705	47.70	54.67	5.238	5.895	5.162	5.057
	7	8.173	5.585	8.543	47.62	52.96	5.672	6.212	5.513	5.476
	8	15.39	7.675	13.02	47.72	52.34	7.687	8.467	7.178	7.346

[a] A “*” means that the water column is vertically mixed.
[b] A “*” means that TWIDTH could not be determined.

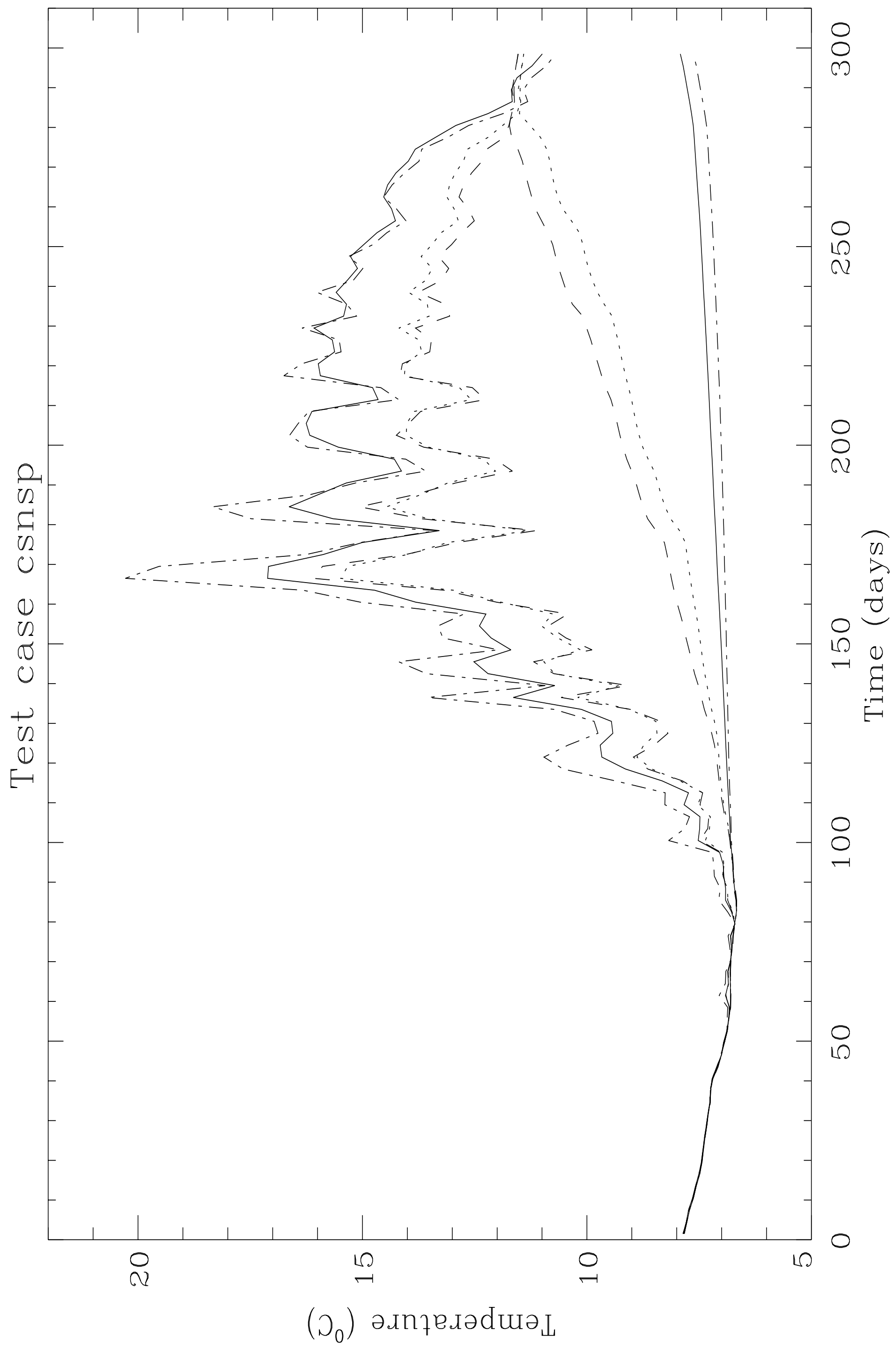


Figure 2.3.4: Time series of surface and bottom temperatures at station CS averaged over 3 days for experiment A (solid), C (dots), D (dashes) and F (dash-dots).

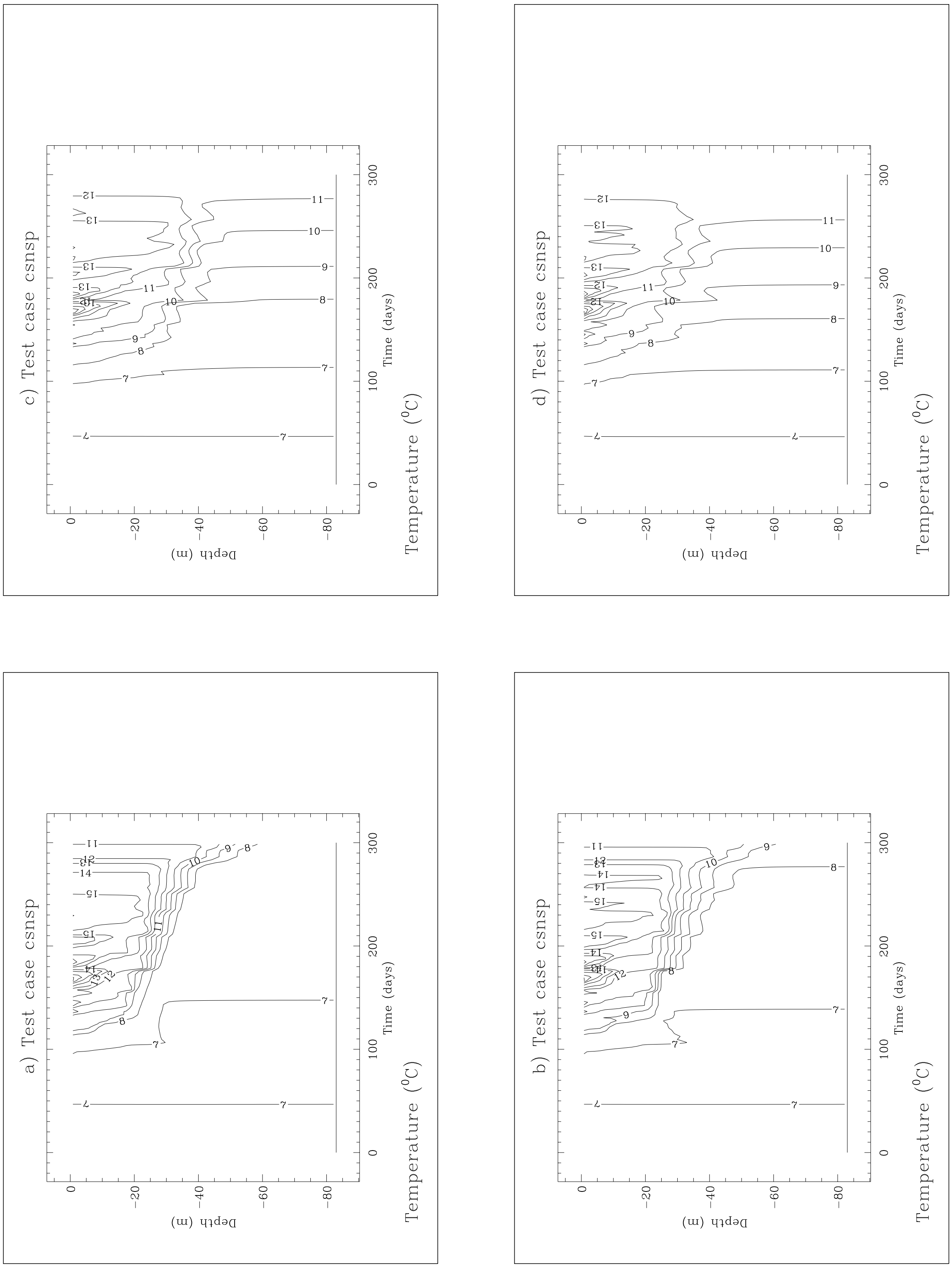
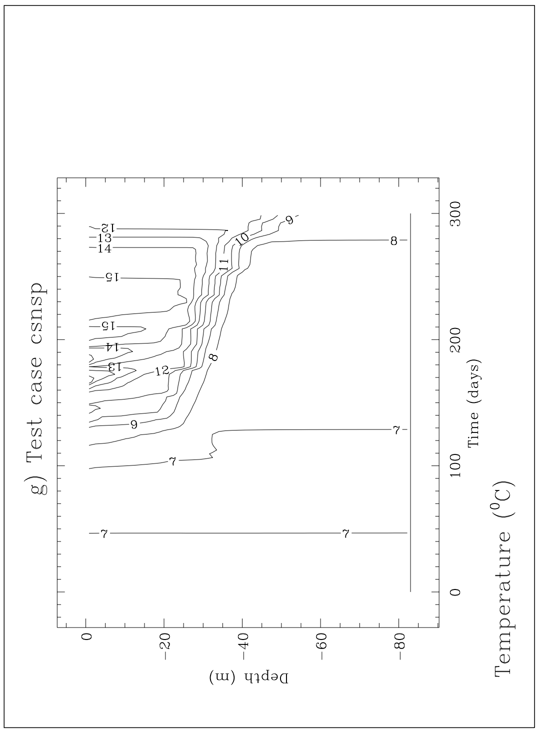
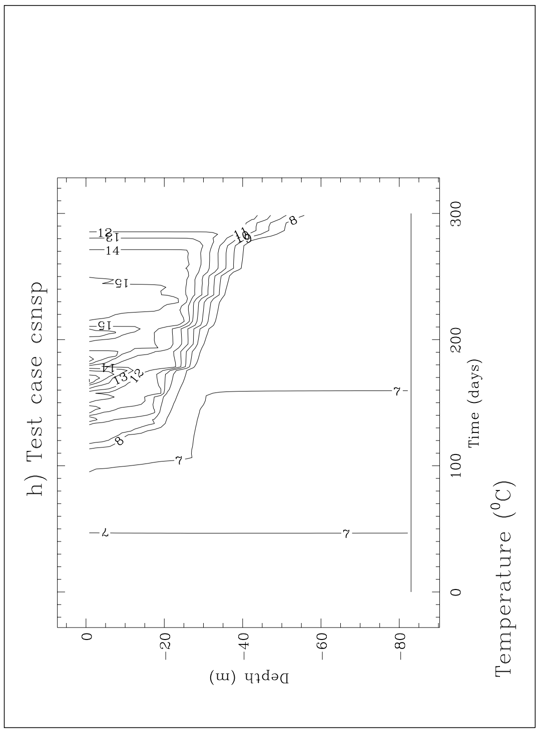
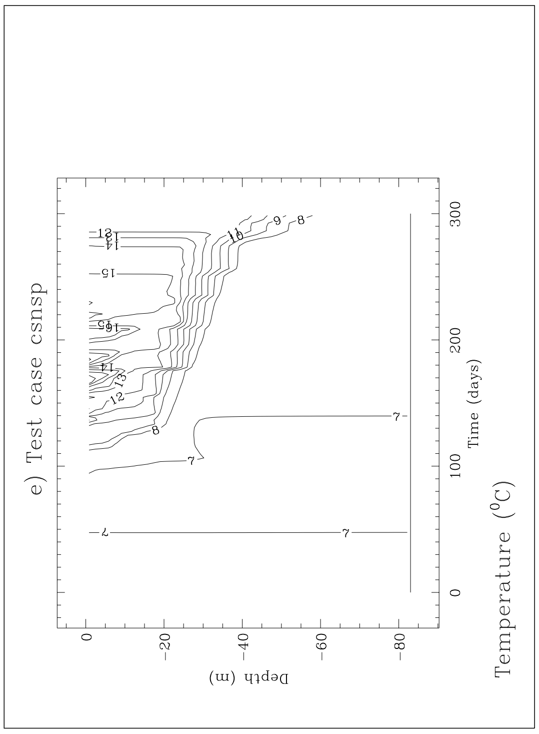


Figure 2.3.5: Depth-time contour plots of the temperature distributions at station CS averaged over 3 days for experiment A (a), B (b), C (c), D (d), E (e), F (f), G (g), H (h).







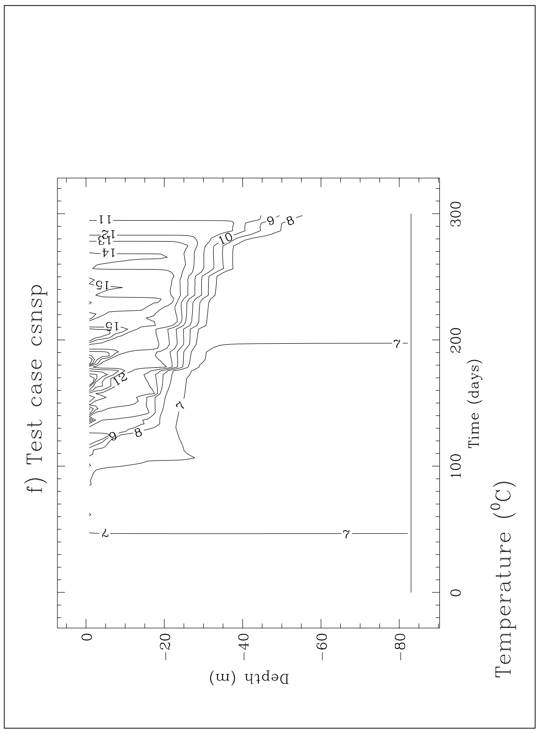


Figure 2.3.5: continued.

Chapter 4

Plume Test Cases

4.1 Test case river

4.1.1 Description of the problem and model setup

A frequently occurring phenomenon in coastal areas and estuaries is the presence of salinity fronts due to the fresh water input by river discharges. The cross-frontal density gradients create the well-known estuarine circulation (see e.g. Heaps, 1972; Garvine, 1974; Officer, 1976) consisting of a surface outflow, a bottom inflow and downwelling motions at the front. The aim of test case **river** is to simulate the circulation and the evolution of a tidally modulated estuarine front and to analyse the influence of different schemes implemented in the program.

The problem is simplified by considering a non-rotating channel of uniform depth with open ends. Lateral effects are ignored. The channel length and depth are taken as 140 km and 20 m respectively. The initial position of the front, shown in Figure 2.4.1a, is represented through the following distribution:

$$\begin{array}{lll} S = 30 & \text{if} & x_1 \leq 25 \\ S = \frac{1}{4}(x_1 - 25) + 30 & \text{if} & 25 < x_1 < 45 \\ S = 35 & \text{if} & x_1 \geq 45 \end{array} \tag{2.4.1}$$

where S is the salinity in PSU and x_1 the distance from the left boundary of the channel in km.

A tidal forcing is imposed by specifying the incoming Riemann variable at the left boundary in harmonic form:

$$R_+ = \overline{U} + c\zeta = 2cF_{har} = 2cA\cos(\omega t + \pi/2) \tag{2.4.2}$$

where $c = \sqrt{gH}$ is the barotropic wave speed, H the water depth, $\overline{U}$ the depth-integrated current, $\omega = 2\pi/12$ rad/h the semi-diurnal frequency and A the forcing amplitude taken as 0.8 m. The meaning of the harmonic function F_{har} is explained in Section III-1.6.3a. The amplitude A is stored into the setup array AMPOBU, the phase $\pi/2$ (with the minus sign) into the setup array PHAOBU. At the right boundary, the normal gradient of the incoming Riemann variable R_- is assumed to vanish, i.e. $\partial R_-/\partial x_1 = 0$. This allows the tidal wave to propagate freely from left to right. Since salinity remains vertically and horizontally homogeneous near the channel ends during the entire simulation, a zero normal gradient condition is selected for salinity at the two open boundaries. For a more detailed discussion about these open boundary conditions, see Section III-1.6.3.

The tidal current field within the simulated domain is initialised by splitting the simulation in two phases. During a first spin-up phase of 2 days the model is run without salinity. The salinity front is introduced at the start of the final run which takes 3 days. Running the test case **river** is performed as usual following the steps outlined in Section I-2.2 except that the shell script **Prepare** first copies the file *defmod_0.f* to *defmod.f*, compiles the program, runs the preprocessor and the main program for the spin-up phase, copies then *defmod_1.f* to *defmod.f*, recompiles the program and runs the preprocessor again to prepare the final run which is executed with the script **Run**.

Model setup parameters are further listed in Table 2.4.1.

4.1.2 Experiments and output parameters

Four experiments have been setup (see Table 2.4.2). The first three show the influence of the turbulence scheme while the fourth is intended to test the role of the advection scheme for momentum.

A : default schemes for turbulence and advection

B : Paconowski-Philander algebraic scheme for turbulence (see Section III-1.2.1a), default schemes for advection

C : the flow-dependent algebraic scheme for turbulence (see Section III-1.2.1b) using the expression (III-3.1.108) for the flow factor α, default schemes for advection

D : default scheme for turbulence, upwind scheme for the advection of momentum.

The final run is performed for 6 tidal cycles. The evolution of the front and the current field is shown in Figures 2.4.1b–f for the last cycle and the default scheme A at intervals of 3 hours. Use is made of the program's utility to perform an harmonic analysis. Residuals, amplitudes and phases of the current and salinity are calculated by the program using a daily period. The distributions of the residual fields, obtained with scheme A, are plotted in Figures 2.4.2a–c at daily intervals. The results for the last day are compared with those obtained for the other three experiments in Figures 2.4.2d–f. The non-analysed distributions of current and salinity at the end of the simulation are compared for the four experiments in Figures 2.4.3a–d. Finally, the time evolution of the position of the surface front, measured by the parameter HFRONT (see below), is displayed in Figure 2.4.4 for the four experiments. Note that the lower 10 m has been omitted in Figures 2.4.1b–f.

The following output parameters are defined:

HFRONT

The location of the point where the front outcrops the surface, measured by the distance in km of the 34 PSU contour line (taken at the first grid cell below the surface) to the left boundary.

PDEP

The depth of the halocline measured 5 km to the left of the surface front. This is determined as the depth in m of the point where the salinity first equals 34.5 PSU.

HGRAD

This parameter measures the strength of the horizontal salinity gradient below the surface layer and is defined as the maximum value of $|dS/dx_1|$ in PSU/km along a layer halfway between the surface and the bottom. Note that the initial value, according to (2.4.1), is given by 0.25 PSU/km.

VGRAD

This parameter measures the sharpness of the halocline and is defined as the maximum value of $|dS/dx_3|$ in PSU/m along a vertical profile at 5 km to the left of the surface front.

The parameters are listed in Table 2.4.3. The table additionally contains a series of harmonically analysed current values. The analysis is performed for the last day of the simulation at x_1=70 km, in the middle of the channel. The following parameters are defined:

SURES

Residual value of the surface current in m/s.

SUAMP

Amplitude of the surface current in m/s.

SUPHA

Phase of the surface current in degrees.

BURES

Mid-depth value of the current in m/s.

BUAMP

Mid-depth amplitude of the current in m/s.

BUPHA

Mid-depth phase of the current in degrees.

4.1.3 Results

The evolution of the frontal system for experiment A, as shown in Figures 2.4.1b–f and 2.4.2a–c can be described as follows. The initial configuration represents a non-equilibrium state. Although the model setup is different from the **seich** problem, where turbulent diffusion is absent, there is some similarity. A circulation is generated with a surface flow to the right, a bottom flow to the left, downwelling at the right edge and upwelling at the left edge of the front, as can be observed from the residual field in Figures 2.4.2a. Contrary to the **seich** test, the current is generated by a balance between the baroclinic pressure gradient and the vertical diffusion term, induced by tidal friction (Officer, 1976). The initial evolution therefore consists of a slight displacement of the surface front to the right and the bottom front to the left. The circulation pattern is modulated by the tide so that at the end of the cycle the front almost returns to its original position. The turbulent diffusion term, however, varies with the tide. This means that turbulence is suppressed during the phases where the current reverses sign and the shear is low. At that time the balance in the momentum equation is now mainly between the baroclinic pressure and inertia terms as in the **seich** problem giving a net displacement of the surface layer to the right. The surface and bottom layer are separated by a halocline of increasing strength. The vertical stratification reduces the turbulence even further and enhances the motion of the surface layer to the right. Figure 2.4.4 shows that the front moves with an almost uniform mean speed on which tidal modulations are superimposed. As the front moves further out downwelling at the front due to frontal convergence, first increases and then decreases. The main aspects of the physical analysis can be observed in Figures 2.4.2a–c and Figure 2.4.4.

The upwelling and downwelling motions are modulated by the tide which generates in combination with the vertical density gradient, an internal tide. The wavelength of the tide (seen in Figures 2.4.1b–f), is ~ 5 km.

The results for the other three experiments can be summarised as follows:

- Experiment B uses the Paconowski-Philander scheme. Contrary to the default scheme, turbulence is less reduced in the case of a strong stable stratification. The result is a more diffusive halocline and a slower outward expansion of the front.

- The flow-dependent scheme of experiment C is much more diffusive. The front returns to its initial position after each cycle so that no halocline forms and the front does not expand outwards. The test clearly shows the importance of the turbulence scheme in the simulation of estuarine fronts.

- Experiment D uses the same turbulence model as scheme A but advects the current with the more diffusive upwind scheme. The results are a reduction of the inertial force and horizontal density gradient and a slower expansion speed of the surface front.

Table 2.4.1: Model setup for test case **river**.

parameter	value	purpose
param.inc		
NC	140	number of grid points in the X-direction
NR	2	number of grid points in the Y-direction
NZ	20	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NVPROF	1	number of profiles at open boundaries
NCON	1	number of tidal frequencies
NCONTO	1	number of frequencies for harmonic analysis
DEFCON		
IOPTSA	0	update of the salinity field disabled (spin-up phase)
	1	update of the salinity field enabled (final run)
IOUTS	0	time series output disabled (spin-up phase)
	1	time series output enabled (final run)
IOUTT	0	harmonic analysis disabled (spin-up phase)
	1	harmonic analysis enabled (final run)
IBDATE	01010000	start date of the spin-up phase (MMDDHHMM)
	01030000	start date of the final run (MMDDHHMM)
IEDATE	01030000	end date of the spin-up phase (MMDDHHMM)
	01060000	end date for the final run (MMDDHHMM)
DELT	30	time step for the external (2-D) mode (s)
IC3D	10	time step for the internal (3-D) mode divided by DELT
ICRFORM	'U'	files with initial/final conditions written in binary format
SREF	32.5	reference salinity (PSU)
CDZ0UN	0.006	uniform roughness length used for the evaluation of the bottom stress (m)
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the horizontal grid (m)
GX0(NC+1)	140000	X-coordinate of the upper right corner of the horizontal grid (m)
GY0(1)	0	Y-coordinate of the lower left corner of the horizontal grid (m)
GY0(NR+1)	2000	Y-coordinate of the upper right corner of the horizontal grid (m)
DEP(NR,NC)	20	array of uniform water depths (m)

Table 2.4.1: continued.

parameter	value	purpose
DEFOB2D		
ITYPOBU(1-2)	1	$\overline{U} + c\zeta$ specified at the left boundary
ITYPOBU(3-4)	2	zero gradient condition at the right boundary
SIGMA(1)	1.4544E-04	semi-diurnal frequency for the tidal forcing (rad/s)
ANALSIG(1)	1.4544E-04	semi-diurnal frequency for harmonic analysis in the final run (rad/s)
AMPOBU(1-2,1)	0.8	amplitude of the tidal forcing (amplitude A of the harmonic function F_{har} in (2.4.2)) at the left boundary (m)
PHAOBU(1-2,1)	$-\pi/2$	phase of the forcing at the left boundary (rad)
DEFOB3D		
IVPOBU(1-4)	1	the same (zero gradient) condition is used for salinity at the open boundaries
DEFICS		
S(NZ,NR,NC)		initial salinity field as given by (2.4.1) for the final run (PSU)

Table 2.4.2: Definition of the **river** experiments.

Test	purpose
A	default schemes for turbulence and advection
B	Pacanowski-Philander scheme for turbulence (IOPTK = 1, ITFORM = 1), default schemes for advection
C	flow-dependent scheme for turbulence (IOPTK = 1, ITFORM = 3), default schemes for advection
D	default scheme for turbulence, upwind scheme for momentum (IADVC = 1)

Table 2.4.3: Output values for the parameters of the test case **river**.

parameter	time (hours)	A	B	C	D
Time series parameters					
HFRONT	12	46.49	47.25	43.58	45.75
	24	56.91	54.65	44.80	54.07
	36	70.09	63.11	45.89	64.86
	48	82.06	71.05	46.90	74.30
	60	93.62	78.39	47.84	83.31
	72	104.6	85.24	48.73	91.49
PDEP	12	20.29	16.06	20.30	20.32
	24	4.475	8.108	20.31	4.443
	36	2.889	5.488	20.31	3.352
	48	2.883	4.783	20.32	2.593
	60	2.915	4.508	20.32	2.469
	72	2.951	4.339	20.33	2.537
HGRAD	12	0.2568	0.2500	0.2516	0.2644
	24	0.3190	0.2882	0.2498	0.2892
	36	0.3563	0.3050	0.2470	0.3267
	48	0.4281	0.3226	0.2422	0.3666
	60	0.4279	0.3259	0.2366	0.3879
	72	0.4266	0.3149	0.2312	0.3596
VGRAD	12	0.3443	0.1622	0.0929	0.2905
	24	1.198	0.4893	0.1775	1.628
	36	2.649	0.9006	0.2228	2.362
	48	2.410	1.047	0.2312	3.230
	60	1.863	1.049	0.2162	2.763
	72	2.045	1.016	0.1916	2.215
Harmonic parameters					
SURES		0.3379	0.2094	0.0157	0.2618
SUAMP		0.5781	0.6225	0.5819	0.5672
SUPHA		327.5	327.9	310.8	328.7
BURES		-0.0212	0.0033	0.0131	-0.0228
BUAMP		0.5391	0.5673	0.5103	0.5383
BUPHA		303.5	302.1	306.2	304.1

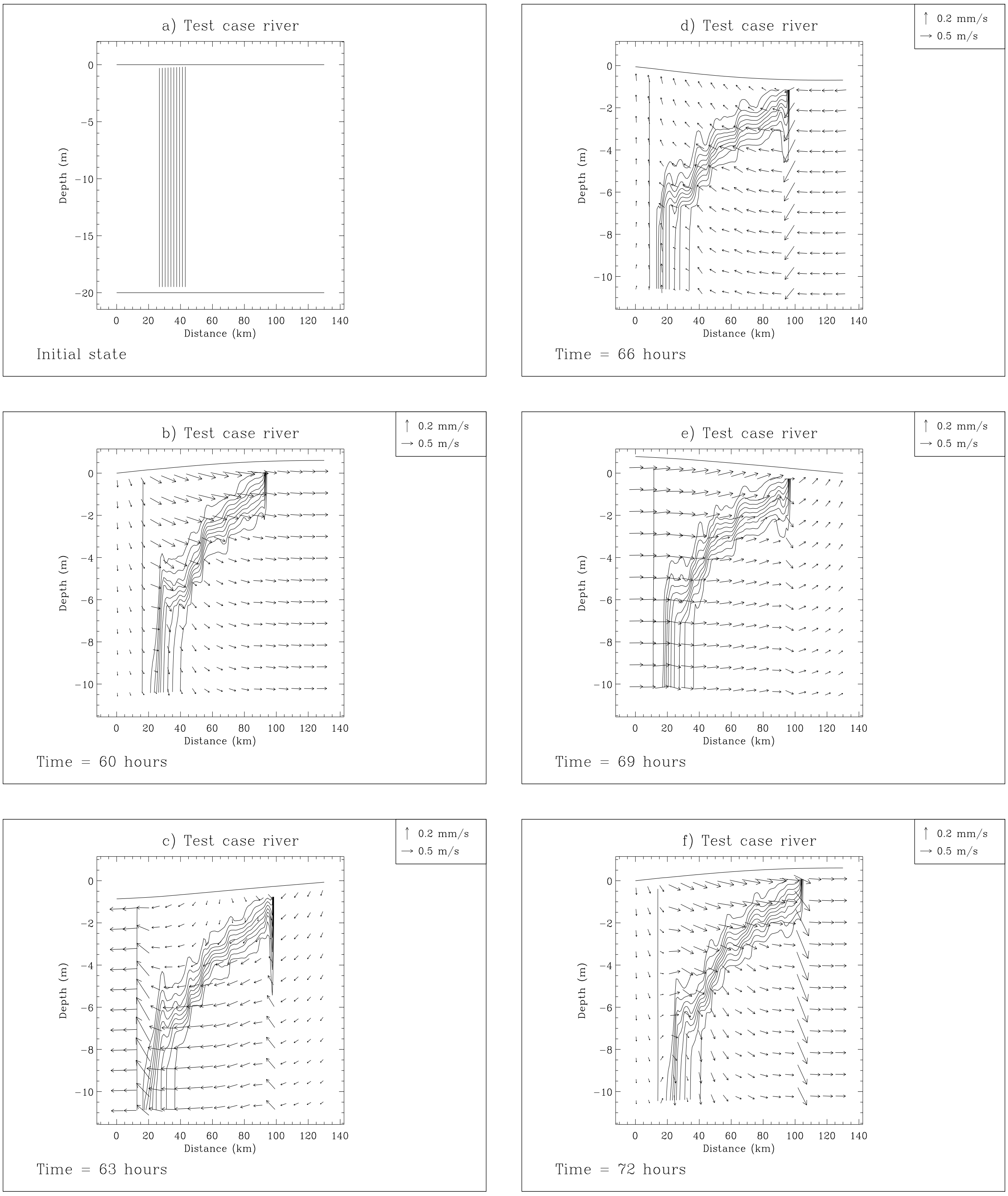


Figure 2.4.1: Initial position of the salinity front (a). Evolution of the current and salinity fields during the last tidal cycle for experiment A at intervals of 3 hours (b–f). Salinity contours are at intervals of 0.5 PSU.

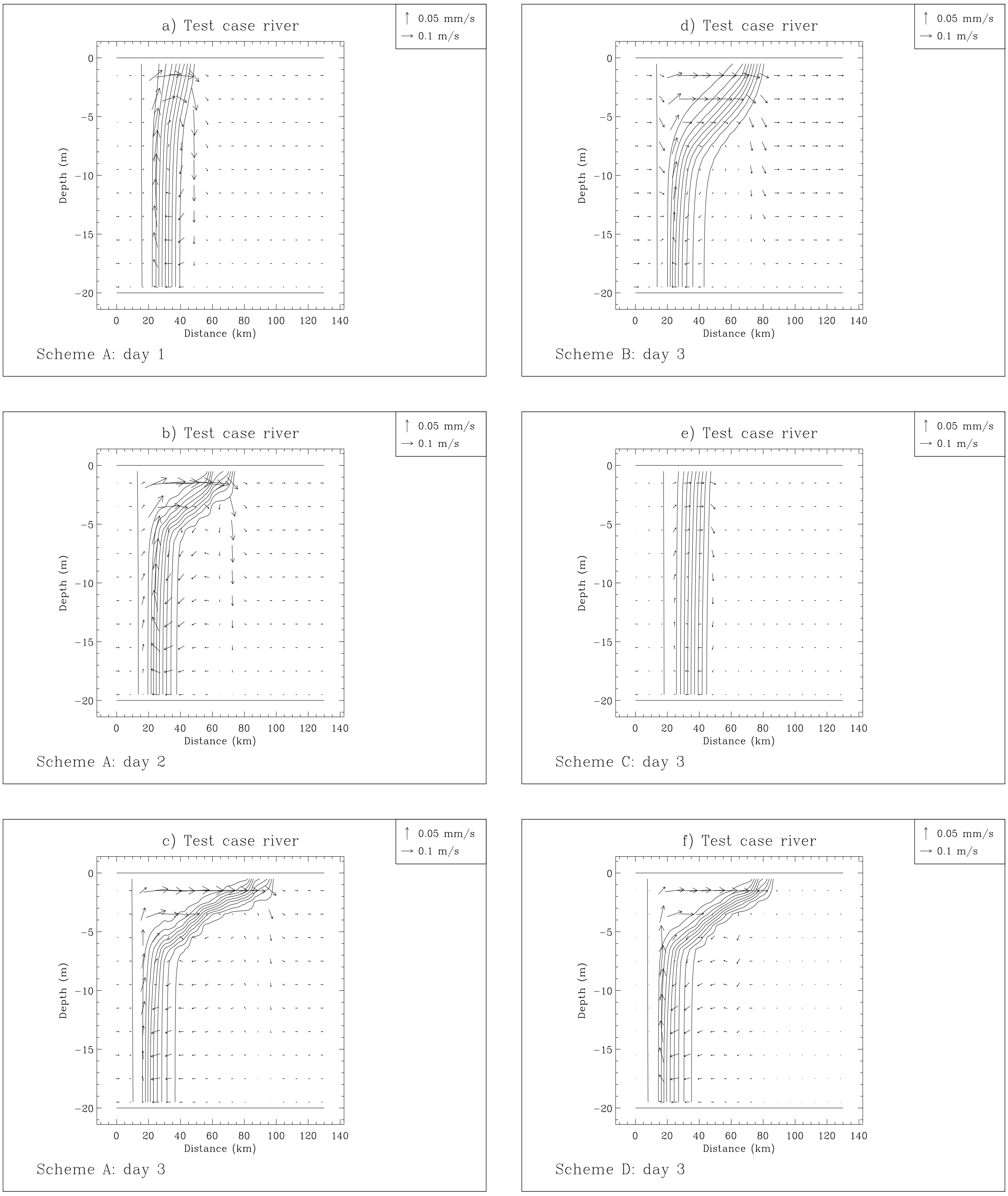


Figure 2.4.2: Residual salinity and current field for experiment A analysed for day 1 (a), 2 (b), 3 (c). The same now for day 3 and experiment B (d), C (e), D (f). Salinity contours are at intervals of 0.5 PSU.

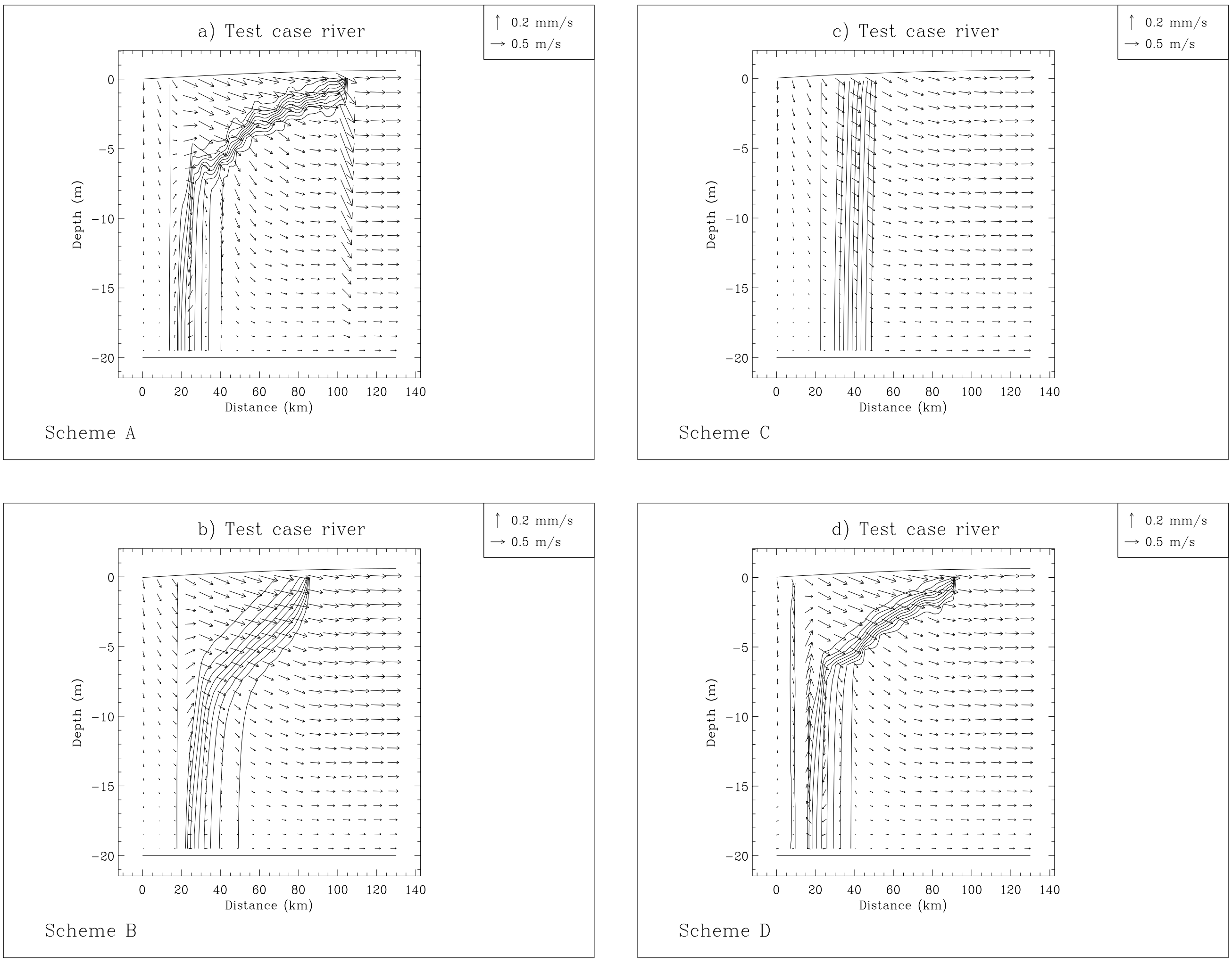


Figure 2.4.3: Distributions of current and salinity at the end of the simulation for experiment A (a), B (b), C (c), D (d). Salinity contours are at intervals of 0.5 PSU.

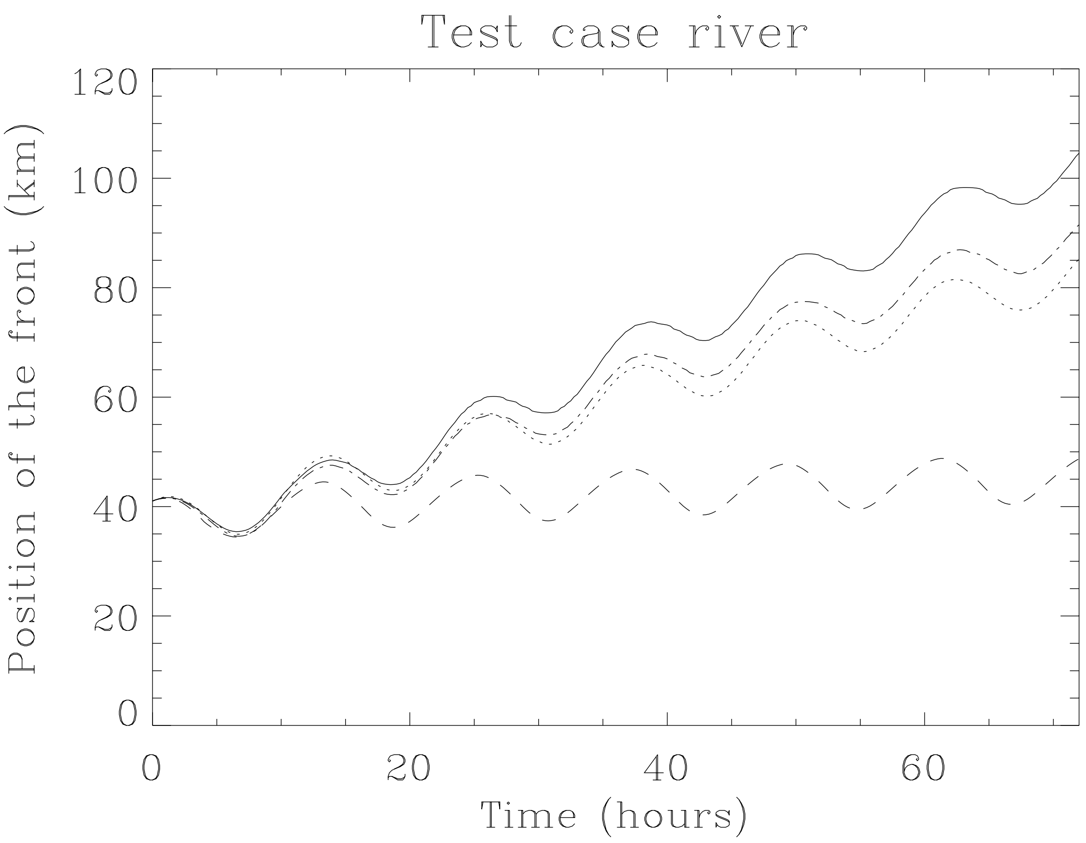


Figure 2.4.4: Time series of the distance to the left boundary of the point where the front outcrops the surface for experiment A (solid), B (dots), C (dashes), D (dash-dots).

- An interesting observation from Figures 2.4.3a–d is that only the first scheme is capable of representing the internal tide. This indicates again the importance of adequate schemes for advection and turbulence to model frontal processes.

4.2 Test case plume

4.2.1 Description of the problem and model setup

In the **river** test case the evolution of a salinity front was simulated within a narrow non-rotating channel. When a river discharges fresh water into the open sea, rotational effects become important and the problem becomes three-dimensional. The formation and evolution of river plumes has been examined numerically by e.g. Chao and Boicourt (1986), Chao (1988), Kourafalou et al. (1996) in the absence of tides and by Ruddick et al. (1995) who simulated the tidally modulated plume of the river Rhine. The general picture, emerging from these studies, is that the fresh water released at the river mouth, first expands seawards and then turns anticyclonically (i.e. to the right looking seawards in the northern hemisphere). Before reaching the coast again, the plume water deflects in the cyclonic sense forming a buoyancy driven coastal jet. The general form of the plume, as shown for example in Figure 2.4.7e, then consists of an anticyclonic bulge with a coastal plume to the right (in the northern hemisphere). The width of the plume is of the order of the baroclinic Rossby radius (∼5–10 km).

The aim of the test case **plume** is to simulate the evolution of a tidally modulated river plume using the idealised conditions of a uniform water depth and no wind forcing. The computational domain, shown in Figure 2.4.5, has the form of a rectangle enclosed by a coastal (solid) boundary and three open sea boundaries. For convenience, the former will be denoted by the southern boundary, the latter by the western, eastern cross-shore boundaries and the northern alongshore boundary. The basin has a length of 120 km, a width of 40 km and a depth of 20 m. The horizontal resolution is 1 km and 20 levels are used in the vertical. The area is filled initially with seawater having a uniform salinity of 33 PSU.

Tidal forcing is imposed in the form of a frictionless Kelvin wave entering at the western boundary and propagating along the coast (Ruddick et al., 1995). The incoming Riemann variable, specified at the western boundary, then takes the form

$$R_+ = \overline{U} + c\zeta = 2cF_{har} = 2cAe^{-fx_2/c}\cos\omega_2 t \qquad (2.4.3)$$

where the Coriolis frequency is evaluated at a latitude of 52^0, ω_2 is the M_2 tidal frequency, $A = 0.8$ m and $\overline{U}$, c, ζ are as before the depth-integrated alongshore current, the barotropic wave speed and the surface elevation. The amplitude of the wave decreases exponentially with distance to the coast with a decay scale given by the barotropic Rossby radius $c/f \simeq$ 120 km. The amplitude $Ae^{-fx_2/c}$ of the harmonic function F_{har} is stored into the setup array AMPOBU for each open boundary node. For more information about this type of open boundary condition see Section III-1.6.3a.

A zero normal gradient condition is selected at the eastern and northern boundaries, i.e.

$$\frac{\partial}{\partial x_1}(\overline{U} - c\zeta) = 0 \quad \text{and} \quad \frac{\partial}{\partial x_2}(\overline{V} - c\zeta) = 0 \tag{2.4.4}$$

The latter condition is justified by the fact that the width of the basin is much smaller than the external Rossby radius c/f.

In a first phase the model is run for a period of two days (almost four tidal cycles) without any stratification to initialise the current field and the sea surface elevation. At the start of the the final run fresh water of 10 PSU is released through an inlet at the southern boundary located 25.5 km from the western boundary (see Figure 2.4.5). The evolution of the plume is simulated for a period of three days. The two runs are set up using a file *defmod_0.f* for the initial and a file *defmod_1.f* for the final run. The procedures are the same as outlined in Section II-4.1.1 for the **river** test case.

Since the value of ζ is unknown at the river mouth, the open boundary condition at the inlet is no longer defined in terms of the incoming Riemann variable but by specifying the cross-shore component of the depth integrated current. This is given as the sum of a residual value, representing the river discharge, and a tidal component

$$\overline{V} = cF_{har} = Q_d/W + A_r H \cos(\omega t - \varphi_r) \tag{2.4.5}$$

where Q_d = 1500 m^3/s is the river discharge, W = 1 km the width of the inlet and A_r = 0.6 m/s the amplitude of the tidal current at the mouth of the river. The phase φ_r is determined by

$$\varphi_r = \omega D_r/c - \pi/2 \tag{2.4.6}$$

where D_r = 25.5 km so that D_r/c represents the time travelled by the Kelvin wave from the western boundary to the river mouth. Observations in the Rhine plume show that the alongshore and cross-shore component are in anti-phase which explains the use of the factor $\pi/2$. The residual part $Q_d/(Wc)$, the tidal amplitude A_rH/c and the phase φ_r of the harmonic function F_{har} are stored into the setup arrays AMPOBV and PHAOBV. See Section III-1.6.3a for more details about this type of open boundary condition.

In addition to the previous conditions for the 2-D mode, open boundary conditions have to be imposed during the final run for the horizontal velocity deviations (u', v')[1] and the salinity S. At the open sea boundaries a zero normal gradient (default) condition is taken for all quantities. In the case of salinity this procedure is a reasonable approximation since the plume never intersects the western and northern boundary while the cross-boundary gradient is much smaller than the along-boundary gradient at the eastern boundary.

The default conditions are no longer applicable at the river mouth where v' and S are specified in the form of a two-layer stratification

$$\begin{array}{ll} S = 10\,,\ v' = 0.6 & \text{if} \quad x_3 > -\delta \\ v' = -0.2 & \text{if} \quad -H \le x_3 \le -\delta \end{array} \tag{2.4.7}$$

[1]The velocity deviations are defined as the current minus its depth-averaged value.

where $\delta = 5$ m is the specified depth of the plume layer at the mouth. In this way fresh water is released through the surface layer whereas saltier seawater flows into the estuary through the bottom layer. A zero gradient condition is applied for salinity in the bottom layer.

Model setup parameters are listed in Table 2.4.4.

4.2.2 Experiments and output parameters

Although the **plume** problem is a valuable test to examine the role of different physical forcing mechanisms (bathymetry, tides, wind, ...) on the plume structure, the intention here is to test some of the schemes implemented in the program. Seven experiments are defined (see Table 2.4.5) testing the role of the different formulations for horizontal diffusion and different boundary conditions.

A : Uses default values and model setup listed in Table 2.4.4.

B : As experiment A now using the upwind scheme for the advection of momentum.

C : As experiment B now with horizontal diffusion enabled using the Smagorinsky formulation.

D : As experiment B now with horizontal diffusion enabled and using the uniform values $\nu_H = \lambda_H = 100$ m^2/s for the horizontal diffusion coefficients.

E : As experiment D but the boundary condition (2.4.3) is replaced by a condition for the surface elevation:

$$\zeta = Ae^{-fx_2/c}\cos\omega_2 t \tag{2.4.8}$$

F : As experiment D but the boundary condition (2.4.3) is replaced by a condition for the depth-integrated current:

$$\overline{U} = cAe^{-fx_2/c}\cos\omega_2 t \tag{2.4.9}$$

G : As experiment A but without tidal forcing which means that condition (2.4.3) is replaced by the zero gradient condition (2.4.4) at the western boundary and A_r is set to zero in the condition (2.4.5) at the inlet.

A series of figures have been prepared for this test case. The intention is not only to examine the structure and the evolution of the river plume and to compare the different experiments, but also to illustrate how harmonic analysis can be applied to model results. The time evolution of the surface plume for experiment A is shown in Figures 2.4.6a–e during the third cycle at intervals of 3 hours. The residual salinity and current field obtained for the last 12 hours of the simulation are compared in Figures 2.4.7a–e for experiments A, B, C, D and G. The results for E and F are similar to D and therefore omitted. The residual fields plotted in Figures 2.4.8a–b and 2.4.9a–b for experiments A and G are taken respectively along the transects AA' perpendicular and DD' parallel to the coast (see Figure 2.4.5 for a location of the transects). The time evolutions of the plume dimensions are compared in Figures 2.4.10a–b for several experiments. The first figure shows the plume width along transect AA', the second the length of the plume measured by the alongshore distance between the inlet and the point to the right along transect CC' where the salinity reaches a value of 32 PSU. Harmonically analysed values and current ellipse parameters are displayed in Figures 2.4.11a–b (surface and bottom ellipticity), 2.4.12a–c (major axis of the tidal ellipse for the depth-averaged current and experiments A, E, F) and 2.4.13a–c (amplitude of the surface elevation for experiments A, E, F).

A series of output parameters are defined. The first four are evaluated at intervals of 6 hours. For a location of the transects and points see Figure 2.4.5.

HBULGE
: The width of the plume bulge. This is measured by the maximum distance in km of the 32 PSU surface contour line from the coast.

HWIDTH
: The width of the coastal plume measured by the distance from the coast in km of the point where the 32 PSU surface contour line intersects the transect BB'.

PDEP
: Depth of the surface plume at point P measured by the distance to the surface in m of the 30 PSU contour line.

HFRONT
: The length of the plume measured by the alongshore distance in km between the point where the 32 PSU surface contour line intersects the transect CC' and the river mouth.

A second series of parameters represent harmonically analysed values of the current and surface elevation and tidal ellipse parameters at the two points P and Q. The analysis is performed for the last 12 hours of the simulation.

SURES
: Surface value of the residual u-current at P (m/s).

SVRES
: Surface value of the residual v-current at P (m/s).

SELMAJ
: Surface value of the tidal ellipse's major axis at P (m/s).

SELLIP
: Ellipticity of the surface ellipse at P.

BURES
: Value of the residual u-current at P and 10 m depth (m/s).

BVRES
: Value of the residual v-current at P and 10 m depth (m/s).

BELMAJ
: Value of the tidal ellipse's major axis at P and 10 m depth (m/s).

BELLIP
: Ellipticity of the tidal ellipse at P and 10 m depth.

DURES
: Residual value of the depth-averaged current at Q (m/s).

DUAMP
: Amplitude of the depth-averaged u-current at Q (m/s).

DUPHA
: Phase of the depth-averaged u-current at Q (degrees).

ZETRES
: Residual surface elevation at Q (m).

ZETAMP
: Amplitude of the sea surface elevation at Q (m).

ZETPHA
: Phase of the sea surface elevation at Q (degrees).

The residual values for experiment G are obtained by taking averaged values over the last 12 hours. All output parameters are listed in Table 2.4.6.

4.2.3 Results

- Figures 2.4.6a–e clearly show how the plume evolves during a tidal cycle. At the time when the alongshore current reverses sign and the outflow reaches its maximum, a new blob of fresh water enters the basin, moving seawards (Figure 2.4.6b). As the eastward directed tidal wave becomes stronger, the fresh water patch is deflected to the right (Figure 2.4.6c). During this phase of the tide both the bulge and the coastal plume expand seawards. When the tidal current reverses sign again and turns to the West, the current inside the plume is first southeastwards pushing the bulge towards the coast (Figure 2.4.6d) and finally southwestwards reducing the extent of the bulge and the coastal plume (Figure 2.4.6e). Since the tidal current enforces the anticyclonic motion inside the bulge, the results are similar to the ones obtained for non-tidal plumes. The main difference here is that the bulge and the coastal plume oscillate with the tide.

- After a few tidal cycles the structure of tidal and non-tidal plumes are manifestly different as can be seen by comparing Figures 2.4.7a and e. While in the latter case the transition between the bulge and the coastal plume is clearly observed, no clear distinction can be made in the former case. This effect is due to increased turbulent diffusivity which reduces the anticyclonic vorticity inside the bulge and the strength of the coastal current. The effect becomes more pronounced when more horizontal diffusion is introduced in the simulation either by using the upwind scheme for momentum advection (Figure 2.4.7b) or by adding horizontal diffusion terms in the momentum and salinity equations (Figures 2.4.7c and d).

- The residual fields along the two transects AA' and DD' show the presence of an estuarine-type circulation (Figures 2.4.8a–b and 2.4.9a–b). In the cross-shore transect upwelling takes place at the coast while downwelling occurs at the edge of the plume by the convergence of the surface outflow current. A similar phenomenon is seen in the coastal jet where downwelling motions are created by the convergence of the coastal jet. In the case of a non-tidal plume the plume layer is shallower and the frontal gradients are stronger compared to the tidal case where turbulent diffusion increases the depth of the surface layer and reduces the vertical stratification.

- The oscillations of the plume shape induced by the tide are clearly observed in Figures 2.4.10a–b. In the absence of tides the width of the bulge first grows in time levelling off after 10–12 hours which is comparable to the inertial period of ~ 15 h. A somewhat similar behaviour can be deduced for the tidal case where the mean growth curve is strongly modulated by the tide. The length of the plume increases linearly in the non-tidal case. In the presence of tides the growth is strongly reduced as found previously and seems to decrease asymptotically. Horizontal diffusion limits the growth of the coastal plume.

- The presence of a stratified surface layer has an important impact on the form of the tidal current. Outside the plume the current is rectilinear (zero ellipticity) while inside the plume the current rotates anticyclonically (negative ellipticity) in the surface and cyclonically (positive ellipticity) in the bottom layer (Figures 2.4.11a–b). This is confirmed by observational data in the Rhine plume and the physical theory discussed in Visser et al. (1994).

- The amplitude of the depth-averaged tidal current measured by the major axis of the tidal current (Figures 2.4.12a–c) is clearly stronger inside the plume due to the reduction of the turbulence and the bottom shear stress. No significant change is found for the amplitude of the surface elevation (2.4.13a–c).

- Figures 2.4.12 and 2.4.13 also compare the three different formulations for the tidal forcing at the western boundary, represented by experiments A, E and F. Although the results are qualitatively the same, there are important quantitative differences. On the other hand, the results shown in Figures 2.4.11, 2.4.12a and 2.4.13a are practically

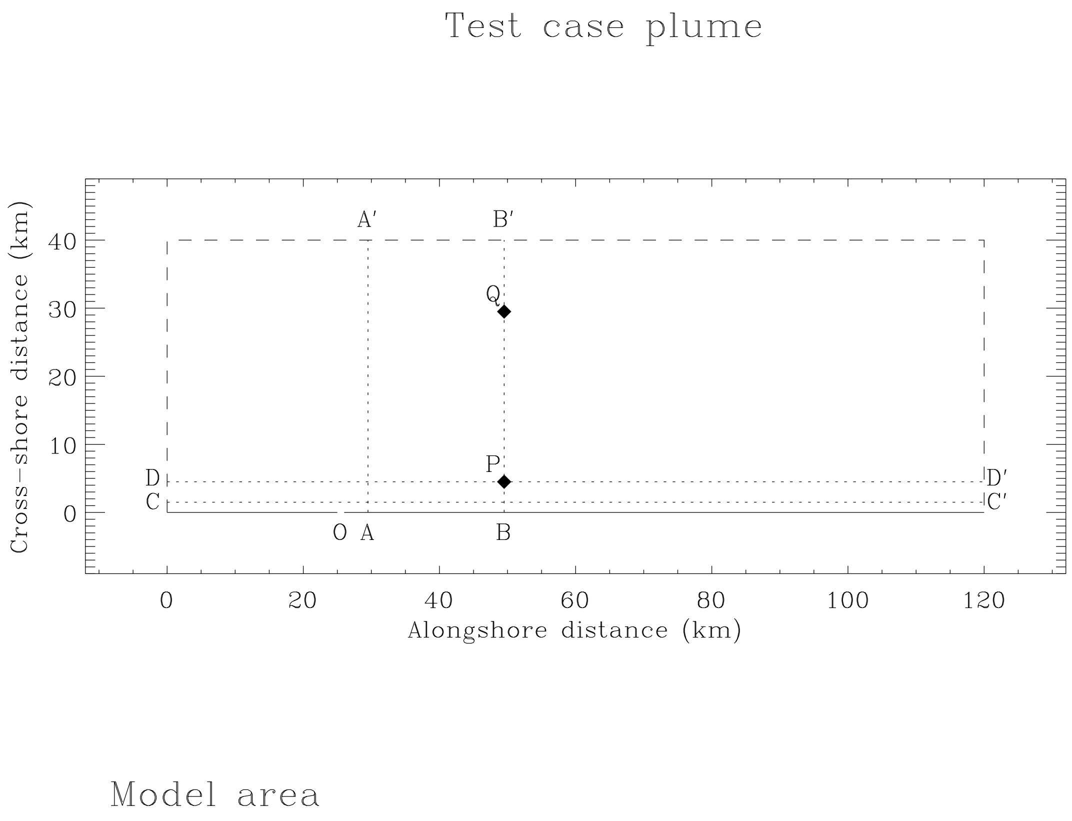


Figure 2.4.5: The computational domain for the test case **plume**. The solid line(s) indicate the coastal (southern) boundary, the dashed lines the open sea boundaries and the dotted lines the two cross-shore transects AA' ($x_1 = 30$), BB' ($x_1 = 50$) and the two alongshore transects CC' ($x_2 = 2$) and DD' ($x_2 = 5$). The inlet is located at O. Points P and Q are used for the evaluation of a number of output parameters (see text).

the same for experiments A, B, C and D suggesting that horizontal diffusion has no significant influence on the tidal currents (at least for this idealised test case).

Table 2.4.4: Model setup for the test case **plume**.

parameter	value	purpose
param.inc		
NC	120	number of grid points in the X-direction
NR	40	number of grid points in the Y-direction
NZ	20	number of grid points in the vertical
NOBU	80	number of open boundary points in the X-direction (western and eastern boundaries)
NOBV	121	number of open boundary points in the Y-direction (northern boundary and river mouth)
NVPROF	2	number of profiles at open boundaries (zero gradient condition at open sea boundaries and external value at the river mouth)
NCON	1	number of tidal frequencies
NCONTO	1	number of frequencies for harmonic analysis
DEFCON		
IOPTSA	0	update of the salinity field disabled (spin-up phase)
	1	update of the salinity field enabled (final run)
IOUTS	0	time series disabled (spin-up phase)
	1	time series output enabled (final run)
IOUTT	0	harmonic analysis disabled (spin-up phase)
	2	harmonic analysis and calculation of tidal ellipses enabled (final run)
IBDATE	01010000	start date for the spin-up phase (MMDDHHMM)
	01030000	start date for the final run
IEDATE	01030000	end date for the spin-up phase (MMDDHHM)
	01060000	end date for the final run
DELT	30	time step for the external (2-D) mode (s)
IC3D	10	time step for the internal (3-D) mode divided by DELT
ICRFORM	'U'	file with initial/final conditions written in binary format
DLAREF	52.0	reference latitude (degrees)
HEDVUN	100.0	uniform horizontal diffusion coefficient for momentum (experiments D, E, F)
HEDDUN	100.0	uniform horizontal diffusion coefficient for salinity (experiments D, E, F)
CDZ0UN	0.006	uniform roughness length used for the evaluation of the bottom stress (m)

Table 2.4.4: continued.

parameter	value	purpose
DEFGRID		
GX0(1)	0	X-coordinate of the lower left corner of the basin (m)
GX0(NC+1)	120000	X-coordinate of the upper right corner of the basin (m)
GY0(1)	0	Y-coordinate of the lower left corner of the basin (m)
GY0(NR+1)	40000	Y-coordinate of the upper right corner of the basin (m)
DEP(NR,NC)	20	array of uniform water depths (m)
NPIX		cell face pointer array at U-faces (1 at internal faces and 2 at the western/eastern open sea boundaries)
NPIY		cell face pointer array at V-faces (0 at the southern coastal boundary, 1 at internal faces, 2 at the northern boundary, 3 at the mouth)
DEFOB2D		
ITYPOBU		type of open boundary condition (2-D mode) at western/eastern open sea boundaries (1, 2, 3 or 4 if the condition is of the form given by (2.4.3), (2.4.4), (2.4.8) or (2.4.9))
ITYPOBV		type of open boundary condition (2-D mode) at the southern/northern boundary (2 at the northern boundary, 4 at the river mouth)
SIGMA(1)	1.4056E-04	M_2-tidal frequency (rad/s)
ANALSIG(1)	1.4056E-04	M_2-tidal frequency for harmonic analysis in the final run (rad/s)
AMPOBU(1:40,1)		amplitude of the Kelvin wave (defined via the harmonic function F_{har}) at the western boundary as given by the factor $Ae^{-fx_2/c}$ in equation (2.4.3) (m)
AMPOBV(121,0)		residual part of the harmonic function F_{har} at the mouth as given by $Q_d/(Wc)$ (m)
AMPOBV(121,1)		amplitude of the tidal forcing (represented by the function F_{har}) at the mouth given by the factor A_rH/c in equation (2.4.5)
PHAOBV(121,1)		phase of the tidal current at the mouth as given by (2.4.6)

Table 2.4.4: continued.

parameter	value	purpose
DEFOB3D		
IVPOBU(1:80)	1	the same (zero gradient) condition is used for salinity and current at the western/eastern boundaries
IVPOBV(1:120)	1	the same (zero gradient) condition is used for salinity at the northern boundary
IVPOBV(121)	2	a scalar condition (see equation (2.4.7)) is used at the mouth
UVP(1-NZ,2)		the two-layer profile for v' at the mouth as given by (2.4.7)
SVP(1-NZ,2)		the two-layer profile for S at the mouth as given by (2.4.7)

Table 2.4.5: Definition of the **plume** experiments.

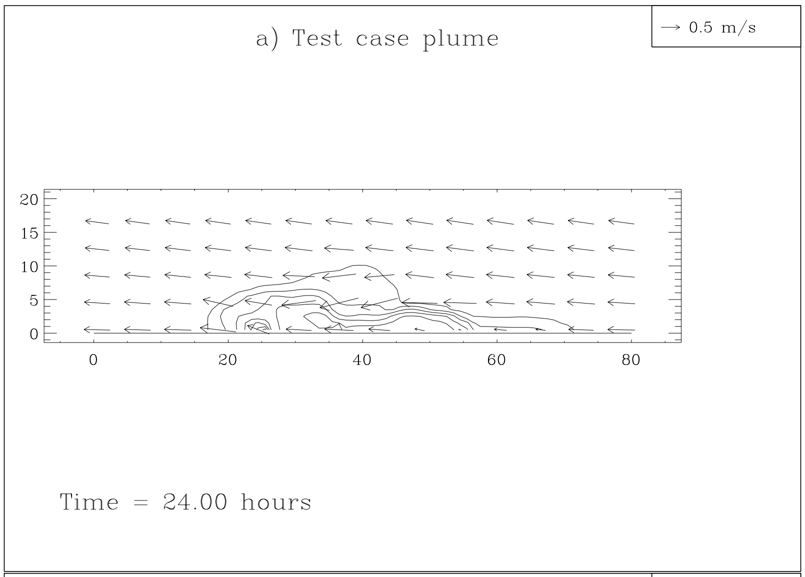
Test	purpose
A	uses default schemes and condition (2.4.3) at the western boundary
B	as experiment A now using the upwind scheme for momentum (IADVC = 1)
C	as experiment A now using the upwind scheme for momentum (IADVC = 1) and the Smagorinsky formulation for horizontal diffusion (IODIF = 2)
D	as experiment A now using the upwind scheme for momentum (IADVC = 1) and horizontal diffusion (IODIF = 1) with the uniform diffusion coefficients $\nu_H = \lambda_H = 100$ m^2/s
E	as experiment D now using condition (2.4.8) at the western boundary
F	as experiment D now using condition (2.4.9) at the western boundary
G	as experiment A but without tidal forcing

Table 2.4.6: Output values for the parameters of the test case **plume**.

parameter	time (h)	A	B	C	D	E	F	G
Time series parameters								
HBULGE	6	11.32	10.86	10.83	10.51	9.950	11.31	10.14
	12	7.622	7.459	7.420	7.492	7.681	9.196	13.15
	18	13.36	13.13	13.08	13.22	12.76	14.04	13.34
	24	9.966	9.481	9.160	9.344	8.868	11.14	14.46
	30	13.46	13.49	13.49	14.21	14.12	15.22	15.17
	36	11.04	10.86	10.40	10.85	10.49	12.87	15.41
	42	13.46	14.12	14.18	14.48	14.72	16.02	16.28
	48	12.35	11.84	11.44	11.97	11.83	14.34	16.42
	54	13.38	14.24	14.27	14.80	15.21	16.36	17.15
	60	13.18	12.49	12.34	12.95	12.68	15.63	17.45
	66	13.22	14.24	14.29	14.65	15.30	16.58	17.87
	72	13.99	13.55	13.36	13.81	13.45	16.79	18.32
HWIDTH	6	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	12	3.539	3.966	3.585	2.587	0.0	1.547	0.0
	18	8.107	9.106	9.199	9.391	9.499	7.433	3.100
	24	4.353	4.490	4.964	5.485	5.190	5.979	3.887
	30	10.29	11.32	11.75	11.81	12.16	10.94	4.181
	36	6.638	8.619	9.058	9.257	8.276	10.63	4.364
	42	11.21	12.24	12.40	12.50	13.16	12.33	4.356
	48	9.086	11.40	11.11	11.28	10.55	13.26	4.401
	54	11.18	12.31	12.46	12.90	13.39	13.19	4.796
	60	10.41	12.28	12.25	12.66	12.06	15.12	5.094
	66	11.30	12.33	12.39	12.74	13.38	13.56	5.220
	72	11.95	13.44	13.36	13.73	13.21	16.46	5.458
PDEP	6	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	12	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	18	0.8346	1.208	1.245	1.232	1.186	0.9342	0.0
	24	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	30	1.469	1.521	1.574	1.731	1.678	1.904	0.0
	36	0.0	0.6162	1.027	1.420	1.889	2.444	0.0
	42	2.096	1.875	1.968	2.181	2.093	2.490	0.0
	48	0.0	1.235	1.413	1.956	2.274	3.322	0.0
	54	2.452	2.084	2.162	2.445	2.303	2.956	0.0
	60	1.145	1.336	1.435	2.144	2.243	3.561	0.0
	66	2.731	2.259	2.315	2.778	2.480	3.377	0.7360
	72	1.871	1.438	1.490	2.289	2.259	3.627	1.268

Table 2.4.6: continued.

parameter	time (h)	A	B	C	D	E	F	G
Time series parameters								
HFRONT	6	18.84	18.90	18.74	18.60	19.88	15.88	10.01
	12	30.32	27.80	26.93	24.92	22.94	24.04	20.92
	18	37.29	34.54	33.59	30.53	31.15	28.72	31.86
	24	44.46	41.99	40.75	34.70	32.24	34.53	42.71
	30	46.81	43.35	42.03	35.89	36.80	35.50	53.73
	36	53.82	52.67	51.54	43.37	41.46	44.20	64.76
	42	54.71	52.43	50.96	41.30	42.01	42.57	75.71
	48	55.48	61.54	60.35	50.28	48.27	52.00	86.81
	54	57.76	58.65	57.46	45.55	45.76	48.43	94.50
	60	57.04	67.14	67.36	56.19	53.88	59.08	94.50
	66	67.21	64.56	63.65	48.95	48.41	53.45	94.50
	72	61.94	74.99	75.02	60.82	58.39	65.02	94.50
Harmonic parameters								
SURES		0.2348	0.1554	0.1559	0.1357	0.1403	0.1497	0.7765
SVRES		-0.1516	-0.1072	-0.0997	-0.0821	-0.0818	-0.0673	-0.0819
SELMAJ		75.60	76.05	75.56	76.20	76.83	60.02	*
SELLIP		-0.4864	-0.4558	-0.4421	-0.4215	-0.4263	-0.4057	*
BURES		-0.0537	-0.0220	-0.0177	-0.0086	-0.0085	-0.0060	-0.0256
BVRES		0.0125	-0.0103	-0.0084	-0.0053	-0.0051	-0.0024	-0.0255
BELMAJ		42.19	42.78	42.76	41.28	42.12	33.97	*
BELLIP		0.2351	0.1772	0.1741	0.2032	0.2007	0.2265	*
DURES		0.0006	-0.0011	-0.0011	-0.0011	-0.0018	-0.0020	0.0034
DUAMP		0.5483	0.5472	0.5472	0.5473	0.5689	0.4533	*
DUPHA		34.50	34.94	34.93	34.82	18.30	46.64	*
ZETRES		-0.0084	-0.0025	-0.0025	-0.0021	-0.0061	*	0.0144
ZETAMP		0.6369	0.6357	0.6356	0.6360	0.6781	0.5068	*
ZETPHA		81.87	81.56	81.56	81.51	64.68	92.71	*



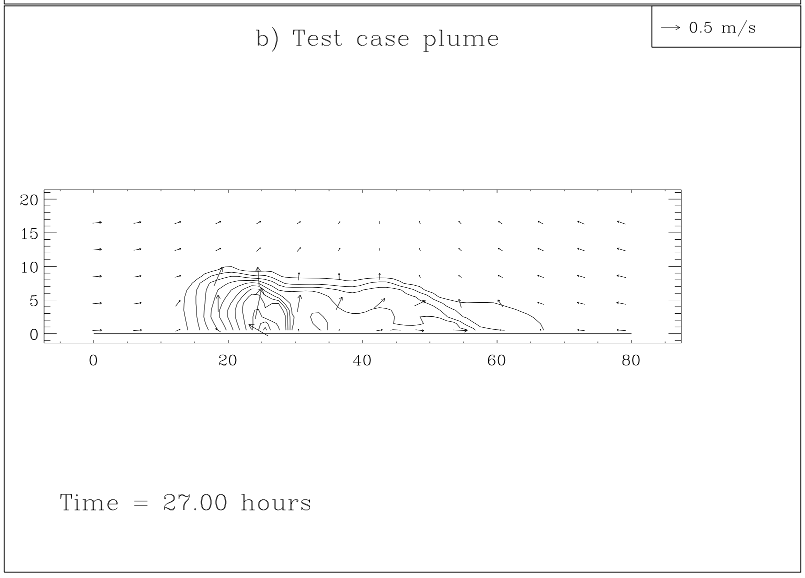
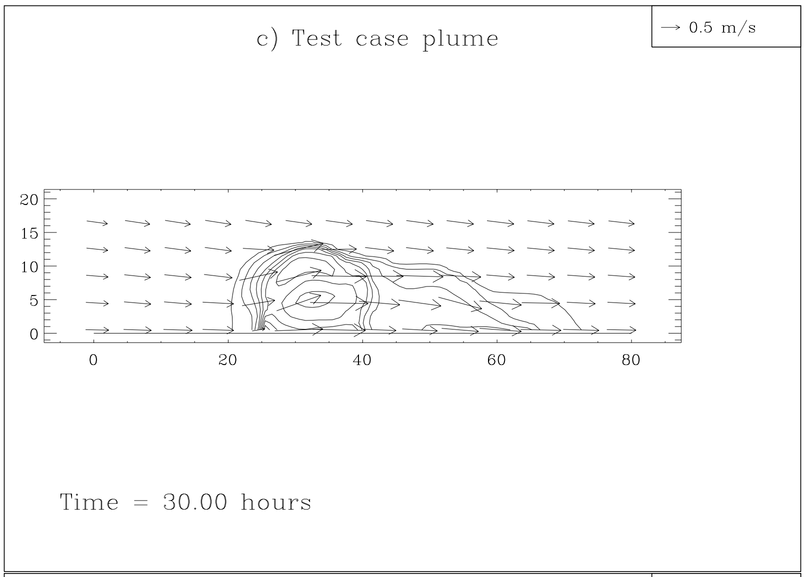
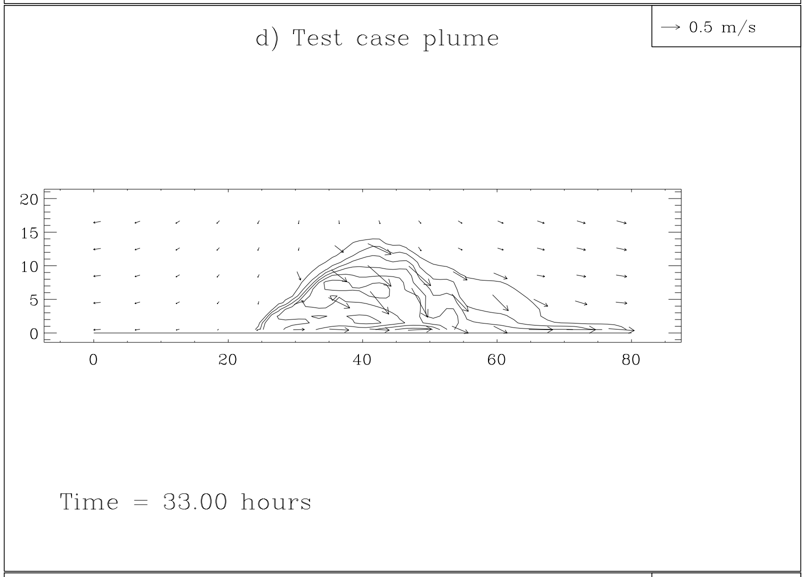


Figure 2.4.6: Surface distributions of currents and salinity for experiment A at 24 h (a), 27 h (b), 30 h (c), 33 h (d), 36 h (e). Contour intervals are at 2 PSU.





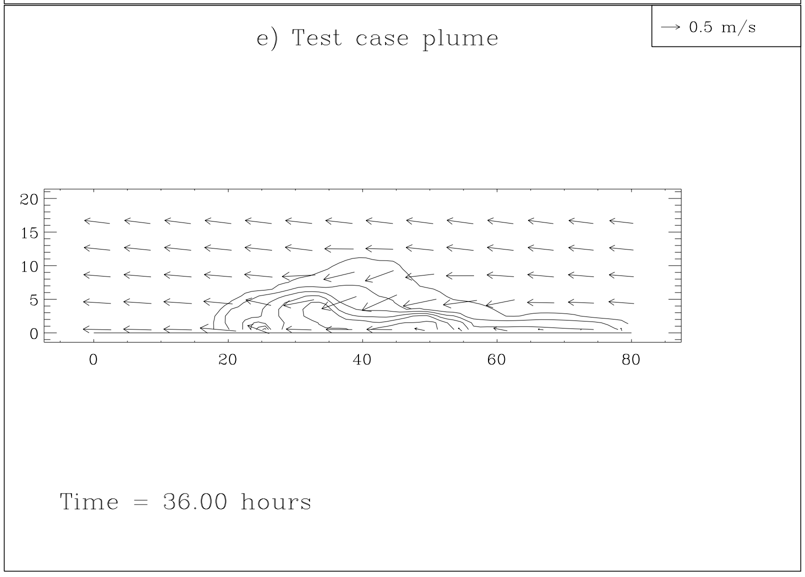
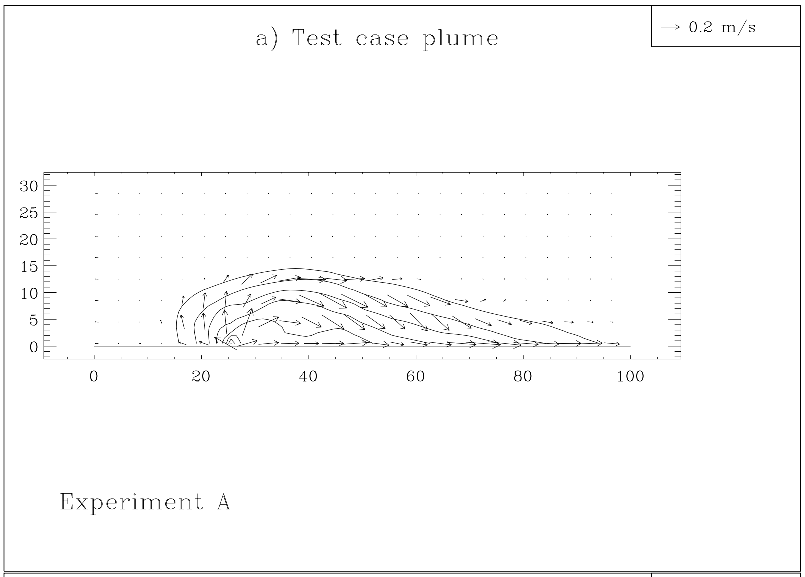


Figure 2.4.6: continued.



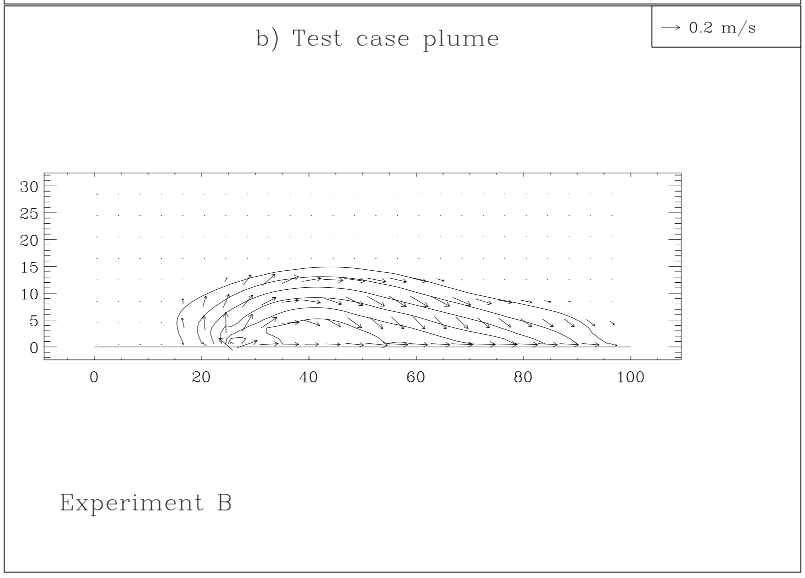


Figure 2.4.7: Residual surface distributions of currents and salinity for the last tidal cycle and experiment A (a), B (b), C (c), D (d), G (e). Contour intervals are at 2 PSU.

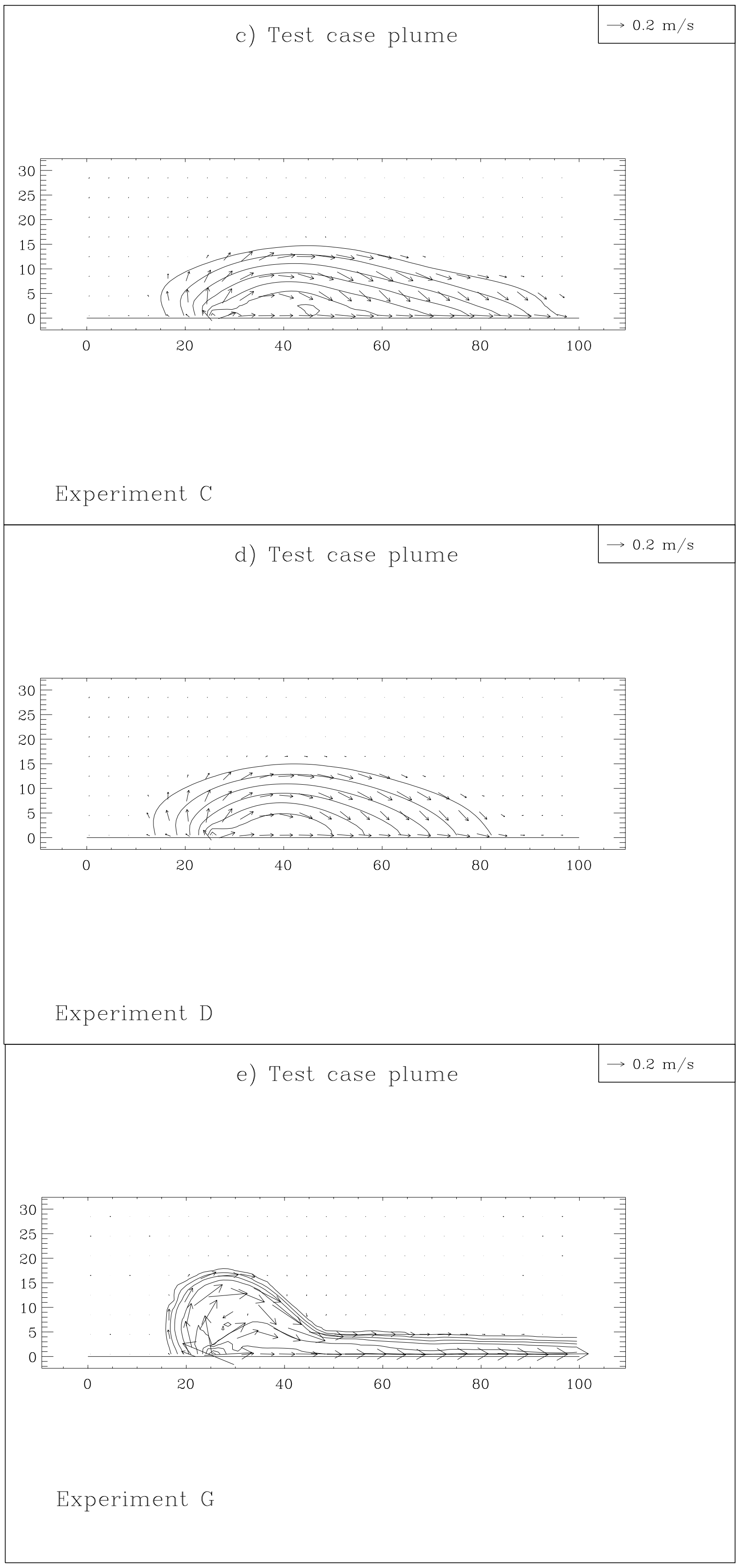


Figure 2.4.7: continued.

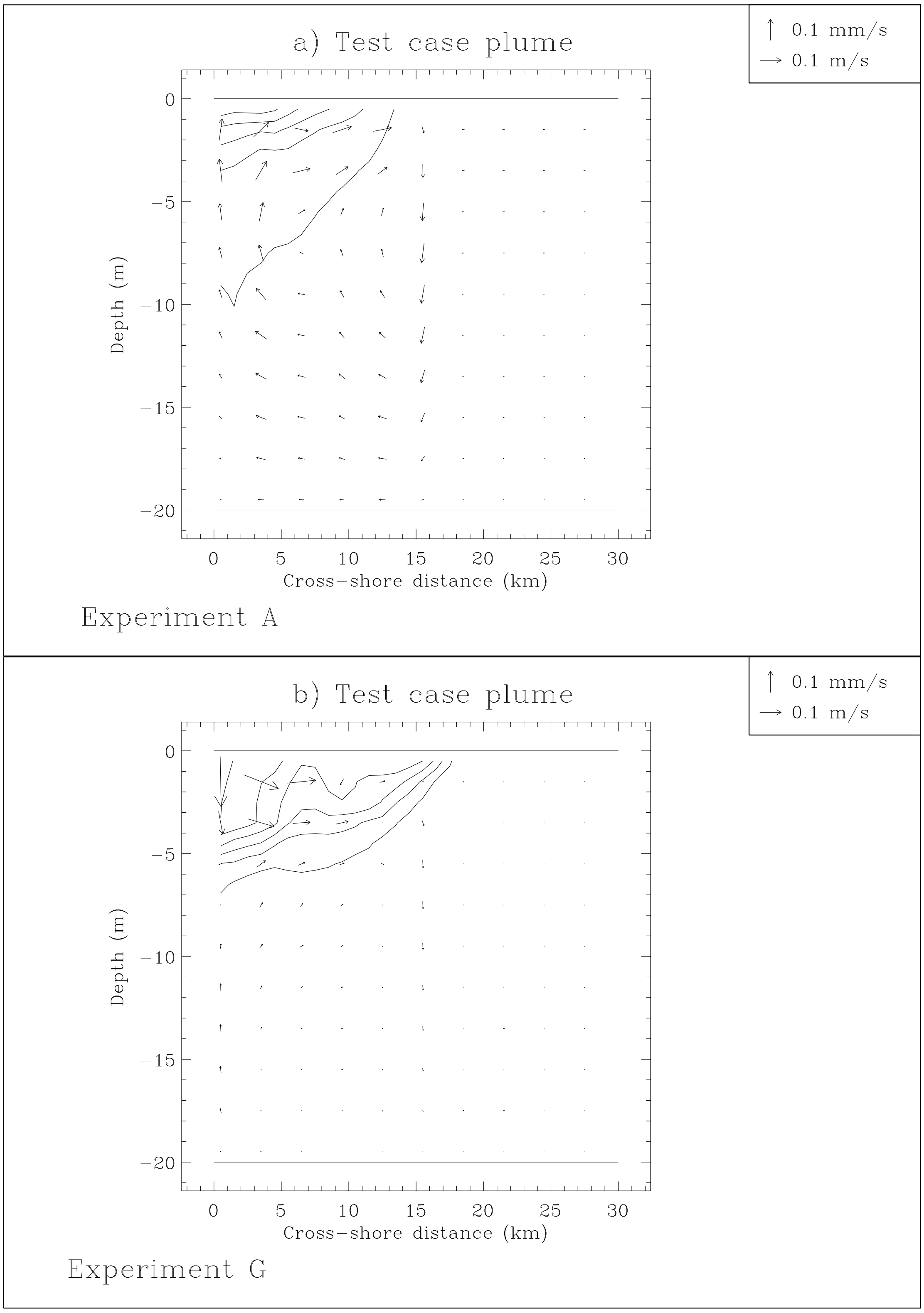


Figure 2.4.8: Residual distributions of currents and salinity for the last tidal cycle at the cross-shore transect AA': experiment A (a), G (b). Contour intervals are at 2 PSU.

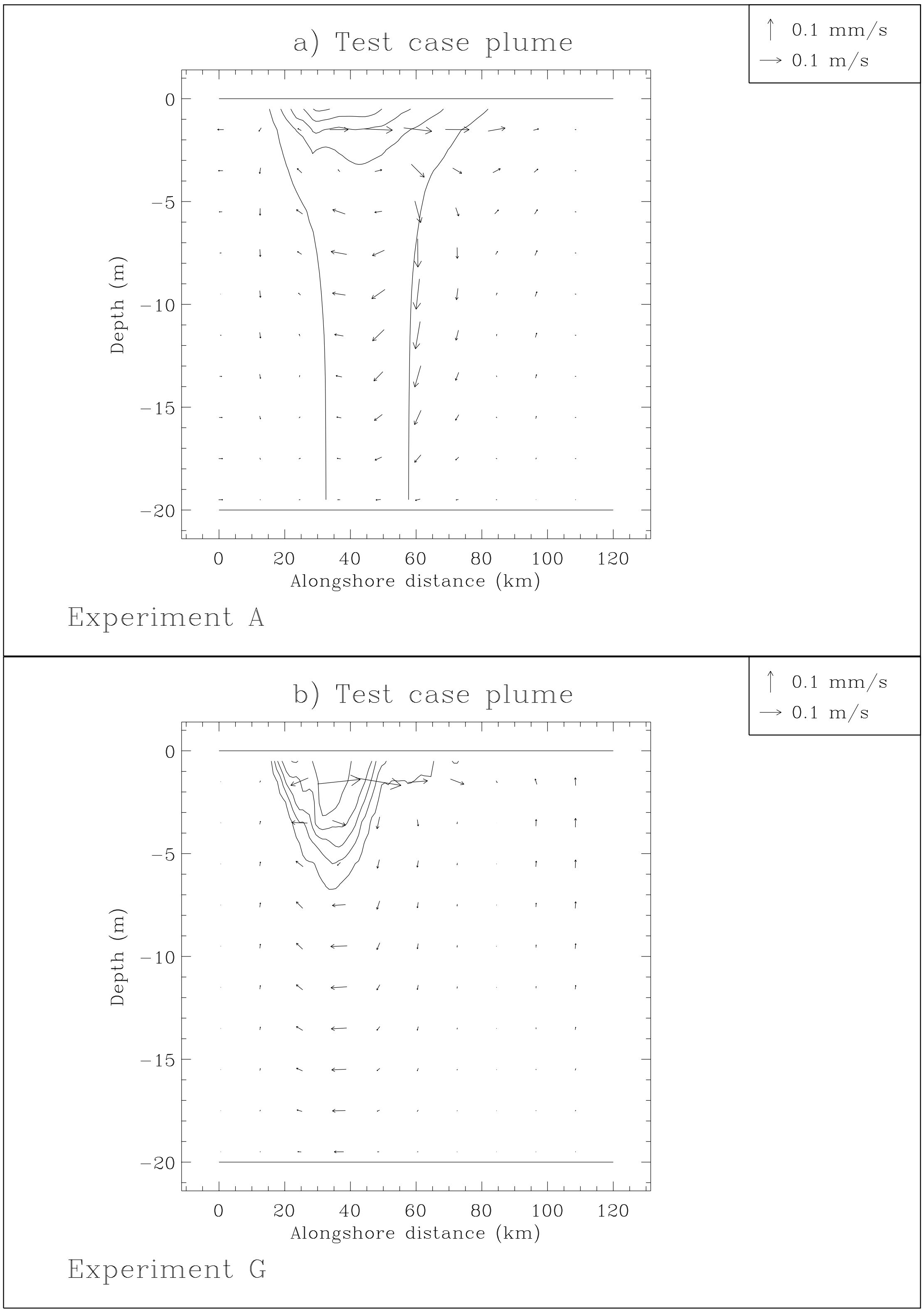
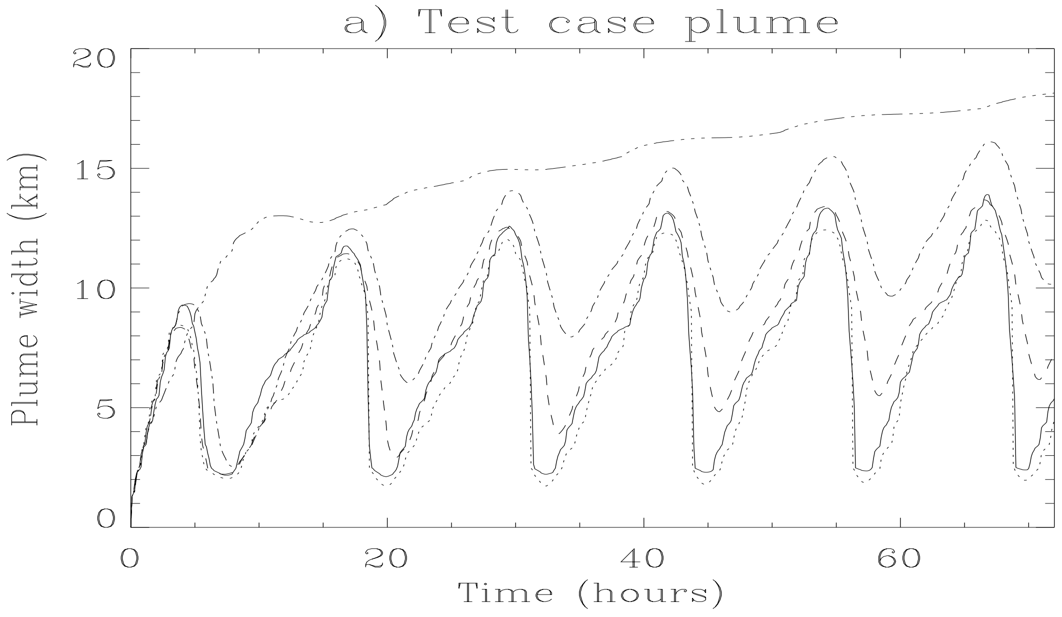


Figure 2.4.9: Residual distributions of currents and salinity for the last tidal cycle at the alongshore transect DD': experiment A (a), G (b). Contour intervals are at 2 PSU.



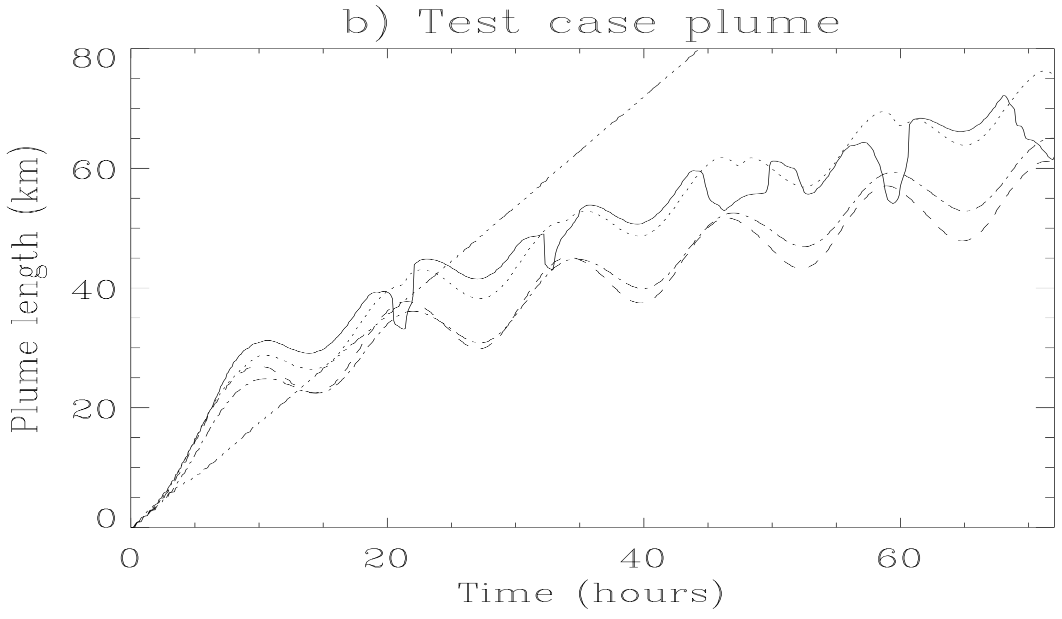
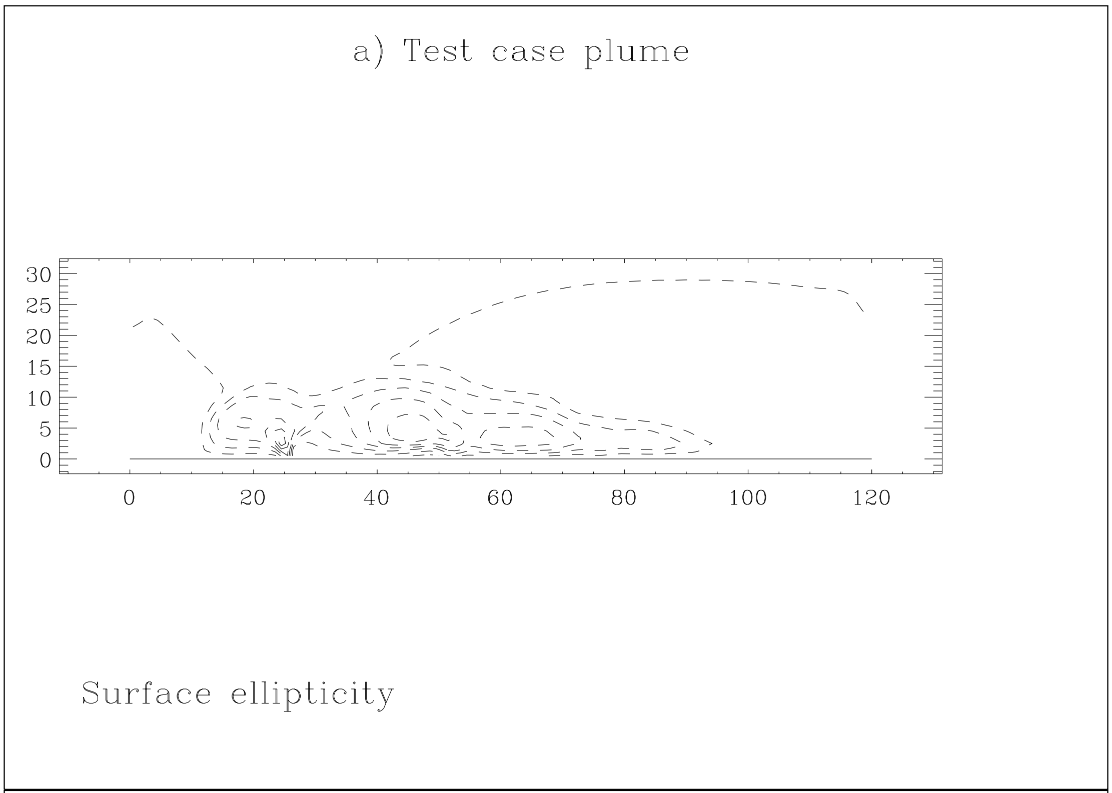


Figure 2.4.10: Time series of the plume width at the transect AA' (a) and the plume length at the transect CC' (b): experiment A (solid), B (dots), D (dashes), F (dash-dots), G (dash and 2 dots).



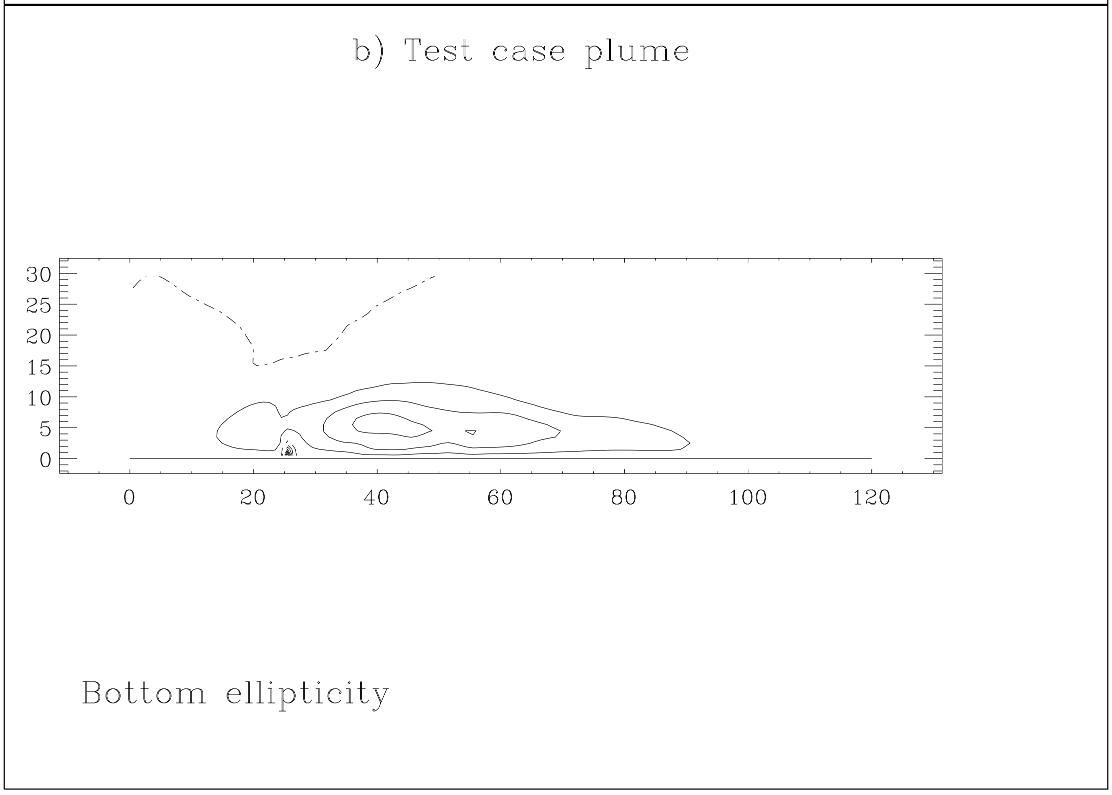


Figure 2.4.11: Ellipticity of the tidal ellipse for experiment A: surface (a) and bottom (b) distributions. Negative and positive values are drawn by dashed, respectively solid lines. Contour interval is 0.1. The analysis is performed for the last tidal cycle.

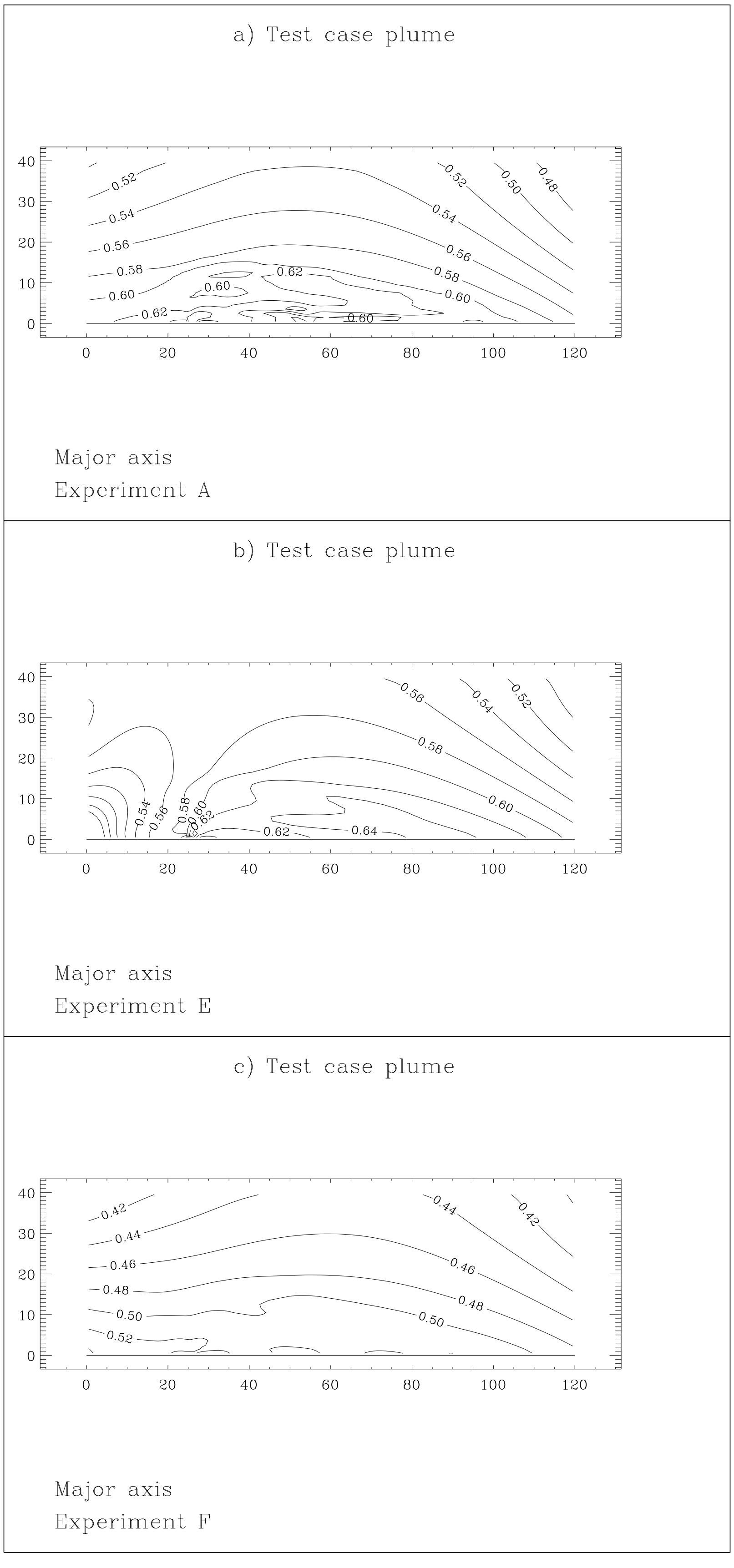


Figure 2.4.12: Distributions of the major axis (m/s) of the tidal ellipse for the depth-averaged current: experiment A (a), E (b), F (c).

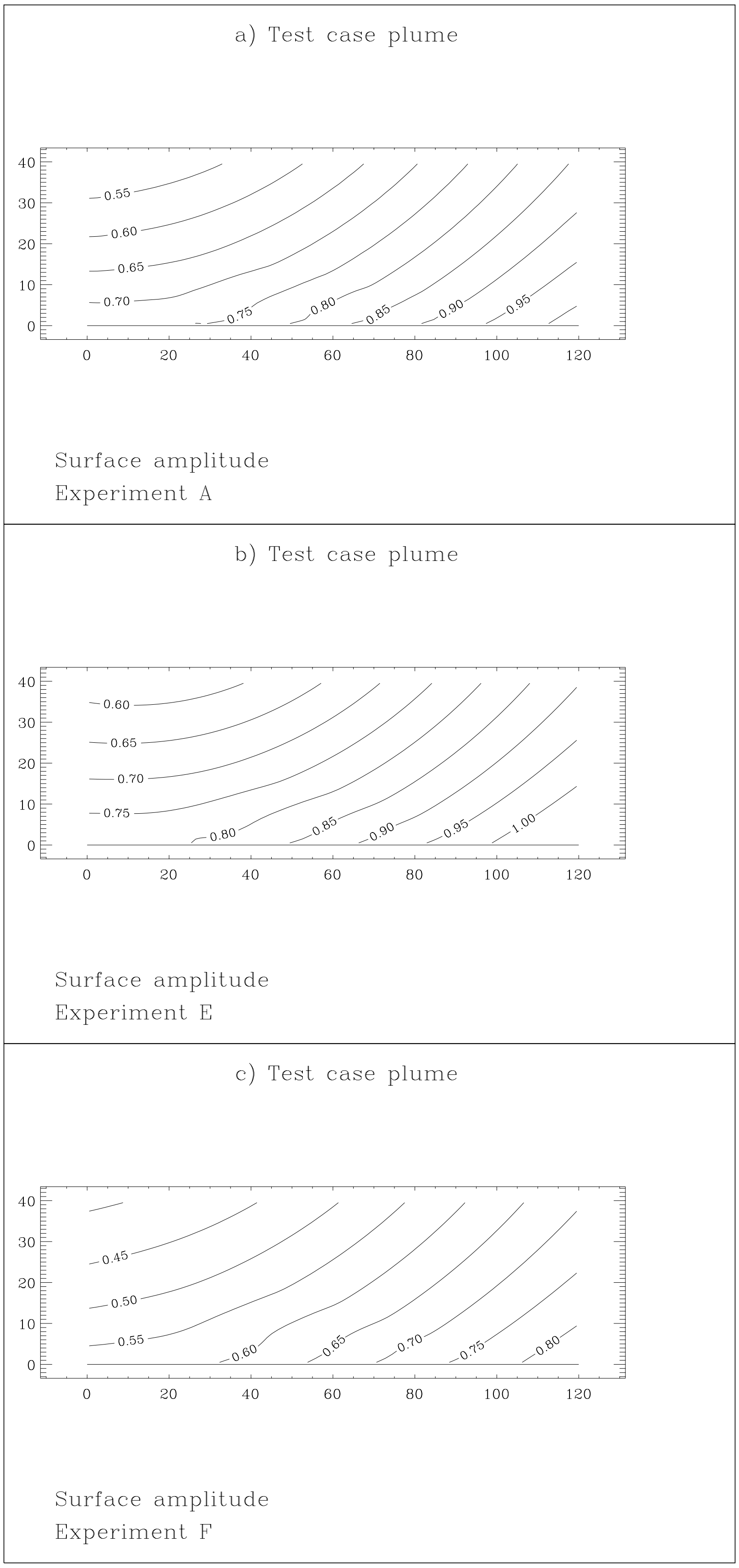


Figure 2.4.13: Distribution of the M_2-amplitude of the surface elevation (m) for experiment A (a), E (b), F (c).

Chapter 5

Biological Test Cases

The biological module is essentially composed of three parts.

1. The first part simulates microplankton growth or decay and cycles the concentrations of organic carbon and nitrogen, their detrital counterparts and dissolved concentrations of nitrate, ammonium and oxygen. Light input is provided from an optical module which calculates the 24 hour mean photosynthetically active radiation (PAR) as a function of day and depth taking account of light attenuation by organic and inorganic particulates and humic substances which are related to salinity.

2. The processes of sinking and resuspension are considered in the second part. Microplankton and detrital carbon and nitrogen have a vertical motion relative to water and sink to the bottom. The deposited material in the bottom ("fluff") layer can be resuspended within the water column through the action of the (tidally varying) bottom stress.

3. The physical processes of horizontal and vertical advection and diffusion are taken into account in the third part.

The sediment module solves an advection-diffusion equation for inorganic suspended particulate matter (iSPM). As in the biological model the suspended sediment interacts with a bottom "fluff" layer through sinking and resuspension. The "fluff" layer part of the biological module is linked to the sediment module since the amount of resuspended microplankton or detritus depends, in the model, on the fraction of deposited microplankton or detritus relative to the amount of deposited sediment. For more details about the biological and sediment model see Chapter III-2.

5.1 Test case batch

5.1.1 Description of the problem and model setup

This test case is designed to imitate a laboratory batch culture experiment where populations of microplankton are grown in a mesocosm under controlled conditions. This allows the biological processes encompassed in the model to be explored within a simple physical framework. It is recommended that users wishing to update biological parameters in the code, run test simulations with the batch culture test case to investigate the implications of their parameter choice.

The physical model uses the 1-D version of the program implemented for a water column of 5 meter depth (2 layers of 2.5 m). All meteorological and tidal forcing is switched off and the system is constantly mixed by an increased level of background diffusion. Particle sinking is set to zero and the fluff layer is turned off, so that all particulate material remains constantly in suspension. The concentration of iSPM is set to zero. All boundaries are reflective so that there is no flux of substance into or out of the model system. The batch culture is grown under constant light, which illuminates the top of the mesocosm and is attenuated with depth in the usual way. The reference salinity is set to 32.5 PSU representative of coastal sea water.

Initially the mesocosm is allowed to evolve without mesozooplankton grazing for 30 days. After this period the light is turned off and grazing pressure turned on. The first part of the simulation demonstrates microplankton dynamics; the second part shows detrital dynamics.

The complete model setup is listed in Table 2.5.1.

5.1.2 Experiments and output parameters

Three experiments are defined (see Table 2.5.2). The first experiment simulates a typical batch culture initialised with a low microplankton biomass and dissolved nitrate supply of 9.85 mmol N m^{-3}. During the initial light period of the experiment the microplankton takes up the nitrate and grows until the nutrient supply is exhausted, tending towards a chlorophyll biomass which is predictable (in the absence of mesozooplankton grazing) from:

$$X \overset{t\to\infty}{\longrightarrow} {}^{X}q^{N}(M_i + N_i + {}^{NH}S_i + {}^{NO}S_i) \tag{2.5.1}$$

where ${}^{X}q^{N}$ is the microplankton chlorophyll to nitrogen quota and M_i, N_i, ${}^{NH}S_i$, ${}^{NO}S_i$ are the initial detrital nitrogen, microplankton nitrogen, ammonium and nitrate concentrations. In experiment A the initial mesocosm total nitrogen concentration is 10.0 mmol N m^{-3} which gives a maximum theoretical chlorophyll yield of 14.0 mg Chl m^{-3}. In the second dark phase of the simulation mesozooplankton grazing depletes the microplankton biomass and generates detritus which is slowly remineralised. After sufficient time the mesocosm will return to something like its original condition, except that part of the nitrogen now remains in the mesozooplankton.

In the second experiment the same amount of nutrient is supplied, but in the form of ammonium. The dynamics of ammonium uptake, oxidation to nitrate and nitrate uptake are clearly demonstrated during the initial light phase. In this test case the theoretical maximum chlorophyll concentration is equivalent to that given in test case A.

The final experiment is initialised with a dissolved nitrate supply of 99.85 mmol N m^{-3}. This generates a considerably higher concentration of microplankton biomass with a theoretical maximum of 140.0 mg Chl m^{-3}. The concentrated biomass increases the attenuation of light within the mesocosm and reduces the PAR available for photosynthesis.

A: Uses the model setup listed in Table 2.5.1. Dissolved nitrate is supplied as 9.85 mmol N m^{-3} whereas the initial ammonium is set to 0.

B: As experiment A but now with dissolved ammonium supplied as 9.85 mmol N m^{-3} whereas the initial nitrate is set to 0.

C: As experiment A but now with dissolved nitrate supplied as 99.85 mmol N m^{-3}.

The results of the experiments are illustrated in Figures 2.5.1a–f for experiments A and B and in Figures 2.5.2a–f for experiment C, showing the time series of microplankton biomass, detritus, dissolved nutrients, mesozooplankton nitrogen and the attenuation coefficient k_d.

The following output parameters are evaluated for each experiment and written to the output *.tst*-files. Output values are listed in Table 2.5.3 and the maximum chlorophyll concentration obtained can be compared with the theoretical solution (2.5.1).

P2XMAX
: Maximum chlorophyll concentration (mg Chl m^{-3}).

IP2XMAX
: The number of the day when the maximum chlorophyll concentration occurred.

P2CMAX
: Maximum detrital carbon concentration (mmol C m^{-3}).

IP2CMAX
: The number of the day when the maximum detrital carbon concentration occurred.

P2ZNMAX
: Maximum mesozooplankton nitrogen concentration (mmol N m^{-3}).

IP2ZNMAX
: The number of the day when the maximum mesozooplankton nitrogen concentration occurred.

ATNMAX
: Maximum value of the attenuation coefficient (m^{-1}).

IATNMAX
: The number of the day when the attenuation coefficient attained its maximum value.

QTMIN
: Minimum value of the microplankton nitrogen to carbon quota (mmol N (mmol C)$^{-1}$).

IQTMIN

The number of the day when the microplankton nitrogen to carbon quota has its lowest value.

QTMAX

Maximum value of the microplankton nitrogen to carbon quota (mmol N (mmol $C)^{-1}$).

IQTMAX

The number of the day when the microplankton quota has its largest value.

IFP2TNMIN

The first day of dissolved nutrient exhaustion.

ILP2TNMIN

The last day of dissolved nutrient exhaustion.

TNMIN

Minimum value of the total mesocosm nitrogen during the simulation (mmol N m^{-2}). Within the limits of numerical accuracy, TNMAX and TNMIN should be equivalent to confirm that the model conserves nitrogen.

TNMAX

Maximum value of the total mesocosm nitrogen during the simulation (mmol N m^{-2}). This is calculated each day by summing the microplankton, mesozooplankton, detritus and dissolved nitrogen pools and integrating over depth.

5.1.3 Results

- In all experiments microplankton biomass accumulated rapidly during the light period of the simulation until the dissolved nutrients were depleted. Additional growth occurred briefly at the end of the light period (days 31 and 32) when the microplankton population where able to utilise the first recycled nutrients generated by grazing and detrital remineralisation. During this period light declined exponentially (by the formulation used to calculate 24 hour mean PAR) but as grazing losses exceeded new growth, the population declined. In all cases maximum chlorophyll biomass values approached the theoretical solution. Simulated values were slightly lower than the theoretical maximum concentration due to a tiny fraction of the dissolved nutrients remaining in solution. (In the model nutrient uptake ceases when concentrations fall below a threshold value of 0.001 mmol N m^{-3} for nitrate and 0.001 mmol N m^{-3} for ammonium. This ensures that concentrations always remain positive). In test case C extremely high biomass concentrations resulted in increased attenuation and reduced availability of PAR in the mesocosm.

Table 2.5.1: Model setup for test case **batch**.

Parameter	Value	Purpose
param.inc		
NC[a]	2	number of grid points in the X-direction
NR[a]	2	number of grid points in the Y-direction
NZ	2	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NOBV	4	number of open boundary points in the Y-direction
DEFCON		
IGRDIM	1	1-D application of the program
IOPTB	1	biology module switched on
IADVC	0	advection of momentum disabled
IADVS	0	advection of scalars disabled
IOPTK	0	vertical diffusion set to a uniform background value
IOPT2	0	mode splitting disabled
IOPT3	0	3-D current calculations disabled
IOPTD	0	uniform density
IBSTR	0	bottom stress set to zero
IOPTM	4	time-dependent meteorological input. Data for solar radiation are supplied in subroutine WRFCDAT
IFLUFF	0	fluff layer disabled
IADVWB	0	vertical sinking disabled in the biological module
IBDATE	04010000	start date (MMDDHHMM)
IEDATE	12070000	end date (MMDDHHMM)
DELT	600.0	time step (s)
ICMET	4320	time interval (30 days) for meteorological input divided by DELT
HEADERS	.TRUE.	header information enabled
DEPUN	5.0	uniform water depth (m) (used in 1-D applications only)
SREF	32.5	reference salinity used to initialise the salinity field, also used to evaluate the diffusive attenuation coefficient for PAR (PSU)
TREF	10.0	reference temperature used to initialise the temperature field (^{0}C)
DIFMOL	0.1	background/uniform eddy diffusivity (m2/s)
FRATE	0.0	benthic relaxation timescale (day^{-1}) (disables the benthic nutrient flux)

[a]Although the program is used as a 1-D model in this application, a minimum value of 2 is required by the program structure.

Table 2.5.1: continued.

Parameter	Value	Purpose
DEFICS		
P2B(NZ,NR,NC)	1.0	initial concentration of microplankton carbon B (mmol C m^{-3})
P2C(NZ,NR,NC)	0.0	initial concentration of detrital carbon C (mmol C m^{-3})
P2M(NZ,NR,NC)	0.0	initial concentration of detrital nitrogen M (mmol N m^{-3})
P2N(NZ,NR,NC)	0.15	initial concentration of microplankton nitrogen N (mmol N m^{-3})
P2NHS(NZ,NR,NC)	0.0	initial concentration of ammonium ^{NH}S for experiments A and C (mmol N m^{-3})
	9.85	initial concentration of ammonium ^{NH}S for experiment B (mmol N m^{-3})
P2NOS(NZ,NR,NC)	9.85	initial concentration of nitrate ^{NO}S for experiment A (mmol N m^{-3})
	0.0	initial concentration of nitrate ^{NO}S for experiment B (mmol N m^{-3})
	99.85	initial concentration of nitrate ^{NO}S for experiment C (mmol N m^{-3})
P2O(NZ,NR,NC)	300.0	initial concentration of oxygen O (mmol O m^{-3})
GRAZING(NR,NC,12)	0.2	grazing pressure (except for the first month when grazing is set to zero) (day^{-1})
WRFCDAT		
QSOLUN	0.0	solar radiation (except for the first month when QSOLUN = 300.0) (W/m^2)

Table 2.5.2: Definition of the **batch** experiments.

Test	Purpose
A	Uses the model setup listed in Table 2.5.1. Initial concentrations of nitrate and ammonium are set to 9.85 and 0.0 mmol N m^{-3}.
B	As experiment A but now with the initial concentration of nitrate set to 0.0 and the initial concentration of ammonium set to 9.85 (mmol N m^{-3})
C	As experiment A but now with the initial concentration of nitrate set to 99.85 (mmol N m^{-3})

Table 2.5.3: Output values for the parameters of the test case **batch**.

Parameter	A	B	C
P2XMAX	13.9963	13.9954	139.9588
IP2XMAX	12	10	18
P2CMAX	21.7200	21.7208	199.0811
IP2CMAX	51	51	50
P2ZNMAX	5.4405	5.4407	53.5212
IP2ZNMAX	56	56	65
ATNMAX	0.4269	0.4269	2.4423
IATNMAX	12	10	18
QTMIN	0.0890	0.0890	0.0924
IQTMIN	31	31	31
QTMAX	0.1963	0.1963	0.1963
IQTMAX	81	207	52
IFP2TNMIN	12	10	18
ILP2TNMIN	30	30	30
TNMIN	49.8926	49.8939	498.8227
TNMAX	49.9991	49.9991	499.9911

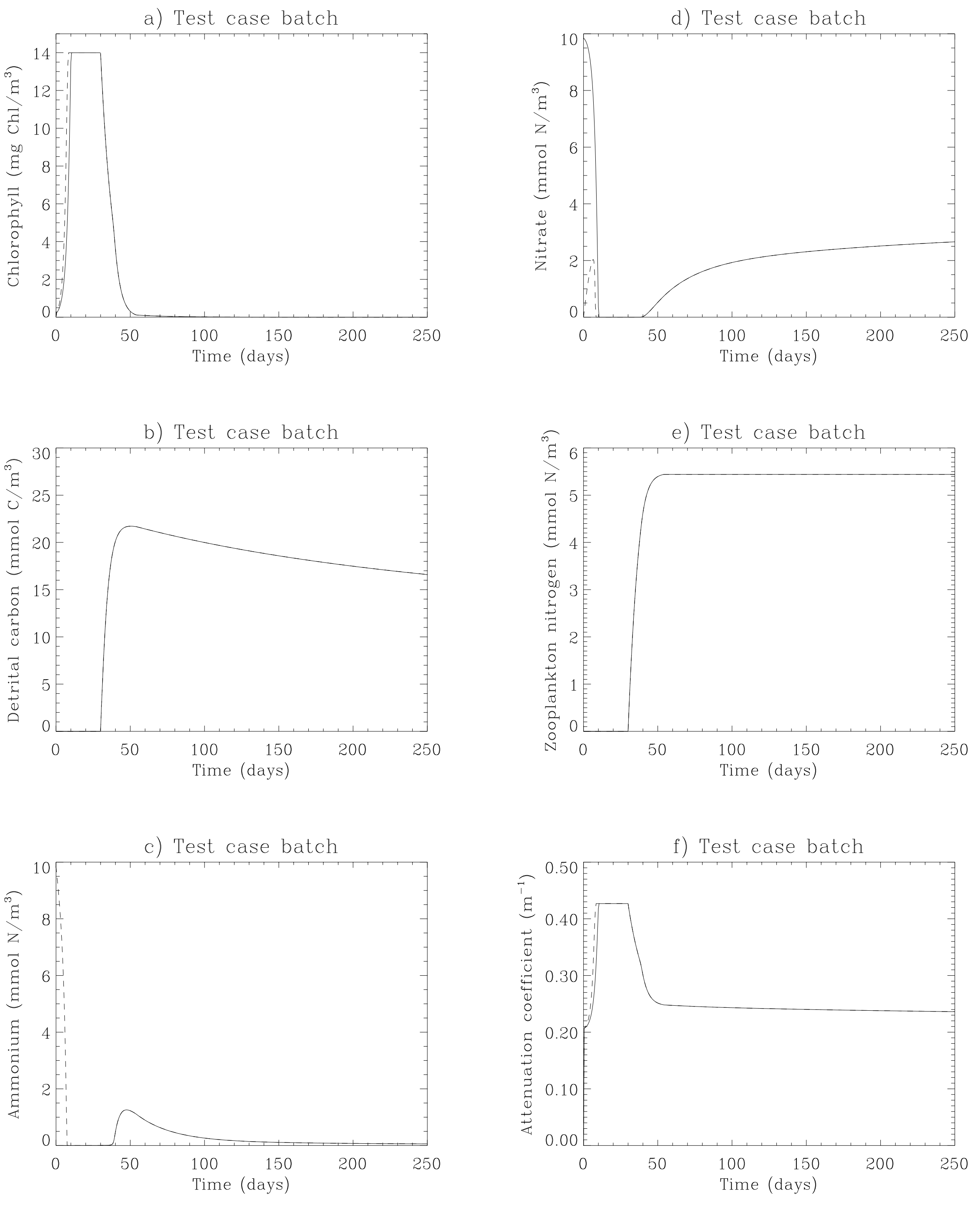


Figure 2.5.1: Time series of (a) microplankton chlorophyll (mg Chl m^{-3}), (b) detrital carbon (mmol C m^{-3}), (c) ammonium (mmol N m^{-3}), (d) nitrate (mmol N m^{-3}), (e) mesozooplankton nitrogen (mmol N m^{-3}), and (f) the attenuation coefficient k_d (m^{-1}): experiment A (solid curve), B (dots).

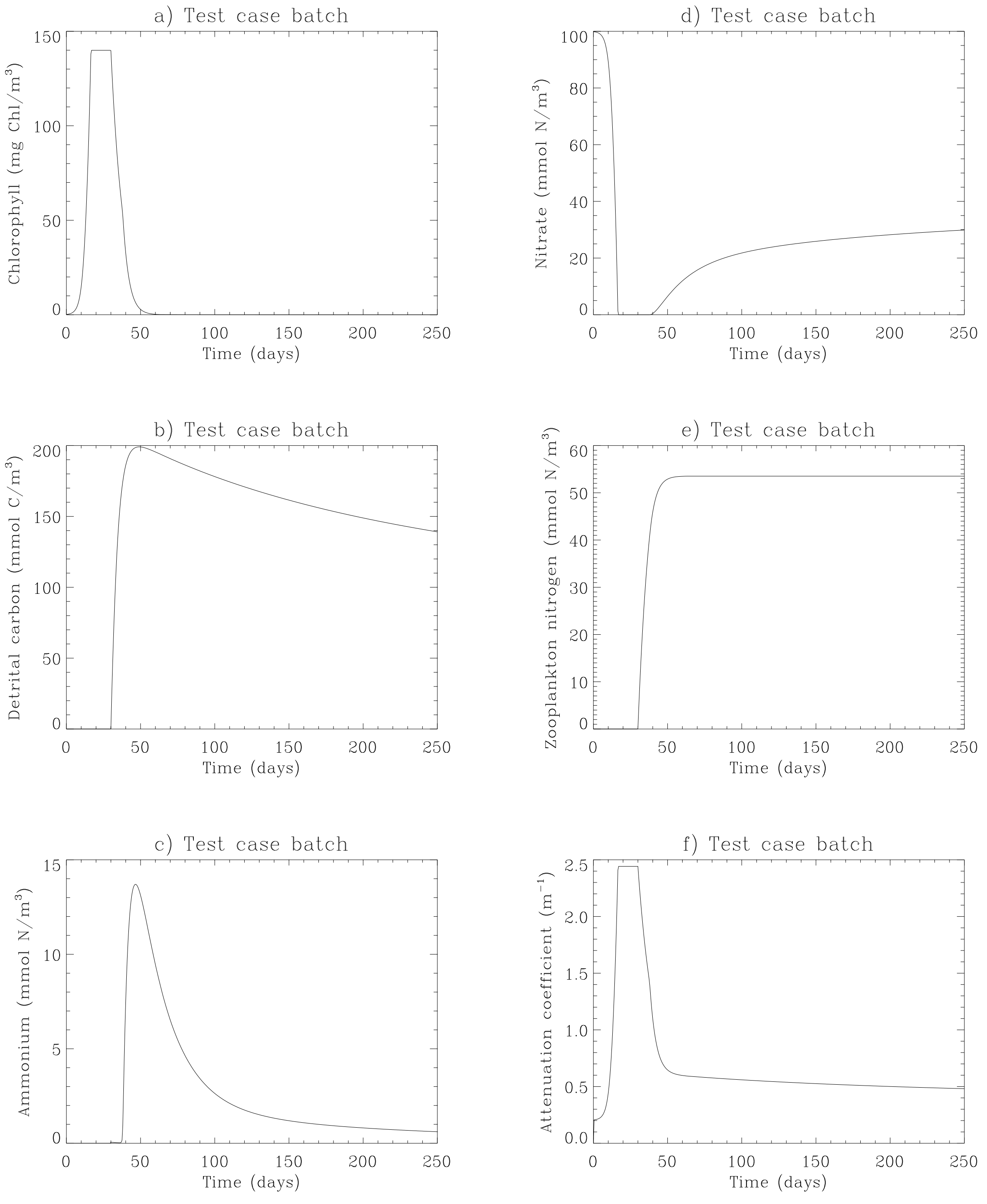


Figure 2.5.2: Time series for experiment C of (a) microplankton chlorophyll (mg Chl m^{-3}), (b) detrital carbon (mmol C m^{-3}), (c) ammonium (mmol N m^{-3}), (d) nitrate (mmol N m^{-3}), (e) mesozooplankton nitrogen (mmol N m^{-3}), and (f) the attenuation coefficient k_d (m^{-1}).

- Nutrients were exhausted in all cases during the light period of the simulation. Depletion occurred most rapidly in case B (day 10) as microplankton uptake of ammonium proceeds at a faster rate than nitrate uptake (to imitate the reduced metabolic cost of ammonium assimilation). In case C the nitrate concentration was not depleted until day 18 due to its high initial value.
- During the dark period of the experiment microplankton growth ceased and the biomass was reduced by basal respiration and mesozooplankton grazing. A portion of the grazed microplankton nitrogen is excreted directly back into the water as ammonium, a further portion of the carbon and nitrogen is defaecated as detritus and a fraction of the nitrogen is retained in the mesozooplankton. From the onset of the grazing period detritus accumulates, however ammonium (and any nitrate generated by oxidation) concentrations remain depleted due to simultaneous microplankton uptake.
- As detritus begins to accumulate it slowly mineralises, releasing further ammonium, which in turn is slowly oxidised to nitrate. The remaining microplankton is now repleted with nutrients and so dissolved concentrations begin to increase. Detrital concentrations increase until the microplankton biomass falls below the grazing threshold, set at 0.42 mmol C m^{-3} and mesozooplankton grazing ceases. Initially the nitrogen–rich labile detritus is remineralised rapidly. This rate then declines over time as the detritus becomes more refractory in nature with a lower nitrogen content.
- From these batch culture experiments it can be seen that the timescales required for microplankton uptake and growth are considerably shorter (of the order of 20 days) than the timescales required for detrital mineralisation of the equivalent biomass (of the order of 3 months).
- In all cases the total nitrogen within the mesocosm was conserved, within the limits of numerical accuracy, verifying that the model and numerical scheme were operating correctly. (When the test case was implimented in double precision total mesocosm nitrogen remained exactly constant throughout the simulation).

5.2 Test case csbio

5.2.1 Description of the problem and model setup

The time scales for the biological model are usually of the order of the seasonal scale. Test case **csbio** has been designed to illustrate the cycling processes for the biological state variables. As in test case **csnsp** the simulation is performed at station CS in the North Sea for a period of two years. To minimise the role of short-time scale processes which are induced by the physics, such as surface fluxes derived from varying meteorological data and tides modulated during a spring-neaps period, mean climate forcing conditions are considered. Advection by the physical current and horizontal diffusion are neglected.

The model equations for the physics are the same as for test case **csnsp** and are given by equations (2.3.21)–(2.3.26) of Section II-3.2.1.

Contrary to the standard formulations used in the program the non-solar part of the surface flux is determined using the empirical formulae of Edinger et al. (1974):

$$\begin{aligned}
Q_{nsol} &= c_T(T_s - T_d) \\
c_T &= 4.5 + 0.05T_s + (\beta + 0.47)f_w \\
\beta &= 0.35 + 0.015T_m + 0.0012T_m^2 \\
T_m &= \frac{1}{2}(T_s + T_d) \\
f_w &= 9.2 + 0.46W^2
\end{aligned} \tag{2.5.2}$$

where T_d and T_s are the dew point and sea surface temperatures in ^{0}C and W the magnitude of the wind speed in m/s. The meteorological data are supplied in the form of an annually varying sine wave

$$F = a_1 + a_2 \sin(\omega_a t_d + a_3) \tag{2.5.3}$$

where F stands either for the wind magnitude, surface solar irradiance or dew point temperature, $\omega_a = 2\pi/365$ rad/day is the annual frequency and t_d the time in days since the beginning of the year. Values of the parameters a_1, a_2, a_3 are given in Table 2.5.5. The wind direction is determined at hourly intervals using a random number generating function with the restriction that changes in direction are limited to 10 degrees per hour. A daily cycle is imposed for solar irradiance $\overline{Q}$ assuming a day length of 12 hours:

$$Q_{sol} = 0.95\pi\overline{Q}\max\Big(0, \sin(2\pi(t_h - 5.5)/24)\Big) \tag{2.5.4}$$

where t_h is the time in hours. The factor 0.95 takes account of a 5% surface albedo and the daily averaged solar irradiance $\overline{Q}$ varies annually using the form (2.5.3).

The surface slopes are prescribed in the following simple form

$$\begin{aligned}
g\frac{\partial\zeta}{\partial x_1} &= (\omega_2 + \varepsilon f)A\cos\omega_2 t \\
g\frac{\partial\zeta}{\partial x_2} &= (f + \varepsilon\omega_2)A\cos(\omega_2 t - \pi/2)
\end{aligned} \tag{2.5.5}$$

where ω_2 is the frequency of the M_2-tide, f the Coriolis frequency and A, ε the semi-major axis and ellipticity of the tidal ellipse in the absence of vertical diffusion. As typical values for station CS, A and ε are set to respectively 0.28 and 0.18.

For the biology and sediments a series of transport equations are solved of the form (see also equations (3.1.82), (3.2.3), (3.2.4)):

$$\frac{\partial\psi}{\partial t} + w_s^\psi\frac{\partial\psi}{\partial x_3} = \frac{\partial}{\partial x_3}\Big(\lambda_T\frac{\partial\psi}{\partial x_3}\Big) + \beta(\psi) \tag{2.5.6}$$

where ψ either represents microplankton carbon or nitrogen, detrital carbon or nitrogen, ammonium, nitrate, oxygen, zooplankton nitrogen or iSPM. For more details about the source/sink terms $\beta(\psi)$, the sinking rates w_s^ψ, the surface and bottom boundary conditions and the exchanges which take place between the water column, the "fluff" layer and the sediment layer see Chapter III-2.

The complete model setup is listed in Table 2.5.4.

5.2.2 Experiments and output parameters

Six experiments have been defined which examine the impact of different physical/biological processes and numerical schemes on the biological cycle and the resuspension of deposited organic and inorganic material. They are also described in Table 2.5.6.

A : Uses the model setup listed in Table 2.5.4, the default turbulence scheme and default values for the biological switches IFLUFF and IADVWB.

B : As experiment A now without sinking ($w_s^\psi = 0$) and resuspension in the biological module. This means that the exchange with the fluff layer is disabled which is performed in the program by setting the switch IFLUFF equal to zero. Sinking and resuspension are still enabled in the sediment module.

C : As experiment A but with the fluff layer and the sediment module switched off. Sinking of organic material (microplankton and detritus) remains enabled.

D : As experiment A now using the upwind scheme for vertical advection instead of the default TVD scheme. This is performed by setting the switch IADVWB equal to 1.

E : As experiment A now using the Paconowski-Philander scheme for turbulence (see Section III-1.2.1a).

F : As experiment A now with the wave-current interaction module enabled (see Section III-1.6.6). Wave height and period are set to the uniform values $h_s = 2$ m and $T_w = 10$ s. The assumption of uniform values is obviously unrealistic for a two years period. The intention here is only to verify whether wave-current interaction can have an appreciable influence on the resuspension rates and water column concentrations of organic and inorganic material.

A series of figures have been produced for each experiment. Results which are similar to the ones obtained for experiment A, have been omitted in the plots. Time series of temperature for experiments A, E and F are displayed in the form of depth-time contour plots in Figures 2.5.3a–c. Figures 2.5.4 shows the time series of the thermocline depth, determined as the distance to the surface where the temperature is 1^0C higher than the bottom temperature, for the same experiments. Depth-time contour plots of chlorophyll are given in Figures 2.5.5a–b for experiments A and B. Time series of depth-integrated chlorophyll are displayed in Figure 2.5.6 for all experiments. Figures 2.5.7a–c and 2.5.8a–c are depth-time contour plots for respectively detrital carbon and ammonium and for experiments A, B, E. Figures 2.5.9a–b are similar to the previous contour plots now for inorganic sediment and experiments A and E. Time series of the tidally averaged values of the depth mean iSPM concentrations are shown in Figure 2.5.10 for experiments A, D, E, F.

The following output parameters are evaluated for each month of the simulation. Output values are listed in Table 2.5.7.

TMEAN
: Monthly averaged value of the depth-averaged temperature in ^{0}C.

P2XINT
: Monthly averaged value of the depth-integrated chlorophyll concentration in mg Chl m^{-2}.

P2CINT
: Monthly averaged value of the depth-integrated concentration of detrital carbon within the water column in mmol C m^{-2}.

P2NHSINT
: Monthly averaged value of the depth-integrated concentration of ammonium in mmol N m^{-2}.

SEDMEAN
: Monthly averaged value of the depth-averaged concentration of iSPM within the water column in g m^{-3}.

SEDDEV
: The total amount of iSPM, either in resuspended form within the water column or in deposited form within the fluff layer, must be conserved. This is measured by the parameter

$$\mathsf{SEDDEV} = 10^4(\langle A_{tot}(t)\rangle - A_{tot}(0))/A_{tot}(0) \tag{2.5.7}$$

where the total amount of iSPM $A_{tot}(t)$ is given by the sum of the depth integral of the suspended concentration and the amount of deposited material in the fluff layer, $A_{tot}(0)$ represents its initial value and $\langle\ \rangle$ a monthly averaged value. The schemes for the vertical advection and diffusion of inorganic (and organic) material and the exchanges of iSPM between the water column and fluff layer are implemented as conservative processes so that SDEV is zero in the absence of numerical rounding errors.

BALNMONT
: An important test for the biology is the balance of nitrogen in the model equations. Adding the equations for microplankton nitrogen, detrital nitrogen, ammonium, nitrate and zooplankton nitrogen (see equations (3.2.18), (3.2.61), (3.2.67), (3.2.66) and (3.2.57)), integrating the result over the whole water column and adding the

amounts in the fluff layer one obtains

$$\frac{\partial \overline{N}_{tot}}{\partial t} = -\lambda_T \frac{\partial}{\partial x_3}({}^{NO}S + {}^{NH}S)|_{bot} - F_l = Q_{tot} \tag{2.5.8}$$

where an overbar denotes a depth-integrated quantity, $\overline{N}_{tot} = \overline{N} + \overline{M} + \overline{{}^{NO}S} + \overline{{}^{NH}S} + \overline{Z_N}$ the total nitrogen content and Q_{tot} all remaining source/sink terms. The first terms on the right of (2.5.8) are the benthic nutrient fluxes. The last term (F_l) represents the losses in the fluff layer consisting of a 10% daily loss of microplankton and detrital nitrogen to the sediment. Equation (2.5.8) is further integrated over a daily period (T) giving

$$\overline{N}_{tot}(t+T) - \overline{N}_{tot}(t) - \int_t^{t+T} Q_{tot} dt = \frac{R_m}{2}(\overline{N}_{tot}(t) + \overline{N}_{tot}(t+T)) \equiv 0 \tag{2.5.9}$$

The parameter R_m denotes the relative gain or loss of nitrogen during the day m and is zero in the exact case. Non-zero values are obtained from the numerical solution due to inaccuracies of the numerical schemes and rounding errors. The parameter BALNMONT, expressed in %, is then defined as the sum of the daily values of R_m over an entire month:

$$\mathsf{BALNMONT} = 100 \sum_{m=1}^{M} R_m \tag{2.5.10}$$

where M is the number of days of the month.

BALNMAX

This parameter is also related to the nitrogen balance and is defined as the maximum value of $|R_m|$ during one month, expressed in %:

$$\mathsf{BALNMAX} = 100 \max_m(|R_m|) \tag{2.5.11}$$

The previous parameters are written to the output *.tst*-files for each month. Values are listed in Table 2.5.7 for all even months.

The following parameters are evaluated yearly:

P2XMAX

Maximum chlorophyll concentration in mg Chl m^{-3} over all grid points and time steps during one year.

IDAYMAX

The day when chlorophyll has its largest value during one year.

DEPMAX

The depth of the grid point in m where chlorophyll attains its largest value during one year.

BALNYEAR

The same as BALNMONT now defined as the annual sum of R_m in %:

$$\mathsf{BALNYEAR} = 100 \sum_{m=1}^{365} R_m \tag{2.5.12}$$

5.2.3 Results

- The seasonal cycle of temperature is less realistic as the one obtained with the test case **csnsp** since the surface fluxes of momentum and heat are derived from mean climatological instead of real forcing data. Although the initial temperature is not exactly the same as the one prevailing at the start of the second year, the summer distributions are highly similar for the two yearly cycles. The thermocline depth is ~30 m increasing to 35 m in the case of experiment E which is qualitatively in agreement with the results obtained for test case **csnsp**. The results for experiments A and F are practically the same indicating that the temperature cycle is insensitive to the enhancement of the bottom stress produced by the wave-current interaction.

- The initial chlorophyll concentration decays during the first two months of the first year and starts to grow in March. A spring bloom is observed when the thermocline starts to form with a maximum of 14.6 mg Chl m^{-3} within the thermocline at mid-April in the case of experiment A (see Table 2.5.7). Although the results are qualitatively similar for all other experiments, some quantitative differences are noted. Vertical sinking is disabled in experiment B yielding a uniform chlorophyll concentration in the mixed upper 30 m of the water column and a maximum which is ~45% lower compared to experiment A. In the case of experiment D which uses the more diffusive upwind scheme for vertical advection, and experiment F where turbulence is enhanced by the effects of wave-current interaction, the chlorophyll maxima are lower by respectively 20% and 35% due to the larger vertical mixing. Chlorophyll decays quickly after the bloom due to nutrient depletion. In the absence of vertical sinking (experiment B) the microplankton is bounded to the thermocline and decays less rapidly whereas a more rapid decay is observed within the well mixed bottom layer in all other experiments. The results are similar during the second year. Main differences are a higher chlorophyll maximum (except for experiment B) occurring 5 to 20 days later compared to the first year.

  The time series of the total amount of chlorophyll (water column and fluff layer), displayed in Figure 2.5.6, shows no appreciable differences between the different experiments during the spring blooms. Less microplankton is produced during the second year which is explained by the lower amount of available nutrients (see Figure 2.5.8). In the absence of sinking (experiment B) chlorophyll decays less rapidly and levels off when microplankton carbon decays below the threshold of 0.42 mmol C m^{-3} for zooplankton grazing.

- Microplankton carbon is converted into detrital carbon by zooplankton grazing and defaecation at a rate given by $(1-\gamma)G \simeq 0.004 - 0.012$ day^{-1}. Detritus decays by remineralisation and sinks to the bottom at a rate of 1–5 m/day. Part of the deposited material is resuspended in the water column by the action of the bottom stress whereas a daily fraction of 10% is drawn into the consolidated sediment. As opposed to sinking remineralisation is a much slower process with a time scale of the order of a few months. This explains the higher concentrations of detrital carbon for experiment B (Figure 2.5.7a–c and Table 2.5.7).

- The distributions of iSPM, plotted in Figures 2.5.9a–b, clearly show that resuspension rates have a seasonal pattern. The presence of the thermocline in the summer reduces the turbulence produced by the tides in the lower part of the water column and thus also the magnitude of the bottom stress, so that a lesser amount of inorganic and organic material is resuspended during this season. Resuspension rates quickly increase as soon as the thermocline disappears. Since there is no direct feedback from the biological to the sediment module, the results for experiments A and B are the same. The results of the other experiments are compared in Figures 2.5.10 showing the tidally averaged values of the depth mean iSPM concentrations (see also the values of the parameter SEDMEAN in Table 2.5.7). Compared to experiment A the mean iSPM concentration is ~6% larger in experiment D (upwind scheme for vertical advection instead of the default TVD scheme). A larger increase of ~17–25% is obtained when wave-current interaction is included in the calculations (experiment F). On the other hand, when the default turbulence scheme is replaced by the simpler Paconowski-Philander formulation, the mean suspended concentrations are mostly 50% lower compared to experiment A. This indicates that both the turbulence scheme as wave-current interaction have an important influence on the resuspension process. It is expected that the latter effect becomes even stronger in shallower waters (see Section III-1.6.6).

- Conservation of the nitrogen balance is an important test for the biological module. The monthly sum of the daily losses or gains of nitrogen, represented by the parameter BALNMONT in Table 2.5.7, has a magnitude which is mostly lower than 0.03% in experiments A, D, E, F and is even lower than 0.002% in experiments B and C (due to the absence of a fluff layer component) indicating that nitrogen is well conserved by the model. Important to note is that the annual sum, given by the parameter BALNYEAR, has a magnitude of the same order as the monthly value showing that there is no net tendency for an increase or decrease of nitrogen.

- Finally, the accumulated loss or gain of the total amount of inorganic particulate material, as represented by the parameter SEDDEV, is not larger that ~0.1%. (The percentage is obtained by dividing the values in Table 2.5.7 by 100).

Table 2.5.4: Model setup for test case **csbio**.

Parameter	Value	Purpose
param.inc		
NC[a]	2	number of grid points in the X-direction
NR[a]	2	number of grid points in the Y-direction
NZ	15	number of grid points in the vertical
NOBU	4	number of open boundary points in the X-direction
NOBV	4	number of open boundary points in the Y-direction
NCON	1	number of tidal frequencies
DEFCON		
IGRDIM	1	1-D application of the program
IOPTB	1	biology module switched on
IOPTS	1	sediment module switched on (except in experiment C where IOPTS = 0)
IADVC	0	advection of momentum disabled
IADVS	0	advection of scalars disabled (except vertical advection by sinking velocities enabled with IADVWB)
IOPT2	0	mode splitting disabled
IOPTHE	2	module for update of temperature switched on, optical module switched on
IOPTD	2	thermal expansion coefficients calculated using the general equation of state of seawater (see Section III-1.5)
IOPTM	4	time-dependent meteorological input. Surface fluxes are calculated as a function of climatological data using a modified version of subroutine METIN (file *metin.f* in the **PROJECT** directory).
IOUTT	1	harmonic analysis enabled
IBDATE	01010000	start date (MMDDHHMM)
IEDATE	01010000	end date (MMDDHHMM)
IBYEAR	1989	start year
IEYEAR	1991	end year
DELT	1200	time step (s)
ICMET	3	time interval (s) for meteorological input divided by DELT
ICRFORM	‘U’	files with initial/final conditions written in binary format
DLAREF	55.515	reference latitude (degrees)
DLOREF	0.90833	reference longitude (degrees)
DEPUN	83.0	uniform water depth (m) (used in 1-D applications only)

[a] Although the program is used as a 1-D model in this application, a minimum value of 2 is required by the program structure.

Table 2.5.4: continued.

Parameter	Value	Purpose
SREF	34.8	reference salinity used to initialise the salinity field, also used to evaluate the diffusive attenuation coefficient for PAR (PSU)
TREF	7.8	reference temperature used to initialise the temperature field (^{0}C)
CDZ0UN	0.0024	uniform roughness length used for the evaluation of the bottom friction coefficient (m)
HSUN	2.0	uniform wave height (m) (experiment F only)
TWUN	10.0	uniform wave period (s) (experiment F only)
DEFOB2D		
SIGMA(1)	1.4056E-04	frequency of the M_2-tide (rad/s)
ANALSIG(1)	1.4056E-04	frequency for harmonic analysis (rad/s)
AMPOBU(1-4,1)	4.5417E-05	the amplitude $(\omega_2+\varepsilon f)A$ of the surface slope (2.5.5) in the X-direction (m/s^2)
PHAOBU(1-4,1)	0	the phase of the surface slope (2.5.5) in the X-direction (rad)
AMPOBV(1-4,1)	4.0744E-05	the amplitude $(f+\varepsilon\omega_2)A$ of the surface slope (2.5.5) in the Y-direction (m/s^2)
PHAOBV(1-4,1)	1.5708	the phase of the surface slope (2.5.5) in the Y-direction (rad)
DEFICS		
P2B(NZ,NR,NC)	3.4	initial concentration of microplankton carbon B (mmol C m^{-3})
P2C(NZ,NR,NC)	1.0	initial concentration of detrital carbon C (mmol C m^{-3})
P2M(NZ,NR,NC)	0.1	initial concentration of detrital nitrogen M (mmol N m^{-3})
P2N(NZ,NR,NC)	0.2	initial concentration of microplankton nitrogen N (mmol N m^{-3})
P2NHS(NZ,NR,NC)	1.0	initial concentration of ammonium ^{NH}S (mmol N m^{-3})
P2NOS(NZ,NR,NC)	5.0	initial concentration of nitrate ^{NO}S (mmol N m^{-3})
P2O(NZ,NR,NC)	300.0	initial concentration of oxygen O (mmol O m^{-3})
GRAZING(NR,NC,12)		monthly averaged grazing pressures (day^{-1}) as listed in Table 3.2.7
SEDC1(NZ,NR,NC)	1.0	initial concentration of inorganic sediment A (g m^{-3})

Table 2.5.5: Parameters for the climatological surface forcing (2.5.3).

	a_1	a_2	a_3
W	7.1 m/s	2.2 m/s	76.80 degrees
Q_{sol}	130 W/m^2	109 W/m^2	281.70 degrees
T_d	8.85 ^{0}C	5.07 ^{0}C	233.51 degrees

Table 2.5.6: Definition of the **csbio** experiments.

Test	Purpose
A	Uses the model setup listed in Table 2.5.4, default turbulence scheme and default values for the biology switches IFLUFF and IADVWB.
B	As experiment A now with IFLUFF = 0 (fluff exchange and sinking disabled in the biological module).
C	As experiment A now with IOPTS = 0 (interaction with the fluff layer through deposition/resuspension disabled and the sediment module switched off). Vertical sinking remains enabled.
D	As experiment A now with IADVWB = 1 (upwind scheme for vertical advection).
E	As experiment A now using the Paconowski-Philander scheme for turbulence (IOPTK = 1, ITFORM = 1).
F	As experiment A now with the wave-current interaction module enabled (IOPTW = 1). The wave height HSUN and the wave period TSUN are uniform in time and are set to respectively 2 m and 10 s.

Table 2.5.7: Output values for the parameters of the test case **csbio**.

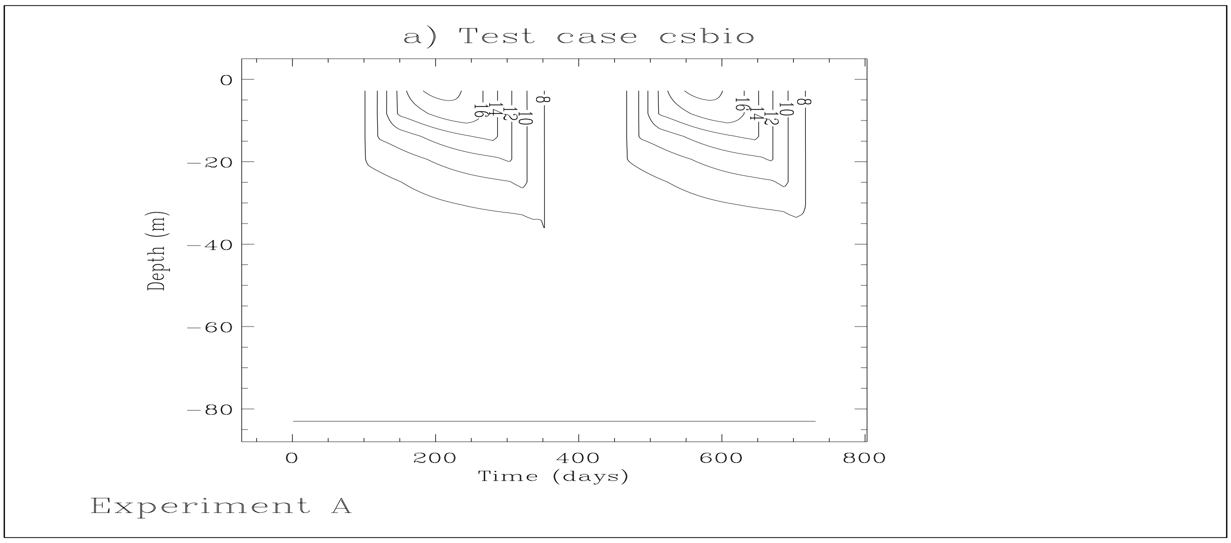
Parameter	Month	A	B	C	D	E	F
TMEAN	2	6.541	6.541	6.541	6.541	6.542	6.540
	4	7.078	7.078	7.078	7.078	7.003	7.082
	6	8.583	8.583	8.583	8.583	8.491	8.588
	8	9.317	9.317	9.317	9.317	9.403	9.322
	10	9.075	9.075	9.075	9.075	9.248	9.085
	12	7.760	7.760	7.760	7.760	7.805	7.825
	14	6.292	6.292	6.292	6.292	6.293	6.333
	16	6.929	6.929	6.929	6.929	6.848	6.958
	18	8.435	8.435	8.435	8.435	8.335	8.464
	20	9.169	9.169	9.169	9.169	9.253	9.199
	22	8.933	8.933	8.933	8.933	9.103	8.967
	24	7.635	7.635	7.635	7.635	7.684	7.722
P2XINT	2	47.25	47.71	61.67	45.18	53.98	45.78
	4	198.0	196.8	205.1	195.9	168.7	190.8
	6	9.829	41.53	10.75	12.77	8.962	10.45
	8	3.558	18.64	5.145	6.916	3.816	5.958
	10	1.160	15.36	2.238	7.890	1.177	2.804
	12	0.1784	19.51	0.3813	16.09	0.1396	1.095
	14	0.1719	9.938	0.5293	10.09	0.1200	1.246
	16	82.37	149.4	112.0	159.5	102.6	119.7
	18	20.63	39.74	17.87	12.85	18.60	12.77
	20	3.992	17.74	5.073	6.641	3.981	5.958
	22	1.295	14.48	2.171	6.222	1.230	2.802
	24	0.1955	19.67	0.3612	11.38	0.1457	0.9796
P2CINT	2	100.4	102.2	105.4	100.0	101.3	100.7
	4	214.8	246.3	243.6	210.9	216.6	210.9
	6	287.0	530.9	319.8	294.4	246.2	283.2
	8	226.0	581.2	262.5	237.9	182.4	227.4
	10	184.1	575.6	222.7	195.1	139.1	188.4
	12	154.5	580.6	195.2	182.6	109.3	161.1
	14	136.1	558.3	176.3	162.2	94.28	143.5
	16	139.1	612.4	193.7	211.8	106.9	166.2
	18	286.7	870.8	333.6	303.4	230.9	273.4
	20	233.0	912.3	284.2	248.5	173.8	225.2
	22	190.2	898.2	244.8	205.5	130.9	189.1
	24	159.7	898.0	217.0	186.3	101.9	163.2

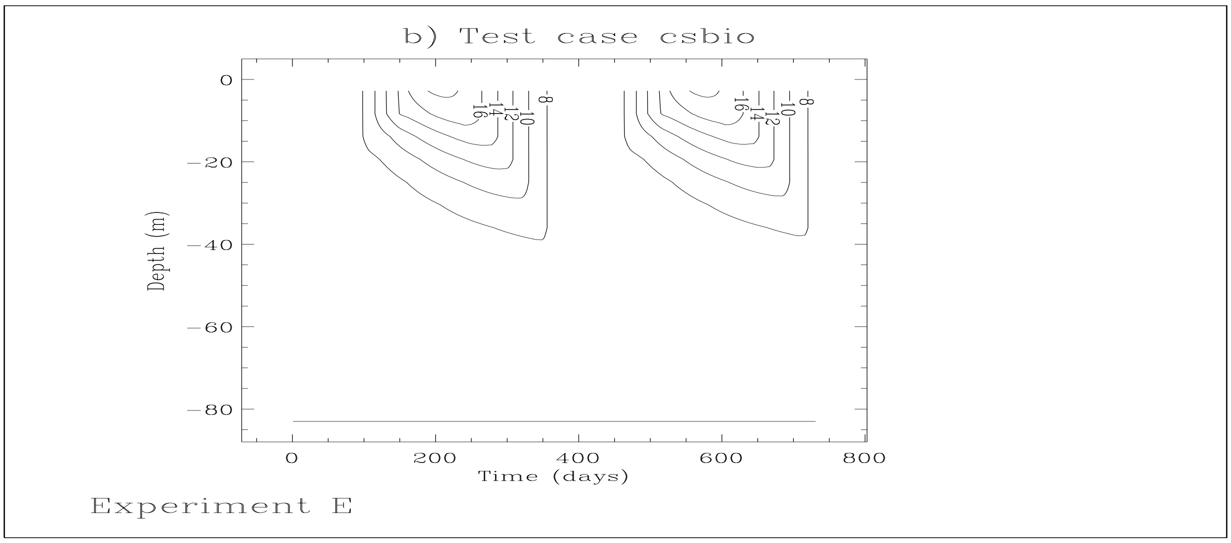
Table 2.5.7: continued.

Parameter	Month	A	B	C	D	E	F
P2NHSINT	2	30.78	30.68	26.85	31.39	28.89	31.22
	4	46.50	36.83	51.65	44.48	51.57	40.73
	6	37.47	28.97	37.33	37.57	33.88	37.15
	8	27.10	26.03	27.24	27.11	25.33	27.79
	10	25.64	25.69	26.08	26.29	23.63	26.78
	12	24.68	25.57	24.99	26.64	22.31	25.77
	14	29.87	30.57	29.99	30.02	29.20	29.86
	16	25.78	27.97	26.90	33.43	26.15	28.63
	18	41.49	28.42	39.63	37.40	38.30	37.65
	20	27.47	26.19	27.36	27.10	25.75	27.69
	22	25.86	25.93	26.18	26.26	23.82	26.78
	24	24.81	25.92	25.09	26.19	22.46	25.72
SEDMEAN	2	0.6513	0.6513	*	0.6932	0.4198	0.7622
	4	0.6084	0.6084	*	0.6453	0.2757	0.7214
	6	0.6256	0.6256	*	0.6589	0.2755	0.7306
	8	0.6226	0.6226	*	0.6560	0.2723	0.7289
	10	0.6049	0.6049	*	0.6412	0.2734	0.7187
	12	0.5847	0.5847	*	0.6244	0.2707	0.7251
	14	0.6438	0.6438	*	0.6866	0.3980	0.7579
	16	0.6091	0.6091	*	0.6457	0.2760	0.7216
	18	0.6254	0.6254	*	0.6586	0.2755	0.7309
	20	0.6223	0.6223	*	0.6556	0.2722	0.7289
	22	0.6053	0.6053	*	0.6414	0.2734	0.7191
	24	0.5711	0.5711	*	0.6112	0.2715	0.7223
SEDDEV	2	0.5054	0.5054	*	-0.4691	0.7665	0.5901
	4	1.063	1.063	*	-0.3898	0.8357	1.216
	6	1.611	1.611	*	-0.1049	0.8464	1.947
	8	2.222	2.222	*	0.2408	0.8786	2.754
	10	2.766	2.766	*	0.5698	0.9573	3.551
	12	3.227	3.227	*	0.7629	0.9561	4.122
	14	3.917	3.917	*	0.5269	1.609	4.910
	16	4.447	4.447	*	0.7331	1.719	5.502
	18	5.040	5.040	*	1.043	1.669	6.282
	20	5.603	5.603	*	1.327	1.695	7.056
	22	6.187	6.187	*	1.624	1.738	7.766
	24	6.535	6.535	*	1.783	1.709	8.302

Table 2.5.7: continued.

Parameter	Month	A	B	C	D	E	F
BALNMONT	2	-0.0173	0.0002	0.0004	-0.0159	-0.0250	-0.0097
	4	-0.0434	0.0004	0.0004	-0.0421	-0.0701	-0.0304
	6	0.0128	-0.0015	-0.0001	0.0111	0.0236	0.0084
	8	0.0018	0.0001	0.0006	0.0012	0.0031	0.0017
	10	-0.0457	0.0004	0.0005	-0.0471	-0.0362	-0.0035
	12	0.0267	0.0000	0.0001	0.0281	-0.0016	-0.0013
	14	-0.0099	-0.0002	0.0001	-0.0132	-0.0089	-0.0061
	16	-0.0108	0.0005	0.0005	-0.0335	-0.0357	-0.0259
	18	0.0084	-0.0015	-0.0002	0.0090	0.0131	0.0066
	20	0.0022	-0.0001	0.0008	0.0012	0.0026	0.0013
	22	-0.0393	0.0004	0.0007	-0.0392	-0.0299	-0.0014
	24	0.0221	-0.0002	-0.0001	0.0215	-0.0041	-0.0035
BALNMAX	2	0.0186	0.0001	0.0001	0.0184	0.0170	0.0124
	4	0.0710	0.0001	0.0001	0.0733	0.0666	0.0546
	6	0.0851	0.0001	0.0001	0.0840	0.0717	0.0601
	8	0.0610	0.0001	0.0001	0.0616	0.0538	0.0468
	10	0.0445	0.0001	0.0001	0.0460	0.0417	0.0354
	12	0.0365	0.0001	0.0001	0.0448	0.0297	0.0296
	14	0.0159	0.0001	0.0001	0.0182	0.0128	0.0139
	16	0.0285	0.0001	0.0001	0.0604	0.0274	0.0303
	18	0.0710	0.0001	0.0001	0.0702	0.0619	0.0523
	20	0.0528	0.0001	0.0001	0.0511	0.0465	0.0396
	22	0.0397	0.0001	0.0001	0.0402	0.0346	0.0303
	24	0.0331	0.0001	0.0001	0.0381	0.0249	0.0255
P2XMAX	1	14.56	7.827	14.10	11.69	16.73	9.615
	2	17.54	7.091	18.25	12.57	17.46	13.70
IDAYMAX	1	104	94	102	106	98	101
	2	124	99	119	111	120	113
DEPMAX	1	30.43	2.767	30.43	30.43	30.43	24.90
	2	30.43	2.767	30.43	30.43	30.43	24.90
BALNYEAR	1	-0.0184	-0.0012	0.0038	-0.0288	-0.0611	-0.0134
	2	0.0049	-0.0017	0.0040	0.0026	0.0126	0.0049





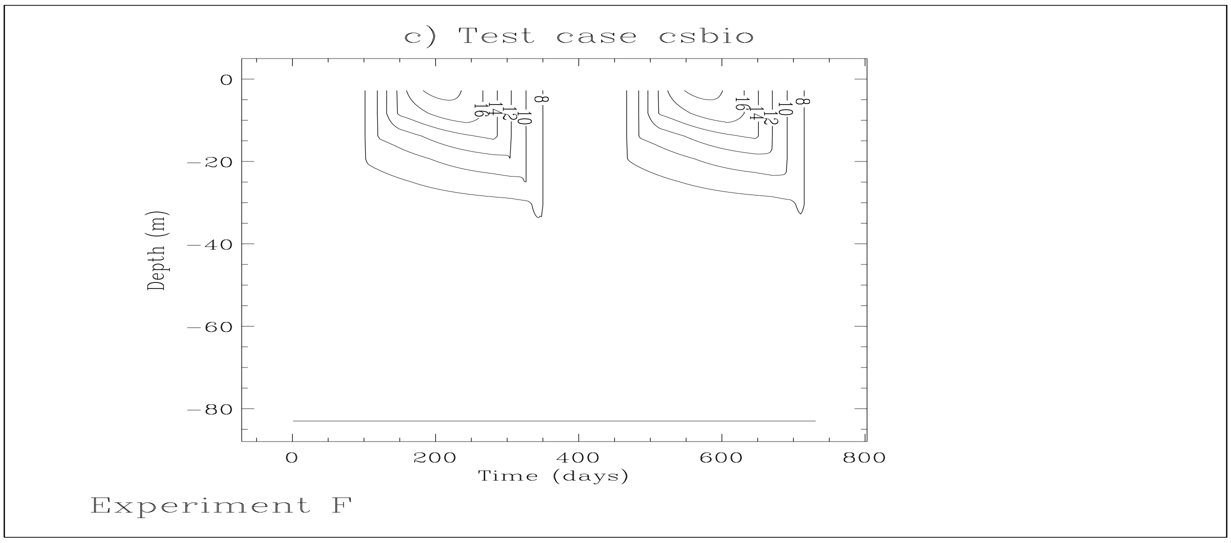


Figure 2.5.3: Depth-time contour plots of temperature for two seasonal cycles and experiment A (a), E (b) and F (c). Contour intervals are at 2^0C.

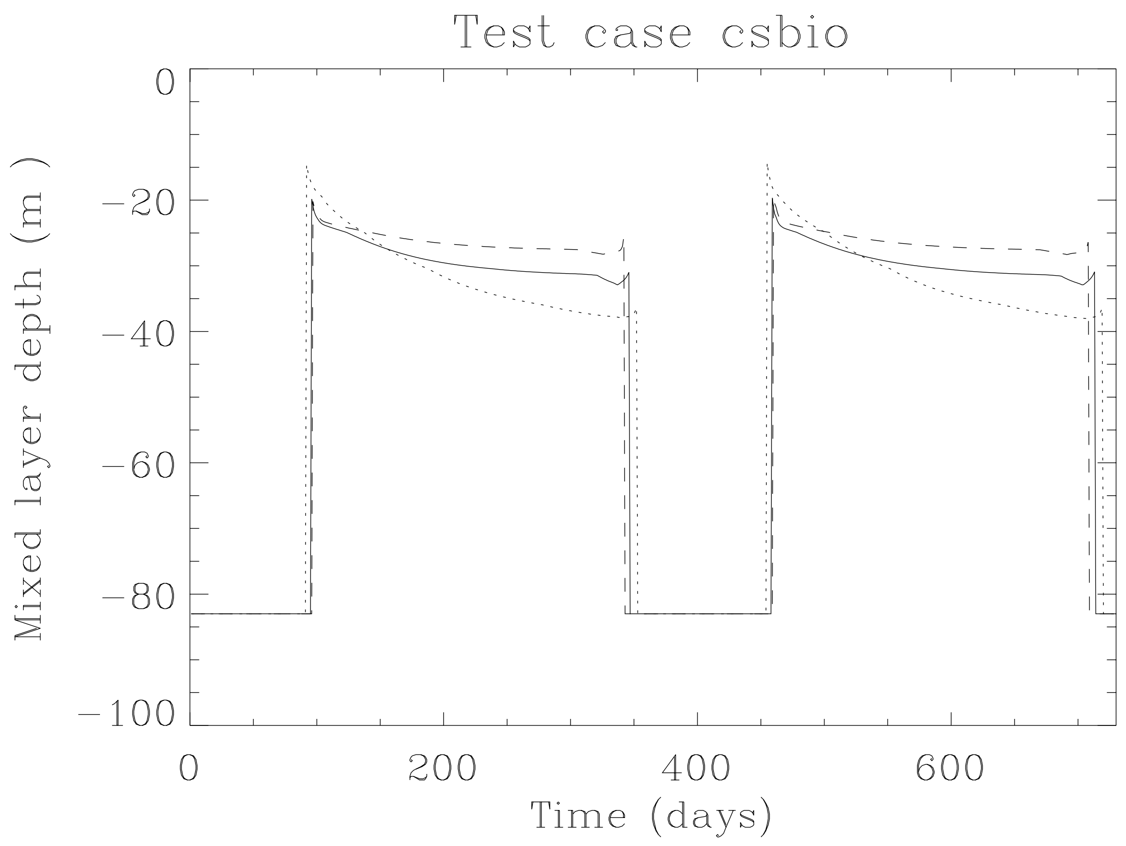
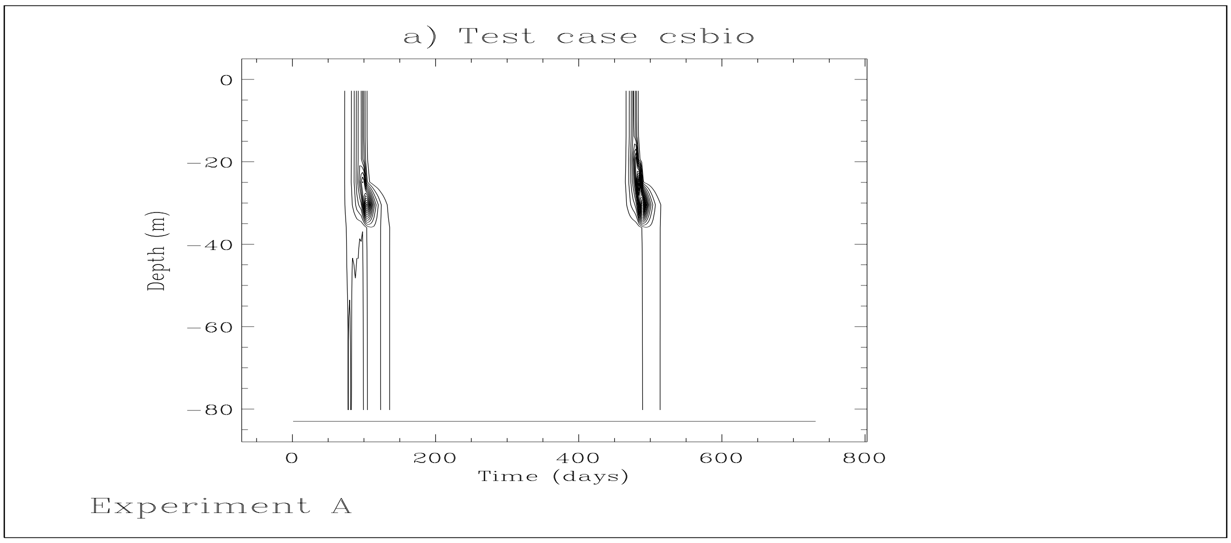


Figure 2.5.4: Time series of the thermocline depth for experiment A (solid), E (dots) and F (dashes).



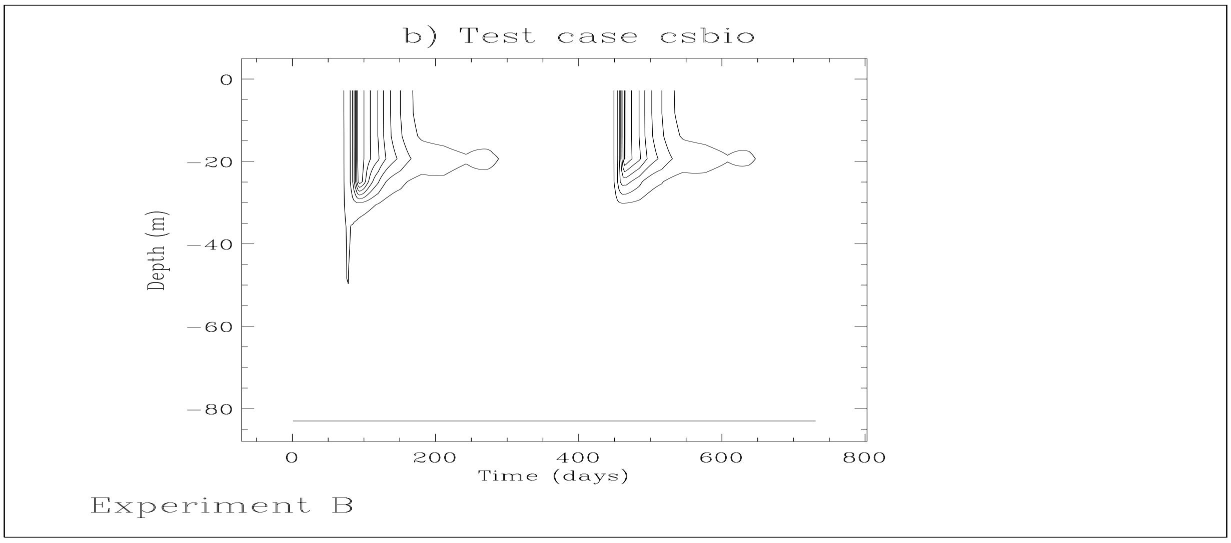


Figure 2.5.5: Depth-time contour plots of chlorophyll for experiment A (a) and B (b). Contour intervals are at 1 mg Chl m^{-3}.

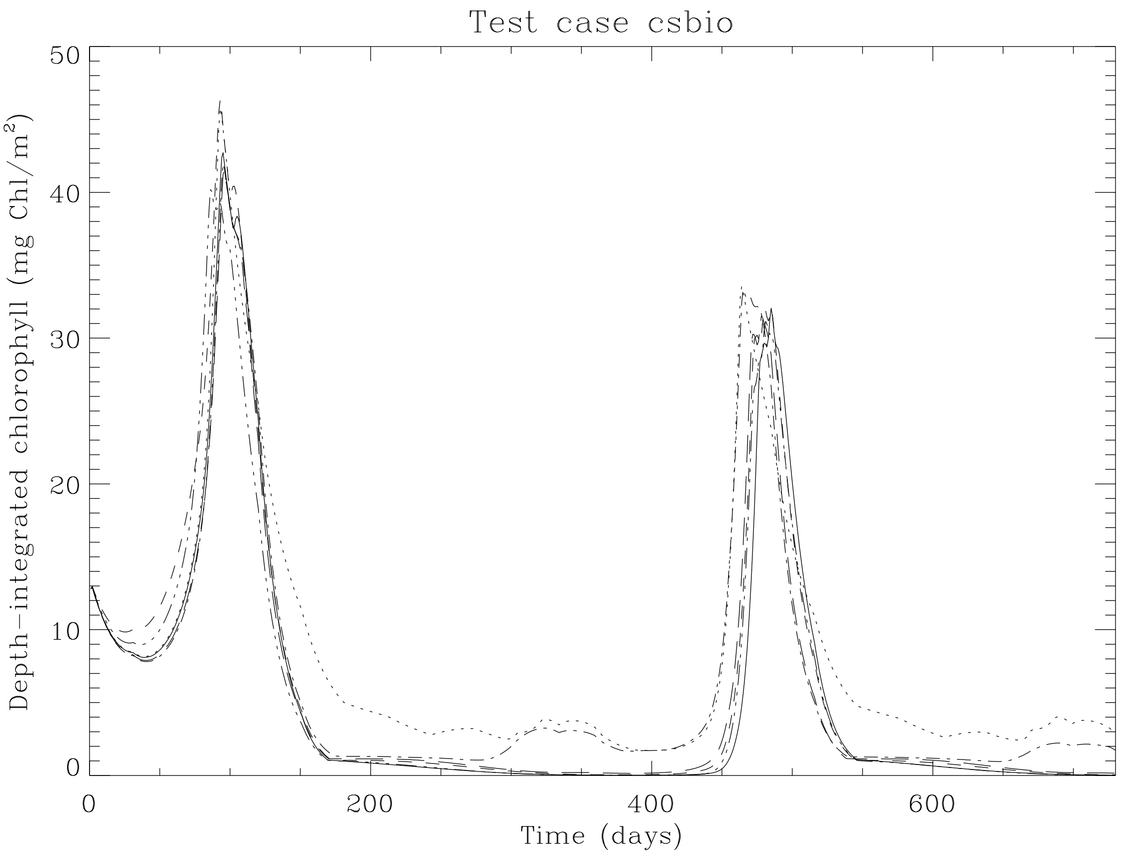
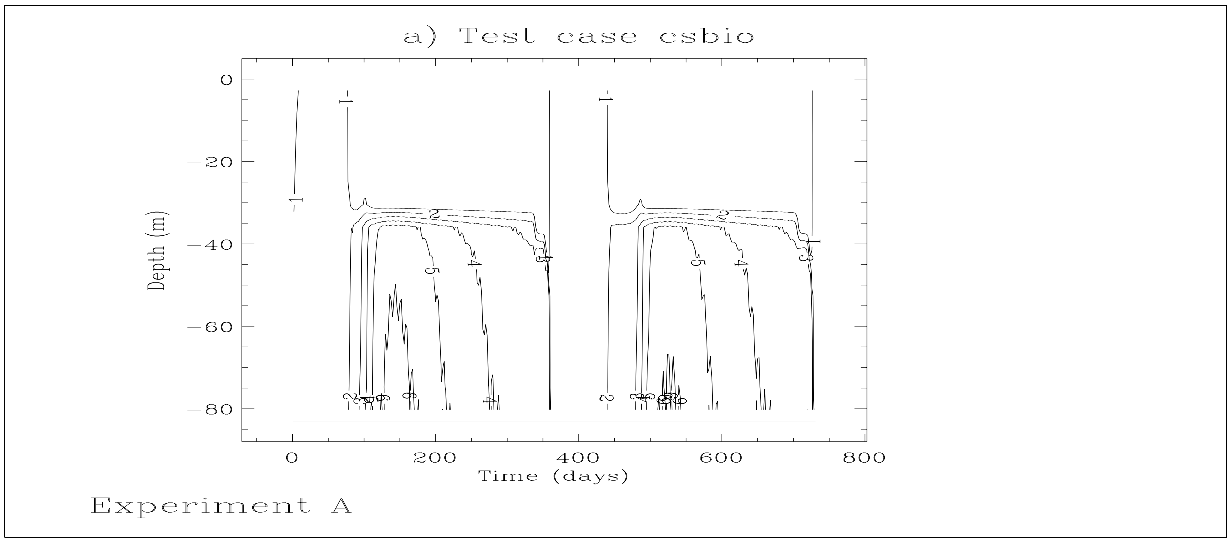
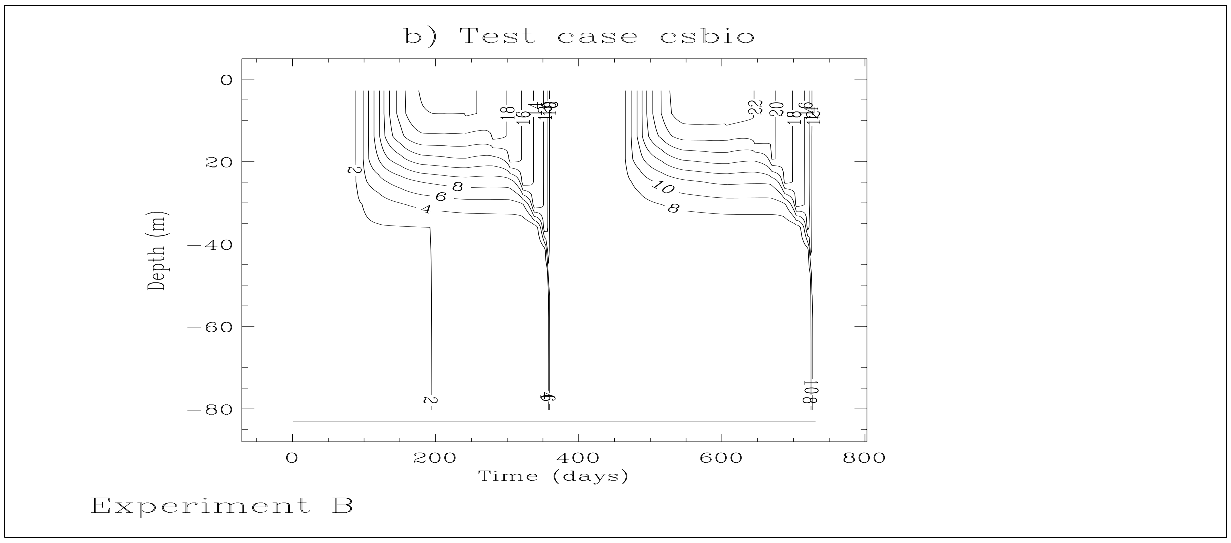


Figure 2.5.6: Time series of depth-integrated chlorophyll (mg Chl m^{-2}): experiment A (solid), B (dots), C (dashes), D (dash-dots), E (dash and 3 dots), F (long dashes).





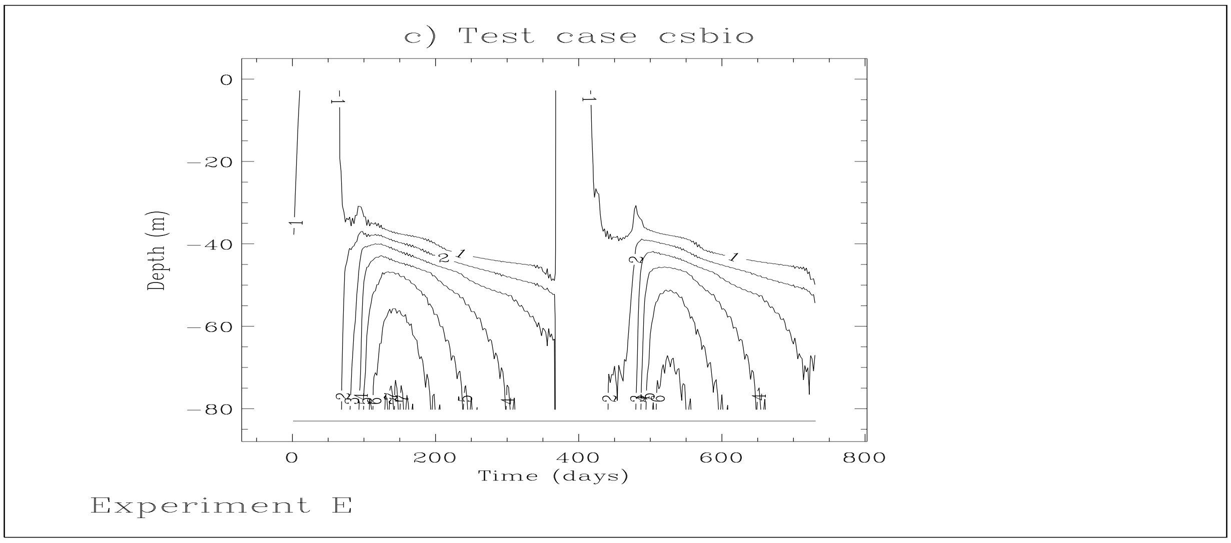
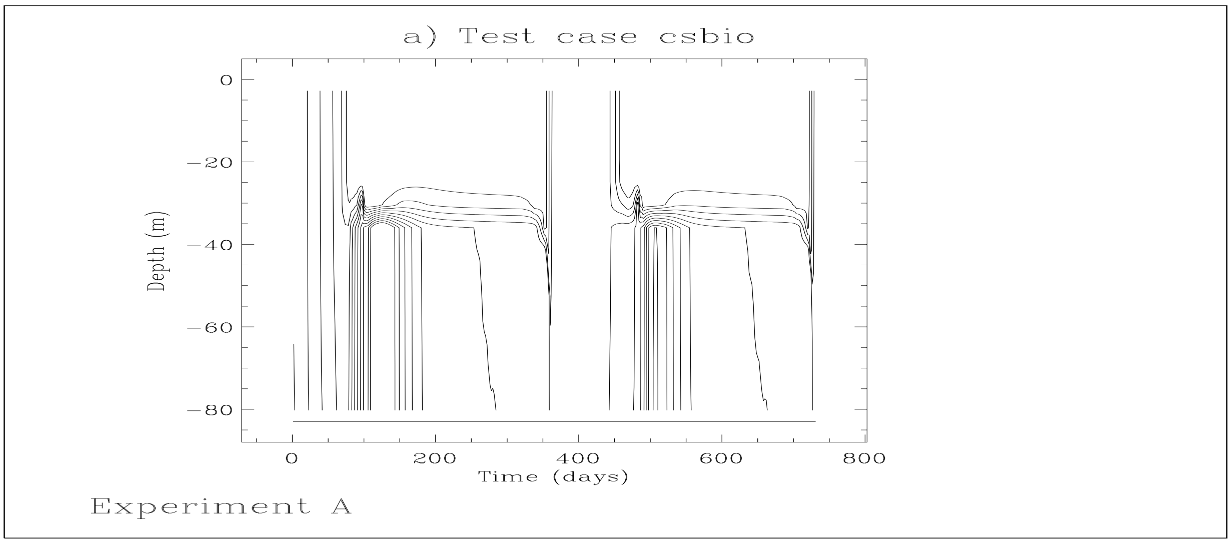
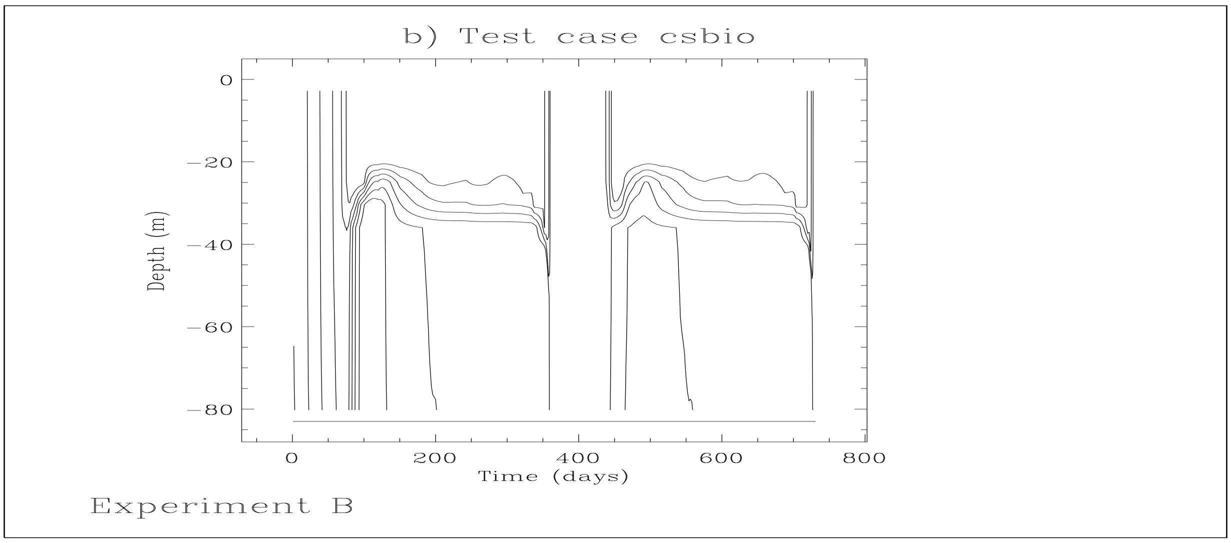


Figure 2.5.7: Depth-time contour plots of detrital carbon for experiment A (a), B (b), E (c). Contour intervals are at 2 mmol C m^{-3} for experiment B, 1 otherwise.





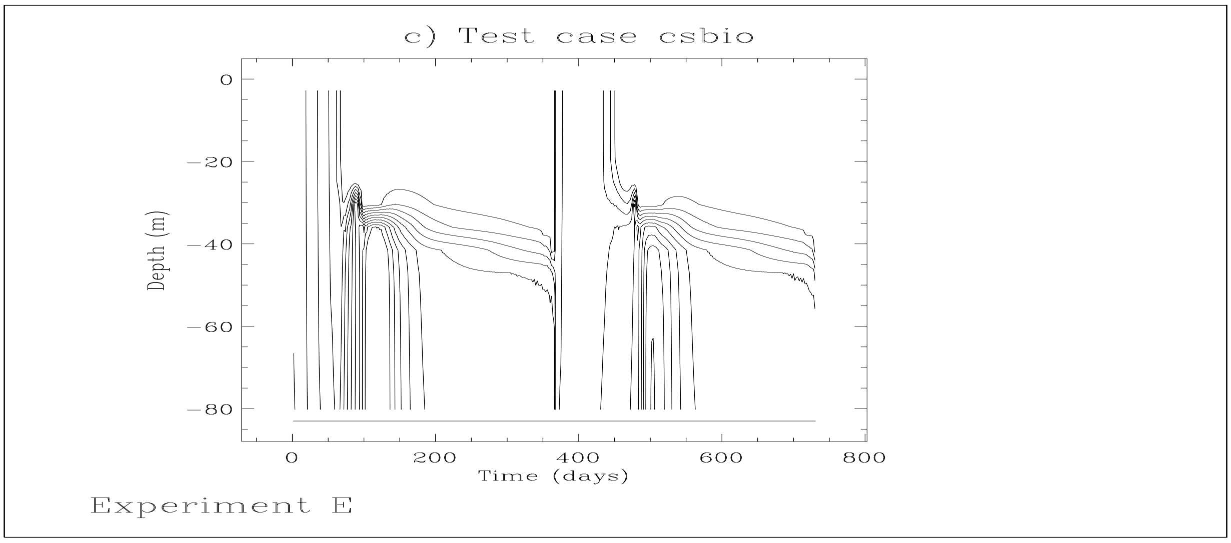
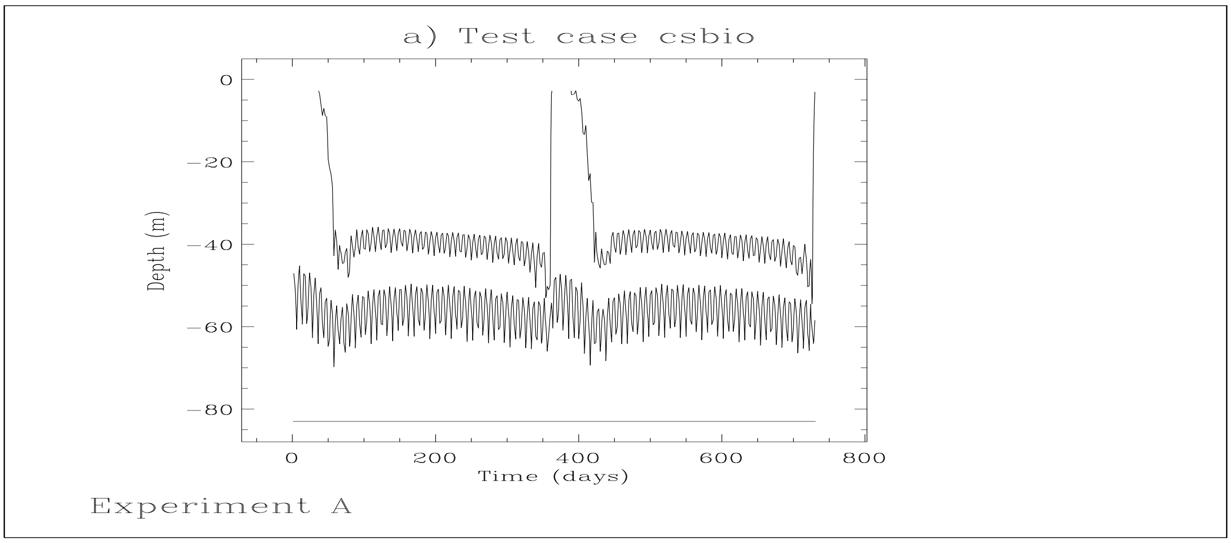


Figure 2.5.8: Depth-time contour plots of ammonium for experiment A (a), B (b), E (c). Contour intervals are at 0.1 mmol N m^{-3}.



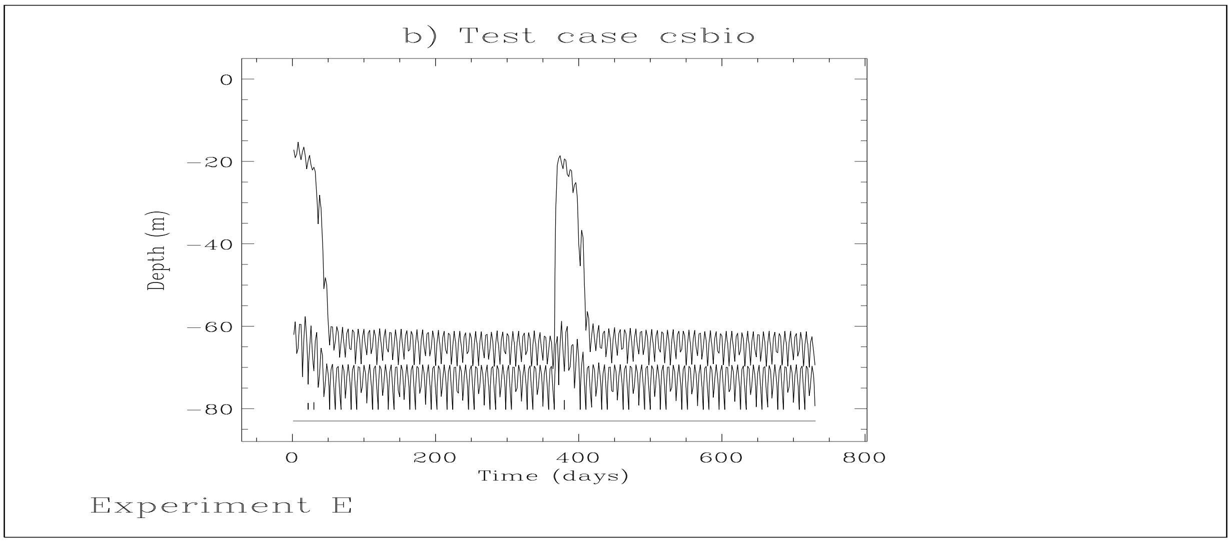


Figure 2.5.9: Depth-time contour plots of iSPM (g m^{-3}) for experiment A (a) and E (b). For convenience, only the 0.1 and 0.6 contour lines are plotted.

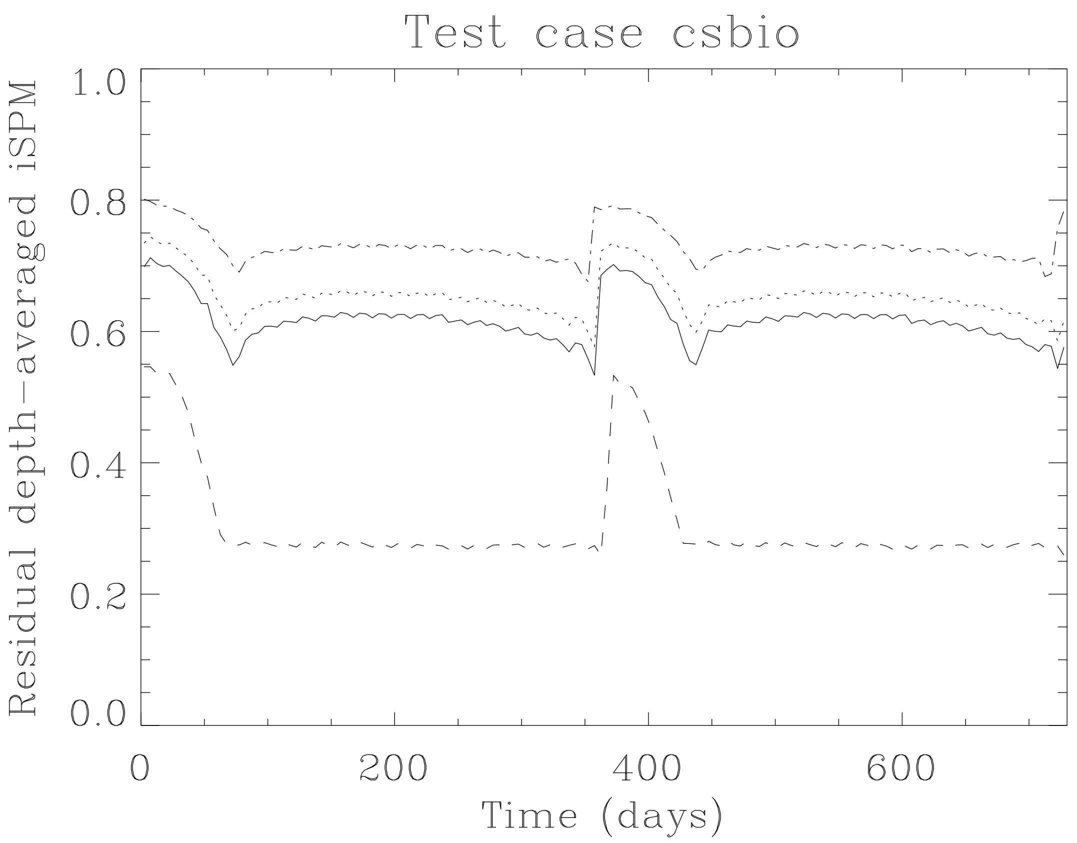


Figure 2.5.10: Time series of the tidally averaged values of depth-averaged iSPM (g m^{-3}): experiment A (solid), D (dots), E (dashes), F (dash-dots).

Chapter 6

NOMADS Tracer Experiments

6.1 Setup of the experimental study

An intercomparison model study has been performed within the Concerted Action NOMADS (North Sea Model Advection Dispersion Study) funded by the European Union during the period 1995–1997 (Proctor, 1997). One of the objectives of NOMADS was to compare the results of different advection-dispersion models by analysing the spatial and temporal coherence of the simulation results for a well defined realistic simulation of the North Sea. The study area is the part of the North Sea between 51^0 N and 55^0 40′ N (see Figure 2.6.1). Four experiments have been defined. The first one involved 2-D models only and will not be discussed here. The second and third one are tracer experiments while the fourth analyses the surface salinity and the volume transport along two transects.

- Six release stations are considered in experiment 2. They are located at (see also Fig 2.6.1):

Mid–Channel (M)	51^0 30′ N 2^0 00′ E
Rhine (R)	52^0 00′ N 3^0 40′ E
Thames (T)	51^0 30′ N 1^0 10′ E
Humber (H)	53^0 35′ N 0^0 30′ E
Elbe (E)	54^0 05′ N 8^0 20′ E
Central North Sea (C)	54^0 00′ N 3^0 00′ E

  At each of these points three particles are released situated initially at the surface, mid-depth and bottom. The trajectories of the 18 particles are determined for two periods of one month.

- The same release stations are taken in experiment 3. The difference is that an initial mass of 1 kg is now uniformly distributed over the vertical. The location of the centre of mass is followed in time for the same two periods. Other output parameters were defined but will not further be considered here.

- In experiment 4 the surface salinity and the accumulated volume are evaluated at daily intervals along two West-East transects. The latter parameter is defined by

$$\overline{V}_{acc}(\lambda, t) = \int_0^t \overline{V}(\lambda, \phi_{tr}, \tau)\, d\tau \tag{2.6.1}$$

  where $\overline{V}$ is the depth integral of the northward current, λ the longitude and the latitude ϕ_{tr} of the two transects is taken at 52.5^0 N and 54.5^0 N (Figure 2.6.1). Simulated periods are the same as in the two previous experiments.

The periods of the simulations are March 1989 and August 1989. A series of data files for the initial conditions and the forcing have been made available. This is further discussed in the section below. The objectives of this chapter are to show some results with the particle tracking method and to compare the Eulerian and Lagrangian transport models for contaminants, which are implemented in COHERENS. For a full discussion of the NOMADS intercomparison study, which involved seven different 3-D models, the reader is referred to Proctor (1997).

6.2 Model setup and experiments

The NOMADS experiments are of interest for the present study mainly for the following two reasons. Firstly, the particle tracking method, as applied in the single release experiments, provides a picture of the residual flow in certain parts of the North Sea. Extraction of the residual field from current data is often a difficult task in the presence of tides with a large number of constituents and time-varying wind forcing. Secondly, it allows the comparison of the Lagrangian and Eulerian approaches, available in COHERENS for the advection and dispersion of dissolved pollutants.

6.2.1 Initialisation of the model

The complete model setup for the NOMADS study is described below. Model setup parameters are further listed in Table 2.6.1.

Grid

- The model equations are solved on a spherical grid.
- The computational domain is the North Sea between $55^0\ 40'$ N with the southern boundary at the Dover Strait taken at 51^0 N.
- The bathymetry is supplied at $5'$ latitude by $7.5'$ longitude giving an horizontal resolution of 8–9 km. A uniform σ-grid is applied using 20 vertical levels.
- The open boundaries of the model area are the southern and northern open sea boundaries at 51^0 N and $55^0\ 40'$ N and 11 river boundaries for fresh water input. The following rivers are considered in the study:

- Thames, Wash, Humber, Tees and Tyne along the East coast of England
- Scheldt, Rhine, Ysselmeer, Ems, Weser and Elbe along the continental coast

Physics

- The program solves the 2-D and 3-D equations for the current, temperature and salinity fields and for the sea surface elevation, described in Section III-1.1.2. The initial temperature and salinity fields are obtained by a (3-D) linear interpolation of externally supplied data arrays. The currents are initialised by first running the program for a spin-up period of two days with zero initial currents and open boundary forcing for the 2-D mode, but without stratification, contaminants and river discharges. The procedure is the same as the one used in the test cases **river** and **plume**.
- The downward absorption of solar irradiance is parameterised using the formulation presented in Section III-1.4. The parameters of the optical module are taken as appropriate for a water type II in the classification scheme of Paulson and Simpson (1977):

$$(R_1, k_1, k^w_{20}, \epsilon^S, R_2) = (0.77, 1/1.5, 1/14, 0, 1) \qquad (2.6.2)$$

- The simulations are performed using the default turbulence scheme, described in Section III-1.2.3.

Numerics

- The time steps for the 2-D and 3-D mode are set to respectively 30 and 600 s.
- The currents are advected with the upwind scheme while the default TVD scheme is used for all scalars (temperature, salinity and contaminants).
- Horizontal diffusion of momentum and scalar quantities is disabled (default setting of the program).

Contaminants

Both the Lagrangian and the Eulerian modules are activated.

- In analogy with the NOMADS experiments 2 and 3 two different particle releases are considered in the Lagrangian module.
- At each of the six release stations, whose positions are given above, three particles are released initially, situated respectively at the surface, halfway between the surface and the bottom, and the bottom. The program calculates the trajectories of all 18 particles during the simulation.
- A series of 5000 particles, distributed uniformly over the vertical and with a total mass of 1 kg, are additionally released at the initial time. Each particle

has the same mass. The coordinates $x_{ci}(t)$ of the centre of mass for each of the six releases are determined by the program with the usual formula

$$x_{ci}(t) = \sum_{n=1}^{N} x_{in}(t)/N \tag{2.6.3}$$

where x_{in} are the (spherical) coordinates of particle n and N equals the number of particles inside the domain. Note that N can be smaller than the initial number of 5000 since particles which traverse one of the open or land boundaries of the computational domain, are considered as lost to the system.

- The Eulerian transport module is only initialised for the NOMADS experiment 3. A vertically uniform concentration $C_i(1 \leq i \leq 6)$ is defined initially at the centre $(x_i^*,\, y_i^*)$ of the grid cell, nearest to the corresponding release station. They are given by

$$\begin{aligned} C_i(\lambda,\phi,x_3,0) &= (H\Delta\lambda\Delta\phi)^{-1} \quad \text{if} \quad (\lambda,\phi) \quad \text{is inside the grid cell} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{2.6.4}$$

  where H is the total water depth, $\Delta\lambda$ and $\Delta\phi$ are the horizontal grid spacings (in m) and C_i is expressed in kg/m^3 so that the total amount of contaminant matter for each distribution equals 1 kg. Note that the Eulerian module consumes significantly more CPU time compared to the Lagrangian one, since the program has to solve six additional advection-diffusion equations.

Surface forcing and open boundary conditions

- Surface meteorological data (wind magnitude and direction, atmospheric pressure, air temperature and relative humidity) are supplied at 3-hourly intervals from a larger area meteorological model with a coarser resolution. The external data are linearly interpolated on the model grid. Cloud coverage was derived from satellite images with a daily value taken to be uniform over the whole domain. A zero surface salinity flux is assumed.
- The external (2-D) forcing at the two open sea boundaries is given by the sum of a residual and tidal component. The conditions are formulated in the present study using an harmonic expansion for the sea surface elevation of the form given by equation (3.1.198):

$$\zeta = F_{har} = A_0 + \sum_{n=1}^{N_T} A_n \cos(\omega_n t + \varphi_{n0} - \varphi_n) \tag{2.6.5}$$

  where $N_T = 27$ is the number of tidal constituents and ω_n are the tidal frequencies. The amplitudes A_n and phases φ_n are supplied externally for each open sea boundary location and are assumed uniform in time during the simulation, while the residual part A_0 is obtained from a data file at hourly intervals. The values of (A_0, A_n), φ_n and φ_{n0} are stored into respectively the setup arrays AMPOBV, PHAOBV and PHASE0. For more information about this type of open boundary condition, see Section III-1.6.3a.

- The magnitude of the volume transport $\overline{U}$ or $\overline{V}$ is specified at river boundaries using

$$\overline{U} = \pm Q_d/W = \pm cF_{har}\,,\ \overline{V} = \pm Q_d/W = \pm cF_{har} \tag{2.6.6}$$

  where Q_d is the river discharge in m^3/s, W the width of the river mouth taken to be equal to one horizontal grid spacing and $c = (gh)^{1/2}$ the barotropic wave speed. The upper sign is taken at western/southern boundaries, the lower sign at eastern/northern boundaries, implying an outflow condition at all river boundaries. Monthly mean values are used for all river discharges so that Q_d remains uniform in time during the simulation. The values of F_{har} are stored into the residual parts of respectively the setup arrays AMPOBU and AMPOBV. The procedure is analogous to the one used in the setup of the North Sea case study (see Section II-7.2.3d). More details concerning this type of open boundary condition are given in Section III-1.6.3a.

- The following open boundary conditions are used for 3-D quantities
  - At open sea boundaries the zero gradient condition (3.1.209) is considered for the velocity deviation v' which is the default condition implemented in the program. At river boundaries either u' or v' is set to zero implying a uniform discharge at all depths.
  - A zero gradient condition is taken for salinity at the open sea boundaries. In analogy with the test case **plume** salinity is specified in a two-layer form at the river mouth with a fresh water value of 10 PSU in a surface layer of 6 m depth and a zero gradient condition below.
  - The default zero gradient condition is applied for temperature and contaminants at all boundaries.

  For more details about the boundary conditions for 3-D variables, see Section III-1.6.3b.

6.2.2 Setup of the experiments

The simulations are performed for two periods: March 1989 and August 1989. The first one is characterised by heavier winds and no thermal stratification. The second period corresponds to a summer regime with lighter winds and the presence of a thermocline in the northern part of the domain. An important feature of the Lagrangian particle module is the implementation of the random walk method for the parameterisation of vertical diffusion (see Chapter III-3). The following four experiments have therefore been set up (see also Table 2.6.2).

Table 2.6.1: Model setup for the NOMADS North Sea study.

parameter	value	purpose
param.inc		
NC	88	number of grid cells in the X-direction (longitude)
NR	56	number of grid cells in the Y-direction (latitude)
NZ	20	number of grid cells in the vertical
NOBU	9	number of open boundary points in the X-direction (river boundaries only)
NOBV	89	number of open boundary points in the Y-direction (southern, northern open sea and river boundaries)
NVPROF	12	number of profiles at open boundaries (different profile at each of the 11 river boundaries and a zero gradient condition at open sea boundaries)
NCON	27	number of tidal forcing frequencies
NCONC	6	number of contaminant distributions used in the Eulerian transport module
MAXNOP	50000	maximum number of particles allowed in the Lagrangian transport module
DEFCON		
IGTRH	1	spherical grid
IOPTC	1	Eulerian module enabled without vertical diffusion (experiments A and B)
	2	Eulerian contaminant module enabled with vertical diffusion (experiments C and D)
IOPTP	1	Lagrangian module enabled without vertical diffusion (experiments A and B)
	2	Lagrangian contaminant module enabled with vertical diffusion (experiments C and D)
IADVC	1	upwind scheme for the advection of momentum
IOPTSA	1	update of the salinity field enabled
IOPTHE	2	update of the temperature field enabled, optical module switched on
IOPTD	2	thermal and salinity expansion coefficients evaluated with the general equation of state of seawater (see Section III-1.5)
IOPTM	3	program reads meteorological input at all grid points and selected time intervals, heat fluxes calculated by the program
IDRAG	2	surface drag coefficient evaluated using the Smith and Banke (1975) formulation
IOUTP	1	the program writes an output file with the positions of a series of particles defined by the user

Table 2.6.1: continued.

parameter	value	purpose
IBDATE	03010000	start date for experiments A and C (MMDDHHMM)
	08010000	start date for experiments B and D (MMDDHHMM)
IEDATE	04010000	end date for experiments A and C (MMDDHHMM)
	09010000	end date for experiments B and D (MMDDHHMM)
IBYEAR	1989	start year
IEYEAR	1989	end year
DELT	30	time step for the external (2-D) mode
IC3D	20	time step for the internal (3-D) mode divided by DELT
ICMET	360	time interval (3 hours) for meteorological data input divided by DELT
IC2BC	120	time step for 2-D open boundary input (river discharge; residuals, amplitudes and phases at open sea boundaries) divided by DELT
ICRFORM	'U'	file with initial/final conditions written in binary format
BCSFORM	'U'	file with open boundary data written in binary format
METFORM	'U'	file with meteorological data written in binary format
OUTTIT		prefix name of the user-defined data output files
ATCF1UN	1/1.5	diffusive attenuation coefficient k_1 (m^{-1}) for infrared radiation
ATCF2UN	1/14	diffusive attenuation coefficient k^w_{20} (m^{-1}) for non-infrared radiation (PAR) and pure seawater
R1OPTUN	0.77	infrared fraction R_1 of solar irradiance
R2OPTUN	1	hyperexponential PAR decay disabled
EPSSAL	0	no salinity dependence for the PAR diffuse attenuation coefficient k_2
CDZ0UN	0.0035	uniform roughness length used for the evaluation of the bottom stress (m)
DEFGRID		
GX0(NC+1)		X-coordinates (longitude) of the cell corners (between 2^0 W and 9^0 E, $\Delta\lambda = 7.5'$)
GY0(NR+1)		Y-coordinates (latitude) of the cell corners (between 51^0 N and 55^0 $40'$ N, $\Delta\phi = 5'$)
DEP(NR,NC)		array of mean water depths (m) read from a data file
NPIX(NR,NC+1)		cell face pointer array at U-nodes (0 at solid boundaries, 1 at internal wet faces and 3 at river boundary faces)
NPIY(NR+1,NC)		cell face pointer array at V-nodes (0 at solid boundaries, 1 at internal wet faces, 2 at open sea and 3 at river boundary faces)

Table 2.6.1: continued.

parameter	value	purpose
DEFGRID		
(IOBU,JOBU)(1:NOBU)		cell face indices of the open boundary points at U-nodes (river boundary locations are read from a data file)
(IOBV,JOBV)(1:NOBV)		cell face indices of the open boundary points at V-nodes (river boundary locations are read from a data file)
DEFOB2D		
ITYPOBU(1:NOBU)	4	type of open boundary condition (2-D mode) at the western/eastern river boundaries (see equation (2.6.6))
ITYPOBV(1:NOBV)		type of open boundary condition (2-D mode) at the southern/northern open boundaries 3: open sea boundaries (see equation (2.6.5)) 4: river boundaries (see equation (2.6.6))
SIGMA(1:NCON)		tidal forcing frequencies (rad/s) from a data file
PHASE0(1:NCON)		initial phases φ_{n0} (rad) from a data file
DEFOB3D		
IVPOBU(1:NOBU)		profile number at western/eastern boundaries: different profiles are taken at river boundaries (profile numbers 2–10)
IVPOBV(1:NOBV)		profile number at southern/northern boundaries • a zero gradient condition is taken at all open sea boundaries (profile number 1) • different profiles are taken at river boundaries (profile numbers 11–12)
UVP(1:NZ,12)		open boundary profiles for the horizontal current • UVP(1:NZ,1): the default zero gradient condition • UVP(1:NZ,2:12): zero velocity deviations at river boundaries
SVP(1:NZ,12)		open boundary profiles for salinity • SVP(1:NZ,1): the default zero gradient condition • SVP(1:NZ,2:12): the 2-layer profiles (PSU) at river boundaries

Table 2.6.1: continued.

parameter	value	purpose
DEFICS		
S(NZ,NR,NC)		initial salinity field (PSU) obtained from an external data file and linear interpolation on the model grid
T(NZ,NR,NC)		initial temperature field (^{0}C) obtained from an external data file and linear interpolation on the model grid
CONCN(NZ,NR,NC,1:NCONC)		initial contaminant distributions for the Eulerian module (see equation (2.6.4))
NUMP	30018	total number of released particles • particles 1–5000: uniform release at station 1 • . . . • particles 25001–30000: uniform release at station 6 • particles 30001–30006: surface particle at stations 1–6 • particles 30007–30012: mid-depth particle at stations 1–6 • particles 30013–30018: bottom particle at stations 1–6
(IT,JT,KT)(1:NUMP)		cell indices in the X-, Y-, Z-direction where the particles are located initially
(XT,YT,ZT)(1:NUMP)		initial particle coordinates with respect to the cell centre (XT in longitude, YT in latitude, ZT in σ-coordinates)
LST(1:NUMP)	1	particle labels
ST(1:NUMP)		particle masses (kg): 1 for single releases, 1/5000 for uniform releases
WRFCDAT		
(WINDU2,WINDV2,P2,SAT2,HUM2)(NR,NC)		wind components (m/s), atmospheric pressure (N/m^2), air temperature (^{0}C), relative humidity (0–1) obtained from data files at 3-hourly intervals. The external data are linearly interpolated on the model grid.
CLOUD2(NR,NC)		spatially uniform array of values for the cloud coverage (0–1), supplied at daily intervals from a data file
EVAPR(NR,NC)	0.0	uniform evaporation minus precipitation rate (zero surface salinity flux)

Table 2.6.1: continued.

parameter	value	purpose
WROBDAT		
AMPOBU(1:NOBU,0:NCON)		residuals and amplitudes (m) for the harmonic forcing at western/eastern boundaries • AMPOBU(1:NOBU,0): values of $F_{har} = Q_d/(cW)$ as defined by (2.6.6), representing the forcing at western/eastern river boundaries • AMPOBU(1:NOBU,1:NCON): tidal amplitudes at river boundaries (set to 0)
PHAOBU(1:NOBU,1:NCON)		phases (rad) of the tidal forcing at western/eastern (river) boundaries (0 everywhere)
AMPOBV(1:NOBV,0:NCON)		residuals and amplitudes (m) for the harmonic forcing at southern/northern boundaries • AMPOBV(1:87,0): residual values (m) for the harmonic forcing at the southern and northern open sea boundaries, as given by the parameter A_0 in the expansion (2.6.5) for the sea surface elevation. Values are obtained from a data file at hourly intervals (determined by the counter IC2BC) and linearly interpolated along the boundaries. • AMPOBV(88:89,0): values of $F_{har} = Q_d/(cW)$ as defined by (2.6.6), representing the forcing at southern/northern river boundaries • AMPOBV(1:87,1:NCON): amplitudes (m) of the tidal forcing at the two open sea boundaries (supplied from an external data file) as given by the coefficients A_n in the expansion (2.6.5) for the sea surface elevation • AMPOBV(88:89,1:NCON): tidal amplitudes at river boundaries (set to 0)

Table 2.6.1: continued.

parameter value	purpose
PHAOBV(1:NOBV,1:NCON)	phases (rad) of the tidal forcing at southern/northern boundaries • PHAOBV(1:87,1:NCON): phases φ_n (rad) of the tidal forcing at the two open sea boundaries (supplied from an external data file), as defined in (2.6.5) • PHAOBV(88:89,1:NCON): phases at river boundaries (taken to be 0)

Table 2.6.2: Definition of the experiments for the NOMADS study.

Test	Purpose
A	March 1989 period without vertical diffusion (IOPTP = IOPTC = 1)
B	August 1989 period without vertical diffusion (IOPTP = IOPTC = 1)
C	March 1989 period with vertical diffusion (IOPTP = IOPTC = 2)
D	August 1989 period with vertical diffusion (IOPTP = IOPTC = 2)

A : March 1989 period, vertical diffusion disabled

B : August 1989 period , vertical diffusion disabled

C : March 1989 period, vertical diffusion enabled

D : August 1989 period , vertical diffusion enabled

Note that horizontal diffusion is disabled by default in all simulations.

6.2.3 Setup of model output

Model output is organised with the following files.

- *defmod.par*
  - time series output ('U'-format) at all grid points and at weekly intervals
  - 30 particle trajectories ('A'-format) at 3-hourly intervals
- *defout.f*

  - 3-D fields: temperature, salinity
  - 2-D fields: depth-integrated contaminant distributions for the uniform release experiment as obtained either from the Eulerian as from the Lagrangian module

- *defpar.f*
  - the positions of the 18 single release particles
  - the position of the 6 centres of mass for the uniform release experiment according to the Lagrangian module
  - the position of the 6 centres of mass for the uniform release experiment according to the Eulerian module

- *print.f*: Two files are created.
  - The first one contains the coordinates of the same particles, defined in *defpar.f*, at 3-hourly intervals. The positions are now expressed in the form (R_p,θ_p,σ_p) where (R_p,θ_p)[1] are the particle's polar coordinates with respect to the initial release point and σ_p the dimensionless vertical σ-coordinate (between 0 at the bottom and 1 at the surface).
  - The second file stores the values of the accumulated volume $\overline{V}_{acc}$, as defined by (2.6.1), at daily intervals and for the two transects at 52.5^0 N and 54.5^0 N.

The trajectories of the single release particles are shown in Figures 2.6.2a–b for experiments A and C and in Figures 2.6.3a–b for experiments B and D. The results for the March runs with and without diffusion are compared in Figures 2.6.4a–b displaying the trajectories of the centres of mass using respectively the Lagrangian and Eulerian method. Time series of σ_p for the surface, mid-depth and bottom particle are shown in Figure 2.6.5a for the Rhine release in March, Figure 2.6.5b for the Central North Sea in August, Figure 2.6.5c for the Humber in March and Figure 2.6.5d for the Elbe in August. The time evolution of σ_p for the centre of mass of the Rhine release in March is illustrated in Figure 2.6.6a according to the simulations with or without diffusion and the two transport models. Figure 2.6.6b is the same as the previous one now for the Thames release in August. The surface distributions of the depth-integrated concentrations in August after four weeks of simulation are shown in Figures 2.6.7a–d for the runs with and without diffusion and the Lagrangian and Eulerian methods. Finally, the evolution in time of the accumulated volume $\overline{V}_{acc}$ is plotted in Figures 2.6.8a–d for the two transects and the two simulated periods. Note that the X-axis in the contour plots represents the relative distance to the East coast of England in % and varies between 0 at the British and 100 at the Continental coast.

Values of the particle coordinates at intervals of 5 days in the form (R_p,θ_p,σ_p) are further listed in Table 2.6.3 for the surface, mid-depth and bottom releases at all stations in March

[1]Note that θ is measured anticlockwise and $\theta = 0^0$ is taken eastwards.

and for the run without diffusion. The corresponding values for the run with diffusion are given in Table 2.6.4. The coordinate values of the centre mass for four stations in August are compared in Table 2.6.5 for the runs with and without diffusion and the Lagrangian and Eulerian method.

6.3 Results

In the case of the single release experiments using the Lagrangian method clearly different results are obtained for the runs with and without vertical diffusion. The surface particles travel over much larger distances in the latter case, especially during March, whereas the directions of the surface, mid-depth and bottom particles, with exception of the Thames and Mid–Channel releases, are more aligned in the runs with diffusion. These tendencies are observed by comparing Figures 2.6.2a-2.6.3a with 2.6.2b-2.6.3b and Table 2.6.3 with 2.6.4. The cause of these discrepancies are the large vertical fluctuations produced by the random walk method. With exception of the Rhine and Elbe cases, the surface and bottom particles stay at the same relative depth in absence of diffusion (Table 2.6.3) while they are displaced at random over the whole water column during the simulation when diffusion is enabled (Table 2.6.4). It is this random motion which makes the trajectories less dependent on the initial vertical position in the water column. The results of experiments A and B may therefore give a more realistic description of the residual flow field.

The differences are smaller for the uniform releases (Figure 2.6.4a) since the random fluctuations are now averaged over a larger (5000) number of particles. The vertical oscillations however do persist, although with smaller amplitudes, as can be observed from Figures 2.6.6a–b. Vertical diffusion is a less important factor — at least qualitatively — in the simulations using the Eulerian method (Figure 2.6.4b and Table 2.6.5). Its main effect is that the centre of mass remains almost exactly at its initial vertical position in the middle of the water column ($\sigma_p = 0.5$), due to turbulent mixing. A general tendency, seen e.g. in Figure 2.6.6a, is that even in the absence of diffusion the centre of mass deviates more strongly from the mean vertical position in the Lagrangian compared to the Eulerian case.

The trajectories of the single releases in experiments A and B (no diffusion) allow to make a few observations concerning the residual and wind-driven flows. A general tendency is that, compared to August, the surface particles travel over larger distances during March which can be explained by the stronger winds prevailing during the latter period.

- The surface particle in the Rhine (March) moves northeastwards while the bottom particle follows an eastward until the 15th of March, southward until the 25th and finally a northeastward trajectory at the end of the month (Table 2.6.3). These periods are clearly related to the vertical position of the bottom particle (Figure 2.6.5a) which is displaced upwards after the 15th towards the middle of the water column and rises to the surface after the 25th. The observations imply the existence of firstly a northeastward surface current along the Dutch coast and an (almost) opposite bottom flow below the Rhine plume, and secondly of important up- and downwelling

Table 2.6.3: Coordinates (R_p,θ_p,σ_p) with R_p in km and θ_p in degrees, at 5-day intervals of the surface, mid-depth and bottom particle for the single release stations in March and for the run without vertical diffusion (experiment A).

type[a]	parameter	day					
		5	10	15	20	25	30
Mid–Channel							
s	R_p	28.6	51.4	97.0	99.7	159.8	187.9
s	θ_p	36.1	42.3	36.6	34.0	32.1	43.7
s	σ_p	0.974	0.974	0.975	0.977	0.953	0.866
m	R_p	11.5	28.8	47.9	55.5	62.3	57.8
m	θ_p	119.7	135.8	121.9	134.1	118.7	101.9
m	σ_p	0.517	0.470	0.518	0.572	0.782	0.809
b	R_p	9.4	21.3	30.9	30.9	42.0	43.7
b	θ_p	124.5	139.7	132.4	144.8	140.8	136.4
b	σ_p	0.025	0.023	0.025	0.023	0.035	0.039
Rhine							
s	R_p	50.7	83.8	113.2	107.4	112.9	110.6
s	θ_p	44.4	48.4	50.8	52.5	54.9	56.4
s	σ_p	0.976	0.976	0.682	0.717	0.125	0.191
m	R_p	8.9	14.3	54.2	78.1	137.2	146.1
m	θ_p	19.7	6.9	47.9	59.1	54.1	51.3
m	σ_p	0.386	0.763	0.894	0.902	0.635	0.801
b	R_p	5.1	13.1	17.7	26.6	16.1	20.7
b	θ_p	24.3	339.9	335.6	297.9	293.0	49.4
b	σ_p	0.016	0.026	0.107	0.547	0.323	0.932
Thames							
s	R_p	28.2	51.5	70.5	62.8	100.3	109.1
s	θ_p	342.9	14.5	21.3	15.5	8.9	14.7
s	σ_p	0.979	0.979	0.981	0.981	0.980	0.983
m	R_p	4.4	8.1	23.4	38.6	58.6	66.2
m	θ_p	228.9	301.9	338.7	308.8	334.7	355.8
m	σ_p	0.409	0.463	0.513	0.622	0.975	0.975
b	R_p	5.0	4.2	3.3	5.6	7.5	7.2
b	θ_p	203.8	241.5	289.4	229.8	222.6	227.4
b	σ_p	0.024	0.023	0.028	0.025	0.027	0.027

[a]The surface, mid-depth and bottom particle are denoted by respectively s, m, b.

Table 2.6.3: continued.

type	parameter	day 5	10	15	20	25	30
Humber							
s	R_p	18.0	44.2	98.8	122.0	187.3	193.1
s	θ_p	21.7	343.3	334.0	330.7	340.9	348.0
s	σ_p	0.983	0.989	0.992	0.990	0.990	0.991
m	R_p	10.1	30.4	53.8	63.3	96.8	102.0
m	θ_p	48.3	326.3	310.3	308.7	316.9	314.1
m	σ_p	0.558	0.644	0.623	0.637	0.721	0.711
b	R_p	6.5	17.3	17.8	15.5	9.9	7.8
b	θ_p	45.7	0.1	341.0	345.1	319.7	295.5
b	σ_p	0.027	0.027	0.028	0.026	0.032	0.043
Elbe							
s	R_p	34.4	46.2	64.7	72.3	60.6	57.7
s	θ_p	67.2	73.4	61.2	55.5	64.4	74.2
s	σ_p	0.972	0.970	0.965	0.698	0.216	0.172
m	R_p	11.4	17.4	30.1	29.5	49.0	42.0
m	θ_p	52.4	89.0	127.2	124.4	151.1	156.2
m	σ_p	0.589	0.690	0.629	0.503	0.481	0.367
b	R_p	4.7	9.4	15.7	7.9	27.4	27.2
b	θ_p	37.6	61.8	115.5	91.7	151.3	170.9
b	σ_p	0.032	0.042	0.051	0.114	0.257	0.773
Central North Sea							
s	R_p	54.1	78.5	136.2	157.7	246.2	252.0
s	θ_p	356.3	16.9	16.4	16.8	14.1	13.5
s	σ_p	0.976	0.976	0.975	0.974	0.970	0.969
m	R_p	29.2	21.2	26.0	28.6	32.0	20.5
m	θ_p	338.2	339.1	327.6	315.1	307.8	296.5
m	σ_p	0.526	0.510	0.504	0.496	0.465	0.476
b	R_p	11.9	6.5	5.5	9.7	15.8	22.2
b	θ_p	348.2	301.3	252.9	235.3	201.6	193.2
b	σ_p	0.028	0.028	0.030	0.031	0.032	0.031

Table 2.6.4: As Table 2.6.3 now for the March run with diffusion (experiment C).

type[a]	parameter	day 5	10	15	20	25	30
Mid–Channel							
s	R_p	11.5	22.8	36.4	26.9	25.6	38.9
s	θ_p	77.1	96.1	78.6	83.2	74.3	67.4
s	σ_p	0.941	0.232	0.961	0.197	0.998	0.616
m	R_p	9.7	20.9	33.5	19.5	18.2	29.3
m	θ_p	72.5	91.3	71.9	73.7	49.6	61.9
m	σ_p	0.881	0.177	0.911	0.937	0.872	0.630
b	R_p	14.0	27.8	33.5	22.3	20.3	33.1
b	θ_p	93.8	86.1	77.0	90.8	72.7	71.5
b	σ_p	0.359	0.665	0.733	0.728	0.962	0.128
Rhine							
s	R_p	19.5	35.0	56.7	64.3	100.8	127.7
s	θ_p	63.4	61.9	63.1	70.8	61.9	63.1
s	σ_p	0.009	0.952	0.025	0.805	0.193	0.343
m	R_p	16.6	32.8	54.1	55.5	87.9	103.9
m	θ_p	65.0	61.8	66.0	72.7	63.4	67.8
m	σ_p	0.114	0.996	0.060	0.505	0.031	0.596
b	R_p	20.8	38.8	67.9	75.5	102.6	115.1
b	θ_p	61.2	62.4	59.9	57.6	58.5	62.6
b	σ_p	0.763	0.021	0.898	0.059	0.995	0.679
Thames							
s	R_p	4.6	40.1	53.8	57.2	67.3	76.6
s	θ_p	270.6	314.5	339.8	340.9	345.8	358.3
s	σ_p	0.079	0.392	0.563	0.313	0.926	0.601
m	R_p	5.1	12.4	41.1	56.9	82.1	82.1
m	θ_p	257.9	320.2	306.1	315.2	317.8	317.8
m	σ_p	0.953	0.667	0.533	0.125	0.487	0.487
b	R_p	10.7	47.6	49.3	40.8	31.2	23.0
b	θ_p	344.0	348.7	13.2	5.4	8.4	13.1
b	σ_p	0.865	0.094	0.041	0.389	0.930	0.998

[a]The surface, mid-depth and bottom particle are denoted by respectively s, m, b.

Table 2.6.4: continued.

type	parameter	day 5	10	15	20	25	30
Humber							
s	R_p	9.2	31.1	57.6	69.4	106.6	128.5
s	θ_p	42.0	314.9	311.0	313.6	319.7	318.7
s	σ_p	0.010	0.104	0.784	0.868	0.397	0.601
m	R_p	5.9	26.8	50.1	62.5	106.2	127.6
m	θ_p	43.5	333.5	312.9	313.2	317.4	316.2
m	σ_p	0.016	0.903	0.884	0.173	0.762	0.878
b	R_p	13.2	30.1	43.7	48.7	73.6	75.1
b	θ_p	38.6	356.1	329.1	326.1	321.0	317.9
b	σ_p	0.274	0.883	0.264	0.380	0.718	0.339
Elbe							
s	R_p	15.2	23.2	35.0	33.9	35.1	47.2
s	θ_p	73.7	109.8	123.1	116.4	121.9	122.8
s	σ_p	0.174	0.753	0.352	0.441	0.286	0.854
m	R_p	16.0	22.8	34.4	28.5	33.8	42.6
m	θ_p	53.2	86.4	118.6	110.0	139.6	133.0
m	σ_p	0.610	0.910	0.918	0.524	0.150	0.639
b	R_p	14.9	21.2	36.7	37.4	40.9	48.5
b	θ_p	85.7	120.1	129.4	120.7	132.4	127.3
b	σ_p	0.202	0.750	0.960	0.578	0.882	0.955
Central North Sea							
s	R_p	27.7	20.6	28.2	33.6	43.4	30.9
s	θ_p	348.0	5.8	358.7	359.5	355.8	0.6
s	σ_p	0.984	0.902	0.078	0.905	0.044	0.626
m	R_p	24.8	18.6	21.3	20.0	43.0	31.8
m	θ_p	346.1	8.5	2.2	8.2	353.3	359.6
m	σ_p	0.409	0.569	0.300	0.006	0.024	0.614
b	R_p	23.4	22.2	29.0	27.9	40.8	29.5
b	θ_p	339.6	348.8	8.7	356.8	344.6	351.6
b	σ_p	0.030	0.975	0.012	0.764	0.502	0.615

motions in the area. Both flow characteristics are typical for plume areas such as the Rhine.

- As known from observations (Hill et al., 1994), the Central North Sea station is closely situated during the summer to a thermal front in the West-East direction, which generates an along-front circulation driven by the cross-frontal density gradient. As observed in Figure 2.6.3a this flow is directed to the East at the surface and to the West in the middle and bottom parts of the water column. This is validated by the observation from Figure 2.6.5b that the relative vertical positions of both the surface and the bottom particle remain unchanged during the whole period. The vertical motion of the mid-depth particle, on the other hand, shows the existence of a steady upwelling current in analogy with the observational data reported in Hill et al. (1994). The thermal front is absent in March. As seen in Figure 2.6.2a all particles now move to the East with a speed which strongly decreases with depth.

- The trajectories of the particles for the Humber release both in March and in August imply a mainly southeastward flow along the East coast of England. The current magnitude decreases with depth while its direction is nearly depth-independent.

- The paths of the Elbe release particles are clearly different during March and August. While the displacement is towards the North at the surface and to the West at larger depths for the March period, the surface particle moves to the Southeast during August. This may indicate that the residual flow is mainly governed by local wind conditions. Interesting phenomena are the cyclonic orbit described by the mid-depth particle and the occurrence of an internal semi-diurnal tide in the middle of the water column (Figure 2.6.5d). Note also the presence of strong upwellings and downwellings.

- Less clear information can be gained for the Thames and Mid–Channel releases. The surface particles move mainly in the northeastward direction, while the mid-depth and bottom particles are displaced in different directions. Interesting to note is the large distance travelled by the surface Mid–Channel particle during March.

The NOMADS experiment 3 can be considered as an ideal test case to examine the spreading of pollutant material released at a point source. The surface distributions of the depth-integrated suspended material for the uniform releases at the six stations are shown in Figures 2.6.7a–d after four weeks for the August simulation. The four different cases (Eulerian and Lagrangian, diffusion and no diffusion) are presented. The following comments can be made:

- Even in the absence of vertical diffusion, the patches, obtained with the Eulerian method, are spread over a significantly larger area compared to the Lagrangian distributions. An immediate consequence is that the concentrations are much larger in the latter case. A comparison between Figures 2.6.7a,c and b,d shows a difference of a factor 10. (Note that the contour intervals are given on a $\log_{10}$-scale).

- In the Eulerian case no substantial differences are seen between the runs with and without diffusion, although the patches appear somewhat larger in the latter case. This is due to the vertical mixing process which constrains the radial expansion of surface material by advective transport. The Lagrangian patches have a more filamentary shape in the absence of diffusion, as can be observed for the Rhine, Humber and Elbe releases, while they have a more compact form in the run with diffusion (e.g. Central North Sea release).

- The patches of the Thames and Mid–Channel releases have been fused together in all cases. This occurs after only a few days of simulation (not shown). The result is not unexpected since the flow patterns at the two stations have different directions at different depths (see Figures 2.6.2–2.6.3) so that the depth-integrated distributions are smeared out over a larger area.

Finally, some results concerning the NOMADS experiment 4 are briefly presented. Time series of the accumulated volume (in units of 10^5 m^2) as a function of coastal distance are illustrated in Figures 2.6.8a–d for the two transects and the two simulated periods. Note that the British coast is at the left and the Continental coast at the right in the plots. A southward directed coastal jet along the English coast is clearly observed in the four figures. Its magnitude is larger in March than during the summer. A clear difference is seen between the northern transect in summer and the other three cases. In the latter cases the jet has a width of 20% of the total transect distance while the flow has a northward component over the remaining part of the transects. The results for the northern transect during summer show a narrower width of 10% and a secondary northwards directed jet of the same magnitude appears to the East of the coastal jet. With exception of a few isolated points the flow in the remaining parts of the transect is now mainly southward contrary to the previous cases. It is unclear how these results are related to the circulation pattern in the area of the thermal front since this requires a more elaborate analysis about the vertical structure of the current field.

In summary, the Lagrangian method has some clear advantages. Firstly, particle tracking is an efficient tool to make predictions concerning the residual and wind-driven flows. Secondly, it allows the simulation of advection and dispersion of small scale patches of pollutant material without numerical diffusion. Thirdly, the method is more efficient in preserving horizontal concentration gradients compared to the Eulerian approach. However, preference should be given to the Eulerian method in the case of large scale distributions (see Section III-3.1). The weak point of the Lagrangian model is the parameterisation of vertical diffusion using the Monte-Carlo method. Better results are to be expected by taking a larger number of particles for the uniform release experiment. Further tests are required to find an optimal number of particles. It is clear that the random method, designed for a large number of particles, cannot be applied to single release experiments.

Table 2.6.5: Coordinates (R_p,θ_p,σ_p) with R_p in km and θ_p in degrees, at 5-day intervals of the centre of mass for the uniform releases in August at the Rhine, Thames, Humber and Central North Sea stations according to the Lagrangian and Eulerian methods and the runs without diffusion (experiment B) and with diffusion (experiment D).

parameter	type[a]	day 5	10	15	20	25	30
Rhine							
R_p	L0	7.7	2.7	12.0	18.5	29.2	13.6
R_p	L1	4.8	15.3	28.2	42.4	50.4	37.7
R_p	E0	5.0	4.5	14.7	20.1	24.1	8.3
R_p	E1	5.2	4.0	11.9	15.8	19.3	4.4
θ_p	L0	233.6	36.2	41.4	49.0	62.2	82.5
θ_p	L1	227.7	68.8	65.6	67.4	68.1	73.9
θ_p	E0	207.8	71.3	53.5	55.0	52.1	42.9
θ_p	E1	205.1	76.8	55.5	57.8	55.7	42.6
σ_p	L0	0.504	0.527	0.563	0.692	0.759	0.642
σ_p	L1	0.644	0.786	0.623	0.624	0.598	0.626
σ_p	E0	0.518	0.553	0.560	0.595	0.625	0.648
σ_p	E1	0.500	0.502	0.500	0.500	0.501	0.502
Thames							
R_p	L0	14.7	27.7	34.2	44.3	50.3	51.5
R_p	L1	17.2	32.6	38.1	46.9	50.5	51.4
R_p	E0	8.1	13.6	11.9	18.2	22.6	17.3
R_p	E1	7.8	13.3	9.5	18.1	22.9	17.5
θ_p	L0	340.0	333.4	341.2	345.0	351.7	336.6
θ_p	L1	336.5	331.2	336.8	342.9	352.5	339.0
θ_p	E0	0.0	360.0	1.4	352.9	351.9	322.9
θ_p	E1	3.4	1.7	355.6	335.5	335.3	312.8
σ_p	L0	0.477	0.517	0.664	0.696	0.678	0.614
σ_p	L1	0.588	0.599	0.587	0.613	0.637	0.603
σ_p	E0	0.495	0.518	0.585	0.606	0.617	0.594
σ_p	E1	0.498	0.502	0.501	0.498	0.502	0.500

[a] L0 means Lagrangian without diffusion, L1 Lagrangian with diffusion, E0 Eulerian without diffusion and E1 Eulerian with diffusion.

Table 2.6.5: continued.

parameter	type	day 5	10	15	20	25	30
Humber							
R_p	L0	16.5	24.0	33.1	53.9	61.3	85.0
R_p	L1	16.9	27.2	37.6	66.7	73.4	104.2
R_p	E0	12.0	15.1	21.7	37.9	45.4	59.4
R_p	E1	11.5	13.9	17.6	33.2	39.2	50.6
θ_p	L0	2.8	355.5	344.9	327.2	319.2	316.9
θ_p	L1	356.8	349.6	340.5	321.7	315.0	315.1
θ_p	E0	13.5	356.6	349.2	328.9	322.2	319.1
θ_p	E1	15.2	358.7	346.2	324.2	316.0	316.6
σ_p	L0	0.552	0.589	0.581	0.591	0.611	0.632
σ_p	L1	0.601	0.814	0.585	0.578	0.553	0.588
σ_p	E0	0.537	0.556	0.588	0.616	0.625	0.617
σ_p	E1	0.500	0.502	0.501	0.501	0.501	0.501
Central North Sea							
R_p	L0	26.7	29.0	35.3	43.5	53.6	59.6
R_p	L1	26.8	30.3	44.7	58.5	77.2	82.8
R_p	E0	24.7	26.6	29.0	34.7	43.3	50.4
R_p	E1	24.7	26.3	28.3	34.1	43.1	50.9
θ_p	L0	171.1	174.5	178.9	181.0	181.8	181.7
θ_p	L1	152.2	156.9	164.9	166.2	168.0	165.4
θ_p	E0	162.4	166.0	171.9	176.1	180.5	180.4
θ_p	E1	162.0	165.9	172.1	176.8	181.5	181.5
σ_p	L0	0.511	0.521	0.519	0.521	0.510	0.521
σ_p	L1	0.497	0.596	0.567	0.639	0.667	0.685
σ_p	E0	0.514	0.523	0.523	0.523	0.518	0.533
σ_p	E1	0.514	0.520	0.519	0.533	0.533	0.539

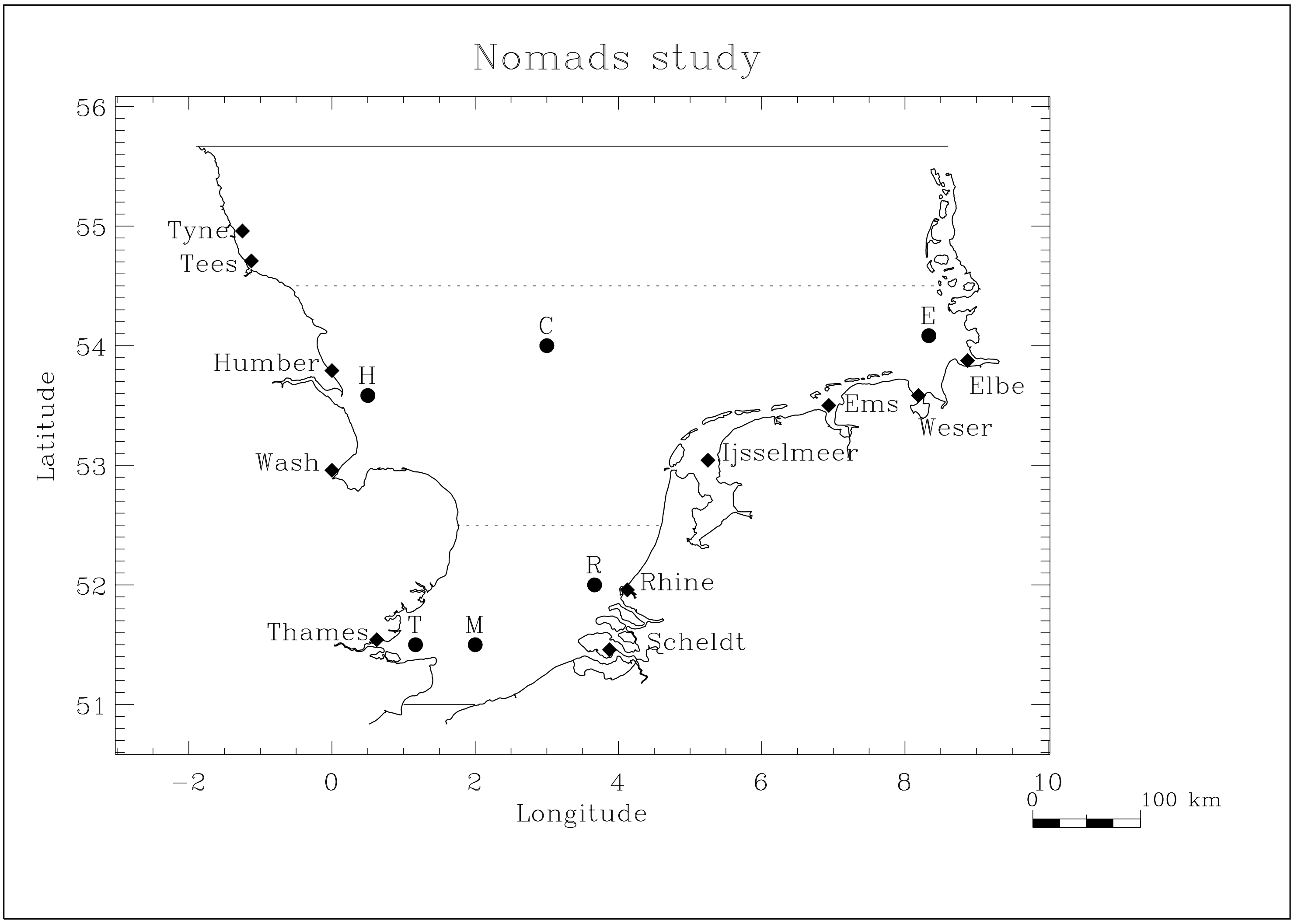
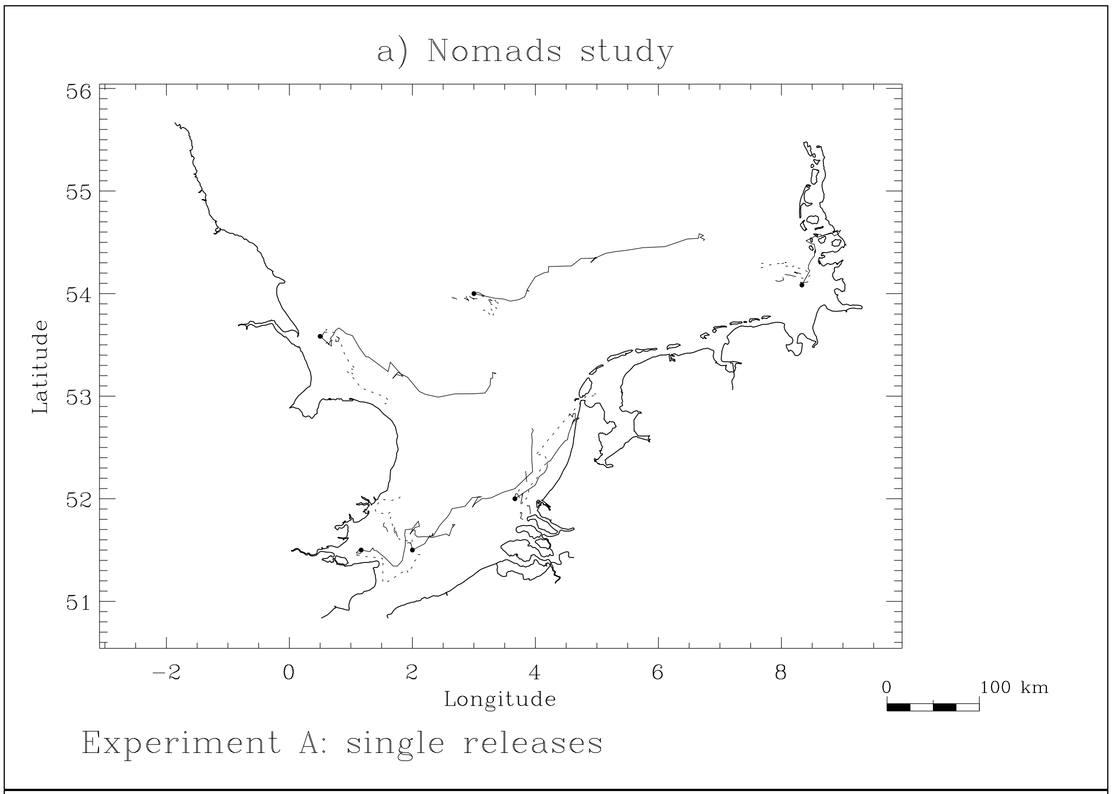


Figure 2.6.1: North Sea area used in the NOMADS study. The two solid lines represent the open sea boundaries, the two dotted lines the transects used in the NOMADS experiment 4. The positions of the release stations are indicated by the solid circles, the locations of the river discharge points by diamonds.



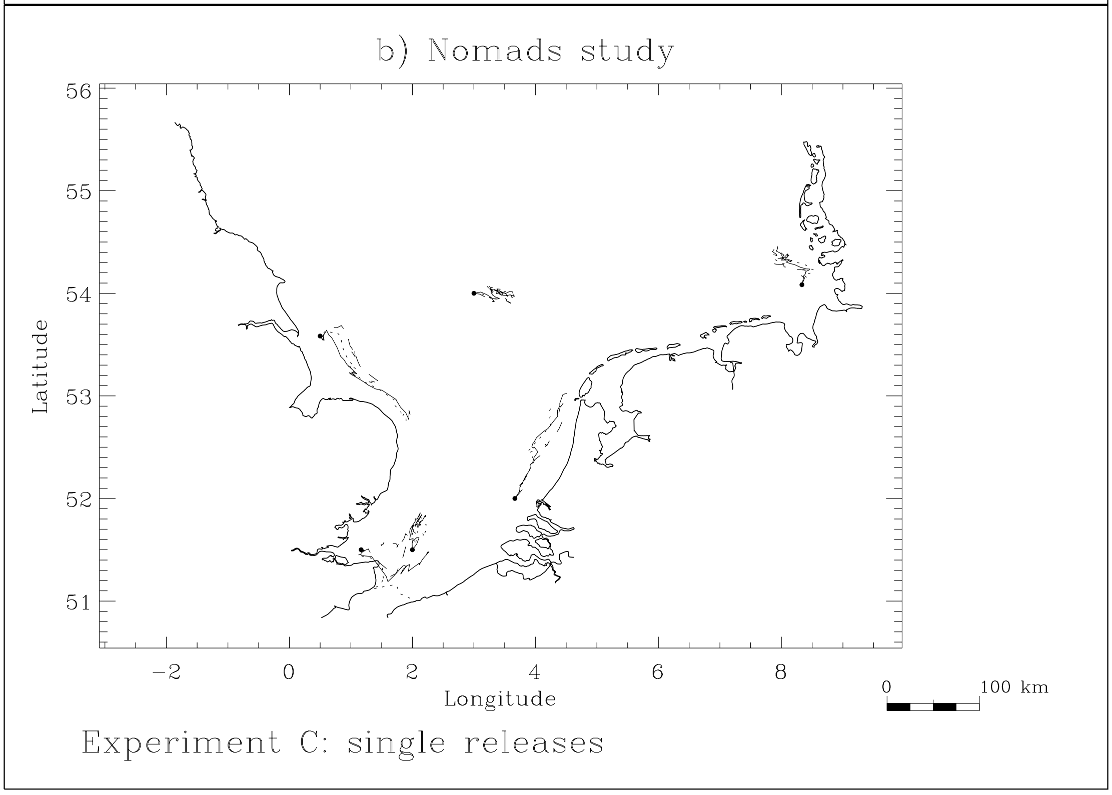
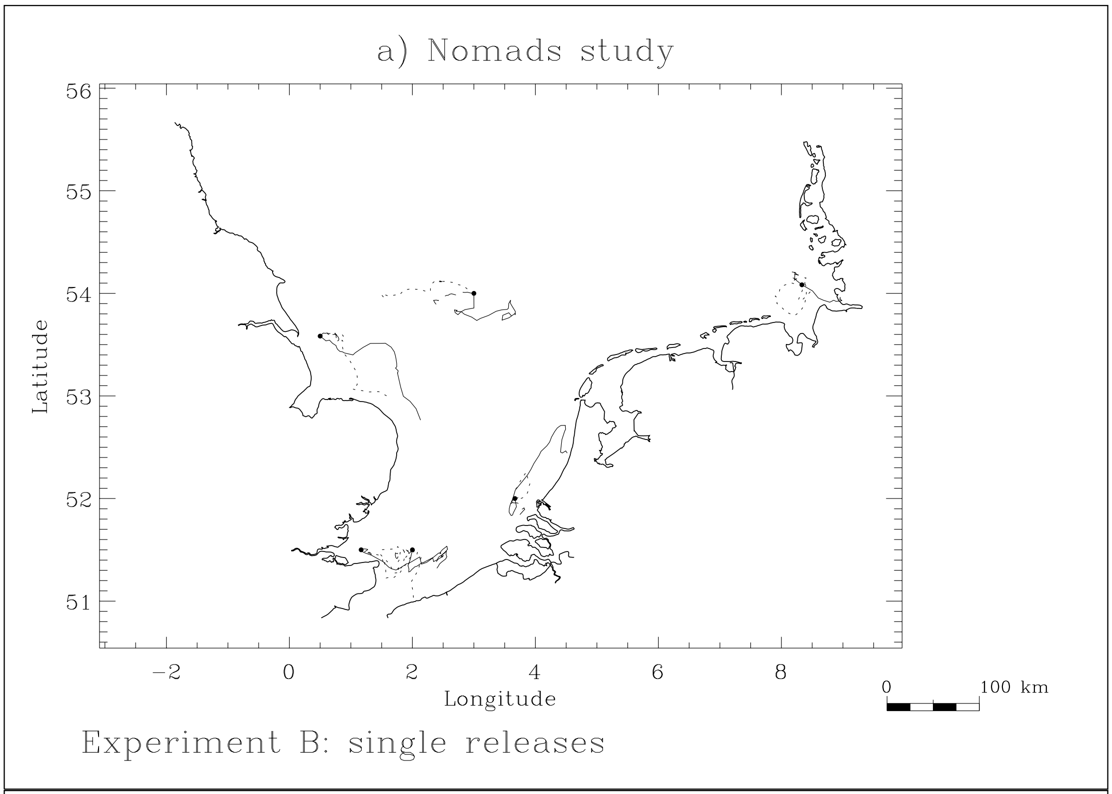


Figure 2.6.2: Particle trajectories for the single releases in March for the run without diffusion (a) and with diffusion (b). Surface, mid-depth and bottom particles are represented by respectively solid, dotted and dashed curves.



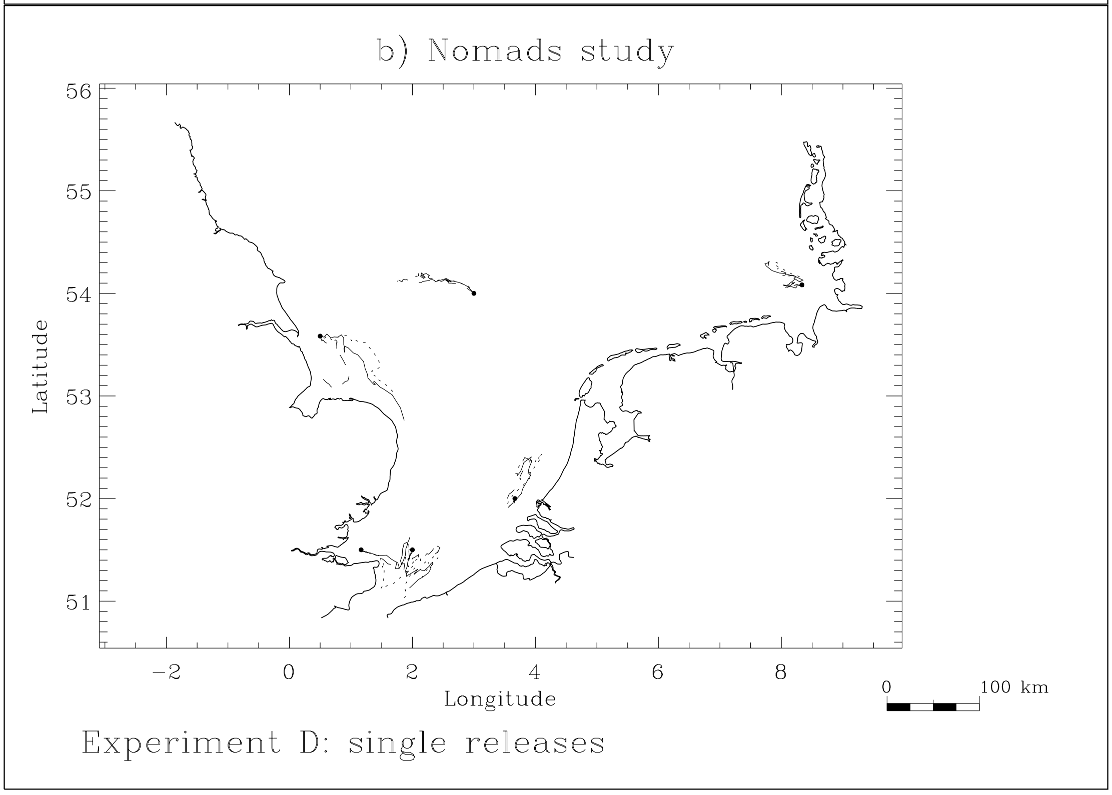
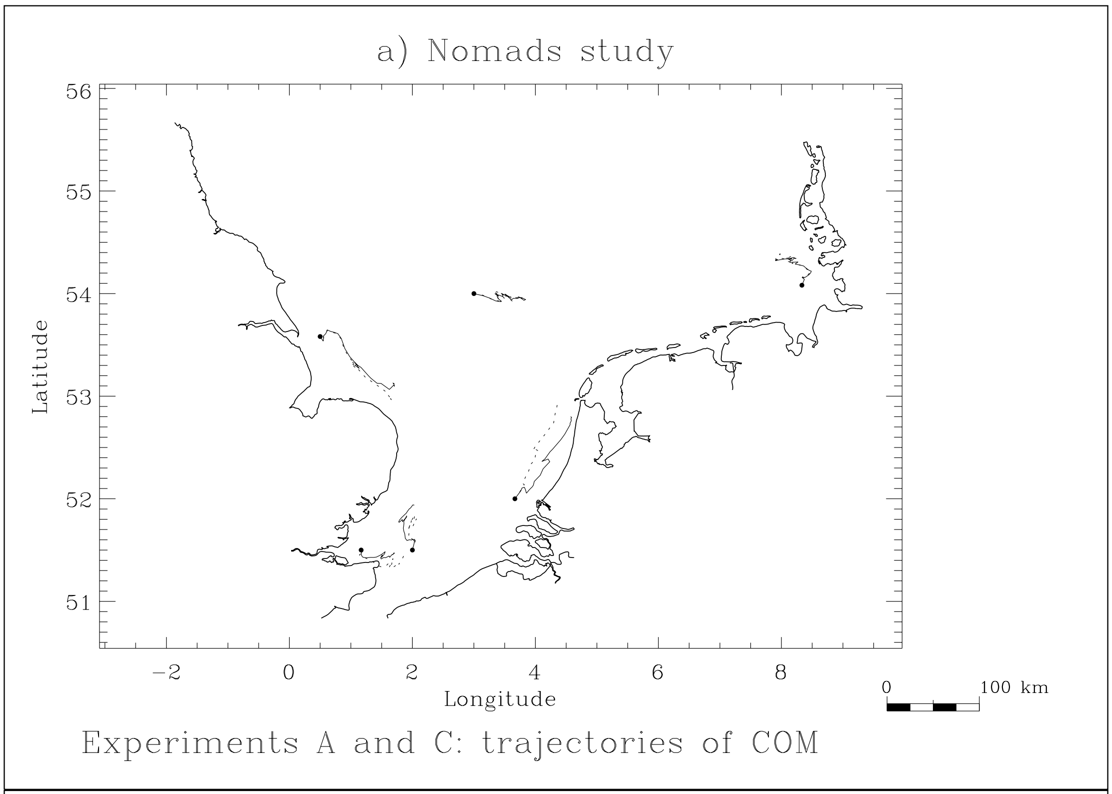


Figure 2.6.3: As Figure 2.6.2 now for the single releases in August.



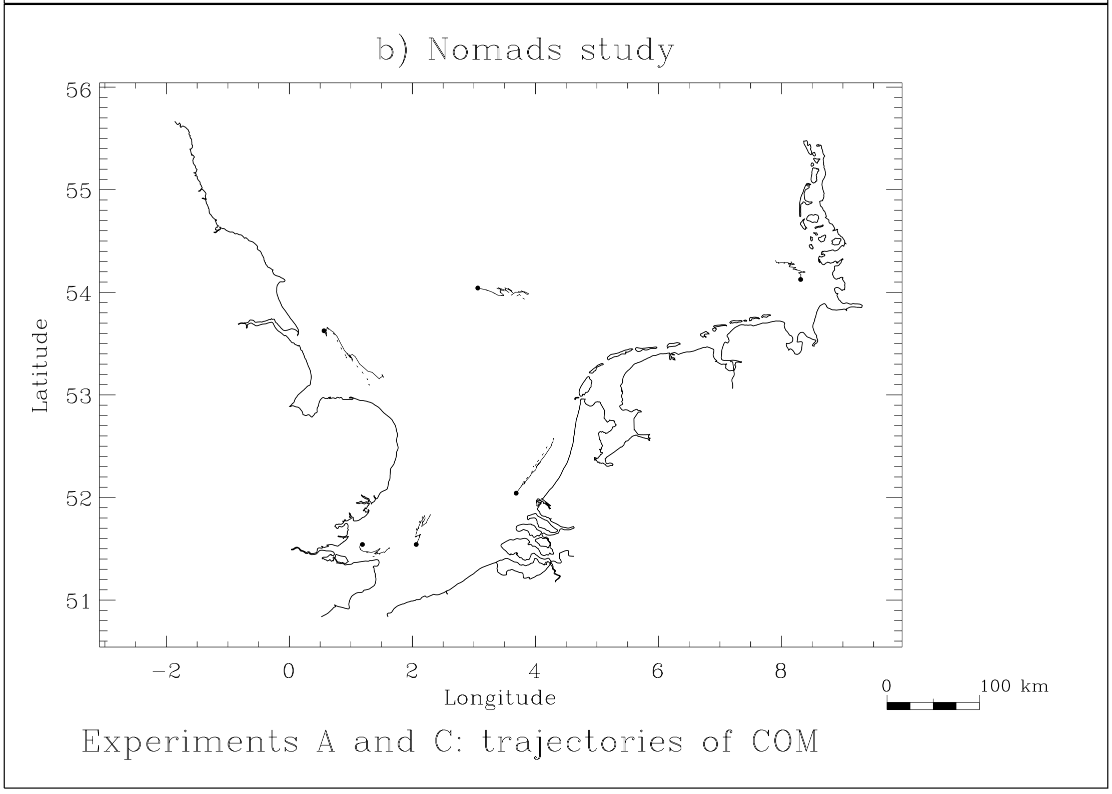


Figure 2.6.4: Trajectories of the centre of mass for the uniform releases in March according to the Lagrangian (a) and Eulerian (b) model: run without diffusion (experiment A, solid curves) and with diffusion (experiment C, dotted curves).

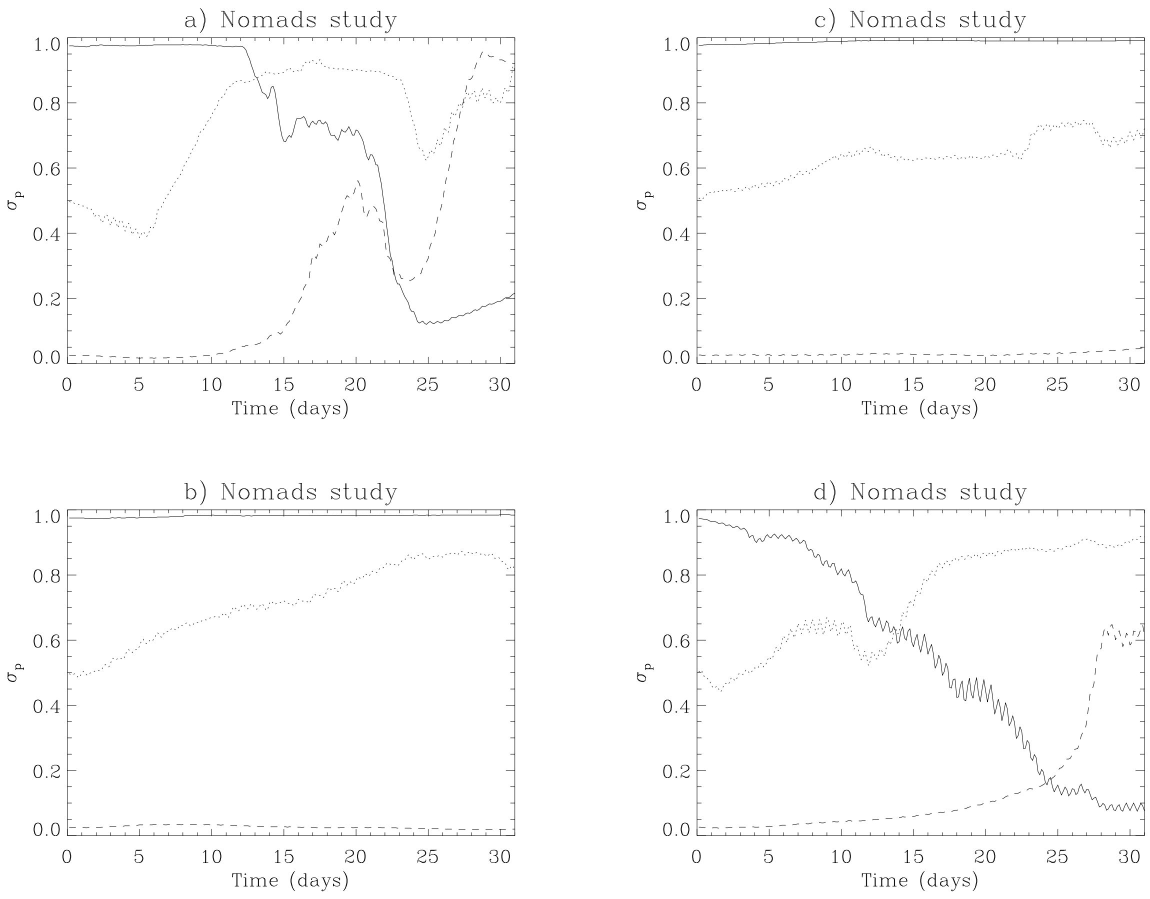
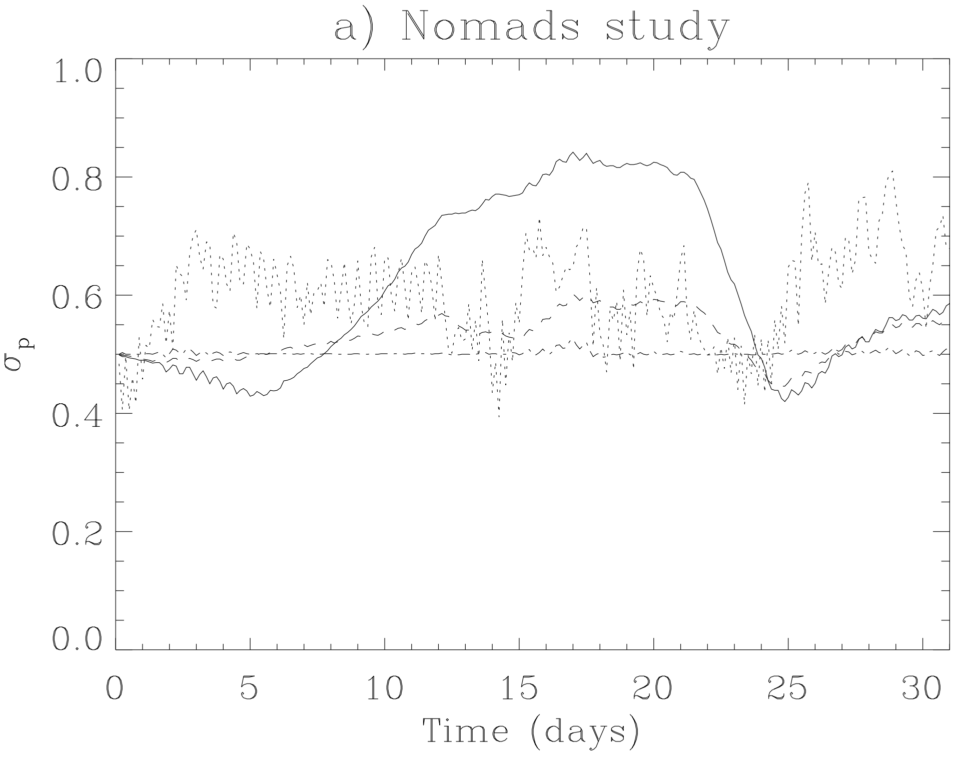


Figure 2.6.5: Time series (days) of the vertical coordinate σ_p for the single releases and the runs without diffusion (experiments A and B): March-Rhine (a), August-Central North Sea (b), March-Humber (c), August-Elbe (d). The surface, mid-depth and bottom particle are represented by respectively the solid, dotted and dashed curves.



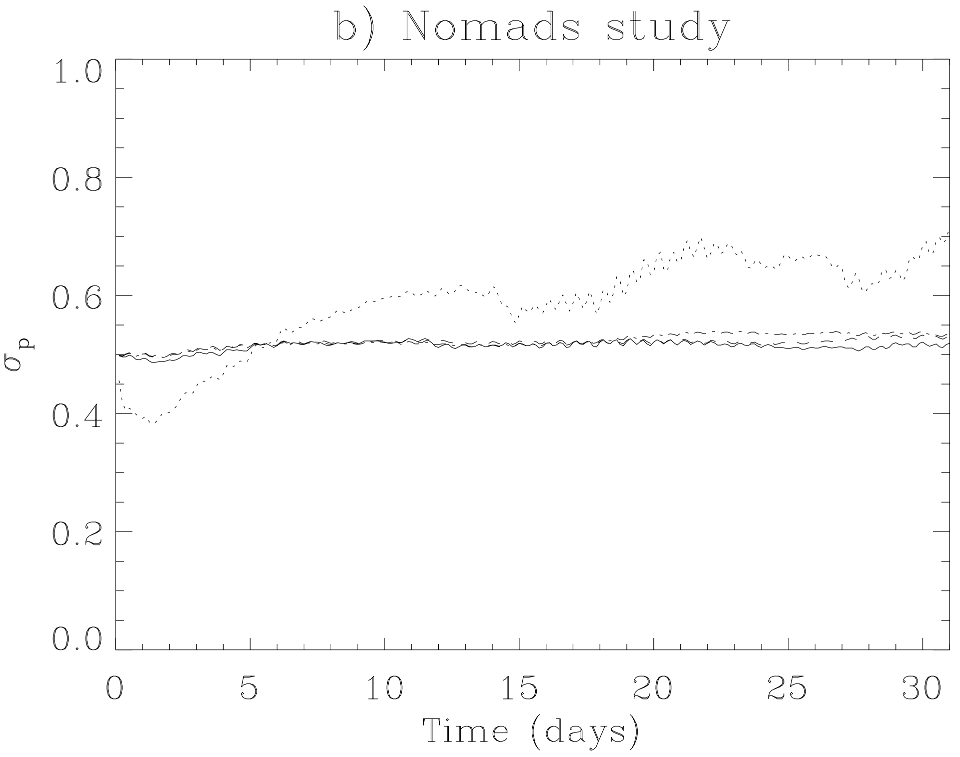
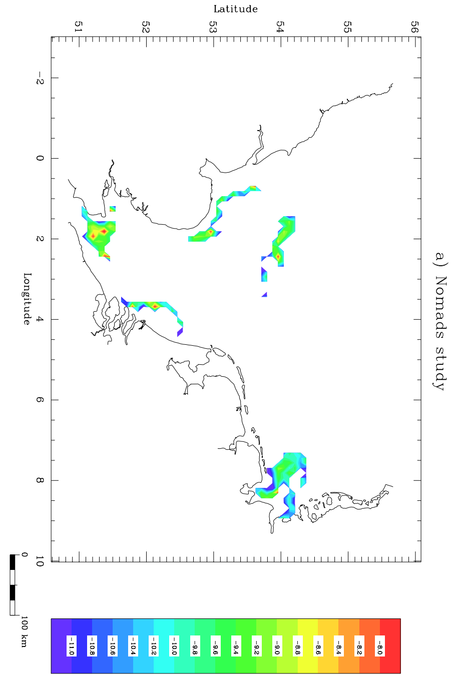
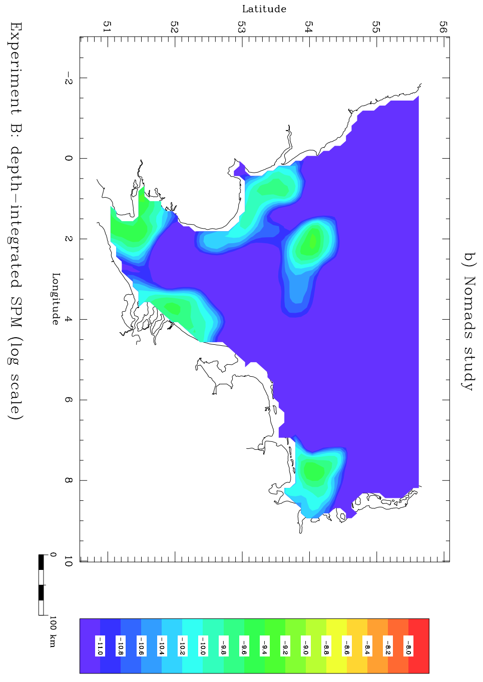
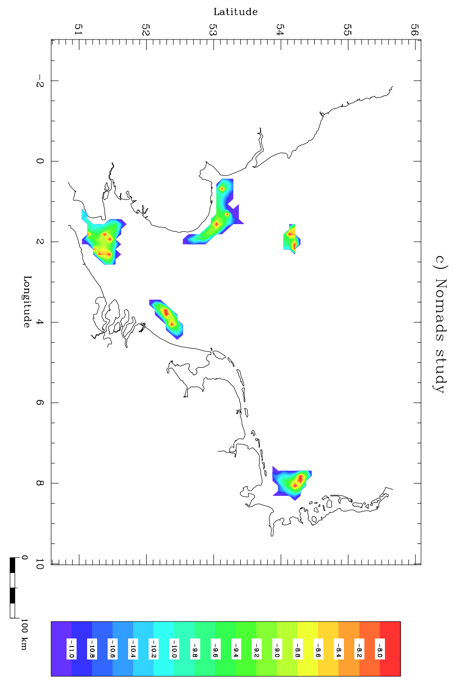


Figure 2.6.6: Time series (days) of the vertical coordinate σ_p of the centre of mass for the Rhine release in March (a) and the Thames release in August (b): Lagrangian without diffusion (solid), Lagrangian with diffusion (dots), Eulerian without diffusion (dashes), Eulerian with diffusion (dash-dots).







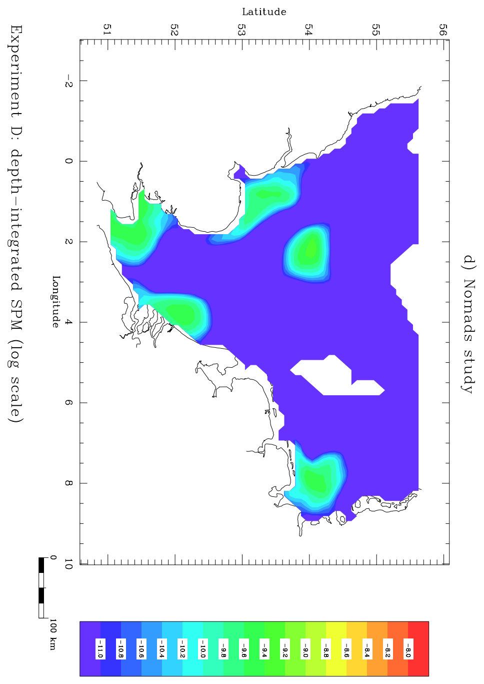


Figure 2.6.7: Surface distributions of the depth-integrated suspended matter (uniform releases) for the August runs: Lagrangian without diffusion (a), Eulerian without diffusion (b), Lagrangian with diffusion (c), Eulerian with diffusion (d).

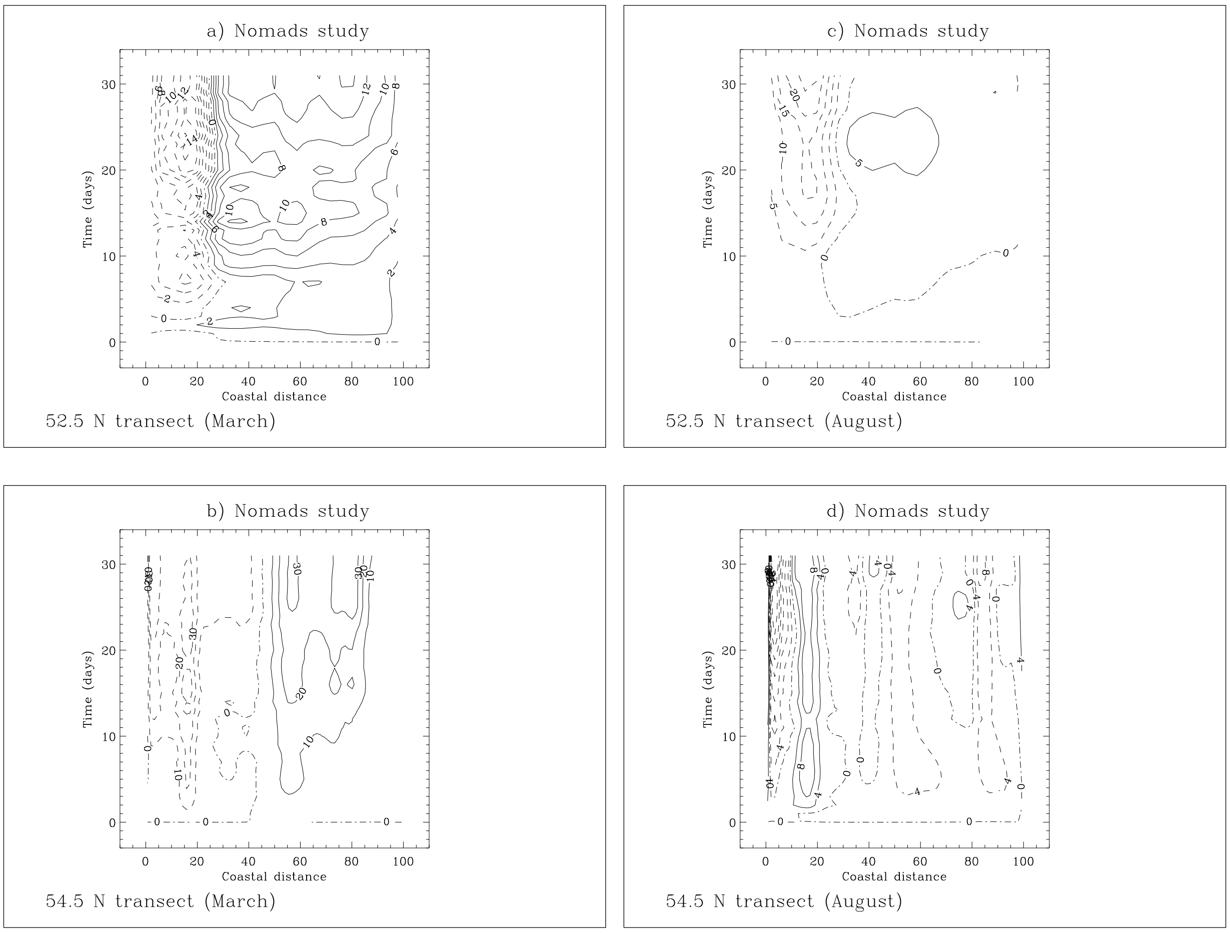


Figure 2.6.8: Values of the accumulated volume $\overline{V}_{acc}$ (in units of 10^5 m^2) as a function of coastal distance and time (days): southern transect in March (a), northern transect in March (b), southern transect in August (c), northern transect in August (d). Coastal distance is measured from the East coast of England and is given in %.

Chapter 7

North Sea Case Study

7.1 Introduction

A series of simulations has been performed testing the ability of COHERENS to simulate realistic seasonal patterns in a typical shelf sea and showing how the model is validated with observational data. The area, selected for this study, is the part of the European Continental Shelf containing the Channel, the southern and central parts of the North Sea. The simulations cover a full yearly cycle for the year 1989. Both the physical (currents, temperature, salinity), as the biological and sediment modules are activated in the model runs. Model results are compared with the observational data of the NERC (National Environment Research Council, UK) 1988–1989 North Sea Project.

The second aim of this study is to show an example how the model can be initialised for a realistic application with different kinds of data inputs (surface, open boundary input, initial conditions) and to illustrate how it can be set up for operational purposes.

A detailed description of model step is given in Section II-7.2. Model results are discussed and compared with the North Sea Project data in Section II-7.3.

7.2 Model setup

The complete model setup is described below. Model setup parameters are listed in Table 2.7.1.

7.2.1 Model grid

- The model equations are solved on a spherical grid.
- The computational domain consists of the part of the Continental Shelf between 4^0W and 57^0N covering the Channel and the southern and central parts of the North Sea.
- The computational grid has a resolution of $4'$ in latitude and $6'$ in longitude corresponding to a grid spacing between 6 and 7.5 km.

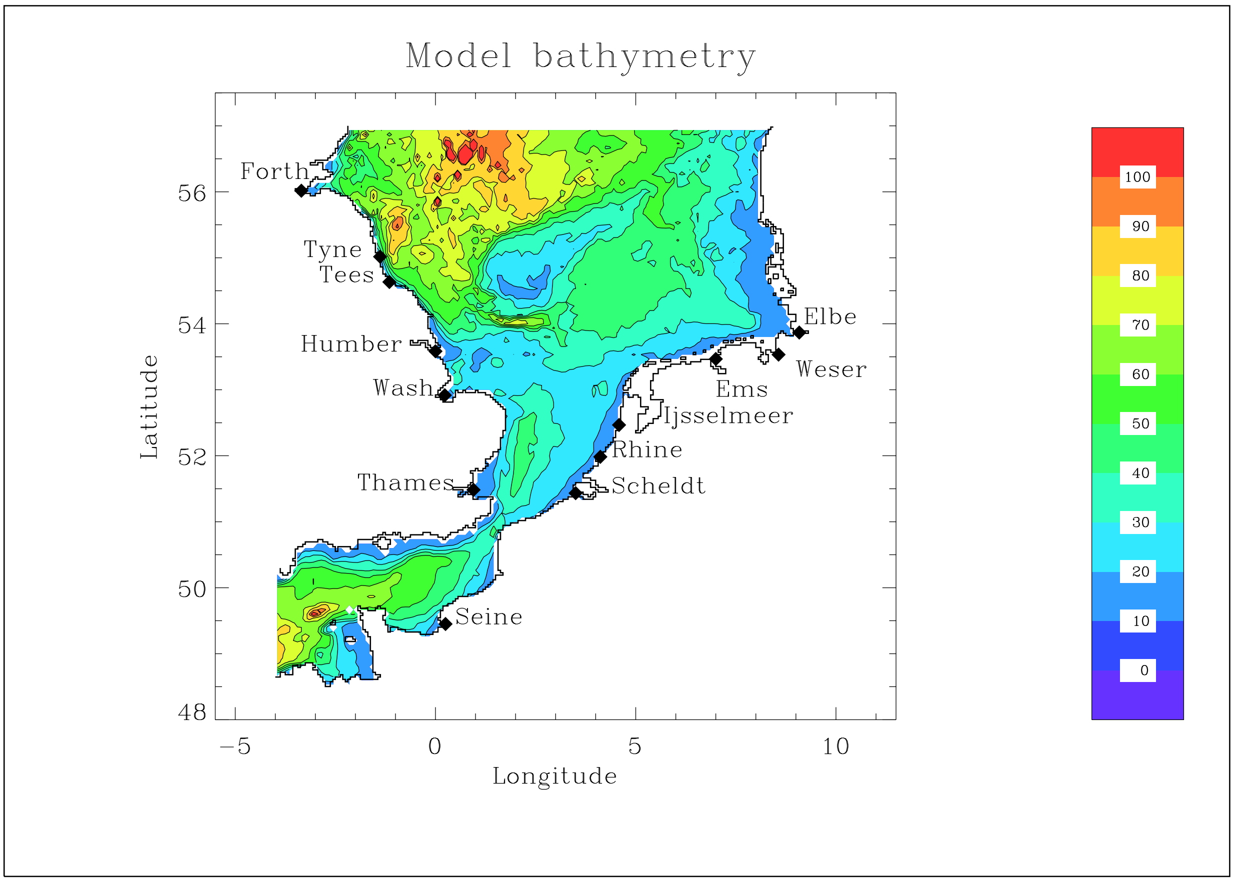


Figure 2.7.1: Bathymetry (m) of the simulated area in the North Sea.

- Open boundaries of the model area are the western and northern boundaries at 4^0W and 57^0N and 13 river boundaries. The bathymetry and the location of the river discharges are shown in Figure 2.7.1.

7.2.2 Numerics

- The time steps for the 2-D and 3-D mode are set to respectively 30 s and 600 s. The simulation starts the 1st of January 1989 0.00 GMT and ends at the 25th December 1989 0.00 GMT.

- The currents are advected with the upwind scheme. Tests using the more sophisticated TVD scheme, showed no appreciable differences whereas the CPU time for the circulation module increased by a factor two compared to the simpler upwind scheme. The default TVD scheme is used for all scalars (temperature, salinity, biological concentrations).

- Horizontal diffusion of momentum and scalar quantities is disabled (default setting of the program).

7.2.3 Physics

a) model parameters and schemes

- The density and the expansion coefficients β_S, β_T are computed using the general equation of state of seawater (see Section III-1.5).
- Both temperature and salinity are included in the simulation. Absorption of solar heat is computed with the theory described in Section III-1.4 using default values for all optical parameters.
- Turbulence effects are parameterised using the default scheme implemented in the program (see Section III-1.2.3 for details).

b) initial conditions

- The initial temperature data are derived by interpolation from the North Sea Project measurements for the area between 51^0N and 57^0N, and by data from a 3-D baroclinic model for the Continental Shelf (POL) outside that area. A depth-uniform field is taken which can be considered as realistic for a winter condition.
- The initial salinity field is taken from the previous model. Vertical stratification by salinity was negligibly small so that a depth-uniform field was assumed initially.
- The 3-D and depth-integrated currents and the sea surface elevation are initialised by first running the program for a spin-up period of three days starting with zero values for currents and sea surface elevation, and open boundary forcing for the 2-D mode (see below), but without temperature, salinity, biology and sediments.

c) open sea boundaries

- The depth-integrated current ($\overline{U}$ or $\overline{V}$) is determined using the method of characteristics described in Section III-1.6.3a and input data for the depth-integrated current and sea surface elevation in harmonic form. The procedure is summarised below.

  At the western boundary, the depth-integrated current $\overline{U}$ and the sea surface elevation ζ are prescribed using expansions of the form

$$\overline{U}(\phi,t) = \overline{U}_0(\phi) + \sum_{n=1}^{N_T} \tilde{F}_n \overline{U}_n(\phi)\cos(\omega_n t - \varphi_{n0} + \tilde{\varphi}_n - \varphi_{nu}(\phi)) \tag{2.7.1}$$

$$\zeta(\phi,t) = \zeta_0(\phi) + \sum_{n=1}^{N_T} \tilde{F}_n \zeta_n(\phi)\cos(\omega_n t - \varphi_{n0} + \tilde{\varphi}_n - \varphi_{ns}(\phi)) \tag{2.7.2}$$

where ϕ is the latitude of a point at the western boundary[1]. The other parameters have the following meaning:

- N_T is the number of tidal components, given by the following constituents: Q_1, O_1, K_1, μ_2, M_2, N_2, L_2, S_2, M_4.
- $(\overline{U}_0,\ \zeta_0)$ represent the residual component supplied at hourly intervals.
- $(\overline{U}_n,\ \zeta_n)$ are the amplitudes and $(\varphi_{nu},\ \varphi_{ns})$ the phases of the tidal forcing provided at each open boundary node.
- $(\tilde{F}_n,\ \tilde{\varphi}_n)$ are the spatially uniform nodal factors which vary slowly in time and are provided at monthly intervals.
- φ_{n0} are the initial phases.

All parameters are obtained from a surge model for the Continental Shelf (POL). The expansions (2.7.1), (2.7.2) are then substituted into an harmonic expansion of the form (see equation (3.4.170)):

$$F_{har}(\phi,t)=\frac{\overline{U}}{2c}+\frac{1}{2}\zeta=A_0(\phi)+\sum_{n=1}^{N_T}A_n(\phi)\cos(\omega_n t-\varphi_{n0}+\tilde{\varphi}_n-\varphi_n(\phi)) \tag{2.7.3}$$

where $c=(gh)^{1/2}$ is the barotropic wave speed and h the mean water depth. The expansion coefficients A_0 and $(A_n,\ \varphi_n)$ are then obtained from

$$A_0=\frac{\overline{U}_0}{2c}+\frac{1}{2}\zeta_0 \tag{2.7.4}$$

$$A_n\cos\varphi_n=\frac{1}{2}\tilde{F}_n\Big(\frac{\overline{U}_n}{c}\cos\varphi_{nu}+\zeta_n\cos\varphi_{ns}\Big) \tag{2.7.5}$$

$$A_n\sin\varphi_n=\frac{1}{2}\tilde{F}_n\Big(\frac{\overline{U}_n}{c}\sin\varphi_{nu}+\zeta_n\sin\varphi_{ns}\Big) \tag{2.7.6}$$

The parameters $(A_0,\ A_n)$ and $\varphi_n-\tilde{\varphi}_n$ are stored into respectively the setup arrays AMPOBU and PHAOBU (see Section IV-2.4). The depth-integrated current $\overline{U}$ is then computed by the program at each 2-D time step along the following steps:

1. $F_{har}(\phi,t)$ is calculated from (2.7.3) using the data stored into the two previous arrays.
2. The incoming Riemann invariant R_+ is determined using

$$R_+=\overline{U}+c\zeta=2cF_{har} \tag{2.7.7}$$

[1]An extra linear interpolation procedure was required in the actual setup of the program since the harmonic parameters were supplied at points which differred from the open boundary locations on the model grid.

3. The outgoing Riemann invariant $R_- = \overline{U} - c\zeta$ is obtained by solving (3.1.196) numerically using the discretised form (3.4.163).
4. The 2-D current is then given by

$$\overline{U} = \frac{1}{2}(R_+ + R_-) \tag{2.7.8}$$

A similar (but slightly different) procedure is taken at the northern open sea boundary. The depth-integrated current $\overline{V}$ and the surface elevation are prescribed using

$$\overline{V}(\lambda,t) = \overline{V}_0(\lambda) + \sum_{n=1}^{N_T} \tilde{G}_n \overline{V}_n(\lambda)\cos(\omega_n t - \varphi_{n0} + \tilde{\varphi}_n - \varphi_{nv}(\lambda)) \tag{2.7.9}$$

$$\zeta(\lambda,t) = \zeta_0(\lambda) + \sum_{n=1}^{N_T} \tilde{G}_n \zeta_n(\lambda)\cos(\omega_n t - \varphi_{n0} + \tilde{\varphi}_n - \varphi_{ns}(\lambda)) \tag{2.7.10}$$

where λ is the longitude of a point at the northern boundary[1] and the parameters have the same meaning as above. Substituting (2.7.9), (2.7.10) into the harmonic form $F_{har}(\lambda,t)$, defined by (see equation (3.4.170)),

$$F_{har}(\lambda,t) = -\frac{\overline{V}}{2c} + \frac{1}{2}\zeta = A_0(\lambda) + \sum_{n=1}^{N_T} A_n(\lambda)\cos(\omega_n t - \varphi_{n0} + \tilde{\varphi}_n - \varphi_n(\lambda)) \tag{2.7.11}$$

one obtains

$$A_0 = -\frac{\overline{V}_0}{2c} + \frac{1}{2}\zeta_0 \tag{2.7.12}$$

$$A_n \cos\varphi_n = \frac{1}{2}\tilde{G}_n\Big(-\frac{\overline{V}_n}{c}\cos\varphi_{nv} + \zeta_n\cos\varphi_{ns}\Big) \tag{2.7.13}$$

$$A_n \sin\varphi_n = \frac{1}{2}\tilde{G}_n\Big(-\frac{\overline{V}_n}{c}\sin\varphi_{nv} + \zeta_n\sin\varphi_{ns}\Big) \tag{2.7.14}$$

The harmonic parameters (A_0, A_n) and $\varphi_n - \tilde{\varphi}_n$ are stored into respectively the setup arrays AMPOBV and PHAOBV (see Section IV-2.4). These arrays allow the program to calculate $\overline{V}$ at each 2-D time step:

1. $F_{har}(\lambda,t)$ is calculated from (2.7.11) using the data stored into the two previous arrays.
2. The incoming Riemann invariant R_- is determined using

$$R_- = \overline{V} - c\zeta = -2cF_{har} \tag{2.7.15}$$

3. The outgoing Riemann invariant $R_+ = \overline{V} + c\zeta$ is obtained by solving (3.1.197) numerically using the discretised form (3.4.164).

4. The 2-D current is then given by

$$\overline{V} = \frac{1}{2}(R_+ + R_-) \tag{2.7.16}$$

- The horizontal current is determined at open sea boundaries using a zero gradient condition for the velocity deviation u' or v' as given by (3.1.208) or (3.1.209).

- A zero normal gradient condition is taken for temperature giving a zero net transport into the computational domain at all open sea boundaries (see Section III-1.6.3b).

- The open boundary condition for salinity has a mixed form. A zero gradient condition is taken at grid points below a fixed surface layer of 6 m depth. Salinity is transported inside the domain during inflow by prescribing a (depth-uniform) external profile within the surface layer, taking the form of an annually varying sine wave

$$S_{ext}(\lambda,\phi,t) = S_{0,ext}(\lambda,\phi) + S_{a,ext}(\lambda,\phi)\cos(\omega_a t_d - \varphi_{ext}(\lambda,\phi)) \tag{2.7.17}$$

  where $S_{0,ext}$, $S_{a,ext}$ and φ_{ext} are derived from the previously mentioned 3-D baroclinic model, $\omega_a = 2\pi/365$ rad/day is the annual frequency and t_d the time in days since the beginning of the year. Values of S_{ext} are stored at all open boundary nodes and daily intervals into the setup array SVP (see Section IV-2.5).

d) river boundaries

Input is considered from the following rivers, situated along the British and continental coasts: Thames, Wash, Humber, Tees, Tyne, Forth, Seine, Scheldt, Rhine, IJsselmeer, Ems, Weser, Elbe (see Figure 2.7.1).

- In analogy with the open sea boundaries the depth-integrated current $\overline{U}$ or $\overline{V}$ is computed using the method of characteristics now applied in a different form.

  1. A river discharge Q_d (in m^3/s) is supplied from an external data file (POL) at daily intervals and for each river location.
  2. In analogy with (2.7.3) or (2.7.11) an "harmonic" function F_{har}, now without tidal part, is defined by

  $$F_{har} = Q_d/(cW) \tag{2.7.18}$$

     where c is the barotropic wave speed and W the width of the river mouth set equal to one horizontal grid spacing in the X- or Y-direction depending on the orientation of the river boundary. The values of F_{har} (which must always be positive for an outflow condition) are stored into the residual part of the arrays AMPOBU and AMPOBV.
  3. The program calculates the outgoing Riemann variable $R_\mp$ by solving (3.1.196) or (3.1.197) using the discretised forms (3.4.163) or (3.4.164).

4. The incoming Riemann variable is determined using (see equation 3.4.173))

$$R_{\pm} = \pm 2cF_{har} - R_{\mp} \tag{2.7.19}$$

   where the upper sign applies at river boundaries located along "western" or "southern" cell faces and the lower sign at boundaries located along "eastern" or "northern" cell faces.

5. The depth-integrated current is then obtained by taking the average of R_+ and R_-.

- The horizontal currents u or v are determined by assuming a uniform discharge at all depths. This means that the velocity deviations u'_0 or v'_0 (see equation (3.1.210) are set to zero.

- Temperature data are provided for some (but not all) rivers at irregular times (POL). These values are stored into the setup array TVP and used to transport temperature inside the domain assuming a depth-uniform value. A zero gradient condition is considered for the rivers for which no data were available (Wash, Humber, Seine, Scheldt).

- Salinity is provided in the form of a two-layer stratification with a fresh water value of 10 PSU (stored into the setup array SVP) within a fixed surface layer of 6 m depth and a zero gradient condition below this fresh water layer.

e) surface forcing

- The surface fluxes of momentum, heat and salinity are determined from the formulations described in Section III-1.6.1. The surface drag and exchange coefficients are obtained using the theory described in Section III-1.6.4 and the empirical relations (3.1.221).

- The following surface data were supplied to the model:

  - Wind speed and direction, atmospheric pressure, air temperature and relative humidity are given at 3-hourly intervals from a meteorological model (UK Meteorological Office).
  - Cloud coverage is derived from satellite images with a daily value taken to be uniform over the whole domain.
  - The evaporation and precipitation rates, used in the expression (3.1.179) for the surface salinity flux, are supplied in the form of monthly mean values.

Table 2.7.1: Model setup for the North Sea case study.

parameter	value	purpose
param.inc		
NC	140	number of grid cells in the X-direction (longitude)
NR	127	number of grid cells in the Y-direction (latitude)
NZ	20	number of grid cells in the vertical
NOBU	29	number of open boundary points in the X-direction (22 open sea and 7 river boundaries)
NOBV	111	number of open boundary points in the Y-direction (105 open sea and 6 river boundaries)
NVPROF	140	number of profiles at open boundaries (a different profile is taken at each boundary node)
NCON	9	number of tidal constituents
DEFCON		
IGTRH	1	spherical grid
IOPTB	1	biology module switched on
IOPTS	1	sediment module switched on
IADVC	1	upwind scheme for the advection of momentum
IOPTSA	1	salinity module switched on
IOPTHE	2	temperature and optical modules switched on
IOPTD	2	thermal expansion coefficients calculated using the general equation of state of seawater (see Section III-1.5).
IOPTM	3	program reads meteorological input at all grid points and selected time intervals, heat fluxes calculated by the program
IDRAG	3	surface drag coefficient evaluated using the empirical formulation of Geernaert et at. (1986)
ITDIF	1	surface exchange and drag coefficients evaluated using the Monin-Obukhov formulation (see Section III-1.6.4)
IOUTF	1	time-averaged output enabled
IBDATE		start date for specific run (MMDDHHMM)
IEDATE		end date for specific run (MMDDHHMM)
IBYEAR	1989	start year
IEYEAR	1989	end year
DELT	30	time step for the external (2-D) mode (s)
IC3D	20	time step (10 min) for the external (3-D) mode divided by DELT
ICMET	360	time interval (3 hours) for meteorological input divided by DELT

Table 2.7.1: continued.

parameter	value	purpose
DEFCON		
IC2BC	120	time interval (1 hour), divided by DELT, for (2-D) open boundary input (residuals, amplitudes, phases in the harmonic function F_{har})
ICHBC	2880	time interval (1 day), divided by DELT, for (3-D) physical open boundary input (currents, temperature, salinity)
ICBBC	2880	time interval (1 day), divided by DELT, for open boundary input in the biological and sediment modules
GRDFORM	'U'	file with grid arrays written in binary format
ICRFORM	'U'	files with initial/final conditions written in binary format
BCSFORM	'U'	files with open boundary data written in binary format
METFORM	'U'	meteorological data file written in binary format
OUTTIT		first 6 characters in the file names of user-defined output data files
SREF	35.0	reference salinity (PSU)
TREF	12.0	reference temperature (^{0}C)
CDZ0UN	0.0035	uniform roughness length (m) used for the evaluation of the bottom stress
DEFGRID		
GX0(NC+1)		X-coordinates (longitude) of the cell corners (between 4^0W and 10^0E)
GY0(NR+1)		Y-coordinates (latitude) of the cell corners (between $48^0 32'$N and 57^0N)
DEP(NR,NC)		array of mean water depths (m)
NPIX(NR,NC+1)		cell face pointer array at U-nodes (0 at solid boundaries, 1 at internal wet faces, 2 at open sea and 3 at river boundary faces)
NPIY(NR+1,NC)		cell face pointer array at V-nodes (0 at solid boundaries, 1 at internal wet faces, 2 at open sea and 3 at river boundary faces)
(IOBU,JOBU)(1:NOBU)		cell face indices of the open boundary points at U-nodes
(IOBV,JOBV)(1:NOBV)		cell face indices of the open boundary points at V-nodes

Table 2.7.1: continued.

parameter	value	purpose
DEFOB2D		
ITYPOBU(1:NOBU)	1	type of open boundary condition (2-D mode) at the western open sea boundary
	4	type of open boundary condition (2-D mode) at the western/eastern river boundaries
ITYPOBV(1:NOBV)	1	type of open boundary condition (2-D mode) at the northern open sea boundary
	4	type of open boundary condition (2-D mode) at the southern/northern river boundaries
SIGMA(NCON)		tidal forcing frequencies (rad/s)
DEFOB3D		
IVPOBU(1:NOBU)		profile number at U-nodal (western/eastern) open boundaries (profiles 1 to 22 for the western open sea boundary points, profiles 23 to 29 for the western/eastern river boundaries)
IVPOBV(1:NOBV)		profile number at V-nodal (southern/northern) open boundaries (profiles 30 to 134 for the northern open sea boundary points, profiles 135 to 140 for the southern/northern river boundaries)
DEFICS		
S(NZ,NR,NC)		initial salinity field (PSU) from an external data file
T(NZ,NR,NC)		initial temperature field (^{0}C) from an external data file
P2B(NZ,NR,NC)	2.0	uniform initial concentration of microplankton carbon (mmol C m^{-3})
P2C(NZ,NR,NC)	1.0	uniform initial concentration of detrital carbon (mmol C m^{-3})
P2M(NZ,NR,NC)	0.1	uniform initial concentration of detrital nitrogen (mmol N m^{-3})
P2N(NZ,NR,NC)	0.4	uniform initial concentration of microplankton nitrogen (mmol N m^{-3})
P2NHS(NZ,NR,NC)	1.0	uniform initial concentration of ammonium (mmol N m^{-3})
P2O(NZ,NR,NC)	300.0	uniform initial concentration of oxygen (mmol O m^{-3})
P2NOS(NZ,NR,NC)		initial concentration of nitrate (mmol N m^{-3}) from an external data file
GRAZING(NR,NC,12)		horizontal array of monthly grazing pressures (day^{-1}) from an external data file
SEDC1(NZ,NR,NC)		initial concentration of iSPM (g m^{-3}) from an external data file

Table 2.7.1: continued.

parameter	value	purpose
WRFCDAT		
This routine is empty since all meteorological data are read by the main program directly from an external data file. This is performed in subroutine WRFCDAT2, located in a modified *metin.f* file (**PROJECT** directory). The procedure avoids creating large input data files for the main program by the preprocessor.		
WRFCDAT2 (file **PROJECT**/*metin.f*)		
(WINDU2,WINDV2,P2,SAT2,HUM2)(NR,NC)		wind components (m/s), atmospheric pressure (N/m^2), air temperature (^{0}C), relative humidity (0–1) obtained from external data files at 3-hourly intervals. The external data are linearly interpolated on the model grid.
CLOUD2(NR,NC)		spatially uniform array of values for the cloud coverage (0–1), supplied at daily intervals from an external data file
EVAPR(NR,NC)		spatially uniform array of values for the evaporation minus precipitation rate, supplied from an external data file at monthly intervals
WROBDAT		
AMPOBU(1:NOBU,0:NCON)		residuals and amplitudes (m) for the harmonic forcing at western/eastern boundaries (with the nodal factors $\tilde{F}_n$ interpolated at hourly intervals) • AMPOBU(1:22,0): residual component A_0 of the open sea boundary forcing as given by (2.7.4) • AMPOBU(1:22,1:NCON): amplitudes A_n of the open sea boundary forcing as obtained from (2.7.5)–(2.7.6) • AMPOBU(23:29,0): residual component of the forcing at river boundaries as given by (2.7.18) with Q_d from an external data file • AMPOBU(23:29,1:NCON): tidal components of the forcing at river boundaries which are set to 0

Table 2.7.1: continued.

parameter	value	purpose
WROBDAT		
PHAOBU(1:NOBU,1:NCON)		phases (rad) of the harmonic forcing at western/eastern boundaries • PHAOBU(1:22,1:NCON): values of the phases $\varphi_n - \tilde{\varphi}_n$ (rad) of the open sea boundary forcing with φ_n obtained from (2.7.5)–(2.7.6) and the nodal factors $\tilde{\varphi}_n$ from an external data file interpolated at hourly intervals • PHAOBU(23:29,1:NCON): phases of the forcing at river boundaries which are set to zero
AMPOBV(1:NOBV,0:NCON)		residuals and amplitudes (m) for the harmonic forcing at southern/northern boundaries (with the nodal factors $\tilde{F}_n$ interpolated at hourly intervals) • AMPOBV(1:105,0): residual component A_0 of the open sea boundary forcing as given by (2.7.12) • AMPOBV(1:105,1:NCON): amplitudes A_n of the open sea boundary forcing as obtained from (2.7.13)–(2.7.14) • AMPOBV(106:111,0): residual component of the forcing at river boundaries as given by (2.7.18) with Q_d from an external data file • AMPOBV(106:111,1:NCON): tidal components of the forcing at river boundaries which are set to 0
PHAOBV(1:NOBV,1:NCON)		phases (rad) of the harmonic forcing at southern/northern boundaries • PHAOBV(1:105,1:NCON): values of the phases $\varphi_n - \tilde{\varphi}_n$ (rad) of the open sea boundary forcing with φ_n obtained from (2.7.13)–(2.7.14) and the nodal factors $\tilde{\varphi}_n$ from an external data file interpolated at hourly intervals • PHAOBU(106:111,1:NCON): phases of the forcing at river boundaries which are set to zero

Table 2.7.1: continued.

parameter	value	purpose
WROBDAT		
SVP(1:NZ,1:NVPROF)		open boundary profiles for salinity (at daily intervals) • SVP(1:NZ,1:22): vertical profiles at the western open sea boundary points using a 2-layer form with the surface values obtained from (2.7.17) (default gradient condition below 6 m depth) • SVP(1:NZ,23:29): 2-layer profile at western/eastern river boundaries (default zero gradient condition below 6 m depth) • SVP(1:NZ,30:134): vertical profiles at the northern open sea boundary points using a 2-layer form with the surface value obtained from (2.7.17) (default zero gradient condition below 6 m depth) • SVP(1:NZ,135:140): 2-layer profile at southern or northern river boundaries (default zero gradient condition below 6 m depth)
TVP(1:NZ,1:NVPROF)		open boundary profiles for temperature (at daily intervals) • TVP(1:NZ,1:22): default zero gradient condition at the western open sea boundary • TVP(1:NZ,23:29): vertically uniform profiles or zero gradient condition at western/eastern river boundaries from an external data file • TVP(1:NZ,30:134): default zero gradient condition at the northern open sea boundary • TVP(1:NZ,135:140): vertically uniform profiles or zero gradient condition at southern/northern river boundaries from an external data file

Table 2.7.1: continued.

parameter	value	purpose
WROBDAT		
UVP(1:NZ,1:NVPROF)		open boundary profiles for the horizontal current (deviation) • UVP(1:NZ,1:22): default zero gradient condition along the western open sea boundary • UVP(1:NZ,23:29): zero velocity deviation (i.e. depth-uniform discharge) at the western/eastern river boundaries • UVP(1:NZ,30:134): default zero gradient condition along the northern open sea boundary • UVP(1:NZ,135:140): zero velocity deviation (i.e. depth-uniform discharge) at the southern/northern river boundaries
P2NOSVP(1:NZ,1:NVPROF)		open boundary profiles for nitrate (at daily intervals) • P2NOSVP(1,1:22): nitrate input at the bottom grid cell (western open sea boundary) as obtained from (2.7.20)–(2.7.21) • P2NOSVP(2:NZ,1:22): default zero gradient condition (western open sea boundary) • P2NOSVP(1:NZ,23:29): vertically uniform profiles at western/eastern river boundaries from an external data file • P2NOSVP(1,30:134): nitrate input at the bottom grid cell (northern open sea boundary) as obtained from (2.7.20), (2.7.22) • P2NOSVP(2:NZ,30:134): default zero gradient condition (northern open sea boundary) • P2NOSVP(1:NZ,135:140): vertically uniform profiles at southern/northern river boundaries from an external data file

7.2.4 Biology

a) model parameters

Default values are taken for all biological model parameters. See Chapter III-2 for further discussion and Table 4.2.3 for a listing of all default values.

b) initial conditions

The model is initialised with depth-uniform and horizontally homogeneous concentrations of microplankton and detrital carbon and nitrogen, dissolved oxygen and ammonium. These values are representative of low biomass mean winter conditions throughout the North Sea and equivalent to the initial values given in the **csbio** test case. Nitrate concentrations are initialised with a grid of depth-uniform concentrations derived from the North Sea Project data set and interpolated throughout the region to the model boundaries.

c) grazing pressure

Zooplankton grazing is implemented as monthly varying grazing pressures calculated for various geographical provinces throughout the model region. These forcing data are derived from North Sea Project zooplankton data interpolated throughout the region.

d) open sea boundaries

Zero gradient open boundary conditions are specified for microplankton and detrital carbon and nitrogen, dissolved oxygen and ammonium. For dissolved nitrate a no-flux condition is implemented in the upper water column although at depth a flux across the boundary was permitted. The nitrate flux is calculated by allowing the deepest model layer to relax towards the concentration prescribed in waters immediately adjacent to the model domain. Deep water nitrate concentrations along the model boundaries are estimated from smoothed curves fitted to seasonal cycles, with parameters varying with longitude, derived from the NOWESP data set. These curves take the form of an harmonic expansion with a residual, annual and semi-annual component

$$ {}^{NO}S(\lambda) = {}^{NO}S_a(\lambda)\cos(\omega_a(t_d - t_1)) + \frac{1}{3}{}^{NO}S_a(\lambda)\cos(2\omega_a(t_d - t_2)) \tag{2.7.20} $$

where

$$ {}^{NO}S_0 = 5.7\,,\ {}^{NO}S_a = 3\,,\ t_1 = 35\,,\ t_2 = 50 \tag{2.7.21} $$

at the western open sea boundary of the domain, and

$$ \begin{aligned} {}^{NO}S_0(\lambda) &= 7.385 + 0.7873\lambda - 0.2756\lambda^2 + 0.0284\lambda^3 \\ {}^{NO}S_a(\lambda) &= 1.7441 - 0.493\lambda + 0.258\lambda^2 - 0.0254\lambda^3 \\ t_1 &= 15\,,\ t_2 = 30 \end{aligned} \tag{2.7.22} $$

at the northern boundary. Note that nitrate is expressed in mmol N m^{-3} and the time t_d in days since the beginning of the year.

e) river boundaries

Nitrate input is provided for most rivers at irregular times. Exceptions are the Humber and Seine for which the same input value is used throughout the year. In analogy with temperature a depth-uniform value is taken. A zero gradient condition is considered for all other biological parameters (microplankton and detrital carbon and nitrogen, ammonium, oxygen).

7.2.5 Sediments

a) model parameters and scheme

- Model parameters are set to their default values listed in Table 4.2.3.
- As the many sources and sinks of inorganic SPM in the North Sea are difficult to quantify, the initial condition is used to retain the essential spatial features throughout the year. This is achieved by enabling vertical sinking and resuspension of iSPM but disabling all horizontal advective processes. In this way a realistic spatial distribution of iSPM is retained throughout the simulation.

b) initial conditions

Inorganic SPM is supplied as a grid of depth-uniform concentrations which varied spatially across the region, derived from NOAA satellite observations (Vos and Schuttelaar 1995, 1997).

c) open boundaries

No open sea or river boundary conditions are required since horizontal advection of iSPM has been disabled.

7.2.6 Setup of model output

Model output is defined in the following files.

defmod.par

- time series output (`'U'`-format) at all grid points and at intervals of 7.5 days.
- time-averaged output (`'U'`)-format at all grid points with an averaging period of 30 days

defout.f

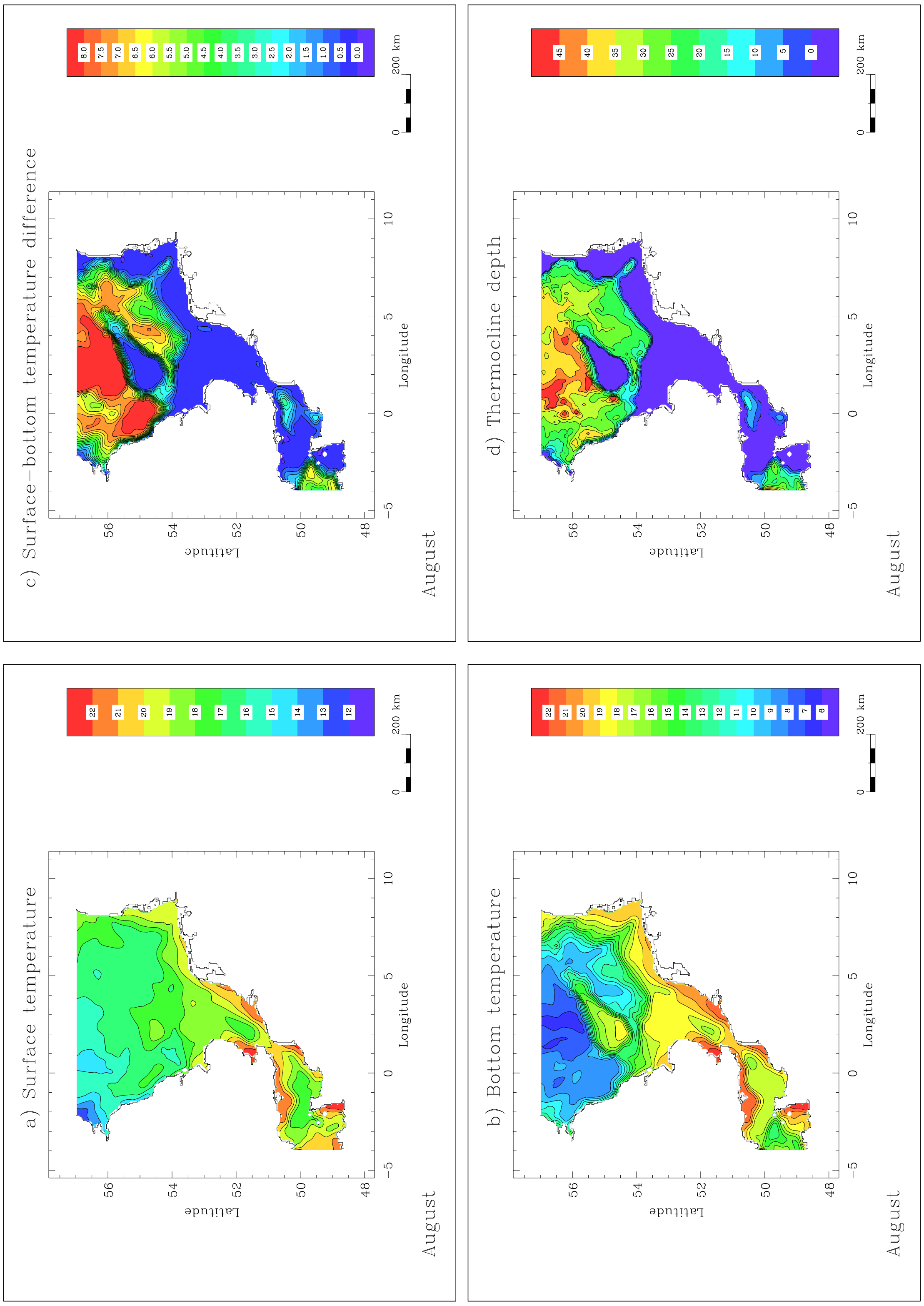


Figure 2.7.2: Thermal stratification in August: monthly averaged surface temperature (a), bottom temperature (b), surface-bottom temperature difference (c), thermocline depth (d). Temperature is in ^{0}C, depth in m.

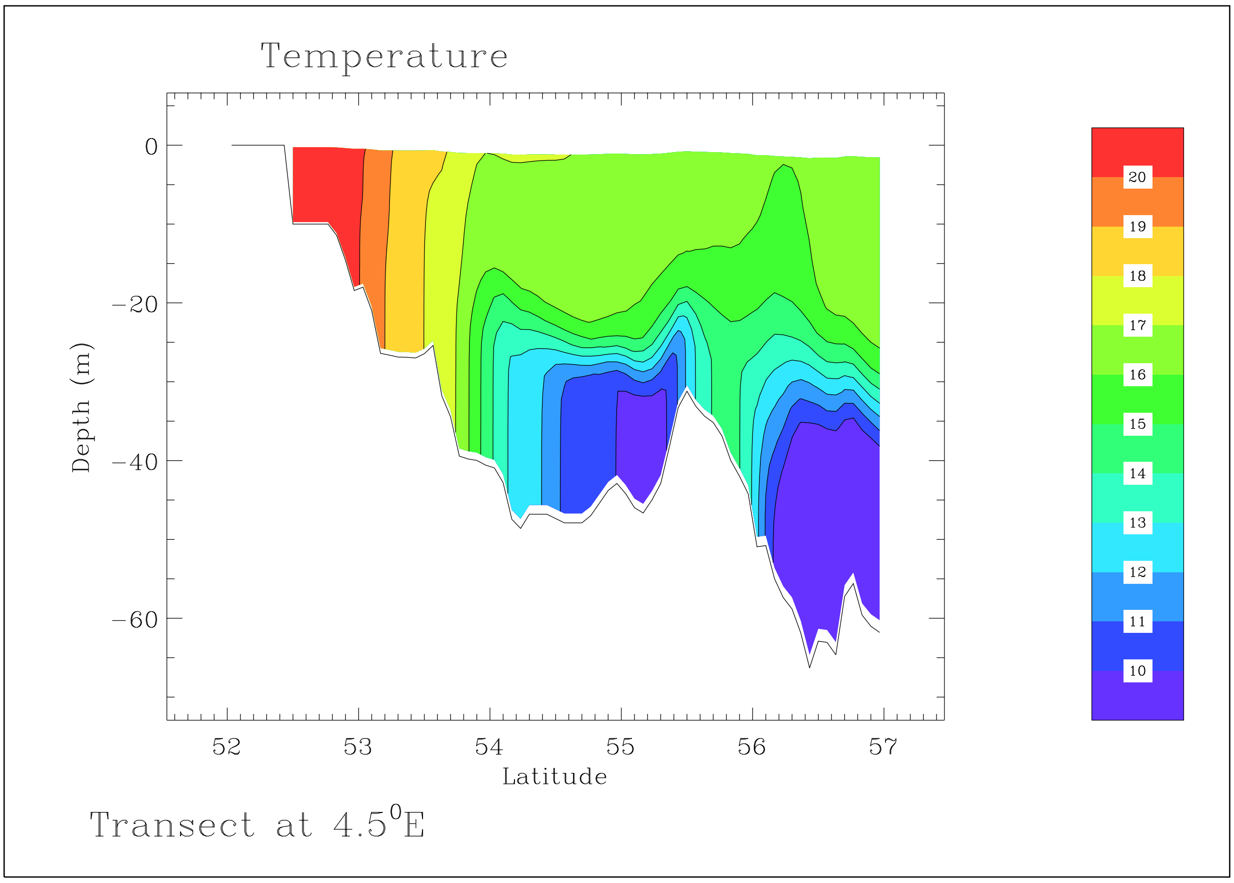
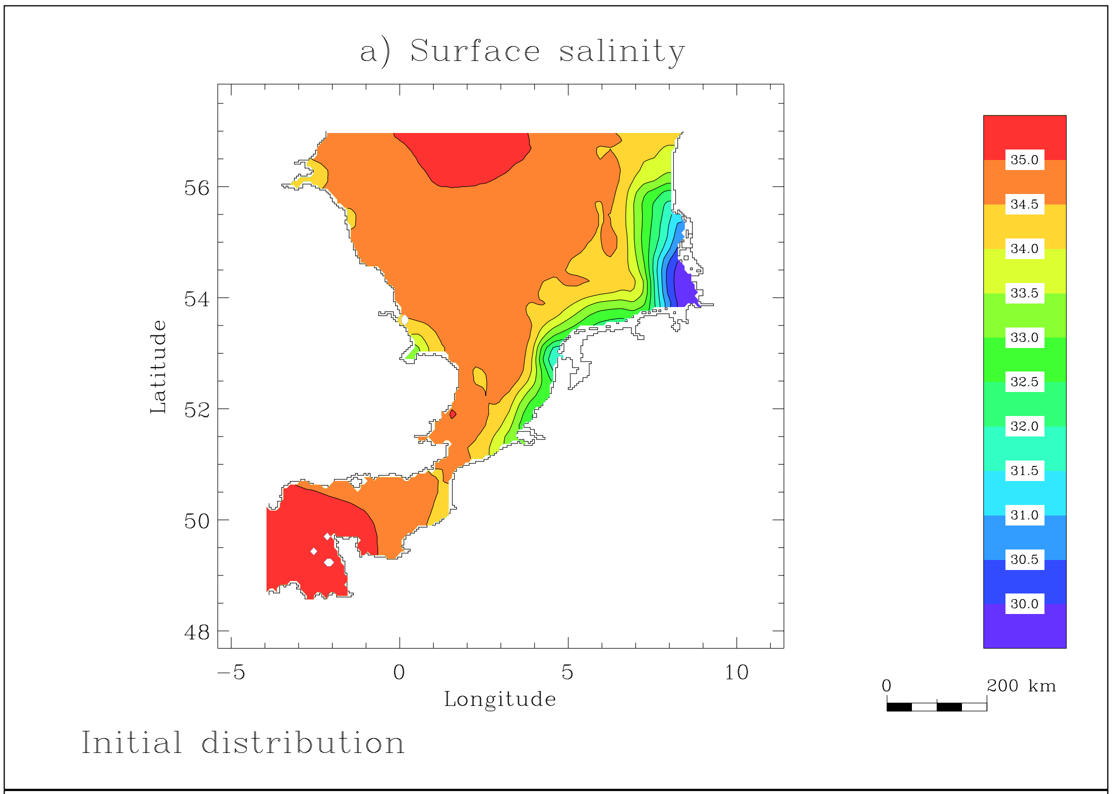


Figure 2.7.3: Monthly averaged temperature (^{0}C) distribution for August along a vertical transect at 4.5^{0}E.



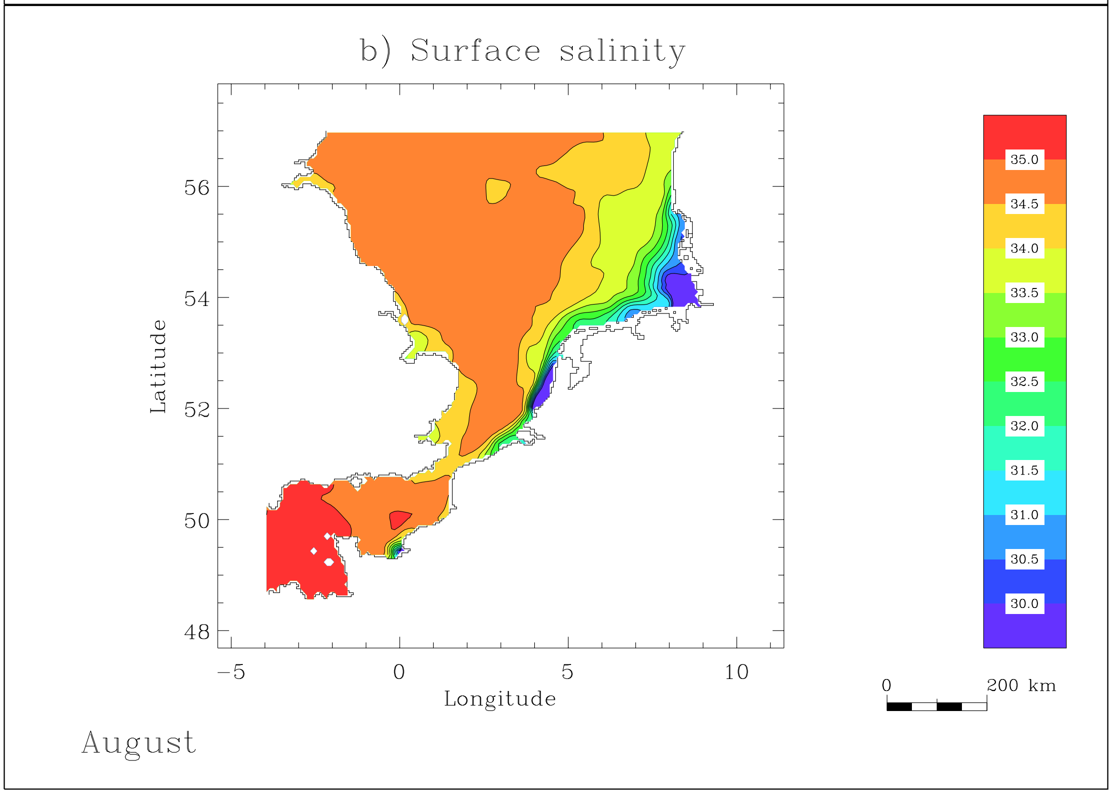


Figure 2.7.4: Surface salinity distribution (PSU): initial (a), monthly averaged field for August (b).

- 3-D fields: Eulerian displacement $\vec{\xi}$, salinity, temperature, biological state variables, iSPM concentration, PAR. The displacement vector is defined by

$$(\xi_\lambda, \xi_\phi, \xi_3) = \int_0^t (u, v, w)\, dt \tag{2.7.23}$$

  and can be considered as a measure of the residual field.

- 2-D fields: accumulated volume $\vec{V}_{acc}$, surface elevation, fluff layer amounts, fluff losses of carbon and nitrogen. The vector $\vec{V}_{acc}$ is defined by

$$(V_{acc,\lambda}, V_{acc,\phi}) = \int_0^t (\overline{U}, \overline{V})\, dt \tag{2.7.24}$$

defavr.f

- 3-D fields: salinity, temperature, biological state variables, iSPM concentration, microplankton growth rate μ, primary production rate μB, PAR
- 2-D fields: fluff layer amounts, fluff losses of carbon and nitrogen

print.f

- time series at all North Sea Project data stations and hourly intervals
  - vertical profiles of salinity, temperature, currents, biological state variables, iSPM concentration, PAR
  - surface elevation, depth-averaged current, fluff layer amounts, fluff losses
- a number of "integral" quantities at each time step (10 min):
  - volume-integrated values of salinity, heat content, biological and iSPM concentrations, ...
  - fractional amount of the area occupied by fresh water or where the water column becomes thermally stratified during summer
  - volume-integrated terms in the transport equations of salinity, temperature, biological state variables, inorganic SPM

  The general form of the latter terms is obtained by integrating the general transport equation (3.1.70) for a scalar ψ over the total volume of the area (excluding dry cells but including the fluff layer). This gives

$$\begin{aligned}\frac{\partial}{\partial t}\int_V \psi\, dV &= \int_W u\psi JR\, d\phi d\tilde{x}_3 - \int_E u\psi JR\, d\phi d\tilde{x}_3 \\ &+ \int_S v\psi \cos\phi JR\, d\lambda d\tilde{x}_3 - \int_N v\psi \cos\phi JR\, d\lambda d\tilde{x}_3 \\ &+ \int_V \beta(\psi)\, dV + \int_S \Big(\frac{\lambda_T}{J}\frac{\partial\psi}{\partial\tilde{x}_3}\Big)\Big|_{sur} dS - \int_S \Big(\frac{\lambda_T}{J}\frac{\partial\psi}{\partial\tilde{x}_3}\Big)\Big|_{bot} dS - \int_S F_l\, dS\end{aligned} \tag{2.7.25}$$

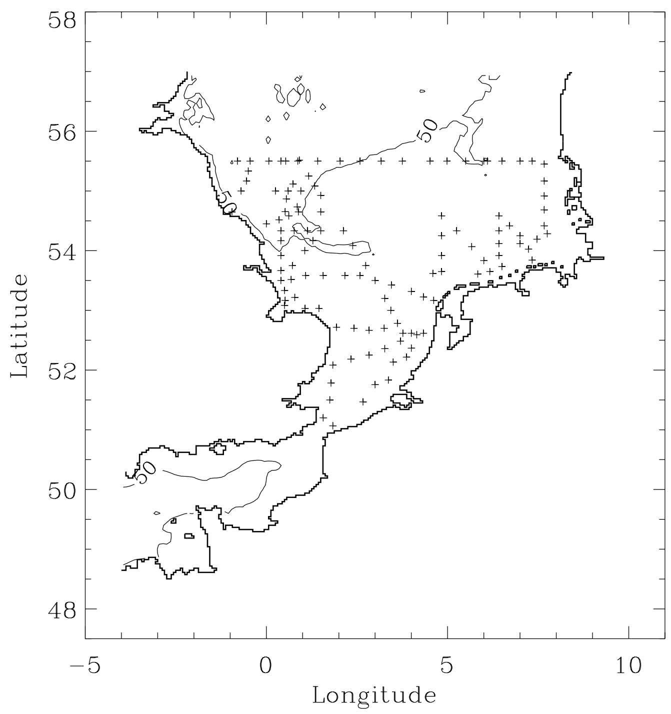
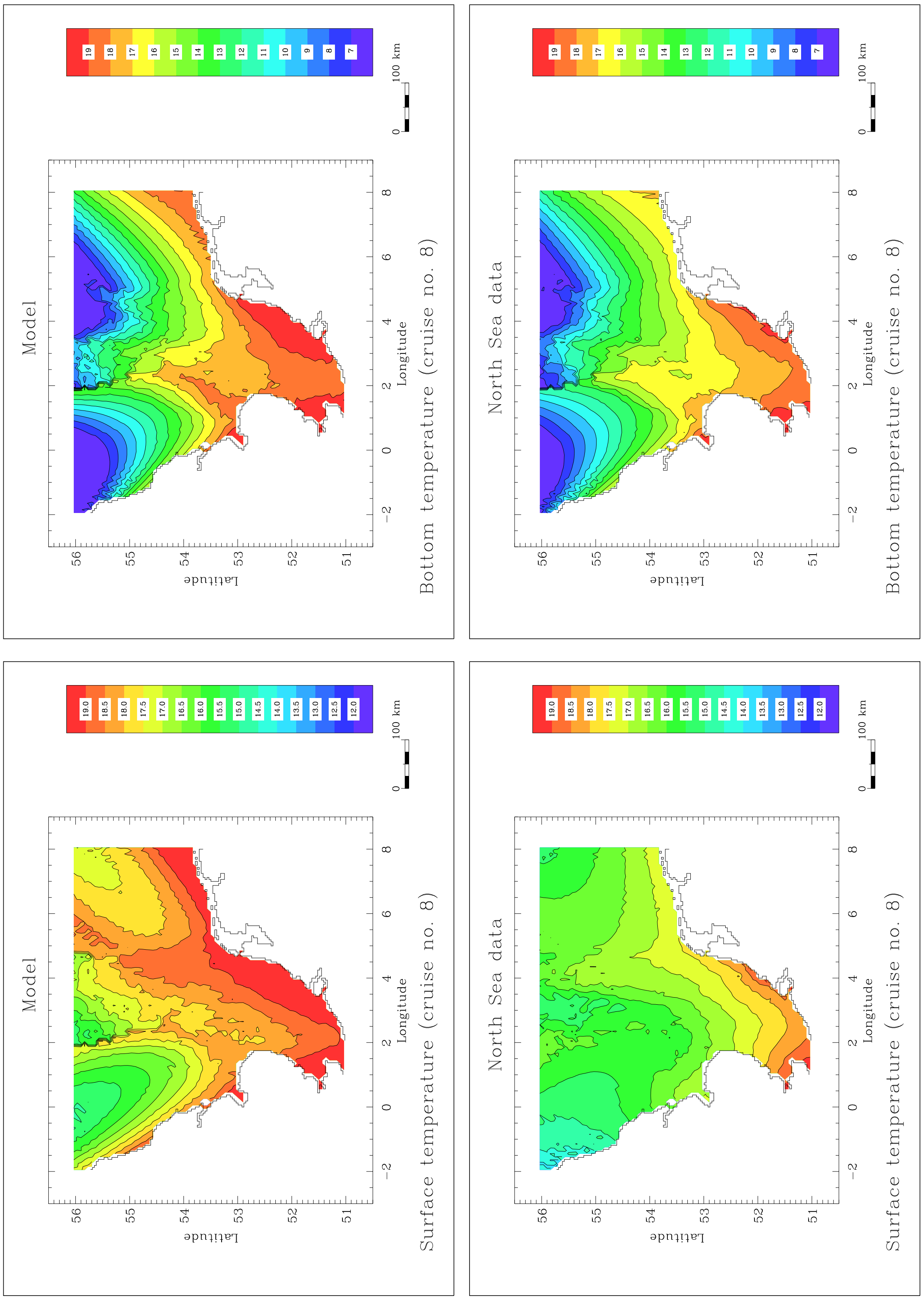


Figure 2.7.5: Location of the North Sea Project data stations.

with $\beta(\psi) = \mathcal{P}(\psi) - \mathcal{S}(\psi)$. See Chapter III-1 for an explanation of the different notations. The first four terms on the right hand side of (2.7.25) represent the net inward fluxes along the western, eastern, southern and northern open boundaries. These terms arise from contributions along the western open sea boundary, the northern open sea boundary and from river inputs. The fifth term is the net production rate from source and sink terms. The sixth and seventh terms are the contributions of the surface and bottom fluxes. The last term represents the amount of material (microplankton or detrital carbon or nitrogen) lost from the fluff layer to the consolidated sediment.

7.2.7 Setup of the simulations

A series of preliminary runs were performed to check the setup of the model. A first run was made with tides only but without winds and stratification. Wind was introduced in a second run, temperature in a third and salinity in a fourth. Biology and sediments were included in the final run. A simulation with only the physics took about 4 days on the



Model
Bottom temperature (cruise no. 8)
North Sea data
Bottom temperature (cruise no. 8)
Model
Surface temperature (cruise no. 8)
North Sea data
Surface temperature (cruise no. 8)
Latitude
Longitude
0
100 km

Figure 2.7.6: Surface and bottom temperature (^{0}C) according to the model and to the data for cruise no. 8 (end of July and beginning of August). The contour plots are made using only values at the North Sea Project data stations.

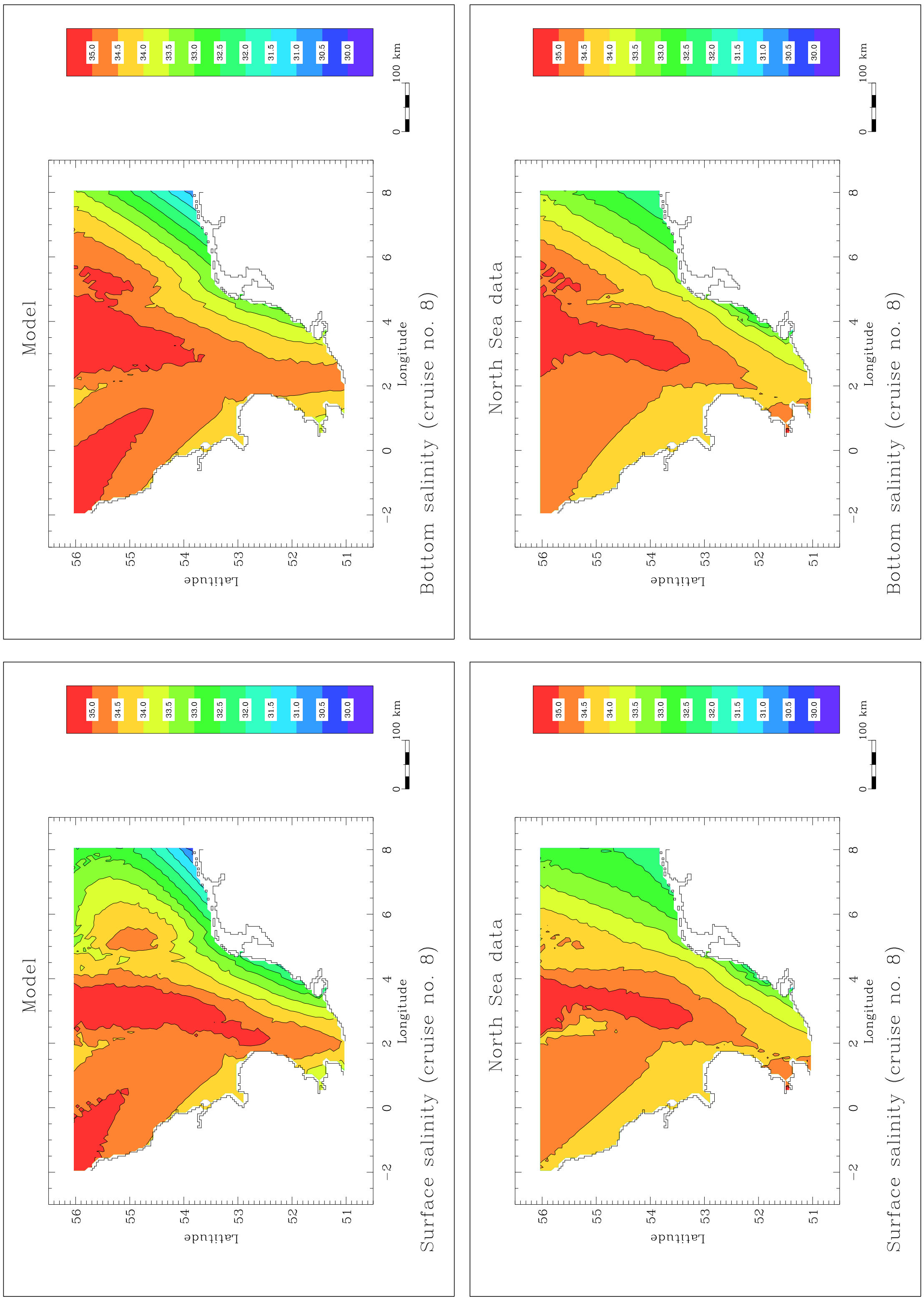


Figure 2.7.7: Surface and bottom salinity (PSU) according to the model and to the data for cruise no. 8 (end of July and beginning of August). The contour plots are made using only values at the North Sea Project data stations.

DEC Alpha machine at MUMM and about 9 days for the full run with the biology and sediments.

In view of the long simulation time and in order to reduce the risks of unexpected problems (e.g. system crash, . . .) the simulation was split up into a spin-up phase, 11 runs of 30 days and a last run of 28 days:

- The program is run for a spin-up phase of 3 days without temperature, salinity, biology and sediments but with 2-D open boundary forcing, to initialise the currents.
- Temperature, salinity, biology and sediments are initialised while the current is read from the result file. Open boundary and surface forcing are provided for the first run.
- The program is run for the first 30 days.
- A second run is prepared with new open boundary input. The current and all scalar fields are read from the result files of the previous run.
- The previous steps are repeated until the end of the year.

The procedure shows how the model can be set up in “operational” mode.

7.3 Results

7.3.1 Physics

The general features of the (modelled) thermal stratification in the summer season are summarised in Figures 2.7.2 and 2.7.3. The first shows the surface temperature (a), bottom temperature (b), surface-bottom temperature difference (c) and the thermocline depth (d), the second the temperature distributions along a vertical transect at 4.5^0E. The data are averaged over a period of 30 days in August. The surface temperature shows an obvious North-South gradient with a stronger heating in shallow coastal areas. The boundary between the (thermally) stratified and mixed areas is marked by a sharp front which extends eastwards from the British coast upto the central North Sea (between 53.5^0N and 54^0N), then curves upwards first northeastwards and finally northwards parallel to the Danish coast (Figures 2.7.2b–d). The phenomenon is clearly related to the bathymetry as can be seen by comparing the thermocline depth in Figure 2.7.2d with the bathymetry in Figure 2.7.1. The transition between the mixed and stratified regime occurs at a water depth of $\sim$30 m (Figure 2.7.3). In particular, the shallow triangular area of the Dogger Bank where the water column remains vertically mixed during summer, is clearly discerned from the surrounding deeper waters.

The initial surface salinity field and its distribution in the beginning of August are given in Figure 2.7.4. No global changes are observed except that the plume fronts are stronger in the Rhine and Elbe-Weser outflow areas, the appearance of a patch of relatively fresh water in the Dover Strait and the Seine plume visible in the South.

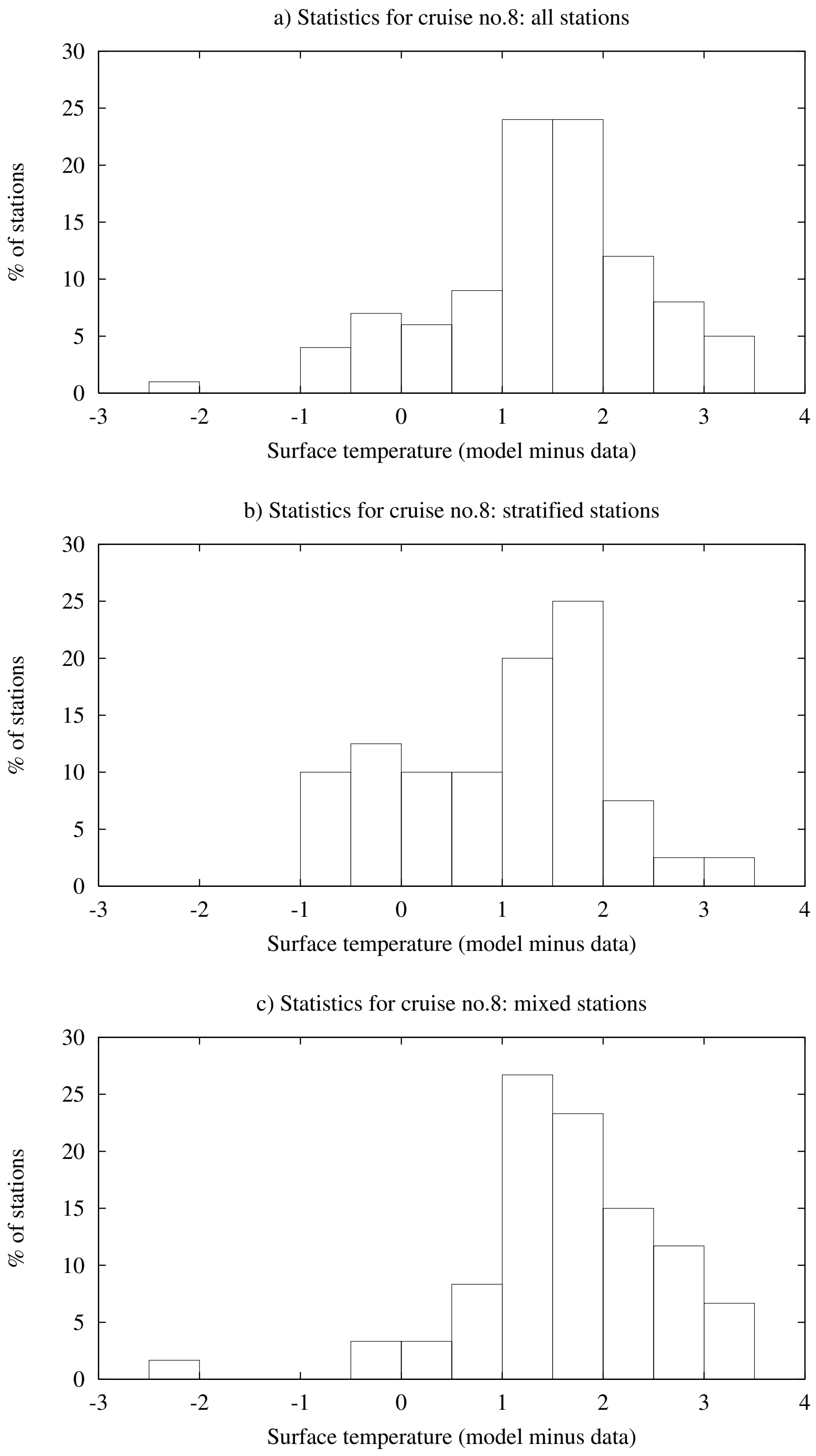


Figure 2.7.8: Histograms of the modelled minus data surface temperature: all stations (a), stratified stations (b), mixed stations (c).

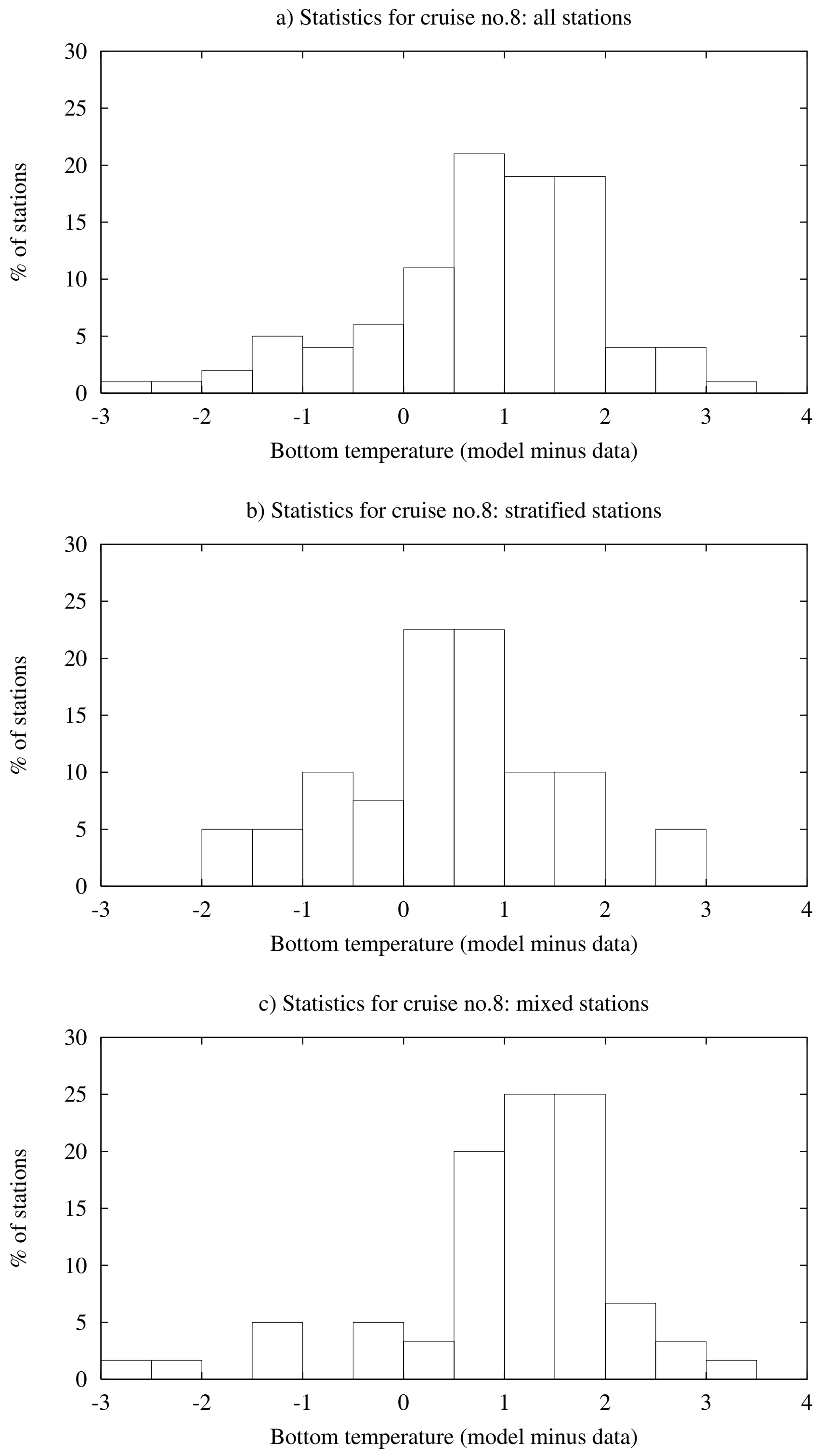


Figure 2.7.9: Histograms of the modelled minus data bottom temperature: all stations (a), stratified stations (b), mixed stations (c).

The model data are compared with the observations of the North Sea Project (Howarth et al, 1994). The data stations are plotted in Figure 2.7.5. Profiles of temperature and salinity were taken at most stations during 10 cruises at intervals of approximately one month. The sampling period in each cruise was between 12 and 14 days. Surface and bottom values for cruise no 8 (July 24 to August 6) according to the model and the data are shown in Figure 2.7.6 for temperature and Figure 2.7.7 for salinity. These composite contour plots were constructed by interpolating the measured value to the surface or bottom grid point and by taking the modelled value at the same time when the measurement was made. It should be noted that, in view of the limited number of data points, especially in the central part of the area, some features of the thermal fronts (see Figure 2.7.2) are not well resolved in these plots. Surface temperatures are mostly overpredicted by 1–2^{0}C with a larger overestimate in the mixed compared to the stratified zone. The largest deviations occur in the shallow coastal plume areas. The results are qualitatively the same for the bottom temperatures but with a better agreement with the observations. Compared to the surface temperatures the overestimates are lower in the mixed area while a slight underestimation seems to occur in the deepest part of the stratified area. The previous conclusions are confirmed by the histograms, shown in Figures 2.7.8 and 2.7.9.

As observed in Figure 2.7.7, the modelled surface and bottom salinity fields are in good agreement with the data. Differences are mostly lower than $\pm$0.5 PSU. Exceptions are the Thames estuary and the coastal plumes along the continental coast extending from the Rhine plume upto the German Bight where both surface and bottom salinity are underpredicted and an area along the British coast where both values are somewhat overestimated by the model. Since the tendencies are the same in the surface and the bottom fields, it can be concluded that salinity stratification is generally well predicted by the model.

To see how model and data compare over a more complete seasonal cycle, the surface and bottom temperatures are averaged for each cruise first over all stations, then over all stratified and finally over all mixed stations. The results are displayed in Figure 2.7.10 for the surface and in Figure 2.7.11 for the bottom temperature. Model values are shown by solid lines, data values by dotted curves. Each symbol (diamond or square) refers to a specific cruise. There is a slight underprediction during the first winter months of $\sim$0.5^{0}C (due to the initial field). Although model and data are in good agreement during early spring, the modelled surface temperatures rise more rapidly afterwards with a maximum deviation for cruise no.8 in the summer (see above). The differences become smaller at the end of August and in September. Although no data values are available after September, there is a clear tendency in all figures for the model temperatures to approach the measured values in autumn. Agreement is generally better for the stratified compared to the mixed stations. The same tendencies apply for the bottom temperatures but with smaller differences (-0.5^{0}C< ΔT<1^{0}C). Interesting to note is that similar results were recently obtained by Holt and James (1999) using a different model but with the same surface forcing data.

The excess heating cannot be explained by the turbulence or the optical module in the program since it occurs both for the stratified as for the mixed areas. To test the role of advection a simulation was performed without (horizontal and vertical) advection in

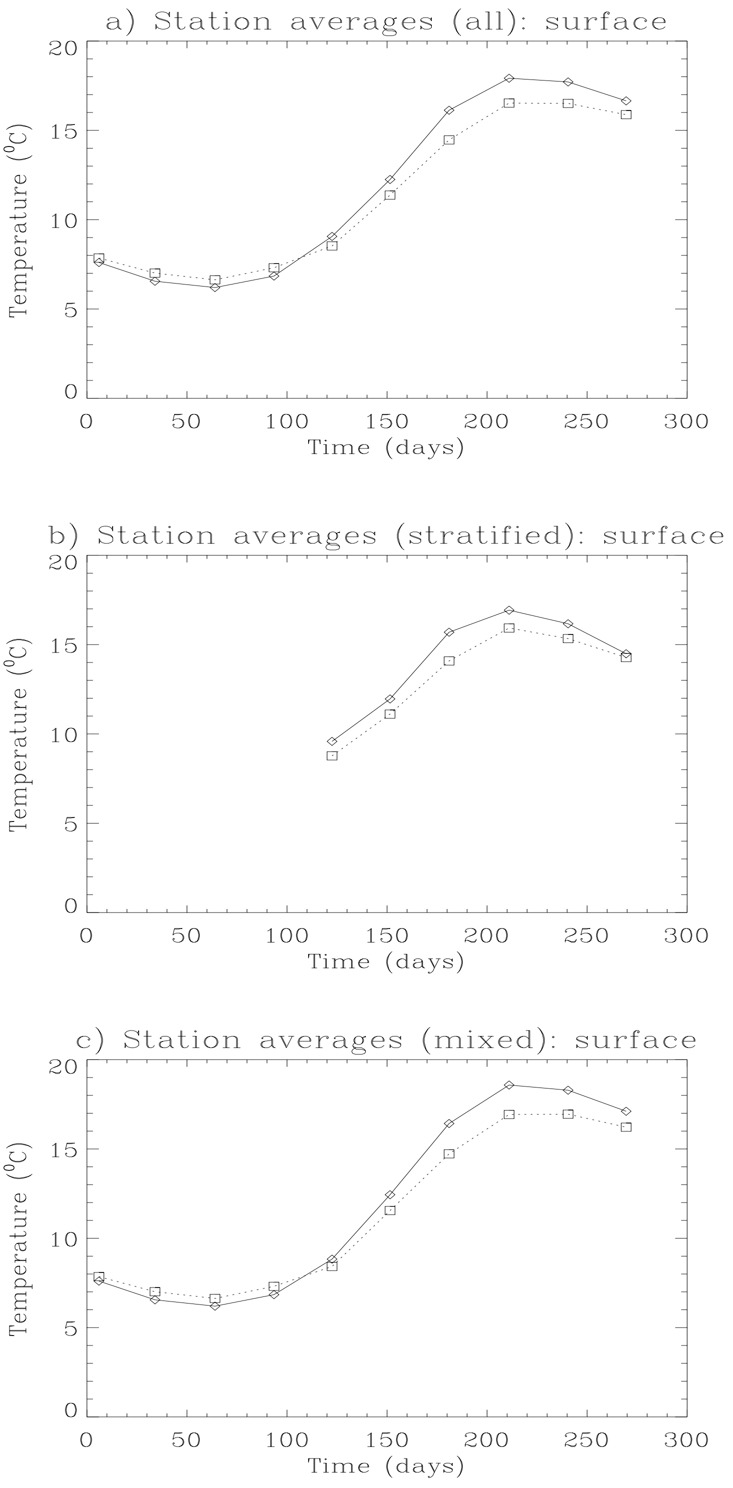
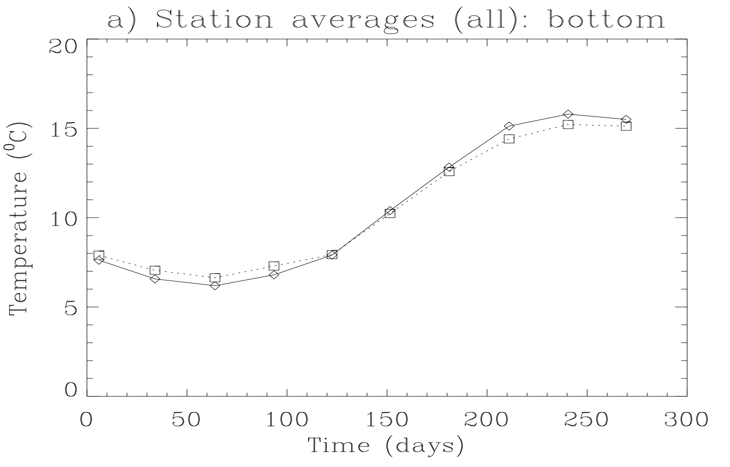
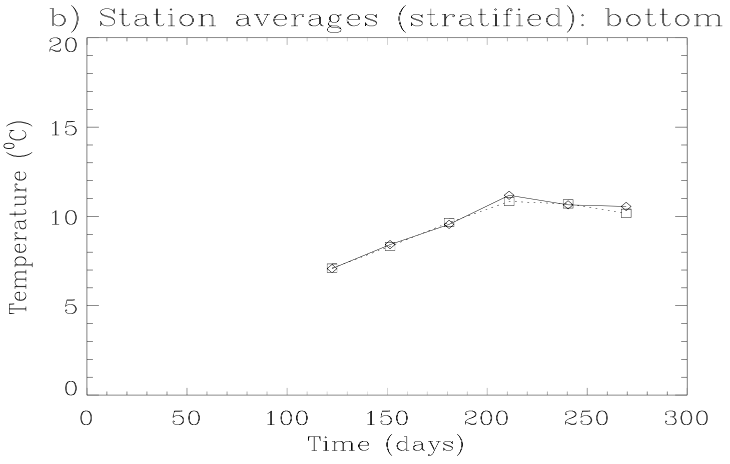


Figure 2.7.10: Surface temperature averaged over all stations (a), stratified stations (b), mixed stations (c) according to the model (solid line and diamonds) and the data (dots and squares).





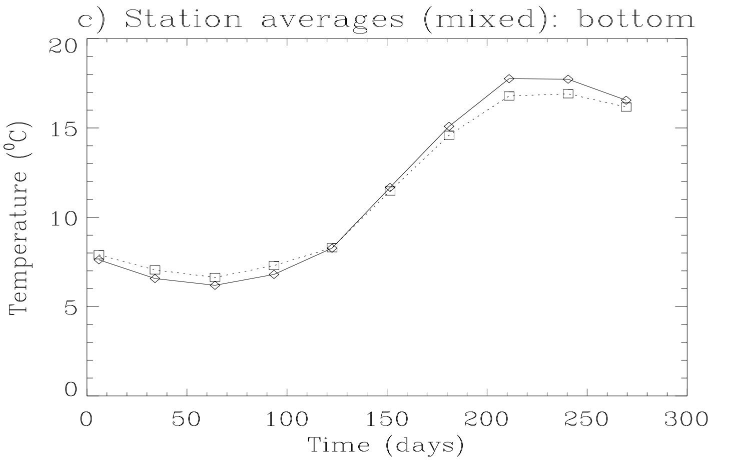


Figure 2.7.11: Bottom temperature averaged over all stations (a), stratified stations (b), mixed stations (c) according to the model (solid line and diamonds) and the data (dots and squares).

the temperature and salinity equations but with advection still enabled in the momentum equations. It was found (not shown) that advection only has a minor impact on the globally averaged temperature field except for the bottom temperatures in the stratified zone.

The stratified area is obviously more difficult to model because of the presence of a thermocline in summer. The thermocline depth and the vertical temperature distribution now depend on the details of the turbulence and optical schemes used in the model. As a typical example the time series of the surface and bottom temperatures at station CS ($55^0 30'$N, $0^0 57'$E) are shown in Figure 2.7.12 according to the model run with advection, the model run without advection and the data. Surface temperatures are generally higher than or in agreement with the data although no firm conclusion can be drawn since the data in the summer were accidentally taken at times of local temperature minima. Good agreement is found at the bottom. Small semi-diurnal oscillations, marking the presence of an internal tide, are seen in the curve for the bottom temperature. They are clearly absent in the run without advection. An important difference between model and data is that vertical stratification starts to decrease by the erosion of the thermocline in September according to the observations while this occurs in the model results only at the end of October. This is seen more clearly in the time-depth contour plots of Figure 2.7.13. Note also the strong deepening of the thermocline in the data during spring after which the thermocline depth remains constant. This occurs more gradually in the model results, although good agreement is obtained in summer.

A global analysis has been made of the different inputs and outputs for temperature (represented by the heat content) and salinity. This is performed by integrating the temperature and salinity equations over the computational domain (see equation (2.7.25)). Results are shown in Figure 2.7.14. It is clear that the surface forcing provides the largest input in the case of temperature. Future studies should therefore concentrate on improvements for the parameterisations of the surface heat fluxes. Also, better data values are needed for cloud coverage. In the case of salinity the largest input comes from the open boundaries. River input and surface forcing are (mostly) of comparable magnitude.

7.3.2 Biology

a) seasonal patterns in spatial distributions

Model results show a seasonal cycle of winter mixing and summer thermal stratification in deeper waters North of 54^0N and persistently vertically mixed waters to the South. Exceptions to this broad pattern were the shallow waters over the Dogger Bank and off the Danish coast, which remained well mixed, and the deeper water in the central approaches to the English Channel which stratified in summer. The model transition from mixed to stratified waters at 54^0N coincided with the Flamborough Head (Hill et al. 1994) and Fresian Islands fronts (see Figure 2.7.2).

The seasonal cycle of depth-integrated chlorophyll, nitrate and primary production (given as the growth rate μ times the microplankton carbon concentration C) and the attenuation coefficient k_d at 10 m depth is illustrated in Figures 2.7.15 and 2.7.16 sho-

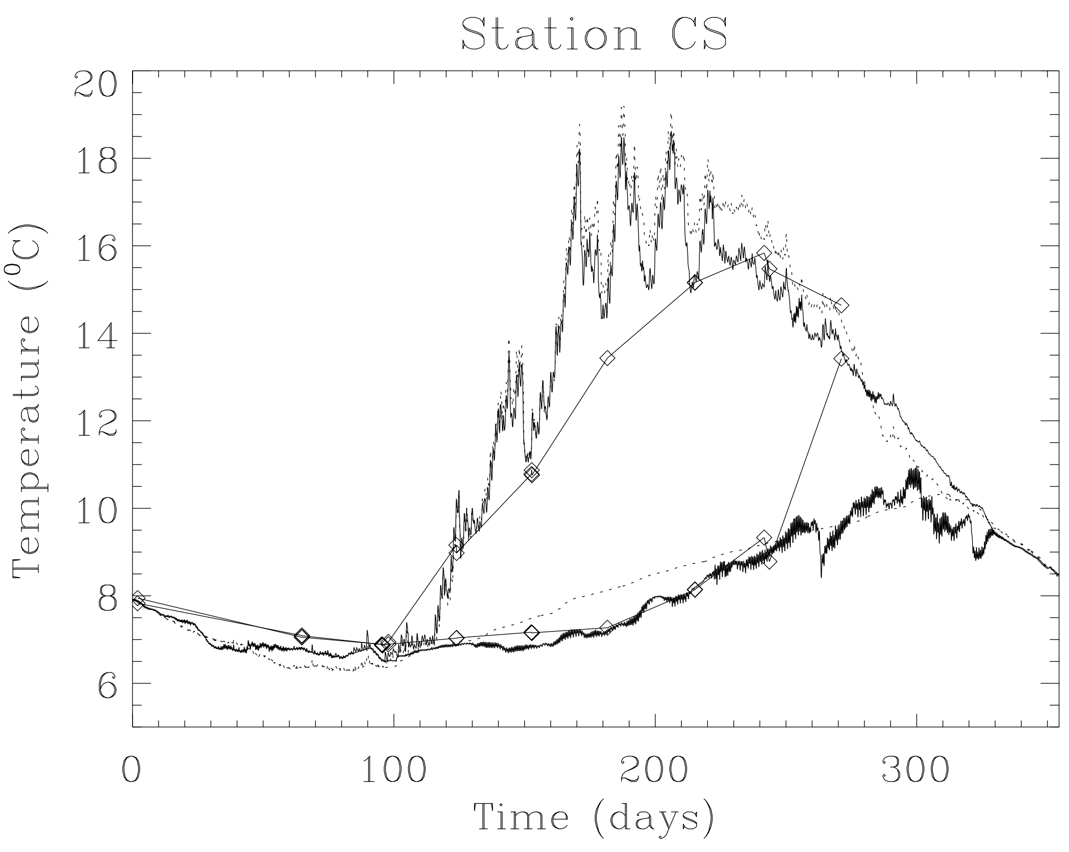
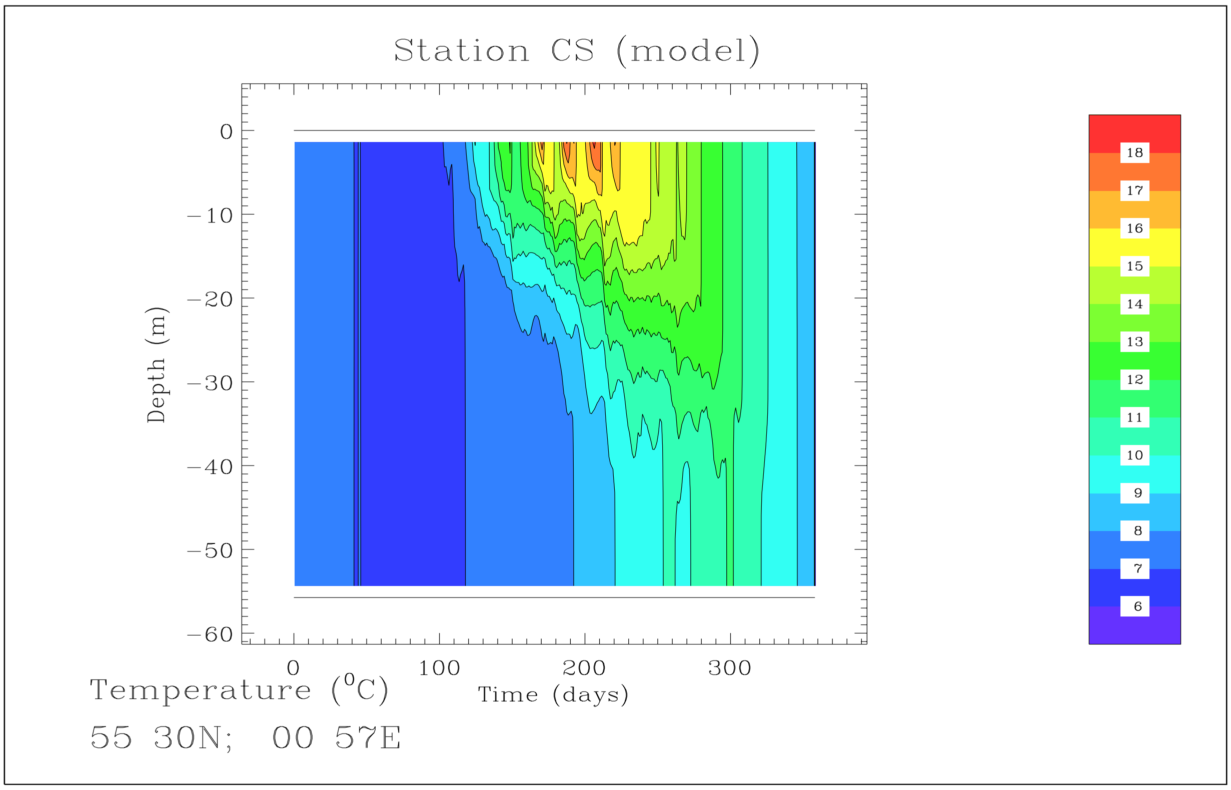


Figure 2.7.12: Time series of surface and bottom temperature at station CS: with advection (solid), without advection (dots) and data (diamonds).



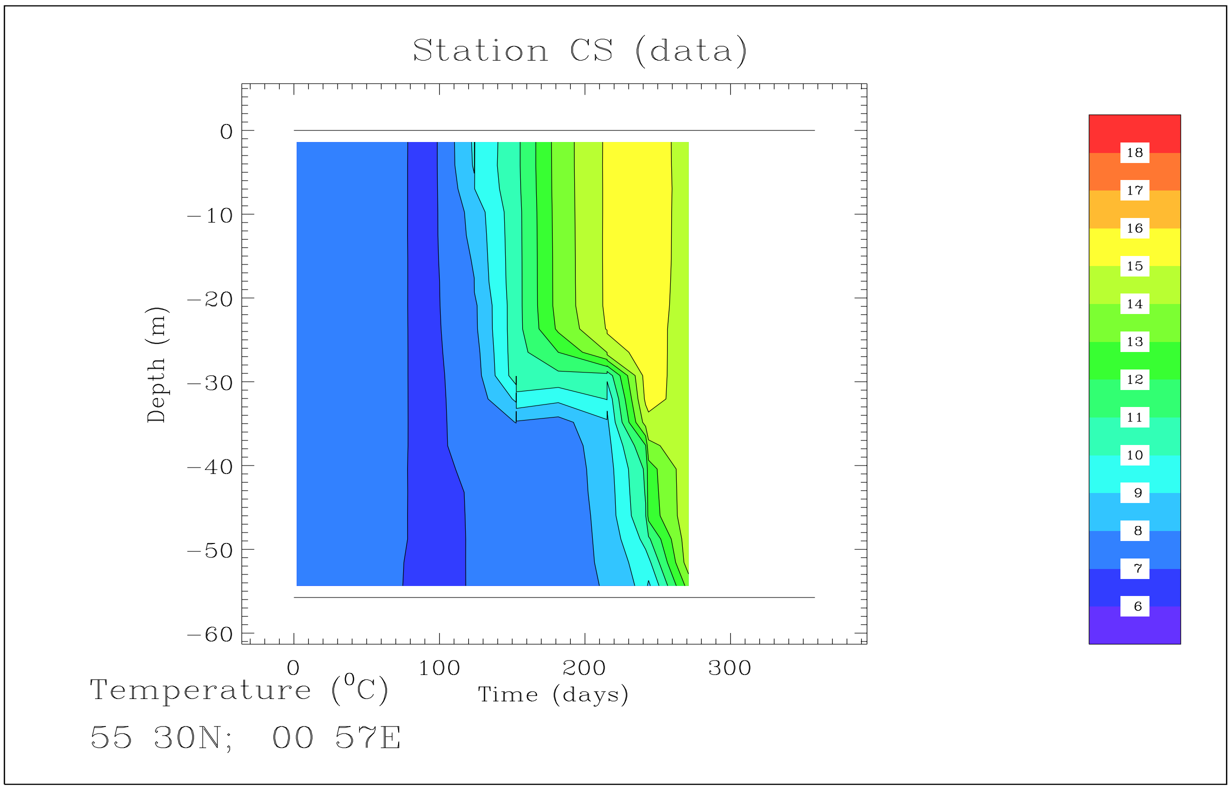
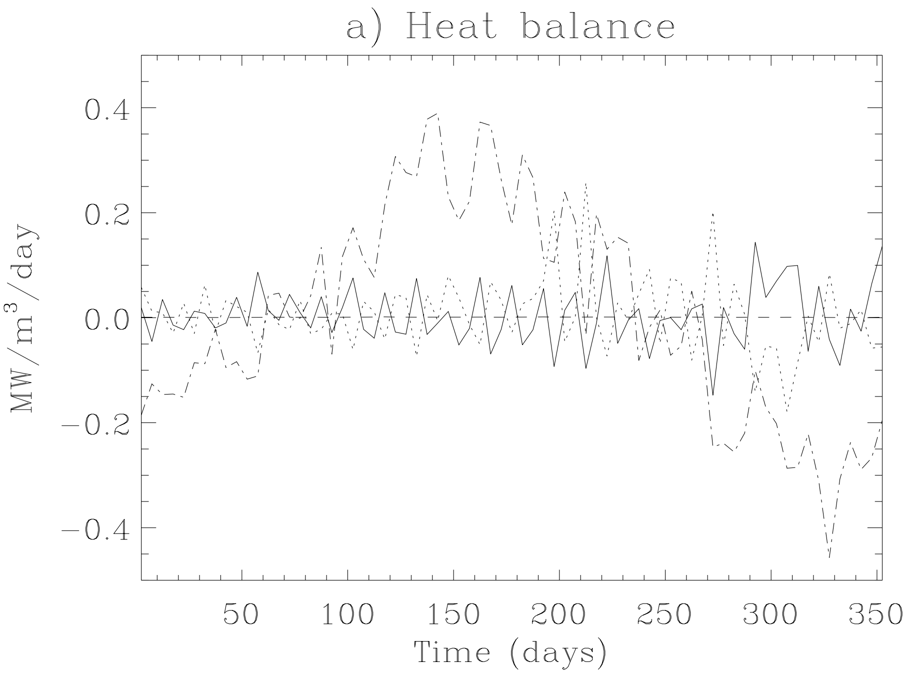


Figure 2.7.13: Depth-time contour plots of temperature (^{0}C) at station CS according to the model (top) and to the data (bottom).



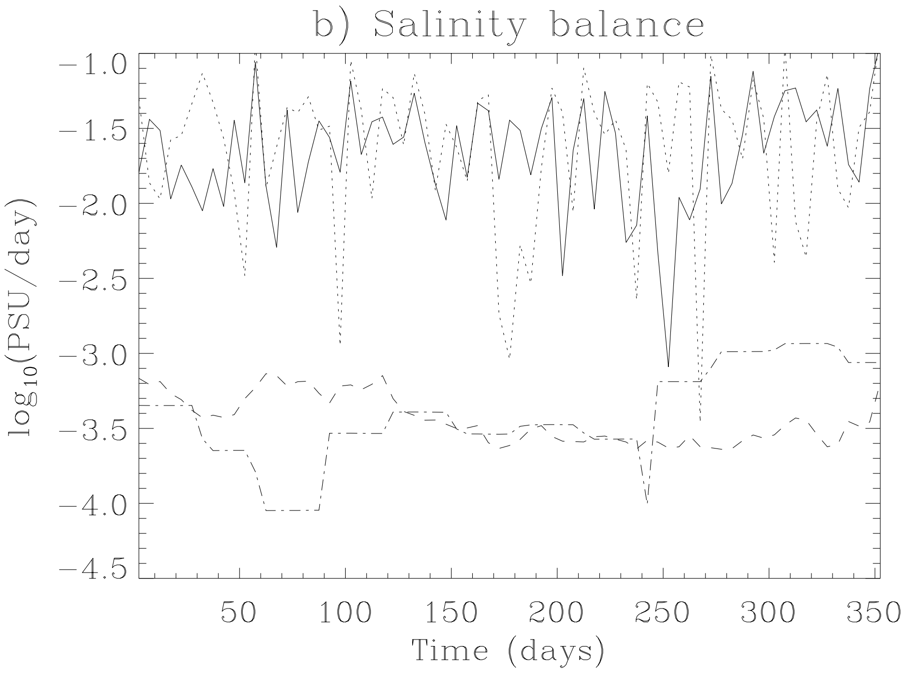


Figure 2.7.14: Input forcing terms (averaged over 5 days) for the global heat content (a) and salinity (b) : western boundary (solid), northern boundary (dots), river input (dashes) and surface forcing (dash-dots). The terms are obtained by integrating the transport equations for temperature (times $\rho_0 c_p$) and salinity over the whole water mass of the computational domain (see equation (2.7.25)) and dividing by the total water volume.

wing monthly averaged distributions from May to December. In the seasonally stratified waters near surface chlorophyll values were typically low. During summer stratification microplankton biomass was concentrated in subsurface layers which deepened over the summer period. Near surface nutrient concentrations were rapidly depleted in spring and were not replenished until the breakdown of stratification occurred in late autumn. The low iSPM, generally clear waters, facilitated the penetration of PAR and allow photosynthetic production to be sustained at depths where nutrients remained. Depth-integrated primary production was high (weighted by the large depth of water included in the sum), during the season of summer stratification (Figure 2.7.16).

The vertically mixed waters sustained higher levels of near surface chlorophyll from distributions which were in broad agreement with observations made during the North Sea Project (Howarth et al. 1994). During the early spring increase a coherent patch of low chlorophyll concentration was simulated and observed off the Norfolk coast (Figure 2.7.15). This feature was thought to arise from relatively deep mixing and high levels of iSPM, which resulted in unfavourable light conditions for microplankton growth. Throughout the summer period near surface chlorophyll values simulated in the English Channel and Southern Bight were generally higher than observed. This probably resulted from the simulation and advection of unrealistically high concentrations of biomass in the approaches to the English Channel. Excessive biomass may have been generated in this region due to an underestimation of iSPM concentration (and hence turbidity), an underestimation of zooplankton grazing, and/or an excessive supply of nutrients at the models western boundary. Depth-integrated primary production was generally lower than for the seasonally stratified waters.

In regions close to major river discharges, especially the Rhine, Seine, Elbe, Wash and Thames, enhanced nutrient concentrations supported additional microplankton growth and high concentrations of chlorophyll (>30 mg m^{-3}). These inshore features could not be verified by the North Sea Project data set although increased levels of chlorophyll were periodically observed at the closest stations to the Dutch, German and Danish coasts (Tett et al. 1993; Howarth et al. 1994).

Comparisons have been made between the COHERENS simulation results and observations made at North Sea Project site CS in 1989 (Mills and Tett, 1990; Tett et al., 1993; Mills et al., 1994; Sharples and Tett, 1994; Tett and Walne, 1995). Depth-time contour plots of chlorophyll and nitrate at CS are shown in Figure 2.7.17. The rapidly-sinking spring bloom, and the mid-summer chlorophyll midwater maximum, seem well simulated. Simulated nitrate behaved realistically during the spring bloom, but not during the autumn, which may be explained by the unrealistic benthic mineralisation model.

Finally, the time series of the daily-averaged chlorophyll and nitrate concentrations, averaged over the whole water mass, are given in Figure 2.7.18. A global chlorophyll maximum occurs around days 120–150 (May) with a corresponding minimum in nitrate.

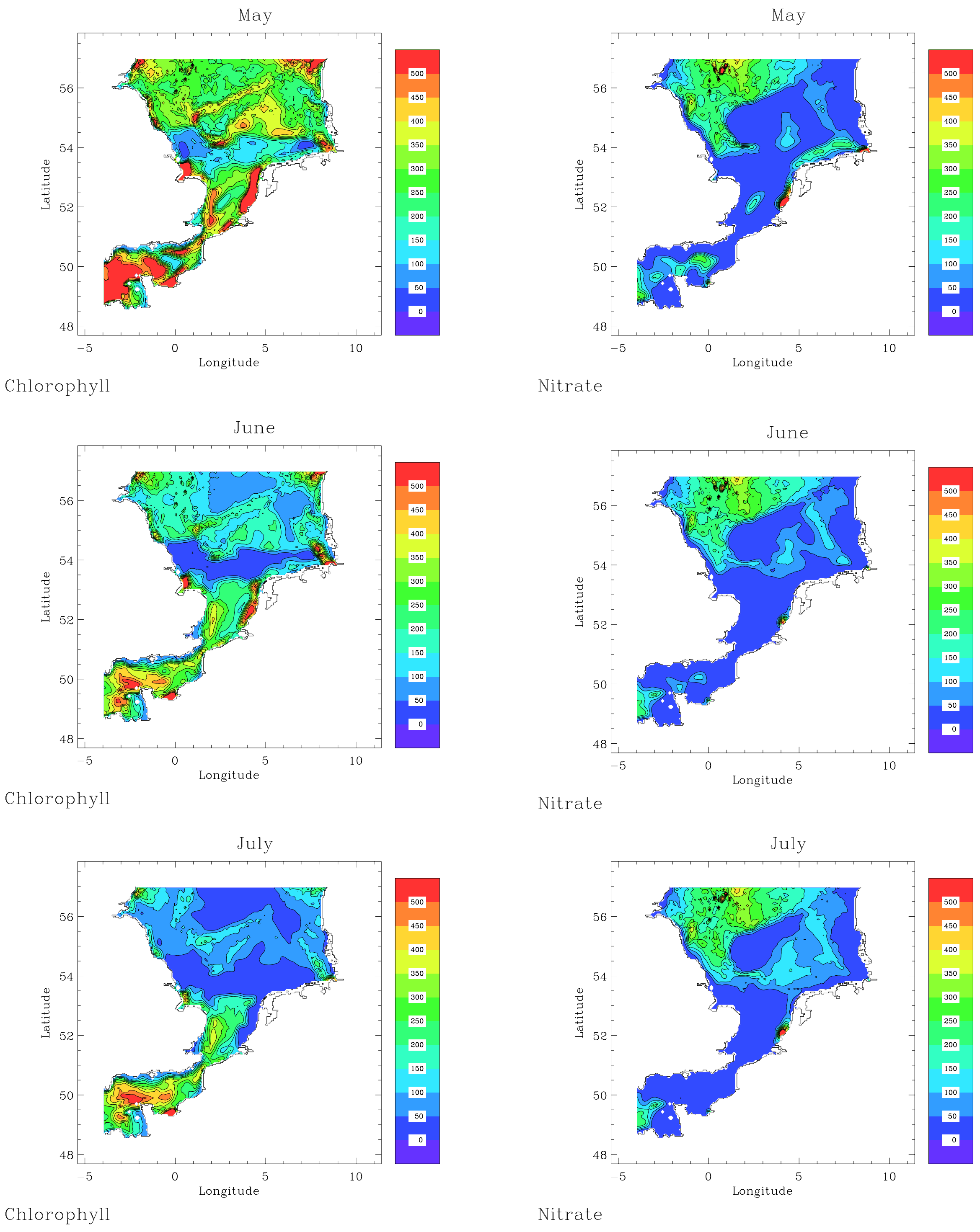


Figure 2.7.15: Monthly averaged values of depth-integrated chlorophyll in mmol Chl m^{-2} (left) and nitrate in mmol N m^{-2} (right) for May, June, July, September, October and December.

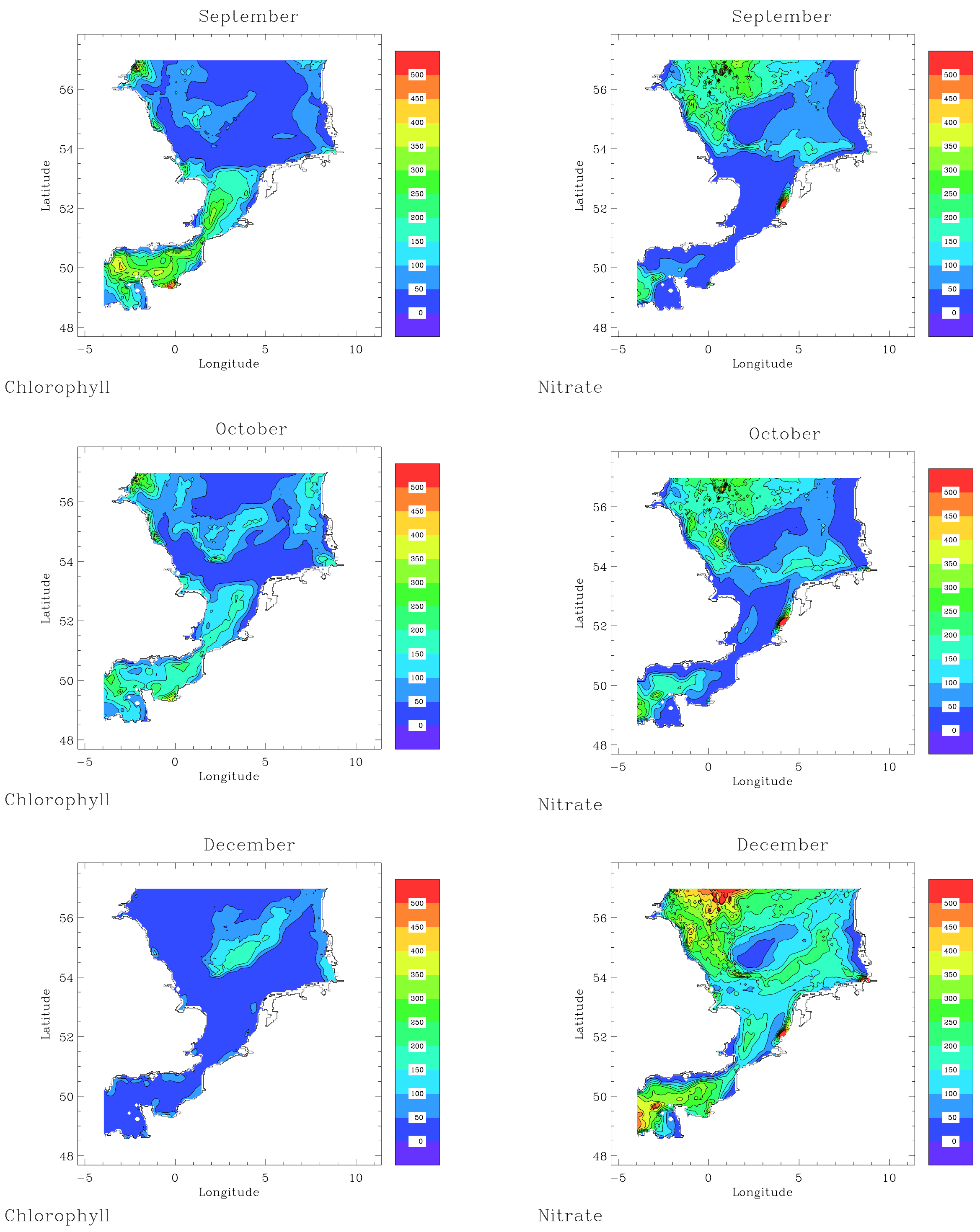


Figure 2.7.15: continued.

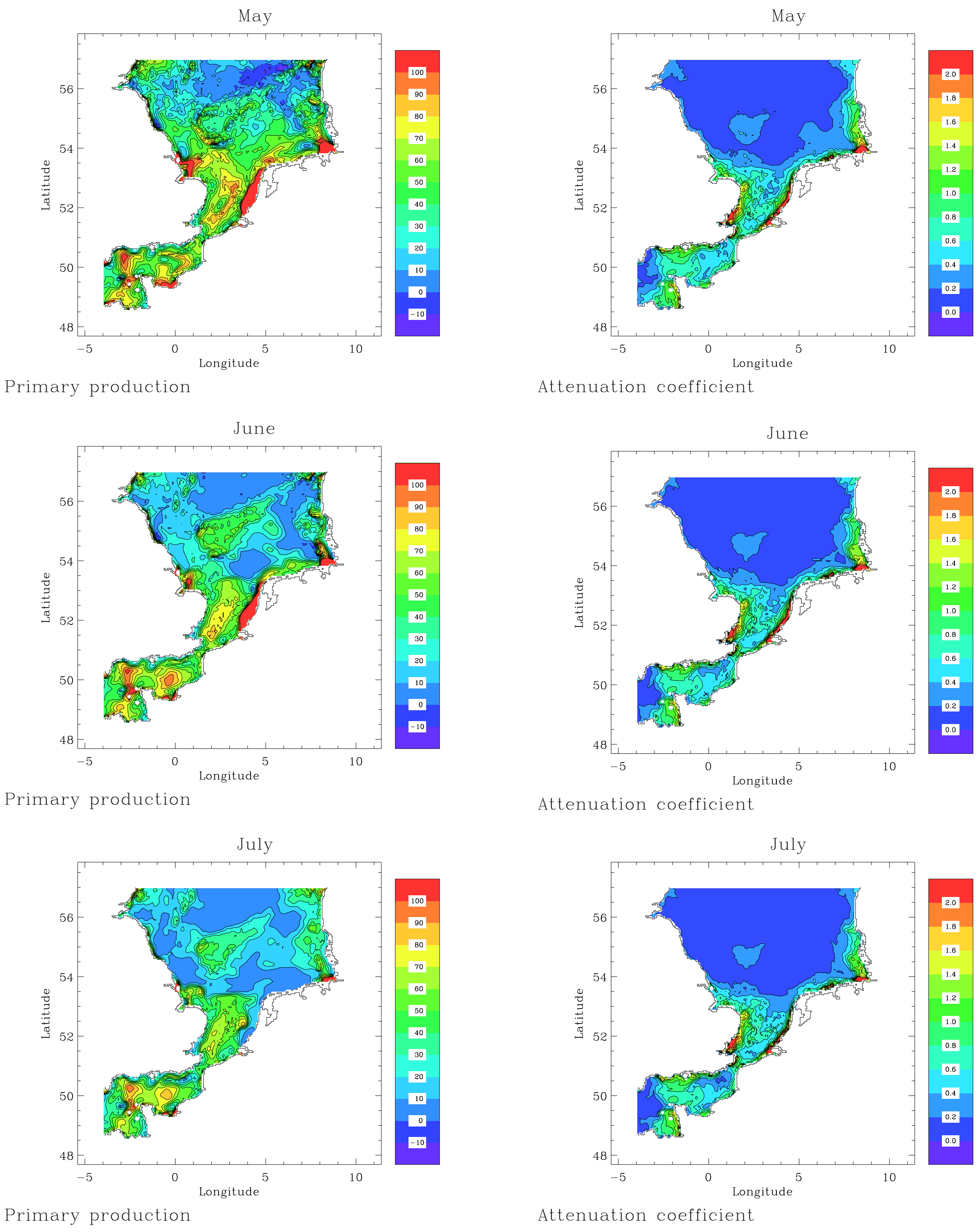


Figure 2.7.16: Monthly averaged values of primary production rate in mmol C m^{-2} day^{-1} (left) and the diffuse attenuation coefficient in m^{-1} at 10 m depth (right).

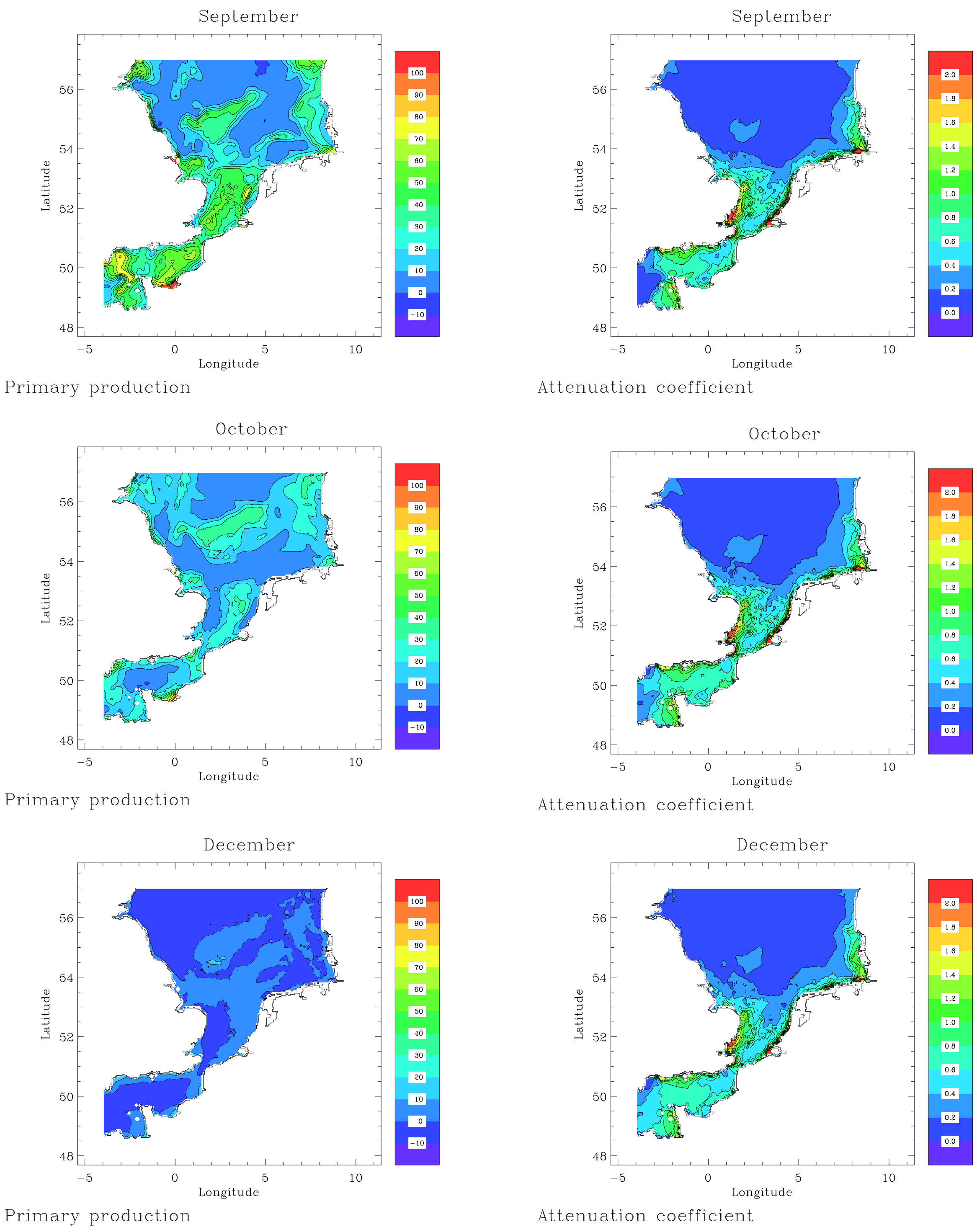
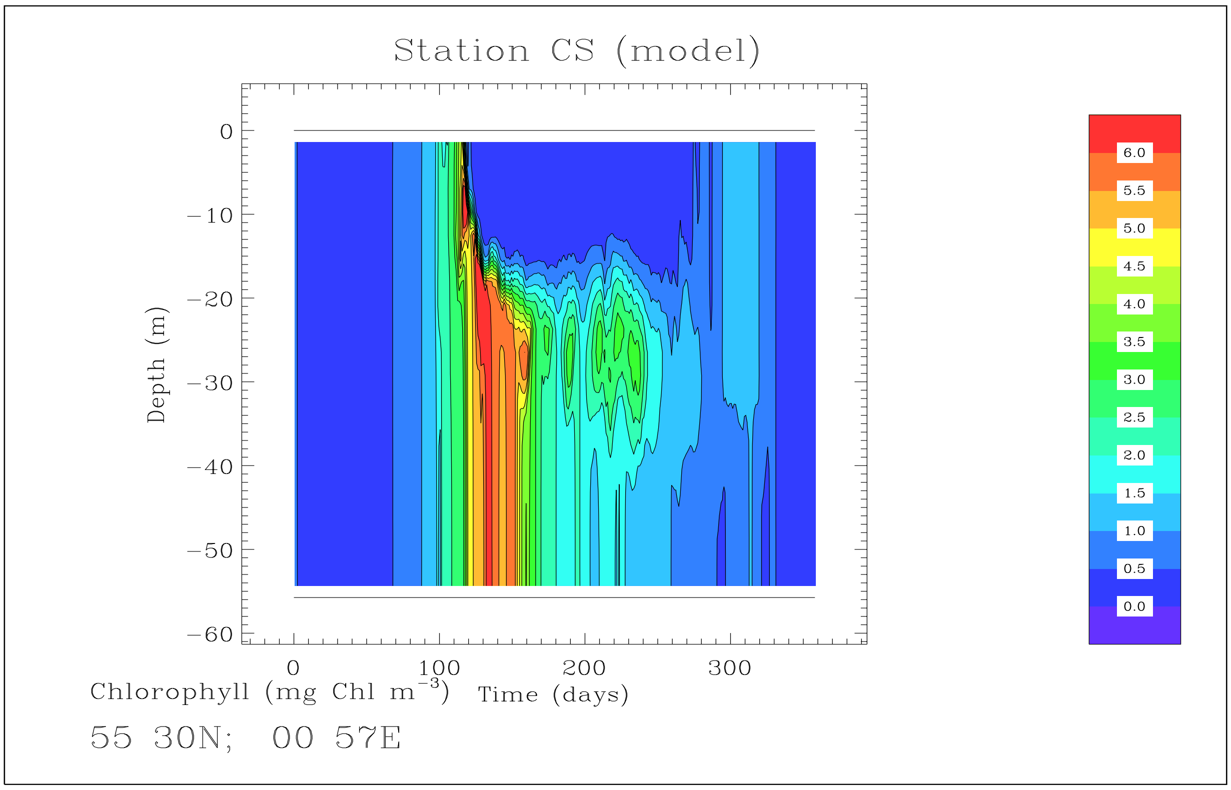


Figure 2.7.16: continued.



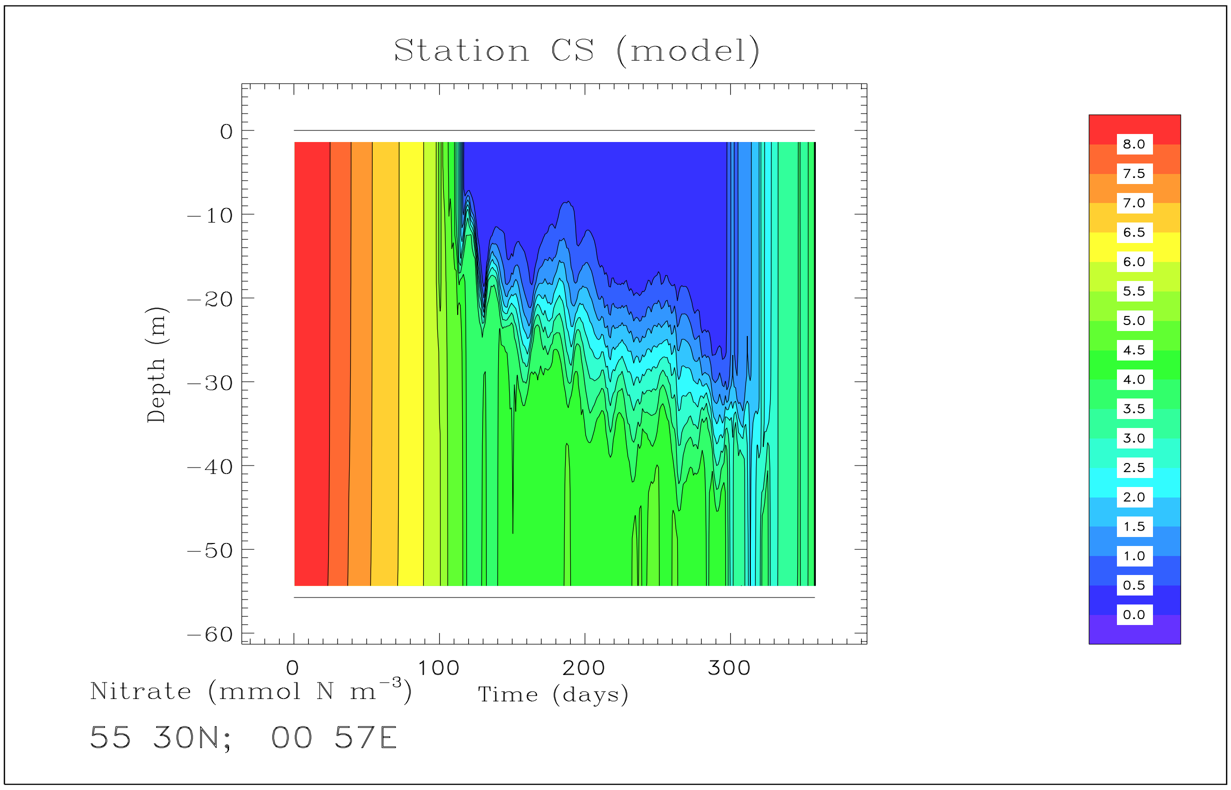
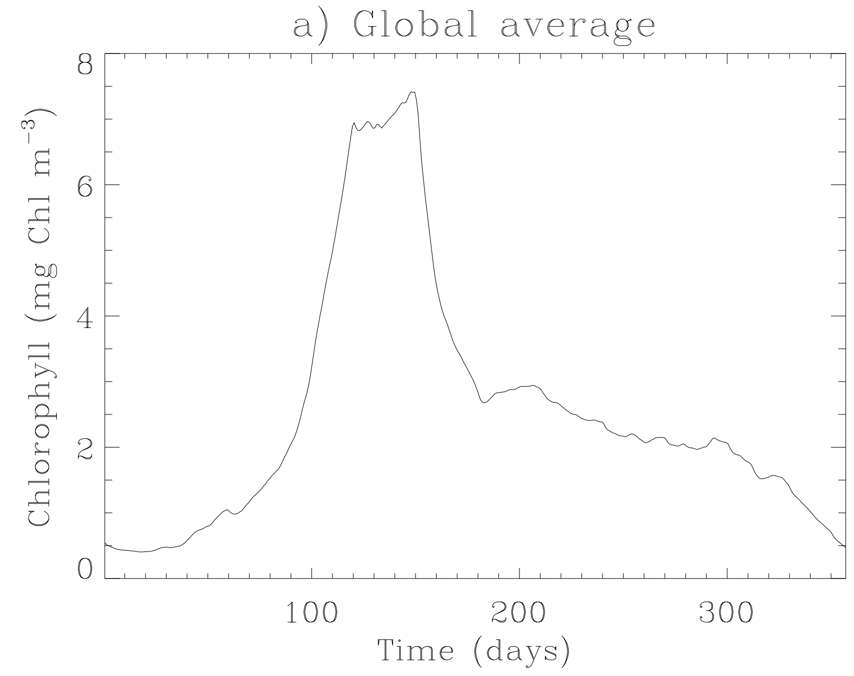


Figure 2.7.17: Depth-time contour plots of chlorophyll (mg Chl m^{-3}) and nitrate (mmol N m^{-3}) at station CS.



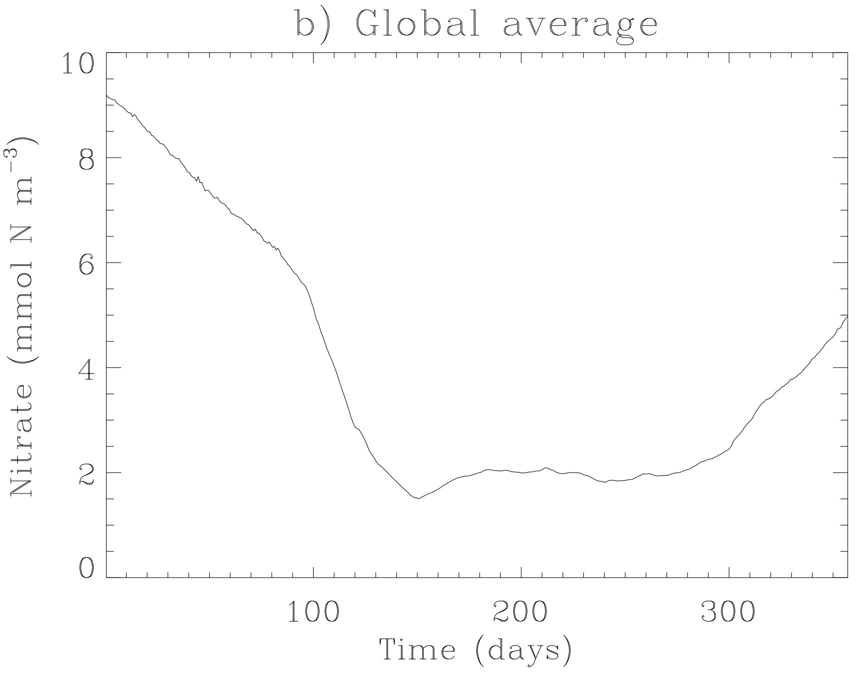


Figure 2.7.18: Time series of the globally averaged chlorophyll (a) and nitrate (b) concentrations.

b) limitations of the simulation

From May until the end of the simulation unusually high near surface nutrient concentrations were simulated along the western boundary and eastern section of the northern boundary close to Denmark. These gave rise to very high chlorophyll concentrations close to the model boundaries which appear to be unrealistic and suggest an error in the implementation of the nitrate and/or ammonium boundary condition. As the model boundaries lie well beyond the extent of the North Sea Project region, valid comparison of the central part of the model domain with the data can still be completed.

Realistic nutrient supply to the water column is critical for the correct simulation of microplankton biomass and chlorophyll. Following a years simulation the near surface ammonium concentration returned close to its initial condition; nitrate concentrations however were depleted indicating a net loss of nitrogen to the sediment. In this version of the COHERENS model the water column and sediment are not fully coupled, and the benthic flux of nitrogen to the water column is formulated as a relaxation expression to a mean bottom-water concentration. A relatively simple improvement would be to allow the benthic flux to relax to the initial nitrate field which varied spatially. A further improvement would be to implement a spatially variable initial ammonium field and a similar benthic flux formulation for ammonium.

The realistic simulation of microplankton growth depends also on the penetration of PAR, the attenuation of which is dominated by iSPM in the southern North Sea. The iSPM initial field (derived from Vos and Schuttelaar 1995, 1997) was poorly resolved in the approaches to the English Channel and north of 55^0N and could be improved by considering North Sea iSPM maps which cover a wider area (van Raaphorst et al. 1998).

Part III

Model Description

Chapter 1

Physical Model

1.1 Basic hydrodynamic equations

The program allows to formulate the model equations either in Cartesian coordinates (x_1, x_2, x_3) or in spherical coordinates (λ, ϕ, x_3), where the x_3-axis is directed upwards along the vertical. The Cartesian system uses the f-plane approximation (uniform Coriolis frequency) so that the (x_1, x_2)-axes can be oriented arbitrarily in the horizontal plane. In the spherical system λ and ϕ represent respectively the longitude (positive in the eastern, negative in the western hemisphere) and the latitude (positive in the northern, negative in the southern hemisphere). The vertical coordinate is chosen such that the surface $x_3 = 0$ corresponds to the mean sea water level. The equations of the free surface and sea bottom then take the form

$$x_3 = \zeta(x_1, x_2, t) \quad \text{or} \quad x_3 = \zeta(\lambda, \phi, t) \tag{3.1.1}$$

$$x_3 = -h(x_1, x_2) \quad \text{or} \quad x_3 = -h(\lambda, \phi) \tag{3.1.2}$$

where ζ is the sea surface elevation and h the mean water depth so that the total water depth H is given by $H = h + \zeta$.

The hydrodynamic part of the model uses the following basic equations:

- the momentum equations using the Boussinesq approximation and the assumption of vertical hydrostatic equilibrium
- the continuity equation
- the equations of temperature and salinity

The equations of momentum and continuity are solved numerically using the mode-splitting technique described in Chapter III-4. The previous system of three-dimensional equations needs, therefore, to be supplemented with an additional two-dimensional set consisting of the depth-integrated forms of the horizontal momentum and continuity equations. The full set of equations are described in the following two subsections for the Cartesian and the spherical formulation.

1.1.1 Cartesian coordinates

a) non-transformed equations

The basic equations for the three-dimensional mode are:

$$\begin{aligned}&\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x_1}+v\frac{\partial u}{\partial x_2}+w\frac{\partial u}{\partial x_3}-fv\\&\quad=-\frac{1}{\rho_0}\frac{\partial p}{\partial x_1}+\frac{\partial}{\partial x_3}\Big(\nu_T\frac{\partial u}{\partial x_3}\Big)+\frac{\partial}{\partial x_1}\tau_{11}+\frac{\partial}{\partial x_2}\tau_{21}\end{aligned}\tag{3.1.3}$$

$$\begin{aligned}&\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x_1}+v\frac{\partial v}{\partial x_2}+w\frac{\partial v}{\partial x_3}+fu\\&\quad=-\frac{1}{\rho_0}\frac{\partial p}{\partial x_2}+\frac{\partial}{\partial x_3}\Big(\nu_T\frac{\partial v}{\partial x_3}\Big)+\frac{\partial}{\partial x_1}\tau_{12}+\frac{\partial}{\partial x_2}\tau_{22}\end{aligned}\tag{3.1.4}$$

$$\frac{\partial p}{\partial x_3}=-\rho g\tag{3.1.5}$$

$$\frac{\partial u}{\partial x_1}+\frac{\partial v}{\partial x_2}+\frac{\partial w}{\partial x_3}=0\tag{3.1.6}$$

$$\begin{aligned}\frac{\partial T}{\partial t}&+u\frac{\partial T}{\partial x_1}+v\frac{\partial T}{\partial x_2}+w\frac{\partial T}{\partial x_3}\\&=\frac{1}{\rho_0c_p}\frac{\partial I}{\partial x_3}+\frac{\partial}{\partial x_3}\Big(\lambda_T\frac{\partial T}{\partial x_3}\Big)+\frac{\partial}{\partial x_1}\Big(\lambda_H\frac{\partial T}{\partial x_1}\Big)+\frac{\partial}{\partial x_2}\Big(\lambda_H\frac{\partial T}{\partial x_2}\Big)\end{aligned}\tag{3.1.7}$$

$$\begin{aligned}\frac{\partial S}{\partial t}&+u\frac{\partial S}{\partial x_1}+v\frac{\partial S}{\partial x_2}+w\frac{\partial S}{\partial x_3}\\&=\frac{\partial}{\partial x_3}\Big(\lambda_T\frac{\partial S}{\partial x_3}\Big)+\frac{\partial}{\partial x_1}\Big(\lambda_H\frac{\partial S}{\partial x_1}\Big)+\frac{\partial}{\partial x_2}\Big(\lambda_H\frac{\partial S}{\partial x_2}\Big)\end{aligned}\tag{3.1.8}$$

where (u,v,w) are the components of the current, T denotes the temperature, S the salinity, $f=2\Omega\sin\phi$ the Coriolis frequency, $\Omega=2\pi/86164$ rad/s the rotation frequency of the Earth, g the acceleration of gravity, p the pressure, ν_T and λ_T the vertical eddy viscosity and diffusion coefficients, λ_H the horizontal diffusion coefficient for salinity and temperature, ρ the density, ρ_0 a reference density, c_p the specific heat of seawater at constant pressure and $I(x_1,x_2,x_3,t)$ solar irradiance. The horizontal components of the stress tensor are defined by

$$\tau_{11}=2\nu_H\frac{\partial u}{\partial x_1}\tag{3.1.9}$$

$$\tau_{21}=\tau_{12}=\nu_H\Big(\frac{\partial u}{\partial x_2}+\frac{\partial v}{\partial x_1}\Big)\tag{3.1.10}$$

$$\tau_{22}=2\nu_H\frac{\partial v}{\partial x_2}\tag{3.1.11}$$

where ν_H is the horizontal diffusion coefficient for momentum.

The pressure can be eliminated from (3.1.3)–(3.1.5) in the usual way by writing p as the sum of an equilibrium and a perturbed part, i.e.

$$p = p_0 + p' = p_0 + \rho_0 q_d \tag{3.1.12}$$

The equilibrium pressure is obtained by solving

$$\frac{\partial p_0}{\partial x_3} = -\rho_0 g \tag{3.1.13}$$

with the boundary condition that p_0 equals the atmospheric pressure P_a at the surface. This gives

$$p_0 = \rho_0 g(\zeta - x_3) + P_a \tag{3.1.14}$$

so that

$$\frac{1}{\rho_0}\frac{\partial p}{\partial x_i} = g\frac{\partial \zeta}{\partial x_i} + \frac{1}{\rho_0}\frac{\partial P_a}{\partial x_i} + \frac{\partial q_d}{\partial x_i} \tag{3.1.15}$$

for $i = 1, 2$. The first two terms on the right hand side of (3.1.15) represent the barotropic, the third term the baroclinic part of the horizontal pressure gradient. The latter is obtained after substituting (3.1.12)–(3.1.14) into the condition for vertical hydrostatic equilibrium (3.1.5):

$$\frac{\partial q_d}{\partial x_3} = -g(\frac{\rho - \rho_0}{\rho_0}) = b \tag{3.1.16}$$

where b denotes the buoyancy. Integrating (3.1.16) using $q_d = 0$ at the surface, one has:

$$q_d = -\int_{x_3}^{\zeta} b\, dx_3 \tag{3.1.17}$$

The following additional comments can be made:

- When a Cartesian grid is selected, the f-plane approximation is assumed using a spatially uniform Coriolis frequency. If a latitudinal dependence of the Coriolis frequency is required, a spherical grid must be selected by the user.

- The vertical diffusion terms appearing on the right hand side of (3.1.3), (3.1.4), (3.1.7) and (3.1.8) have been parameterised by making the usual “downgradient diffusion” assumption which means that the vertical fluxes of momentum, heat and salinity are proportional to their vertical gradients but with opposite signs. Hence,

$$-(\langle u'w'\rangle, \langle v'w'\rangle) = \nu_T(\frac{\partial u}{\partial x_3}, \frac{\partial v}{\partial x_3})\ ,\ -(\langle w'T'\rangle, \langle w'S'\rangle) = \lambda_T(\frac{\partial T}{\partial x_3}, \frac{\partial S}{\partial x_3}) \tag{3.1.18}$$

  where a $'$ denotes a turbulent fluctuation and $\langle\ \rangle$ an ensemble average. The vertical eddy viscosity and diffusion coefficients are evaluated with a turbulence closure scheme. Several schemes are implemented in the program and are described in Section III-1.2. Note that double diffusion effects are not included so that vertical mixing of heat and salinity is parameterised using the same diffusion coefficient λ_T.

- The horizontal diffusion coefficients ν_H and λ_H represent horizontal subgrid-scale processes. Their explicit forms are given in Section III-1.3.
- The first term on the right hand side of the temperature equation (3.1.7) represents the absorption of solar heat within the water column. This is further discussed in Section III-1.4.
- The density, used in the evaluation of the buoyancy in (3.1.16) is determined with the aid of an equation of state which is discussed in Section III-1.5.

b) σ-transformed equations

The numerical solutions of the model equations are greatly simplified by introducing a new vertical coordinate which transforms both the surface and the bottom into coordinate surfaces (Phillips, 1957). The following coordinate transformation is applied

$$\Big(\tilde{t},\tilde{x}_1,\tilde{x}_2,\tilde{x}_3\Big)=\Big(t,x_1,x_2,Lf(\sigma)\Big) \tag{3.1.19}$$

where

$$\sigma=\frac{x_3+h}{\zeta+h}=\frac{x_3+h}{H} \tag{3.1.20}$$

is the commonly used σ-coordinate varying between 0 at the bottom and 1 at the surface. Taking $f(0)=0$ and $f(1)=1$ the equation of the bottom takes the simple form $\tilde{x}_3=0$ while the moving surface transforms into $\tilde{x}_3=L$. This is further illustrated in Figure 3.1.1. The length scale L will be determined in Chapter III-4 as part of the numerical discretisation. The Jacobian of the transformation

$$J=\frac{\partial x_3}{\partial\tilde{x}_3}=H/(L\frac{df}{d\sigma}) \tag{3.1.21}$$

represents the ratio of a unit length in the physical to a unit length in the transformed space. A complete derivation of the transformed equations will not be presented here. A more detailed discussion can be found in Deleersnijder and Ruddick (1992).

The equations take a simpler form by defining a new vertical velocity

$$\tilde{w}=\frac{\partial\tilde{x}_3}{\partial t}+u\frac{\partial\tilde{x}_3}{\partial x_1}+v\frac{\partial\tilde{x}_3}{\partial x_2}+w\frac{\partial\tilde{x}_3}{\partial x_3} \tag{3.1.22}$$

Multiplying (3.1.22) by J and using (3.1.19)–(3.1.21) the "old" vertical velocity can then be written as

$$w=J\tilde{w}+\sigma\frac{\partial\zeta}{\partial t}+u\Big(\sigma\frac{\partial\zeta}{\partial\tilde{x}_1}-(1-\sigma)\frac{\partial h}{\partial\tilde{x}_1}\Big)+v\Big(\sigma\frac{\partial\zeta}{\partial\tilde{x}_2}-(1-\sigma)\frac{\partial h}{\partial\tilde{x}_2}\Big) \tag{3.1.23}$$

The first term on the right of (3.1.23), further denoted by the "transformed" vertical velocity, represents the intrinsic vertical upwelling or downwelling current normal to the

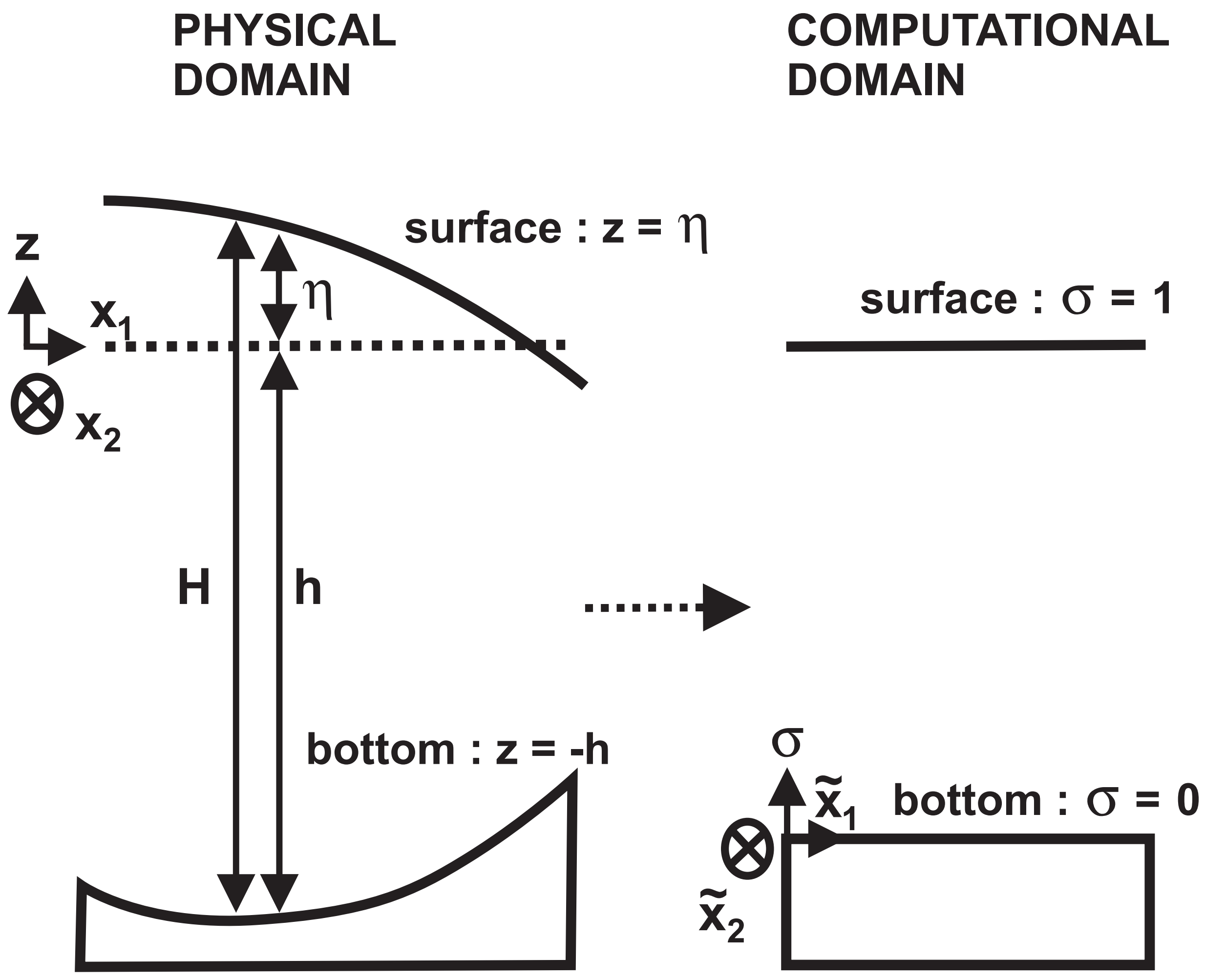


Figure 3.1.1: The σ-coordinate transformation in the vertical.

iso-σ surfaces while the other terms are induced by the variations in space and time of the shapes of the sea surface and bottom (see e.g. Deleersnijder, 1989).

The transformed versions of the equations of continuity, horizontal momentum, hydrostatic equilibrium, temperature and salinity are given by

$$\frac{1}{J}\frac{\partial J}{\partial \tilde{t}} + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(Ju) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(Jv) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}) = 0 \tag{3.1.24}$$

$$\begin{aligned}&\frac{1}{J}\frac{\partial}{\partial \tilde{t}}(Ju) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(Ju^2) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(Jvu) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}u) - fv \\ &\quad = -g\frac{\partial \zeta}{\partial \tilde{x}_1} - \frac{1}{\rho_0}\frac{\partial P_a}{\partial \tilde{x}_1} + Q_1 + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial u}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(J\tau_{11}) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(J\tau_{21})\end{aligned} \tag{3.1.25}$$

$$\begin{aligned}&\frac{1}{J}\frac{\partial}{\partial \tilde{t}}(Jv) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(Juv) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(Jv^2) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}v) + fu \\ &\quad = -g\frac{\partial \zeta}{\partial \tilde{x}_2} - \frac{1}{\rho_0}\frac{\partial P_a}{\partial \tilde{x}_2} + Q_2 + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial v}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(J\tau_{12}) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(J\tau_{22})\end{aligned} \tag{3.1.26}$$

$$\frac{1}{J}\frac{\partial q_d}{\partial \tilde{x}_3} = b \tag{3.1.27}$$

$$\begin{aligned}&\frac{1}{J}\frac{\partial}{\partial \tilde{t}}(JT) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(JuT) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(JvT) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}T) \\ &\quad = \frac{1}{J\rho_0 c_p}\frac{\partial I}{\partial \tilde{x}_3} + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial T}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}\Big(J\lambda_H\frac{\partial T}{\partial \tilde{x}_1}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}\Big(J\lambda_H\frac{\partial T}{\partial \tilde{x}_2}\Big)\end{aligned} \tag{3.1.28}$$

$$\begin{aligned}&\frac{1}{J}\frac{\partial}{\partial \tilde{t}}(JS) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}(JuS) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}(JvS) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}S) \\ &\quad = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial S}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_1}\Big(J\lambda_H\frac{\partial S}{\partial \tilde{x}_1}\Big) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_2}\Big(J\lambda_H\frac{\partial S}{\partial \tilde{x}_2}\Big)\end{aligned} \tag{3.1.29}$$

The components of the baroclinic pressure gradient in the transformed coordinate system take the form

$$\begin{aligned}Q_i &= -\frac{1}{J}\frac{\partial}{\partial \tilde{x}_i}(Jq_d) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(q_d\frac{\partial x_3}{\partial \tilde{x}_i}\Big) \\ &= -\frac{1}{J}\frac{\partial}{\partial \tilde{x}_i}(Jq_d) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(q_d(\sigma\frac{\partial H}{\partial \tilde{x}_i} - \frac{\partial h}{\partial \tilde{x}_i})\Big)\end{aligned} \tag{3.1.30}$$

The components of the horizontal stress are still defined by (3.1.9)–(3.1.11) with (x_1, x_2) replaced by $(\tilde{x}_1, \tilde{x}_2)$.

The following points should be noted:

- The horizontal diffusion terms have been simplified according to the recommendations of Mellor and Blumberg (1985). This means that a series of terms have been omitted to avoid unrealistic diffusion across iso-σ surfaces which may become comparable or larger than the diffusion induced by vertical mixing.

- The conservative form (3.1.30) used for the baroclinic pressure gradient ensures that no circulation is generated of vertically integrated momentum along contours of constant depth (Arakawa and Suarez, 1983).

- As discussed by several authors (e.g. Janjić, 1977; Arakawa and Suarez, 1983; Mesinger and Janjić, 1985; Haney, 1991) a major drawback of the σ-coordinate is that even small truncation errors in the numerical computation of the pressure gradient near a steep bottom can generate spurious geostrophic flows. Gary (1973) and Haney (1991) showed that the removal of a reference state, analogous to the equilibrium pressure gradient in the present formulation, can reduce the truncation error. A second problem is that the numerical scheme only converges if the condition for "hydrostatic consistency" is satisfied (Mesinger and Janjić, 1985). This criterion is given by

$$\frac{1-\sigma}{H}\Big(\Big|\frac{\partial H}{\partial x_1}\Big|\Delta x_1\,,\Big|\frac{\partial H}{\partial x_2}\Big|\Delta x_2\Big) < \Delta\sigma \tag{3.1.31}$$

  where $\Delta\sigma$ is the vertical spacing in the σ-coordinate system. Adapted numerical schemes have been proposed (e.g. Stelling and Van Kester, 1994; Mc Calpin, 1994) but have not yet been implemented in the current version of the program.

c) depth-integrated equations

The equation set for the two-dimensional mode consists of an equation for the surface elevation ζ and two equations for the vertically integrated currents, defined by

$$(\overline{U},\overline{V}) = \int_{-h}^{\zeta}(u,v)\,dx_3 = \int_0^L(u,v)J\,d\tilde{x}_3 \tag{3.1.32}$$

Integrating the equations of continuity and horizontal momentum (3.1.24)–(3.1.26) over the vertical one obtains

$$\frac{\partial\zeta}{\partial\tilde{t}} + \frac{\partial\overline{U}}{\partial\tilde{x}_1} + \frac{\partial\overline{V}}{\partial\tilde{x}_2} = 0 \tag{3.1.33}$$

$$\begin{aligned}\frac{\partial\overline{U}}{\partial\tilde{t}} &+ \frac{\partial}{\partial\tilde{x}_1}\Big(\frac{\overline{U}^2}{H}\Big) + \frac{\partial}{\partial\tilde{x}_2}\Big(\frac{\overline{V}\,\overline{U}}{H}\Big) - f\overline{V}\\ &= -gH\frac{\partial\zeta}{\partial\tilde{x}_1} - \frac{H}{\rho_0}\frac{\partial P_a}{\partial\tilde{x}_1} + \overline{Q}_1 + \frac{1}{\rho_0}(\tau_{s1}-\tau_{b1})\\ &\quad + \frac{\partial}{\partial\tilde{x}_1}\overline{\tau_{11}} + \frac{\partial}{\partial\tilde{x}_2}\overline{\tau_{21}} - \overline{A}_1^h + \overline{D}_1^h\end{aligned} \tag{3.1.34}$$

$$\begin{aligned}\frac{\partial\overline{V}}{\partial\tilde{t}} &+ \frac{\partial}{\partial\tilde{x}_1}\Big(\frac{\overline{U}\,\overline{V}}{H}\Big)+\frac{\partial}{\partial\tilde{x}_2}\Big(\frac{\overline{V}^2}{H}\Big)+f\overline{U}\\ &= -gH\frac{\partial\zeta}{\partial\tilde{x}_2}-\frac{H}{\rho_0}\frac{\partial P_a}{\partial\tilde{x}_2}+\overline{Q}_2+\frac{1}{\rho_0}(\tau_{s2}-\tau_{b2})\\ &\quad+\frac{\partial}{\partial\tilde{x}_1}\overline{\tau_{12}}+\frac{\partial}{\partial\tilde{x}_2}\overline{\tau_{22}}-\overline{A}_2^h+\overline{D}_2^h \qquad (3.1.35)\end{aligned}$$

where (τ_{s1},τ_{s2}) and (τ_{b1},τ_{b2}) are the components of respectively the surface and the bottom stress and use is made of the condition that $J\tilde{w}$ vanishes at the surface and bottom. The depth-integrated baroclinic pressure terms on the right hand side of (3.1.34)–(3.1.35) are given by

$$(\overline{Q}_1,\overline{Q}_2)=\int_0^L J(Q_1,Q_2)\,d\tilde{x}_3 \qquad (3.1.36)$$

The stress components have the same form as in (3.1.9)–(3.1.11) with (u,v) replaced by the depth-averaged current $(\overline{U}/H,\overline{V}/H)$ and ν_H by $\overline{\nu_H}$, i.e.

$$\overline{\tau_{11}}=2\overline{\nu_H}\frac{\partial}{\partial\tilde{x}_1}\Big(\frac{\overline{U}}{H}\Big) \qquad (3.1.37)$$

$$\overline{\tau_{21}}=\overline{\tau_{12}}=\overline{\nu_H}\Big[\frac{\partial}{\partial\tilde{x}_2}\Big(\frac{\overline{U}}{H}\Big)+\frac{\partial}{\partial\tilde{x}_1}\Big(\frac{\overline{V}}{H}\Big)\Big] \qquad (3.1.38)$$

$$\overline{\tau_{22}}=2\overline{\nu_H}\frac{\partial}{\partial\tilde{x}_2}\Big(\frac{\overline{V}}{H}\Big) \qquad (3.1.39)$$

where

$$\overline{\nu_H}=\int_0^L\nu_H J d\tilde{x}_3 \qquad (3.1.40)$$

represents the horizontal diffusion coefficient integrated over the water depth.

The last two terms on the right of (3.1.34)–(3.1.35) are the depth integrals of the horizontal advection and diffusion terms with $(u,\,v)$ replaced by

$$(u',v')=(u-\overline{U}/H,v-\overline{V}/H) \qquad (3.1.41)$$

which represent the deviation of the horizontal current with respect to its depth-averaged value. Their explicit forms are

$$\overline{A}_1^h=\int_0^L\Big(\frac{\partial}{\partial\tilde{x}_1}(Ju'^2)+\frac{\partial}{\partial\tilde{x}_2}(Jv'u')\Big)\,d\tilde{x}_3 \qquad (3.1.42)$$

$$\overline{A}_2^h=\int_0^L\Big(\frac{\partial}{\partial\tilde{x}_1}(Ju'v')+\frac{\partial}{\partial\tilde{x}_2}(Jv'^2)\Big)\,d\tilde{x}_3 \qquad (3.1.43)$$

$$\overline{D}_1^h=\int_0^L\Big[\frac{\partial}{\partial\tilde{x}_1}\Big(2\nu_H J\frac{\partial u'}{\partial\tilde{x}_1}\Big)+\frac{\partial}{\partial\tilde{x}_2}\Big(\nu_H J(\frac{\partial u'}{\partial\tilde{x}_2}+\frac{\partial v'}{\partial\tilde{x}_1})\Big)\Big]d\tilde{x}_3 \qquad (3.1.44)$$

$$\overline{D}_2^h=\int_0^L\Big[\frac{\partial}{\partial\tilde{x}_1}\Big(\nu_H J(\frac{\partial u'}{\partial\tilde{x}_2}+\frac{\partial v'}{\partial\tilde{x}_1})\Big)+\frac{\partial}{\partial\tilde{x}_2}\Big(2\nu_H J\frac{\partial v'}{\partial\tilde{x}_2}\Big)\Big]d\tilde{x}_3 \qquad (3.1.45)$$

The transformed vertical velocity can in principle be determined by integrating the continuity equation (3.1.24). A more useful relation can be derived by first multiplying (3.1.33) with the horizontally uniform factor J/H and subtracting the result from J times equation (3.1.24). Using (3.1.21) one finds that

$$\frac{\partial}{\partial\tilde{x}_1}\Big(J(u-\overline{U}/H)\Big)+\frac{\partial}{\partial\tilde{x}_2}\Big(J(v-\overline{V}/H)\Big)+\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w})=0 \qquad (3.1.46)$$

The evaluation of the "physical" velocity w is longer needed in the program. Its value can be determined *a posteriori* using the expression derived in Deleersnijder and Ruddick (1992):

$$w=\frac{1}{J}\Big(\frac{\partial}{\partial\tilde{t}}(Jx_3)+\frac{\partial}{\partial\tilde{x}_1}(Jux_3)+\frac{\partial}{\partial\tilde{x}_2}(Jvx_3)+\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}x_3)\Big) \qquad (3.1.47)$$

This form is equivalent to the more physical expression (3.1.23) but can more easily be implemented numerically.

1.1.2 Spherical coordinates

a) three-dimensional equations

The main difference between the Cartesian and spherical formulation is that

$$\begin{array}{llll}\dfrac{\partial}{\partial x_1} & \text{is replaced by} & \dfrac{\partial}{\partial\lambda} & \text{and}\\[2ex] \dfrac{\partial}{\partial x_2} & \text{is replaced either by} & \dfrac{1}{R}\dfrac{\partial}{\partial\phi} & \text{or}\quad \dfrac{1}{R\cos\phi}\dfrac{\partial}{\partial\phi}\cos\phi\end{array} \qquad (3.1.48)$$

where R denotes the radius of the Earth and the last formulation applies in the derivatives with respect to ϕ of all advective and diffusion fluxes. The σ-transformation can be implemented without any additional complexities. The transformed equations of continuity, horizontal momentum, hydrostatic equilibrium, temperature and salinity now take the form (where for simplicity the ˜ has been omitted to denote the transformed coordinates $(\tilde{t},\tilde{\lambda},\tilde{\phi})$):

$$\frac{1}{J}\frac{\partial J}{\partial t}+\frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(Ju)+\frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, v)+\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w})=0 \qquad (3.1.49)$$

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(Ju) &+ \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(Ju^2)+\frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, vu)+\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}u)\\ &- \frac{\tan\phi}{R}uv-2\Omega\sin\phi\, v\\ &= -\frac{1}{R\cos\phi}\Big(g\frac{\partial\zeta}{\partial\lambda}+\frac{1}{\rho_0}\frac{\partial P_a}{\partial\lambda}\Big)+Q_\lambda+\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial u}{\partial\tilde{x}_3}\Big)+\\ &\quad \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(J\tau_{\lambda\lambda})+\frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\,\tau_{\phi\lambda})-\frac{\tan\phi}{R}\tau_{\phi\lambda}\end{aligned} \qquad (3.1.50)$$

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(Jv) &+ \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(Juv) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, v^2) + \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}v)\\ &+ \frac{\tan\phi}{R}u^2 + 2\Omega\sin\phi\, u\\ &= -\frac{1}{R}\Big(g\frac{\partial\zeta}{\partial\phi} + \frac{1}{\rho_0}\frac{\partial P_a}{\partial\phi}\Big) + Q_\phi + \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial v}{\partial\tilde{x}_3}\Big) +\\ &\quad \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(J\tau_{\lambda\phi}) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\,\tau_{\phi\phi}) + \frac{\tan\phi}{R}\tau_{\lambda\lambda}\end{aligned} \tag{3.1.51}$$

$$\frac{1}{J}\frac{\partial q_d}{\partial\tilde{x}_3} = b \tag{3.1.52}$$

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(JT) &+ \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(JuT) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, vT) + \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}T)\\ &= \frac{1}{J\rho_0 c_p}\frac{\partial I}{\partial\tilde{x}_3} + \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial T}{\partial\tilde{x}_3}\Big) +\\ &\quad \frac{1}{JR^2\cos^2\phi}\frac{\partial}{\partial\lambda}\Big(J\lambda_H\frac{\partial T}{\partial\lambda}\Big) + \frac{1}{JR^2\cos\phi}\frac{\partial}{\partial\phi}\Big(J\lambda_H\cos\phi\frac{\partial T}{\partial\phi}\Big)\end{aligned} \tag{3.1.53}$$

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(JS) &+ \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(JuS) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, vS) + \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}S)\\ &= \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial S}{\partial\tilde{x}_3}\Big) +\\ &\quad \frac{1}{JR^2\cos^2\phi}\frac{\partial}{\partial\lambda}\Big(J\lambda_H\frac{\partial S}{\partial\lambda}\Big) + \frac{1}{JR^2\cos\phi}\frac{\partial}{\partial\phi}\Big(J\lambda_H\cos\phi\frac{\partial S}{\partial\phi}\Big)\end{aligned} \tag{3.1.54}$$

The components of the baroclinic pressure gradient and the horizontal shear stress are given by

$$Q_\lambda = -\frac{1}{JR\cos\phi}\Big[\frac{\partial}{\partial\lambda}(Jq_d) - \frac{\partial}{\partial\tilde{x}_3}\Big(q_d(\sigma\frac{\partial H}{\partial\lambda} - \frac{\partial h}{\partial\lambda})\Big)\Big] \tag{3.1.55}$$

$$Q_\phi = -\frac{1}{JR}\Big[\frac{\partial}{\partial\phi}(Jq_d) - \frac{\partial}{\partial\tilde{x}_3}\Big(q_d(\sigma\frac{\partial H}{\partial\phi} - \frac{\partial h}{\partial\phi})\Big)\Big] \tag{3.1.56}$$

$$\tau_{\lambda\lambda} = 2\nu_H\Big(\frac{1}{R\cos\phi}\frac{\partial u}{\partial\lambda} - \frac{\tan\phi}{R}v\Big) \tag{3.1.57}$$

$$\tau_{\phi\lambda} = \tau_{\lambda\phi} = \nu_H\Big(\frac{1}{R}\frac{\partial u}{\partial\phi} + \frac{\tan\phi}{R}u + \frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}\Big) \tag{3.1.58}$$

$$\tau_{\phi\phi} = 2\nu_H\frac{1}{R}\frac{\partial v}{\partial\phi} \tag{3.1.59}$$

Expressions (3.1.57)–(3.1.59) are derived from the more general relations derived in Blumberg and Herring (1987).

b) depth-integrated equations

The equations are similar to the ones derived in the Cartesian system but are given here for completeness. The equations for the surface elevation and the depth-integrated currents $(\overline{U},\overline{V})$ are now

$$\frac{\partial\zeta}{\partial t}+\frac{1}{R\cos\phi}\frac{\partial\overline{U}}{\partial\lambda}+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{V})=0 \tag{3.1.60}$$

$$\begin{aligned}\frac{\partial\overline{U}}{\partial t} &+ \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\frac{\overline{U}^2}{H}\Big)+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\cos\phi\frac{\overline{V}\,\overline{U}}{H}\Big)-\frac{\tan\phi}{R}\frac{\overline{U}\,\overline{V}}{H}-2\Omega\sin\phi\,\overline{V}\\ &= -\frac{H}{R\cos\phi}\Big(g\frac{\partial\zeta}{\partial\lambda}+\frac{1}{\rho_0}\frac{\partial P_a}{\partial\lambda}\Big)+\overline{Q}_\lambda+\frac{1}{\rho_0}(\tau_{s\lambda}-\tau_{b\lambda})+\\ &\quad\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\overline{\tau_{\lambda\lambda}}+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{\tau_{\phi\lambda}})-\frac{\tan\phi}{R}\overline{\tau_{\phi\lambda}}-\overline{A}^h_\lambda+\overline{D}^h_\lambda\end{aligned} \tag{3.1.61}$$

$$\begin{aligned}\frac{\partial\overline{V}}{\partial t} &+ \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\frac{\overline{U}\,\overline{V}}{H}\Big)+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\cos\phi\frac{\overline{V}^2}{H}\Big)+\frac{\tan\phi}{R}\frac{\overline{U}^2}{H}+2\Omega\sin\phi\,\overline{U}\\ &= -\frac{H}{R}\Big(g\frac{\partial\zeta}{\partial\phi}+\frac{1}{\rho_0}\frac{\partial P_a}{\partial\phi}\Big)+\overline{Q}_\phi+\frac{1}{\rho_0}(\tau_{s\phi}-\tau_{b\phi})+\\ &\quad\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\overline{\tau_{\lambda\phi}}+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{\tau_{\phi\phi}})+\frac{\tan\phi}{R}\overline{\tau_{\lambda\lambda}}-\overline{A}^h_\phi+\overline{D}^h_\phi\end{aligned} \tag{3.1.62}$$

where

$$(\overline{Q}_\lambda,\overline{Q}_\phi)=\int_0^L J(Q_\lambda,Q_\phi)\,d\tilde{x}_3 \tag{3.1.63}$$

and the stress components $(\overline{\tau_{\lambda\lambda}},\overline{\tau_{\phi\lambda}}=\overline{\tau_{\lambda\phi}},\overline{\tau_{\phi\phi}})$ are obtained from (3.1.57)–(3.1.59) with (u,v) replaced by $(\overline{U}/H,\overline{V}/H)$ and ν_H by $\overline{\nu_H}$ defined through (3.1.40). The depth-integrated horizontal advection and diffusion terms now take the form

$$\overline{A}^h_\lambda=\int_0^L\Big(\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}(Ju'^2)+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\,v'u')-J\frac{\tan\phi}{R}u'v'\Big)\,d\tilde{x}_3 \tag{3.1.64}$$

$$\overline{A}^h_\phi=\int_0^L\Big(\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}(Ju'v')+\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\,v'^2)+J\frac{\tan\phi}{R}u'^2\Big)\,d\tilde{x}_3 \tag{3.1.65}$$

$$\begin{aligned}\overline{D}^h_\lambda=\int_0^L\ \Big[\ &\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(2\nu_H J\big(\frac{1}{R\cos\phi}\frac{\partial u'}{\partial\lambda}-\frac{\tan\phi}{R}v'\big)\Big)+\\ &\frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\nu_H J\cos\phi\,\big(\frac{1}{R}\frac{\partial u'}{\partial\phi}+\frac{\tan\phi}{R}u'+\frac{1}{R\cos\phi}\frac{\partial v'}{\partial\lambda}\big)\Big)-\\ &\frac{\tan\phi}{R}\nu_H J\Big(\frac{1}{R}\frac{\partial u'}{\partial\phi}+\frac{\tan\phi}{R}u'+\frac{1}{R\cos\phi}\frac{\partial v'}{\partial\lambda}\Big)\Big]d\tilde{x}_3\end{aligned} \tag{3.1.66}$$

$$\overline{D}^h_\phi = \int_0^L \Big[\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\nu_H J(\frac{1}{R}\frac{\partial u'}{\partial\phi} + \frac{\tan\phi}{R}u' + \frac{1}{R\cos\phi}\frac{\partial v'}{\partial\lambda})\Big) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(2\nu_H J\cos\phi\frac{1}{R}\frac{\partial v'}{\partial\phi}\Big) + 2\frac{\tan\phi}{R}\nu_H J\Big(\frac{1}{R\cos\phi}\frac{\partial u'}{\partial\lambda} - \frac{\tan\phi}{R}v'\Big)\Big] d\tilde{x}_3 \tag{3.1.67}$$

Equations (3.1.46) and (3.1.47) for the new and the old vertical velocity are now replaced by

$$\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(J(u - \overline{U}/H)\Big) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(J\cos\phi\,(v - \overline{V}/H)\Big) + \frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}) = 0 \tag{3.1.68}$$

and

$$w = \frac{1}{J}\Big(\frac{\partial}{\partial t}(Jx_3) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}(Jux_3) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, vx_3) + \frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}x_3)\Big) \tag{3.1.69}$$

1.1.3 General form of a scalar advection-diffusion equation

The equations of temperature and salinity represent scalar transport equations of the advection-diffusion type. A whole series of scalar quantities are defined in the program for which a similar transport equation must be solved. These quantities may represent temperature, salinity, turbulence variables, biological state variables, sediment or contaminant concentrations. The general form of a transport equation for a quantity ψ can be written as follows:

$$\Big(\mathcal{I} + \mathcal{A}_h + \mathcal{A}_v + \mathcal{A}_s - \mathcal{D}_v - \mathcal{D}_h\Big)\psi = \mathcal{P}(\psi) - \mathcal{S}(\psi) \tag{3.1.70}$$

The meaning of the different operators is explained below. For simplicity the ˜ has been omitted to denote the transformed horizontal coordinates and time.

- $\mathcal{I}$ is the time derivative operator defined by

$$\mathcal{I}(\psi) = \frac{1}{J}\frac{\partial}{\partial t}(J\psi) \tag{3.1.71}$$

- $\mathcal{A}_h$ is the horizontal advection operator defined by

$$\mathcal{A}_h(\psi) = \frac{1}{J}\frac{\partial}{\partial x_1}(Ju\psi) + \frac{1}{J}\frac{\partial}{\partial x_2}(Jv\psi) \tag{3.1.72}$$

  in Cartesian and

$$\mathcal{A}_h(\psi) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\lambda}(Ju\psi) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, v\psi) \tag{3.1.73}$$

  in spherical coordinates.

- $\mathcal{A}_v$ is the vertical advection operator given by

$$\mathcal{A}_v(\psi) = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}\psi) \tag{3.1.74}$$

- $\mathcal{A}_s$ is the vertical sinking operator

$$\mathcal{A}_s(\psi) = \frac{w_s^\psi}{J}\frac{\partial \psi}{\partial \tilde{x}_3} \tag{3.1.75}$$

  where w_s^ψ is a "non-physical" sinking (or swimming) rate specific for the quantity ψ. Note that w_s^ψ is negative (positive) in the case of sinking (swimming).

- $\mathcal{D}_v$ is the vertical diffusion operator defined by

$$\frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T^\psi}{J}\frac{\partial \psi}{\partial \tilde{x}_3}\Big) \tag{3.1.76}$$

  where λ_T^ψ is the vertical diffusion coefficient appropriate for the quantity ψ and down-gradient turbulent diffusion is assumed in analogy with (3.1.18).

- $\mathcal{D}_h$ is the horizontal diffusion operator given by

$$\mathcal{D}_h(\psi) = \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\lambda_H^\psi\frac{\partial \psi}{\partial x_1}\Big) + \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\lambda_H^\psi\frac{\partial \psi}{\partial x_2}\Big) \tag{3.1.77}$$

  in Cartesian and

$$\mathcal{D}_h(\psi) = \frac{1}{JR^2\cos^2\phi}\frac{\partial}{\partial \lambda}\Big(J\lambda_H^\psi\frac{\partial \psi}{\partial \lambda}\Big) + \frac{1}{JR^2\cos\phi}\frac{\partial}{\partial \phi}\Big(J\lambda_H^\psi\cos\phi\frac{\partial \psi}{\partial \phi}\Big) \tag{3.1.78}$$

  in spherical coordinates and where λ_H^ψ is the (uniform) horizontal diffusion coefficient related to the quantity ψ.

- $\mathcal{P}(\psi)$ represents all other positive source terms in the transport equation.

- $\mathcal{S}(\psi)$ represents all other negative sink terms in the transport equation.

For example, the temperature equation can be cast in the form (3.1.70) with

$$w_s^T = 0\ ,\ \lambda_T^T = \lambda_T\ ,\ \lambda_H^T = \lambda_H\ ,\ \mathcal{P}(T) = \frac{1}{J\rho_0 c_p}\frac{\partial I}{\partial \tilde{x}_3}\ ,\ \mathcal{S}(T) = 0 \tag{3.1.79}$$

1.1.4 One-dimensional form of the basic equations

An option is foreseen, selected by the model switch IGRDIM, to use the program as a point model in the vertical. In a 1-D application all horizontal gradients, except for the slope of the sea surface elevation and the horizontal pressure gradient, and vertical advection by the sinking velocity w_s^ψ are neglected in the momentum and scalar transport equations which are now given by

$$\frac{1}{J}\frac{\partial}{\partial t}(Ju) - fv = -g\frac{\partial \zeta}{\partial x_1} + Q_1 + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial u}{\partial \tilde{x}_3}\Big) \tag{3.1.80}$$

$$\frac{1}{J}\frac{\partial}{\partial t}(Jv) + fu = -g\frac{\partial \zeta}{\partial x_2} + Q_2 + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial v}{\partial \tilde{x}_3}\Big) \tag{3.1.81}$$

$$\frac{1}{J}\frac{\partial}{\partial t}(J\psi) + \frac{w_s^\psi}{J}\frac{\partial \psi}{\partial \tilde{x}_3} = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T^\psi}{J}\frac{\partial \psi}{\partial \tilde{x}_3}\Big) + \mathcal{P}(\psi) - \mathcal{S}(\psi) \tag{3.1.82}$$

The depth-integrated momentum and continuity equations are not solved. The sea surface elevation and the horizontal pressure gradient are determined by the following harmonic expansions

$$\zeta = \sum_{n=1}^{N_T} \zeta_n \cos(\omega_n t + \varphi_{n0} - \varphi_{ns}) \tag{3.1.83}$$

$$g\frac{\partial \zeta}{\partial x_1} - Q_1 = F_0 + \sum_{n=1}^{N_T} F_n \cos(\omega_n t + \varphi_{n0} - \varphi_{nx}) \tag{3.1.84}$$

$$g\frac{\partial \zeta}{\partial x_2} - Q_2 = G_0 + \sum_{n=1}^{N_T} G_n \cos(\omega_n t + \varphi_{n0} - \varphi_{ny}) \tag{3.1.85}$$

where ω_n are the frequencies of the tidal forcing in rad/s, φ_{n0} the initial value of the phase $\omega_n t + \varphi_{n0}$ in radians, ζ_n are the tidal amplitudes of the surface elevation in m, $(F_n,\ G_n)$ the amplitudes of the surface slope in m/s^2, $(F_0,\ G_0)$ the baroclinic pressure gradient in m/s^2 and $(\varphi_{ns},\ \varphi_{nx},\ \varphi_{ny})$ the tidal phases in radians.

1.1.5 Program switches

A series of switches have been implemented in the program allowing the user to select the type of model grid, enable/disable different processes, select the type of forcing, boundary condition or numerical scheme, or to switch on/off different program units (biology, sediments, contaminants, particle method). The implementation in the program of the hydrodynamic equations, described in this section, depends on the setting of a number of switches, listed below in Table 3.1.1. A full description of all program switches is discussed in Section IV-2.2.1.

Table 3.1.1: Program switches for the hydrodynamics.

IGTRH	All model equations are solved on a Cartesian or a spherical grid if this switch is set to 0/1.
IGRDIM	The 3-D or 1-D version of the program is selected if this switch is set to respectively 3 or 1.
IOPT3	The solution of the 3-D horizontal momentum equation is disabled/enabled if this switch takes the value 0/1.
IOPT2	The solution of the 2-D equation set is disabled/enabled if this switch is set to 0/1.
IOPTHE	The solution of the temperature equation is disabled/enabled if this switch is 0 or has a positive value.
IOPTSA	The solution of the salinity equation is disabled/enabled if this switch is 0 or has a positive value.
IADVC	The evaluation of all horizontal and vertical advective terms in the 3-D and 2-D momentum equations is disabled/enabled if this switch is set to 0 or takes a positive value.
IADVS	The evaluation of the horizontal and vertical advective terms in the temperature and salinity equations is disabled/enabled when this switch is set to 0 or has a positive value.
IODIF	The evaluation of all horizontal diffusion terms in the 3-D and 2-D momentum equations and in the equations of temperature and salinity is disabled/enabled when this switch is 0 or takes a positive value.
IOPTD	The density is uniform in space and time or evaluated using an equation of state if this switch is 0 or takes a positive value.
IOPTK	Evaluation of vertical diffusion in the momentum equations or in the equations of temperature and salinity can be disabled if this switch is set to 0. In that case, other model parameters must be set to 0. This is further discussed in Sections III-1.2 and IV-2.2.

Note

The equations in the remainder of the manual are mostly given in Cartesian coordinates. Since all equations are formulated in the transformed coordinate system, the $\tilde{}$ is omitted (except for the vertical coordinate) to denote transformed coordinates. Unless stated otherwise their equivalents for the spherical system are obtained simply by replacing (x_1, x_2) with (λ, ϕ) and substituting (3.1.48) for the horizontal derivatives.

1.2 Turbulence schemes

One of the most intricate problems in oceanographic modelling is an adequate parameterisation of vertical exchange processes. In the present model they are represented through the eddy coefficients ν_T and λ_T. Values for these two parameters are to be provided by a turbulence scheme. The choice of an appropriate turbulence model not only affects the physical but also the biological, sediment and contaminant parts of the program, since the same vertical diffusion coefficient λ_T will be used for T, S, biological state variables, sediment and contaminant concentrations, and for the diffusion of Lagrangian particles[1]. A large variety of turbulence parameterisations with a substantial range of complexity have been proposed and validated in the literature. The selection of a suitable scheme is often a difficult task since it depends on the type of physical processes specific for the simulated area (e.g. tides, thermoclines, river fronts, ...), the vertical resolution of the model and the amount of CPU time. For this reason a broad range of turbulence schemes are incorporated, ranking from simple algebraic formulations up to second-order closure schemes with additional transport equations for turbulence quantities. To facilitate the use of the program for non-specialists in turbulence, a default scheme is implemented which should give realistic results in most applications. Also available is the more widely known level 2.5 turbulence closure of Mellor and Yamada (1982) with the modifications introduced by Galperin et al. (1988).

The schemes implemented in the program have been developed and tested to represent one or more of the following physical processes:

- turbulence generated by tidal friction in the bottom layer
- wind-induced turbulence in the surface layer
- enhancement of the bottom stress due to the interaction of waves and currents at the sea bottom
- diurnal and seasonal cycles of heating and cooling, including the evolution of thermoclines
- shear-induced mixing at river fronts

[1]Different vertical diffusion coefficients are used in the transport equations for turbulence variables.

Not included in the present version, but possibly foreseen in future releases are:

- surface wave effects at the sea surface
- sediment damping of turbulence
- double-diffusion mixing occurring when an unstable (stable) salinity and a stable (unstable) temperature gradient combine to form a stable density gradient in the vertical. Empirical formulations using different coefficients for temperature and salinity have recently been proposed (Large et al., 1994) but are not yet fully tested.
- a specific scheme taking account of the influence of internal wave activity on turbulence in the thermocline or halocline. However, a background diffusion coefficient as a function of stratification can be included in some of the turbulence schemes using one of the turbulence switches. This is discussed in Section III-1.2.2c.

The remainder of this section is organised as follows: the simpler algebraic formulations are described in Section III-1.2.1; the more sophisticated turbulence closure schemes are presented in Section III-1.2.2; a summary of model parameters and available switches is given in Section III-1.2.3.

1.2.1 Algebraic formulations

The most simple parameterisations of turbulence are obviously constant values for ν_T and λ_T, i.e. $\nu_T = \nu_b$, $\lambda_T = \lambda_b$. This is selected in the program by setting the switch IOPTK equal to 0. If IOPTK is set to 1, five different formulations are available and selected by the switch ITFORM. The first two are Richardson number dependent, the next three flow-dependent formulations.

a) Richardson number dependent formulations

If ITFORM is set to 1, the program selects the formulation of Pacanowski and Philander (1981):

$$\nu_T = \nu_{0p} f_p^{n_p}(Ri) + \nu_{bp} \tag{3.1.86}$$

$$\lambda_T = \nu_T f_p(Ri) + \lambda_{bp} \tag{3.1.87}$$

$$f_p(Ri) = (1 + \alpha_p Ri)^{-1} \tag{3.1.88}$$

The Richardson number is defined as the ratio of the squared buoyancy to the squared shear frequency:

$$Ri = N^2/M^2 \tag{3.1.89}$$

where

$$N^2 = \frac{g}{J}\Big(\beta_T \frac{\partial T}{\partial \tilde{x}_3} - \beta_S \frac{\partial S}{\partial \tilde{x}_3}\Big) \tag{3.1.90}$$

$$M^2 = \frac{1}{J^2}\Big((\frac{\partial u}{\partial \tilde{x}_3})^2 + (\frac{\partial v}{\partial \tilde{x}_3})^2\Big) \tag{3.1.91}$$

and β_T, β_S are the thermal and salinity expansion coefficients. To prevent turbulence becoming too large in the case of unstable stratification ($Ri < 0$), the following limiting condition for f_p is imposed

$$\nu_T/\nu_{0p} \simeq f_p^{n_p} < \nu_{max} \tag{3.1.92}$$

so that

$$\lambda_T/\nu_{0p} \simeq f_p^{n_p+1} < \nu_{max}^{1+1/n_p} \tag{3.1.93}$$

The following default values are used[2]

$$\nu_{0p} = 10^{-2} \ , \ n_p = 2 \ , \ \alpha_p = 5 \ , \ \nu_{bp} = 10^{-4} \ , \ \lambda_{bp} = 10^{-5} \ , \ \nu_{max} = 3 \tag{3.1.94}$$

The upper bounds for ν_T and λ_T are then given by 0.03 and respectively 0.052.

The scheme has been primarily developed for application in global ocean models (e.g. Semtner and Chervin, 1988). It has the advantage of being less sensitive to vertical resolution than the more advanced turbulence closures discussed in Section III-1.2.2. In the absence of stratification the coefficients take uniform values which makes the scheme less reliable for the study of neutral tidal and wind-driven flows. Test simulations in the Rhine plume (Ruddick et al., 1995) showed that the results are sensitive to a calibration of the model constants. Peters et al. (1988) derived a similar formulation using different values of the parameters calibrated from microstructure measurements in the Pacific Ocean.

The more historical empirical relations proposed by Munk and Anderson (1948), are selected with ITFORM = 2:

$$\nu_T = \nu_{0m} f_m(Ri) + \nu_b \tag{3.1.95}$$

$$\lambda_T = \nu_{0m} g_m(Ri) + \lambda_b \tag{3.1.96}$$

with

$$f_m(Ri) = (1 + \alpha_m Ri)^{-n_1} \tag{3.1.97}$$

$$g_m(Ri) = (1 + \beta_m Ri)^{-n_2} \tag{3.1.98}$$

In analogy with (3.1.92) and (3.1.93) the following upper limits are imposed

$$f_m < \nu_{max} \ , \ g_m < \lambda_{max} \tag{3.1.99}$$

The following default parameter values are used[2]

$$\nu_{0m} = 0.06 \ , \ \alpha_m = 10 \ , \ \beta_m = 3.33 \ , \ n_1 = 0.5 \ , \ n_2 = 1.5 \ , \ \nu_{max} = 3 \ , \ \lambda_{max} = 4 \tag{3.1.100}$$

[2]Note that ν_{0p}, ν_{bp}, λ_{bp} and ν_{0m} in (3.1.94) and (3.1.100) are expressed in the same unit as ν_T and λ_T, i.e. m^2/s.

b) flow-dependent formulations

In shelf and coastal seas tides are a prominent source of turbulence. Observations in the Irish Sea (Bowden et al, 1959) indicate that the eddy viscosity is proportional to the magnitude of the tidal current. A suitable parameterisation for tidal flow can then be written as

$$\nu_T = (\alpha(x_1, x_2, t)\Phi(\sigma) + \nu_w)f_m(Ri) + \nu_b \tag{3.1.101}$$

$$\lambda_T = (\alpha(x_1, x_2, t)\Phi(\sigma) + \nu_w)g_m(Ri) + \lambda_b \tag{3.1.102}$$

The flow field is represented by the depth-independent factor α. Explicit forms are described below. In the absence of stratification the vertical variation of turbulence is presented by a prescribed profile for $\Phi(\sigma)$ (with σ defined by (3.1.20)) which takes account of the reduction of turbulence in the near bottom and surface layers. Following Davies (1990) the following piecewise linear profile is adopted

$$\begin{aligned}\Phi(\sigma) = \Big((1-r_1)\sigma/\delta_1 + r_1\Big)/D \quad &\text{for} \quad 0 \leq \sigma \leq \delta_1\\ \Phi(\sigma) = 1/D \quad &\text{for} \quad \delta_1 \leq \sigma \leq 1-\delta_2\\ \Phi(\sigma)\Big(r_2 - (r_2-1)(1-\sigma)/\delta_2\Big)/D \quad &\text{for} \quad 1-\delta_2 \leq \sigma \leq 1\end{aligned} \tag{3.1.103}$$

where

$$D = 1 + \frac{1}{2}\delta_1(r_1-1) + \frac{1}{2}\delta_2(r_2-1) \tag{3.1.104}$$

is a normalisation factor such that the depth-integral of $\Phi(\sigma)$ equals 1. The parameters δ_1, δ_2 are the fractional depths of the bottom and surface layers, and r_1, r_2 the ratios of the bottom and surface values of Φ with respect to the interior value. Default values for the parameters are

$$\delta_1 = \delta_2 = 0\ ,\ r_1 = r_2 = 1 \tag{3.1.105}$$

giving a uniform vertical profile. More details about a proper selection of these parameters can be found in e.g. Davies (1993).

In analogy with the formulation used by Naimie et al. (1994) for the simulation of the circulation around Georges Bank the damping functions $f_m(Ri)$ and $g_m(Ri)$ take the form given by the Munk-Anderson expressions (3.1.97)–(3.1.98). Following Glorioso and Davies (1995) wind-induced turbulence is related to the surface friction velocity using the simple form

$$\nu_w = \lambda_* u_{*s} \tag{3.1.106}$$

where λ_* is a constant tunable parameter and the surface friction velocity u_{*s} is given by

$$\rho_0 u_{*s} = \tau_s^{1/2} = (\tau_{s1}^2 + \tau_{s2}^2)^{1/2} \tag{3.1.107}$$

The last terms on the right of (3.1.101)–(3.1.102) are the uniform background eddy viscosity ν_b and diffusivity λ_b.

The following three formulations for the flow factor α can be selected by setting the switch ITFORM equal to 3, 4, 5:

$$\alpha = K_1(\overline{U}^2 + \overline{V}^2)^{1/2} \tag{3.1.108}$$

$$\alpha = K_2(\overline{U}^2 + \overline{V}^2)/(H^2\omega_1) \tag{3.1.109}$$

$$\alpha = K_1(\overline{U}^2 + \overline{V}^2)^{1/2}\Delta_b/H \tag{3.1.110}$$

where Δ_b measures the thickness of the bottom boundary layer as a function of the bottom friction velocity u_{*b}:

$$\Delta_b = \min(C_\nu u_{*b}/\omega_1, H) \tag{3.1.111}$$

$$\rho_0 u_{*b} = \tau_b^{1/2} = (\tau_{b1}^2 + \tau_{b2}^2)^{1/2} \tag{3.1.112}$$

and ω_1 is a characteristic frequency. In shallow areas $\Delta_b = H$ so that (3.1.110) reduces to (3.1.108). The following default values are taken

$$K_1 = 2.5 \times 10^{-3}\ ,\ K_2 = 2 \times 10^{-5}\ ,\ C_\nu = 2.0\ ,\ \omega_1 = 10^{-4}\mathrm{s}^{-1} \tag{3.1.113}$$

The homogeneous form of the eddy viscosity parameterisation (3.1.101) has been used in recent years for the prediction of tidal currents and surface elevations in the Northwest European Continental Shelf (e.g. Davies, 1990; Davies et al., 1997), the Irish and Celtic Seas (e.g. Davies and Jones, 1992; Davies, 1993) and the shelf edge off the West coast of Scotland (Proctor and Davies, 1996).

1.2.2 Turbulence closure schemes

This type of schemes is selected when the switch IOPTK equals 2. In analogy with molecular diffusion where the eddy viscosity and diffusion coefficients are proportional to the mean velocity times the mean free path of the molecules, the eddy coefficients ν_T and λ_T are considered as the product of a turbulent velocity scale and a length scale l usually denoted by the Kolmogorov-Prandtl "mixing length". A commonly used velocity scale is the square root $k^{1/2}$ of the turbulent kinetic energy. This parameter can be obtained by solving a transport equation. The most general form of this equation, as used in the program, is written as

$$\Big(\mathcal{I} + \mathcal{A}_h + \mathcal{A}_v - \mathcal{D}_h\Big)k - \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\big(\frac{\nu_T}{\sigma_k} + \nu_b\big)\frac{1}{J}\frac{\partial k}{\partial\tilde{x}_3}\Big) = \nu_T M^2 - \lambda_T N^2 - \varepsilon \tag{3.1.114}$$

where the time derivative, the horizontal and vertical advection and the diffusion operators are defined by (3.1.71)–(3.1.74), (3.1.76)–(3.1.78), N^2 and M^2 are the squared buoyancy and shear frequencies given by (3.1.90)–(3.1.91) and ε denotes the dissipation rate of turbulence energy. All turbulence transport equations are solved with the same horizontal diffusion coefficient λ_H which is the same as the one used in the equations of temperature and salinity. Surface and bottom boundary conditions will be derived in Sections III-1.6.1c and III-1.6.2c. The eddy coefficients are then expressed as

$$\nu_T = S_m k^{1/2} l + \nu_b\ ,\ \lambda_T = S_h k^{1/2} l + \lambda_b \tag{3.1.115}$$

where S_m, S_h are usually referred as the stability functions and ν_b, λ_b are prescribed background coefficients. The dissipation rate is parameterised according to

$$\varepsilon = \varepsilon_0 k^{3/2}/l \tag{3.1.116}$$

where ε_0 is a constant determined below.

Less certainty exists about a proper formulation for the mixing length. An alternative of this "$k - l$" formulation is the "$k - \varepsilon$" theory where ε is taken as the second turbulence variable instead of l. Expressions (3.1.115) are then replaced by

$$\nu_T = S_u k^2/\varepsilon + \nu_b \ , \ \lambda_T = S_b k^2/\varepsilon + \lambda_b \tag{3.1.117}$$

The $k - l$ or $k - \varepsilon$ formulation is selected in the program by setting the switch ITCPAR equal to 1 or 2. Expressions for the stability functions, different formulations for l or ε, and the use of limiting conditions are discussed in the following subsections.

a) stability functions

Two different formulations can be selected for the stability functions depending on the value of the switch ISTPAR. If ISTPAR $= 1$, they are expressed as a function of the stability parameter G_h if ITCPAR $= 1$ or α_N if ITCPAR $= 2$. Their explicit forms are

$$S_m = \frac{0.556 + 2.18G_h}{1 + 20.4G_h + 53.1G_h^2} \ , \ S_h = \frac{0.699}{1 + 17.3G_h} \tag{3.1.118}$$

where

$$G_h = \frac{l^2}{k}N^2 \tag{3.1.119}$$

and

$$S_u = \frac{0.108 + 0.0229\alpha_N}{1 + 0.471\alpha_N + 0.0275\alpha_N^2} \ , \ S_b = \frac{0.177}{1 + 0.403\alpha_N} \tag{3.1.120}$$

where

$$\alpha_N = \frac{k^2}{\varepsilon^2}N^2 \tag{3.1.121}$$

Expressions (3.1.118)–(3.1.119) are taken from Mellor and Yamada (1982) with the modifications discussed in Galperin et al. (1988). They are obtained from an original set of 10 equations for all second order correlations (Reynolds stresses, buoyancy fluxes and density variance) after making the necessary closure assumptions and additional approximations, denoted as the "level 2.5" scheme in the classification used by Mellor and Yamada (1974). Their $k - \varepsilon$ analogues (3.1.120)–(3.1.121) are derived in Luyten et al. (1996) using similar approximations. Interesting to note is that the stability functions have no explicit dependence on the current shear as in the original Mellor-Yamada formulation. As shown by Deleersnijder and Luyten (1994) this improves the stability of the scheme.

An important property of these types of stability functions is the inhibition of turbulence with increasing stable stratification. This can be deduced from the turbulence energy equation (3.1.114) assuming that the terms on the left hand side are negligible. It can then be shown that (S_m, S_h) or (S_u, S_b) are functions of the Richardson number (see Section III-1.2.2c). In particular the stability functions tend to zero and turbulence ceases if $Ri \to Ri_c$ where Ri_c equals 0.195 in the $k-l$ and 0.579 in the $k-\varepsilon$ formulation. This suppression of turbulence appears not to be in agreement with observational data (e.g. Simpson et al., 1996). A possible remedy is the introduction of limiting conditions discussed in Section III-1.2.2c below. An alternative is to adopt a formulation where the stability functions are expressed in terms of the Richardson number and decrease smoothly to zero if $Ri \to \infty$. This can be selected in the program by setting the switch ISTPAR equal to 2. In that case the following expressions are used

$$S_m = S_{m0} f_m(Ri)\ ,\ S_h = S_{h0} g_m(Ri) \tag{3.1.122}$$

$$S_u = S_{u0} f_m(Ri)\ ,\ S_b = S_{b0} g_m(Ri) \tag{3.1.123}$$

where S_{m0}, S_{h0}, S_{u0} and S_{b0} are the neutral values (for $N^2 = 0$) of the stability coefficients from (3.1.118) or (3.1.120) and f_m, g_m are given by the Munk-Anderson relations (3.1.97)–(3.1.98). Note that if $f_m(Ri)$ and $g_m(Ri)$ are set to their neutral value 1 in the expressions (3.1.122)–(3.1.123), one obtains the formulation used in the "standard" $k-\varepsilon$ model (Rodi, 1984; Burchard and Baumert, 1995).

The value of the neutral stability coefficient S_{m0} or S_{u0} can be used to determine the parameter ε_0 which relates ε and l in (3.1.116) by taking the boundary layer approximation (see Section III-1.6.1c). This gives

$$\varepsilon_0 = S_{m0}^3 \quad \text{or} \quad \varepsilon_0 = S_{u0}^{3/4} \tag{3.1.124}$$

A further discussion of the stability functions is postponed to Section III-1.2.2c.

b) parameterisations for the mixing length or dissipation rate

Probably the weakest part in most turbulence energy models is a suitable parameterisation either for the mixing length l or the dissipation rate ε. A series of schemes are available in the program selected by the switches NTRANS and ILENG. The former determines the number of transport equations for turbulence variables solved by the program and may take the values 0, 1, 2. The corresponding schemes are denoted by zero-, one- and two-equation models. The latter selects an algebraic mixing length formulation and is redundant if NTRANS equals 2.

- two-equation models

  In a two-equation model the k-equation (3.1.114) is solved. If the $k-l$ option is considered, ε is parameterised using (3.1.116) and an additional equation is supplied for the quantity kl of the form given in Mellor and Yamada (1982):

$$\Big(\mathcal{I}+\mathcal{A}_h+\mathcal{A}_v-\mathcal{D}_h\Big)kl-\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big((\frac{\nu_T}{\sigma_k}+\nu_b)\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(kl)\Big) = \frac{1}{2}E_1 l(\nu_T M^2-\lambda_T N^2)-\frac{1}{2}\varepsilon_0 k^{3/2}\tilde{W} \tag{3.1.125}$$

The wall proximity function $\tilde{W}$ is defined by

$$\tilde{W} = 1 + E_2\Big(\frac{l}{\kappa}\big(\frac{1}{\zeta - x_3 + z_{0s}} + \frac{1}{h + x_3 + z_{0b}}\big)\Big)^2 \qquad (3.1.126)$$

where $\kappa = 0.4$ is von Kármán's constant. The bottom and surface roughness length scales z_{0b} and z_{0s} are set to zero by default. The presence of the wall term in (3.1.125) ensures that the solution tends to the wall layer forms given by (3.1.130) below near the surface and the bottom provided that E_1 and E_2 are related by

$$E_1 - E_2 = 1 - 2\kappa^2/(\sigma_k S_{m0}^2) \qquad (3.1.127)$$

In the $k-\varepsilon$ version the k-equation is solved together with a transport equation for ε (e.g., Rodi, 1984; Launder and Spalding, 1974):

$$\Big(\mathcal{I} + \mathcal{A}_h + \mathcal{A}_v - \mathcal{D}_h\Big)\varepsilon - \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\big(\frac{\nu_T}{\sigma_\varepsilon} + \nu_b\big)\frac{1}{J}\frac{\partial\varepsilon}{\partial\tilde{x}_3}\Big) = c_{1\varepsilon}\frac{\varepsilon}{k}(\nu_T M^2 - c_{3\varepsilon}\lambda_T N^2) - c_{2\varepsilon}\frac{\varepsilon^2}{k} \qquad (3.1.128)$$

A parameterisation of the dissipation rate in the k-equation is no longer required but (3.1.116) may be used *a posteriori* to evaluate the mixing length using the known values of k and ε. The boundary layer approximation can be obtained as a special case without the need for an extra wall term but imposes a constraint on the model parameters $c_{1\varepsilon}$ and $c_{2\varepsilon}$ similar to (3.1.127):

$$c_{1\varepsilon} - c_{2\varepsilon} = -\kappa^2/(\sigma_\varepsilon S_{u0}^{1/2}) \qquad (3.1.129)$$

Baumert et al. (1997) pointed out that the main difference between the kl- and ε-equation is the weight of the buoyancy term. In the ε-equation this is presented by the constant $c_{3\varepsilon}$ considered as the most uncertain part in the model. The value adopted in the current version of the program is 0.2 for stable and 1 for unstable stratification as suggested by Rodi (1987). A different value of -1.4 for both stable and unstable flows has more recently been proposed by Burchard and Baumert (1995) based on theoretical arguments and numerical simulations of thermal stratification.

- one-equation models

A generally accepted view is that the kl- and ε-equation have a less physical basis than the k-equation. Replacing one of the latter two equations by a simpler algebraic prescription for the mixing length may yield realistic results while saving CPU time. When a one-equation model is chosen, the k-equation is still solved with ε modelled according to (3.1.116) while l is determined using one of the four formulations, available in the program, selected by the switch ILENG. The main differences between the $k-l$ and $k-\varepsilon$ version are now the different parameterisations of the stability coefficients. No explicit stratification dependence is included since this is already taken into account by the stability functions. The basic requirement in each formulation is that l reduces to the following forms near, respectively, the bottom and the surface

$$l \simeq l_1 = \kappa(\sigma H + z_{0b}) \ , \ l \simeq l_2 = \kappa(H - \sigma H + z_{0s}) \tag{3.1.130}$$

with σ defined by (3.1.20).
The first and simplest expression is the parabolic law

$$\frac{1}{l} = \frac{1}{l_1} + \frac{1}{l_2} \tag{3.1.131}$$

having a maximum at $\sigma = 0.5$.
The second is the "quasi-parabolic" law given by

$$\frac{1}{l} = \frac{1}{l_1}\Big(\frac{l_1 + l_2}{l_2}\Big)^{1/2} \tag{3.1.132}$$

which differs from the first one in that l has a maximum at $\sigma \simeq 2/3$ closer to the surface.
The third, recommended by Xing (Xing and Davies, 1996) has the same form as (3.1.131) but with l_1 replaced by

$$l_1 = \kappa(\sigma H e^{\beta_1 \sigma} + z_{0b}) \tag{3.1.133}$$

allowing for a larger reduction of the mixing length in the lower parts of the water column.
The fourth formulation, initially proposed by Blackadar (1962), has the form

$$\frac{1}{l} = \frac{1}{l_1} + \frac{1}{l_2} + \frac{1}{l_a} \tag{3.1.134}$$

so that $l \rightarrow l_a$ far from the boundaries. Mellor and Yamada (1974) defined l_a as the ratio of the first to the zeroth moment of the vertical profile of the turbulent velocity scale $k^{1/2}$. Hence

$$l_a = \alpha_1 \int_0^L H(1-\sigma) k^{1/2} J \, d\tilde{x}_3 \, / \int_0^L k^{1/2} J \, d\tilde{x}_3 \tag{3.1.135}$$

- zero-equation models

At the lowest level the time derivative and all advective and diffusion terms are neglected in the k-equation (terms on the left hand side of equation (3.1.114)). The solution method now depends on the value of the switch ISTPAR. If ISTPAR $= 1$, the stability functions are defined by (3.1.118)–(3.1.119) or (3.1.120)–(3.1.121). In the terminology of Mellor and Yamada (1974) this is known as the level 2 model. A quantity R is first determined by taking the largest root of the following second order equation

$$R^2 + (c_{21}N^2 + c_{22}M^2)R + N^2(c_{23}M^2 + c_{24}N^2) = 0 \tag{3.1.136}$$

with

$$R = k/l^2 \quad \text{or} \quad R = \varepsilon^2/k^2 \tag{3.1.137}$$

in the $k-l$ or the $k-\varepsilon$ version. The turbulence energy is obtained by combining (3.1.136), (3.1.137) using (3.1.116) to express ε as function of l in the second relation of (3.1.137) and l determined using one of the mixing length formulations selected by ILENG. The model constants in (3.1.136) have the following values for respectively the $k-l$ or $k-\varepsilon$ formulation

$$(c_{21}, c_{22}, c_{23}, c_{24}) = (24.5, -3.26, -12.8, 65.7) \tag{3.1.138}$$

$$(c_{21}, c_{22}, c_{23}, c_{24}) = (0.648, -0.1085, -0.0229, 0.03953) \tag{3.1.139}$$

If ISTPAR $= 2$, turbulence energy is determined by $k = l^2R/\varepsilon_0$ in the $k-l$ or $k = l^2R/\varepsilon_0^2$ in the $k-\varepsilon$ parameterisation where

$$R = S_mM^2 - S_hN^2 \quad \text{or} \quad R = S_uM^2 - S_bN^2 \tag{3.1.140}$$

The level 2 scheme is at first sight more attractive than the more complicated one- and two-equation models since it consumes the least computing time and incorporates most of the physics. This is confirmed by the simulations performed by e.g. Martin (1985), Ruddick et al. (1995), Luyten et al. (1996) who found no appreciable difference between the level 2 and level 2.5 schemes. The main problem however is that the scheme may become numerically unstable. As shown by Frey (1991) this occurs for large time steps or small vertical resolutions. The problem can be resolved by retaining the time derivative and vertical diffusion terms in the k-equation (Luyten et al., 1996). This means that preference should be given to a one- (or two-) equation model.

c) limiting conditions

In Section III-1.2.2a it was stated that if the switch ISTPAR is set to 1, the stability functions can be related to the Richardson number by making the additional assumption of equilibrium between production and dissipation in the equation of turbulence energy. This follows from equations (3.1.136) and (3.1.137) which allow to express the stability parameters G_h and α_N as function of Ri. The variation of the stability functions versus Ri is plotted in Figures 3.1.2a and 3.1.2c (stable stratification) and Figures 3.1.2b and 3.1.2d (unstable stratification) for the four different formulations given by equations (3.1.118)–(3.1.119), (3.1.120)–(3.1.121), (3.1.122), (3.1.123). The values of S_u and S_b are normalised with ε_0 so that the ordinate either represents $\nu_T/(k^{1/2}l)$ or $\lambda_T/(k^{1/2}l)$. The cessation of turbulence above a critical Richardson number is clearly observed. The Munk-Anderson type formulations have in contrast a much smoother behaviour. None of these schemes are however qualitatively in agreement with for example the observations of equatorial turbulence reported in Peters et al. (1988), Moum et al. (1989) who found a rapid decrease of turbulence for low Ri leveling off at some threshold value Ri_c . Above this value turbulence remains active probably due to internal wave mixing. More data are required to verify this behaviour. (The more recent data of Peters et al. (1995) show less evidence for a critical Ri_c).

A possible way to incorporate background mixing in the level 2.5 scheme is to use a limiting condition for the mixing length (e.g. Hassid and Galperin, 1983; Galperin et al., 1988; Luyten et al., 1996). This takes the form

$$l = \min(k^{1/2}c/N, l) \tag{3.1.141}$$

where c is a model parameter. From (3.1.118)–(3.1.119) or (3.1.120)–(3.1.121) it can be seen that a limiting condition of this form imposes upper limits

$$(G_{h,max}, \alpha_{N,max}) = (c^2, c^2/\varepsilon_0^2) \tag{3.1.142}$$

for the stability parameters so that the stability functions (S_m, S_h) or (S_u, S_b) are bounded from below. The evolution of the stability functions with a limiting condition is also shown in Figures 3.1.2a and 3.1.2c. Compared to the unlimited formulations the functions no longer decrease to zero but level off at a constant value when Ri exceeds a critical value Ri_c. The precise value of Ri_c depends on the parameter c. Hassid and Galperin (1983) choose $c = 0.75$ giving $Ri_c = 0.15$. In the $k - \varepsilon$ model $c = 0.495$ so that $Ri_c = 0.25$ in agreement with observational data. Once l attains its limiting value, the dissipation rate ε and the eddy coefficients ν_T, λ_T reduce to a "background" level depending on stratification. Using (3.1.115)–(3.1.117) one has

$$\varepsilon = \varepsilon_{min} kN \ , \ \nu_T = \nu_{min} k/N \ , \ \lambda_T = \lambda_{min} k/N \tag{3.1.143}$$

where

$$(\varepsilon_{min}, \nu_{min}, \lambda_{min}) = (0.229, 0.046, 0.049) \tag{3.1.144}$$

in the $k - l$ and

$$(\varepsilon_{min}, \nu_{min}, \lambda_{min}) = (0.381, 0.126, 0.123) \tag{3.1.145}$$

in the $k - \varepsilon$ model.

The relations (3.1.143) still contain the unknown parameter k. The observational data reviewed by Gregg (1987), indicate that $\varepsilon \sim N^\alpha$ with $\alpha \simeq 1 - 1.5$. Substituting into the first relation of (3.1.143) this gives $k \sim N^{\alpha-1}$. Gaspar et al. (1990) assume that $\alpha \simeq 1$ so that k equals a value k_{min} independent of stratification. They propose $k_{min} = 10^{-6}$J/kg which is the value selected as the default in the program. Luyten and Rippeth (1997) compared the $k - \varepsilon$ model using the limiting conditions with turbulence measurements in the Irish Sea. Good agreement was found if $k_{min} = 3 \times 10^{-6}$J/kg. A further improvement

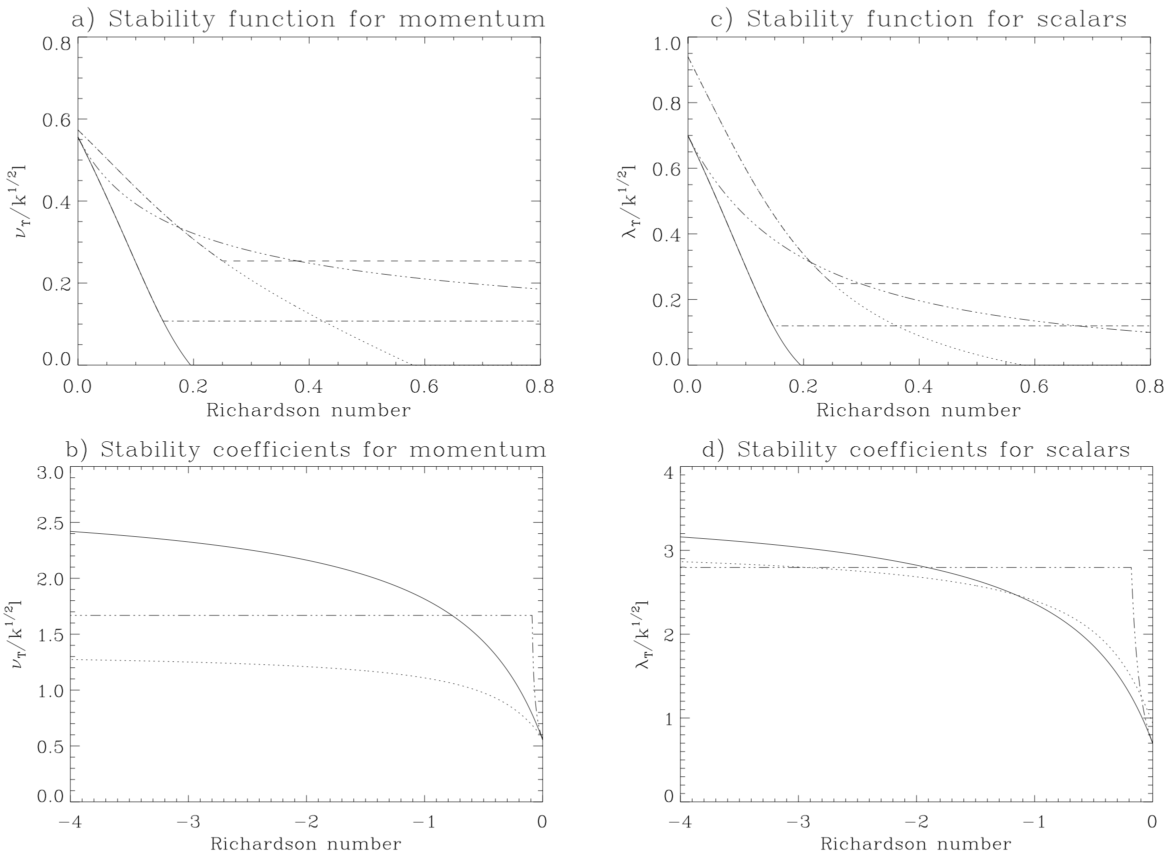


Figure 3.1.2: Evolution of the stability functions for momentum and scalar quantities in terms of the Richardson number for the $k-l$ model without limiting conditions (solid), the $k-l$ model with limiting conditions (dash-dots), the $k-\varepsilon$ model without limiting conditions (dots), the $k-\varepsilon$ model with limiting conditions (dashes) and the Munk-Anderson formulation (3.1.122)–(3.1.123) (dash and 3 dots).To allow a comparison between the different formulations S_u and S_b are divided by ε_0 in the case of the $k-\varepsilon$ model so that the ordinate either represents $\nu_T/(k^{1/2}l)$ or $\lambda_T/(k^{1/2}l)$.

should be to consider a limiting value of k depending on stratification which vanishes in the limit $N \to 0$. In a recent study Stacey et al. (1995) used a condition of the form $k \sim N^{-1/2}$ so that $\varepsilon \sim N^{1/2}$ and $\nu_T, \lambda_T \sim N^{-3/2}$ but this form diverges when $N \to 0$.

The limiting conditions for k and l are selected in the program by setting the switch ILIM equal to 1. Note that limiting conditions are not allowed by the program for the level 2 scheme (NTRANS = 0).

In the case of unstable stratification a lower bound must be imposed on the stability parameter to avoid the stability functions (3.1.118) or (3.1.120) becoming negative or the denominators taking a zero value. An additional constraint is that the "shear" stability parameter must remain positive. Details can be found in Galperin et al. (1988) or in Luyten et al. (1996). This gives

$$G_h > G_{h,min} = -0.046 \ , \ \alpha_N > \alpha_{N,min} = -1.725 \tag{3.1.146}$$

1.2.3 Model parameters, default scheme and program switches

For practical reasons a similar notation has been used for parameters in the $k-l$ and $k-\varepsilon$ formulations. Equivalence can be made with the Mellor-Yamada terminology by making the substitutions

$$\begin{aligned}
(k, \varepsilon_0, \sigma_k) &= (q^2/2, 8^{1/2}/B_1, S_{m0}/(2^{1/2}S_q)) \\
(S_m, S_h) &= 2^{1/2}(S_M, S_H) \\
G_h &= -2G_H \\
(\nu_T, \lambda_T) &= (K_m, K_H)
\end{aligned} \tag{3.1.147}$$

The only difference is that, in this model, the vertical diffusion coefficient defined by ν_T/σ_k now depends explicitly on stratification.

The values of the model constants used in the expressions (3.1.120) are different from the ones used in Luyten et al. (1996). They are obtained using the same analysis but with an updated set of empirical constants (see equations (A7) and (A9) of that paper) recommended by Hossain and Rodi (1982):

$$(c_1, c_{21}, c_{22}, c_{23}, c_3, c_{1\beta}, c_{2\beta}, c_{3\beta}) = (2.2, 0.55, 0, 0, 0.55, 3.0, 0.5, 0.8) \tag{3.1.148}$$

A list of all model parameters and their default values is given in Table 3.1.2.

Users not familiar with turbulence modelling, may find it difficult to select a turbulence scheme appropriate for a particular application of the program. A default scheme is therefore implemented. The choice is based on the following observations:

- The simple algebraic formulations are easier to understand but may give inadequate results when applied to a combined regime of wind, tides and stratification.

- The level 2 schemes can exhibit numerical instabilities.

Table 3.1.2: Turbulence model parameters.

			Pacanowski-Philander			
ν_{0p}	n_p	α_p	ν_{bp}	λ_{bp}	ν_{max}	
10^{-2} m^2/s	2.0	5.0	10^{-4} m^2/s	10^{-5} m^2/s	3.0	
			Munk-Anderson			
ν_{0m}	α_m	β_m	n_1	n_2	λ_{max}	
0.06 m^2/s	10.0	3.33	0.5	1.5	4.0	
			flow-dependent formulations			
K_1	K_2	ω_1	C_ν	λ_*		
2.5×10^{-3}	2×10^{-5}	10^{-4} s^{-1}	2.0	0.0		
δ_1	δ_2	r_1	r_2			
0.0	0.0	1.0	1.0			
			$k-l$ model			
ε_0	σ_k	E_1	E_2	k_{min}	$G_{h,min}$	$G_{h,max}$
0.172	1.96	1.8	1.33	10^{-6} J/kg	-0.046	0.560
			$k-\varepsilon$ model			
ε_0	σ_k	σ_ε	$c_{1\varepsilon}$	$c_{2\varepsilon}$		
0.188	1.0	1.3	1.55	1.92		
$c_{3\varepsilon}$		k_{min}	$\alpha_{N,min}$	$\alpha_{N,max}$		
0.6-0.4Sign(N^2)		10^{-6} J/kg	-1.725	6.903		
			mixing length formulations			
z_{0s}	z_{0b}	β_1	α_1			
0.0 m	0.0 m	-2.0	0.2			

Table 3.1.3: Program switches for turbulence.

IOPTK	Determines the general type of the turbulence scheme. 0 : uniform values for ν_T and λ_T 1 : ν_T and λ_T evaluated using one of the algebraic expressions described in Section III-1.2.1. Further selection is made with the switch ITFORM. 2 : ν_T and λ_T evaluated using one of the turbulence closure schemes described in Section III-1.2.2. Further selection is made using the switches NTRANS, ITCPAR, ISTPAR, ILENG, ILIM, IAHDHT. Default value is 2.
ITFORM	Determines the type of formulation if IOPTK is set to 1. 1 : Pacanowski-Philander formulations (3.1.86)–(3.1.88), (3.1.92) 2 : Munk-Anderson relations (3.1.95)–(3.1.99) 3 : flow-dependent formulation (3.1.101) using (3.1.108) for the flow factor α 4 : flow-dependent formulation (3.1.101) using (3.1.109) for the flow factor α 5 : flow-dependent formulation (3.1.101) using (3.1.110) for the flow factor α Default value is 3.
NTRANS	Selects the number of transport equations. 0 : zero-equation model with k determined by (3.1.136)–(3.1.137) or (3.1.140) and l by ILENG 1 : one-equation model using the k-equation (3.1.114) and l determined by ILENG 2 : two-equation model using the k-equation (3.1.114) and the kl-equation (3.1.125) if ITCPAR equals 1, or the ε-equation (3.1.128) if ITCPAR equals 2. Default value is 1.
ITCPAR	Selects the type of the scheme. 1 : $k-l$ formulation with the stability functions evaluated either by (3.1.118)–(3.1.119) if ISTPAR $=1$ or by (3.1.122) if ISTPAR $=2$ and l determined either by ILENG if NTRANS $=0,1$ or using the kl-equation (3.1.125) if NTRANS $=2$ 2 : $k-\varepsilon$ formulation with the stability functions evaluated either by (3.1.120)–(3.1.121) if ISTPAR $=1$ or by (3.1.123) if ISTPAR $=2$ and ε determined either using (3.1.116) with l selected by ILENG if NTRANS $=0,1$ or by solving the ε-equation (3.1.128) if NTRANS $=2$. Default value is 2.

Table 3.1.3: continued.

ISTPAR	Selects whether the stability functions are evaluated as function of the stability parameter or the Richardson number. 1 : stability parameter formulation (3.1.118)–(3.1.119) or (3.1.120)–(3.1.121) 2 : Richardson number formulation (3.1.122) or (3.1.123) Default value is 1.
ILENG	Selects a mixing length formulation when NTRANS equals 0 or 1. (If NTRANS $= 2$, ILENG is only used to initialise turbulence arrays at the start of the program.) 1 : parabolic law (3.1.131) 2 : "quasi-parabolic" law (3.1.132) 3 : "Xing" formulation (3.1.133) 4 : "Blackadar" formulation (3.1.134)–(3.1.135) Default value is 4.
ILIM	Disables or enables the use of limiting conditions. 0 : limiting conditions disabled 1 : limiting conditions enabled Default value is 1.
IAHDHT	The evaluation of the advective and horizontal diffusion terms represented by the operators $\mathcal{A}_h, \mathcal{A}_v, \mathcal{D}_h$ in the turbulent transport equations (3.1.114), (3.1.125), (3.1.128) is disabled/enabled when this switch is set to 0/1. Default value is 0.

- While consuming more CPU time no explicit evidence has yet been found from model comparisons with oceanographic data that the two-equation models always give superior results compared to the simpler mixing length formulations.
- The $k - \varepsilon$ formulation of the stability functions appears to give a more realistic (higher) value for the critical Richardson number than the Mellor-Yamada formulation. This is confirmed by numerical experiments of wind-driven mixed layer deepening (Luyten et al., 1996) and simulations of seasonal stratification in the North Sea (Luyten, 1996).
- On physical grounds preference is given to the "Blackadar" mixing length formulation since it is related to the turbulence energy although in vertically integrated form.
- The limiting conditions provide background values for the eddy viscosity and diffusivity in the case of a stable stratification. The theory is based on physical and observational evidence and may eliminate the necessity of introducing arbitrary background coefficients.

The default values for the turbulence switches are therefore:

$$(\mathsf{IOPTK}, \mathsf{NTRANS}, \mathsf{ITCPAR}, \mathsf{ISTPAR}, \mathsf{ILENG}, \mathsf{ILIM}) = (2,1,2,1,4,1) \tag{3.1.149}$$

Users who prefer to use the Mellor-Yamada formulation as in the model of Blumberg and Mellor (1987) must select

$$(\mathsf{IOPTK}, \mathsf{NTRANS}, \mathsf{ITCPAR}, \mathsf{ISTPAR}, \mathsf{ILENG}, \mathsf{ILIM}) = (2,2,1,1,4,0) \tag{3.1.150}$$

The meaning of all turbulence switches is explained in Table 3.1.3. Note that an additional switch IAHDHT is provided to disable/enable the inclusion of the advective and horizontal diffusion terms in the turbulent transport equations (3.1.114), (3.1.125), (3.1.128). It is noted that these terms can be considered as small in most oceanographic applications while consuming more CPU time.

1.3 Horizontal diffusion

Horizontal subgrid-scale processes not resolved by the model are parameterised by the horizontal diffusion coefficients ν_H and λ_H. Two options are available in the program, selected by the model switch IODIF. In the first case (IODIF $= 1$) ν_H and λ_H are uniform in space and time. In the second one (IODIF $= 2$) they are taken proportional to the horizontal grid spacings and the magnitude of the velocity deformation tensor in analogy with Smagorinsky's (1963) parameterisation

$$\nu_H = C_{m0}\Delta x_1 \Delta x_2 D_T \ , \ \lambda_H = C_{s0}\Delta x_1 \Delta x_2 D_T \tag{3.1.151}$$

where

$$D_T^2 = \Big(\frac{\partial u}{\partial x_1}\Big)^2 + \Big(\frac{\partial v}{\partial x_2}\Big)^2 + \frac{1}{2}\Big(\frac{\partial u}{\partial x_2} + \frac{\partial v}{\partial x_1}\Big)^2 \tag{3.1.152}$$

and Δx_1, Δx_2 are the horizontal grid spacings. In view of the uncertainty concerning the values of the numerical coefficients C_{m0} and C_{s0} they are assumed to be equal by default. Oey and Chen (1992b) adopted a value of 0.2 mainly to damp numerical oscillations. The default value used in the program is 0.1.

In the case of spherical coordinates the formulations (3.1.151)–(3.1.152) are replaced by

$$\nu_H = C_{m0}R^2\cos\phi\Delta\lambda\Delta\phi D_T \ , \ \lambda_H = C_{s0}R^2\cos\phi\Delta\lambda\Delta\phi D_T \tag{3.1.153}$$

and

$$D_T^2 = \Big(\frac{1}{R\cos\phi}\frac{\partial u}{\partial\lambda} - \frac{\tan\phi}{R}v\Big)^2 + \frac{1}{R^2}\Big(\frac{\partial v}{\partial\phi}\Big)^2 + \frac{1}{2}\Big(\frac{1}{R}\frac{\partial u}{\partial\phi} + \frac{\tan\phi}{R}u + \frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}\Big)^2 \tag{3.1.154}$$

where $\Delta\lambda$ and $\Delta\phi$ are the horizontal grid spacings (radians) in spherical coordinates.

1.4 Physical optics

The absorption of solar heat in the water column is represented by the source term $\partial I/\partial x_3$ in the temperature equation (3.1.7). The solar irradiance I is expressed as the sum of an infrared and a short-wave component so that

$$I(x_1,x_2,x_3,t) = R_1I_1(x_1,x_2,x_3,t) + (1-R_1)I_2(x_1,x_2,x_3,t) \tag{3.1.155}$$

At the surface ($x_3 = \zeta$) I_1 and I_2 are equal to the solar radiation Q_{rad} incident at the sea surface (taking account of reflectance by the albedo). A formulation for Q_{rad} is further discussed in Section III-1.6.5.

The infrared part decays exponentially with depth using a law of the form

$$\frac{\partial I_1}{\partial x_3} = k_1I_1 \tag{3.1.156}$$

where k_1 is an inverse attenuation depth taken to be constant. This depth is usually of the order of 1 m or smaller so that infrared radiation is almost completely absorbed in a shallow surface layer.

Throughout most of the water column, I_2 obeys a decay law similar to (3.1.156):

$$\frac{\partial I_2}{\partial x_3} = (k_2+k_3)I_2 \tag{3.1.157}$$

The coefficient k_2 represents the "standard" diffuse attenuation coefficient for monochromatic light. The corresponding attenuation depth $1/k_2$ is usually much larger than $1/k_1$. In the model k_2 includes not only the absorption of pure seawater, but also the contributions due to dissolved "yellow-substance" and very small (non-sinking) particles of iSPM. We assume that these contributions to be related to salinity

$$k_2 = k_2^w(S) = k_a + k_b((S_a - S)/S_a) \tag{3.1.158}$$

where k_a is the minimum value in offshore waters at salinity S_a, and $k_b = k_2^w(0) - k_a$. In the present version of the model we assume that all factors contributing to an increase in background coefficient in coastal waters are related to decreasing salinity. Yellow substance is, at least partly, contributed by freshwaters. Fine iSPM is material which does not sink at any significant rate, and is not dealt with by the sediment model described in Chapter III-2 (although it may, in reality, be removed by aggregation with larger sinking particles). In some cases fine iSPM results from precipitation when freshwater mixes with seawater, in other cases it may be resuspended from estuarine sediments. Taking $k_a = 0.06$ m^{-1}, $k_b = 2$ m^{-1} and $S_a = 35$ PSU, $k_2^w(S)$ reduces to the form

$$k_2 = k_{20}^w - \epsilon^S S \tag{3.1.159}$$

with $k_{20}^w = k_a + k_b = 2.06$ m^{-1} and $\epsilon^S = k_b/S_a$=0.05714 PSU^{-1}m^{-1}.

The second term on the right hand side of (3.1.156) takes account of the hyper-exponential decay of polychromatic, wide-angle, short-wave radiation near the sea surface. The corresponding coefficient k_3 is given by

$$\begin{array}{lll} k_3 = -(\ln R_2)/\Delta_{opt} & \text{if} & |x_3 - \zeta| \leq \Delta_{opt} \\ k_3 = 0 & \text{if} & |x_3 - \zeta| > \Delta_{opt} \end{array} \tag{3.1.160}$$

where R_2 denotes the exponential fraction of hyper-exponential decay and Δ_{opt} the tickness of the absorption layer.

The following default values are adopted for the optical parameters

$$(R_1, k_1, k_{20}^w, \epsilon^S, R_2, \Delta_{opt}) = (0.54, 10.0, 2.06, 0.05714, 0.4, 7.0) \tag{3.1.161}$$

giving $k_2 = 0.06$ at the reference salinity S_a and $k_3 = 0.13$ (if $|x_3 - \zeta| \leq \Delta_{opt}$).

Note that (3.1.155)–(3.1.156) become identical to the more classical but simpler formulation of Paulson and Simpson (1977) for oceanic waters by using $\epsilon^S = 0$, $R_2 = 1$ and selecting values for k_1, k_{20}^w following one of the water types in Jerlov's (1968) optical classification scheme.

1.5 Equation of state

An equation of state is required for the following reasons:

- to evaluate the buoyancy in the equation of vertical hydrostatic equilibrium (3.1.27)
- to determine values for the thermal and salinity expansion coefficients β_S and β_T which are needed in equation (3.1.90) for the evaluation of the buoyancy frequency.

Two different options are available in the program depending on the value of the switch IOPTD.

If IOPTD equals 1, a linearised equation of state is taken of the form

$$\rho = \rho_0\Big(1 + \beta_S(S - S_0) - \beta_T(T - T_0)\Big) \tag{3.1.162}$$

Table 3.1.4: Values of the parameters in the equation of state (3.1.163).

a_1	a_2	a_3	a_4	a_5
999.842594	6.793952×10^{-2}	-9.095290×10^{-3}	1.001685×10^{-4}	-1.120083×10^{-6}
a_6	a_7	a_8	a_9	a_{10}
6.536332×10^{-9}	0.824493	-4.0899×10^{-3}	7.6438×10^{-5}	-8.2467×10^{-7}
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
5.3875×10^{-9}	-5.72466×10^{-3}	1.0227×10^{-4}	-1.6546×10^{-6}	4.8314×10^{-4}

where ρ_0, S_0 and T_0 are reference values for the density, salinity and temperature. Uniform values are used for β_S and β_T.

The more general equation of state of seawater defined by the Joint Panel on Oceanographic Tables and Standards (UNESCO, 1981) is selected when IOPTD takes the value of 2. The formulae are simplified by assuming a zero pressure. This approximation is at least valid for depths less than ∼1 km where the turbulence processes occur (see e.g. Gill, 1982). The expressions are written in the form

$$\begin{aligned}\rho(T,S) = {} & a_1 + a_2T + a_3T^2 + a_4T^3 + a_5T^4 + a_6T^5 \\ & +S(a_7 + a_8T + a_9T^2 + a_{10}T^3 + a_{11}T^4) \\ & +S^{3/2}(a_{12} + a_{13}T + a_{14}T^2) + a_{15}S^2 \end{aligned} \tag{3.1.163}$$

$$\begin{aligned}\frac{\partial\rho}{\partial T} = {} & a_2 + 2a_3T + 3a_4T^2 + 4a_5T^3 + 5a_6T^4 \\ & +S(a_8 + 2a_9T + 3a_{10}T^2 + 4a_{11}T^3) \\ & +S^{3/2}(a_{13} + 2a_{14}T)\end{aligned} \tag{3.1.164}$$

$$\begin{aligned}\frac{\partial\rho}{\partial S} = {} & a_7 + a_8T + a_9T^2 + a_{10}T^3 + a_{11}T^4 \\ & +\frac{3}{2}S^{1/2}(a_{12} + a_{13}T + a_{14}T^2) + 2a_{15}S\end{aligned} \tag{3.1.165}$$

$$(\beta_T, \beta_S) = \frac{1}{\rho}\left(\frac{\partial\rho}{\partial T}, \frac{\partial\rho}{\partial S}\right) \tag{3.1.166}$$

where ρ is expressed in kg/m^3, S in PSU and T in ^{0}C. Equations (3.1.164) and (3.1.165) are obviously derived from (3.1.163). Important to note is that evaluating N^2 using (3.1.90), with β_T, β_S determined through equations (3.1.164)–(3.1.166), is numerically more accurate then using the more convential formula

$$N^2 = -\frac{g}{\rho_0}\frac{1}{J}\frac{\partial\rho}{\partial\tilde{x}_3} \tag{3.1.167}$$

Values of the constants a_1 to a_{15} are listed in Table 3.1.4.

1.6 Boundary and initial conditions

This section is organised as follows: surface and bottom boundary conditions are given in Sections III-1.6.1 and III-1.6.2; open sea and land boundaries in Section III-1.6.3; values for the surface drag and exchange coefficients are discussed in Section III-1.6.4 including an option to derive these parameters from Monin-Obukhov similarity theory; the evaluation of the solar radiation flux at the sea surface as function of geographical position and time is discussed in Section III-1.6.5. The program offers the possibility to include the effects of surface waves on the evaluation of the bottom stress. The method follows the theory of Grant and Madsen (1979) and is described in Section III-1.6.6. Initial conditions are finally presented in Section III-1.6.7.

1.6.1 Surface boundary conditions

a) currents

The surface condition for the horizontal current is as usual obtained by specifying the surface stress as function of the wind components

$$\rho_0\frac{\nu_T}{J}\Big(\frac{\partial u}{\partial\tilde{x}_3},\frac{\partial v}{\partial\tilde{x}_3}\Big)=(\tau_{s1},\tau_{s2})=\rho_a C_D^s(U_{10}^2+V_{10}^2)^{1/2}(U_{10},V_{10}) \tag{3.1.168}$$

where (U_{10},V_{10}) are the components of the wind vector at a reference height of 10 m and $\rho_a=1.2$ kg/m^3 is the air density. The boundary condition for the transformed vertical velocity takes the simple form

$$J\tilde{w}=0 \tag{3.1.169}$$

b) temperature and salinity

The surface flux of temperature is given by

$$\frac{\rho_0 c_p}{J}\lambda_T\frac{\partial T}{\partial\tilde{x}_3}=Q_s \tag{3.1.170}$$

where Q_s is the downwards directed heat flux at the surface and c_p the specific heat of seawater at constant pressure. The total heat flux is composed of a term $-Q_{nsol}$ of all non-solar contributions plus the radiative flux Q_{rad}. The program offers two possibilities to include these terms in the model depending on the value of the switch IOPTHE. If IOPTHE is set to 1, solar radiation is assumed to be absorbed at the sea surface so that $Q_s=-Q_{nsol}+Q_{rad}$, while the absorption term in the temperature equation (first term on the right hand side of (3.1.28) or (3.1.53)) is taken to be zero. When IOPTHE equals 2, solar radiation is absorbed in the upper parts of the water column and $Q_s=-Q_{nsol}$.

The parameterisation of the solar heat flux is further discussed in Section III-1.6.5. The (upwards directed) non-solar heat flux has three components, i.e.

$$Q_{nsol} = Q_{la} + Q_{se} + Q_{lw} \tag{3.1.171}$$

where Q_{la} is the latent heat flux released by evaporation, Q_{se} the sensible heat flux due to the turbulent transport of temperature across the air/sea interface and Q_{lw} the net long-wave radiation emitted at the sea surface. The first two terms are related to the turbulent fluxes of humidity and temperature

$$Q_{la} = \rho_a L_\nu C_E |U_{10}|(q_s - q_a) \tag{3.1.172}$$

$$Q_{se} = \rho_a c_{pa} C_H |U_{10}|(T_s - T_a) \tag{3.1.173}$$

where T_s and T_a are the sea surface and air temperature at the reference height and

$$c_{pa} = 1004.6(1 + 0.8375 q_a) \text{ J kg}^{-1} \, (^0\text{C})^{-1} \tag{3.1.174}$$

the specific heat of air at constant pressure. The latent heat of vaporization is given as a function of the sea surface temperature

$$L_\nu = 2.5008 \times 10^6 - 2300 T_s \ \text{ J/kg} \tag{3.1.175}$$

The sea surface and air humidities q_s and q_a are calculated using

$$q = \frac{0.62e}{P_{a0} - 0.38e} \tag{3.1.176}$$

where $P_{a0} = 1013.25$ mb is a reference atmospheric pressure. The vapour pressure e is obtained in mb from the empirical relation (Gill, 1982)

$$\log_{10} e = \log_{10} RH + \frac{0.7859 + 0.03477T}{1 + 0.00412T} \tag{3.1.177}$$

where RH is the relative humidity (between 0 and 1). In equations (3.1.176) and (3.1.177) the humidity q, the vapour pressure e and the temperature T either represent sea surface or atmospheric values at the reference height. Note that a relative humidity of 100% is taken at the sea surface.

The long-wave radiation flux term is parameterised following Gill (1982):

$$Q_{lw} = \varepsilon_s \sigma_{rad} (T_s + 273.15)^4 (0.39 - 0.05 e_a^{1/2})(1 - 0.6 f_c^2) \tag{3.1.178}$$

where $\varepsilon_s = 0.985$ is the emissivity at the sea surface, $\sigma_{rad} = 5.67 \times 10^{-8}$ W m^{-2} K^{-4} Stefan's constant, f_c the fractional cloud cover (between 0 and 1) and e_a the vapour pressure evaluated by (3.1.177).

The surface fluxes of momentum and heat involve the surface drag coefficient C_D^s and two dimensionless parameters C_E, C_H sometimes referred as the Dalton and Stanton number. Various empirical schemes for these transfer coefficients have been presented in the literature (see e.g. Blanc, 1985; Geernaert, 1990). A few formulations, selected with the switches IDRAG and ITDIF are available in the program. This is further discussed in Section III-1.6.4.

The surface salinity flux is determined using the formula given by Steinhorn (1991):

$$\rho_0 \frac{\lambda_T}{J}\frac{\partial S}{\partial \tilde{x}_3} = \frac{S_s(E_{vap} - R_{pr})}{1 - 0.001 S_s} \tag{3.1.179}$$

where $E_{vap} = Q_{la}/L_\nu$ and R_{pr} are the evaporation and precipitation rates in kg m^{-2} s^{-1} and S_s the surface salinity in PSU. The evaluation of the surface salinity flux requires the input of precipitation data which are often not available or given by a long term average. The program therefore allows the evaluation of the right hand side of (3.1.179) in two different ways depending on the value of the switches IOPTSA and IOPTM. The surface flux is set to zero if IOPTM = 0 or 1. When IOPTSA = 1 and IOPTM > 1, values of $E_{vap} - R_{pr}$ need to be supplied by the user. When IOPTSA = 2 and IOPTM = 2 or 3, only R_{pr} is supplied by the user while the program evaluates E_{vap} from (3.1.172) by dividing Q_{la} through L_ν. For more details see Table 3.1.5.

c) turbulence

To solve the turbulence transport equations (3.1.114), (3.1.125) and (3.1.128) boundary conditions must be supplied at the surface and the bottom. They are obtained from the "wall-layer" approximation. Assuming a uniform shear stress, equilibrium between production and dissipation of turbulence and the boundary layer form (3.1.130) for the mixing length one has

$$k = u_{*s}^2/\varepsilon_0^{2/3}\ ,\ kl = u_{*s}^2 l_2/\varepsilon_0^{2/3}\ ,\ \varepsilon = k^{3/2}\varepsilon_0/l_2 \tag{3.1.180}$$

with l_2 defined by (3.1.130). In the numerical scheme the boundary condition for k is applied at the surface, those for kl and ε one grid distance away from the surface.

As pointed out by Craig and Banner (1994) and Craig (1996) the equilibrium assumption in the surface boundary layer is no longer valid if surface wave effects are taken into account. They proposed to replace the surface boundary condition for k by a flux condition. The input of wave energy is represented using a surface flux of turbulent energy proportional to the cube of the surface friction velocity u_{*s}. Although better results have been obtained with this new condition (e.g. Stacey and Pond, 1997), the problem of relating the proportionality factor and the surface roughness length to the characteristics of the wave field, remains unsolved. Implementation of surface wave effects can be considered in a future release of the program.

d) generalised surface boundary conditions

Most of the surface boundary conditions discussed in this section and the ones used in the biological, sediment and contaminant modules are Neumann type conditions for the surface fluxes and are written into the more general form

$$\frac{\lambda_T^\psi}{J}\frac{\partial\psi}{\partial\tilde{x}_3} = F_{s0}^\psi + F_{s1}^\psi(\psi_{s,e} - \psi_{s,i}) \tag{3.1.181}$$

where ψ either represents one of the horizontal current components or a general scalar quantity, λ_T^ψ the vertical diffusion coefficient appropriate for the quantity ψ (see Section III-1.1.3), F_{s0}^ψ a prescribed surface flux, F_{s1}^ψ a transfer coefficient having the dimension of m/s, $\psi_{s,e}$ a specified external value (above the surface) and $\psi_{s,i}$ the calculated value of ψ at the first grid point below the surface. For example, the surface conditions (3.1.168) for u and v take the form (3.1.181) with

$$F_{s0}^u = \tau_{s1}\ ,\ F_{s0}^v = \tau_{s2}\ ,\ F_{s1}^u = F_{s1}^v = 0 \tag{3.1.182}$$

A second form of surface boundary condition is the Dirichlet type where the value of ψ at the surface or at the first interior grid point is specified. Examples are the conditions (3.1.180) for turbulence.

1.6.2 Bottom boundary conditions

a) currents

A slip boundary condition is applied for the horizontal current at the bottom which takes the form

$$\frac{\rho_0\nu_T}{J}\Big(\frac{\partial u}{\partial\tilde{x}_3},\frac{\partial v}{\partial\tilde{x}_3}\Big) = (\tau_{b1},\tau_{b2}) \tag{3.1.183}$$

The following formulations, selected with the switch IBSTR, can be used in the program

- zero stress condition

$$(\tau_{b1},\tau_{b2}) = (0,0) \tag{3.1.184}$$

- quadratic friction law

$$(\tau_{b1},\tau_{b2}) = \rho_0 C_D^b(u_b^2+v_b^2)^{1/2}(u_b,v_b) \tag{3.1.185}$$

- linear friction law

$$(\tau_{b1},\tau_{b2}) = \rho_0 k_{lin}(u_b,v_b) \tag{3.1.186}$$

The bottom velocities (u_b,v_b) are evaluated at the grid point nearest to the bottom. A constant value is taken for the linear friction coefficient k_{lin}. In the boundary layer approximation a vertically uniform shear stress is assumed yielding a logarithmic profile for the current. The quadratic bottom drag coefficient can then be expressed as a function of the roughness length z_0 and the vertical grid spacing. This gives

$$C_D^b = \Big(\kappa/\ln(z_r/z_0)\Big)^2 \tag{3.1.187}$$

where z_r is a reference height taken at the grid centre of the bottom cell. The value of z_0 which may vary in the horizontal directions, depends on the geometry and composition of the seabed. Values of z_0, measured from logarithmic current profiles can be found in Heathershaw (1981) and Soulsby (1983) for various bed type forms. Enhancement of the bottom stress and friction coefficient through wave-current interaction is available in the program using a separate option. This is further discussed in Section III-1.6.6.

In analogy with the surface condition (3.1.169) the bottom value of the transformed vertical velocity equals zero, i.e.

$$J\tilde{w} = 0 \tag{3.1.188}$$

b) temperature and salinity

The bottom boundary conditions for temperature and salinity are obtained by considering a zero flux normal to the seabed:

$$\frac{\rho_0\lambda_T}{J}\frac{\partial T}{\partial \tilde{x}_3} = 0 \ , \ \frac{\rho_0\lambda_T}{J}\frac{\partial S}{\partial \tilde{x}_3} = 0 \tag{3.1.189}$$

A similar assumption is applied for the absorption term in the temperature equation (3.1.28) or (3.1.53) where I is taken to be zero at the sea bottom. It is remarked that the non-allowance of any heat exchange at the bottom interface may not be realistic but is only imposed in the absence of a useful parameterisation which takes account of a bottom exchange.

c) turbulence

The boundary conditions for the turbulent variables are derived from the "wall-layer" approximation and are similar to the ones applied at the surface

$$k = u_{*b}^2/\varepsilon_0^{2/3} \ , \ kl = u_{*b}^2 l_1/\varepsilon_0^{2/3} \ , \ \varepsilon = k^{3/2}\varepsilon_0/l_1 \tag{3.1.190}$$

where l_1 is defined by (3.1.130). The bottom boundary condition for k is applied at the bottom, those for kl and ε one grid distance away from the bottom.

d) generalised bottom boundary condition

In analogy with Section III-1.6.1d most of the bottom boundary conditions discussed in this section and the ones used in the biological, sediment and contaminant modules are Neumann type conditions for the bottom flux and can be written into the more general form

$$\frac{\lambda_T^\psi}{J}\frac{\partial \psi}{\partial \tilde{x}_3} = -F_{b0}^\psi + F_{b1}^\psi(\psi_{b,i} - \psi_{b,e}) \tag{3.1.191}$$

where ψ represents one of the horizontal current components or a general scalar quantity, λ_T^ψ the vertical diffusion coefficient appropriate for a quantity ψ (see Section III-1.1.3), F_{b0}^ψ a prescribed bottom flux, F_{b1}^ψ a transfer coefficient having the dimension of m/s, $\psi_{b,e}$ a specified external value (below the bottom) and $\psi_{b,i}$ the calculated value of ψ at the first grid point above the bottom. For example, the bottom conditions (3.1.183) and (3.1.185) for u and v are of the form (3.1.191) with

$$\begin{aligned} &F^u_{b0} = F^v_{b0} = 0 \\ &F^u_{b1} = F^v_{b1} = C^b_D(u_b^2 + v_b^2)^{1/2} \\ &u_{b,e} = v_{b,e} = 0 \end{aligned} \tag{3.1.192}$$

An alternative form is a Dirichlet boundary condition where the value of ψ at the bottom or the first interior point is specified. Examples are the conditions (3.1.190) for turbulence.

1.6.3 Open sea, river and land boundary conditions

The model equations are solved on an Arakawa C-grid[3]. Each grid cell has six lateral boundary faces. For convenience, the following notation is adopted in this section. The boundary, perpendicular to the x_1-axis and located in the positive (negative) x_1-direction with respect to the cell centre is denoted by the "eastern" ("western") boundary. In a similar notation the two boundaries, perpendicular to the x_2-axis are denoted by the "northern" and "southern" boundary. The two remaining boundaries are obviously the lower and upper boundary. The notion of "eastern", "western", "southern" and "northern" boundaries becomes obvious in the case of spherical coordinates.

To each vertical array of grid cells an index is attributed taking the value of 0 (dry) or 1 (wet). More details are given in Section IV-2.3. Assume that the simulated area is situated between $x_1^w \le x_1 \le x_1^e$ and $x_2^s \le x_2 \le x_2^n$. The boundary faces of a cell have one of the four following attributes:

- Open sea boundary faces are always located along one of the four vertical planes ($x_1 = x_1^w, x_1^e$ and $x_2 = x_2^s, x_2^n$) which delineate the simulated area.
- Interior boundaries are inside the domain and separate two wet cells.
- Land boundaries are either situated along a boundary plane or in the interior domain where at least one of the two neighbouring cells must have the "dry" attribute.
- River boundaries are either situated along one of the four domain boundaries or in the interior domain at the interface between a "wet" and a "dry" cell. The difference with land boundaries is that the latter are considered as impregnable walls whereas the former constitute the "head" of a river where a specific boundary condition (e.g. river discharge) is applied.

The open boundary conditions, discussed below, are sufficiently general so that no formal distinction needs to be made between open sea and river boundaries despite their different nature.

[3]The model grid is further discussed in Section III-4.2.1.

a) open boundary conditions for the 2-D mode

Open sea (or river) boundary conditions need to be supplied for $\overline{U}$ at western and eastern boundaries, and for $\overline{V}$ at southern and northern boundaries.

A selection can be made between different types of open boundary conditions. They have the form of a radiation condition derived using the method of characteristics (Hedstrom, 1979; Røed and Cooper, 1987; Ruddick, 1995). This is based on the integration of the equations for the incoming and outgoing Riemann variables

$$(R^u_\pm,\, R^v_\pm) = (\overline{U} \pm c\zeta,\, \overline{V} \pm c\zeta) \tag{3.1.193}$$

where $c = (gH)^{1/2}$ is the wave speed of the barotropic mode. The characteristic equations are obtained by multiplying the continuity equation (3.1.33) with $\pm c$ and adding the depth-integrated momentum equations (3.1.34) or (3.1.35). This gives

$$\begin{aligned}\Big(\frac{\partial}{\partial t} \pm c\frac{\partial}{\partial x_1}\Big)R^u_\pm &= \mp c\frac{\partial\overline{V}}{\partial x_2} + f\overline{V} - \frac{\partial}{\partial x_1}\Big(\frac{\overline{U}^2}{H}\Big) - \frac{\partial}{\partial x_2}\Big(\frac{\overline{V}\,\overline{U}}{H}\Big)\\ &\quad - \frac{H}{\rho_0}\frac{\partial P_a}{\partial x_1} + \overline{Q}_1 + \frac{1}{\rho_0}(\tau_{s1} - \tau_{b1}) + \frac{\partial}{\partial x_1}\overline{\tau_{11}} + \frac{\partial}{\partial x_2}\overline{\tau_{21}} - \overline{A}^h_1 + \overline{D}^h_1\end{aligned} \tag{3.1.194}$$

$$\begin{aligned}\Big(\frac{\partial}{\partial t} \pm c\frac{\partial}{\partial x_2}\Big)R^v_\pm &= \mp c\frac{\partial\overline{U}}{\partial x_1} - f\overline{U} - \frac{\partial}{\partial x_1}\Big(\frac{\overline{U}\,\overline{V}}{H}\Big) - \frac{\partial}{\partial x_2}\Big(\frac{\overline{V}^2}{H}\Big)\\ &\quad - \frac{H}{\rho_0}\frac{\partial P_a}{\partial x_2} + \overline{Q}_2 + \frac{1}{\rho_0}(\tau_{s2} - \tau_{b2}) + \frac{\partial}{\partial x_1}\overline{\tau_{12}} + \frac{\partial}{\partial x_2}\overline{\tau_{22}} - \overline{A}^h_2 + \overline{D}^h_2\end{aligned} \tag{3.1.195}$$

where it is assumed that the wave speed c has a zero gradient normal to the boundary and does not vary in time. In the case of spherical coordinates (3.1.194)-(3.1.195) are replaced by

$$\begin{aligned}\Big(\frac{\partial}{\partial t} \pm \frac{c}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big)R^u_\pm &= \mp\frac{c}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{V}) + 2\Omega\sin\phi\,\overline{V}\\ &\quad - \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\frac{\overline{U}^2}{H}\Big) - \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\cos\phi\frac{\overline{V}\,\overline{U}}{H}\Big) + \frac{\tan\phi}{R}\frac{\overline{U}\,\overline{V}}{H} - \frac{H}{\rho_0 R\cos\phi}\frac{\partial P_a}{\partial\lambda} + \overline{Q}_\lambda\\ &\quad + \frac{1}{\rho_0}(\tau_{s\lambda} - \tau_{b\lambda}) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\overline{\tau_{\lambda\lambda}} + \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{\tau_{\phi\lambda}}) - \frac{\tan\phi}{R}\overline{\tau_{\phi\lambda}} - \overline{A}^h_\lambda + \overline{D}^h_\lambda\end{aligned} \tag{3.1.196}$$

$$\begin{aligned}\Big(\frac{\partial}{\partial t} \pm \frac{c}{R}\frac{\partial}{\partial\phi}\Big)R^v_\pm &= \mp\frac{c}{R\cos\phi}\frac{\partial\overline{U}}{\partial\lambda} \pm c\frac{\tan\phi}{R}\overline{V} - 2\Omega\sin\phi\,\overline{U}\\ &\quad - \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\frac{\overline{U}\,\overline{V}}{H}\Big) - \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\cos\phi\frac{\overline{V}^2}{H}\Big) - \frac{\tan\phi}{R}\frac{\overline{U}^2}{H} - \frac{H}{\rho_0 R}\frac{\partial P_a}{\partial\phi} + \overline{Q}_\phi\\ &\quad + \frac{1}{\rho_0}(\tau_{s\phi} - \tau_{b\phi}) + \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\overline{\tau_{\lambda\phi}} + \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}(\cos\phi\,\overline{\tau_{\phi\phi}}) + \frac{\tan\phi}{R}\overline{\tau_{\lambda\lambda}} - \overline{A}^h_\phi + \overline{D}^h_\phi\end{aligned} \tag{3.1.197}$$

The analysis below will be restricted to West/East boundaries but can easily be extended to South/North boundaries by substituting "South" for "West", "North" for "East", $\overline{V}$ for $\overline{U}$, $R^v_\pm$ for $R^u_\pm$ and (x_2, ϕ) for (x_1, λ).

The method consists in solving (3.1.194) for the outgoing characteristic either given by R^u_+ or R^u_-. The input of residual and tidal forcing data at the open boundary is performed by a specification for the incoming characteristic using an harmonic expansion of the form

$$F_{har}(t, x_1, x_2) = A_0(x_1, x_2) + \sum_{n=1}^{N_T} A_n \cos(\omega_n t + \varphi_{n0} - \varphi_n(x_1, x_2)) \tag{3.1.198}$$

where A_0 represents the residual input (river discharge, Ekman transport, ...) in m, A_n are the tidal amplitudes in m, ω_n the tidal frequencies in rad/s, φ_{n0} the initial value of the phase $\omega_n t + \varphi_{n0}$ in radians, φ_n the tidal phases in radians at each open boundary location, t the time in seconds elapsed since the start of the program and N_T the number of tidal constituents. Once R^u_+ and R^u_- are known, the depth integrated current is obtained using

$$\overline{U} = \frac{1}{2}(R^u_+ + R^u_-) \tag{3.1.199}$$

The procedures are slightly different when applied at a western or an eastern boundary. The two cases are discussed separately below.

At a western boundary equation (3.1.194) or (3.1.196) is solved for the outgoing characteristic R^u_-. The incoming Riemann variable R^u_+ is determined by selecting one of the four following formulations.

1. An harmonic expansion is applied for R^u_+, i.e.

$$R^u_+ = \overline{U}_{in} + c\zeta_{in} = 2cF_{har} \tag{3.1.200}$$

   where $\overline{U}_{in}$ and ζ_{in} are input data in harmonic form. This method is recommended but requires that both current and surface elevation are available.

2. A zero gradient condition is applied. In this case R^u_+ is obtained from (3.1.194) or (3.1.196) after substituting

$$\frac{\partial R^u_+}{\partial x_1} = 0 \quad \text{or} \quad \frac{\partial R^u_+}{\partial \lambda} = 0 \tag{3.1.201}$$

   This method could be used if no input data are available.

3. Input data are available for ζ_{in} but not for $\overline{U}_{in}$. Writing ζ_{in} in harmonic form, R^u_+ is obtained using

$$\zeta_{in} = F_{har} \ , \ R^u_+ = R^u_- + 2c\zeta_{in} = R^u_- + 2cF_{har} \tag{3.1.202}$$

4. Input data are available for $\overline{U}_{in}$ but not for ζ_{in}. An harmonic expansion is applied for $\overline{U}_{in}$ so that R^u_+ is determined by

$$\overline{U}_{in} = cF_{har} \ , R^u_+ = 2\overline{U}_{in} - R^u_- = 2cF_{har} - R^u_- \tag{3.1.203}$$

Note that in formulations one, three and four the harmonic function F_{har} is expressed in meters but has a different meaning for each case.

The procedure is analogous at an eastern boundary with the role of R_+^u and R_-^u interchanged and c replaced by $-c$ and is given below to avoid any ambiguity for the user. The outgoing Riemann variable R_+^u is obtained by solving equation (3.1.194) or (3.1.196) whereas the incoming characteristic R_-^u is determined using one of the following formulations

1. R_-^u is prescribed in the harmonic form

$$R_-^u = \overline{U}_{in} - c\zeta_{in} = -2cF_{har} \tag{3.1.204}$$

2. The normal gradient of R_-^u is set to zero in which case (3.1.194) or (3.1.196) is solved for R_-^u using

$$\frac{\partial R_-^u}{\partial x_1} = 0 \quad \text{or} \quad \frac{\partial R_-^u}{\partial \lambda} = 0 \tag{3.1.205}$$

3. R_-^u is obtained by prescribing ζ_{in} harmonically

$$\zeta_{in} = F_{har}\ ,\ R_-^u = R_+^u - 2c\zeta_{in} = R_+^u - 2cF_{har} \tag{3.1.206}$$

4. R_-^u is determined using an harmonic prescription for $\overline{U}_{in}$:

$$\overline{U}_{in} = -cF_{har}\ ,\ R_-^u = 2\overline{U}_{in} - R_+^u = -2cF_{har} - R_+^u \tag{3.1.207}$$

The program allows different types of boundary conditions at different open boundary locations. For example, the first type given by (3.1.200) or (3.1.204) is recommended at open sea boundaries whereas the fourth one given by (3.1.203) or (3.1.207) is more appropriate at river boundaries. In the latter case the residual part of $\overline{U}_{in}$ is given by $\pm Q_d/\Delta x_2$ where Q_d is the river discharge in m^3/s and the +(-) sign applies at western (eastern) river boundaries. The residual component A_0 has no time variation in the default version of the program. Time dependence can however be included (e.g. varying river discharge) in which case residual input data need to be supplied externally to the program at selected time intervals (see Chapter IV-2 for more details). The open boundary conditions for the 2-D mode are further discussed in Section III-4.3.11b (numerical aspects) and IV-2.4 (model setup).

b) open boundary conditions for the 3-D mode equations

At an open boundary the values of the 2-D current are calculated prior to the 3-D current. The boundary condition for the 3-D horizontal current can then conveniently be formulated in the form of a prescription for the deviation of the current from its depth-averaged value. The components (u', v') of the velocity deviation are defined by (3.1.41).

Two different formulations can be used in the program. The first one, recommended by Deleersnijder et al. (1992), simply imposes a zero gradient, normal to the boundary. This takes the form

$$\frac{\partial}{\partial x_1}(Ju') = 0 \quad \text{or} \quad \frac{\partial}{\partial \lambda}(Ju') = 0 \tag{3.1.208}$$

at western/eastern and

$$\frac{\partial}{\partial x_2}(Jv') = 0 \quad \text{or} \quad \frac{\partial}{\partial \phi}(J\cos\phi\, v') = 0 \tag{3.1.209}$$

at southern/northern boundaries. Substituting (3.1.208)–(3.1.209) into the continuity equation (3.1.46) or (3.1.68) it is seen that this particular form of boundary condition avoids spurious upwellings and downwellings which may be produced when the boundary value of the horizontal current deviates from the values calculated in the interior of the simulated area.

The second formulation, more appropriate at a river boundary, uses a prescribed vertical profile which remains fixed in time

$$u' = u'_0(x_1, x_2, \tilde{x}_3) \quad \text{or} \quad v' = v'_0(x_1, x_2, \tilde{x}_3) \tag{3.1.210}$$

By default a zero normal gradient condition is applied at all open boundaries. Different type of conditions can be selected at different open boundary locations.

Scalar quantities are always evaluated inside the computational domain. Boundary conditions are however required for the advective and diffusive fluxes at the open boundaries. As for the 3-D current a “scalar” or a “gradient” condition can be selected. The advective fluxes of a quantity ψ at open boundaries are determined with the upwind scheme. This means that an exterior value ψ_{ext} — considered as an approximate value of ψ – located one half grid distance outside the domain, must be available within the program when an inflow condition occurs.

In the case of a scalar condition a vertical value for $\psi_{ext}(x_1, x_2, \tilde{x}_3)$ which is fixed in time by default, must be provided. The advective and diffusive fluxes are then given by

$$Ju_n\psi = \frac{1}{2}Ju_n\Big((1+p)\psi_{ext} + (1-p)\psi_{int}\Big) \tag{3.1.211}$$

$$\lambda_H\frac{\partial\psi}{\partial n} = \lambda_H\frac{\psi_{int} - \psi_{ext}}{\Delta n} \tag{3.1.212}$$

where u_n is the current component normal to the boundary (i.e. equal to u at western/eastern boundaries and to v at southern/northern boundaries), ψ_{int} is the interior value of ψ calculated by the program at one-half grid distance inside the boundary, $p = \pm\text{Sign}(u_n)$ and Δn equals $\pm$ the horizontal grid spacing normal to the boundary with the “+” sign taken at western/southern boundaries and the “-” sign at eastern/northern boundaries.

When a gradient condition is applied, ψ_{ext} is determined by imposing a zero normal gradient for ψ at the open boundary point so that $\psi_{ext} = \psi_{in}$. Hence

$$Ju_n\psi = Ju_n\psi_{int} \tag{3.1.213}$$

$$\lambda_H \frac{\partial \psi}{\partial n} = 0 \tag{3.1.214}$$

As before a scalar or a gradient condition can be applied at different open boundary locations and even at different vertical positions. For example, a typical condition for salinity at a river boundary can be determined by specifying a two-layer stratification whereby a fresh water value is prescribed within a surface layer with a user-defined depth, while a zero gradient condition can be applied in the bottom layer.

When the turbulent transport equations (3.1.114), (3.1.125) or (3.1.128) are solved with advective terms included (IAHDHT = 1), a zero normal gradient condition is always applied so that no specific boundary condition must be provided by the user.

c) coastal boundaries

Coastal boundaries are considered as impregnable walls. This means that all currents, advective and diffusive fluxes are set to zero

$$\overline{U} = 0 \ , \ u = 0 \ , \ Ju\psi = 0 \ , \ \lambda_H \frac{\partial \psi}{\partial x_1} = 0 \tag{3.1.215}$$

at western/eastern boundaries and

$$\overline{V} = 0 \ , \ v = 0 \ , \ Jv\psi = 0 \ , \ \lambda_H \frac{\partial \psi}{\partial x_2} = 0 \tag{3.1.216}$$

at southern/northern boundaries.

1.6.4 Surface drag and exchange coefficients

This section describes the different formulations available in the program for the evaluation of the transfer coefficients C_D^s, C_E, C_H appearing in the expressions (3.1.168), (3.1.172), (3.1.173) for the momentum, latent and sensible heat fluxes at the surface. A selection can be made with the switches ITDIF and IDRAG.

If ITDIF equals 1, stratification effects are ignored. The Dalton and Stanton numbers have constant values

$$C_E = 0.00113 \ , \ C_H = 0.00113 \tag{3.1.217}$$

Measurements (e.g. Anderson and Smith, 1981) indicate that these coefficients depend strongly on the wind speed but the data show a large amount of scatter. The values (3.1.217) should therefore be considered as a rough estimate. The surface drag coefficient C_D^s is determined as a function of the wind speed using one of the five available empirical relations selected by IDRAG.

- constant value

$$C_D^s = 0.0013 \tag{3.1.218}$$

- Large and Pond (1981)

$$\begin{aligned} C_D^s &= 0.0012 \qquad \text{if} \qquad |U_{10}| < 11\text{m/s} \\ C_D^s &= 10^{-3}(0.49 + 0.065|U_{10}|) \qquad \text{if} \qquad |U_{10}| \geq 11\text{m/s} \end{aligned} \tag{3.1.219}$$

- Smith and Banke (1975)

$$C_D^s = 10^{-3}(0.63 + 0.066|U_{10}|) \tag{3.1.220}$$

- Geernaert et al. (1986)

$$C_D^s = 10^{-3}(0.43 + 0.097|U_{10}|) \tag{3.1.221}$$

- Charnock (1955)

$$\ln(z_a g/(a|U_{10}|^2)) - \ln C_D^s = \kappa/(C_D^s)^{1/2} \tag{3.1.222}$$

  where $z_a = 10$ m is the reference height and $a = 0.014$ (Charnock's constant).

The second and third relations were derived from ocean measurements, the fourth from flux measurements in the North Sea. An iteration scheme is applied to solve C_D^s in Charnock's expression. The five relations are plotted in Figure 3.1.3. Note the high similarity between the Smith-Banke and Charnock formulations. Several other drag relations are reported in the literature and may be implemented in the program. An overview can be found in Geernaert (1990) and Blanc (1985).

If the switch ITDIF is set to 1, the transfer coefficients are determined as a function of wind speed and stratification. The method is based on the Monin-Obukhov similarity theory as described in Geernaert (1990) and Luyten and De Mulder (1992). Some details of the physical theory, also presented in the latter report, are given here to understand how it is implemented in the program.

The surface values of the momentum, latent and sensible heat fluxes at the air-sea interface depend on the turbulent structure of the lower atmosphere, usually called the planetary boundary layer. It is generally assumed that the fluxes of momentum, heat and specific humidity have nearly constant values within this layer. Following Monin and Obukhov (1954) the structure of this layer can be described by means of a velocity scale u_* (friction velocity), a temperature scale T_* and a humidity scale q_* defined by

$$u_* = (\langle u'w'\rangle^2 + \langle v'w'\rangle^2)^{1/2} \tag{3.1.223}$$

$$u_* T_* = -\langle w'T'\rangle \tag{3.1.224}$$

$$u_* q_* = -\langle w'q'\rangle \tag{3.1.225}$$

where (u', v', w'), T' and q' are the turbulent fluctuations of the wind velocity, temperature and humidity, and $\langle\ \rangle$ denotes an ensemble average. The vertical gradient of the windspeed U is written as the ratio of u_* to a mixing length, or

$$\frac{\partial U}{\partial z} = \frac{u_*}{l} \tag{3.1.226}$$

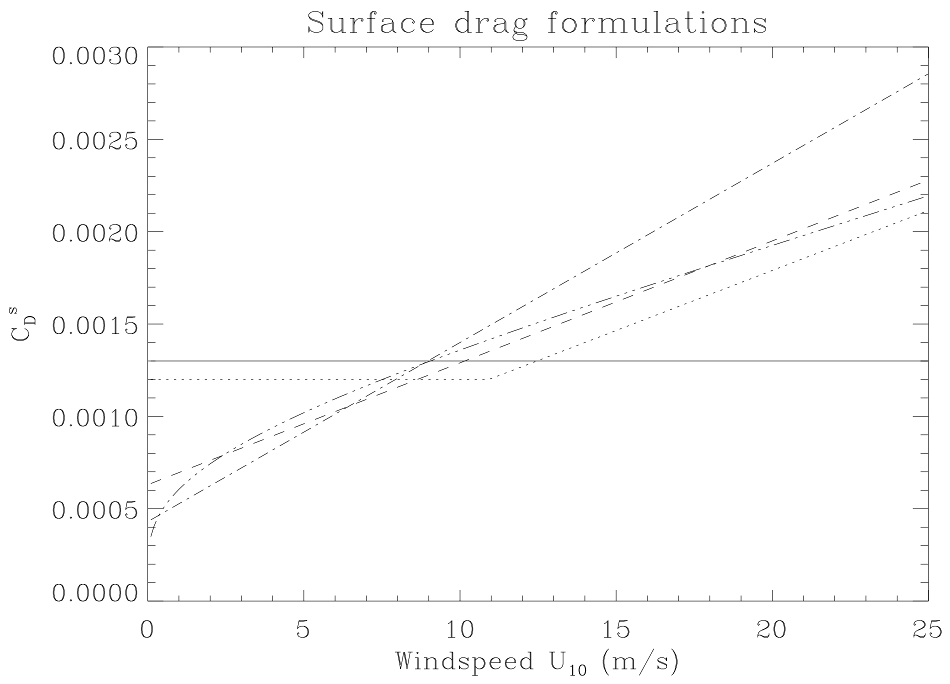


Figure 3.1.3: Formulations for the (neutral) surface drag coefficient: constant (solid), Large and Pond (dots), Smith and Banke (dashes), Geernaert et al. (dash-dots), Charnock (dash and 3 dots).

where z is the height above the sea surface. For neutral stratification one has

$$l = \kappa z \tag{3.1.227}$$

Since l decreases or increases with respect to its neutral value, according as the stratification is stable or unstable, equation (3.1.226) can be rewritten in the more general form

$$\frac{\partial U}{\partial z} = \frac{u_*}{\kappa z}\phi_m \tag{3.1.228}$$

The dimensionless function ϕ_m describes the effect of stratification and is smaller (or larger) than 1 for unstable (stable) stratification. In a similar way, the gradients of temperature and relative humidity are given by

$$\frac{\partial T}{\partial z} = \frac{T_*}{\kappa z}\phi_h \tag{3.1.229}$$

$$\frac{\partial q}{\partial z} = \frac{q_*}{\kappa z}\phi_q \tag{3.1.230}$$

The functions ϕ_m, ϕ_h and ϕ_q are expressed in terms of the dimensionless height $\xi = z/L_{mo}$ with the Monin-Obukhov length L_{mo} defined by

$$\frac{1}{L_{mo}} = -\frac{g\kappa}{T_v u_*^3}(\langle w'T'\rangle + 0.61 T_k \langle w'q'\rangle) \tag{3.1.231}$$

where T_k represents the temperature in degrees Kelvin and the virtual temperature T_v is given by

$$T_v = T_k(1 + 0.61q) \tag{3.1.232}$$

Note that $\xi > 0$ for stable and $\xi < 0$ for unstable stratification. Based upon atmospheric measurements (e.g. Businger et al., 1971) the following parameterisations are adopted

$$\begin{array}{ll} \phi_m = (1 - \alpha\xi)^{-1/4} & \text{for} \quad \xi < 0 \\ \phi_m = 1 + \beta\xi & \text{for} \quad \xi > 0 \end{array} \tag{3.1.233}$$

and

$$\begin{array}{ll} \phi_h = \phi_m^2 & \text{for} \quad \xi < 0 \\ \phi_h = \phi_m & \text{for} \quad \xi > 0 \end{array} \tag{3.1.234}$$

with $\alpha = 16$, $\beta = 5$, while it is further assumed that $\phi_q = \phi_h$. Integrating (3.1.228)–(3.1.230) one obtains

$$U = \frac{u_*}{\kappa}(\ln\frac{z}{z_{0U}} - \psi_m) \tag{3.1.235}$$

$$T - T_s = \frac{T_*}{\kappa}(\ln\frac{z}{z_{0T}} - \psi_h) \tag{3.1.236}$$

$$q - q_s = \frac{q_*}{\kappa}(\ln\frac{z}{z_{0q}} - \psi_h) \tag{3.1.237}$$

where

$$\psi_m = \int_0^\xi \frac{1 - \phi_m(\xi)}{\xi}\, d\xi \tag{3.1.238}$$

$$\psi_h = \int_0^\xi \frac{1 - \phi_h(\xi)}{\xi}\, d\xi \tag{3.1.239}$$

The subscript s indicates a quantity evaluated at the sea surface. Expressions (3.1.235)–(3.1.237) are valid for $z \gg z_{0U}, z_{0T}, z_{0q}$ while it is further assumed that $U \gg U_s$. The roughness lengths z_{0U}, z_{0T} and z_{0q} prevent the quantities becoming too large near the surface. Evaluating the integrals (3.1.238)–(3.1.239) one has

$$\begin{array}{ll} \psi_m = 2\ln(1 + \phi_m^{-1}) + \ln(1 + \phi_m^{-2}) - 2\arctan(\phi_m^{-1}) + \frac{\pi}{2} - 3\ln 2 & \text{for} \quad \xi < 0 \\ \psi_m = 1 - \phi_m & \text{for} \quad \xi > 0 \end{array} \tag{3.1.240}$$

and

$$\begin{array}{ll} \psi_h = 2\ln(1 + \phi_h^{-1}) - 2\ln 2 & \text{for} \quad \xi < 0 \\ \psi_h = 1 - \phi_h & \text{for} \quad \xi > 0 \end{array} \tag{3.1.241}$$

Parameterisations for u_*, T_*, q_* are given by the following equations, in which (3.1.235)–(3.1.237) have been substituted

$$u_*^2 = \kappa^2(\ln\frac{z}{z_{0U}} - \psi_m)^{-2}U^2 = C_D^s U^2 \tag{3.1.242}$$

$$u_*T_* = \kappa^2(\ln\frac{z}{z_{0U}} - \psi_m)^{-1}(\ln\frac{z}{z_{0T}} - \psi_h)^{-1}U(T - T_s) = C_H U(T - T_s) \tag{3.1.243}$$

$$u_* q_* = \kappa^2 (\ln \frac{z}{z_{0U}} - \psi_m)^{-1} (\ln \frac{z}{z_{0q}} - \psi_h)^{-1} U(q - q_s) = C_E U(q - q_s) \qquad (3.1.244)$$

where C_D^s, C_H and C_E are the drag coefficient, the Stanton and the Dalton numbers used previously in the expressions (3.1.168), (3.1.172), (3.1.173) for the momentum, latent and sensible heat flux. The z-dependence of the coefficients is usually eliminated by evaluating them at the standard height $z = z_a = 10$ m. Suitable expressions are required for z_0, z_{0T}, z_{0q}. Parameterisations exist for the roughness length as function of the wave state (e.g. Janssen, 1991). However, since little information is available concerning the form of z_{0T} and z_{0q}, the simpler approach described in Geernaert (1990) is used. This consists in defining a drag coefficient C_{DN} valid for a neutral stratification. In analogy with (3.1.242) one has

$$U = C_{DN}^{-1/2} u_{*N} = \frac{u_{*N}}{\kappa} \ln \frac{z}{z_{0N}} \qquad (3.1.245)$$

Eliminating z between (3.1.242) and (3.1.245) yields the following relation between C_D^s and C_{DN}:

$$C_D^s = [C_{DN}^{-1/2} + (\ln \frac{z_{0N}}{z_{0U}} - \psi_m)/\kappa]^{-2} \qquad (3.1.246)$$

Following Charnock (1955) one further assumes that the roughness length scales with the wind stress or[4]

$$z_{0U} = a u_*^2 / g \qquad (3.1.247)$$

so that

$$\frac{z_{0U}}{z_{0N}} = \frac{u_*^2}{u_{*N}^2} = \frac{C_D^s}{C_{DN}} \qquad (3.1.248)$$

Hence

$$C_D^s = [C_{DN}^{-1/2} + (\ln(C_{DN}/C_D^s) - \psi_m)/\kappa]^{-2} \qquad (3.1.249)$$

The neutral Stanton and Dalton numbers are defined in a similar way

$$C_{HN} = \kappa^2 (\ln \frac{z}{z_{0N}} \ln \frac{z}{z_{0T}})^{-1} \qquad (3.1.250)$$

$$C_{EN} = \kappa^2 (\ln \frac{z}{z_{0N}} \ln \frac{z}{z_{0q}})^{-1} \qquad (3.1.251)$$

Expressions for C_H and C_E in terms of their neutral counterparts are then obtained by combining (3.1.250)–(3.1.251) with (3.1.243)–(3.1.244) (taking $z_{0U} \simeq z_{0N}$ for simplicity) and (3.1.245). This gives

$$C_H = C_{HN} [1 - (\psi_m C_{DN}^{1/2} + \psi_h C_{HN} C_{DN}^{-1/2})/\kappa + C_{HN} \psi_m \psi_h / \kappa^2]^{-1} \qquad (3.1.252)$$

$$C_E = C_{EN} [1 - (\psi_m C_{DN}^{1/2} + \psi_h C_{EN} C_{DN}^{-1/2})/\kappa + C_{EN} \psi_m \psi_h / \kappa^2]^{-1} \qquad (3.1.253)$$

The neutral coefficients are obtained from (3.1.217) and one of the expressions (3.1.218)–(3.1.222) selected by IDRAG.

[4]Note that the Charnock's formulation (3.1.222) is recovered by combining (3.1.242) and (3.1.247).

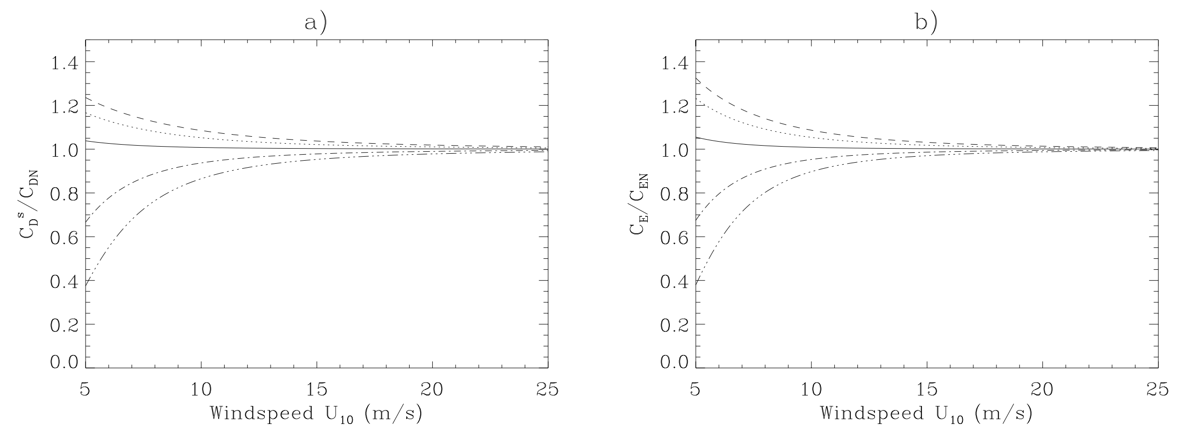


Figure 3.1.4: Evolution of the surface drag coefficient (a) and the heat exchange coefficient (b) measured with respect to their neutral values for different values of the air-sea temperature difference: $\Delta T = 0$ (solid), $\Delta T = -2$ (dots), $\Delta T = -4$ (dashes), $\Delta T = 2$ (dash-dots), $\Delta T = 4$ (dash and 3 dots).

Since the functions ψ_m and ψ_h depend on the dimensionless height ξ, a further equation needs to be added. Eliminating u_*, T_* and q_* in (3.1.231) after substituting (3.1.224)-(3.1.225), using (3.1.242)–(3.1.244) and evaluating at the reference height, one has

$$\xi = \frac{g\kappa z_a}{T_v U_{10}^2} \frac{C_H(T_a - T_s) + 0.61 T_k C_E(q_a - q_s)}{(C_D^s)^{3/2}} \tag{3.1.254}$$

In summary, the coefficients C_D^s, C_H and C_E are obtained by solving the system of equations (3.1.249), (3.1.252), (3.1.253) using an iteration scheme. Input parameters are the wind speed $|U_{10}|$, the air temperature T_a, the sea surface temperature T_s and the relative humidity RH (needed to evaluate q_a, q_s through (3.1.176)–(3.1.177)).

The effects of stratification on the values of the transfer coefficients can be observed in Figures 3.1.4a and b where the ratios C_D^s/C_{DN} and C_E/C_{EN} are plotted as a function of wind speed using $RH = 80\%$, $T_s = 12^0$C and different values for the air-sea temperature difference. The coefficients only differ by a few percent from their neutral values if $|U_{10}| >$ 10 m/s which is smaller than the error margin for the neutral coefficients. On the other hand, for low wind speeds ($|U_{10}| <$ 10 m/s) thermal stratification leads to a substantial reduction or increase of the ratios according as the stratification is stable ($\Delta T > 0$) or unstable ($\Delta T < 0$).

To implement this theory into the program some simplifications have been made. First of all, it is noted that $C_{HN} = C_{EN}$ from (3.1.217) and thus $C_H = C_E$ from (3.1.252)–(3.1.253) so that only two equations need to be solved. Tests showed that the relative humidity used in the evaluation of q_a and q_s is only of secondary importance and is set to a reference value of 80%. It appeared also that the air-sea temperature difference ΔT has a much larger effect than the actual values of the air and sea temperature. The coefficients

C_D^s and C_E are then written as function of ΔT, $|U_{10}|$ and the reference temperature T_0.[5] To avoid computational overhead the values of C_D^s and C_E are stored initially in two arrays for a fixed number of $|U_{10}|$ and ΔT values. During the execution of the program the actual values of C_D^s and C_E are calculated at each grid point and meteorological time step from these pre-determined values using a bilinear interpolation which consumes much less computing time.

1.6.5 Solar radiation

Deriving a suitable expression for the solar radiation flux is not straightforward in view of its dependence on atmospheric parameters (atmospherical absorption and reflection, cloud coverage, albedo of the sea surface) whose influence is difficult to parameterise. The approach, described here, partially follows Rosati and Miyakoda (1988). The radiation entering at the top of the atmosphere is given by

$$Q_s = Q_0 p_{cor} H(\sin\gamma_\odot) \tag{3.1.255}$$

where $Q_0 = 1367.0$ W/m^2 is the solar constant, $\gamma_\odot$ the altitude of the sun and H the Heaviside function ($H(x) = 0$ for $x < 0$ and $= x$ otherwise). The factor p_{cor} represents a correction term due to the elliptical orbit of the earth and is usually expressed as a function of the day number of the year J:

$$p_{cor} = 1 + 0.03344\cos(J' - 2.8^0) \tag{3.1.256}$$

$$J' = 0.9856J \tag{3.1.257}$$

The altitude of the sun is calculated from

$$\sin\gamma_\odot = \sin\phi\sin\delta_\odot + \cos\phi\cos\delta_\odot\cos H_\odot \tag{3.1.258}$$

where $\delta_\odot$ is the declination of the sun, $H_\odot$ the sun's hour angle and ϕ the latitude. The angle $\delta_\odot$, measured in degrees, is obtained from the series expansion

$$\delta_\odot = \delta_0 + \sum_{n=1}^{3}(a_n\cos nJ' + b_n\sin nJ') \tag{3.1.259}$$

with

$$\begin{aligned} \delta_0 &= 0.33281 \\ (a_1, a_2, a_3) &= (-22.984, -0.34990, -0.13980) \\ (b_1, b_2, b_3) &= (3.7872, 0.03205, 0.07187) \end{aligned} \tag{3.1.260}$$

[5]By default, the temperature field is initialised by the value of T_0 (see equation (3.1.285)). The initial temperature is mostly not the same as the value of T_0 in the current theory which should be taken as typical for the whole simulation. It is therefore recommended to use a different value for the initial temperature if ITDIF is set to 1.

The sun's hour angle, measured in hours, is computed by

$$H_\odot = t_h - 12 + TE + \lambda_h \tag{3.1.261}$$

where t_h is the hour of the day, λ_h the longitude (expressed in hours) and TE the equation of time which can be written as

$$TE = \sum_{n=1}^{3} (c_n \cos nJ' + d_n \sin nJ') \tag{3.1.262}$$

with

$$\begin{aligned} (c_1, c_2, c_3) &= (0.0072, -0.0528, -0.0012) \\ (d_1, d_2, d_3) &= (-0.1229, -0.1565, -0.0041) \end{aligned} \tag{3.1.263}$$

Note that t_h must be given in GMT.

Taking account of the effect of absorption by the atmosphere, the direct solar radiation incident on the ocean surface, is given by

$$Q_{dir} = Q_s e^{-\tau} \tag{3.1.264}$$

The following form, proposed by Dogniaux (1984, 1985), is considered for the extinction factor

$$\tau = m_o \delta_R t_L \tag{3.1.265}$$

The optical air mass m_o, Rayleigh's optical thickness δ_R and Linke's factor t_L are expressed as function of the solar altitude $\gamma_\odot$ in degrees, according to

$$\delta_R = (0.9 m_o + 9.4)^{-1} \tag{3.1.266}$$

$$t_L = 0.021\gamma_\odot + 3.55 \tag{3.1.267}$$

$$m_o = [\sin\gamma_\odot + 0.15(\gamma_\odot + 3.885)^{-1.253}]^{-1} \tag{3.1.268}$$

The formulation, given by (3.1.266)–(3.1.268), has the advantage that it does not diverge at low solar altitudes.

The direct component of solar radiation must be supplemented by the diffuse sky radiation Q_{dif}. Following Rosati and Miyakoda (1988) it is assumed that one half of the scattered radiation reaches the ocean surface so that

$$Q_{dif} = \Big((1 - A_\alpha)Q_s - Q_{dir}\Big)/2 \tag{3.1.269}$$

They considered the value of 0.09 for the water vapor and ozone absorption coefficient A_α. The total radiation flux at the ocean surface under clear sky conditions is then given by

$$Q_{cs} = Q_{dir} + Q_{dif} = \frac{1}{2} Q_s (e^{-\tau} + 1 - A_\alpha) \tag{3.1.270}$$

The clear sky value (3.1.270) must be corrected for the cloud coverage and for the reflection by the ocean surface. The empirical formula, derived by Reed (1977), appears to have a better agreement with observational data compared to other formulations (Katsaros, 1990). The short-wave radiation flux at the sea surface then finally takes the form

$$Q_{rad} = Q_{cs}(1 - 0.62 f_c + 0.0019\gamma_{\odot,max})(1 - A_s) \tag{3.1.271}$$

where $\gamma_{\odot,max}$ is the solar altitude at noon. A constant value of 0.06 is assumed for the sea surface albedo A_s. Variations of the albedo as function of the solar altitude and atmospheric transmittance have been tested using the empirical fits derived by Payne (1972). No appreciable difference was found with the formulation (3.1.271). The only parameter, which needs to be supplied externally, is the fractional cloud cover f_c also used in the expression (3.1.178) for the long-wave radiation flux. Values of f_c are often difficult to obtain or are only provided in the form of climatological means, whereas direct measurements of Q_{rad} are often available. The program has therefore an option, depending on the value of the switch IOPTM, to determine Q_{rad} and the non-solar heat flux Q_{nsol} directly through an input file. This is further discussed in Section IV-2.2.1.

1.6.6 Wave-current interaction at the sea bed

As shown by the numerical simulations of Davies and Lawrence (1994, 1995) interaction of currents and surface waves has an important impact on the circulation in coastal waters due to the enhancement of the bottom stress. This effect can be included in the program with the switch IOPTW. The method is based upon the formulation proposed by Signell et al. (1990) which is an adapted form of the theory originally developed by Grant and Madsen (1979). Only a general outline of the procedures will be given here. Details can be found in the latter papers.

When surface waves are included in the program, their effect is determined by an increase of the bottom friction coefficient C_D^b. The wave field is supplied externally by a significant wave height h_s and a wave period T_w. The wave amplitude and frequency are defined by

$$a_w = h_s/2\ ,\ \omega_w = 2\pi/T_w \tag{3.1.272}$$

The near-bottom wave orbital velocity determined from linear wave theory is given by

$$U_w = \frac{a_w\omega_w}{\sinh k_w H} \tag{3.1.273}$$

with the wavenumber k_w obtained from the dispersion relation

$$\omega_w^2 = g k_w \tanh k_w H \tag{3.1.274}$$

A maximum bottom shear stress is then defined as the sum of the shear stress associated with the steady current and the maximum bottom wave stress, or

$$\tau_{b,max} = \tau_c + \tau_{w,max} = \rho_0 u_{*cw}^2 \tag{3.1.275}$$

where u_{*cw} is the shear velocity due to the combined interaction of currents and waves. Following Signell et al. (1990), the assumption is made that the wave and the current act in the same direction. The wave component and the associated shear velocity U_{*w} are determined from

$$\tau_{w,max} = \rho_0 U_{*w}^2 = \frac{1}{2}\rho_0 f_w U_w^2 \tag{3.1.276}$$

The wave parameters are not calculated by the program but supplied externally through an input file. It is therefore reasonable to assume that, in the model, the current has no influence on the wave. The wave friction factor f_w is evaluated with the aid of the empirical relations which according to Signell et al. (1990) have the best agreement with the data

$$\begin{array}{lll} f_w = 0.13(k_b/A_b)^{0.4} & \text{if} & k_b/A_b < 0.08 \\ f_w = 0.23(k_b/A_b)^{0.62} & \text{if} & 0.08 \leq k_b/A_b < 1 \\ f_w = 0.23 & \text{if} & k_b/A_b \geq 1 \end{array} \tag{3.1.277}$$

where $A_b = U_w/\omega_w$ is the near-bottom excursion amplitude. Following Grant and Madsen (1979) the physical bottom roughness k_b can be related to the roughness length z_0 by $k_b = 30z_0$ valid for rough turbulent flow.

The friction velocity u_{*cw} arising form the combined shear stress (3.1.275) is given by

$$u_{*cw} = (u_{*b}^2 + U_{*w}^2)^{1/2} \tag{3.1.278}$$

with

$$u_{*b}^2 = C_D^b(u_b^2 + v_b^2) = C_D^b u_c^2 \tag{3.1.279}$$

from (3.1.112) and (3.1.185).

An apparent bottom roughness is next defined using a logarithmic profile for the steady current. This takes the following form above the wave boundary layer

$$u_c = \frac{u_{*b}}{\kappa}\ln\frac{30\sigma H}{k_{bc}} \tag{3.1.280}$$

Evaluating (3.1.280) at the reference height z_r defined as the distance to the sea bed of the centre of the bottom grid cell. Combining (3.1.278) and (3.1.280) one has

$$C_D^b = \Big(\kappa/\ln(30z_r/k_{bc})\Big)^2 \tag{3.1.281}$$

From the theory explained in Grant and Madsen (1979) the apparent and physical bottom roughness are related by

$$k_{bc} = k_b\Big(24\frac{u_{*cw}}{U_w}\frac{A_b}{k_b}\Big)^\beta \tag{3.1.282}$$

where

$$\beta = 1 - \frac{u_{*b}}{u_{*cw}} \tag{3.1.283}$$

Equations (3.1.281) and (3.1.282) must in principle be solved with an iterative procedure since u_{*cw} depends on C_D^b through (3.1.278) and (3.1.279). Following Davies and Lawrence

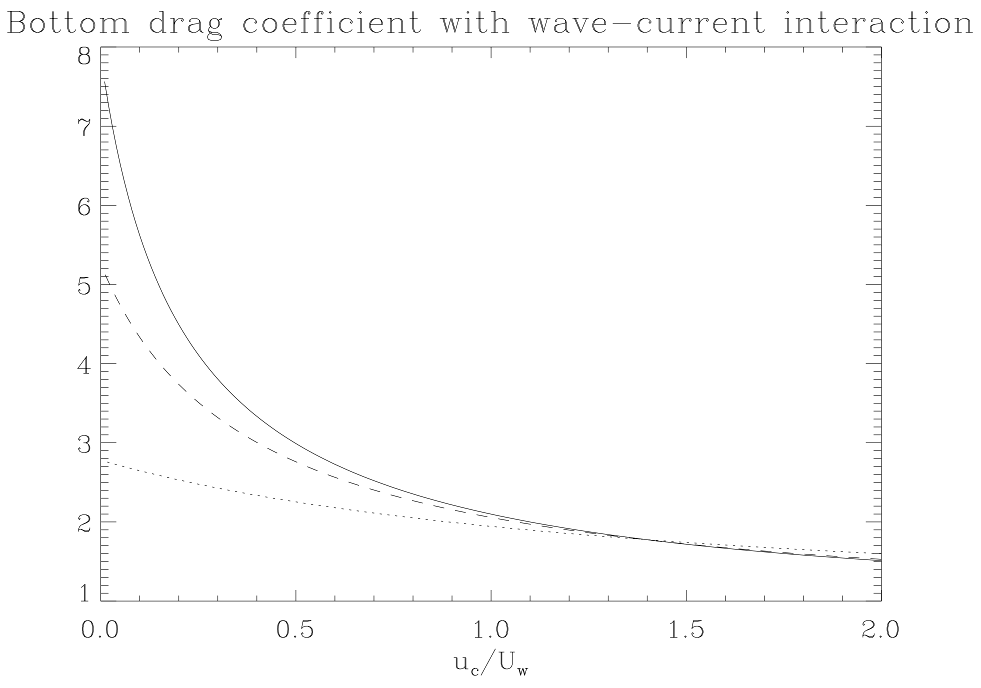


Figure 3.1.5: Values of the bottom drag coefficient, divided by its value in the absence of wave effects, as function of the ratio u_c/U_w for $h_s = 2.0$ m, $T_w = 10$ s (solid), $h_s = 2.0$ m, $T_w = 5$ s (dots), $h_s = 1$ m, $T_w = 10$ s (dashes).

(1994) a simpler numerical approach is considered whereby k_{bc} is set equal to k_b at the first time step, C_D^b calculated using (3.1.281) and a new value of k_{bc} is determined from (3.1.282) which is used in (3.1.281) at the next time step.

As an example the values of the bottom drag coefficient, measured with respect to its value in the absence of wave effects, are displayed in Figure 3.1.5 as a function of the ratio u_c/U_w, using a water depth of 20 m, a reference height of 1 m, a roughness length of 10^{-3} m and different values for h_s and T_w. A significant increase of the bottom friction coefficient is observed for small values of the bottom current whereas wave effects are negligible in the limit $u_c/U_w \to \infty$. The importance of wave-current interaction in shallow waters is further confirmed by the 3-D simulations performed by Davies and Lawrence (1994, 1995).

1.6.7 Initial conditions

All currents are set to zero initially, i.e.

$$u = 0 \ , \ \overline{U} = 0 \quad \text{and} \quad v = 0 \ , \ \overline{V} = 0 \tag{3.1.284}$$

Uniform values are taken for temperature and salinity, given by their reference values

$$T = T_0 \ , \ S = S_0 \tag{3.1.285}$$

A linear profile is considered for turbulence energy, satisfying the bottom and surface boundary conditions (3.1.190) and (3.1.180):

$$k = \Big((u_{*s}^2 - u_{*b}^2)\sigma + u_{*b}^2\Big)/\varepsilon_0^{2/3} \tag{3.1.286}$$

Initial values for the mixing length are obtained using one of the mixing length formulations selected by ILENG[6]. Once k and l are determined, the dissipation rate ε is determined from (3.1.116).

The above initial conditions are set by default but can be changed by the user. This is further discussed in Section IV-2.6.

1.6.8 Program switches

The different formulations and processes described in this section (surface and bottom boundary conditions, surface fluxes, wave-current interaction) are implemented in the program with the aid of a number of switches. They are listed in Table 3.1.5 below. A complete description of all program switches is discussed in Section IV-2.2.1. The type and form of the open boundary conditions are determined using a series of arrays defined by the user. This is further explained in Sections IV-2.4 and IV-2.5.

1.7 Harmonic analysis

1.7.1 Residuals, amplitudes and phases

The program offers the possibility to apply an harmonic analysis on a given number of user-defined quantities. This is selected with the switch IOUTT. The method is closely related to the theory described in Godin (1972). An harmonic expansion approximates a function $F(x_1, x_2, \tilde{x}_3, t)$ by a series of the form

$$F(x_1, x_2, \tilde{x}_3, t) = a_0 + \sum_{n=1}^{N_h} a_n \cos\omega_n(t - t_c) + \sum_{n=1}^{N_h} b_n \sin\omega_n(t - t_c) \tag{3.1.287}$$

where a_0, a_n and b_n are spatially dependent parameters obtained with an optimisation procedure, ω_n are a series of user-defined frequencies, N_h the number of harmonics in the analysis, t_c a "central" time and t the time since the start of the simulation. The procedure is the following. Firstly, the time t_c and a period T for the analysis, given by an even number of time steps[7], i.e. $T = 2M\Delta t$, are defined. Secondly, the program evaluates the values of F at times $t_c + k\Delta t$ with $-M \leq k \leq M$, denoted by

$$F_k = F(x_1, x_2, \tilde{x}_3, t_c + k\Delta t) \tag{3.1.288}$$

[6]If NTRANS is set to 2, a second turbulent transport equation is solved and a value for ILENG is only required to initialise l at the initial time.

[7]Note that Δt equals the time step Δt_{3D} for the 3-D mode calculations.

Table 3.1.5: Program switches related to the boundary conditions.

IOPTM	Determines the type of meteorological forcing. 0 : Surface stress, heat fluxes, salinity fluxes and solar radiation are zero. 1 : Surface stress is constant in space and time. Heat fluxes, salinity fluxes and solar radiation are zero. 2 : Surface stress, heat fluxes, salinity fluxes and solar radiation are uniform in space but non-uniform in time. They are calculated as a function of the meteorological forcing data using the formulations provided in the program. For the salinity flux either $E_{vap} - R_{pr}$ (IOPTSA $= 1$) or R_{pr} (IOPTSA $= 2$) is supplied by the user. 3 : Surface stress, heat fluxes, salinity fluxes and solar radiation are non-uniform in space and time. They are calculated as a function of the meteorological forcing data using the formulations provided in the program. For the salinity flux either $E_{vap} - R_{pr}$ (IOPTSA $= 1$) or R_{pr} (IOPTSA $= 2$) is supplied by the user. 4 : Surface stress, heat fluxes, salinity fluxes and solar radiation are uniform in space but non-uniform in time. Surface stress is evaluated using equation (3.1.168). Non-solar and solar heat fluxes and the evaporation minus precipitation rate are supplied by the user. 5 : Surface stress, heat fluxes, salinity fluxes and solar radiation are non-uniform in space and time. Surface stress is evaluated using equation (3.1.168). Non-solar and solar heat fluxes and the evaporation minus precipitation rate are supplied by the user. Default value is 0.
IOPTHE	Selects the type of solution for the temperature equation. 0 : The temperature equation is not solved. The temperature field is initialised at the start of the program but not updated in time. 1 : The temperature equation is solved assuming that all solar radiation is absorbed at the surface. 2 : The temperature equation is solved with the solar radiation absorbed throughout an upper layer using the theory described in Section III-1.4. Default value is 0.
IOPTSA	Selects the type of solution for the salinity equation. 0 : The salinity equation is not solved. The salinity field is initialised at the start of the program but not updated in time. 1 : The salinity equation is solved. The surface salinity flux is zero if IOPTM $= 0, 1$ or determined using (3.1.179) with $E_{vap} - R_{pr}$ supplied by the user if IOPTM > 1. 2 : The salinity equation is solved. The surface salinity flux is zero if IOPTM $= 0, 1$ or determined using (3.1.179) with R_{pr} supplied by the user if IOPTM $= 2, 3$ or with $E_{vap} - R_{pr}$ supplied by the user if IOPTM $= 4, 5$. Default value is 0.

Table 3.1.5: continued.

IBSTR	Selects the bottom stress formulation. 0 : The bottom stress is set to zero. 1 : The bottom stress is evaluated using the quadratic friction law (3.1.185). 2 : The bottom stress is evaluated using the linear friction law (3.1.186). Default value is 1.
IDRAG	Selects the formulation for the surface drag coefficient. 0 : constant value given by (3.1.218) 1 : Large and Pond (1981) formulation (3.1.219) 2 : Smith and Banke (1975) formulation (3.1.220) 3 : Geernaert et al. (1986) formulation (3.1.221) 4 : Charnock (1955) formulation (3.1.222) Default value is 0.
ITDIF	Selects the formulation for the surface exchange coefficients C_E and C_H. 0 : uniform values given by (3.1.217) 1 : as function of the air-sea temperature difference and wind speed using the theory described in Section III-1.6.4. Note that this option also affects the value of the surface drag coefficient. Default value is 0.
IOPTW	Selects the type of wave input for wave-current interaction. 0 : Wave-current interaction is disabled. No wave data need to be supplied. 1 : Wave-current interaction is enabled. Wave height and period are uniform in space and time. 2 : Wave-current interaction is enabled. Wave height and period are non-uniform in time but uniform in space. 3 : Wave-current interaction is enabled. Wave height and period are non-uniform in space and time. Default value is 0.

The harmonic parameters a_0, a_n and b_n are then determined by minimising the quantity

$$R = \sum_{k=-M}^{M} [F_k - a_0 - \sum_{n=1}^{N_h} a_n \cos(\omega_n k\Delta t) - \sum_{n=1}^{N_h} b_n \sin(\omega_n k\Delta t)]^2 \tag{3.1.289}$$

Setting the first derivatives of R with respect to a_0, a_m and b_m ($1 \leq m \leq N_h$) equal to zero yields the following set of $2N_h + 1$ equations

$$\sum_{k=-M}^{M} [F_k - a_0 - \sum_{n=1}^{N_h} a_n \cos(\omega_n k\Delta t) - \sum_{n=1}^{N_h} b_n \sin(\omega_n k\Delta t)] = 0 \tag{3.1.290}$$

$$\sum_{k=-M}^{M} [F_k - a_0 - \sum_{n=1}^{N_h} a_n \cos(\omega_n k\Delta t) - \sum_{n=1}^{N_h} b_n \sin(\omega_n k\Delta t)] \cos(\omega_m k\Delta t) = 0 \tag{3.1.291}$$

$$\sum_{k=-M}^{M} [F_k - a_0 - \sum_{n=1}^{N_h} a_n \cos(\omega_n k\Delta t) - \sum_{n=1}^{N_h} b_n \sin(\omega_n k\Delta t)] \sin(\omega_m k\Delta t) = 0 \tag{3.1.292}$$

with $1 \leq m \leq N_h$. The system can be simplified with the aid of known summation rules for trigonometric functions (Gradshteyn and Ryzhik, 1981). The final result can be written as

$$\sum_{n=1}^{N_h} X_{mn} a_n = R_m \tag{3.1.293}$$

$$\sum_{n=1}^{N_h} Y_{mn} b_n = S_m \tag{3.1.294}$$

$$a_0 = \frac{1}{2M+1}[\sum_{k=-M}^{M} F_k - \sum_{n=1}^{N_h} a_n \sin(\frac{2M+1}{2}\omega_n \Delta t) \csc(\frac{\omega_n \Delta t}{2})] \tag{3.1.295}$$

where

$$\begin{aligned} X_{mn} = & \frac{1}{2}\sin\Big(\frac{2M+1}{2}(\omega_m + \omega_n)\Delta t\Big) \csc\Big(\frac{\Delta t}{2}(\omega_m + \omega_n)\Big) \\ & + \frac{1}{2}\sin\Big(\frac{2M+1}{2}(\omega_m - \omega_n)\Delta t\Big) \csc\Big(\frac{\Delta t}{2}(\omega_m - \omega_n)\Big) \\ & - \frac{1}{2M+1}\sin(\frac{2M+1}{2}\omega_m \Delta t) \csc(\frac{\omega_m \Delta t}{2}) \sin(\frac{2M+1}{2}\omega_n \Delta t) \csc(\frac{\omega_n \Delta t}{2}) \end{aligned} \tag{3.1.296}$$

$$\begin{aligned} Y_{mn} = & \frac{1}{2}\sin\Big(\frac{2M+1}{2}(\omega_m - \omega_n)\Delta t\Big) \csc\Big(\frac{\Delta t}{2}(\omega_m - \omega_n)\Big) \\ & - \frac{1}{2}\sin\Big(\frac{2M+1}{2}(\omega_m + \omega_n)\Delta t\Big) \csc\Big(\frac{\Delta t}{2}(\omega_m + \omega_n)\Big) \end{aligned} \tag{3.1.297}$$

$$R_m = \sum_{k=-M}^{M} F_k \cos(\omega_m k\Delta t) - \frac{1}{2M+1}\sin(\frac{2M+1}{2}\omega_m \Delta t) \csc(\frac{\omega_m \Delta t}{2}) \sum_{k=-M}^{M} F_k \tag{3.1.298}$$

$$S_m = \sum_{k=-M}^{M} F_k \sin(\omega_m k\Delta t) \tag{3.1.299}$$

The following replacements need to be made at singular points of the csc functions:

$$\sin\Big(\frac{2M+1}{2}\omega_*\Delta t\Big)\csc\Big(\frac{\omega_*\Delta t}{2}\Big) \rightarrow 2M+1 \quad \text{if} \quad \text{mod}(\omega_*\Delta t, 2\pi) = 0 \tag{3.1.300}$$

where ω_* equals either ω_m , ω_n , $\omega_m + \omega_n$, $\omega_m - \omega_n$. The matrices X and Y only depend on the values of the frequencies, the time step and the analysed period and can thus be evaluated at the start of the program. The numerical solution of the linear system (3.1.293) and (3.1.294) is facilitated by first writing the matrices X and Y as the product of a lower and an upper triangular matrix, known as LU-decomposition, which only needs to be performed once. This reduces the number of arithmetic operations and computing time since the systems are to be solved at a number of selected grid points and for a given number of user-defined quantities. Details of the numerical procedure are described in Press et al. (1989).

Once the parameters a_0, a_n and b_n are determined, the harmonic expansion (3.1.287) is written into the final form

$$F(x_1, x_2, \tilde{x}_3, t) = A_0 + \sum_{n=1}^{N_h} A_n \cos(\omega_n(t - t_c) - \varphi_n) \tag{3.1.301}$$

where the residual A_0, the amplitudes A_n and the phases φ_n (with respect to the central time t_c) are obtained from

$$A_0 = a_0 \ , \ A_n = (a_n^2 + b_n^2)^{1/2} \ , \ \varphi_n = \text{mod}(\arctan(b_n/a_n), 2\pi) \tag{3.1.302}$$

Important to note is that the number and values of the frequencies ω_n used in the harmonic analysis do not need to be the same as the ones appearing in the tidal forcing at the open boundaries (see equation (3.1.198)). For example, if the model is forced with the M_2-tide only, higher order harmonics (M_4, M_6, ...) are generated by the non-linearities of the model equations. These higher order terms can be investigated by applying an harmonic analysis. Alternatively, the method can be used to analyse the evolution of a M_2-tide during a spring-neaps cycle, e.g. by forcing the model with a M_2- and S_2-tide and performing the analysis with only the M_2-frequency at different times $t = t_c, t_c + T$, ... (e.g. Luyten, 1997).

The period T needs to be selected with some care. A general rule is that it must be of the same order or larger than any of the analysed periods $2\pi/\omega_n$ and must increase with the number of frequencies.

1.7.2 Tidal ellipses

During the course of a tidal cycle the current vector describes a curve, known as the "tidal ellipse". Characteristic parameters of tidal ellipses are the semi-major axis, semi-minor axis, ellipticity, orientation and elliptic angle. They can optionally be derived by the program in the usual way (e.g. Godin, 1972) once the harmonic parameters of the current are available using the analysis of the preceding section. The "nth" harmonic component of the horizontal current can be written as

$$u_n = u_{an}\cos(\omega_n t - \varphi_{nu}) \ , \ v_n = v_{an}\cos(\omega_n t - \varphi_{nv}) \tag{3.1.303}$$

Introducing the complex notation

$$U = u_a e^{-i\varphi_u} \ , \ V = v_a e^{-i\varphi_v} \tag{3.1.304}$$

where the subscript n has been omitted for simplicity, the complex current can then be decomposed into a cyclonically and an anticyclonically rotating component

$$\begin{aligned} u + iv &= \frac{1}{2}(Ue^{i\omega t} + \tilde{U}e^{-i\omega t}) + \frac{i}{2}(Ve^{i\omega t} + \tilde{V}e^{-i\omega t}) \\ &= S_+ e^{i\omega t} + \tilde{S}_- e^{-i\omega t} \end{aligned} \tag{3.1.305}$$

where a ˜ denotes the complex conjugate and

$$S_\pm = \frac{1}{2}(U \pm iV) = |S_\pm| e^{i\alpha_\pm} \tag{3.1.306}$$

The semi-major axis A, semi-minor axis B and ellipticity ε of the ellipse, described by the current, are then given by

$$A = |S_+| + |S_-| \ , \ B = ||S_+| - |S_-|| \ , \ \varepsilon = (|S_+| - |S_-|)/(|S_+| + |S_-|) \tag{3.1.307}$$

The inclination Θ of the ellipse with respect to the x_1-axis and the elliptic angle Φ, i.e. the angle between the initial current at $t = 0$ and its position when the current achieves its first maximum, are obtained using

$$\Theta = (\alpha_+ - \alpha_-)/2 \ , \ \Phi = -(\alpha_+ + \alpha_-)/2 \tag{3.1.308}$$

with the restriction that

$$0 \le \Theta, \Phi \le \pi \tag{3.1.309}$$

The orientation of the ellipse is determined by the sign of the ellipticity. A positive value of ε means that the current vector rotates cyclonically (anticlockwise in the northern hemisphere) and $|S_+| > |S_-|$ whereas a negative value indicates anticyclonic rotation (clockwise in the northern hemisphere) and $|S_+| < |S_-|$. If $\varepsilon = 0$, the flow is rectilinear and $|S_+| = |S_-|$. A useful discussion of cyclonic and anticyclonic components and their impact on the depth of the tidal bottom layer can be found in e.g., Prandle (1982), Soulsby (1983), Luyten (1996).

Chapter 2

Biological and Sediment Models

The biological model cycles concentrations of organic carbon and nitrogen through microplankton and detrital compartments with associated changes in dissolved concentrations of nitrate, ammonium and oxygen. The concentrations are updated in time by solving a transport equation for each state variable whereby the biological interactions are included as source and sink terms and which takes account of vertical sinking and the physical transport by advection and diffusion. As an exception, chlorophyll is derived algebraically from microplankton carbon and nitrogen concentrations. The sediment model determines the time evolution and transport of inorganic particulate material. Exchanges between the water column and the seabed are modelled through a "fluff" layer in the microplankton and detritus compartments and in the sediment model.

2.1 Biology overview

The physical state of the sea can be accurately represented as fields of a few variables (current, temperature, salinity, ...). The marine biological system is in principle much more elaborate, for its living part consists of many discrete individuals belonging to a number of species. In addition are nonliving populations of particles of several sizes and a variety of dissolved substances. A common classification of the water column ecosystem distinguishes three particulate components:

phytoplankton : microscopic floating plants ("algae") and certain bacteria, able to use light in photosynthesis and thus to convert dissolved inorganic substances (including the "nutrients" nitrate and phosphate) to organic material according to the following (averaged) reaction: $106CO_2 + 122H_2O + 16NO_3^- + PO_{4^{3-}} + 19H^+$ (+ >1104 photons) $\longrightarrow (CH_2O)_{106}(NH_3)_{16}H_3PO_4 + 138O_2$;

zooplankton : small drifting animals that eat phytoplankton and each other, providing food for fish and breaking-down ("remineralising") part of the photosynthetically-created organic material into its inorganic components;

detritus : inorganic and non-living organic particles, in part formed by the death of plankton, and tending to sink.

Such a classification, however, hides the importance of the “Microbial Loop”. This involves bacteria, nourished by dissolved organic matter leaked by phytoplankton or during feeding by small zooplankton, and single-celled animals (protozoans) which feed on the bacteria as well as on small phytoplankton, and are themselves eaten by the zooplankton described above (Azam et al. 1983; Williams, 1981).

Because of the complexity of the biological system, there is no agreed biological analogue of the physical equations for salinity, temperature and momentum. A variety of models have been proposed for the marine pelagic ecosystem, starting with Riley (1946) and including recent models of Fasham et al. (1990) and ERSEM (Varela et al., 1995; Baretta-Bekker et al., 1995). Some of these models are reviewed by Fransz et al. (1991). The biological submodel used in COHERENS was proposed by Tett (1990a) (see also Tett and Grenz, 1994; Tett and Walne, 1995). Its distinguishing features are (1) the use of a *microplankton* compartment to include the biomasses and activity of microbial loop organisms as well as those of phytoplankton, and (2) the use of variations in the chemical composition (especially, the nitrogen:carbon ratio) of microplankton and detrital components to control many of the biological rate processes. Further details follow. A comprehensive account of the microplankton part of the model is given by Tett (1998), included on the COHERENS CD-ROM.

The term “nutrient”, in the present context, could refer to dissolved inorganic compounds of nitrogen, phosphorus, silicon, iron, and perhaps other elements, required by phytoplankton. The elements nitrogen and phosphorus are essential parts of organic matter; silicon is an important part of the cell wall of some phytoplankton; iron is used by some enzymes. The “limiting” nutrient is the compound or element which at any given moment is most scarce relative to the requirement of the phytoplankton. Most often, in the sea, this element is nitrogen: the growth rate and biomass of phytoplankton is often increased as a result of adding nitrogen compounds to sea (for example, as part of eutrophication). The commonest such compounds disassociate in seawater to give the ions nitrate (NO_3^-), nitrite (NO_2^-) or ammonium (NH_4^+). The present version of the biological model in COHERENS is concerned only with nitrogen as a potentially limiting nutrient, and simulates changes in the concentration of ammonium (resulting directly from the metabolic activity of zooplankton) and nitrate (deemed to include nitrite) formed by oxidation from ammonium.

Of course, phytoplankton also need light for growth, and under some conditions (e.g. during winter) light rather than nitrogen supply limits microplankton growth. The biological model deals with both cases. It also deals with the oxygen changes resulting from photosynthesis or the decomposition of organic matter. Finally, it accounts for what happens to microplankton and detritus as these sink to the seabed, and accumulate there in a loosely-packed layer of “fluff”, like freshly-fallen snow. It does not explicitly represent the benthos, the animals and micro-organisms living on and in the sea-bed. Nor does it describe the dynamics of the larger planktonic animals (mesozooplankton and others): it is, essentially, a model of water column microbiology.

2.2 Fundamental principles

The following general processes are implemented in the biological part of COHERENS:

- internal non-conservative biological or chemical processes and photosynthesis by the absorption of PAR
- physical transport by advection and diffusion
- vertical sinking
- deposition and erosion via a "fluff" layer
- exchanges between the water column and the sediment layer.

The general features (except physical transport) are illustrated schematically in Figure 3.2.1. The processes, described in the sediment part of the model, are:

- physical transport by advection and diffusion
- vertical sinking
- deposition and erosion via a "fluff" layer

The biological, chemical and detrital processes are modelled through their effects on a number of state variables. A separate equation has to be solved numerically for each state variable. It is desirable to minimise the list of such variables, not only in order to minimise computer storage and calculation (especially in 3-D simulations, or during the repeated simulations necessary for parameter optimisation), but also because of Occam's Razor ("do not unnecessarily multiply explanations") and on account of practical and theoretical difficulties in estimating parameters (which, in general, increase with variable number). Taking into account this need for parsimony in state variables, Tett (1990a) proposed a microbiological model with three pelagic compartments and 7 independent state variables:

- microplankton organic carbon and nitrogen (and non-independent chlorophyll);
- detrital organic carbon and nitrogen;
- dissolved ammonium, nitrate and oxygen.

The biological model of COHERENS is based on Tett (1990a) and contains eight state variables: microplankton carbon and nitrogen, detrital carbon and nitrogen, concentrations of dissolved nitrate, ammonium and oxygen, and accumulated zooplankton nitrogen. Chlorophyll is derived from microplankton carbon and nitrogen and is therefore not included as a state variable. The sediment model includes one state variable for inorganic

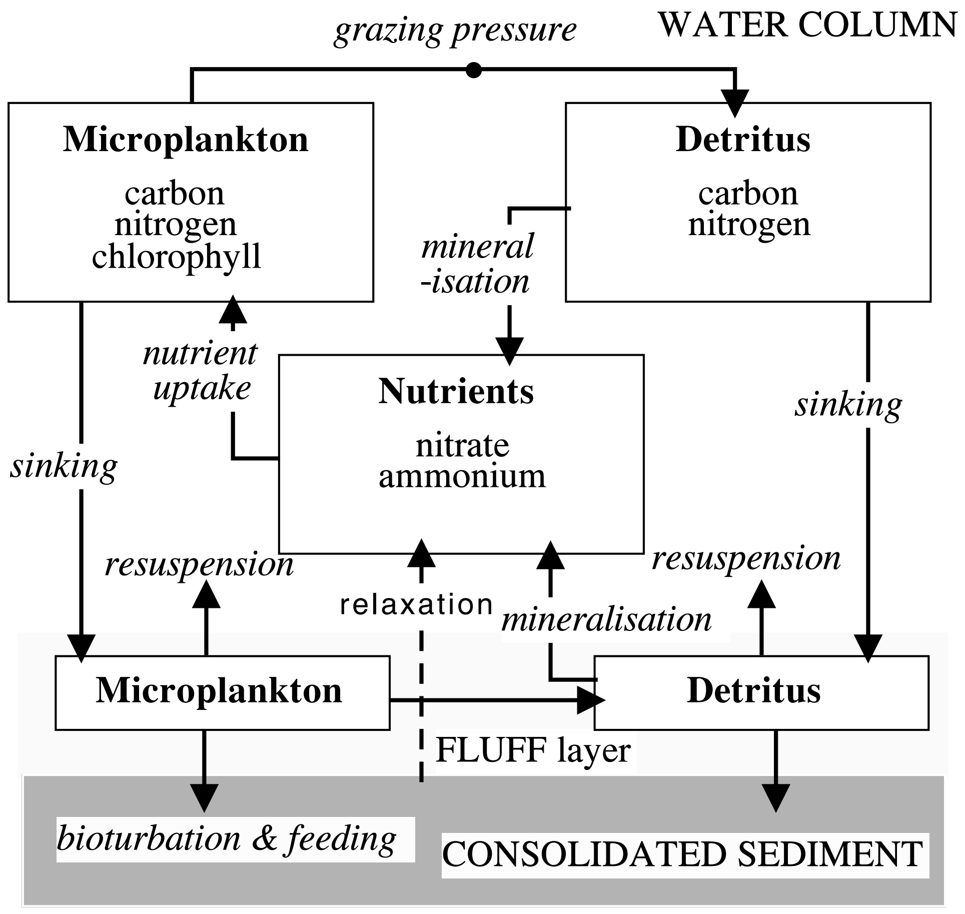


Figure 3.2.1: General features of the microplankton-detritus model.

Table 3.2.1: State variables and internal non-conservative processes in the biological and sediment models.

variable	symbol	unit	source/sinks
microplankton carbon	B	mmol C m^{-3}	$\beta(B) = (\mu - G)B$
microplankton nitrogen	N	mmol N m^{-3}	$\beta(N) = ({}^{NO}u + {}^{NH}u)B - GN$
detrital carbon	C	mmol C m^{-3}	$\beta(C) = (1-\gamma)GB - {}^{C}rC$
detrital nitrogen	M	mmol N m^{-3}	$\beta(M) = (1-\gamma)GN - {}^{M}rM$
dissolved nitrate	${}^{NO}S$	mmol N m^{-3}	$\beta({}^{NO}S) = -{}^{NO}uB + {}^{NH}r.{}^{NH}S$
dissolved ammonium	${}^{NH}S$	mmol N m^{-3}	$\beta({}^{NH}S) = -{}^{NH}uB - {}^{NH}r.{}^{NH}S + e\gamma GN + {}^{M}rM$
oxygen	O	mmol O m^{-3}	$\beta(O) = ({}^{O}q^{B}\mu + {}^{O}q^{NO}.{}^{NO}u - {}^{O}q^{C}e\gamma G)B$
			$-{}^{O}q^{NH}.{}^{NH}r.{}^{NH}S - {}^{O}q^{C}.{}^{C}rC$
zooplankton nitrogen	Z_N	mmol N m^{-3}	$\beta(Z_N) = \gamma(1-e)GN$
inorganic sediment	A	g m^{-3}	$\beta(A) = 0$

particulate material. All modelled state variables are listed in Table 3.2.1 with their symbolic name and unit.

The model is forced by seasonal cycles of vertical water column mixing (determined by the physical model), light and zooplankton grazing pressure, and boundary conditions. Zooplankton are not modelled but mesozooplankton grazing pressure is imposed as a time varying series determined from observations. The ecosystem described by the model is therefore "open" at the second trophic level. The zooplankton nitrogen variable accumulates potential losses at this level.

Although the sediment part of the program can be run independently from the biology, the biology depends on the sediments in two ways: (1) the attenuation of light by inorganic suspended particles (see Section III-2.3.1), (2) the resuspension of organic material from the bottom which depends in the model on the amount of available inorganic sediment in the fluff layer (see Section III-2.8.3). It is for this reason that the biological and sediment models are presented in the same chapter.

The water column concentrations of the state variables, listed in Table 3.2.1, are determined by solving nine transport equations whose general form is given by (3.1.70). In the absence of an adequate theory, the vertical turbulent fluxes of the concentrations are parameterised in the same way as the physical fluxes (see equation (3.1.18)) using the assumption of downgradient diffusion

$$-\langle w'\psi' \rangle = \lambda_T^{\psi}\frac{\partial\psi}{\partial x_3} \tag{3.2.1}$$

where λ_T^{ψ} is set to the vertical diffusion coefficient λ_T used in the physical part of the program for temperature and salinity. Similarly, the horizontal diffusion coefficient λ_H^{ψ} is set equal to its physical analogue λ_H. The only terms which need to be determined in the biological or sediment modules, are the vertical sinking term and the source/sink terms on the right hand side of (3.1.70). The former depends on the value, chosen for w_s^ψ. The latter represent the processes, specific for the biology.

In the remainder of this chapter the transport equation for any state variable is written in the more convenient form

$$\mathcal{H}(\psi) = \beta(\psi) \tag{3.2.2}$$

where $\mathcal{H}(\psi)$ represents the time derivative, all advective and diffusive terms and the vertical sinking term

$$\mathcal{H}(\psi) = (\mathcal{I} + \mathcal{A}_h + \mathcal{A}_v + \mathcal{A}_s - \mathcal{D}_v - \mathcal{D}_h)(\psi) \tag{3.2.3}$$

and

$$\beta(\psi) = \mathcal{P}(\psi) - \mathcal{S}(\psi) \tag{3.2.4}$$

all source/sink terms[1]. Apart form the boundary conditions, the biological and sediment models only need to determine the values of w_s^ψ and $\beta(\psi)$.

The model conserves all substances cycling within the water column, except for an accountable portion of grazed microplankton which is considered assimilated into zooplankton biomass. At the water-column/sediment interface an accountable portion of organic particulate material is lost to the sediment and a flux of dissolved nutrients is returned to the water-column. A similar flux of oxygen is permitted across the air–sea interface.

To verify that the model code conserves material the conservation of nitrogen is checked. This is achieved by summing all forms of particulate and dissolved nitrogen and by taking into account cumulative losses to zooplankton assimilation and benthic fluxes. Such tests have been performed and are further described in Chapter II-5 (Biological Test Cases).

2.3 Biological optics

2.3.1 Attenuation

Using the notations of Section III-1.4 the photosynthetically active radiation (PAR) I_p satisfies an exponential law, similar to (3.1.157):

$$\frac{\partial I_p}{\partial x_3} = (k_d + k_3) I_p \tag{3.2.5}$$

with I_p equal to $(1 - R_1)Q_{rad}$ at the sea surface. The diffuse downward attenuation coefficient k_d is calculated by summing the effects of several components:

$$k_d = k_{20}^w - \epsilon^S S + \epsilon^A A + \epsilon^C C + \epsilon^X X \tag{3.2.6}$$

where the first two terms on the right hand side of (3.2.6) represent the absorption of pure seawater and the contributions due to dissolved "yellow-substance", as discussed in

[1]Note that the distinction between the source and sink terms is of no relevance in the remainder of this chapter. Since $\beta(\psi)$ is the sum of several contributions, the positive ones add to $\mathcal{P}(\psi)$ and the negative to $\mathcal{S}(\psi)$.

Table 3.2.2: Optical parameters and their typical ranges.

symbol	unit	range	purpose
R_1	—	0.5–0.58	infrared fraction of all-wavelength solar radiation
R_2	—	0.34–0.4	rapid attenuation of PAR in the near surface layer
Δ_{opt}	m	5–15	layer thickness for rapid PAR attenuation
k_1	m^{-1}	>0.6	diffuse attenuation coefficient for infrared solar radiation
k_{20}^w	m^{-1}	2.06	diffuse attenuation coefficient for PAR in the case of pure seawater
ϵ^S	m^{-1} PSU^{-1}	0.05714	slope factor in the linear relationship between the PAR diffuse attenuation coefficient and salinity
ϵ^A	m^2 g^{-1}	0.1	diffuse attenuation cross section of iSPM
ϵ^C	m^2 (mmol $C)^{-1}$	0.002	diffuse attenuation cross section of detrital carbon
ϵ^X	m^2 (mg Chl)$^{-1}$	0.01–0.04	diffuse attenuation cross section of microplankton chlorophyll
k_a	m^{-1}	0.06	background (offshore) diffuse attenuation coefficient at the reference salinity S_a
k_b	m^{-1}	2.0	background diffuse attenuation coefficient at zero salinity minus k_a
S_a	PSU	35.0	reference (offshore) salinity

Section III-1.4. The other ϵ-terms are the diffuse PAR attenuation cross-sections of each type of light-attenuator, with units of m^2 per unit of attenuance. The three components are inorganic suspended particulate matter (iSPM) concentration A (mg m^{-3}), detrital organic carbon concentration C (mmol C m^{-3}) and microplankton chlorophyll concentration X (mg Chl m^{-3}). Acomplete list of all optical parameters is given in Table 3.2.2.

The model uses (diffuse) attenuation cross-sections ϵ to take account of concentrations of optically active particles. However, diffuse attenuation includes scattering as well as absorption of light. The model simplifies the effect of scattering by assuming that it merely lengthens the path taken by a typical photon when crossing from top to bottom of a layer, and hence increases the photon's probability of absorption within the layer. The path-length increase is proportional to the inverse of the mean cosine m_d of the angular distribution of downwards-traveling photons, and hence the attenuation contribution of, for example, chlorophyll, is $\epsilon^X X = m_d^{-1}.a_*^X X$ m^{-1} where a_*^X is the absorption cross-section (for chlorophyll), and is a constant (only) for a given PAR spectrum and phytoplankton physiological state and taxonomic composition (Tett, 1990b). In order to treat the attenuation cross section as a constant, the model assumes constant m_d as well as constant a_*^X. In reality, both can change with changing optical conditions, for example as the PAR spectrum and the photon angular distribution change with depth or time of the day. Furthermore, the assumption used in the model that (for any component) $k_d = m_d^{-1} a$ is only good for low concentrations of scattering particles.

It should be remarked that, following Kirk (1983), the contribution for a given component to the downwards diffuse attenuation coefficient at a given depth is, more accurately,

$$k_d = a_d + b_{bd} - b_{bu} R \tag{3.2.7}$$

where a_d — the diffuse absorption coefficient for downwards PAR (for that optical component) — is indeed $a m_d^{-1}$, a being the (beam) absorption coefficient, but there is also the backscattering of diffuse downwards PAR (specified by b_{bd}), to some extent opposed by the downward scattering of photons reflected upwards ($b_{bu}R$, R being the reflectance ratio). Kirk (1983) shows that the a_d term in equation (3.2.7) is dominant for low and moderate values of the ratio of (beam) scattering to (beam) absorption ($b/a < 3$). Such conditions are typical of oceanic and other shelf waters, except during a coccolithophorid bloom. Scattering effects are more important, however, in waters with high loads of iSPM, or when some diatoms or coccolithophorids bloom. The model may thus underestimate PAR attenuation under these conditions.

2.3.2 Light absorbed by photosynthesis

Photosynthesis depends on PAR absorbed by phytoplankton. This is $a_*^X X I_s$, where I_s is scalar irradiance, the photon flux from all directions. We assume that the ratio I_p/I_s is approximately the mean cosine of the angular distribution of downward travelling photons m_d, and hence that the downwards PAR absorbed per unit chlorophyll is $m_d^{-1} a_*^X$, or the attenuation cross-section ϵ^X. This assumption allows the same parameter to be used in relation to photosynthesis and PAR attenuation.

2.4 Microplankton compartment

The microplankton compartment of the model includes all pelagic micro-organisms less than 200 μm: that is, including heterotrophic bacteria and protozoa (zooflagellates, ciliates, heterotrophic dinoflagellates, . . .) as well as photo-autotrophic cyanobacteria and micro-algae (diatoms, dinoflagellates, flagellates, . . .). Flows of energy and materials through these organisms link them in the "microbial loop". All reproduce mainly by binary division, in contrast to the herbivorous mesozooplankton, such as copepods and the planktonic larvae of benthic animals, which typically have relatively long periods of individual growth after the laying of many eggs by those few females that survive to maturity. Many of the microbial populations in the loop have turnover rates of order 0.1 day^{-1} during the productive season, and can thus be parameterised as a unit without unduly distorting the response of the model at weekly frequencies.

An argument for explicit modelling of the compartments in the microbial loop is in order to make a distinction between autotrophs and heterotrophs. However, the distinction is not clear-cut in microplanktonic organisms. Not only do phytoplankton respire, but some protozoans retain ingested chloroplasts and some micro-algae can grow heterotrophically or mixotrophically. The microplankton may thus be seen from a functional viewpoint as a suspension of chloroplasts (and cyanobacteria) and mitochondria (and heterotrophic bacteria) with associated organic carbon and nitrogen. Tett (1987a) employed this viewpoint in a simple model in which bulk "photoautotrophic processes are made simple functions of chlorophyll concentration, representing algal biomass . . . ; heterotrophic processes are made simple functions of ATP concentration, representing total microplankton biomass . . . and *including the algal component.*"

As the diagrams in Figure 3.2.2 suggest, the microplankton compartment is the "microbial loop" in a box. Originally (Tett, 1990a), this box was seen as predominantly algal in character, with a heterotroph "contamination" merely exaggerating the effect of the algae's own heterotrophic processes — for example, increasing the microplankton's respiration rate. In the present model, however, the effects of an autotroph-heterotroph mixture are taken explicitly into account. This involves the parameter η, the ratio of microheterotroph to microplankton carbon biomass B:

$$\eta = \frac{B_h}{B_a + B_h} \tag{3.2.8}$$

where subscripts $_a$ and $_h$ indicate respectively *a*utotrophs (phytoplankton) and (pelagic micro-) *h*eterotrophs (protozoa and heterotrophic bacteria). In the equations hereafter, terms without subscripts refer to the microplankton as a whole. *It is a crucial assumption of this version of microplankton that the value of the heterotroph fraction does not change during a simulation.* Variation in η has been addressed by Smith and Tett (*m.s.*) using two microplankton compartments that vary in their relative contribution to total biomass and which differ in η.

It is further assumed than any organic matter excreted by phytoplankton or leaked during "messy feeding" by protozoans is re-assimilated by microplankton very rapidly (that is, at rates such that the turnover times of *labile* DOM pools are less than a day), so enabling the existence of a labile pool to be neglected. It is, similarly, assumed that inorganic nutrients especially, ammonium excreted by microheterotrophs are rapidly re-assimilated by the autotrophs and thus retained within the microplankton compartment.

The microplankton compartment has two independent state variables which are the sum of autotroph and heterotroph contributions, organic carbon B:

$$\mathcal{H}(B) = \beta(B) \tag{3.2.9}$$

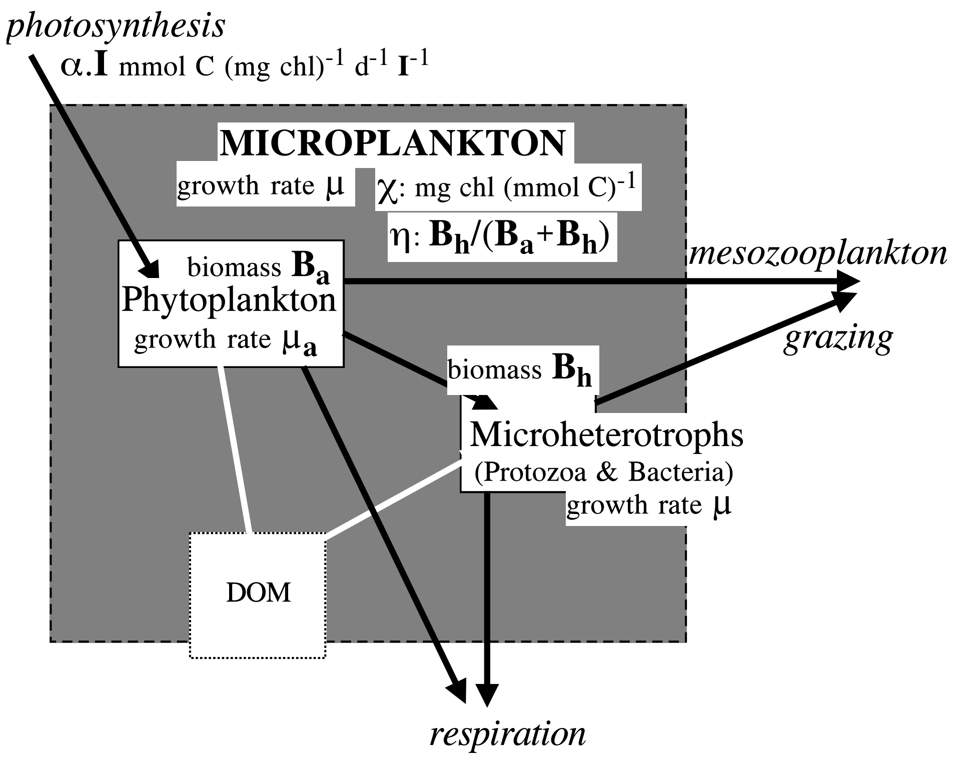


Figure 3.2.2: The microplankton compartment.

and organic nitrogen N [2]:

$$\mathcal{H}(N) = \beta(N) \tag{3.2.10}$$

with $\mathcal{H}$ and β defined by (3.2.3)–(3.2.4). Two other variables are linked to the above; the nitrogen quota Q (mmol N (mmol C)$^{-1}$):

$$Q = \frac{N}{B} \tag{3.2.11}$$

and the chlorophyll concentration X (mg Chl m^{-3}):

$$X = \chi B = {}^{X}q^{N} N \tag{3.2.12}$$

where χ is the ratio of chlorophyll to microplankton carbon, and ${}^{X}q^{N}$ is the ratio of chlorophyll to microplankton nitrogen.

Using the notation introduced by equation (3.2.4), the non-conservative term for microplankton carbon rate of change $\beta(B)$ (mmol C m^{-3} day^{-1}) in equation (3.2.9) can be expanded:

$$\beta(B) = \beta_a(B_a) + \beta_b(B_h) = (\mu_a - c_h B_h - G)B_a + (c_h B_a - r_h - G)B_h \tag{3.2.13}$$

where:

- μ_a is the relative growth rate (day^{-1}) of the autotrophs;
- G is mesozooplankton grazing pressure (the instantaneous probability per unit time that a microplankter will be consumed by a copepod or similar animal); it is assumed that this pressure applies equally to heterotrophs and autotrophs;
- $c_h B_h$ is the "transfer pressure" (day^{-1}) on autotrophs due to microplankton heterotrophs; it is analogous to the mesozooplankton grazing pressure;
- r_h is the relative respiration rate (day^{-1}) of microplankton heterotrophs.

If the heterotrophic component were solely herbivorous protozoa, then c_h might be seen as a clearance rate (volume of water made free of autotrophs by the grazers, per unit grazer biomass and time). However, there are other ways in which organic matter is transferred from producers to consumers, including photosynthetic, or grazing-induced, leakage of dissolved organic matter (DOM) which is assimilated by bacteria, and in turn grazed by protozoa (Figure 3.2.2). The quantity $c_h B_h B_a$ (mmol C m^{-3} day^{-1}) is thus best understood as the total organic carbon flux from autotrophs to microplankton heterotrophs. Details of the routes are of no importance so long as the transfer term appears identically (except for opposite sign) in $\beta_a(B_a)$ and $\beta_h(B_h)$, and so cancels. It is thus assumed that all DOM produced by leakage is labile and rapidly assimilated by other microplankters. It is also

[2] In this chapter only, the symbol N is used for microplankton nitrogen; it should not be confused with N for buoyancy frequency used elsewhere in the turbulence model (Section III-1.2).

assumed that protozoans do not defecate, and hence that the only intrinsic loss of carbon experienced by the microheterotrophs is through their respiration r_h (day^{-1}), so that their growth rate is:

$$\mu_h = c_h B_a - r_h \tag{3.2.14}$$

As intended, the transfer term $c_h B_h B_a$ cancels in equation (3.2.13), which can then be simplified:

$$\beta(B) = (\mu - G)B \tag{3.2.15}$$

where:

$$\mu = \mu_a(1 - \eta) - r_h\eta \tag{3.2.16}$$

It may be noted that microheterotroph growth rate μ_h must be the same as microplankton growth rate μ if the heterotrophs are to remain a constant fraction of the microplankton.

The non-conservative term for microplankton nitrogen rate of change $\beta(N)$ (mmol N m^{-3} day^{-1}) in equation (3.2.10) can be expanded:

$$\beta(N) = \beta_a(N) + \beta_h(N) = (u_a - (c_h B_h + G)Q_a)B_a + (c_h Q_a B_a - {}^N r_h - Gq_h)B_h \tag{3.2.17}$$

where:

- u_a is the autotroph biomass-related nutrient uptake rate, in the case of nitrogen the sum of ammonium and nitrate uptakes (mmol N (mmol C)$^{-1}$ day^{-1});
- ${}^N r_h$ is the heterotroph ammonium excretion rate (mmol N (mmol C)$^{-1}$ day^{-1}).

As in the case of carbon, the transfer of nitrogen from autotrophs to heterotrophs is not a loss when included within the microplankton compartment, and the equation simplifies to:

$$\beta(N) = uB - GN = (u - GQ)B \tag{3.2.18}$$

where:

$$Q = Q_a(1 - \eta) + q_h\eta \tag{3.2.19}$$

and

$$u = u_a(1 - \eta) - {}^N r_h\eta \tag{3.2.20}$$

where q_h is the heterotroph nitrogen to carbon quota.

It will be seen that the parameters for the microplankton model can be derived from autotroph and heterotroph parameters given the choice of value for η. It is convenient, for the derivations that follow, to assume that autotrophs and heterotrophs are always in analogous physiological states — i.e. both are either simultaneously nitrogen-limited or simultaneously carbon-limited at a given time. This is not a fundamental requirement, but serves to simplify the model equations, avoid discontinuities, and reduce the number of logical tests to be made during numerical simulations (see Figure 3.2.3). Table 3.2.3 summarises the parameters on which the microplankton model is based, and gives the range of possible values. Derived parameters are given in Table 3.2.4. Their values are calculated by the program using the expressions given in that table and the parameters listed in Table 3.2.3.

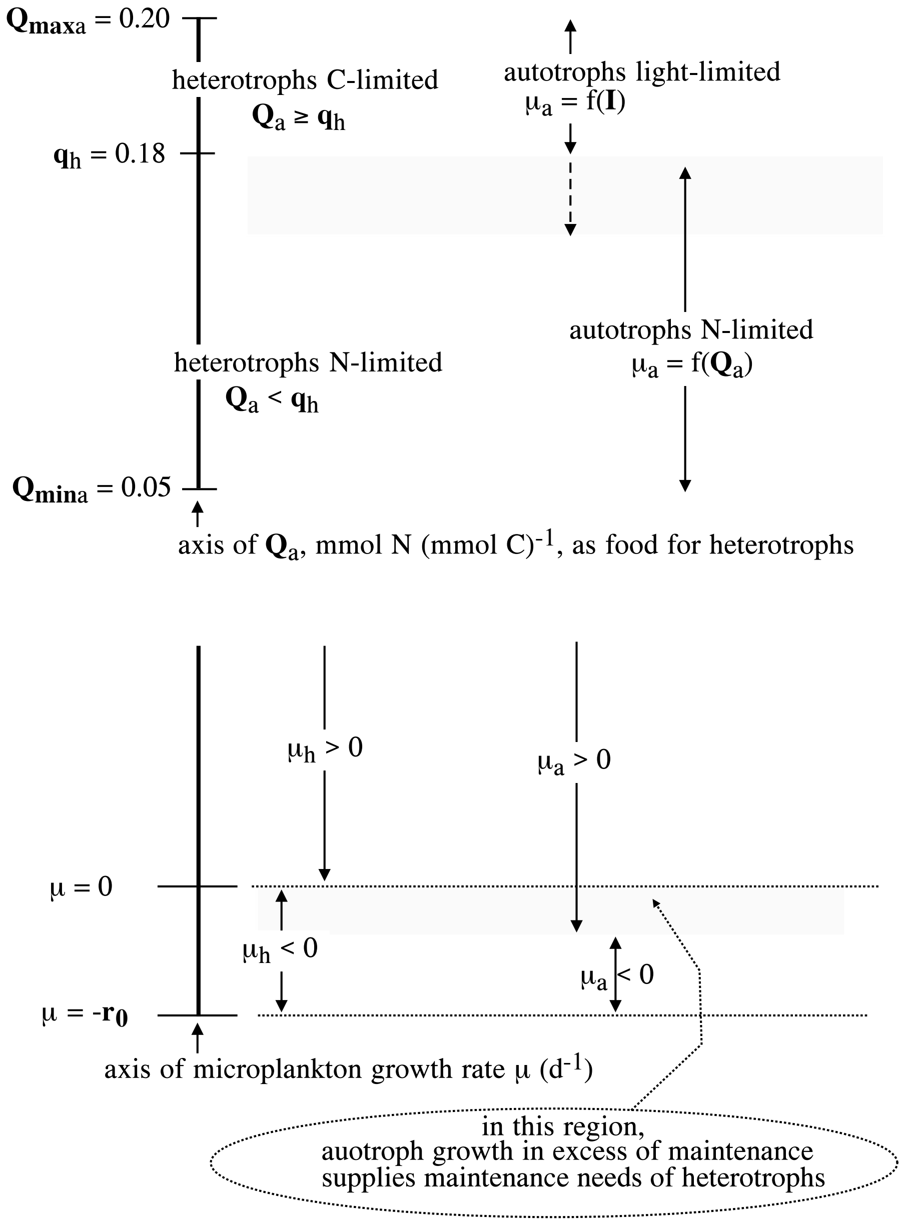


Figure 3.2.3: Limitation states of the microplankton.

Table 3.2.3: Basic parameters for the microplankton model and their typical ranges.

symbol	unit	range	purpose
autotrophs			
Φ	nmol C μE^{-1}	40–60	photosynthetic quantum yield
k	mmol nmol^{-1} day^{-1} μE W^{-1}	0.0864*4.15	converts $\epsilon^X\Phi$ to typical units of α
${}^Xq_a^N$	mg Chl (mmol N)$^{-1}$	0.5–7	chlorophyll to nitrogen ratio
r_{0_a}	day^{-1}	0.03–0.41	basal respiration
b_a	—	0.2–0.7	growth rate related respiration slope
Q_{min_a}	mmol N (mmol C)$^{-1}$	0.02–0.07	minimum nitrogen to carbon quota
Q_{max_a}	mmol N (mmol C)$^{-1}$	0.15–0.25	maximum nitrogen to carbon quota
μ_{max_a}	day^{-1}	1.2–3.0	maximum (nutrient-controlled) growth rate
${}^{NO}u_{max_a}$	mmol N (mmol C)$^{-1}$day^{-1}	0.2–0.6	maximum nitrate uptake rate
${}^{NH}u_{max_a}$	mmol N (mmol C)$^{-1}$day^{-1}	1.5	maximum ammonium uptake rate
k_{NOS}	mmol N m^{-3}	0.32	half-saturation constant for nitrate uptake
k_{NHS}	mmol N m^{-3}	0.24	half-saturation constant for ammonium uptake
k_{in}	mmol N m^{-3}	0.5–7.0	coefficient for inhibition of nitrate uptake due to ammonium
heterotrophs			
q_h	mmol N (mmol C)$^{-1}$	0.15–0.22	nitrogen to carbon quota
r_{0_h}	day^{-1}	0–0.05	basal respiration
b_h	—	1–3	growth rate related respiration slope
mesozooplankton			
γ	—	0.8	grazed fraction assimilated by zooplankton
e	—	0.5	fraction of nitrogen excreted as ammonium

Table 3.2.4: Derived microplankton parameters.

parameter	unit	derived
heterotroph fraction	—	$\eta = B_h/(B_a + B_h)$
photosynthetic efficiency	mmol C (mg Chl)$^{-1}$ day^{-1} (W m^{-2})$^{-1}$	$\alpha = k\epsilon^X\Phi$
yield of chlorophyll from N	mg Chl (mmol N)$^{-1}$	${}^Xq^N = {}^Xq_a^N(1-\eta)$
basal respiration rate	day^{-1}	$r_0 = r_{0_a}(1-\eta)+r_{0_h}\eta(1+b_a)$
respiration slope	day^{-1}	$b = b_a(1 + b_h\eta) + b_h\eta$ for $\mu > 0$ $b = 0$ for $\mu \leq 0$
max. N-limited growth rate	day^{-1}	$\mu_{max} = \mu_{max_a}(1-\eta)/(1.2-\eta)$
minimum N quota	mmol N (mmol C)$^{-1}$	$Q_{min} = Q_{min_a}(1-\eta) + q_h\eta$
maximum N quota	mmol N (mmol C)$^{-1}$	$Q_{max} = Q_{max_a}(1-\eta) + q_h\eta$
maximum NO_3 uptake	mmol N (mmol C)$^{-1}$ day^{-1}	${}^{NO}u_{max} = {}^{NO}u_{max_a}(1-\eta)$
maximum NH_4 uptake	mmol N (mmol C)$^{-1}$ day^{-1}	${}^{NH}u_{max} = {}^{NH}u_{max_a}(1-\eta)$
half-sat. conc. NH_4 uptake	mmol N m^{-3}	k_{NHS} = autotroph value
half-sat. conc. NO_3 uptake	mmol N m^{-3}	k_{NOS} = autotroph value
half-sat. conc. NH_4 inhibition	mmol N m^{-3}	k_{in} = autotroph value

Table 3.2.5: Values of the parameters in the temperature factor (3.2.21).

symbol	unit	value	purpose
q_T	$(^0C)^{-1}$	0.07	growth rate coefficient in temperature growth factor $f(T)$
T_r	^0C	20	reference temperature in temperature growth factor $f(T)$

2.4.1 Temperature effects

In the model, water temperature influences four key rates, those of nutrient-controlled growth, nutrient uptake, detrital remineralisation and nitrification. The influence is given by a factor based on Eppley (1972):

$$f(T) = e^{q_T(T-T_r)} \tag{3.2.21}$$

where the reference temperature T_r is 20 ^{0}C. This is the temperature at which several biological rates are measured in the laboratory. Parameters values are given in Table 3.2.5.

2.4.2 Growth rate

Microplankton growth rate is calculated on the basis of the threshold limitation

$$\mu = \min[\mu_1(I_p), \mu_2(Q)] \tag{3.2.22}$$

The equation is expanded on the basis of laboratory derived theory for the growth rate of photoautotrophs and microheterotrophs. In the case of the autotrophs (i.e. phytoplankton) growth rate is calculated from cell quota threshold limitation (CQTL) theory (Droop et al., 1982) as the least growth rate predicted from light or nutrient controlled growth:

$$\mu_a = \min[\mu_{a1}(I_p), \mu_{a2}(Q)] \tag{3.2.23}$$

For a mixed population of autotrophs and heterotrophs growth rate is the result of net growth less net respiration.

$$\mu = \mu_h = \mu_a(1-\eta) - r_h\eta \tag{3.2.24}$$

The equivalence of μ and μ_h is required if η is to be a constant and autotroph growth rate is therefore greater than that of the microplankton as a whole, in order to supply the respiratory needs of the heterotrophs.

a) light-controlled growth

Under *light-limiting* conditions, growth rate is the sum of photosynthetic production and respiration losses. Photosynthesis is described as a linear photosynthesis-irradiance relationship which simplifies integration over an optically thick layer (Tett, 1990b) and is a reasonable approximation under most light-limiting conditions (Droop et al., 1982).

$$\mu_1(I_p) = (\alpha\chi_a I_p - r_a)(1-\eta) - r_h\eta \tag{3.2.25}$$

where α is the photosynthetic efficiency in (mmol C)(mg Chl)$^{-1}$day^{-1}W^{-1}m^2 and χ_a is the ratio of chlorophyll to autotroph carbon. Photosynthetic efficiency is the product of phytoplankton attenuation cross-section ϵ^X (m^2(mg Chl)$^{-1}$), photosynthetic quantum yield Φ (nmol C μE^{-1}) and a factor k to convert units, which includes a conversion from μE to Joules. This is not subscripted because chlorophyll is a property of autotrophs only, and thus its concentration is unaffected by adding heterotrophs. Droop et al. (1982) showed that photosynthetic efficiency was high in light-limited *Pavlova*, and lower when the algae were nutrient-controlled. The model treats the efficiency parameters ϵ^X and Φ as constants (in a given simulation), on the grounds that they are not used to calculate nutrient-controlled growth rate.

Respiration losses r_a and r_h are formulated as basal and growth rate related components as there is evidence that micro-algal and microheterotroph respiration depend more on growth rate than on temperature (some temperature effect being expected because of the assumed dependency of maximum growth rate) (Tett and Droop, 1988; Tett, 1990b). For micro-algae:

$$r_a = r_{0_a} + b_a\mu_{a1}(I_p) \tag{3.2.26}$$

and assuming that heterotroph carbon-limitation always corresponds to autotroph light-limitation, heterotroph respiration can be written as:

$$r_h = r_{0_h} + b_h\mu_1(I_p) \tag{3.2.27}$$

The standard values for basal respiration were chosen as 0.05 day^{-1} for r_{0_a} and 0.02 day^{-1} for r_{0_h} because higher values make it difficult for algae to survive over-winter during simulations of the seasonal cycle in temperate seas. Respiration slopes b_a and b_h were set at 0.5 and 1.5 respectively.

Using (3.2.24)–(3.2.27) the microplankton growth rate under light-limiting conditions can then be written:

$$\mu_1(I_p) = \frac{\alpha I_p\chi - r_0}{1+b} \tag{3.2.28}$$

where the terms are:

- the ratio of chlorophyll to microplankton carbon:

$$\chi = \chi_a(1-\eta) = {}^{X}q_a^N(Q - q_h\eta) \quad \text{mg Chl (mmol } C)^{-1} \tag{3.2.29}$$

- the basal respiration rate:

$$r_0 = r_{0_a}(1-\eta) + r_{0_h}\eta(1+b_a) \quad \text{day}^{-1} \tag{3.2.30}$$

- the respiration slope:

$$\begin{array}{ll} b = b_a(1 + b_h\eta) + b_h\eta & \text{if} \quad \mu > 0 \\ b = 0 & \text{if} \quad \mu \leq 0 \end{array} \tag{3.2.31}$$

The chlorophyll to carbon ratio has been expanded using $\chi_a = {}^Xq_a^N Q_a$ and the condition when $\mu \leq 0$ is a simplification of two conditions $\mu \leq 0$ and $\mu_a \leq 0$.

The ratio of chlorophyll to autotroph organic carbon, χ_a is known to vary over a wide range 0.04 to 1 mg Chl (mmol C)$^{-1}$ as a function of temperature, irradiance and nutrient status in the laboratory (Sakshaug et al., 1989; Laws and Bannister, 1980; Geider et al., 1997; Cloern et al., 1995; Baumert, 1996). Droop et al. (1982) showed that $\alpha\chi_a$ decreased under nutrient-limiting conditions, the explanation developed by Droop (1985) including a decrease in quantum yield. These complex interactions can be simply (if approximately) represented by changes in the phytoplankton chlorophyll to carbon ratio, and in turn linked to the cell quota. The chlorophyll to nitrogen quota ${}^Xq_a^N$ is treated as a constant estimated from published data for algal cultures (Caperon and Meyer, 1972a,b; Levasseur et al., 1993; Sakshaug et al., 1989; Sosik and Mitchell, 1991; Tett et al., 1985; Zehr et al., 1988). The pigment data were obtained by spectrophotometric or fluorometric methods, and thus overestimate chlorophyll*a* determined by precise chromatographic methods (Mantoura et al., 1997; Gowen et al., 1983). Nevertheless, they are appropriate for a model intended for comparison with most field data. The culture data contained a wide range of values of the ratio of chlorophyll to nitrogen without any clear overall pattern in relation to cell size, growth rate or irradiance (below 300 μE m^{-2} s^{-1}). Ignoring a few values of more than 7 mg Chl (mmol N)$^{-1}$, the median was 2.2 mg Chl (mmol N)$^{-1}$, and the default value for ${}^Xq_a^N$ is taken in the model as 2.0 mg Chl (mmol N)$^{-1}$.

b) nutrient-controlled growth

Under *nitrogen-limiting* conditions growth rate should, ideally, be calculated as a fraction of the maximum specific growth rate depending on the cell nutrient quota, less respiration.

$$\mu_2(Q) = \mu_{max_a} f(T)(1 - \frac{Q_{min_a}}{Q_a})(1 - \eta) - r_h\eta \tag{3.2.32}$$

where Q_a is the (variable) autotroph cell quota and $f(T)$ the temperature growth factor given by equation (3.2.21). Assuming that heterotrophs as well as autotrophs are nitrogen-limited, heterotroph respiration is:

$$r_h = \frac{\mu_h(1 + b_h) + r_{0_h}}{q^*} \tag{3.2.33}$$

where $q^* = Q_a/q_h$. Substituting Q_a by $(Q - q_h\eta)/(1 - \eta)$ gives:

$$\mu_2(Q) = \frac{\mu_{max} f(T) f_1(Q)}{1 + (1 + b_h) f_2(Q) - \eta} - r_{0_h} f_2(Q) \tag{3.2.34}$$

where:

$$\mu_{max} = \mu_{max_a}(1-\eta) \quad \text{day}^{-1} \tag{3.2.35}$$

$$\begin{aligned} f_1(Q) = \frac{Q - Q_{min}}{Q - q_h\eta} \quad &\text{for} \quad Q \geq Q_{min} \\ f_1(Q) = 0 \quad &\text{for} \quad Q < Q_{min} \end{aligned} \tag{3.2.36}$$

$$\begin{aligned} f_2(Q) &= \frac{q_h\eta(1-\eta)}{Q - q_h\eta} \quad \text{for} \quad Q \geq Q_{min} \\ f_2(Q) &= \frac{q_h\eta(1-\eta)}{Q_{min} - q_h\eta} \quad \text{for} \quad Q < Q_{min} \end{aligned} \tag{3.2.37}$$

$$Q_{min} = Q_{min_a}(1-\eta) + q_h\eta \quad \text{mmol } N \text{ (mmol } C)^{-1} \tag{3.2.38}$$

In (3.2.34)–(3.2.38) Q_{min} denotes the minimum microplankton nutrient quota and μ_{max} the limit for microplankton relative growth rate as Q becomes infinite.

Equation (3.2.34) is complicated, and can be approximated (Figure 3.2.4) by:

$$\mu_2(Q) = \mu_{max} f(T) f_3(Q) \tag{3.2.39}$$

where:

$$\mu_{max} = \mu_{max_a} \frac{1-\eta}{1.2-\eta} \quad \text{day}^{-1} \tag{3.2.40}$$

$$\begin{aligned} f_3(Q) = \frac{Q - Q_{min}}{Q} = 1 - \frac{Q_{min}}{Q} \quad &\text{for} \quad Q \geq Q_{min} \\ f_3(Q) = 0 \quad &\text{for} \quad Q < Q_{min} \end{aligned} \tag{3.2.41}$$

with Q_{min} given by (3.2.38). The function $f_3(Q)$ in (3.2.39) is the cell-quota function of Droop (1968). The contribution of microheterotroph basal respiration r_{0_h} is assumed to be insignificant. The term $1.2 - \eta$ was obtained empirically. Equation (3.2.39) is the version used in COHERENS.

2.4.3 Nutrient uptake

Autotrophs take up both nitrate ${}^{NO}S$ and ammonium ${}^{NH}S$ (mmol N (mmol C)$^{-1}$ day^{-1}):

$$u = u_a(1-\eta) = ({}^{NO}u_a + {}^{NH}u_a)(1-\eta) \tag{3.2.42}$$

It is assumed that nitrate uptake can be inhibited by the presence of ammonium as well as by a high cell quota, but never involves excretion. Thus, substituting microplankton for autotroph parameters

$${}^{NO}u = {}^{NO}u_a(1-\eta) = {}^{NO}u_{max} f(T) f({}^{NO}S) f_{in}({}^{NH}S) f_{in1}(Q) \tag{3.2.43}$$

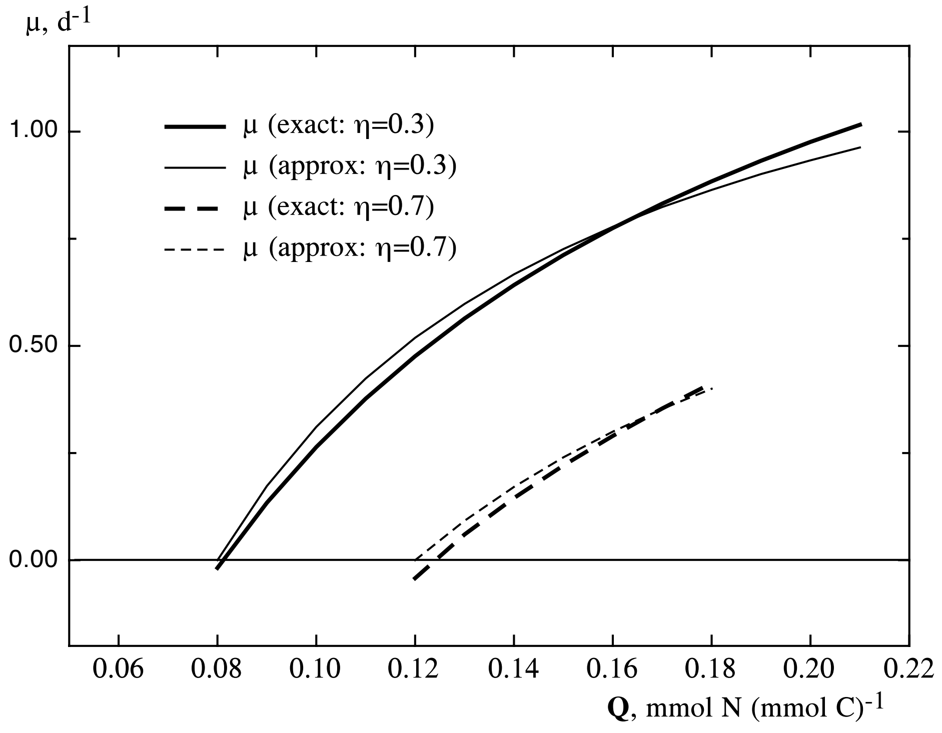


Figure 3.2.4: Comparison of microplankton nutrient-limited growth according to the exact equation (3.2.34) and the approximation (3.2.39).

where:

$$^{NO}u_{max} = {}^{NO}u_{max_a}(1-\eta) \tag{3.2.44}$$

$$f(^{NO}S) = \frac{^{NO}S}{k_{NOS} + {}^{NO}S} \tag{3.2.45}$$

$$f_{in}(^{NH}S) = (1 + \frac{^{NH}S}{k_{in}})^{-1} \tag{3.2.46}$$

$$\begin{aligned} f_{in1}(Q) &= 1 - \frac{Q - q_h\eta}{Q_{max} - q_h\eta} \quad \text{for} \quad Q \le Q_{max} \\ f_{in1}(Q) &= 0 \quad \text{for} \quad Q > Q_{max} \end{aligned} \tag{3.2.47}$$

where:

$$Q_{max} = Q_{max_a}(1-\eta) + q_h\eta \tag{3.2.48}$$

and $f(T)$ is the temperature growth factor given by equation (3.2.21). The function $f(^{NO}S)$ represents the nitrate uptake by Michaelos-Menton type kinetics, where k_{NOS} is the half saturation rate for nitrate uptake; $f_{in}(^{NH}S)$ accounts for inhibition of nitrate uptake due to the presence of ammonium; and $f_{in1}(Q)$ allows nitrate uptake to vary with cell quota where Q_{max} is the maximum microheterotroph nitrogen to carbon quota.

Ammonium uptake is not inhibited by nitrate, but could be negative if the microplankton nitrogen quota exceeds a maximum, or if heterotroph excretion exceeds autotroph assimilation:

$$^{NH}u = {}^{NH}u_{max} f(T) f(^{NH}S) f_{in2}(Q) - {}^{N}r_h\eta \tag{3.2.49}$$

where:

$$^{NH}u_{max} = {}^{NH}u_{max_a}(1-\eta) \tag{3.2.50}$$

$$\begin{aligned} f(^{NH}S) &= \frac{^{NH}S}{k_{NHS} + {}^{NH}S} \quad \text{for} \quad Q \le Q_{max} \\ f(^{NH}S) &= 1 \quad \text{for} \quad Q > Q_{max} \end{aligned} \tag{3.2.51}$$

$$f_{in2}(Q) = 1 - \frac{Q - q_h\eta}{Q_{max} - q_h\eta} \tag{3.2.52}$$

where Q_{max} is given by (3.2.48) and $f(T)$ is the temperature growth factor given by equation (3.2.21). The term for ammonium excretion by microheterotrophs can be defined with microplankton parameters using $q^\dagger$:

$$^{N}r_h = \mu((1+b_h).q^\dagger - q_h) + r_{0_h}q^\dagger \tag{3.2.53}$$

where:

$$\begin{aligned} q^\dagger &= \frac{Q - q_h\eta}{1-\eta} \quad \text{for} \quad Q \ge q_h \\ q^\dagger &= q_h \quad \text{for} \quad Q < q_h \end{aligned} \tag{3.2.54}$$

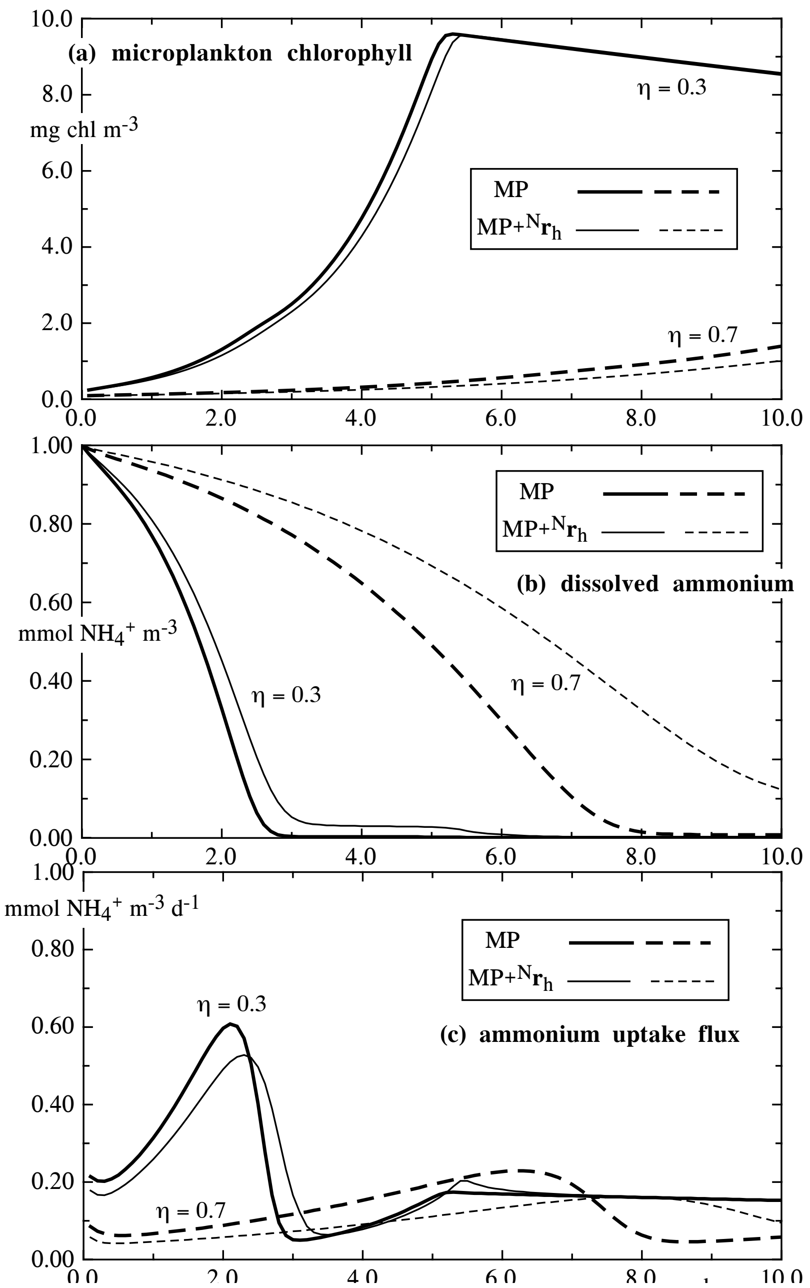


Figure 3.2.5: The figures show the effect on microplankton biomass (top), dissolved ammonium (middle) and and ammonium uptake (bottom) of the two formulations for ammonium uptake. MP uses equation (3.2.55), MP+$^{N}r_h$ uses (3.2.49).

Table 3.2.6: Microplankton rate equations.

parameter	unit	equation	
growth rate	day^{-1}	$\mu = \min[\mu_1(I_p), \mu_2(Q)]$	
	day^{-1}	$\mu_1(I_p) = (\alpha {I_p}^X q_a^N (Q - q_h\eta) - r_0)/(1+b)$	
	day^{-1}	$\mu_2(Q) = \mu_{max} f(T)(1 - Q_{min}/Q)$	$Q \geq Q_{min}$
		$\mu_2(Q) = 0$	$Q < Q_{min}$
uptake rate	mmol N (mmol C)$^{-1}$ day^{-1}	$u = {}^{NO}u + {}^{NH}u$	
NO_3^-	mmol N (mmol C)$^{-1}$ day^{-1}	${}^{NO}u = {}^{NO}u_{max} f(T) f({}^{NO}S) f_{in}({}^{NH}S) f_{in1}(Q)$	
NH_4^+	mmol N (mmol C)$^{-1}$ day^{-1}	${}^{NH}u = {}^{NH}u_{max} f(T) f({}^{NH}S) f_{in2}(Q)$	
uptake terms			
		$f({}^{NO}S) = {}^{NO}S/(k_{NOS} + {}^{NO}S)$	
		$f({}^{NH}S) = {}^{NH}S/(k_{NHS} + {}^{NH}S)$	$Q \leq Q_{max}$
		$f({}^{NH}S) = 1$	$Q > Q_{max}$
		$f_{in}({}^{NH}S) = (1 + {}^{NH}S/k_{in})^{-1}$	
		$f_{in1}(Q) = 1 - (Q - q_h\eta)/(Q_{max} - q_h\eta)$	$Q \leq Q_{max}$
		$f_{in1}(Q) = 0$	$Q > Q_{max}$
		$f_{in2}(Q) = 1 - (Q - q_h\eta)/(Q_{max} - q_h\eta)$	
grazing rate	day^{-1}	$G =$ prescribed	

This formulation for the uptake rate is complicated because ${}^{N}r_h$ is a function of two variables, Q and $\mu(I_p, Q)$. Test cases (see Figure 3.2.5) show that uptake predicted by equation (3.2.49), with ${}^{N}r_h$ defined by (3.2.53), does not becomes negative. This is because ammonium excreted by heterotrophs increases the ambient concentration and so gives rise to additional autotroph uptake. Thus, in accord with the view that excreted material (but not respired carbon) is recycled within the microplankton, ammonium excretion is assumed to be zero (except in special circumstances, see below), and so

$$ {}^{NH}u = {}^{NH}u_{max} f(T) f({}^{NH}S) f_{in2}(Q) \tag{3.2.55} $$

where the parameters and included functions are as in (3.2.49). This equation does allow for ammonium to be excreted, but only when $Q > Q_{max}$ (when $f_{in2}(Q) < 0$). Such a condition is possible, for example when irradiance is very low and as a result light-controlled growth is negative because of the effects of respiration. Without allowing (in the model) for nitrogen excretion, the autotroph (and hence microplankton) cell quota would continue to increase. Equation (3.2.55) is the version used in COHERENS.

2.4.4 Grazing

Phytoplankton are grazed by a range of micro- and meso-zooplankton. In the model micro-zooplankton grazing is implicitly included in the parameterisation of the microplankton community equations. Meso-zooplankton grazing is implemented as a seasonally varying grazing pressure. Grazing pressure is calculated from observations of zooplankton abundance and type. In European shelf waters a good source of data is from the continuous plankton recorder (CPR) programme. Where copepods dominate the meso-zooplankton community, grazing pressure $G(t)$ may be calculated as:

$$ G(t) = f^Z Z(t) \tag{3.2.56} $$

where $Z(t)$ is the number of adult copepods and f^Z is the rate at which seawater is stripped of phytoplankton and micro-zooplankton by individual animals (e.g. Paffenhöfer 1971; Paffenhöfer and Harris 1976). The grazing pressure is supplied to the program at monthly intervals. Example values are listed in Table 3.2.7. In the absence of any knowledge of local mesozooplankton the default may be taken as a constant value of 0.05 day^{-1}.

A fraction γ of the grazed microplankton carbon and nitrogen is digested and assimilated by the animals and a portion e of the assimilated nitrogen is deemed immediately metabolised and excreted as ammonium. The remaining fraction of the grazed material $1 - \gamma$ is defaecated and becomes detritus. By this formulation an amount $GN\gamma(1 - e)$ of nitrogen is lost from the model system and so the ecosystem described by the model is "open" at the second trophic level and simulated production will "run down" unless the lost nitrogen is replaced (see benthic boundary conditions, Section III-2.9.1)[3].

[3]Note that the accumulation of zooplankton nitrogen "lost" from the model system must be recorded and accounted for in the summation of total nitrogen when nitrogen conservation is checked in time (see Chapter II-5).

Table 3.2.7: Example values for the monthly averaged grazing pressures (day^{-1}) taken from station CS in the North Sea (Tett and Walne, 1995).

January	0.01	April	0.03	July	0.05	October	0.04
February	0.02	May	0.05	August	0.06	November	0.06
March	0.02	June	0.07	September	0.03	December	0.02

Table 3.2.8: Detritus parameters and their default values in COHERENS.

symbol	unit	value	purpose
${}^{M}r_{max}$	day^{-1}	0.08	maximum detrital nitrogen remineralisation rate
${}^{C}r_{max}$	day^{-1}	0.06	maximum detrital carbon remineralisation rate
$O_{1/2}$	mmol O m^{-3}	10.0	half-saturation constant for nitrogen dependent detrital carbon respiration
${}^{M}q^{C}_{min}$	mmol N (mmol C)$^{-1}$	0.09	minimum detrital nitrogen to carbon quota

These losses are represented in the model in the form of accumulated zooplankton nitrogen. Hence,

$$\beta(Z_N) = GN\gamma(1-e) \tag{3.2.57}$$

2.5 Detritus compartment

Organic detritus is composed of faecal material, organic waste products, dead and dying plant and animal material and hosts a diverse fauna of bacteria, protozoa and microbes. Suspended particles attenuate light in the euphotic layer and release minerals and nutrients back into the water column as micro-organisms break down the organic material. These remineralised nutrients are taken up by the phytoplankton and fuel additional regenerated production. Sinking detritus particles fall to the sea bed where they are consumed by benthic animals, buried in the sediment or resuspended into the water column during periods of increased current velocity.

2.5.1 Generation

Modelled organic detritus originates solely from zooplankton defaecation which is one of the most important sources of detritus in the water column away from coastal and river influences. Subsequent versions of the model may include the formation of phytodetritus by aggregation of some types of phytoplankton (Alldredge et al., 1995; Hill, 1992). Detritus is modelled as organic carbon C:

$$\mathcal{H}(C) = \beta(C) \tag{3.2.58}$$

and nitrogen M [4]

$$\mathcal{H}(M) = \beta(M) \tag{3.2.59}$$

with $\mathcal{H}$, β defined by (3.2.3)–(3.2.4). The non-conservative terms $\beta(C)$ and $\beta(M)$ can be expanded:

$$\beta(C) = (1-\gamma)GB - {}^{C}rC \tag{3.2.60}$$

$$\beta(M) = (1-\gamma)GN - {}^{M}rM \tag{3.2.61}$$

where $1-\gamma$ is the fraction of defecated grazed carbon GB or nitrogen GN and ${}^{C}r$ and ${}^{M}r$ are the remineralisation rates for carbon and nitrogen respectively. Both C and M are expressed as mmol m^{-3}.

2.5.2 Remineralisation

Once in the detrital pool the organic material is assumed to be colonised by a range of remineralising bacteria, protozoa and microheterotrophs. These micro-organisms are distinguished from those included within the microplankton compartment by their slower turn-over rates compared to those of microplankton. Their biomass in the model is included in detrital carbon and nitrogen and their remineralising activities parameterised in relation to the total amount of detritus, its "food quality" and the temperature. Fresh labile detritus is rich in nutrients and minerals which are rapidly utilised and remineralised as the particle ages. The "food quality" of the detritus can be modelled in terms of its nitrogen quota with a high nitrogen to carbon ratio corresponding to more labile, and a low nitrogen to carbon ratio to more refractory material. There is some evidence to suggest that microbial activity increases with temperature. In this case detrital remineralisation occurs faster in warmer water and rates will vary seasonally and with depth.

Detrital remineralisation converts organic detrital nitrogen into ammonium whilst the degradation of detrital carbon consumes oxygen.

The rate of remineralisation of detrital carbon and nitrogen are:

$$\begin{aligned} {}^{C}r = f(T)f(O)\,{}^{C}r_{max}\Big(1 - \frac{{}^{M}q^{C}_{min}}{{}^{M}q^{C}}\Big)^2 \quad &\text{for} \quad {}^{M}q^{C} \geq {}^{M}q^{C}_{min} \\ {}^{C}r = 0 \quad &\text{for} \quad {}^{M}q^{C} < {}^{M}q^{C}_{min} \end{aligned} \tag{3.2.62}$$

$$\begin{aligned} {}^{M}r = f(T)\,{}^{M}r_{max}\Big(1 - \frac{{}^{M}q^{C}_{min}}{{}^{M}q^{C}}\Big)^2 \quad &\text{for} \quad {}^{M}q^{C} \geq {}^{M}q^{C}_{min} \\ {}^{M}r = 0 \quad &\text{for} \quad {}^{M}q^{C} < {}^{M}q^{C}_{min} \end{aligned} \tag{3.2.63}$$

[4]In this chapter only, the symbol M is used for detrital nitrogen; it should not be confused with M for shear frequency used elsewhere in the turbulence model (Section III-1.2).

where:

$$^{M}q^{C} = \frac{M}{C} \tag{3.2.64}$$

is the detrital nitrogen quota and

$$f(O) = \frac{O}{O_{1/2} + O} \tag{3.2.65}$$

In (3.2.62)–(3.2.63), $^{M}r_{max}$ and $^{C}r_{max}$ are the maximum detrital nitrogen and carbon remineralisation rates, $^{M}q^{C}_{min}$ is the minimum detrital nitrogen quota, O is the ambient oxygen concentration and $O_{1/2}$ is the half-saturation constant for nitrogen-dependent detrital carbon respiration. Maximum remineralisation rates are usually measured under laboratory conditions at $T_r = 20$ ^{0}C and so the temperature is offset by this amount (see Section III-2.4.1). The maximum detrital nitrogen remineralisation rate is always greater than the corresponding rate for carbon so that the remineralisation of nitrogen proceeds faster than the degradation of carbon, and the detritus becomes more refractory with time. As the detritus becomes very refractory $^{M}q^{C}$ approaches $^{M}q^{C}_{min}$ and bacterial production of ammonium tends to zero (Lancelot and Billen, 1985).

2.6 Dissolved nutrients and oxygen

The biological model includes three dissolved substances: oxygen, nitrate and ammonium. Oxygen and nitrate are reciprocally linked, with ammonium oxidation producing nitrate but consuming oxygen and, with microplankton growth in general consuming nitrate but producing oxygen. The direct utilisation of ammonium by microplankton leads to additional oxygen production. This is balanced by the oxygen utilised during detrital degradation and so the system remains in quasi-equilibrium. Under stratified conditions, near-surface waters become saturated in oxygen and depleted in nutrient, and some oxygen may be lost across the air–sea interface. Bottom waters tend to have lower oxygen concentrations and increased levels of nitrate and ammonium resulting from the mineralisation of detritus. The rate of change of each dissolved substance is written in the form given by (3.2.3)–(3.2.4). For nitrate the non-conservative term results from losses to microplankton uptake ^{NO}u and gains from ammonium nitrification ^{NH}r:

$$\beta(^{NO}S) = -^{NO}uB + {^{NH}r}.{^{NH}S} \tag{3.2.66}$$

In the case of ammonium the non-conservative processes are losses by microplankton uptake and nitrification and gains from zooplankton grazing and detrital remineralisation ^{M}r:

$$\beta(^{NH}S) = -^{NH}uB - {^{NH}r}.{^{NH}S} + e\gamma GN + {^{M}r}M \tag{3.2.67}$$

The non-conservative terms for oxygen include gains from microplankton uptake of nitrate and growth (which is the result of photosynthesis less respiration), and losses to nitrification, detrital respiration ^{C}r and remineralisation due to grazing by mesozooplankton:

$$\beta(O) = ({^{O}q^{B}}\mu + {^{O}q^{NO}}.{^{NO}u} - {^{O}q^{C}}e\gamma G)B - {^{O}q^{NH}}.{^{NH}r}.{^{NH}S} - {^{O}q^{C}}.{^{C}r}C \tag{3.2.68}$$

Table 3.2.9: Dissolved nutrients and oxygen parameters and their default values in COHERENS.

symbol	unit	value	purpose
${}^{O}q^{B}$	mmol O (mmol C)$^{-1}$	1.0	photosynthetic and respiratory quotient for microplankton carbon growth
${}^{O}q^{NO}$	mmol O (mmol N)$^{-1}$	2.0	photosynthetic and respiratory quotient for nitrate uptake
${}^{O}q^{NH}$	mmol O (mmol N)$^{-1}$	2.0	photosynthetic and respiratory quotient for nitrification
${}^{O}q^{C}$	mmol O (mmol C)$^{-1}$	1.0	photosynthetic and respiratory quotient for detrital respiration
${}^{NH}r_{max}$	day^{-1}	0.1	maximum rate of nitrification
$O_{1/2,nit}$	mmol O m^{-3}	30.0	half-saturation constant for oxygen used in nitrification

The "q" coefficients are the photosynthetic and respiratory quotients for microplankton carbon growth ${}^{O}q^{B}$, and nitrate uptake ${}^{O}q^{NO}$, nitrification ${}^{O}q^{NH}$ and detrital respiration ${}^{O}q^{C}$.

Nitrification is the oxidation of ammonium to nitrate by the nitrifying bacteria *Nitromonas* and *Nitrobacter*. These bacteria are assumed to be abundant at all times, and given sufficient oxygen, allow nitrification to proceed rapidly. Thus their biomass is not explicitly represented in the model. The process of nitrification involves two steps: the oxidation of ammonium to nitrite, and the oxidation of nitrite to nitrate. As the intermediate species nitrite is not included in the model, nitrification is modelled as a single stage first-order process and the parameters are chosen to reflect the slower oxidation of ammonium to nitrite:

$$ {}^{NH}r = f(T)\,{}^{NH}r_{max}\Big(\frac{O}{O_{1/2,nit}+O}\Big) \tag{3.2.69}$$

where $f(T)$ is the temperature function given by (3.2.21) (equivalent to that used in detritus degradation),${}^{NH}r_{max}$ is the maximum rate of nitrification, and $O_{1/2,nit}$ is the half saturation constant for oxygen. The maximum nitrification rate ${}^{NH}r_{max}$ was originally (Tett, 1990a) taken as 1.0 day^{-1}, but simulations for the North Sea (D. Hydes, *pers. com.*) suggest that 0.1 day^{-1} is better.

2.7 Inorganic sediment

Inorganic sediment in the water column consists of a range of mineral particles, fine clays, silts and sands defined by the geology of the sediment source region (river, coast, sea bed). Coarse particles (e.g. quartz sands) sink to the bed rapidly whilst fine clay and

Table 3.2.10: Sinking and resuspension parameters and their default values in COHERENS.

symbol	unit	value	purpose
w_s^A	m/s	-0.002	inorganic sediment sinking rate
$w_{s_{det}}$	m/day	-5.0	organic detritus sinking rate
$w_{s_{min}}$	m/day	-0.5	minimum microplankton sinking rate
$w_{s_{max}}$	m/day	-5.0	maximum microplankton sinking rate
α_s	g m^{-2}s^{-1}	0.0025	parameter for sediment resuspension (equation (3.2.72))
n_s	—	3.0	exponential factor used for sediment resuspension (equation (3.2.72))

silt particles remain in the water column and influence turbidity for much longer periods. Inorganic sediment is included in the model as a population of fine particulate material, concentration A g m^{-3}, which sinks to the sea bed and is resuspended during periods of increased tidal velocity. These fine particles contribute to the attenuation of light in the euphotic layer (see Section III-2.3.1), but are otherwise inert and conservative. Coarse particles are not included in this version of the model.

The transport of inorganic sediment is again of the form (3.2.3)–(3.2.4). Since there are no sink and source terms for suspended inorganic sediment one has

$$\beta(A) = 0 \tag{3.2.70}$$

2.8 Seabed processes

Modelled suspended particulate material (SPM) consists of microplankton and detritus (oSPM), and inorganic sediment (iSPM). Particles are advected and diffused throughout the water column in an identical fashion to dissolved substances and heat (these processes are dealt with by the physical model). In addition SPM may be transported vertically by sinking and/or resuspension from a "fluff" layer immediately above the bed. Sinking is treated as an additional advective term in the physical model, using sinking rates passed from the biological or sediment model. Deposition to and resuspension from, the fluff layer are dealt with in the biology/SPM model. Note that simulations can be performed with COHERENS involving iSPM and without the biology.

2.8.1 Sinking

SPM sinking rates vary with particle size, shape and density. Inorganic sediment is modelled as one size class of fine particulate material. It sinks very slowly at a constant rate and so remains in suspension for very long periods of time. Organic detrital material also sinks at a constant rate. As these particles are typically larger than the inorganic sediment they sink faster and, depending on the rate of detrital remineralisation, may contribute a significant flux of organic material to the bed.

The microplankton compartment in the model includes phytoplankton which in reality employ various strategies to reduce sinking and prolong time spent in the euphotic zone. Some phytoplankton have morphologies with increased surface area and drag, others have the ability to vary their buoyancy and a few swim to maintain an optimal position in the water column. There is evidence to suggest that organisms which vary their sinking rate do so in response to variations in nutrient concentration, sinking rapidly when depleted of nutrient and more slowly when nutrients replete. Only this mechanism is adopted in the model. The simulated microplankton sinks faster or slower depending on its nitrogen quota:

$$w_s^B = w_s^N = (w_{s_{min}} - w_{s_{max}})\Big(\frac{Q - Q_{min}}{Q_{max} - Q_{min}}\Big) + w_{s_{max}} \tag{3.2.71}$$

where $w_{s_{min}}$ and $w_{s_{max}}$ are the minimum and maximum sinking rates (see Table 3.2.10) and Q_{min}, Q_{max} the minimum and maximum nitrogen quota (see Table 3.2.4). As the microplankton always sink (in this version of the model), its included phytoplankton may be considered to have the properties of diatoms rather than of flagellates (which can swim).

Uniform particle sinking rates are specified for inorganic sediment (w_s^A) and organic detritus ($w_{s_{det}}$). These values are passed to the physical model and sinking is implemented in the advection-diffusion equation as an additional advective term in the vertical transport equation.

2.8.2 The fluff layer

At the base of the water column and immediately above the sediment surface, sinking organic and inorganic material accumulates in an unconsolidated layer a few mm thick which is easily resuspended. During quiescent periods this detritus rich "fluff" layer has been observed floating on the sediment surface by underwater photography and divers. During periods of increased tidal currents and enhanced turbulent mixing this fluff layer is rapidly eroded and the material is partially or totally resuspended into the water column.

In the model the fluff layer is included as a compartment with negligible physical thickness. Material which sinks into the layer is then considered as an amount (per unit area) rather than a concentration.

2.8.3 Resuspension

Resuspension can partially or wholly transfer material from the fluff layer into the bottom water column grid point. This resuspended particulate material is then subject to advection, diffusion and sinking and is re-integrated with the existing water column populations of inorganic sediment, detritus and microplankton. Following quiescent periods resuspended detritus may be older and more refractory in nature than the water column detrital population. In this case the net detrital nitrogen quota (and associated remineralisation rate) becomes the mean of the two detrital sources. Under conditions of high bed shear stress, material sinking into the fluff layer may be immediately resuspended and returned to the water column.

Following Jones et al. (1995) the flux of particles being eroded or entrained from the fluff layer and resuspended in the water column is given by the erosion rate (amount m^{-2} s^{-1}):

$$E(\psi) = F(\psi)\alpha_s \left| \frac{\tau_{100}}{\tau_{b,ref}} \right|^{n_s} \tag{3.2.72}$$

where τ_{100} is the bed shear stress at a reference height of 1 m, α_s [5] and n_s are adjustable parameters adopted from Jones et al. (1995), and $\tau_{b,ref}$ is a nominal "reference" stress of value 0.1 N/m^2, included to cancel the dimensions of τ_{100} and hence to allow α_s to have constant units whatever the value is of n_s. In the case of a quadratic bottom friction law (equation (3.1.185)), the stress τ_{100} is obtained by

$$\tau_{100} = \rho_0 C_{100}(u_{100}^2 + v_{100}^2) \tag{3.2.73}$$

where (u_{100}, v_{100}) are the linearly interpolated values of the horizontal current at 1 m above the seabed. The bottom friction coefficient at the reference height is determined from (3.1.187) in the absence of wave-current interaction or from (3.1.281) if wave-current interaction is enabled, giving respectively

$$C_{100} = \Big(\kappa/\ln z_0\Big)^2 \quad \text{or} \quad C_{100} = \Big(\kappa/\ln(30/k_{bc})\Big)^2 \tag{3.2.74}$$

When the bottom stress is calculated using the linear law (3.1.186), τ_{100} is given by

$$\tau_{100} = \rho_0 k_{lin}(u_{100}^2 + v_{100}^2)^{1/2} \tag{3.2.75}$$

The factor $F(\psi)$ is a function of the amount of deposited matter of the different fractions and is determined as follows. Let $f_l(\psi)$ be the deposited amount per unit area of the quantity ψ, where "ψ" stands either for inorganic sediment A, microplankton carbon B or nitrogen N, detrital carbon C or nitrogen M. The fraction of material, available for resuspension, is then given as

$$\begin{array}{lll} \psi = A \Longrightarrow & F(A) = 1 & \\ \psi \neq A \Longrightarrow & F(\psi) = f_l(\psi)/f_l(A) & \text{if} \quad f_l(A) > 0 \\ & F(\psi) = 0 & \text{if} \quad f_l(A) = 0 \end{array} \tag{3.2.76}$$

An obvious constraint for the erosion rate $E(\psi)$ is that the amount of material which becomes resuspended during a time step Δt, cannot exceed the amount of available material in the fluff layer, i.e.

$$E(\psi) \leq f_l(\psi)/\Delta t \tag{3.2.77}$$

[5]The parameter α_s for sediment should not be confused with the microplankton photosynthetic parameter α.

2.8.4 Seabed exchange

At the sediment/water-column interface particulate material is transferred from the fluff layer into the sediment and dissolved nutrients diffuse in and out of the water column. Diagenetic processes in the sediment are not modelled and as the diffusion of nutrients across the sediment-water interface is independent of material incorporated into the sediment from the fluff layer, the sediment and water column are not truly coupled. The model includes no oxygen demand from the consolidated sediment although simulated mineralisation in the fluff layer may make a large demand on oxygen concentration at the deepest water column grid point.

2.8.5 Fluff layer processes

As a result of deposition, at a rate given by $w_s\psi$, from the deepest water column grid point, sinking organic and inorganic material accumulates in an unconsolidated and easily resuspended "fluff" layer immediately above the sediment surface. Within the fluff layer detrital remineralisation continues as in the water column, and the resulting fluxes of dissolved substances are added to concentrations in the bottom water column layer. A fraction of the microplankton in the fluff layer are deemed to die due to lack of light, predation, or physical damage from collisions with other particles and the bed, and this dead material is then added to the fluff layer detrital pool. This fraction is taken in the model as 10% per day.

Particulate material in the fluff layer does not sink further but a fraction of the organic material is considered to be actively drawn into the consolidated sediment by the feeding of meio- and macro-benthic organisms. Once in the sediment this material cannot be resuspended and the carbon and nitrogen is lost from the model system. This loss of nitrogen has similar implications as the loss of nitrogen by zooplankton assimilation (Section III-2.4.4) but is partially compensated for by a benthic nutrient flux (Section III-2.9.1). A 10% daily loss of microplankton and detritus is assumed in the model[6]. As in the case of the microplankton death rate, this value is notional, but probably of the right order of magnitude. An improvement of this parameterisation is needed.

2.9 Boundary conditions

2.9.1 Bottom fluxes

The seabed exchange of microplankton and detritus is already taken into account in the fluff layer part of the model (see Section III-2.8). No bottom flux of oxygen is assumed in the model. The flux of dissolved nutrients into or out of the sediment is regulated as a relaxation function. Current concentrations of dissolved substances in the bottom

[6]Note that the accumulation of deposited nitrogen "lost" from the model must be recorded and accounted for in the summation of total nitrogen when nitrogen conservation is checked over time (see Chapter II-5).

Table 3.2.11: Parameters for the bottom and surface fluxes with their default values in COHERENS.

symbol	unit	value	purpose
F_{br}	day^{-1}	0.02	relaxation rate for benthic nutrient fluxes
${}^{NO}S_b^0$	mmol N m^{-3}	5.0	assumed nitrate concentration in the sediment taken as equal to the winter concentration in the water column[a]
${}^{NH}S_b^0$	mmol N m^{-3}	1.0	assumed ammonium concentration in the sediment taken as equal to the winter concentration in the water column[a]
k_w	s/m	5×10^{-7}	factor relating the surface exchange coefficient for oxygen to the wind speed
a_{sT}	m^3 (mmol O)$^{-1}$	2.74×10^{-3}	coefficient in the formulation for the air oxygen concentration
b_{sT}	m^3 (mmol O)$^{-1}$ (^{0}C)$^{-1}$	7.8×10^{-5}	temperature coefficient in the formulation for the air oxygen concentration

[a]This value is specific to the North Sea Project station CS (Tett and Walne, 1995) as used in the test case **csbio** (see Chapter II-5).

water column layer are allowed to relax towards their initial winter concentration over a period of 50 days. This ensures that the model system does not become cumulatively depleted in nutrients but is steadily replenished, particularly during periods of deep mixing in winter. It has the advantage of correctly resetting winter concentrations of nutrients, but is unrealistic in that it does not require the nitrogen cycle to be closed. (This will be remedied in later versions of the model.)

The benthic fluxes are then parameterised in the form given by the general expression (3.1.191):

$$\frac{\lambda_T}{J}\frac{\partial}{\partial \tilde{x}_3}({}^{NO}S) = F_{br}H({}^{NO}S_b - {}^{NO}S_b^0) \tag{3.2.78}$$

$$\frac{\lambda_T}{J}\frac{\partial}{\partial \tilde{x}_3}({}^{NH}S) = F_{br}H({}^{NH}S_b - {}^{NH}S_b^0) \tag{3.2.79}$$

where ${}^{NO}S_b$, ${}^{NH}S_b$ are the values of nitrate and ammonium at the lowest grid cell, ${}^{NO}S_b^0$, ${}^{NH}S_b^0$ their assumed concentrations in the sediment (see Table 3.2.11), $F_{br}H$ the transfer velocity and H the total water depth.

2.9.2 Surface fluxes

All substances in the biological-sediment model have a no flux surface boundary condition, except for oxygen. Oxygen is allowed to diffuse into or out of the top layer of the water column as:

$$\frac{\lambda_T}{J}\frac{\partial O}{\partial \tilde{x}_3} = C_O(O_{air}(T_s) - O_s) \tag{3.2.80}$$

where O_s is the surface value of the modelled oxygen concentration. Following Liss (1988) the exchange coefficient C_O depends on the wind speed $|U_{10}|$ (see Section III-1.6.1):

$$C_O = k_w|U_{10}|^2 \tag{3.2.81}$$

The actual air concentration $O_{air}(T_s)$ modified by the (temperature dependent) ratio of the equilibrium air and water oxygen concentration is related to temperature as

$$O_{air}(T_s) = (a_{sT} + b_{sT}T_s)^{-1} \tag{3.2.82}$$

where T_s is the sea surface temperature. The relation (3.2.82) is derived on the basis of oxygen solubility values in Carpenter (1966) and a typical chlorinity of 19 ppt for the southern North Sea. The (default) values used for the parameters k_w, a_{sT}, b_{sT} in the program, are given in Table 3.2.11.

2.9.3 Lateral boundary conditions

Either a "scalar" or a "zero normal gradient" condition can be selected by the user at open sea or river boundaries. Exceptions are oxygen O and zooplankton nitrogen Z_N for which a zero gradient condition is always applied. For more details see Section III-1.6.3b. All advective and diffusive fluxes are obviously set to zero at land boundaries.

2.10 Summary of equations and model switches

Apart from physical transport, the processes discussed in the previous sections of this chapter can be summarised as water column processes, fluff layer processes and fluxes at the surface or bottom boundaries. A summary is given below.

2.10.1 Water column processes

- microplankton carbon

  $\beta(B) = (\mu - G)B$: the growth rate μ is determined from (3.2.22), (3.2.28), (3.2.39); sinking velocity given by (3.2.71)

- microplankton nitrogen

$\beta(N) = ({}^{NO}u + {}^{NH}u)B - GN$: the nitrate and ammonium uptake rates ${}^{NO}u$ and ${}^{NH}u$ are given by (3.2.43) and (3.2.55);
sinking velocity given by (3.2.71)

- detrital carbon

  $\beta(C) = (1-\gamma)GB - {}^{C}rC$: the carbon remineralisation rate ${}^{C}r$ is determined by (3.2.62);
  uniform sinking velocity $w_{s_{det}}$

- detrital nitrogen

  $\beta(M) = (1-\gamma)GN - {}^{M}rM$: the nitrogen remineralisation rate ${}^{M}r$ is given by (3.2.63);
  uniform sinking velocity $w_{s_{det}}$

- nitrate

  $\beta({}^{NO}S) = -{}^{NO}uB + {}^{NH}r.{}^{NH}S$: the nitrate uptake and the nitrification rates ${}^{NO}u$ and ${}^{NH}r$ are given by (3.2.43) and (3.2.69);
  zero sinking velocity

- ammonium

  $\beta({}^{NH}S) = -{}^{NH}uB - {}^{NH}r.{}^{NH}S + e\gamma GN + {}^{M}rM$: the ammonium uptake, nitrification and remineralisation rates ${}^{NH}u$, ${}^{NH}r$ and ${}^{M}r$ are determined by (3.2.55), (3.2.69) and (3.2.63);
  zero sinking velocity

- oxygen

  $\beta(O) = ({}^{O}q^{B}\mu + {}^{O}q^{NO}.{}^{NO}u - {}^{O}q^{C}e\gamma G)B - {}^{O}q^{NH}.{}^{NH}r.{}^{NH}S - {}^{O}q^{C}.{}^{C}rC$: the nitrate uptake, nitrification and carbon remineralisation rates ${}^{NO}u$, ${}^{NH}r$ and ${}^{C}r$ are determined by (3.2.43), (3.2.69), (3.2.62); the microplankton growth rate μ by (3.2.22), (3.2.28), (3.2.39);
  zero sinking velocity

- zooplankton

  $\beta(Z_N) = \gamma(1-e)GN$;
  zero sinking velocity

- inorganic sediment

  $\beta(A) = 0$;
  uniform sinking velocity w_s^A

The grazing pressure G is supplied externally to the program at monthly intervals.

2.10.2 Fluff layer processes

This involves microplankton carbon B and nitrogen N, detrital carbon C and nitrogen M, inorganic sediment A.

- The fluff layer and water column exchange organic and inorganic material by deposition and resuspension. The exchange process is conservative. The resuspension rates are calculated using (3.2.72)–(3.2.76).
- Microplankton carbon and nitrogen are converted into detrital carbon and nitrogen at a rate of 10% per day.
- Microplankton and detrital carbon and nitrogen are lost to the consolidated sediment at a rate of 10% per day.
- In analogy with the water column, remineralisation of detritus takes place in the fluff layer.

2.10.3 Surface and bottom fluxes

- The surface oxygen flux is given by (3.2.80)–(3.2.82). All other surface fluxes are set to zero.
- The only non-zero bottom fluxes are those for nitrate and ammonium given by (3.2.78)–(3.2.79).

2.10.4 Model switches

A number of program switches are related to the biological and sediment part of the program. They are listed in Table 3.2.12. A full description of all program switches is given in Section IV-2.2.1.

Table 3.2.12: Program switches for the biological and sediment modules.

IOPTB	The biological module is disabled/enabled if this switch is set to 0/1.
IOPTS	The sediment module is disabled/enabled if this switch is set to 0/1.
IADVS	The evaluation of the horizontal and vertical (except sinking) advective terms in all transport equations is disabled/enabled when this switch is set to 0 or has a positive value.
IODIF	The evaluation of the horizontal diffusion terms in the transport equations is disabled/enabled when this switch is 0 or takes a positive value.
IFLUFF	The fluff layer part and the vertical sinking terms in the transport equations are disabled/enabled when this switch is set to 0/1.
IADVWB	If IADVS $= 0$, the vertical sinking terms are disabled/enabled when this switch is 0 or takes a positive value.

Chapter 3

Contaminant Transport Models

This chapter describes the Lagrangian particle tracking procedure used in the transport module SEDLAG and the contaminant module MASS which uses Eulerian advection-dispersion equations to represent the evolution of contaminants. Only a general outline is given here. A detailed description of the numerical methods used in the Lagrangian transport module is presented in Section III-4.8.

3.1 Introduction

The use of Lagrangian particle tracking techniques in the marine environment, especially those in which tidal mixing is predominant, offers advantages over more traditional solution methods of the advection-dispersion equation (e.g. finite differences). Computational effort is expended in areas where the particles are concentrated, allowing sources and sinks to be easily represented, and sharp fronts and gradients to be resolved. This compares with the finite difference approach in which all areas, regardless of concentration, are treated equally on a resolution defined by the grid structure, which, for sharp fronts, is usually of a coarser resolution than needed, introducing numerical dispersion into the solution and a smoothing of frontal structure. The particle tracking algorithm is well suited to use on parallel computers and properties of the contaminant represented are more easily determined, e.g. residence times.

The technique has been used extensively in many marine applications e.g. to describe chaotic tidal stirring (Ridderinkhof and Zimmerman, 1992), contaminant dispersal (Allen, 1982), oil spill modelling (Proctor et al., 1994), to estimate shear diffusion from dye dispersion (Elliott, 1986), fish larvae tracking (Bartsch and Coombs, 1996), sediment transport (Puls and Sündermann, 1993). Also see Van Damm (1994) for many others.

In the absence of diffusion the accuracy of the time integration of the velocity field is critical to the solution. However, with diffusion, first or second order accuracy of the time integration is sufficient as errors in discretisation will be masked by the diffusion process. In the particle tracking technique, the diffusion process is modelled by the Monte-Carlo, or random walk, process where particles are subjected to a random motion with variance

$\sigma^2 = 2K\Delta t$, where K is the local diffusivity and Δt is the model time step, which is added to the advective displacement.

As with other stochastic methods, the accuracy of solution varies as the square root of the computational effort. This can be contrasted with deterministic methods where the accuracy generally varies in proportion to the computational effort. Hunter (1987) shows that, when comparing particle tracking methods to second order finite difference methods, the ratio of the proportional errors is

$$\frac{E_p}{E_{fd}} = f^{(n/2+2)} N^{(n/2)} \tag{3.3.1}$$

where E_p is the error in particle tracking, E_{fd} the error in the finite difference solution, f the fraction of the computational domain covered by the contaminant, N the number of mesh points and n the number of dimensions (1, 2, or 3). The conclusion drawn is that for a given computational cost the particle tracking method is more accurate where the contaminant covers less than the whole domain ($f \ll 1$), and for cases of high dimension and small numbers of particles. Spaulding (1986) postulated that the particle tracking method will become the main method of solution of the advection-dispersion equation for oil spill models.

The random distribution usually chosen is the “two-step” distribution, where values lie within the range $\pm(2K\Delta t)$, or the uniform “top-hat” distribution $\pm(6K\Delta t)$ (as here), rather than a Gaussian distribution with the same mean and variance. This choice is made on the grounds of computational efficiency and relies heavily on the Central Limit Theorem to achieve Gaussianity. Hunter et al. (1993) question the validity of this approach and argue that in terms of the temporal resolution of particle concentrations it is, in fact, more computationally efficient to use a Gaussian distribution.

3.2 Lagrangian transport

The transport of particles is determined by horizontal and vertical advection and turbulent diffusion. The basic equation, rewritten from (3.1.70)[1], is given by

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(JC) + \frac{1}{J}\frac{\partial}{\partial x_1}(JuC) + \frac{1}{J}\frac{\partial}{\partial x_2}(JvC) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}C) \\ = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial C}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\lambda_H\frac{\partial C}{\partial x_1}\Big) + \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\lambda_H\frac{\partial C}{\partial x_2}\Big)\end{aligned} \tag{3.3.2}$$

The concentration $C(x_1, x_2, \tilde{x}_3, t)$ of suspended matter is represented by a number of particles each having a certain amount of load m_i and with initial positions specified by the user. Each particle has a specific label, assigned by the user, which allows to trace the

[1]The equation is written here in Cartesian form only. The extension to spherical coordinates, the “transformed” vertical velocity $J\tilde{w}$ and the meaning of the Jacobian J (related to the vertical coordinate transformation) are discussed in Section III-1.1.

evolution of individual particles or to make a distinction between different contaminant distributions.

Equation (3.3.2) is solved in two steps. Firstly, the concentration is updated at a new time step without diffusion terms. The solution in Lagrangian notation is given by

$$C(x_1, x_2, \tilde{x}_3, t + \Delta t) = C(x_1 - \int_t^{t+\Delta t} u dt, x_2 - \int_t^{t+\Delta t} v dt, \tilde{x}_3 - \int_t^{t+\Delta t} J\tilde{w} dt, t) \tag{3.3.3}$$

This is determined in the model by adding the distance travelled by each particle during the interval Δt (given by the three integrals in (3.3.3)) to the old particle positions which gives the particle positions and hence the updated contaminant distributions for advection[2].

Diffusive transport is determined similarly with (u, v, $J\tilde{w}$) replaced by the fluctuating current velocities (u', v', w'). These fluctuations have to be parameterised in the model because their scale is far below the grid spacings in the model and the scales of the advective velocities. To get these terms in Lagrangian notation the Monte-Carlo method is applied. From a statistical velocity distribution with bandwith (u'_{max}, v'_{max}, w'_{max}) a random fluctuating velocity

$$(u', v', w') \in (-(u'_{max}, v'_{max}, w'_{max}), (u'_{max}, v'_{max}, w'_{max})) \tag{3.3.4}$$

is chosen. Hence, turbulence is described by the probability that a particle is shifted a certain distance within a time step (Maier-Reimer and Sündermann, 1982). Following Maier-Reimer (1980) the maximum velocities are related to the horizontal and vertical eddy diffusion coefficients by

$$(u'_{max}, v'_{max}, w'_{max}) = \sqrt{\frac{6(\lambda_H^C, \lambda_H^C, \lambda_T)}{\Delta t}} \tag{3.3.5}$$

where λ_H^C is the uniform horizontal diffusion coefficient. The turbulent velocities are then determined for each particle with the Monte-Carlo method which consists in multiplying each maximum current component by a random generated number between -1 and 1. After updating the particle positions the contaminant distribution is obtained by adding the loads of the particles, resident in each grid cell, and dividing this sum by the volume of the cell.

The following boundary conditions are imposed:

- Particles cannot be advected across the bottom and surface, i.e.

$$J\tilde{w}C = 0 \quad \text{at} \quad \tilde{x}_3 = 0, L \tag{3.3.6}$$

- Particles advected across an open sea or land boundary are lost to the system. This is performed in the program by setting the particle's label to zero. New particle

[2]With the notations used in Chapter III-4, Δt equals the time step Δt_{3D} for the 3-D mode calculations.

input can be supplied by the user. The form depends on the type of open boundary. In the case of an open sea boundary an ambient external concentration in ton/m^3 of suspended matter can be specified by the user. The program calculates the amount of matter entering the computational domain during inflow and distributes this mass into a series of new particles. A similar condition applies at river boundaries except that a river discharge in ton/s can be provided. Both the external concentration as the river discharge can be taken as time-independent or supplied at a selected time interval. For more details see Section III-4.8.5.

- The sea bottom and surface, open sea and land boundaries are considered as reflective boundaries for turbulent transport which corresponds to the zero flux conditions

$$\frac{\lambda_T}{J}\frac{\partial C}{\partial \tilde{x}_3} = 0 \quad \text{at} \quad \tilde{x}_3 = 0, L \tag{3.3.7}$$

$$\lambda_H \frac{\partial C}{\partial x_1} = 0 \quad \text{or} \quad \lambda_H \frac{\partial C}{\partial x_2} = 0 \tag{3.3.8}$$

at open sea or land boundaries.

Finally, it is noted that the method remains essentially the same in the case of a spherical grid except that (3.3.2) and (3.3.7)–(3.3.8) are replaced by their equivalent spherical forms (which has no influence on the solution method) and the horizontal particle positions are now given in spherical coordinates (λ, ϕ).

3.3 Eulerian transport

The program solves a series of transport equations for a given number of contaminant distributions C_i ($i = 1, N_c$) of the form given by (3.3.2):

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(JC_i) + \frac{1}{J}\frac{\partial}{\partial x_1}(JuC_i) + \frac{1}{J}\frac{\partial}{\partial x_2}(JvC_i) + \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}C_i)\\ = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial C_i}{\partial \tilde{x}_3}\Big) + \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\lambda_H\frac{\partial C_i}{\partial x_1}\Big) + \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\lambda_H\frac{\partial C_i}{\partial x_2}\Big)\end{aligned} \tag{3.3.9}$$

Initial distributions are specified by the user. The N_c equations (3.3.9) are solved with the following boundary conditions:

- A zero flux condition is applied at the sea bottom and surface, i.e.

$$\frac{\lambda_T}{J}\frac{\partial C_i}{\partial \tilde{x}_3} = 0 \quad \text{at} \quad \tilde{x}_3 = 0, L \tag{3.3.10}$$

- Advective and diffusive fluxes at open boundaries are determined using one of the methods described in Section III-1.6.3b (specification of an external value or zero normal gradient condition).

Table 3.3.1: Program switches for the contaminant modules.

IOPTP	The Lagrangian particle module is disabled/enabled if this switch is 0 or has a positive value. Horizontal and vertical diffusion are enabled only when the switch takes a value 2.
IOPTC	The Eulerian contaminant transport module is disabled/enabled if this switch is 0 or has a positive value. Horizontal and vertical diffusion are enabled only when the switch takes a value of 2.
IADVS	The evaluation of the horizontal and vertical advective terms in the Eulerian contaminant transport equations is disabled/enabled when this switch is set to 0 or takes a positive value.
IODIF	The evaluation of the horizontal diffusion terms in the Eulerian contaminant transport equations is disabled/enabled when this switch is set to 0 or takes a positive value. Note that horizontal diffusion is only enabled if IOPTC is set to 2 as well.
IOPTK	The evaluation of the vertical diffusion term in the Eulerian contaminant transport equations can be disabled if this switch is set to 0. In that case, other model parameters must be set to 0 (see Sections III-1.2 and IV-2.2).

- No fluxes are allowed across land boundaries

$$\lambda_H \frac{\partial C_i}{\partial x_1} = 0\,,\; uC_i = 0 \quad \text{or} \quad \lambda_H \frac{\partial C_i}{\partial x_2} = 0\,,\; vC_i = 0 \tag{3.3.11}$$

It is obvious that in the case of a spherical grid equations (3.3.9) and (3.3.11) are replaced by their equivalent spherical forms.

3.4 Model switches

A number of switches, listed in Table 3.3.1, are related to the contaminant models. A full description of all program switches is given in Section IV-2.2.1.

Chapter 4

Numerical Methods

4.1 Introduction

The numerical methods described in this chapter are based upon previous work described in Blumberg and Mellor (1987), Deleersnijder (1992), Beckers (1992) and Ruddick (1995). The Lagrangian particle method has been implemented following the original work by Mirbach (1997).

Conservative finite differences (equivalent to a finite volume technique for the Cartesian mesh) are used to discretise the mathematical model in space. The grid chosen for horizontal discretisation is the well known Arakawa "C" grid (Mesinger and Arakawa, 1976) which staggers the currents and pressure/elevation nodes to give a good representation of the crucial gravity waves and provides simple representations of open and coastal boundaries. The commonly used "σ" vertical coordinate transformation, whereby varying surface and bottom boundaries are transformed into constant surfaces, is also used. This provides for accurate representation of surface and bottom boundary processes. It also results in an equal number of elements in each vertical water column.

The momentum equations are solved with the mode-splitting technique as in the model of Blumberg and Mellor (1987). The method consists in solving the depth-integrated momentum and continuity equations for the "external" or barotropic mode with a small time step to satisfy the stringent CFL stability criterium for surface gravity waves, and the 3-D momentum and scalar transport equations for the "internal" or "baroclinic" mode with a larger time step. A "predictor" and a "corrector" step are applied for the horizontal momentum equations to satisfy the basic requirement that the depth-integrated currents obtained from the the 2-D and 3-D mode equations, have identical values.

Much effort has been made to implement suitable schemes for the advection of momentum and scalars. Although a variety of schemes are available from the literature (e.g. Hirsch, 1990; James, 1996), the basic choice in the program is between the upwind and the TVD (Total Variation Diminishing) scheme to reduce the programming and computational overhead. The latter scheme is implemented with the symmetrical operator splitting method for time integration and can be considered as a useful tool for the simulation of frontal structures and areas with strong current gradients. The upwind scheme, on the other hand, is only first order accurate and therefore more diffusive, and should be used if CPU time is considered of more importance than accuracy.

The following additional issues are noted:

- Scalar quantities are advected with a "filtered" velocity (u^F,v^F) derived from the "predicted" currents and the depth-integrated current averaged over the internal time step (Deleersnijder, 1993).

- Sink terms are discretised explicitly in time for cell-centered scalars to conserve the processes in the biological module whereas a quasi-implicit formulation is implemented for turbulence transport to ensure positivity (Patankar, 1980).

This chapter is organised as follows:

- The model grid, the grid indexing system and notational conventions are described in Section III-4.2.

- The solution of the momentum equations and the mode splitting technique are presented in Section III-4.3.

- The scalar transport equations are discussed in Section III-4.4.

- Numerical aspects of the turbulence module are given in Section III-4.5.

- Extensions for spherical coordinates are given in Section III-4.6.

- Numerical methods for 1-D applications are briefly discussed in Section III-4.7.

- The Lagrangian particle module is described in Section III-4.8.

- The order of update for the transported quantities in the program is summarised in Section III-4.9.

- Model switches to select a particular numerical scheme are listed in Section III-4.10.

4.2 Model grid and discretisations

Sections III-4.2–4.5 only deal with the case of a Cartesian grid. Spherical coordinates are discussed in Section III-4.6.

4.2.1 Location of variables on the model grid

The model equations are discretised horizontally on a C-grid and vertically using the σ-coordinate transformation described in Section III-1.1.1b.

The location of the state variables on the grid is shown in Figure 3.4.1, whereby a 3-D quantity is situated at the boundaries or at the centre of a 3-D cell and a 2-D quantity at the sides or at the centre of a rectangle in the horizontal plane. Each 3-D cell has six lateral faces:

- The West and East boundary faces are perpendicular to the X-direction and are located respectively in the negative or positive X-direction with respect to the centre of the cell.
- The South and North boundary faces are perpendicular to the Y-direction and are located respectively in the negative or positive Y-direction with respect to the centre of the cell.
- The bottom and top faces are perpendicular to the Z-direction and are located respectively below and above the centre.

The four sides of a 2-D rectangle are denoted in a similar way by the western, eastern, southern and northern sides.

The variables are then located on one of the four following grid locations:

- centres: the centre of the grid cell
  - 2-D: all scalars (mean and total water depth h and H, sea surface elevation ζ, magnitude of the surface and bottom stress τ_s and τ_b, ...)
  - 3-D: all scalar quantities not related to turbulence (temperature T, salinity S, biological state variables, inorganic suspended matter A, contaminant distributions, ...)
- U-nodes: the centres of the western and eastern boundary faces (3-D) or sides (2-D)
  - 2-D: the x_1-component of 2-D vectors (depth integrated current $\overline{U}$, bottom stress τ_{b1})
  - 3-D: the x_1-component of 3-D vectors (u-current)
- V-nodes: the centres of the southern and northern boundary faces (3-D) or sides (2-D)
  - 2-D: the x_2-component of 2-D vectors (depth integrated current $\overline{V}$, bottom stress τ_{b2})
  - 3-D: the x_2-component of 3-D vectors (v-current)

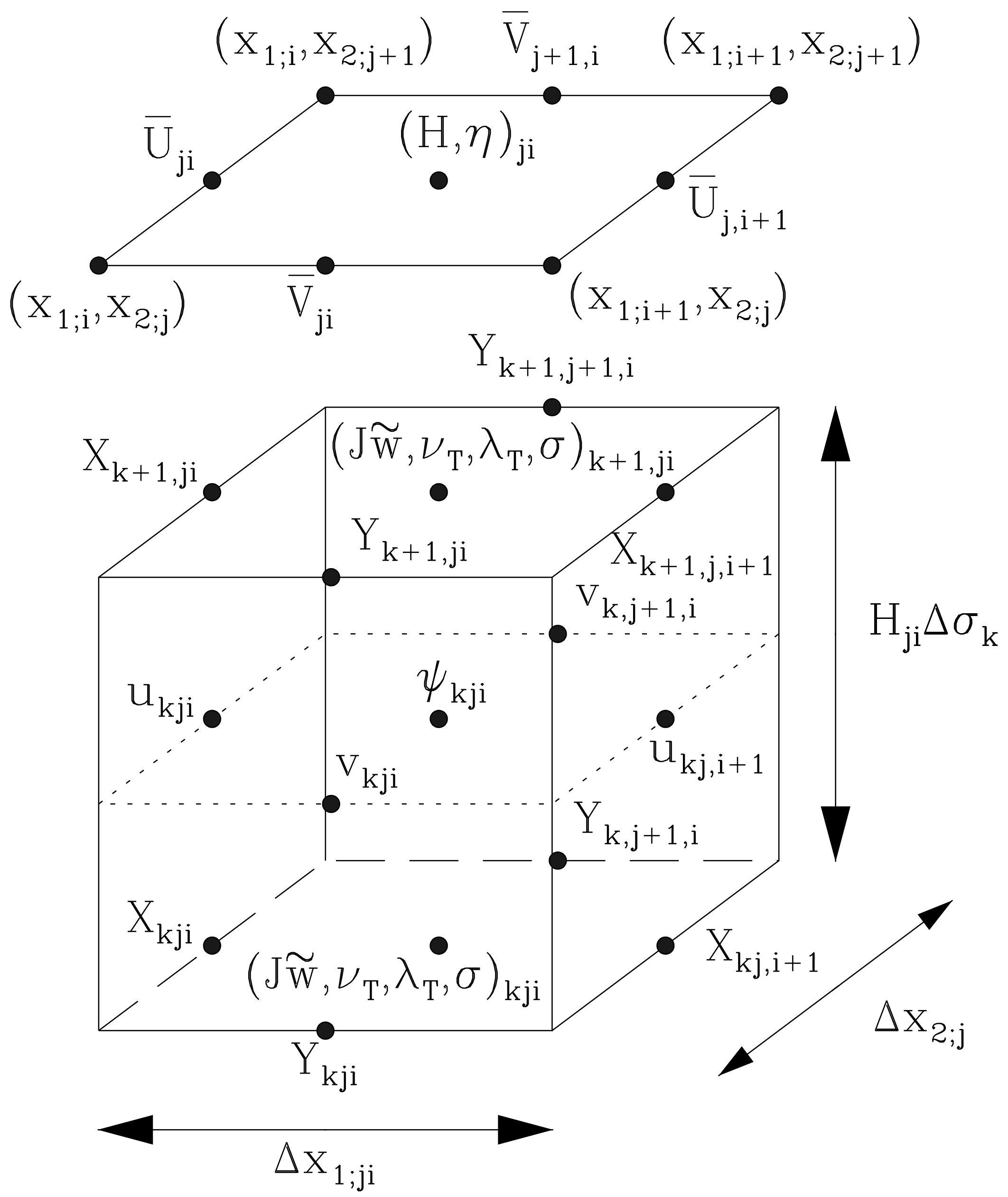


Figure 3.4.1: Location of the variables on the model grid.

- W-nodes (3-D only): the centres of the bottom and top boundary faces which are used for the location of the (transformed) vertical current $J\tilde{w}$ and all quantities related to turbulence (ν_T,λ_T,k,l,ε, squared buoyancy and shear frequencies N^2 and M^2, Richardson number Ri).

- X-nodes: the points located at the centres of the sides formed by the intersection of the West/East and bottom/top boundary faces of the grid cell. The points have the same horizontal position as the U-nodes and the same vertical position as the W-nodes. The X-nodes are only used in the program for the location of some of the advective and diffusion fluxes.

- Y-nodes: the points located at the centres of the sides formed by the intersection of the South/North and bottom/top boundary faces of the grid cell. The points have the same horizontal position as the V-nodes and the same vertical position as the W-nodes. The Y-nodes are only used in the program for the location of some of the advective and diffusion fluxes.

Only exceptions to the above rules are the components of the surface stress (τ_{s1},τ_{s2}) which are evaluated at the cell centres for internal consistency in the surface fluxes module, and the physical vertical current w which is evaluated at the cell centre using the discretised form of (3.1.47).

4.2.2 Grid indexing system

The position of a variable on the model grid is represented by the two indices (j,i) for 2-D and the three indices (k,j,i) for 3-D variables where

- k represents the grid number in the (vertical) σ-direction

- j represents the grid number in the x_2-direction

- i represents the grid number in the x_1-direction

As shown in Figure 3.4.1, the indices refer to the position of the variable at its "natural" boundary (cell centre, U-, V-, W-, X-, Y-node). The number of grid cells in the σ-, x_2- and x_1-direction are denoted by respectively N_z, N_y and N_x. The indices have a lower limit of 1 while

- The upper limit of k is N_z for variables located at the centres, U- or V-nodes and $N_z + 1$ for variables at the W-, X- or Y-nodes.

- The upper limit of j is N_y for variables located at the centres, U-, W- or X-nodes and $N_y + 1$ for variables at the V- or Y-nodes.

- The upper limit of i is N_x for variables located at the centres, V-, W- or Y-nodes and $N_x + 1$ for variables at the U- or X-nodes.

Note that the i-index increases in the positive X-direction, the j-index in the positive Y-direction and the k-index in the upward direction. To simplify the notations, the indices k, j, i are omitted if no confusion is possible. This means e.g. that $Q_{k,j+1,i,}$ (3-D quantity) can be written as Q_{j+1} or that $\overline{Q}_{i-1}$ (2-D quantity) is the same as $\overline{Q}_{j,i-1}$.

If a quantity needs to be evaluated at a point, different from its natural position, its value is determined by taking the average over the neighbouring points. This is indicated by one of the superscripts ^{c}, ^{u}, ^{v}, ^{w}, ^{X}, ^{Y} referring to the point at which the quantity is interpolated. For example, the Coriolis terms in the momentum equations require a 4-point interpolation of the u and v velocities:

$$\begin{aligned} u^v_{kji} &= \frac{1}{4}(u_{kji} + u_{k,j-1,i} + u_{kj,i+1} + u_{k,j-1,i+1}) \\ v^u_{kji} &= \frac{1}{4}(v_{kji} + v_{kj,i-1} + v_{k,j+1,i} + v_{k,j+1,i-1}) \end{aligned} \tag{3.4.1}$$

The convention is further illustrated by the following example of a centered quantity Q evaluated at respectively the U-, V-, W-, X- and Y-node with the same index values:

$$\begin{aligned} Q^u_{kji} &= \frac{1}{2}(Q_{kj,i-1} + Q_{kji}) \\ Q^v_{kji} &= \frac{1}{2}(Q_{k,j-1,i} + Q_{kji}) \\ Q^w_{kji} &= \frac{1}{2}(Q_{k-1,ji} + Q_{kji}) \\ Q^X_{kji} &= \frac{1}{4}(Q_{k-1,ji} + Q_{kji} + Q_{k-1,j,i-1} + Q_{kj,i-1}) \\ Q^Y_{kji} &= \frac{1}{4}(Q_{k-1,ji} + Q_{kji} + Q_{k-1,j-1,i} + Q_{k,j-1,i}) \end{aligned} \tag{3.4.2}$$

A double index notation of the form $i_1 : i_2$ or $j_1 : j_2$ is sometimes introduced in expressions related to open boundary conditions, where the first index i_1 (j_1) is used at western (southern) boundaries and the second index i_2 (j_2) at eastern (northern) boundaries. This is illustrated with the following example expressions

$$u_{kj,i+1:i-1}\ ,\ Q_{k,j:j-1,i}$$

4.2.3 Space discretisation

As indicated in Figure 3.4.1, the grid is defined in the Cartesian reference frame by specifying the following three 1-D arrays at the cell corners:

- the x_1-coordinates $x_{1;i}$ of the corners ($N_x + 1$ values)
- the x_2-coordinates $x_{2;j}$ of the corners ($N_y + 1$ values)

- the σ-coordinates σ_k of the W-nodes ($N_z + 1$ values)

They are represented in the program by the arrays GX0, GY0, GZ0 further discussed in Section IV-2.3. Irregular spacing is allowed which means that the grid spacings Δx_1, Δx_2 may vary in respectively the x_1- or x_2-direction[1] while Δx_3 varies in all three directions. Note however that $\sigma_1 = 0$ (bottom) and $\sigma_{N_z+1} = 1$ (surface).

The x_3- and $\tilde{x}_3$-coordinates are related by the transformation (3.1.19)–(3.1.20). The length scale L is chosen such that $\Delta\tilde{x}_3 = 1$ so that $L = N_Z$ since $\tilde{x}_3$ varies between 0 and L. Hence, it follows from (3.1.21) and (3.1.20) that

$$J = \frac{\partial x_3}{\partial \tilde{x}_3} = \Delta x_3 = H\Delta\sigma \tag{3.4.3}$$

Although J and Δx_3 are identical, the two notations are used throughout this chapter. To comply with the notations of Chapter III-1, Δx_3 is only used to represent vertical derivatives, i.e. $J\Delta\tilde{x}_3 = \Delta x_3$.

Spatial differences in the x_1-, x_2- or vertical direction are represented respectively by the operators Δ_x, Δ_y, Δ_z. A superscript ^{c}, ^{u}, ^{v}, ^{w}, ^{X} or ^{Y} is added if the difference operator is applied to a quantity at a point, not located at its natural boundary. This is illustrated with the following examples (where Q represents a centered quantity in the third example):

$$\begin{aligned}
\Delta^c_x u_{kji} &= u_{kj,i+1} - u_{kji} \\
\Delta^v_y u^c_{kji} &= \frac{1}{2}(u_{kji} + u_{kj,i+1} - u_{k,j-1,i} - u_{k,j-1,i+1}) \\
\Delta_z Q_{kji} &= Q_{kji} - Q_{k-1,ji} \\
\Delta^c_y \overline{V}_{ji} &= \overline{V}_{j+1,i} - \overline{V}_{ji}
\end{aligned} \tag{3.4.4}$$

Grid spacings (including the Jacobian J) are evaluated at the cell centre. Conforming the previous rules interpolated values at other grid locations are indicated by a superscript, e.g.

$$\begin{aligned}
\Delta^u x_{1;ji} &= \frac{1}{2}(\Delta x_{1;j,i-1} + \Delta x_{1;ji}) \\
\Delta^w x_{3;kji} &= \frac{1}{2}(\Delta x_{3;k-1,ji} + \Delta x_{3;kji})
\end{aligned} \tag{3.4.5}$$

An overview of all subscript and superscript notations, used in this chapter, is given in Table 3.4.1.

4.2.4 Time discretisation

The time discretisation of the model equations is summarised below. A detailed description is given in the sections below.

[1] In the case of a spherical grid Δx_1 varies also in the Y-direction. A two-index notation, i.e. $\Delta x_{1;ji}$ will be used throughout to avoid any ambiguity.

Table 3.4.1: Subscript and superscript notation used in the numerical discretisations.

Type	Purpose
subscripts	
k	vertical index of the variable on the model grid between 1 and either N_z (centre, U-, V-node) or N_z+1 (W-, X-, Y-node)
j	Y-index of the variable on the model grid between 1 and either N_y (centre, U-, W-, X-node) or N_y+1 (V-, Y-node)
i	X-index of the variable on the model grid between 1 and either N_x (centre, V-, W-, Y-node) or N_x+1 (U-, X-node)
$i_1 : i_2$	expression used in the spatial discretisation of open boundary conditions, whereby the first index is taken at western or southern boundaries and the second index at eastern or northern boundaries
superscripts	
c	quantity evaluated or interpolated at the cell centre
u	quantity evaluated or interpolated at the U-node
v	quantity evaluated or interpolated at the V-node
w	quantity evaluated or interpolated at the W-node
X	quantity evaluated or interpolated at the X-node
Y	quantity evaluated or interpolated at the Y-node
n	quantity evaluated at the old baroclinic time t^n
$n+1$	quantity evaluated at the new baroclinic time t^{n+1}
m	quantity evaluated at the old barotropic time t^m
$m+1$	quantity evaluated at the new barotropic time t^{m+1}
p	"predicted" value
$n_1 : n_2$	expression used in the temporal discretisation of the Coriolis terms in the 2-D and 3-D momentum equations whereby the first index is taken at odd and the second at even barotropic or baroclinic time steps

- A mode-splitting technique is used (Blumberg and Mellor, 1987) with separate time steps for the 2-D "external" barotropic equations (Δt_{2D}) and the "internal" baroclinic equations (Δt_{3D}). The 2-D time step Δt_{2D} has to be small enough to satisfy the Courant-Friedrichs-Lewy (CFL) criterion. The 3-D time step is a multiple, M_t, of Δt_{2D} (typically of the order of 10–20) and the model is integrated forward in time for N_t baroclinic time steps (equal to $N_t M_t = M_{tot}$ barotropic time steps). From stability analysis for linear surface-gravity waves (Deleersnijder et al., 1997)

$$\Delta t_{2D} \leq \min(\frac{1}{f}, \frac{\Delta h_{min}}{2\sqrt{g h_{max}}}) \tag{3.4.6}$$

and

$$\Delta t_{3D} \leq \min(\frac{1}{f}, \frac{\Delta h_{min}}{2\sqrt{g' h_{max}}}) \tag{3.4.7}$$

where Δh_{min} is the minimum horizontal grid spacing, $g' = g\Delta\rho/\rho_0$ the reduced gravity, h_{max} the maximum water depth and $\Delta\rho$ a typical value for the vertical density difference. Since $g' \ll g$ the second condition is less constraining than the first one. A more stringent condition for the 3-D mode, imposed by the scheme for horizontal advection, is that the horizontal distance travelled by a fluid element during Δt_{3D}, must be smaller than the grid spacing, or

$$\big(\frac{u\Delta t_{3D}}{\Delta x_1}, \frac{v\Delta t_{3D}}{\Delta x_2}\big) \leq 1 \tag{3.4.8}$$

- All horizontal derivatives are evaluated explicitly while vertical diffusion is computed fully implicitly and vertical advection quasi-implicitly.

- A predictor-corrector method is used to solve the horizontal momentum equations (3.1.25)–(3.1.26). This satisfies the requirement (Blumberg and Mellor, 1987) that, when using a mode-splitting technique of solution, the currents in the 3-D equations should have the same depth integral as the ones obtained from the 2-D depth-integrated equations.

- Forward-backward interpolation of the Coriolis term is implemented (Sielecki, 1968).

- Time integration is performed with the operator splitting method in conjunction with the TVD scheme for advection, whereas a simpler forward scheme is considered when advection is discretised with the upwind scheme.

- The sink terms in all non-biological transport equations, representing e.g. the bottom stress in the momentum equation or the dissipation rate ε and work against stable density gradients in e.g. the k-equation (3.1.114), are discretised quasi-implicitly to ensure positivity (Patankar, 1980). The sink terms in the biological module are taken explicitly to ensure that the source and sink terms in the nitrogen balance cancel exactly.

- The time step at which a quantity is evaluated in the discretised equations, is represented by one of the following superscripts (see also Table 3.4.1):
  - n: 3-D quantity at the old baroclinic time level $t^n = n\Delta t_{3D}$
  - $n+1$: 3-D quantity at the new baroclinic time level $t^{n+1} = (n+1)\Delta t_{3D}$
  - m: 2-D quantity at the old barotropic time level $t^m = n\Delta t_{3D} + m\Delta t_{2D}$
  - $m+1$: 2-D quantity at the new barotropic time level $t^{m+1} = n\Delta t_{3D} + (m+1)\Delta t_{2D}$
  - p: horizontal current at the "predicted" time step
  - a double index $n_1 : n_2$ indicating that a quantity is taken at the time t^{n_1} for odd and at time t^{n_2} for even (barotropic or baroclinic) time steps.

  The superscript is omitted if no confusion is possible.

4.3 Momentum equations

4.3.1 General procedure

The 3-D momentum equations are solved by a predictor-corrector method in which the sequence of operations for each baroclinic time step is as follows:

1. An initial (predictor) estimate of the currents u^p, v^p is calculated from the equations of three-dimensional motion.
2. The 2-D depth-integrated equations of continuity and momentum are solved for ζ, $\overline{U}$ and $\overline{V}$. This involves M_t integrations in time.
3. The 3-D horizontal current u^p and v^p are corrected yielding u^{n+1} and v^{n+1} by adjusting u^p and v^p to ensure that the integrated currents obtained from the 2-D and 3-D momentum equations are identical.
4. The transformed and physical vertical current are obtained by solving (3.1.46) and (3.1.47).

a) predictor step

- Firstly, the equation of hydrostatic equilibrium (3.1.27) is integrated for the quantity q_d. The components of the baroclinic pressure gradient are then obtained by substituting q_d in the discretised forms of equation (3.1.30). This is further discussed in Section III-4.3.9.
- Secondly, the integral quantities (3.1.42)–(3.1.45) are evaluated by integrating the horizontal advective and diffusion terms for the velocity deviations u' and v' at the old time t^n.

Table 3.4.2: Parameters and variables used in the numerical description.

Symbol	Name[a]	Purpose
N_x	NC	number of grid cells in the X-direction
N_y	NR	number of grid cells in the Y-direction
N_z	NZ	number of grid cells in the vertical direction
Δx_1	GX2	grid spacing in the X-direction at the cell centre
Δx_2	GY2	grid spacing in the Y-direction at the cell centre
Δx_3	GZ2	grid spacing in the vertical direction (physical space) at the cell centre $(= J\Delta\tilde{x}_3 = J)$
$\Delta\sigma_k$	GZS	grid spacing in the vertical direction (σ-space) at the cell centre
Δt_{2D}	DELT	time step for the 2-D mode equations
Δt_{3D}	—	time step used for the update of 3-D momentum (3-D mode), all scalar quantities and the particle positions in the Lagrangian module
M_t	IC3D	number of 2-D (barotropic) time step within one 3-D (baroclinic) time step $(= \Delta t_{3D}/\Delta t_{2D})$
N_t	—	total number of 3-D time steps used in the simulation
M_{tot}	NSTEP	total number of 2-D time steps used in the simulation
θ_a	THIMP	implicity factor for vertical advection with a value between 0 (explicit) and 1 (implicit). The value, currently used in the program, is 0.501.
θ_v	DTHIMP	implicity factor for vertical diffusion with a value between 0 (explicit) and 1 (implicit). The value, currently used in the program, is 1.
$\Omega(r)$	—	weight function between the upwind and Lax-Wendroff (central) fluxes used in the evaluation of the horizontal (vertical) advective fluxes. Its value depends on the value of the switches IADVC (momentum) and IADVS (scalars), as given by (3.4.52)–(3.4.55).
u^F	U2F	X-component of the "filtered" advective velocity, defined by (3.4.21), used for the advection of scalar quantities
v^F	V2F	Y-component of the "filtered" advective velocity, defined by (3.4.22), used for the advection of scalar quantities
$\overline{U}^F$	UD2F	value of the depth-integrated current $\overline{U}$ averaged over one baroclinic time step, as given by (3.4.18)
$\overline{V}^F$	VD2F	value of the depth-integrated current $\overline{V}$ averaged over one baroclinic time step, as given by (3.4.18)

[a]FORTRAN name used in the program.

- Thirdly, the 3-D momentum equations (3.1.25) and (3.1.26) are integrated in time at each (internal) grid point (k,j,i). Their discretised forms without operator splitting, is given by

$$\begin{aligned}\frac{u^p-u^n}{\Delta t_{3D}} &= fv^{u;n:p} - \mathcal{A}_h(u^n) - \theta_a\mathcal{A}_v(u^p) - (1-\theta_a)\mathcal{A}_v(u^n) + \theta_v\mathcal{D}_v(u^p)\\ &+(1-\theta_v)\mathcal{D}_v(u^n) - g\frac{\Delta^u_x\zeta^n}{\Delta^u x_1} - \frac{1}{\rho_0}\frac{\Delta^u_x P_a}{\Delta^u x_1} + Q^n_1 + \mathcal{D}_{xx}(u^n) + \mathcal{D}_{yx}(u^n,v^n)\end{aligned} \tag{3.4.9}$$

$$\begin{aligned}\frac{v^p-v^n}{\Delta t_{3D}} &= -fu^{v;p:n} - \mathcal{A}_h(v^n) - \theta_a\mathcal{A}_v(v^p) - (1-\theta_a)\mathcal{A}_v(v^n) + \theta_v\mathcal{D}_v(v^p)\\ &+(1-\theta_v)\mathcal{D}_v(v^n) - g\frac{\Delta^v_y\zeta^n}{\Delta^v x_2} - \frac{1}{\rho_0}\frac{\Delta^v_y P_a}{\Delta^v x_2} + Q^n_2 + \mathcal{D}_{xy}(u^n,v^n) + \mathcal{D}_{yy}(v^n)\end{aligned} \tag{3.4.10}$$

  where $\mathcal{A}_h$, $\mathcal{A}_v$ are the horizontal and vertical advection operators defined by (3.1.72)–(3.1.76) and the horizontal diffusion operators $\mathcal{D}_{xx}$, $\mathcal{D}_{yx}$, $\mathcal{D}_{xy}$, $\mathcal{D}_{yy}$ represent the last two terms in (3.1.25)–(3.1.26).

- Fourthly, the "predicted" values for the depth-integrated current are obtained by integrating u^p and v^p over the vertical

$$\overline{U}^p = \sum_{k=1}^{N_z} u^p_k J^n_k \ , \ \overline{V}^p = \sum_{k=1}^{N_z} v^p_k J^n_k \tag{3.4.11}$$

The following features are to be noted:

- The forward (Euler) scheme for the time discretisation in (3.4.9)–(3.4.10) is only considered if either all advective terms are set to zero or the upwind scheme is considered for advection. The operator splitting method, discussed in Section III-4.3.2, is used in all other cases.

- As indicated by the double time index in the Coriolis terms, the u-equation is solved before the v-equation at odd time steps, whereas the v-equation is solved before the u-equation at even time steps.

- The implicity factor θ_a for vertical advection and θ_v for vertical diffusion are set to respectively 0.501 (semi-implicit) and 1 (fully implicit method). This is further discussed in Section III-4.3.2.

b) depth-integrated equations

The 2-D continuity equation (3.1.33) for the surface elevation ζ and the depth-integrated momentum equations (3.1.34)–(3.1.35) for $\overline{U}$, $\overline{V}$ are solved at each (internal) grid point (j,i) for $M_t = \Delta t_{3D}/\Delta t_{2D}$ barotropic time steps

$$\frac{\zeta^{m+1}-\zeta^m}{\Delta t_{2D}} = \frac{\Delta^c_x\overline{U}^m}{\Delta x_1} + \frac{\Delta^c_y\overline{V}^m}{\Delta x_2} \tag{3.4.12}$$

$$\begin{aligned}\frac{\overline{U}^{m+1}-\overline{U}^m}{\Delta t_{2D}} + \frac{k_b^{u;p}}{H^{u;n}}\overline{U}^{m+1} &= f\overline{V}^{u;m:m+1} - \overline{A}_1^h - \overline{\mathcal{A}_h}(\overline{U}^m)\\ &-gH^{u;n}\frac{\Delta^u_x\zeta^{m+1}}{\Delta^u x_1} - gH^{u;n}\frac{\Delta^u_x P_a}{\Delta^u x_1} + \overline{Q}_1^n + \frac{\tau^u_{s1}}{\rho_0}\\ &-k_b^{u;p}(u_b^p - \frac{\overline{U}^p}{H^{u;n}}) + \overline{D}_1^h + \overline{\mathcal{D}_{xx}}(\overline{U}^m) + \overline{\mathcal{D}_{yx}}(\overline{U}^m,\overline{V}^m)\end{aligned} \tag{3.4.13}$$

$$\begin{aligned}\frac{\overline{V}^{m+1}-\overline{V}^m}{\Delta t_{2D}} + \frac{k_b^{v;p}}{H^{v;n}}\overline{V}^{m+1} &= -f\overline{U}^{v;m+1:m} - \overline{A}_2^h - \overline{\mathcal{A}_h}(\overline{V}^m)\\ &-gH^{v;n}\frac{\Delta^v_y\zeta^{m+1}}{\Delta^v x_2} - gH^{v;n}\frac{\Delta^v_y P_a}{\Delta^v x_2} + \overline{Q}_2^n + \frac{\tau^v_{s2}}{\rho_0}\\ &-k_b^{v;p}(v_b^p - \frac{\overline{V}^p}{H^{v;n}}) + \overline{D}_2^h + \overline{\mathcal{D}_{xy}}(\overline{U}^m,\overline{V}^m) + \overline{\mathcal{D}_{yy}}(\overline{V}^m)\end{aligned} \tag{3.4.14}$$

where, according to (3.1.184)–(3.1.186), k_b is given by

$$\begin{array}{ll} k_b = 0 & \text{in the absence of bottom stress}\\ k_b = k_{lin} & \text{for a linear friction law}\\ k_b^p = C_D^b\Big((u_b^p)^2+(v_b^p)^2\Big)^{1/2} & \text{in the case of a quadratic friction law}\end{array} \tag{3.4.15}$$

and u_b^p, v_b^p are the velocities taken at the bottom grid cell. Note that $k_b^{u;p}$ in (3.4.13) is obtained by interpolating C_D^b and v_b^p at the U-node, and $k_b^{v;p}$ in (3.4.14) by interpolating C_D^b and u_b^p at the V-node. The advection operator $\overline{\mathcal{A}_h}$ for the depth-integrated current is defined by

$$\begin{aligned}\overline{\mathcal{A}_h}(\overline{Q}) &= \frac{\partial}{\partial x_1}\Big(\frac{\overline{U}\,\overline{Q}}{H}\Big) + \frac{\partial}{\partial x_2}\Big(\frac{\overline{V}\,\overline{Q}}{H}\Big)\\ &= \overline{\mathcal{A}_{hx}}(\overline{Q}) + \overline{\mathcal{A}_{hy}}(\overline{Q})\end{aligned} \tag{3.4.16}$$

and the diffusion operators $\overline{\mathcal{D}_{xx}}$, $\overline{\mathcal{D}_{yx}}$, $\overline{\mathcal{D}_{xy}}$, $\overline{\mathcal{D}_{yy}}$ by the horizontal depth-integrated shear stress terms in (3.1.34) and (3.1.35).

The following remarks are to be given:

- As indicated by the double index in the Coriolis terms, the $\overline{U}$-equation is solved before the $\overline{V}$-equation at odd (barotropic) time steps, whereas the $\overline{V}$-equation is solved before the $\overline{U}$-equation at even time steps.

- A quasi-implicit formulation is used for the bottom stress in the $\overline{U}$-equation of the form
$$\frac{\tau_{b1}^u}{\rho_0} = k_b^{u;p}\Big(u_b^p + \frac{\overline{U}^{m+1} - \overline{U}^p}{H^{u;n}}\Big) \qquad (3.4.17)$$
and a similar expression for τ_{b2}^v in the $\overline{V}$-equation.

After solving (3.4.12)–(3.4.14) M_t times, the solutions are averaged over the baroclinic time step, giving
$$\overline{U}^F = \frac{1}{M_t}\sum_{m=1}^{M_t}\overline{U}^m\,,\,\overline{V}^F = \frac{1}{M_t}\sum_{m=1}^{M_t}\overline{V}^m \qquad (3.4.18)$$
where $M_t = \Delta t_{3D}/\Delta t_{2D}$ is the number of barotropic time steps.

c) corrector step

The predicted values u^p, v^p of the horizontal current are corrected to ensure that the depth-integrated currents obtained from the 2-D mode equations (3.4.13)–(3.4.14) are identical to the depth-integrated values of the 3-D current. The corrected values are then given by

$$u^{n+1} = \frac{J^{u;n}}{J^{u;n+1}}u^p + \frac{\overline{U}^{n+1} - \overline{U}^p}{H^{u;n+1}} \qquad (3.4.19)$$

$$v^{n+1} = \frac{J^{v;n}}{J^{v;n+1}}v^p + \frac{\overline{V}^{n+1} - \overline{V}^p}{H^{v;n+1}} \qquad (3.4.20)$$

The “filtered” advective velocities u^F and v^F, used for the advection of scalar quantities (see Section III-4.4), are similarly derived from the predicted currents and the depth-integrated current averaged over the baroclinic time step:

$$u^F = \frac{J^{u;n}}{J^{u;n+1}}u^p + \frac{\overline{U}^F - \overline{U}^p}{H^{u;n+1}} \qquad (3.4.21)$$

$$v^F = \frac{J^{v;n}}{J^{v;n+1}}v^p + \frac{\overline{V}^F - \overline{V}^p}{H^{v;n+1}} \qquad (3.4.22)$$

d) vertical current

The transformed vertical current $J\tilde{w}$ is obtained by integrating (3.1.46) from the bottom using the “filtered” velocities (u^F,v^F) and ($\overline{U}^F$,$\overline{V}^F$). Omitting the i- and j-indices this gives

$$\begin{aligned}
(J\tilde{w})^{n+1}_1 &= \; = 0 \\
(J\tilde{w})^{n+1}_{k+1} &= (J\tilde{w})^{n+1}_k - \frac{1}{\Delta x_1}\Delta^c_x\Big[J^{u;n+1}_k\Big(u^F_k - \frac{\overline{U}^F}{H^{u;n+1}}\Big)\Big] - \frac{1}{\Delta x_2}\Delta^c_y\Big[J^{v;n+1}_k\Big(v^F_k - \frac{\overline{V}^F}{H^{v;n+1}}\Big)\Big]
\end{aligned} \tag{3.4.23}$$

Finally, the physical vertical current w is computed from (3.1.47):

$$\begin{aligned}
w^{n+1}_k = \frac{1}{J^{n+1}_k}\Big[&\frac{J^{n+1}_k x^{c;n+1}_{3;k} - J^n_k x^{c;n}_{3;k}}{\Delta t_{3D}} + \frac{\Delta^c_x(J^{u;n+1}_k u^{n+1}_k x^{u;n+1}_{3;k})}{\Delta x_1} \\
&+\frac{\Delta^c_y(J^{v;n+1}_k v^{n+1}_k x^{v;n+1}_{3;k})}{\Delta x_2} + \Delta^c_z((J\tilde{w})^{n+1}_k x^{w;n+1}_{3;k})\Big]
\end{aligned} \tag{3.4.24}$$

where

$$\begin{aligned}
x^{c;n}_{3;k} = H^n\sigma^c_k - h \quad &, \quad x^{u;n}_{3;k} = H^{u;n}\sigma^c_k - h^u \\
x^{v;n}_{3;k} = H^{v;n}\sigma^c_k - h^v \quad &, \quad x^{w;n}_{3;k} = H^n\sigma_k - h
\end{aligned} \tag{3.4.25}$$

and similar expressions at the new time step t^{n+1}.

4.3.2 Advection schemes and time discretisation

The time discretisation of the momentum equations depends on the type of advection scheme employed for the spatial discretisation of the horizontal and vertical advection terms. Several schemes are implemented in the program, selected with the model switch IADVC which may take the following values:

0 : horizontal and vertical advection of momentum disabled

1 : upwind scheme for horizontal and vertical advection

2 : Lax-Wendroff scheme for horizontal, central scheme for vertical advection[2]

3 : TVD (Total Variation Diminishing) scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical

4 : as the previous case now using the monotonic limiter.

The upwind scheme has the interesting property to preserve monotonicity, but has the disadvantage of being only first order accurate. The Lax-Wendroff scheme, on the other hand, is accurate to second order but non-monotone which means that spurious over- and

[2]The “pure” Lax-Wendroff scheme has only been implemented for illustrative purposes and should be avoided in realistic simulations (see Chapter II-2).

Table 3.4.3: Operators used in the numerical discretisations.

Type	Purpose
difference operators	
Δ_x	difference operator in the X-direction. A superscript (c,u,v,w,X,Y) is added if the operator is applied to a quantity at a point, not located at its natural boundary
Δ_y	difference operator in the Y-direction. A superscript (c,u,v,w,X,Y) is added if the operator is applied to a quantity at a point, not located at its natural boundary
Δ_z	difference operator in the vertical direction. A superscript (c,u,v,w,X,Y) is added if the operator is applied to a quantity at a point, not located at its natural boundary
advective operators (Cartesian and spherical)	
$\mathcal{A}_{hx}$	horizontal advection in the X-direction[a] $(u,v$ and scalars) $$\mathcal{A}_{hx}(Q) = \frac{1}{J}\frac{\partial}{\partial x_1}(Ju\psi)$$
$\mathcal{A}_{hy}$	horizontal advection in the Y-direction[a] (u, v and scalars) $$\mathcal{A}_{hy}(Q) = \frac{1}{J}\frac{\partial}{\partial x_2}(Jv\psi)\,,\ \mathcal{A}_{hy}(Q) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, Q)$$
$\mathcal{A}_h$	horizontal advection (u, v and scalars) $$\mathcal{A}_h = \mathcal{A}_{hx} + \mathcal{A}_{hy}$$
$\mathcal{A}_v$	vertical advection (u, v and scalars excluding the sinking rate term) $$\mathcal{A}_v(Q) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w}Q)$$
$\tilde{\mathcal{A}}_v$	vertical advection (scalars including sinking rate) $$\tilde{\mathcal{A}}_v(Q) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big((J\tilde{w} + w_s^{\psi})\psi\Big) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(W\psi)$$

[a]In the case of a scalar (u,v) is replaced by (u^F,v^F).

Table 3.4.3: continued.

advective operators (Cartesian and spherical)	
$\overline{\mathcal{A}_{hx}}$	horizontal advection in the X-direction (2-D mode) $\overline{\mathcal{A}_{hx}}(\overline{Q}) = \frac{\partial}{\partial x_1}\Big(\frac{\overline{U}\,\overline{Q}}{H}\Big)$
$\overline{\mathcal{A}_{hy}}$	horizontal advection in the Y-direction (2-D mode) $\overline{\mathcal{A}_{hy}}(\overline{Q}) = \frac{\partial}{\partial x_2}\Big(\frac{\overline{V}\,\overline{Q}}{H}\Big)\,,\ \overline{\mathcal{A}_{hy}}(\overline{Q}) = \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big(\cos\phi\frac{\overline{V}\,\overline{Q}}{H}\Big)$
$\overline{\mathcal{A}_{h}}$	horizontal advection (2-D mode) $\overline{\mathcal{A}_{h}} = \overline{\mathcal{A}_{hx}} + \overline{\mathcal{A}_{hy}}$
diffusion operators (Cartesian and spherical)	
$\mathcal{D}_{hx}$	horizontal diffusion in the X-direction (scalars) $\mathcal{D}_{hx}(\psi) = \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\lambda_H\frac{\partial\psi}{\partial x_1}\Big)$
$\mathcal{D}_{hy}$	horizontal diffusion in the Y-direction (scalars) $\mathcal{D}_{hy}(\psi) = \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\lambda_H\frac{\partial\psi}{\partial x_2}\Big)$; $\mathcal{D}_{hy}(\psi) = \frac{1}{R^2\cos\phi}\frac{\partial}{\partial\phi}\Big(J\cos\phi\lambda_H\frac{\partial\psi}{\partial\phi}\Big)$
$\mathcal{D}_{h}$	horizontal diffusion (scalars) $\mathcal{D}_h = \mathcal{D}_{hx} + \mathcal{D}_{hy}$
$\mathcal{D}_{xx}$	horizontal diffusion of u in the X-direction $\mathcal{D}_{xx}(u) = \frac{1}{J}\frac{\partial}{\partial x_1}\Big(2\nu_H J\frac{\partial u}{\partial x_1}\Big)$; $\mathcal{D}_{xx}(u) = \frac{1}{R}\frac{\partial}{\partial\lambda}\Big[2\nu_H J\Big(\frac{1}{R\cos\phi}\frac{\partial u}{\partial\lambda} - \frac{\tan\phi}{R}v\Big)\Big]$

Table 3.4.3: continued.

diffusion operators (Cartesian and spherical)	
$\mathcal{D}_{yx}$	horizontal diffusion of u in the Y-direction $\mathcal{D}_{yx}(u,v) = \frac{1}{J}\frac{\partial}{\partial x_2}\big[\nu_H J\big(\frac{\partial u}{\partial x_2}+\frac{\partial v}{\partial x_1}\big)\big]$ $\mathcal{D}_{yx}(u,v) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}\big[\nu_H J\cos\phi\big(\frac{1}{R}\frac{\partial u}{\partial\phi}+\frac{\tan\phi}{R}u+\frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}\big)\big]$
$\mathcal{D}_{xy}$	horizontal diffusion of v in the X-direction $\mathcal{D}_{xy}(u,v) = \frac{1}{J}\frac{\partial}{\partial x_1}\big[\nu_H J\big(\frac{\partial u}{\partial x_2}+\frac{\partial v}{\partial x_1}\big)\big]$ $\mathcal{D}_{xy}(u,v) = \frac{1}{JR}\frac{\partial}{\partial\lambda}\big[\nu_H J\big(\frac{1}{R}\frac{\partial u}{\partial\phi}+\frac{\tan\phi}{R}u+\frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}\big)\big]$
$\mathcal{D}_{yy}$	horizontal diffusion of v in the Y-direction $\mathcal{D}_{yy}(v) = \frac{1}{J}\frac{\partial}{\partial x_2}\big(2\nu_H J\frac{\partial v}{\partial x_2}\big)$ $\mathcal{D}_{yy}(v) = \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\big(2\nu_H J\cos\phi\frac{\partial v}{\partial\phi}\big)$
$\mathcal{D}_v$	vertical diffusion[a] (u, v and scalars) $\mathcal{D}_v(Q) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\big(\lambda_T^\psi\frac{\partial Q}{\partial\tilde{x}_3}\big)$
$\overline{\mathcal{D}_{xx}}$	horizontal diffusion of $\overline{U}$ in the X-direction $\overline{\mathcal{D}_{xx}}(\overline{U}) = \frac{\partial}{\partial x_1}\big[2\overline{\nu_H}\frac{\partial}{\partial x_1}\big(\frac{\overline{U}}{H}\big)\big]$ $\overline{\mathcal{D}_{xx}}(\overline{U}) = \frac{1}{R}\frac{\partial}{\partial\lambda}\big[2\overline{\nu_H}\big(\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\big(\frac{\overline{U}}{H}\big)-\frac{\tan\phi}{R}\frac{\overline{V}}{H}\big)\big]$
$\overline{\mathcal{D}_{yx}}$	horizontal diffusion of $\overline{U}$ in the Y-direction $\overline{\mathcal{D}_{yx}}(\overline{U},\overline{V}) = \frac{\partial}{\partial x_2}\big[\overline{\nu_H}\big(\frac{\partial}{\partial x_2}\big(\frac{\overline{U}}{H}\big)+\frac{\partial}{\partial x_1}\big(\frac{\overline{V}}{H}\big)\big)\big]$ $\overline{\mathcal{D}_{yx}}(\overline{U},\overline{V}) = \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\big[\overline{\nu_H}\cos\phi\big(\frac{1}{R}\frac{\partial}{\partial\phi}\big(\frac{\overline{U}}{H}\big)+\frac{\tan\phi}{R}\frac{\overline{U}}{H}+\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\big(\frac{\overline{V}}{H}\big)\big)\big]$

[a]Note that λ_T^ψ is replaced by ν_T in the u- and v-equation.

Table 3.4.3: continued.

diffusion operators (Cartesian and spherical)	
$\overline{\mathcal{D}_{xy}}$	horizontal diffusion of $\overline{V}$ in the X-direction $$\overline{\mathcal{D}_{xy}}(\overline{U},\overline{V}) = \frac{\partial}{\partial x_1}\Big[\overline{\nu_H}\Big(\frac{\partial}{\partial x_2}\Big(\frac{\overline{U}}{H}\Big)+\frac{\partial}{\partial x_1}\Big(\frac{\overline{V}}{H}\Big)\Big)\Big]$$ $$\overline{\mathcal{D}_{xy}}(\overline{U},\overline{V}) = \frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big[\overline{\nu_H}\Big(\frac{1}{R}\frac{\partial}{\partial\phi}\Big(\frac{\overline{U}}{H}\Big)+\frac{\tan\phi}{R}\frac{\overline{U}}{H}+\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\Big(\frac{\overline{V}}{H}\Big)\Big)\Big]$$
$\overline{\mathcal{D}_{yy}}$	horizontal diffusion of $\overline{V}$ in the Y-direction $$\overline{\mathcal{D}_{yy}}(\overline{v}) = \frac{\partial}{\partial x_2}\Big[2\overline{\nu_H}\frac{\partial}{\partial x_2}\Big(\frac{\overline{V}}{H}\Big)\Big]$$ $$\overline{\mathcal{D}_{yy}}(\overline{V}) = \frac{1}{R\cos\phi}\frac{\partial}{\partial\phi}\Big[2\overline{\nu_H}\cos\phi\frac{\partial}{\partial\phi}\Big(\frac{\overline{V}}{H}\Big)\Big]$$
other operators	
$\mathcal{P}$	source terms in the scalar transport equations
$\mathcal{S}$	minus the sink terms in the scalar transport equations
β	source and sink terms in the scalar transport equations $$\beta = \mathcal{P} - \mathcal{S}$$
$\mathcal{M}_x$	X-corrector term in the scalar transport equations $$\mathcal{M}_x(\psi) = \frac{\psi}{J}\frac{\partial}{\partial x_1}(Ju^F)$$
$\mathcal{M}_y$	Y-corrector term in the scalar transport equations $$\mathcal{M}_y(\psi) = \frac{\psi}{J}\frac{\partial}{\partial x_2}(Jv^F)\,,\ \mathcal{M}_y(\psi) = \frac{\psi}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi v^F)$$
$\mathcal{M}_z$	Z-corrector term in the scalar transport equations $$\mathcal{M}_z(\psi) = \frac{\psi}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{w})$$

undershootings are created in regimes of strong gradients. This is clearly illustrated by the results of the test cases **cones** and **front** described in Sections II-2.1 and II-2.2. The TVD scheme has the advantage of combining the monotonicity of the upwind scheme with the second order accuracy of the Lax-Wendroff scheme.

Horizontal advection is evaluated explicitly to prevent the solution of large-banded matrix systems. A necessary stability condition for both the upwind and the Lax-Wendroff scheme is given by the criterion (3.4.8) (see e.g. Hirsch, 1988). The restriction to explicit schemes does not apply in the vertical where the discretised equations are written into a tridiagonal form (see Section III-4.3.12). A semi-implicit scheme in the vertical allows to replace the Lax-Wendroff by the central scheme which is a monotone scheme and stable provided that the implicity factor $\theta_a \geq 0.5$.

The aim of the limiter function is to reduce the numerical diffusion due to the upwind scheme in areas of low gradients and to provide sufficiently large diffusion in regions of large gradients so that over- and undershooting due to the non-monotonicity of the Lax-Wendroff scheme are suppressed. Both the superbee (Roe, 1985) as the monotonic limiter are available in the program. The **cones** and **front** test case simulations (see Sections II-2.1 and II-2.2) showed that the superbee limiter is the least diffusive and is therefore taken as the default formulation in the program.

For more details about the advection schemes, see Ruddick (1995). The spatial discretisation of the advective terms in the momentum equations and the form of the limiter function are further discussed in the subsections below.

In the absence of advection (IADVC $= 0$) or when the upwind scheme is selected (IADVC $= 1$), the momentum equations are solved by forward time-stepping as given by the time-discretised forms (3.4.9) and (3.4.10). If IADVC > 1, the spatial discretisation of the advective terms involves the Lax-Wendroff and central schemes which are both second order accurate in space. The equations are then integrated in time with the aid of the "fractional step" or "operator splitting" method as proposed by Yanenko (1971). The procedure consists in splitting the time integration into three fractional steps. During the first and second step only the advection-diffusion terms in respectively the X- and Y-direction are taken into account. The vertical advection and diffusion terms and all other terms (Coriolis force and pressure gradient) are included during the third time step. To preserve the second-order accuracy of the 1-D schemes in the fractional step approach the method of symmetric splitting (e.g. Hirsch, 1990) is implemented. This means that the previous procedure ("A"-steps) is repeated now in reverse order ("B"-steps) , i.e. vertical advection/diffusion and other terms followed by advection-diffusion in the Y-direction, followed by advection-diffusion in the X-direction. The final "predicted" value of u^p or v^p is then obtained by taking the average of the values at the end of the A- and B-steps.

The implicity factors θ_a and θ_v have a range between 0 and 1 where a 0 corresponds to a fully explicit, 1 a fully implicit and 0.5 a semi-implicit (Crank-Nicholson) method. The schemes are stable provided that $\theta_a, \theta_v > 0.5$. To retain the same accuracy in time for horizontal as well as for vertical advection the semi-implicit option is taken for vertical advection, i.e. $\theta_a = 0.501$ [3,4]. Vertical diffusion is treated fully implicitly ($\theta_v = 1$) in the current version of the program[5]. For more details see Ruddick (1995).

The time-discretised form of the u-equation is then given by

- Part A

$$\frac{u_A^{n+1/3} - u_A^n}{\Delta t_{3D}} = -\mathcal{A}_{hx}(u^n) + \mathcal{D}_{xx}(u^n) \tag{3.4.26}$$

$$\frac{u_A^{n+2/3} - u_A^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(u_A^{n+1/3}) + \mathcal{D}_{yx}(u^n, v^n) \tag{3.4.27}$$

$$\frac{u_A^p - u_A^{n+2/3}}{\Delta t_{3D}} = -\theta_a \mathcal{A}_v(u_A^p) - (1-\theta_a)\mathcal{A}_v(u_A^{n+2/3}) + \theta_v \mathcal{D}_v(u_A^p) + (1-\theta_v)\mathcal{D}_v(u_A^{n+2/3}) + \mathcal{O}_x^n \tag{3.4.28}$$

- Part B

$$\frac{u_B^{n+1/3} - u_B^n}{\Delta t_{3D}} = -\theta_a \mathcal{A}_v(u_B^{n+1/3}) - (1-\theta_a)\mathcal{A}_v(u^n) + \theta_v \mathcal{D}_v(u_B^{n+1/3}) + (1-\theta_v)\mathcal{D}_v(u^n) + \mathcal{O}_x^n \tag{3.4.29}$$

$$\frac{u_B^{n+2/3} - u_B^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(u_B^{n+1/3}) + \mathcal{D}_{yx}(u^n, v^n) \tag{3.4.30}$$

$$\frac{u_B^p - u_B^{n+2/3}}{\Delta t_{3D}} = -\mathcal{A}_{hx}(u_B^{n+2/3}) + \mathcal{D}_{xx}(u^n) \tag{3.4.31}$$

- Predictor value

$$u^p = \frac{1}{2}(u_A^p + u_B^p) \tag{3.4.32}$$

Note that in (3.4.26)–(3.4.27) and (3.4.30)–(3.4.31) the horizontal advection operator has been split into a X- and a Y-derivative part

$$\mathcal{A}_h(u) = \mathcal{A}_{hx}(u) + \mathcal{A}_{hy}(v) \tag{3.4.33}$$

The $\mathcal{O}_x$-terms are defined by

$$\mathcal{O}_x^n = f v^{u;n:p} - g\frac{\Delta_x^u \zeta^n}{\Delta^u x_1} - \frac{1}{\rho_0}\frac{\Delta_x^u P_a}{\Delta^u x_1} + Q_1^n \tag{3.4.34}$$

A similar procedure is applied for the v-equation.

[3]The central scheme is second accurate in time if $\theta_a = 0.5$.

[4]Note that the schemes are only first order accurate in space in the case of an irregular grid spacing.

[5]A semi-implicit option for vertical diffusion requires only a minor modification of the source code and can be adopted in future versions of the program.

- Part A

$$\frac{v_A^{n+1/3} - v_A^n}{\Delta t_{3D}} = -\mathcal{A}_{hx}(v^n) + \mathcal{D}_{xy}(u^n, v^n) \tag{3.4.35}$$

$$\frac{v_A^{n+2/3} - v_A^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(v_A^{n+1/3}) + \mathcal{D}_{yy}(v^n) \tag{3.4.36}$$

$$\frac{v_A^p - v_A^{n+2/3}}{\Delta t_{3D}} = -\theta_a\mathcal{A}_v(v_A^p) - (1-\theta_a)\mathcal{A}_v(v_A^{n+2/3}) + \theta_v\mathcal{D}_v(v_A^p) + (1-\theta_v)\mathcal{D}_v(v_A^{n+2/3}) + \mathcal{O}_y^n \tag{3.4.37}$$

- Part B

$$\frac{v_B^{n+1/3} - v_B^n}{\Delta t_{3D}} = -\theta_a\mathcal{A}_v(v_B^{n+1/3}) - (1-\theta_a)\mathcal{A}_v(v^n) + \theta_v\mathcal{D}_v(v_B^{n+1/3}) + (1-\theta_v)\mathcal{D}_v(v^n) + \mathcal{O}_y^n \tag{3.4.38}$$

$$\frac{v_B^{n+2/3} - v_B^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(v_B^{n+1/3}) + \mathcal{D}_{yy}(v^n) \tag{3.4.39}$$

$$\frac{v_B^p - v_B^{n+2/3}}{\Delta t_{3D}} = -\mathcal{A}_{hx}(v_B^{n+2/3}) + \mathcal{D}_{xy}(u^n, v^n) \tag{3.4.40}$$

- Predictor value

$$v^p = \frac{1}{2}(v_A^p + v_B^p) \tag{3.4.41}$$

The $\mathcal{O}_y$-terms are defined by

$$\mathcal{O}_y^n = -fu^{v;p:n} - g\frac{\Delta_y^v\zeta^n}{\Delta^v x_2} - \frac{1}{\rho_0}\frac{\Delta_y^v P_a}{\Delta^v x_2} + Q_2^n \tag{3.4.42}$$

Important to note again is that, compared to the simpler forward scheme, the computation using symmetrical operator splitting increases the CPU time for the circulation module by a factor two, but has the adventage of being more accurate which is an important property in regions of strong horizontal and vertical shear.

The 2-D mode equations are solved with a much smaller time step than the 3-D mode, so that second-order accuracy is of less relevance. The equations are therefore integrated forward in time as given by the forms (3.4.13)–(3.4.14).

4.3.3 Discretisation of horizontal advection (3-D)

Four advective terms are to be evaluated in the 3-D momentum equations. Their discretised forms are separately discussed below. To simplify the mathematical expressions the following notations are introduced (see also Table 3.4.4):

$$\mathcal{A}_{hx}(u) = \frac{1}{J}\frac{\partial}{\partial x_1}(Juu) = \frac{1}{J}\frac{\partial}{\partial x_1}F_{xx} \tag{3.4.43}$$

Table 3.4.4: Notations for the fluxes used in the numerical discretisations.

Type	Purpose
advective fluxes (Cartesian and spherical)	
F_x	advective flux of a scalar in the X-direction at the U-node $$F_x = Ju\psi$$
F_y	advective flux of a scalar in the Y-direction at the V-node $$F_y = Jv\psi\,,\; F_y = J\cos\phi\, v\psi$$
F_z	advective flux of a scalar in the vertical direction at the W-node $$F_z = (J\tilde{w} + w_s^{\psi})\psi = W\psi$$
F_{xx}	advective flux of u in the X-direction at the cell centre $$F_{xx} = Juu$$
F_{yx}	advective flux of a u in the Y-direction at the V-node $$F_{yx} = Jvu\,,\; F_{yx} = J\cos\phi\, uv$$
F_{yx}^W	advective flux of u in the Y-direction using u-values “West” of the V-node
F_{yx}^E	advective flux of u in the Y-direction using u-values “East” of the V-node
F_{xy}	advective flux of a v in the X-direction at the U-node $$F_{xy} = Juv$$
F_{xy}^S	advective flux of v in the X-direction using v-values “South” of the U-node
F_{xy}^N	advective flux of v in the X-direction using v-values “North” of the U-node
F_{xz}	advective flux of u in the vertical direction at the X-node $$F_{xz} = J\tilde{w}u$$
F_{yz}	advective flux of v in the vertical direction at the Y-node $$F_{yz} = J\tilde{w}v$$
diffusive fluxes (Cartesian and spherical)	
D_x	diffusive flux of a scalar in the X-direction at the U-node $$D_x = \lambda_H J\frac{\partial\psi}{\partial x_1}$$
D_y	diffusive flux of a scalar in the Y-direction at the V-node $$D_y = \lambda_H J\frac{\partial\psi}{\partial x_2}\,,\; D_y = \lambda_H J\cos\phi\frac{1}{R}\frac{\partial\psi}{\partial\phi}$$
D_z	diffusive flux of a scalar in the vertical direction at the W-node $$D_z = \frac{\lambda_T^{\psi}}{J}\frac{\partial\psi}{\partial\tilde{x}_3}$$
D_{xx}	diffusive flux in the X-direction (u-equation) at the cell centre $$D_{xx} = 2\nu_H J\frac{\partial u}{\partial x_1}\,,\; D_{xx} = 2\nu_H J\Big(\frac{1}{R}\frac{\partial u}{\partial\lambda} - \frac{\tan\phi}{R}v\Big)$$

Table 3.4.4: continued.

diffusive fluxes (Cartesian and spherical)	
D^U_{yx}	part of the diffusive flux in the Y-direction (u-equation) taken at the V-node $$D^U_{yx}=\nu_H J\frac{\partial u}{\partial x_2}\,,\; D^U_{yx}=\nu_H J\cos\phi\frac{1}{R}\frac{\partial u}{\partial\phi}$$
D^V_{yx}	part of the diffusive flux in the Y-direction (u-equation) taken at the U-node $$D^V_{yx}=\nu_H J\frac{\partial v}{\partial x_1}\,,\; D^V_{yx}=\nu_H J\cos\phi\Big(\frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}+\frac{\tan\phi}{R}u\Big)$$
D^U_{xy}	part of the diffusive flux in the X-direction (v-equation) taken at the V-node $$D^U_{xy}=\nu_H J\frac{\partial u}{\partial x_2}\,,\; D^U_{xy}=\nu_H J\cos\phi\frac{1}{R}\frac{\partial u}{\partial\phi}$$
D^V_{xy}	part of the diffusive flux in the X-direction (v-equation) taken at the U-node $$D^V_{xy}=\nu_H J\frac{\partial v}{\partial x_1}\,,\; D^V_{xy}=\nu_H J\Big(\frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}+\frac{\tan\phi}{R}u\Big)$$
D_{yy}	diffusive flux in the Y-direction (v-equation) at the cell centre $$D_{yy}=2\nu_H J\frac{\partial v}{\partial x_2}\,,\; D_{yy}=2\nu_H J\cos\phi\frac{1}{R}\frac{\partial v}{\partial\phi}$$
$\tilde{D}_{xx}$	horizontal shear stress $\tau_{\lambda\lambda}$ at the cell centre $$\tilde{D}_{xx}=2\nu_H\Big(\frac{1}{R}\frac{\partial u}{\partial\lambda}-\frac{\tan\phi}{R}v\Big)$$
$\tilde{D}^U_{yx}$	part of the horizontal shear stress $\tau_{\phi\lambda}$ taken at V-node $$\tilde{D}^U_{yx}=\nu_H\frac{1}{R}\frac{\partial u}{\partial\phi}$$
$\tilde{D}^V_{yx}$	part of the horizontal shear stress $\tau_{\phi\lambda}$ taken at the U-node $$\tilde{D}^V_{yx}=\nu_H\Big(\frac{1}{R\cos\phi}\frac{\partial v}{\partial\lambda}+\frac{\tan\phi}{R}u\Big)$$
D_{xz}	diffusive flux of u in the vertical direction at the X-node $$D_{xz}=\frac{\nu_T}{J}\frac{\partial u}{\partial\tilde{x}_3}$$
D_{yz}	diffusive flux of v in the vertical direction at the Y-node $$D_{yz}=\frac{\nu_T}{J}\frac{\partial v}{\partial\tilde{x}_3}$$

$$\mathcal{A}_{hy}(u) = \frac{1}{J}\frac{\partial}{\partial x_2}(Jvu) = \frac{1}{J}\frac{\partial}{\partial x_2}F_{yx} \tag{3.4.44}$$

$$\mathcal{A}_{hx}(v) = \frac{1}{J}\frac{\partial}{\partial x_1}(Juv) = \frac{1}{J}\frac{\partial}{\partial x_1}F_{xy} \tag{3.4.45}$$

$$\mathcal{A}_{hy}(v) = \frac{1}{J}\frac{\partial}{\partial x_2}(Jvv) = \frac{1}{J}\frac{\partial}{\partial x_2}F_{yy} \tag{3.4.46}$$

For simplicity, the k-index has been omitted in the expressions given below.

a) X-derivative in the u-equation

The advective term is obtained by differencing the centered flux F^c_{xx} at the U-node

$$\mathcal{A}_{hx}(u)_{ji} = \frac{F^c_{xx;ji} - F^c_{xx;j,i-1}}{J^u_{ji}\Delta^u x_{1;ji}} \tag{3.4.47}$$

The flux is calculated from

$$F^c_{xx;ji} = J_{ji}\Big((1-\Omega(r^{U;c}_{ji}))F^c_{up;ji} + \Omega(r^{U;c}_{ji})F^c_{lw;ji}\Big) \tag{3.4.48}$$

where F^c_{up} and F^c_{lw} are the upwind and Lax-Wendroff flux at the cell centre

$$F^c_{up;ji} = \frac{1}{2}u^c_{ji}\Big((1+s_{ji})u_{ji} + (1-s_{ji})u_{j,i+1}\Big) \tag{3.4.49}$$

$$F^c_{lw;ji} = \frac{1}{2}u^c_{ji}\Big((1+c_{ji})u_{ji} + (1-c_{ji})u_{j,i+1}\Big) \tag{3.4.50}$$

and s_{ji}, c_{ji} are the sign and CFL-number of the advecting current

$$s_{ji} = \mathrm{Sign}(u_{ji})\,,\; c_{ji} = \frac{u^c_{ji}\Delta t_{3D}}{\Delta x_1} \tag{3.4.51}$$

The form of the function $\Omega(r)$ depends on the type of advection scheme, selected with the switch IADVC.

- upwind
$$\Omega(r) = 0 \tag{3.4.52}$$
- Lax-Wendroff
$$\Omega(r) = 1 \tag{3.4.53}$$
- TVD with superbee limiter
$$\Omega(r) = \max(0, \min(2r,1), \min(r,2)) \tag{3.4.54}$$
- TVD with monotonic limiter
$$\Omega(r) = \frac{r+|r|}{1+|r|} \tag{3.4.55}$$

The argument r depends on the sign of the advecting current

$$\begin{aligned} r_{ji}^{U;c} &= \frac{F^c_{lw;j,i-1} - F^c_{up;j,i-1}}{F^c_{lw;ji} - F^c_{up;ji}} \qquad \text{if} \quad u^c_{ji} > 0 \\ r_{ji}^{U;c} &= \frac{F^c_{lw;j,i+1} - F^c_{up;j,i+1}}{F^c_{lw;ji} - F^c_{up;ji}} \qquad \text{if} \quad u^c_{ji} < 0 \end{aligned} \tag{3.4.56}$$

b) Y-derivative in the v-equation

The procedure is similar to the previous case. Differencing the centered flux F^c_{yy} at the V-node one has

$$\mathcal{A}_{hy}(v)_{ji} = \frac{F^c_{yy;ji} - F^c_{yy;j-1,i}}{J^v_{ji}\Delta^v x_{2;j}} \tag{3.4.57}$$

where

$$F^c_{yy;ji} = J_{ji}\Big((1-\Omega(r^{V;c}_{ji}))F^c_{up;ji} + \Omega(r^{V;c}_{ji})F^c_{lw;ji}\Big) \tag{3.4.58}$$

and

$$F^c_{up;ji} = \frac{1}{2}v^c_{ji}\Big((1+s_{ji})v_{ji} + (1-s_{ji})v_{j+1,i}\Big) \tag{3.4.59}$$

$$F^c_{lw;ji} = \frac{1}{2}v^c_{ji}\Big((1+c_{ji})v_{ji} + (1-c_{ji})v_{j+1,i}\Big) \tag{3.4.60}$$

$$s_{ji} = \text{Sign}(v_{ji})\,,\; c_{ji} = \frac{v^c_{ji}\Delta t_{3D}}{\Delta x_2} \tag{3.4.61}$$

The form of the function $\Omega(r)$ depends on the value of the switch IADVC as given by (3.4.52)–(3.4.55). In analogy with (3.4.56) one has

$$\begin{aligned} r_{ji}^{V;c} &= \frac{F^c_{lw;j-1,i} - F^c_{up;j-1,i}}{F^c_{lw;ji} - F^c_{up;ji}} \qquad \text{if} \quad v^c_{ji} > 0 \\ r_{ji}^{V;c} &= \frac{F^c_{lw;j+1,i} - F^c_{up;j+1,i}}{F^c_{lw;ji} - F^c_{up;ji}} \qquad \text{if} \quad v^c_{ji} < 0 \end{aligned} \tag{3.4.62}$$

c) Y-derivative in the u-equation

The discretisation of the cross-derivative terms is more complicated due to the staggering of u and v on the C-grid. The procedure for computing $\mathcal{A}_{hy}(u)$ consists of the following steps.

1. The upwind and Lax-Wendroff fluxes are evaluated at the V-nodes using values of the advected quantity u "West" of the V-node

$$F^{W;v}_{up;ji} = \frac{1}{2}v_{ji}\Big((1+s_{ji})u_{j-1,i} + (1-s_{ji})u_{ji}\Big) \tag{3.4.63}$$

$$F^{W;v}_{lw;ji} = \frac{1}{2}v_{ji}\Big((1+c_{ji})u_{j-1,i} + (1-c_{ji})u_{ji}\Big) \tag{3.4.64}$$

where

$$s_{ji} = \mathrm{Sign}(v_{ji})\,,\ c_{ji} = \frac{v_{ji}\Delta t_{3D}}{\Delta^v x_{2;j}} \tag{3.4.65}$$

The flux "West" of the V-node is then determined by

$$F^{W;v}_{yx;ji} = J^v_{ji}\Big((1-\Omega(r^{W;v}_{ji}))F^{W;v}_{up;ji} + \Omega(r^{W;v}_{ji})F^{W;v}_{lw;ji}\Big) \tag{3.4.66}$$

where $\Omega(r)$ is given by one the expressions (3.4.52)–(3.4.55) selected by IADVC and

$$\begin{aligned} r^{W;v}_{ji} &= \frac{F^{W;v}_{lw;j-1,i} - F^{W;v}_{up;j-1,i}}{F^{W;v}_{lw;ji} - F^{W;v}_{up;ji}} \qquad \text{if} \quad v_{ji} > 0 \\ r^{W;v}_{ji} &= \frac{F^{W;v}_{lw;j+1,i} - F^{W;v}_{up;j+1,i}}{F^{W;v}_{lw;ji} - F^{W;v}_{up;ji}} \qquad \text{if} \quad v_{ji} < 0 \end{aligned} \tag{3.4.67}$$

2. The upwind and Lax-Wendroff fluxes are evaluated at the V-nodes using values of the advected quantity u "East" of the V-node

$$F^{E;v}_{up;ji} = \frac{1}{2}v_{ji}\Big((1+s_{ji})u_{j-1,i+1} + (1-s_{ji})u_{j,i+1}\Big) \tag{3.4.68}$$

$$F^{E;v}_{lw;ji} = \frac{1}{2}v_{ji}\Big((1+c_{ji})u_{j-1,i+1} + (1-c_{ji})u_{j,i+1}\Big) \tag{3.4.69}$$

with s_{ji} and c_{ji} defined by (3.4.65). The flux "East" of the V-node is then determined by

$$F^{E;v}_{yx;ji} = J^v_{ji}\Big((1-\Omega(r^{E;v}_{ji}))F^{E;v}_{up;ji} + \Omega(r^{E;v}_{ji})F^{E;v}_{lw;ji}\Big) \tag{3.4.70}$$

where $\Omega(r)$ is given by one the expressions (3.4.52)–(3.4.55) selected by IADVC and

$$\begin{aligned} r^{E;v}_{ji} &= \frac{F^{E;v}_{lw;j-1,i} - F^{E;v}_{up;j-1,i}}{F^{E;v}_{lw;ji} - F^{E;v}_{up;ji}} \qquad \text{if} \quad v_{ji} > 0 \\ r^{E;v}_{ji} &= \frac{F^{E;v}_{lw;j+1,i} - F^{E;v}_{up;j+1,i}}{F^{E;v}_{lw;ji} - F^{E;v}_{up;ji}} \qquad \text{if} \quad v_{ji} < 0 \end{aligned} \tag{3.4.71}$$

3. The advective term in the Y-direction is computed by averaging the "West"-fluxes East of the U-node and the "East"-fluxes West of the U-node and differencing the resulting fluxes in the Y-direction at the U-node

$$\mathcal{A}_{hy}(u)_{ji} = \frac{F^{W;v}_{yx;j+1,i} + F^{E;v}_{yx;j+1,i-1} - F^{W;v}_{yx;ji} - F^{E;v}_{yx;j,i-1}}{2J^u_{ji}\Delta^u x_{2;j}} \tag{3.4.72}$$

The procedure is illustrated schematically in Figures 5.1.7a–b of the Reference Manual.

d) X-derivative in the v-equation

The discretisation of $\mathcal{A}_{hx}(v)$ is similar to the previous case and proceeds along the following steps.

1. The upwind and Lax-Wendroff fluxes are evaluated at the U-nodes using values of the advected quantity v "South" of the U-node

$$F^{S;u}_{up;ji} = \frac{1}{2}u_{ji}\Big((1+s_{ji})v_{j,i-1} + (1-s_{ji})v_{ji}\Big) \tag{3.4.73}$$

$$F^{S;u}_{lw;ji} = \frac{1}{2}u_{ji}\Big((1+c_{ji})v_{j,i-1} + (1-c_{ji})v_{ji}\Big) \tag{3.4.74}$$

where

$$s_{ji} = \mathrm{Sign}(u_{ji})\,,\; c_{ji} = \frac{u_{ji}\Delta t_{3D}}{\Delta^u x_{1;ji}} \tag{3.4.75}$$

The flux "South" of the U-node is then determined by

$$F^{S;u}_{xy;ji} = J^u_{ji}\Big((1-\Omega(r^{S;u}_{ji}))F^{S;u}_{up;ji} + \Omega(r^{S;u}_{ji})F^{S;u}_{lw;ji}\Big) \tag{3.4.76}$$

where $\Omega(r)$ is given by one the expressions (3.4.52)–(3.4.55) selected by IADVC and

$$\begin{aligned} r^{S;u}_{ji} &= \frac{F^{S;u}_{lw;j,i-1} - F^{S;u}_{up;j,i-1}}{F^{S;u}_{lw;ji} - F^{S;u}_{up;ji}} & \text{if} \quad u_{ji} > 0 \\ r^{S;u}_{ji} &= \frac{F^{S;u}_{lw;j,i+1} - F^{S;u}_{up;j,i+1}}{F^{S;u}_{lw;ji} - F^{S;u}_{up;ji}} & \text{if} \quad u_{ji} < 0 \end{aligned} \tag{3.4.77}$$

2. The upwind and Lax-Wendroff fluxes are evaluated at the U-nodes using values of the advected quantity u "North" of the U-node

$$F^{N;u}_{up;ji} = \frac{1}{2}u_{ji}\Big((1+s_{ji})v_{j+1,i-1} + (1-s_{ji})v_{j+1,i}\Big) \tag{3.4.78}$$

$$F^{N;u}_{lw;ji} = \frac{1}{2}u_{ji}\Big((1+c_{ji})v_{j+1,i-1} + (1-c_{ji})v_{j+1,i}\Big) \tag{3.4.79}$$

with s_{ji} and c_{ji} defined by (3.4.75). The flux "North" of the U-node is then determined by

$$F^{N;u}_{xy;ji} = J^u_{ji}\Big((1-\Omega(r^{N;u}_{ji}))F^{N;u}_{up;ji} + \Omega(r^{N;u}_{ji})F^{N;u}_{lw;ji}\Big) \tag{3.4.80}$$

where $\Omega(r)$ is given by one the expressions (3.4.52)–(3.4.55) selected by IADVC and

$$\begin{aligned} r^{N;u}_{ji} &= \frac{F^{N;u}_{lw;j,i-1} - F^{N;u}_{up;j,i-1}}{F^{N;u}_{lw;ji} - F^{N;u}_{up;ji}} & \text{if} \quad u_{ji} > 0 \\ r^{N;u}_{ji} &= \frac{F^{N;u}_{lw;j,i+1} - F^{N;u}_{up;j,i+1}}{F^{N;u}_{lw;ji} - F^{N;u}_{up;ji}} & \text{if} \quad u_{ji} < 0 \end{aligned} \tag{3.4.81}$$

3. The advective term in the Y-direction is computed by averaging the "South"-fluxes North of the V-node and the "North"-fluxes South of the V-node and differencing the resulting fluxes in the X-direction at the V-node

$$\mathcal{A}_{hx}(v)_{ji} = \frac{F^{S;u}_{xy;j,i+1} + F^{N;u}_{xy;j-1,i+1} - F^{S;u}_{xy;ji} - F^{N;u}_{xy;j-1,i}}{2J^v_{ji}\Delta^v x_{1;ji}} \tag{3.4.82}$$

The procedure is illustrated schematically in Figures 5.1.7c–d of the Reference Manual.

4.3.4 Discretisation of horizontal advection (2-D)

The formulation is completely analogous to the previous case. Only the following replacements need to be made in the mathematical expressions given in Section III-4.3.3.

- replace u by $\overline{U}$ and v by $\overline{V}$
- replace $(J_{ji}$,J^u_{ji},$J^v_{ji})$ by $(1/H_{ji}$,$1/H^u_{ji}$,$1/H^v_{ji})$ in (3.4.48), (3.4.58), (3.4.66), (3.4.70), (3.4.76), (3.4.80)
- omit J^u_{ji} or J^v_{ji} in the denominator of (3.4.47), (3.4.57), (3.4.72), (3.4.82)

The depth-integrated advection terms, expressed in terms of the velocity deviations $(u'$,$v')$ are obtained by integrating the 3-D advection terms in the u- and v-equation over the vertical and substracting the corresponding 2-D advection terms. This gives

$$\begin{aligned}\overline{A}^h_{1;ji} &= \overline{\mathcal{A}_{hx}(u)}_{ji} + \overline{\mathcal{A}_{hy}(u)}_{ji} - \overline{\mathcal{A}_{hx}}(\overline{U})_{ji} - \overline{\mathcal{A}_{hy}}(\overline{U})_{ji} \\ &= \sum_{k=1}^{N_z} J^u_{kji}(\mathcal{A}_{hx}(u)_{kji} + \mathcal{A}_{hy}(u)_{kji}) - \overline{\mathcal{A}_{hx}}(\overline{U})_{ji} - -\overline{\mathcal{A}_{hy}}(\overline{U})_{ji}\end{aligned} \tag{3.4.83}$$

$$\begin{aligned}\overline{A}^h_{2;ji} &= \overline{\mathcal{A}_{hx}(v)}_{ji} + \overline{\mathcal{A}_{hy}(v)}_{ji} - \overline{\mathcal{A}_{hx}}(\overline{V})_{ji} - \overline{\mathcal{A}_{hy}}(\overline{V})_{ji} \\ &= \sum_{k=1}^{N_z} J^v_{kji}(\mathcal{A}_{hx}(v)_{kji} + \mathcal{A}_{hy}(v)_{kji}) - \overline{\mathcal{A}_{hx}}(\overline{V})_{ji} - \overline{\mathcal{A}_{hy}}(\overline{V})_{ji}\end{aligned} \tag{3.4.84}$$

where all 3-D and 2-D currents are taken at the old time t^n.

4.3.5 Discretisation of horizontal diffusion (3-D)

The horizontal diffusion terms in the u- and v-equation are split up in three components. The following notations are introduced (see also Table 3.4.4):

$$\mathcal{D}_{xx}(u) = \frac{1}{J}\frac{\partial}{\partial x_1}\Big(2J\nu_H\frac{\partial u}{\partial x_1}\Big) = \frac{1}{J}\frac{\partial}{\partial x_1}D_{xx} \tag{3.4.85}$$

$$\begin{aligned}\mathcal{D}_{yx}(u,v) &= \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\nu_H\frac{\partial u}{\partial x_2}\Big) + \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\nu_H\frac{\partial v}{\partial x_1}\Big) \\ &= \frac{1}{J}\frac{\partial}{\partial x_2}D^U_{yx} + \frac{1}{J}\frac{\partial}{\partial x_2}D^V_{yx} \qquad (3.4.86)\end{aligned}$$

$$\begin{aligned}\mathcal{D}_{xy}(u,v) &= \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\nu_H\frac{\partial u}{\partial x_2}\Big) + \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\nu_H\frac{\partial v}{\partial x_1}\Big) \\ &= \frac{1}{J}\frac{\partial}{\partial x_1}D^U_{xy} + \frac{1}{J}\frac{\partial}{\partial x_1}D^V_{xy} \qquad (3.4.87)\end{aligned}$$

$$\mathcal{D}_{yy}(v) = \frac{1}{J}\frac{\partial}{\partial x_2}\Big(2J\nu_H\frac{\partial v}{\partial x_2}\Big) = \frac{1}{J}\frac{\partial}{\partial x_2}D_{yy} \qquad (3.4.88)$$

For simplicity, the k-index is omitted in the equations given below.

a) u-equation

The diffusive fluxes D_{xx}, D^U_{yx}, D^V_{yx} are evaluated at respectively the cell centres, the V-nodes and the U-nodes

$$D^c_{xx;ji} = 2J_{ji}\nu^c_{H;ji}\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} \qquad (3.4.89)$$

$$D^{U;v}_{yx;ji} = J^v_{ji}\nu^v_{H;ji}\frac{\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} \qquad (3.4.90)$$

$$D^{V;u}_{yx;ji} = J^u_{ji}\nu^u_{H;ji}\frac{\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} \qquad (3.4.91)$$

The fluxes are differenced in the X- or Y-direction as follows

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_1}D_{xx}\Big)_{ji} = \frac{D^c_{xx;ji} - D^c_{xx;j,i-1}}{J^u_{ji}\Delta^u x_{1;ji}} \qquad (3.4.92)$$

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_2}D^U_{yx}\Big)_{ji} = \frac{D^{U;v}_{yx;j+1,i} + D^{U;v}_{yx;j+1,i-1} - D^{U;v}_{yx;ji} - D^{U;v}_{yx;j,i-1}}{2J^u_{ji}\Delta^u x_{2;j}} \qquad (3.4.93)$$

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_2}D^V_{yx}\Big)_{ji} = \frac{2(D^{V;u}_{yx;j+1,i} - D^{V;u}_{yx;j-1,i})}{J^u_{ji}(\Delta^u x_{2;j-1} + \Delta^u x_{2;j+1} + 2\Delta^u x_{2;j})} \qquad (3.4.94)$$

b) v-equation

The diffusive fluxes D_{yy}, D^V_{xy} and D^U_{xy} are evaluated at respectively the cell centres, the U-nodes and the V-nodes

$$D^c_{yy;ji} = 2J_{ji}\nu^c_{H;ji}\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}} \qquad (3.4.95)$$

$$D^{V;u}_{xy;ji} = D^{V;u}_{yx;ji} \qquad (3.4.96)$$

$$D^{U;v}_{xy;ji} = D^{U;v}_{yx;ji} \tag{3.4.97}$$

The fluxes are differenced in the Y- or X-direction as follows

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_2}D_{yy}\Big)_{ji} = \frac{D^c_{yy;ji} - D^c_{yy;j-1,i}}{J^v_{ji}\Delta^v x_{2;j}} \tag{3.4.98}$$

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_1}D^V_{xy}\Big)_{ji} = \frac{D^{V;u}_{xy;j,i+1} + D^{V;u}_{xy;j-1,i+1} - D^{V;u}_{xy;ji} - D^{V;u}_{xy;j-1,i}}{2J^v_{ji}\Delta^v x_{1;ji}} \tag{3.4.99}$$

$$\Big(\frac{1}{J}\frac{\partial}{\partial x_1}D^U_{xy}\Big)_{ji} = \frac{2(D^{U;v}_{xy;j,i+1} - D^{U;v}_{xy;j,i-1})}{J^v_{ji}(\Delta^v x_{1;j,i-1} + \Delta^v x_{1;j,i+1} + 2\Delta^v x_{1;ji})} \tag{3.4.100}$$

c) horizontal diffusion coefficient

The form of the horizontal diffusion coefficient depends on the value of the switch IODIF:

0 : All horizontal diffusion terms are set to zero.

1 : The coefficient takes a constant value so that

$$\nu^c_{H;kji} = \nu^u_{H;kji} = \nu^v_{H;kji} = \nu_H \tag{3.4.101}$$

2 : The coefficient is proportional to the horizontal grid spacings and the magnitude of the deformation tensor, as given by (3.1.151)–(3.1.152):

$$\nu^c_{H;kji} = C_{m0}\Delta x_{1;ji}\Delta x_{2;j}D^c_{T;kji} \tag{3.4.102}$$

$$\nu^u_{H;kji} = C_{m0}\Delta^u x_{1;ji}\Delta^u x_{2;j}D^u_{T;kji} \tag{3.4.103}$$

$$\nu^v_{H;kji} = C_{m0}\Delta^v x_{1;ji}\Delta^v x_{2;j}D^v_{T;kji} \tag{3.4.104}$$

The spatial discretisation of the strain rate D_T involves several interpolations due to the staggering of the velocity components on the C-grid. The explicit expressions (without the k-index) are as follows

$$\big(D^c_{T;ji}\big)^2 = \Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}}\Big)^2 + \Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}}\Big)^2 + \frac{1}{8}\Big(\frac{\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j+1,i}}{\Delta^v x_{2;j+1}} + \frac{\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} + \frac{\Delta^u_x v^c_{j,i+1}}{\Delta^u x_{1;j,i+1}}\Big)^2 \tag{3.4.105}$$

$$\begin{aligned}\big(D^u_{T;ji}\big)^2 = {} & \frac{1}{4}\Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} + \frac{\Delta^c_x u_{j,i-1}}{\Delta x_{1;j,i-1}}\Big)^2 + \frac{1}{4}\Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}} + \frac{\Delta^c_y v_{j,i-1}}{\Delta x_{2;j}}\Big)^2 \\ & + \frac{1}{32}\Big(\frac{\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j,i-1}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j+1,i}}{\Delta^v x_{2;j+1}} + \frac{\Delta^v_y u^c_{j+1,i-1}}{\Delta^v x_{2;j+1}} + \frac{4\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}}\Big)^2\end{aligned} \tag{3.4.106}$$

$$\begin{aligned}\big(D^v_{T;ji}\big)^2 = {} & \frac{1}{4}\Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} + \frac{\Delta^c_x u_{j-1,i}}{\Delta x_{1;j-1,i}}\Big)^2 + \frac{1}{4}\Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}} + \frac{\Delta^c_y v_{j-1,i}}{\Delta x_{2;j-1}}\Big)^2 \\ & + \frac{1}{32}\Big(\frac{4\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} + \frac{\Delta^u_x v^c_{j-1,i}}{\Delta^u x_{1;j-1,i}} + \frac{\Delta^u_x v^c_{j,i+1}}{\Delta^u x_{1;j,i+1}} + \frac{\Delta^u_x v^c_{j-1,i+1}}{\Delta^u x_{1;j-1,i+1}}\Big)^2\end{aligned} \tag{3.4.107}$$

4.3.6 Discretisation of horizontal diffusion (2-D)

The formulation is completely analogous to the previous 3-D case. Only the following replacements need to be made in the mathematical expressions given in Section III-4.3.5:

- replace u by $\overline{U}/H$ and v by $\overline{V}/H$
- replace the Jacobians $(J_{ji}, J^u_{ji}, J^v_{ji})$ by 1
- replace the diffusion coefficient ν_H by its depth integral $\overline{\nu_H}$, or

$$\begin{aligned}\overline{\nu_H}^c_{;ji} &= \sum_{k=1}^{N_z} J_{kji}\nu^c_{H;kji} \\ \overline{\nu_H}^u_{;ji} &= \sum_{k=1}^{N_z} J^u_{kji}\nu^u_{H;kji} \\ \overline{\nu_H}^v_{;ji} &= \sum_{k=1}^{N_z} J^v_{kji}\nu^v_{H;kji}\end{aligned} \qquad (3.4.108)$$

Note that, even when ν_H is constant, $\overline{\nu_H}$ is still non-uniform since then

$$\overline{\nu_H} = \nu_H H(x_1, x_2, t) \qquad (3.4.109)$$

In analogy with advection, the depth-integrated diffusion terms, expressed in terms of the velocity deviations (u',v') are obtained by integrating the 3-D advective terms in the u- and v-equation over the vertical and substracting the corresponding 2-D diffusion terms. This gives

$$\begin{aligned}\overline{D}^h_{1;ji} &= \overline{\mathcal{D}_{xx}(u)}_{ji} + \overline{\mathcal{D}_{yx}(u,v)}_{ji} - \overline{\mathcal{D}_{xx}}(\overline{U})_{ji} - \overline{\mathcal{D}_{yx}}(\overline{U},\overline{V})_{ji} \\ &= \sum_{k=1}^{N_z} J^u_{kji}(\mathcal{D}_{xx}(u)_{kji} + \mathcal{D}_{yx}(u,v)_{kji}) - \overline{\mathcal{D}_{xx}}(\overline{U})_{ji} - \overline{\mathcal{D}_{yx}}(\overline{U},\overline{V})_{ji}\end{aligned} \qquad (3.4.110)$$

$$\begin{aligned}\overline{D}^h_{2;ji} &= \overline{\mathcal{D}_{xy}(u,v)}_{ji} + \overline{\mathcal{D}_{yy}(v)}_{ji} - \overline{\mathcal{D}_{xy}}(\overline{U},\overline{V})_{ji} - \overline{\mathcal{D}_{yy}}(\overline{U})_{ji} \\ &= \sum_{k=1}^{N_z} J^v_{kji}(\mathcal{D}_{xy}(u,v)_{kji} + \mathcal{D}_{yy}(v)_{kji}) - \overline{\mathcal{D}_{xy}}(\overline{U},\overline{V})_{ji} - \overline{\mathcal{D}_{yy}}(\overline{V})_{ji}\end{aligned} \qquad (3.4.111)$$

where all 3-D and 2-D currents are taken at the old time t^n.

4.3.7 Discretisation of vertical advection

The vertical advection terms are written as (see Table 3.4.4):

$$\mathcal{A}_v(u) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{\omega}u) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}F_{xz} \qquad (3.4.112)$$

$$\mathcal{A}_v(v) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{\omega}v) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}F_{yz} \qquad (3.4.113)$$

a) u-equation

The fluxes F_{xz} are evaluated at the X-nodes (see Figure 3.4.1) so that

$$\mathcal{A}_v(u)_{kji} = \frac{F^X_{xz;k+1,ji} - F^X_{xz;kji}}{\Delta^u x_{3;kji}} \tag{3.4.114}$$

where it is recalled that $\Delta x_3 = J\Delta\tilde{x}_3 = J$.

The vertical flux is defined by a linear combination of the upwind and central fluxes

$$F^X_{xz;kji} = (1 - \Omega(r^X_{kji}))F^X_{up;kji} + \Omega(r^X_{kji})F^X_{ce;kji} \tag{3.4.115}$$

where F^X_{up} and F^X_{ce} are given by

$$F^X_{up;kji} = \frac{1}{2}(J\tilde{w})^X_{kji}\Big((1 + s_{kji})u_{k-1,ji} + (1 - s_{kji})u_{kji}\Big) \tag{3.4.116}$$

$$F^X_{ce;kji} = \frac{1}{2}(J\tilde{w})^X_{kji}(u_{k-1,ji} + u_{kji}) \tag{3.4.117}$$

and

$$(J\tilde{w})^X_{kji} = \frac{1}{2}\Big((J\tilde{w})_{kj,i-1} + (J\tilde{w})_{kji}\Big) \tag{3.4.118}$$

denotes the vertical current interpolated horizontally at the X-node, and

$$s_{kji} = \text{Sign}((J\tilde{w})^X_{kji}) \tag{3.4.119}$$

is the sign of the advecting current.

The limiter function $\Omega(r)$ is determined using one of the formulations (3.4.52)–(3.4.55) selected by the switch IADVC. The argument r depends on the sign of the advecting current.

$$\begin{aligned} r^X_{kji} &= \frac{F^X_{ce;k-1,ji} - F^X_{up;k-1,ji}}{F^X_{ce;kji} - F^X_{up;kji}} \quad &\text{if} \quad (J\tilde{w})^X_{kji} > 0 \\ r^X_{kji} &= \frac{F^X_{ce;k+1,ji} - F^X_{up;k+1,ji}}{F^X_{ce;kji} - F^X_{up;kji}} \quad &\text{if} \quad (J\tilde{w})^X_{kji} < 0 \end{aligned} \tag{3.4.120}$$

Important to note is that, since r^X_{kji} is taken at the old time t^n whereas the vertical advection term is evaluated semi-implicitly (depending on the value of the implicity factor θ_a), the upwind and central fluxes have to be calculated both at the old and the new time level (see Section III-4.3.12b for implementation in the program).

b) v-equation

The procedure is analogous to the previous case. The fluxes F_{yz} are evaluated at the Y-nodes (see Figure 3.4.1) so that

$$\mathcal{A}_v(v)_{kji} = \frac{F^Y_{yz;k+1,ji} - F^Y_{yz;kji}}{\Delta^v x_{3;kji}} \tag{3.4.121}$$

where it is recalled that $\Delta x_3 = J\Delta\tilde{x}_3 = J$.

The vertical flux is defined by a linear combination of the upwind and central fluxes

$$F^Y_{yz;kji} = (1-\Omega(r^Y_{kji}))F^Y_{up;kji} + \Omega(r^Y_{kji})F^Y_{ce;kji} \tag{3.4.122}$$

where F^Y_{up} and F^Y_{ce} are given by

$$F^Y_{up;kji} = \frac{1}{2}(J\tilde{w})^Y_{kji}\Big((1+s_{kji})v_{k-1,ji} + (1-s_{kji})v_{kji}\Big) \tag{3.4.123}$$

$$F^Y_{ce;kji} = \frac{1}{2}(J\tilde{w})^Y_{kji}(v_{k-1,ji} + v_{kji}) \tag{3.4.124}$$

and

$$(J\tilde{w})^Y_{kji} = \frac{1}{2}\Big((J\tilde{w})_{k,j-1,i} + (J\tilde{w})_{kji}\Big) \tag{3.4.125}$$

denotes the vertical current interpolated horizontally at the Y-node, and

$$s_{kji} = \mathrm{Sign}((J\tilde{w})^Y_{kji}) \tag{3.4.126}$$

is the sign of the advecting current.

The limiter function $\Omega(r)$ is determined using one of the formulations (3.4.52)–(3.4.55) selected by the switch IADVC. The argument r depends on the sign of the advecting current.

$$\begin{aligned} r^Y_{kji} &= \frac{F^Y_{ce;k-1,ji} - F^Y_{up;k-1,ji}}{F^Y_{ce;kji} - F^Y_{up;kji}} \qquad \text{if} \quad (J\tilde{w})^Y_{kji} > 0 \\ r^Y_{kji} &= \frac{F^Y_{ce;k+1,ji} - F^Y_{up;k+1,ji}}{F^Y_{ce;kji} - F^Y_{up;kji}} \qquad \text{if} \quad (J\tilde{w})^Y_{kji} < 0 \end{aligned} \tag{3.4.127}$$

As stated in Section III-4.3.7a, r^Y_{kji} is taken at the old time whereas the upwind and central schemes are evaluated at the old and new time.

4.3.8 Discretisation of vertical diffusion

The vertical diffusion terms are written in the form

$$\mathcal{D}_v(u) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial u}{\partial\tilde{x}_3}\Big) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}D_{xz} \tag{3.4.128}$$

$$\mathcal{D}_v(v) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big(\frac{\nu_T}{J}\frac{\partial v}{\partial\tilde{x}_3}\Big) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}D_{yz} \tag{3.4.129}$$

The vertical fluxes D_{xz} and D_{yz} are taken at respectively the X- and Y-nodes. The discretised forms of (3.4.128)–(3.4.129) are then

$$\mathcal{D}_v(u)_{kji} = \frac{D^X_{xz;k+1,ji} - D^X_{xz;kji}}{\Delta^u x_{3;kji}} \tag{3.4.130}$$

$$\mathcal{D}_v(v)_{kji} = \frac{D^Y_{yz;k+1,ji} - D^Y_{yz;kji}}{\Delta^v x_{3;kji}} \tag{3.4.131}$$

The fluxes are computed by

$$D^X_{xz;kji} = \nu^X_{T;kji} \frac{\Delta^X_z u_{kji}}{\Delta^X x_{3;kji}} \tag{3.4.132}$$

$$D^Y_{yz;kji} = \nu^Y_{T;kji} \frac{\Delta^Y_z v_{kji}}{\Delta^Y x_{3;kji}} \tag{3.4.133}$$

where

$$\nu^X_{T;kji} = \frac{1}{2}(\nu_{T;kj,i-1} + \nu_{T;kji}) \tag{3.4.134}$$

$$\Delta^X x_{3;kji} = \frac{1}{4}(J_{kji} + J_{kj,i-1} + J_{k-1,ji} + J_{k-1,j,i-1}) \tag{3.4.135}$$

$$\nu^Y_{T;kji} = \frac{1}{2}(\nu_{T;k,j-1,i} + \nu_{T;kji}) \tag{3.4.136}$$

$$\Delta^Y x_{3;kji} = \frac{1}{4}(J_{kji} + J_{k,j-1,i} + J_{k-1,ji} + J_{k-1,j-1,i}) \tag{3.4.137}$$

are the values of ν_T and Δx_3 interpolated at respectively the X- and Y-nodes.

4.3.9 Discretisation of the baroclinic pressure gradient

The components of the baroclinic pressure gradient are evaluated along the following steps.

1. The density ρ is calculated using either the linear state equation or the general equation of state of seawater described in Section III-1.5.

2. The buoyancy is determined by

$$b = -g(\frac{\rho}{\rho_0} - 1) \tag{3.4.138}$$

3. The equation of hydrostatic equilibrium is integrated for q_d:

$$\begin{aligned} q^w_{d;N_z+1,ji} &= 0 \\ q^w_{d;kji} &= q^w_{d;k+1,ji} - b_{kji}J_{kji} \end{aligned} \tag{3.4.139}$$

   Note that q_d is taken at the W-nodes.

4. The X- and Y-components of the baroclinic pressure gradient, as defined by (3.1.30), are the sum of two terms. The following notations are introduced:

$$Q_1 = -\frac{1}{J}\frac{\partial}{\partial x_1}(Jq_d) - \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}G_1 \tag{3.4.140}$$

$$Q_2 = -\frac{1}{J}\frac{\partial}{\partial x_2}(Jq_d) - \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}G_2 \tag{3.4.141}$$

where

$$G_1 = -q_d(\sigma\frac{\partial H}{\partial x_1} - \frac{\partial h}{\partial x_1})\,,\; G_2 = -q_d(\sigma\frac{\partial H}{\partial x_2} - \frac{\partial h}{\partial x_2}) \tag{3.4.142}$$

The discretised forms of (3.4.140)–(3.4.142) are

$$G^X_{1;kji} = -\frac{q^X_d(\sigma_k\Delta^X_x H_{ji} - \Delta^X_x h_{ji})}{\Delta^X x_{1;ji}} \tag{3.4.143}$$

$$G^Y_{2;kji} = -\frac{q^Y_d(\sigma_k\Delta^Y_y H_{ji} - \Delta^Y_y h_{ji})}{\Delta^Y x_{2;j}} \tag{3.4.144}$$

$$Q^u_{1;kji} = -\frac{1}{J^u_{kji}}\Big(\frac{q^c_{d;kji}J_{kji} - q^c_{d;kj,i-1}J_{kj,i-1}}{\Delta^u x_{1;ji}} + G^X_{1;k+1,ji} - G^X_{1;kji}\Big) \tag{3.4.145}$$

$$Q^v_{2;kji} = -\frac{1}{J^v_{kji}}\Big(\frac{q^c_{d;kji}J_{kji} - q^c_{d;k,j-1,i}J_{k,j-1,i}}{\Delta^v x_{2;j}} + G^Y_{2;k+1,ji} - G^Y_{2;kji}\Big) \tag{3.4.146}$$

5. The depth-integrated components are then given by

$$\overline{Q}^u_{1;ji} = \sum_{k=1}^{N_z} J^u_{kji}Q^u_{1;kji}\,,\; \overline{Q}^v_{2;ji} = \sum_{k=1}^{N_z} J^v_{kji}Q^v_{2;kji} \tag{3.4.147}$$

4.3.10 Boundary conditions (3-D)

a) surface and bottom boundaries

- The Neumann conditions (3.1.168) and (3.1.183) are applied at respectively the surface and bottom. The numerical implementation is further discussed in Section III-4.3.12.
- The vertical fluxes for advection are set to zero at the surface and the bottom

$$F^X_{xz;N_z+1,ji} = 0\,,\; F^Y_{yz;N_z+1,ji} = 0\,,\; F^X_{xz;1ji} = 0\,,\; F^Y_{yz;1ji} = 0 \tag{3.4.148}$$

in accordance with the conditions (3.1.169) and (3.1.188).

b) land boundaries

The following quantities are set to zero at land boundaries:

- the horizontal currents u and v
- the advective fluxes F^W_{yx}, F^E_{yx} at V-boundaries and F^S_{xy} and F^N_{xy} at U-boundaries
- the diffusive fluxes D^V_{yx}, D^V_{xy} at U-boundaries and D^U_{yx}, D^U_{xy} at V-boundaries

c) open boundaries

As discussed in Section III-1.6.3b two types of conditions can be used

1. The zero gradient conditions (3.1.208) or (3.1.209) are applied. The discretised forms are:

$$u_{kji} = \frac{u_{kj,i+1:i-1}J^u_{kj,i+1:i-1}}{J_{kj,i:i-1}} + \frac{\overline{U}_{ji} - \sum_{k=1}^{N_z} u_{kj,i+1:i-1}J^u_{kj,i+1:i-1}}{H_{j,i:i-1}} \tag{3.4.149}$$

$$v_{kji} = \frac{v_{k,j+1:j-1,i}J^v_{k,j+1:j-1,i}}{J_{k,j:j-1,i}} + \frac{\overline{V}_{ji} - \sum_{k=1}^{N_z} v_{k,j+1:j-1,i}J^v_{k,j+1:j-1,i}}{H_{j:j-1,i}} \tag{3.4.150}$$

   The meaning of the double index notation $i_1 : i_2$ is explained in Section III-4.2.2.

2. The external profile u'_0 or v'_0 is prescribed for the velocity deviation u' or v' (see equation (3.1.210)). This gives

$$u_{kji} = u'_{0;kji} + \frac{\overline{U}_{ji}}{H_{j,i:i-1}} \tag{3.4.151}$$

$$v_{kji} = v'_{0;kji} + \frac{\overline{V}_{ji}}{H_{j:j-1,i}} \tag{3.4.152}$$

The boundary values of (u^F,v^F) are obtained from (3.4.149)–(3.4.152) after replacing ($\overline{U}$,$\overline{V}$) by ($\overline{U}^F$,$\overline{V}^F$).

d) advective and diffusion fluxes

- The advective fluxes $F^{W;v}_{yx}$, $F^{E;v}_{yx}$ at V-open boundaries and $F^{S;u}_{xy}$, $F^{N;u}_{xy}$ at U-open boundaries, which are used to evaluate the cross-derivative terms in the momentum equations (see Sections III-4.3.3c–d), are determined using the upwind scheme in the case of outflow whereas a zero normal gradient conditions is applied in the case of inflow. Hence,

$$\begin{aligned} F^{W;v}_{yx;ji} &= J_{j:j-1,i}v_{ji}u_{j:j-1,i} && \text{if} \quad pv_{ji} < 0 \\ F^{W;v}_{yx;ji} &= F^{W;v}_{yx;j+1:j-1,i} && \text{if} \quad pv_{ji} > 0 \end{aligned} \tag{3.4.153}$$

$$\begin{aligned} F^{E;v}_{yx;ji} &= J_{j:j-1,i}v_{ji}u_{j:j-1,i+1} && \text{if} \quad pv_{ji} < 0 \\ F^{E;v}_{yx;ji} &= F^{E;v}_{yx;j+1:j-1,i} && \text{if} \quad pv_{ji} > 0 \end{aligned} \tag{3.4.154}$$

$$\begin{aligned} F^{S;u}_{xy;ji} &= J_{j,i:i-1}u_{ji}v_{j,i:i-1} && \text{if} \quad pu_{ji} < 0 \\ F^{S;u}_{xy;ji} &= F^{S;u}_{xy;j,i+1:i-1} && \text{if} \quad pu_{ji} > 0 \end{aligned} \tag{3.4.155}$$

$$\begin{aligned} F^{N;u}_{xy;ji} &= J_{j,i:i-1}u_{ji}v_{j+1,i:i-1} \qquad \text{if} \qquad pu_{ji} < 0 \\ F^{N;u}_{xy;ji} &= F^{N;u}_{xy;j,i+1;i-1} \qquad \text{if} \qquad pu_{ji} > 0 \end{aligned} \tag{3.4.156}$$

where $p = +1$ at West and South and $p = -1$ at East and North open boundaries.

- A zero normal gradient condition is applied for the diffusive fluxes D^U_{yx}, D^U_{xy} at V-open boundaries and D^V_{yx}, D^V_{xy} at U-open boundaries

$$D^{U;v}_{yx;ji} = D^{U;v}_{xy;ji} = D^{U;v}_{yx;j+1:j-1,i} \tag{3.4.157}$$

$$D^{V;u}_{yx;ji} = D^{V;u}_{xy;ji} = D^{V;u}_{yx;j,i+1:i-1} \tag{3.4.158}$$

- To avoid the computation of fluxes outside the domain or spurious discontinuities near open and land boundaries, the TVD scheme is replaced by the upwind scheme ($\Omega(r) = 0$) in the following cases:
    - The central fluxes F^c_{xx}, F^c_{yy} are evaluated at cells adjacent to the domain boundary and an inflow condition occurs.
    - The central fluxes F^c_{xx}, F^c_{yy} are evaluated at cells adjacent to a dry cell and the advecting current is pointing away from the land boundary.
    - The fluxes (F^W_{yx},F^E_{yx}) or (F^S_{xy},F^N_{xy}) are evaluated one grid distance away from a land/open boundary in the Y- or X-direction and the advecting current is pointing away from this boundary.

Since $\Omega(0) = 0$, the following conditions are imposed:

$$r^{U;c}_{ji} = 0 \quad \begin{aligned} &\text{if } u_{ji} > 0 \text{ and either } i = 1 \text{ or the cell } (j,i-1) \text{ is dry} \\ &\text{or } u_{ji} < 0 \text{ and either } i = N_x \text{ or the cell } (j,i+1) \text{ is dry} \end{aligned} \tag{3.4.159}$$

$$r^{V;c}_{ji} = 0 \quad \begin{aligned} &\text{if } v_{ji} > 0 \text{ and either } j = 1 \text{ or the cell } (j-1,i) \text{ is dry} \\ &\text{or } v_{ji} < 0 \text{ and either } j = N_y \text{ or the cell } (j+1,i) \text{ is dry} \end{aligned} \tag{3.4.160}$$

$$r^{W;v}_{ji} = r^{E;v}_{ji} = 0 \quad \begin{aligned} &\text{if } v_{ji} > 0 \text{ and the V-node } (j-1,i) \text{ is an open or land cell face} \\ &\text{or } v_{ji} < 0 \text{ and the V-node } (j+1,i) \text{ is an open or land cell face} \end{aligned} \tag{3.4.161}$$

$$r^{S;u}_{ji} = r^{N;u}_{ji} = 0 \quad \begin{aligned} &\text{if } u_{ji} > 0 \text{ and the U-node } (j,i-1) \text{ is an open or land cell face} \\ &\text{or } u_{ji} < 0 \text{ and the U-node } (j,i+1) \text{ is an open or land cell face} \end{aligned} \tag{3.4.162}$$

4.3.11 Boundary conditions (2-D)

a) land boundaries

The depth-integrated currents $\overline{U}$ and $\overline{V}$ and the advective/diffusive fluxes are set to zero at land boundaries.

b) open boundaries

As explained in Section III-1.6.3a the open boundary values for the depth-integrated current are determined with the aid of the method of Riemann characteristics. The method consists in deriving the values of the incoming and outgoing Riemann characteristics $R^u_\pm$, $R^v_\pm$ defined by (3.1.193). The currents are then obtained by taking the average of the incoming and outgoing variables.

The outgoing characteristic is computed by integrating equations (3.1.194) or (3.1.195). The discretised forms are

$$\begin{aligned}\frac{R^{u;m+1}_{\mp;i} - R^{u;m}_{\mp;i}}{\Delta t_{2D}} &= -2c^n_{i:i-1}\frac{R^{u;m}_{\mp;i} - \overline{U}^{c;m+1}_{i:i-1} \pm c^n_{i:i-1}\zeta^{m+1}_{i:i-1}}{\Delta x_{1;i:i-1}} \pm c^n_{i:i-1}\frac{\Delta^c_y\overline{V}_{i:i-1}}{\Delta x_2} + f\overline{V}^{u;m+1}_{ji} \\ &\quad -\overline{\mathcal{A}_h}(\overline{U}^m)_{i+1:i-1} - \frac{H^n_{i+1:i-1}}{\rho_0}\frac{\Delta^u_x P_{a;i+1:i-1}}{\Delta^u x_{1;i+1:i-1}} + \overline{Q}^n_{1;i+1:i-1} + \frac{1}{\rho_0}(\tau_{s1;i:i-1} - k^{u;p}_{b;i+1:i-1}u^p_{b;i+1:i-1}) \\ &\quad +\overline{\mathcal{D}_{xx}}(\overline{U}^m)_{i+1:i-1} + \overline{\mathcal{D}_{yx}}(\overline{U}^m,\overline{V}^m)_{i+1:i-1} - \overline{A}^{h;n}_{1;i+1:i-1} + \overline{D}^{h;n}_{1;i+1:i-1} \qquad (3.4.163)\end{aligned}$$

$$\begin{aligned}\frac{R^{v;m+1}_{\mp;j} - R^{v;m}_{\mp;j}}{\Delta t_{2D}} &= -2c^n_{j:j-1}\frac{R^{v;m}_{\mp;j} - \overline{V}^{c;m+1}_{j:j-1} \pm c^n_{j:j-1}\zeta^{m+1}_{j:j-1}}{\Delta x_{2;j:j-1}} \pm c^n_{j:j-1}\frac{\Delta^c_x\overline{U}_{j:j-1}}{\Delta x_{1;j:j-1}} - f\overline{U}^{v;m+1}_{ji} \\ &\quad -\overline{\mathcal{A}_h}(\overline{V}^m)_{j+1:j-1} - \frac{H^n_{j+1:j-1}}{\rho_0}\frac{\Delta^v_y P_{a;j+1:j-1}}{\Delta^v x_{2;j+1:j-1}} + \overline{Q}^n_{2;j+1:j-1} + \frac{1}{\rho_0}(\tau_{s2;j:j-1} - k^{v;p}_{b;j+1:j-1}v^p_{b;j+1:j-1}) \\ &\quad +\overline{\mathcal{D}_{xy}}(\overline{U}^m,\overline{V}^m)_{j+1:j-1} + \overline{\mathcal{D}_{yy}}(\overline{V}^m)_{j+1:j-1} - \overline{A}^{h;n}_{2;j+1:j-1} + \overline{D}^{h;n}_{2;j+1:j-1} \qquad (3.4.164)\end{aligned}$$

where

$$c^n_{ji} = (gH^n_{ji})^{1/2} \qquad (3.4.165)$$

is the barotropic wave speed, and

$$u_{b;ji} = u_{1ji}\,,\; v_{b;ji} = v_{1ji} \qquad (3.4.166)$$

the values of the 3-D current at the bottom grid cell. For simplicity, the index j has been omitted in (3.4.163) and the index i in (3.4.164). Note that in (3.4.163)–(3.4.164) and in the equations below the upper sign applies at western or southern boundaries and the lower sign at eastern or northern boundaries.

The incoming characteristic is determined either using a zero normal gradient condition or from input data written in the harmonic form

$$F^{u;m+1}_{har;ji} = A^u_{0;ji} + \sum_{k=1}^{N_h} A^u_{k;ji}\cos(\omega_k t + \varphi_{k0} - \varphi^u_{k;ji}) \qquad (3.4.167)$$

at U-boundaries, and

$$F^{v;m+1}_{har;ji} = A^v_{0;ji} + \sum_{k=1}^{N_h} A^v_{k;ji}\cos(\omega_k t + \varphi_{k0} - \varphi^v_{k;ji}) \qquad (3.4.168)$$

at V-boundaries where $t = (nM_t + m + 1)\Delta t_{2D}$ is the time elapsed since the start of the simulation. The meaning of the harmonic function F_{har} depends on the type of boundary condition. Its value is given by one of the equations (3.4.170), (3.4.172), (3.4.174) below. The residuals A_0 and the harmonic parameters A_k, ω_k, φ_{k0}, φ_k are user-defined. The incoming variable is obtained using one of the four following formulations[6].

1. Input data are supplied for the 2-D current ($\overline{U}_{in}$,$\overline{V}_{in}$) and the surface elevation (ζ^u_{in},ζ^v_{in}):

$$R^{u;m+1}_{\pm;i} = \pm 2c^n_{i:i-1}F^{u;m+1}_{har;i}\ ,\ R^{v;m+1}_{\pm;j} = \pm 2c^n_{j:j-1}F^{v;m+1}_{har;j} \tag{3.4.169}$$

   where

$$F^{u;m+1}_{har;i} = \frac{1}{2}(\pm\frac{\overline{U}^{m+1}_{in;i}}{\tilde{c}_{i:i-1}} + \zeta^{u;m+1}_{in;i})\ ,\ F^{v;m+1}_{har;j} = \frac{1}{2}(\pm\frac{\overline{V}^{m+1}_{in;j}}{\tilde{c}_{j:j-1}} + \zeta^{v;m+1}_{in;j}) \tag{3.4.170}$$

2. A zero gradient condition is selected in which case (3.4.163) or (3.4.164) is solved for the outgoing variable without the first term on the right hand side and using the lower sign at western or southern boundaries and the upper sign at eastern or northern boundaries. No harmonic expansion is required in this case.

3. Input data are only supplied for the surface elevation:

$$R^{u;m+1}_{\pm;i} = \pm 2c^n_{i:i-1}F^{u;m+1}_{har;i} + R^{u;m+1}_{\mp;i}\ ,\ R^{v;m+1}_{\pm;j} = \pm 2c^n_{j:j-1}F^{v;m+1}_{har;j} + R^{v;m+1}_{\mp;j} \tag{3.4.171}$$

   where

$$F^{u;m+1}_{har;i} = \zeta^{u;m+1}_{in;i}\ ,\ F^{v;m+1}_{har;j} = \zeta^{v;m+1}_{in;j} \tag{3.4.172}$$

4. Input data are only supplied for the 2-D current:

$$R^{u;m+1}_{\pm;i} = \pm 2\tilde{c}^n_{i:i-1}F^{u;m+1}_{har;i} - R^{u;m+1}_{\mp;i}\ ,\ R^{v;m+1}_{\pm;j} = \pm 2\tilde{c}^n_{j:j-1}F^{v;m+1}_{har;j} - R^{v;m+1}_{\mp;j} \tag{3.4.173}$$

   where

$$F^{u;m+1}_{har;i} = \pm\frac{\overline{U}^{m+1}_{in;i}}{\tilde{c}_{i:i-1}}\ ,\ F^{v;m+1}_{har;j} = \pm\frac{\overline{V}^{m+1}_{in;j}}{\tilde{c}_{j:j-1}} \tag{3.4.174}$$

The right hand side of (3.4.170), (3.4.172), (3.4.174) must be supplied by the user in harmonic form as part of the model setup (see Section IV-2.4). Since the total water depth H is unknown when the model is set up, the barotropic wave speed is aproximated by replacing H with the mean water depth h in F_{har}, i.e.

$$\tilde{c}_{ji} = (gh_{ji})^{1/2} \tag{3.4.175}$$

[6]Note that ζ_{in} must be supplied at the U- or V-open boundary itself and not at the cell centre.

c) advective and diffusion fluxes

The conditions are analogous as the ones used for the 3-D current.

- The advective fluxes at the open boundaries used to evaluate the cross-derivative terms in the 2-D momentum equations, are evaluated with the upwind scheme in the case of outflow whereas a zero normal gradient condition is applied in the case of inflow.
- A zero normal gradient condition is considered for the 2-D diffusive fluxes in the cross-derivative terms.
- To avoid the computation of fluxes outside the domain or spurious discontinuities near open or land boundaries the TVD scheme is replaced by the upwind scheme in a number of cases.

For more details see Section III-4.3.10d and the equations therein.

4.3.12 Tridiagonal matrix system

The discretised forms of the u-equations (3.4.9), (3.4.28), (3.4.29) or the v-equations (3.4.10), (3.4.37), (3.4.38) reduce to a system of linear equations which has the following tridiagonal form

$$\begin{aligned} B_{1ji}u^p_{1ji} + C_{1ji}u^p_{2ji} &= D_{1ji} \\ A_{kji}u^p_{k-1,ji} + B_{kji}u^p_{kji} + C_{kji}u^p_{k+1,ji} &= D_{kji} \\ A_{N_z,ji}u^p_{N_z-1,ji} + B_{N_z,ji}u^p_{N_z,ji} = D_{N_z,ji} & \end{aligned} \tag{3.4.176}$$

where $2 \leq k \leq N_z - 1$ and all explicit terms are written in the D-term on the right hand side. In view of the high similarity between the u- and v-equation only the former case will be considered here.

Each of the vertical vectors A, B, C, D is the sum of various contributions, marked by a superscript, which will be discussed separately below. The horizontal indices j and i are omitted for simplicity.

a) time derivative

The contribution of the time derivative is given by

$$A^t_k = 0\,,\; B^t_k = 1\,,\; C^t_k = 0\,,\; D^t_k = u^n_k \tag{3.4.177}$$

where $1 \leq k \leq N_z$.

b) vertical advection

The vertical advection term is split up into two contributions arising from the fluxes below and above the k-level. The former are given by

$$\begin{aligned}
A_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1+f_k^X) \\
B_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1-f_k^X) \\
D_k^{a-} &= \frac{1}{2}(1-\theta_a)c_k^-[(1+f_k^X)u_{k-1}^n+(1-f_k^X)u_k^n]
\end{aligned} \tag{3.4.178}$$

where $2 \le k \le N_z$,

$$f_k^X = (1-\Omega(r_k^X))s_k \tag{3.4.179}$$

$$c_k^- = \frac{\Delta t_{3D}(J\tilde{w})_k^X}{\Delta^u x_{3;k}} \tag{3.4.180}$$

and r_k^X is defined by (3.4.120) using the value u^n at the old time level.

The terms arising from the flux above the k-level, are

$$\begin{aligned}
B_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1+f_{k+1}^X) \\
C_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1-f_{k+1}^X) \\
D_k^{a+} &= -\frac{1}{2}(1-\theta_a)c_k^+[(1+f_{k+1}^X)u_k^n+(1-f_{k+1}^X)u_{k+1}^n]
\end{aligned} \tag{3.4.181}$$

where $1 \le k \le N_z - 1$ and

$$c_k^+ = \frac{\Delta t_{3D}(J\tilde{w})_{k+1}^X}{\Delta^u x_{3;k}} \tag{3.4.182}$$

c) vertical diffusion

As for vertical advection the fluxes below and above the k-level are taken separately. The former are given by

$$\begin{aligned}
A_k^{d-} &= -\theta_v \frac{\Delta t_{3D}\nu_{T;k}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k}} \\
B_k^{d-} &= \theta_v \frac{\Delta t_{3D}\nu_{T;k}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k}} \\
D_k^{d-} &= -(1-\theta_v)\frac{\Delta t_{3D}\nu_{T;k}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k}}(u_k^n - u_{k-1}^n)
\end{aligned} \tag{3.4.183}$$

where $2 \le k \le N_z$.

The terms taken from the flux above the k-level, are

$$\begin{aligned} B_k^{d+} &= \theta_v \frac{\Delta t_{3D}\nu_{T;k+1}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k+1}} \\ C_k^{d+} &= -\theta_v \frac{\Delta t_{3D}\nu_{T;k+1}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k+1}} \\ D_k^{d+} &= (1-\theta_v)\frac{\Delta t_{3D}\nu_{T;k+1}^X}{\Delta^u x_{3;k}\Delta^X x_{3;k+1}}(u_{k+1}^n - u_k^n) \end{aligned} \tag{3.4.184}$$

where $1 \leq k \leq N_z - 1$.

d) surface and bottom boundary conditions

The contributions arising from the surface and bottom boundary conditions (3.1.168) and (3.1.183) are added to the highest respectively the lowest vertical level

$$B_1^C = \frac{\Delta t_{3D} k_b^{u;n}}{\Delta^u x_{3;1}}\ ,\ D_{N_z}^C = \frac{\Delta t_{3D}\tau_{s1}^u}{\rho_0 \Delta^u x_{3;N_z}} \tag{3.4.185}$$

e) other explicit terms

The other explicit terms are given by

$$D_k^e = \Big(\mathcal{O}_{x;k}^n - \mathcal{A}_{hx}(u^n)_k - \mathcal{A}_{hy}(u^n)_k + \mathcal{D}_{xx}(u^n)_k + \mathcal{D}_{yx}(u^n, v^n)_k\Big)\Delta t_{3D} \tag{3.4.186}$$

when the equations are solved without operator splitting, or

$$D_k^e = \mathcal{O}_{x;k}^n \Delta t_{3D} \tag{3.4.187}$$

if operator splitting is applied and where $\mathcal{O}_x^n$ is defined by (3.4.34).

f) solution of the linear system

The linear system (3.4.176) is solved using the algorithm, recommended by Press et al. (1989)

$$\begin{aligned} \beta_1 &= B_1 \\ \tilde{u}_1^p &= D_1/\beta_1 \end{aligned} \tag{3.4.188}$$

$$\left.\begin{aligned} &\gamma_k = C_{k-1}/\beta_{k-1} \\ &\beta_k = B_k - A_k\gamma_k \\ &\tilde{u}_k^p = (D_k - A_k\tilde{u}_{k-1}^p)/\beta_k \end{aligned}\right\} k = 2, \cdots, N_z \tag{3.4.189}$$

$$\begin{aligned} &u_{N_z}^p = \tilde{u}_{N_z}^p \\ &u_k^p = \tilde{u}_k^p - \gamma_{k+1}u_{k+1}^p\ ; \quad k = N_z - 1, \cdots, 1 \end{aligned} \tag{3.4.190}$$

4.4 Scalar transport equations

This section describes the numerical solution of the transport equation (3.1.70) for a scalar quantity ψ evaluated at the cell centre, where ψ stands for temperature, salinity, a biological state variable, inorganic sediment concentration or a contaminant distribution. Transport of turbulence variables, which are located at the W-nodes, will be discussed in Section III-4.5.

The basic features are:

- Horizontal advection and diffusion terms are discretised explicitly in time.
- In analogy with the momentum equations vertical advection is taken semi-implicitly while vertical diffusion is treated fully implicitly. The equations for vertical advection and diffusion are presented here in a more general form covering both the explicit, implicit and semi-implicit cases.
- The vertical sinking rate term is written into the conservative form

$$\frac{w_s^\psi}{J}\frac{\partial\psi}{\partial\tilde{x}_3} \cong \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(w_s^\psi\psi) \qquad (3.4.191)$$

  Although this procedure is only applicable if w_s^ψ is vertically uniform, the formulation prevents the creation of extra sources and sinks in the vertical. To simplify the notations a new vertical advection operator $\tilde{\mathcal{A}}_v$ and vertical velocity W are defined by

$$\tilde{\mathcal{A}}_v(\psi) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(J\tilde{\omega}\psi) + \frac{w_s^\psi}{J}\frac{\partial\psi}{\partial\tilde{x}_3} \cong \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}\Big((J\tilde{\omega} + w_s^\psi)\psi\Big) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(W\psi) \qquad (3.4.192)$$

- As recommended by Ruddick (1995), the Jacobian J is eliminated from the time derivative (except in the absence of advection) by adding the corrector terms $\mathcal{M}_x$, $\mathcal{M}_y$, $\mathcal{M}_z$ to the right hand side of the transport equation. This is further discussed in Section III-4.4.1b below.
- Source terms are discretised explicitly. Contrary to the momentum and turbulence transport equations the sink terms are evaluated explicitly. This affects only the biological part of the program and has been introduced to conserve the processes involving nitrogen.
- The advecting current (u^F,v^F) used for horizontal advection is composed of the "predictor" current plus a filtered depth-independent component obtained by averaging over the more rapidly varying 2-D mode. As shown by Deleersnijder (1993) this method ensures mass conservation.
- The transport equation is integrated in time with or without the operator splitting method depending on the type of advection scheme. Note that the program allows to use different advection schemes in the momentum and scalar transport modules.

4.4.1 Time discretisation

In analogy with momentum (see Section III-4.3.2) the time discretisation of a scalar transport equation depends on the type of advection scheme selected in the program. Several schemes are available. The type is selected with the model switch IADVS which, in analogy with the switch IADVC for momentum, has the following meaning.

0 : horizontal and vertical advection disabled[7]

1 : upwind scheme for horizontal and vertical advection

2 : Lax-Wendroff scheme for horizontal, central scheme for vertical advection[8]

3 : TVD (Total Variation Diminishing) scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical

4 : as the previous case now using the monotonic limiter.

A general description of the different schemes is given in Section III-4.3.2. Three cases can be distinguished for the time integration. They are discussed in the subsections below.

a) integration without (physical) advection

In the absence of physical advection (IADVS $= 0$) the transport equation is integrated in time using

$$\begin{aligned}\frac{J^{n+1}\psi^{n+1}-J^n\psi^n}{J^{n+1}\Delta t_{3D}} &= -\ \theta_a\tilde{\mathcal{A}}_v(\psi^{n+1})-(1-\theta_a)\tilde{\mathcal{A}}_v(\psi^n)+\theta_v\mathcal{D}_v(\psi^{n+1})\\ &+\ (1-\theta_v)\mathcal{D}_v(\psi^n)+\mathcal{D}_h(\psi^n)+\beta(\psi^n)\end{aligned}\qquad(3.4.193)$$

where

$$\beta(\psi)=\mathcal{P}(\psi)-\mathcal{S}(\psi)\qquad(3.4.194)$$

represents the sum of all source and sinks. The vertical advection term in the transport equations for the biological and iSPM concentrations is given either by

$$\tilde{\mathcal{A}}_v(\psi)=\frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(w_s^\psi\psi)\qquad(3.4.195)$$

or

$$\tilde{\mathcal{A}}_v(\psi)=0\qquad(3.4.196)$$

depending on the value of the switch IADVWB (see Section III-4.4.9).

[7]Note that, if IADVS $= 0$, vertical sinking can still be enabled using the switch IADVWB. This is further discussed in Section III-4.4.9.

[8]The "pure" Lax-Wendroff scheme has only been implemented for illustrative purposes and should be avoided in realistic simulations (see Chapter II-2).

b) integration with advection but without operator splitting

This applies for the case when horizontal and vertical advection are discretised with the upwind scheme (IADVS = 1). The time derivative is first rewritten in the following form with the aid of the continuity equation (3.1.24):

$$\begin{aligned}\frac{1}{J}\frac{\partial}{\partial t}(J\psi) &= \frac{\partial\psi}{\partial t} + \frac{\psi}{J}\frac{\partial J}{\partial t} \\ &= \frac{\partial\psi}{\partial t} - \mathcal{M}_x(\psi) - \mathcal{M}_y(\psi) - \mathcal{M}_z(\psi) \qquad (3.4.197)\end{aligned}$$

where the "corrector" terms are defined by

$$\mathcal{M}_x(\psi) = \frac{\psi}{J}\frac{\partial}{\partial x_1}(Ju^F) \qquad (3.4.198)$$

$$\mathcal{M}_y(\psi) = \frac{\psi}{J}\frac{\partial}{\partial x_2}(Jv^F) \qquad (3.4.199)$$

$$\mathcal{M}_z(\psi) = \frac{\psi}{J}\frac{\partial}{\partial \tilde{x}_3}(J\tilde{w}) \qquad (3.4.200)$$

For a more profound analysis of this procedure see Ruddick (1995).

The transport equation is then forward integrated in time using

$$\begin{aligned}\frac{\psi^{n+1}-\psi^n}{\Delta t_{3D}} = &- \mathcal{A}_h(\psi_n) + \mathcal{M}_x(\psi^n) + \mathcal{M}_y(\psi^n) - \theta_a\tilde{\mathcal{A}}_v(\psi^{n+1}) - (1-\theta_a)\tilde{\mathcal{A}}_v(\psi^n) \\ &+ \mathcal{M}_z(\psi^n) + \theta_v\mathcal{D}_v(\psi^{n+1}) + (1-\theta_v)\mathcal{D}_v(\psi^n) + \mathcal{D}_h(\psi^n) + \beta(\psi^n)\end{aligned} \qquad (3.4.201)$$

c) integration with operator splitting

If IADVS > 1, time integration is performed, in analogy with the momentum equations, using the operator splitting method. Rearranging the time derivative as in Section III-4.4.1b, the transport equation is integrated in time along the following steps:

- Part A

$$\frac{\psi_A^{n+1/3}-\psi^n}{\Delta t_{3D}} = -\mathcal{A}_{hx}(\psi^n) + \mathcal{D}_{hx}(\psi^n) + \mathcal{M}_x(\psi^n) \qquad (3.4.202)$$

$$\frac{\psi_A^{n+2/3}-\psi_A^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(\psi_A^{n+1/3}) + \mathcal{D}_{hy}(\psi^n) + \mathcal{M}_y(\psi^n) \qquad (3.4.203)$$

$$\begin{aligned}\frac{\psi_A^{n+1}-\psi_A^{n+2/3}}{\Delta t_{3D}} = &- \theta_a\tilde{\mathcal{A}}_v(\psi_A^{n+1}) - (1-\theta_a)\tilde{\mathcal{A}}_v(\psi_A^{n+2/3}) + \mathcal{M}_z(\psi^n) \\ &+ \theta_v\mathcal{D}_v(\psi_A^{n+1}) + (1-\theta_v)\mathcal{D}_v(\psi_A^{n+2/3}) + \beta(\psi^n) \qquad (3.4.204)\end{aligned}$$

- Part B

$$\begin{aligned}\frac{\psi_B^{n+1/3}-\psi^n}{\Delta t_{3D}} &= -\ \theta_a\tilde{\mathcal{A}}_v(\psi_B^{n+1/3})-(1-\theta_a)\tilde{\mathcal{A}}_v(\psi^n)+\mathcal{M}_z(\psi^n)\\ &+\ \theta_v\mathcal{D}_v(\psi_B^{n+1/3})+(1-\theta_v)\mathcal{D}_v(\psi^n)+\beta(\psi^n)\end{aligned} \qquad (3.4.205)$$

$$\frac{\psi_B^{n+2/3}-\psi_B^{n+1/3}}{\Delta t_{3D}} = -\mathcal{A}_{hy}(\psi_B^{n+1/3})+\mathcal{D}_{hy}(\psi^n)+\mathcal{M}_y(\psi^n) \qquad (3.4.206)$$

$$\frac{\psi_B^{n+1}-\psi_B^{n+2/3}}{\Delta t_{3D}} = -\mathcal{A}_{hx}(\psi_B^{n+2/3})+\mathcal{D}_{hx}(\psi^n)+\mathcal{M}_x(\psi^n) \qquad (3.4.207)$$

- Updated value

$$\psi^{n+1}=\frac{1}{2}(\psi_A^{n+1}+\psi_B^{n+1}) \qquad (3.4.208)$$

Note that the horizontal advection and diffusion operators $\mathcal{A}_h$ and $\mathcal{D}_h$ in (3.4.202)–(3.4.203), (3.4.206)–(3.4.207) have been split into a X-derivative ($\mathcal{A}_{hx}$,$\mathcal{D}_{hx}$) and a Y-derivative ($\mathcal{A}_{hy}$,$\mathcal{D}_{hy}$) part.

For the reasons discussed in Section III-4.3.2 vertical advection is discretised semi-implicitly and vertical diffusion implicitly[9]. The values, taken for the implicity factors, are then given by $\theta_a = 0.501$, $\theta_v = 1$. As stated before in Section III-4.3.2, the use of the TVD scheme with the operator splitting method increases the CPU time for solving the transport equation by a factor two compared to the forward scheme (3.4.201) in the upwind case but has the capacity to preserve horizontal and vertical gradients in frontal areas. The user therefore needs to make a balance between CPU time and accuracy when selecting an appropriate scheme.

4.4.2 Discretisation of horizontal advection

The following notations are introduced

$$\mathcal{A}_{hx}(\psi)=\frac{1}{J}\frac{\partial}{\partial x_1}(Ju^F\psi)=\frac{1}{J}\frac{\partial}{\partial x_1}F_x \qquad (3.4.209)$$

$$\mathcal{A}_{hy}(\psi)=\frac{1}{J}\frac{\partial}{\partial x_2}(Jv^F\psi)=\frac{1}{J}\frac{\partial}{\partial x_2}F_y \qquad (3.4.210)$$

where the advecting current is represented by the "filtered" velocities (u^F,v^F) defined by (3.4.21)–(3.4.22).

[9] A semi-implicit option for vertical diffusion requires only a minor adaptation of the source code and can be adopted in future versions of the program.

a) advection in the X-direction

In this subsection the j- and k-indices have been dropped for convenience. The advective term is computed by differencing the fluxes F_x, calculated at the U-nodes, at the cell centre

$$\mathcal{A}_{hx}(\psi)_i = \frac{F^u_{x;i+1} - F^u_{x;i}}{J_i \Delta x_{1;i}} \tag{3.4.211}$$

where

$$F^u_{x;i} = J^u_i \Big((1 - \Omega(r^u_i)) F^u_{up;i} + \Omega(r^u_i) F^u_{lw;i} \Big) \tag{3.4.212}$$

The upwind and Lax-Wendroff fluxes are defined by

$$F^u_{up;i} = \frac{1}{2} u^F_i \Big((1 + s_i)\psi_{i-1} + (1 - s_i)\psi_i \Big) \tag{3.4.213}$$

$$F^u_{lw;i} = \frac{1}{2} u^F_i \Big((1 + c_i)\psi_{i-1} + (1 - c_i)\psi_i \Big) \tag{3.4.214}$$

where s_i and c_i are the sign and CFL-number of the advecting current

$$s_i = \text{Sign}(u^F_i)\,,\; c_i = \frac{u^F_i \Delta t_{3D}}{\Delta^u x_1} \tag{3.4.215}$$

and $\Omega(r)$ is given by one of the formulations (3.4.52)–(3.4.55), selected with the model switch IADVS.

The value of the argument r in the weighting function depends on the sign of the advecting current

$$\begin{aligned} r^u_i &= \frac{F^u_{lw;i-1} - F^u_{up;i-1}}{F^u_{lw;i} - F^u_{up;i}} \qquad \text{if} \quad u^F_i > 0 \\ r^u_i &= \frac{F^u_{lw;i+1} - F^u_{up;i+1}}{F^u_{lw;i} - F^u_{up;i}} \qquad \text{if} \quad u^F_i < 0 \end{aligned} \tag{3.4.216}$$

b) advection in the Y-direction

In this subsection the i- and k-indices have been dropped for convenience. The advective term is computed by differencing the fluxes F_y, calculated at the V-nodes, at the cell centre

$$\mathcal{A}_{hy}(\psi)_j = \frac{F^v_{y;j+1} - F^v_{y;j}}{J_j \Delta x_{2;j}} \tag{3.4.217}$$

where

$$F^v_{y;j} = J^v_j \Big((1 - \Omega(r^v_j)) F^v_{up;j} + \Omega(r^v_j) F^v_{lw;j} \Big) \tag{3.4.218}$$

The upwind and Lax-Wendroff fluxes are defined by

$$F^v_{up;j} = \frac{1}{2} v^F_j \Big((1 + s_j)\psi_{j-1} + (1 - s_j)\psi_j \Big) \tag{3.4.219}$$

$$F^v_{lw;j} = \frac{1}{2} v^F_j \Big((1 + c_j)\psi_{j-1} + (1 - c_j)\psi_j \Big) \tag{3.4.220}$$

where s_j and c_j are the sign and CFL-number of the advecting current

$$s_j = \text{Sign}(v^F_j)\;,\; c_j = \frac{v^F_j \Delta t_{3D}}{\Delta^v x_2} \tag{3.4.221}$$

and $\Omega(r)$ is given by one of the formulations (3.4.52)–(3.4.55), selected with the model switch IADVS.

The value of the argument r in the weighting function depends on the sign of the advecting current

$$\begin{aligned} r^v_j &= \frac{F^v_{lw;j-1} - F^v_{up;j-1}}{F^v_{lw;j} - F^v_{up;j}} \qquad \text{if} \quad v^F_j > 0 \\ r^v_j &= \frac{F^v_{lw;j+1} - F^v_{up;j+1}}{F^v_{lw;j} - F^v_{up;j}} \qquad \text{if} \quad v^F_j < 0 \end{aligned} \tag{3.4.222}$$

4.4.3 Discretisation of horizontal diffusion

The diffusion terms are written in the form

$$\mathcal{D}_{hx}(\psi) = \frac{1}{J}\frac{\partial}{\partial x_1}\Big(J\lambda_H \frac{\partial \psi}{\partial x_1}\Big) = \frac{1}{J}\frac{\partial}{\partial x_1} D_x \tag{3.4.223}$$

$$\mathcal{D}_{hy}(\psi) = \frac{1}{J}\frac{\partial}{\partial x_2}\Big(J\lambda_H \frac{\partial \psi}{\partial x_2}\Big) = \frac{1}{J}\frac{\partial}{\partial x_2} D_y \tag{3.4.224}$$

a) diffusion in the X-direction

For simplicity, the j- and k-indices are omitted in this subsection. The diffusion term is discretised by differencing the fluxes D_x, computed at the U-nodes, at the cell centre

$$\mathcal{D}_{hx}(\psi)_{ji} = \frac{D^u_{x;i+1} - D^u_{x;i}}{J_i \Delta x_{1;i}} \tag{3.4.225}$$

where

$$D^u_{x;i} = \lambda^u_{H;i} \frac{\Delta^u_x \psi_i}{\Delta^u x_{1;i}} \tag{3.4.226}$$

b) diffusion in the Y-direction

For simplicity, the i- and k-indices are omitted in this subsection. The diffusion term is discretised by differencing the fluxes D_y, computed at the V-nodes, at the cell centre

$$\mathcal{D}_{hy}(\psi)_{ji} = \frac{D^v_{y;j+1} - D^v_{y;j}}{J_j \Delta x_{2;j}} \tag{3.4.227}$$

where

$$D^v_{y;j} = \lambda^v_{H;j} \frac{\Delta^v_y \psi_j}{\Delta^v x_{2;j}} \tag{3.4.228}$$

c) horizontal diffusion coefficient

The value of the horizontal diffusion coefficient λ_H depends on the type of the scheme, selected with the model switch IODIF.

0 : Horizontal diffusion is disabled.

1 : A uniform value, assigned by the user, is taken.

2 : Horizontal diffusion is parameterised as a function of the velocity deformation tensor using the formulation (3.1.151)–(3.1.152).

In the last case the coefficient is discretised using

$$\lambda^c_{H;kji} = C_{s0}\Delta x_{1;ji}\Delta x_{2;j}D^c_{T;kji} \tag{3.4.229}$$

$$\lambda^u_{H;kji} = C_{s0}\Delta^u x_{1;ji}\Delta^u x_{2;j}D^u_{T;kji} \tag{3.4.230}$$

$$\lambda^v_{H;kji} = C_{s0}\Delta^v x_{1;ji}\Delta^v x_{2;j}D^v_{T;kji} \tag{3.4.231}$$

where D^c_T, D^u_T, D^v_T are given by (3.4.105)–(3.4.107).

4.4.4 Discretisation of vertical advection

The vertical advection term is written as

$$\tilde{\mathcal{A}}_v(\psi) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}(W\psi) = \frac{1}{J}\frac{\partial}{\partial\tilde{x}_3}F_z \tag{3.4.232}$$

where

$$W = J\tilde{w} + w_s \tag{3.4.233}$$

is the sum of the transformed vertical current and the sinking velocity, as defined in (3.4.192). Omitting the j- and i-indices the vertical advection term is spatially discretised by differencing the vertical fluxes F_z, computed at the W-nodes, at the cell centres. This gives

$$\tilde{\mathcal{A}}_v(\psi)_k = \frac{F^w_{z;k+1} - F^w_{z;k}}{\Delta x_{3;k}} \tag{3.4.234}$$

The advective flux is defined by

$$F^w_{z;k} = J^w_k\Big((1-\Omega(r^w_k))F^w_{up;k} + \Omega(r^w_k)F^w_{ce;k}\Big) \tag{3.4.235}$$

where the upwind and central fluxes are given by

$$F^w_{up;k} = \frac{1}{2}W_k\Big((1+s_k)\psi_{k-1} + (1-s_k)\psi_k\Big) \tag{3.4.236}$$

$$F^w_{ce;k} = \frac{1}{2}W_k(\psi_{k-1} + \psi_k) \tag{3.4.237}$$

and s_k equals the sign of the advecting current

$$s_k = \mathrm{Sign}(W_k) \tag{3.4.238}$$

The weighting factor $\Omega(r)$ is determined using one of the formulations (3.4.52)–(3.4.55), selected with the model switch IADVS. The parameter r depends on the sign of the advecting current

$$\begin{aligned} r_k^w &= \frac{F^w_{ce;k-1} - F^w_{up;k-1}}{F^w_{ce;k} - F^w_{up;k}} \qquad \text{if} \quad W_k > 0 \\ r_k^w &= \frac{F^w_{ce;k+1} - F^w_{up;k+1}}{F^w_{ce;k} - F^w_{up;k}} \qquad \text{if} \quad W_k < 0 \end{aligned} \tag{3.4.239}$$

Important to note here is that, since r_k^w is taken at old time t^n whereas the vertical advection term is evaluated semi-implicitly (depending on the value of the implicity factor θ_a), the upwind and central fluxes have to be calculated both at the old and the new time level (see Section III-4.4.8b for implementation in the program).

4.4.5 Discretisation of vertical diffusion

The vertical diffusion term is written in the form

$$\mathcal{D}_v(\psi) = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3}\Big(\frac{\lambda_T}{J}\frac{\partial \psi}{\partial \tilde{x}_3}\Big) = \frac{1}{J}\frac{\partial}{\partial \tilde{x}_3} D_z \tag{3.4.240}$$

Omitting the j- and i-indices the diffusion term is spatially discretised by differencing the vertical fluxes D_z, computed at the W-nodes, at the cell centres. This gives

$$\mathcal{D}_v(\psi)_k = \frac{D^w_{z;k+1} - D^w_{z;k}}{\Delta x_{3;k}} \tag{3.4.241}$$

where

$$D^w_{z;k} = \nu_{T;k}\frac{\Delta^w_z \psi_k}{\Delta^w x_{3;k}} \tag{3.4.242}$$

4.4.6 Discretisation of the corrector terms

The corrector terms, defined by (3.4.198)–(3.4.200), are discretised spatially using

$$\mathcal{M}_x(\psi)_{kji} = \frac{\psi_{kji}(J^u_{kj,i+1}u^F_{kj,i+1} - J^u_{kji}u^F_{kji})}{J_{kji}\Delta x_{1;ji}} \tag{3.4.243}$$

$$\mathcal{M}_y(\psi)_{kji} = \frac{\psi_{kji}(J^v_{k,j+1,i}v^F_{k,j+1,i} - J^v_{kji}v^F_{kji})}{J_{kji}\Delta x_{2;j}} \tag{3.4.244}$$

$$\mathcal{M}_z(\psi)_{kji} = \frac{\psi_{kji}((J\tilde{w})_{k+1,ji} - (J\tilde{w})_{kji})}{\Delta x_{3;kji}} \tag{3.4.245}$$

4.4.7 Boundary conditions

a) surface and bottom boundaries

- A Neumann type boundary condition of the general form (3.1.181) or (3.1.191) is applied at respectively the surface and the bottom. Their discretised forms are discussed in Section III-4.4.8 in connection with the tridiagonal matrix system.

- The surface and bottom fluxes for vertical advection are set to zero in accordance with (3.1.169) and (3.1.188)[10], i.e.

$$F^w_{z;N_z+1} = 0\,,\; F^w_{z;1} = 0 \tag{3.4.246}$$

b) land boundaries

All horizontal advection and diffusion fluxes are set to zero at land boundaries, i.e.

$$\begin{array}{ll} F^u_{x;i} = D^u_{x;i} = 0 & \text{at western/eastern boundaries} \\ F^v_{y;j} = D^v_{y;j} = 0 & \text{at southern/northern boundaries} \end{array} \tag{3.4.247}$$

c) open boundaries

The advective fluxes are evaluated with the upwind scheme at open boundaries.

- As discussed in Section III-1.6.3b two types of conditions can be selected for the advective and diffusion fluxes at the open boundaries.

  1. An external vertical profile $\psi_{ext;kji}$ is specified so that

$$F^u_{x;i} = \frac{1}{2}J^c_{i:i-1}u^F_i\Big((1\pm s_i)\psi_{ext;i} + (1\mp s_i)\psi^{n+1}_{i:i-1}\Big) \tag{3.4.248}$$

$$F^v_{y;j} = \frac{1}{2}J^c_{j:j-1}v^F_j\Big((1\pm s_j)\psi_{ext;j} + (1\mp s_j)\psi^{n+1}_{j:j-1}\Big) \tag{3.4.249}$$

$$D^u_{x;i} = \pm\lambda^c_{H;i:i-1}\frac{\psi^{n+1}_i - \psi_{ext;i}}{\Delta x_{1;i:i-1}} \tag{3.4.250}$$

$$D^v_{y;j} = \pm\lambda^c_{H;j:j-1}\frac{\psi^{n+1}_j - \psi_{ext;j}}{\Delta x_{2;j:j-1}} \tag{3.4.251}$$

  where the upper sign applies at western/southern and the lower sign at eastern/northern boundaries, and s_i, s_j are the signs of the advecting current as defined by (3.4.215) and (3.4.221). Note that the advective fluxes are, as a sole exception, evaluated implicitly at open boundaries.

[10]Sinking of organic and inorganic material into the sediment layer is taken into account in the biological and sediment modules through the fluff layer. See Section III-4.4.9c.

2. A zero normal gradient is applied which takes the following form

$$F^u_{x;i} = \frac{1}{2} J^c_{i:i-1} u^F_i \Big((1 \pm s_i) \psi^n_{i:i-1} + (1 \mp s_i) \psi^{n+1}_{i:i-1} \Big) \tag{3.4.252}$$

$$F^v_{y;j} = \frac{1}{2} J^c_{j:j-1} v^F_j \Big((1 \pm s_j) \psi^n_{j:j-1} + (1 \mp s_j) \psi^{n+1}_{j:j-1} \Big) \tag{3.4.253}$$

$$D^u_{x;i} = 0\,,\; D^v_{y;j} = 0 \tag{3.4.254}$$

- To avoid spurious discontinuities near open or land boundaries, the TVD scheme is replaced by the upwind scheme ($\Omega(r) = 0$) when the fluxes are evaluated one grid distance away from a land/open boundary and the advecting current is pointing away from that boundary. Since $\Omega(0) = 0$, this means that

$$\begin{array}{ll} r^u_i = 0 & \text{if } u^F_i > 0 \text{ and the U-node } (j{,}i-1) \text{ is an open or land boundary} \\ & \text{or } u^F_i < 0 \text{ and the U-node } (j{,}i+1) \text{ is an open or land boundary} \end{array} \tag{3.4.255}$$

$$\begin{array}{ll} r^v_j = 0 & \text{if } v^F_j > 0 \text{ and the V-node } (j-1{,}i) \text{ is an open or land boundary} \\ & \text{or } v^F_j < 0 \text{ and the V-node } (j+1{,}i) \text{ is an open or land boundary} \end{array} \tag{3.4.256}$$

4.4.8 Tridiagonal matrix

In analogy with the 3-D momentum equations (see Section III-4.3.12) the discretised forms of equations (3.4.193), (3.4.201), (3.4.204) and (3.4.205) are written as a system of linear equations in tridiagonal form

$$\begin{array}{rcl} B_{1ji}\psi^{n+1}_{1ji} + C_{1ji}\psi^{n+1}_{2ji} & = & D_{1ji} \\ A_{kji}\psi^{n+1}_{k-1,ji} + B_{kji}\psi^{n+1}_{kji} + C_{kji}\psi^{n+1}_{k+1,ji} & = & D_{kji} \\ A_{N_z,ji}\psi^{n+1}_{N_z-1,ji} + B_{N_z,ji}\psi^{n+1}_{N_z,ji} & = & D_{N_z,ji} \end{array} \tag{3.4.257}$$

where $2 \le k \le N_z - 1$ and the explicit terms are represented by the D-term on the right hand side. Each of the vertical vectors A, B, C, D is the sum of various contributions, marked by a superscript, which are discussed separately below. The horizontal indices j and i are mostly omitted.

a) time-derivative

The contributions of the time derivative are given by

$$A^t_k = 0\,,\; B^t_k = 1\,,\; C^t_k = 0 \tag{3.4.258}$$

and

$$D^t_k = \psi^n_k \quad \text{or} \quad D^t_k = \frac{J^n_k}{J^{n+1}_k}\psi^n_k \tag{3.4.259}$$

depending on whether the equation is solved with or without the corrector terms. In (3.4.258)–(3.4.259), $1 \le k \le N_z$.

b) vertical advection

The vertical advection term is split up into two contributions arising from the fluxes below and above the k-level. The former are given by

$$\begin{aligned}A_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1+f_k^w)\\ B_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1-f_k^w)\\ D_k^{a-} &= \frac{1}{2}(1-\theta_a)c_k^-[(1+f_k^w)\psi_{k-1}^n+(1-f_k^w)\psi_k^n]\end{aligned} \tag{3.4.260}$$

where $2 \leq k \leq N_z$,

$$f_k^w = (1-\Omega(r_k^w))s_k \tag{3.4.261}$$

$$c_k^- = \frac{\Delta t_{3D}W_k}{\Delta x_{3;k}} \tag{3.4.262}$$

and r_k^w is defined by (3.4.239) using the value ψ^n at the old time level.

The terms arising from the fluxes above the k-level, are

$$\begin{aligned}B_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1+f_{k+1}^w)\\ C_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1-f_{k+1}^w)\\ D_k^{a+} &= -\frac{1}{2}(1-\theta_a)c_k^+[(1+f_{k+1}^w)\psi_k^n+(1-f_{k+1}^w)\psi_{k+1}^n]\end{aligned} \tag{3.4.263}$$

where $1 \leq k \leq N_z - 1$ and

$$c_k^+ = \frac{\Delta t_{3D}W_{k+1}}{\Delta x_{3;k}} \tag{3.4.264}$$

c) vertical diffusion

As for vertical advection the fluxes below and above the k-level are taken separately. The former are given by

$$\begin{aligned}A_k^{d-} &= -\theta_v\frac{\Delta t_{3D}\lambda_{T;k}}{\Delta x_{3;k}\Delta^w x_{3;k}}\\ B_k^{d-} &= \theta_v\frac{\Delta t_{3D}\lambda_{T;k}}{\Delta x_{3;k}\Delta^w x_{3;k}}\\ D_k^{d-} &= -(1-\theta_v)\frac{\Delta t_{3D}\lambda_{T;k}}{\Delta x_{3;k}\Delta^w x_{3;k}}(\psi_k^n-\psi_{k-1}^n)\end{aligned} \tag{3.4.265}$$

where $2 \leq k \leq N_z$.

The terms, taken from the flux above the k-level, are

$$\begin{aligned}
B_k^{d+} &= \theta_v \frac{\Delta t_{3D}\lambda_{T;k+1}}{\Delta x_{3;k}\Delta^w x_{3;k+1}} \\
C_k^{d+} &= -\theta_v \frac{\Delta t_{3D}\lambda_{T;k+1}}{\Delta x_{3;k}\Delta^w x_{3;k+1}} \\
D_k^{d+} &= (1-\theta_v)\frac{\Delta t_{3D}\lambda_{T;k+1}}{\Delta x_{3;k}\Delta^w x_{3;k+1}}(\psi^n_{k+1}-\psi^n_k)
\end{aligned} \tag{3.4.266}$$

where $1 \le k \le N_z - 1$.

d) surface and bottom boundary conditions

The surface and bottom boundary conditions of all cell-centered quantities in the program are of the Neumann type and can be cast in the general form given by (3.1.181) and (3.1.191). Replacing $\psi_{s,i}$ by ψ_{N_z} and $\psi_{b,i}$ by ψ_1, the following terms are added to the highest respectively the lowest vertical level

$$B^C_{N_z} = F^{\psi}_{s1;ji}\frac{\Delta t_{3D}}{\Delta x_{3;N_z}}\,,\; D^C_{N_z} = (F^{\psi}_{s0;ji} + F^{\psi}_{s1;ji}\psi_{s,e,;ji})\frac{\Delta t_{3D}}{\Delta x_{3;N_z}} \tag{3.4.267}$$

$$B^C_1 = F^{\psi}_{b1;ji}\frac{\Delta t_{3D}}{\Delta x_{3;1}}\,,\; D^C_1 = (F^{\psi}_{b0;ji} + F^{\psi}_{b1;ji}\psi_{b,e,;ji})\frac{\Delta t_{3D}}{\Delta x_{3;1}} \tag{3.4.268}$$

e) open boundary conditions

The following contributions are obtained from the the formulations (3.4.248)–(3.4.251) for the advective fluxes in the case that an external profile ψ_{ext} is specified

$$\begin{aligned}
B^O_{kj,i:i-1} &= \Big(\mp\frac{1}{2}u^F_{kji}(1\mp s_{kji}) + \frac{\lambda^c_{H;kj,i:i-1}}{\Delta x_{1;j,i:i-1}}\Big)\frac{\Delta t_{3D}}{\Delta x_{1;j,i:i-1}} \\
D^O_{kj;i:i-1} &= \Big(\pm\frac{1}{2}u^F_{kji}(1\pm s_{kji}) + \frac{\lambda^c_{H;kj,i:i-1}}{\Delta x_{1;j,i:i-1}}\Big)\psi_{ext;kji}\frac{\Delta t_{3D}}{\Delta x_{1;j,i:i-1}}
\end{aligned} \tag{3.4.269}$$

at U-boundaries, and

$$\begin{aligned}
B^O_{k,j:j-1,i} &= \Big(\mp\frac{1}{2}v^F_{kji}(1\mp s_{kji}) + \frac{\lambda^c_{H;k,j:j-1,i}}{\Delta x_{2;j:j-1}}\Big)\frac{\Delta t_{3D}}{\Delta x_{2;j:j-1}} \\
D^O_{k,j:j-1;i} &= \Big(\pm\frac{1}{2}v^F_{kji}(1\pm s_{kji}) + \frac{\lambda^c_{H;k,j:j-1,i}}{\Delta x_{2;j:j-1}}\Big)\psi_{ext;kji}\frac{\Delta t_{3D}}{\Delta x_{2;j:j-1}}
\end{aligned} \tag{3.4.270}$$

at V-boundaries where the upper sign applies at western/southern and the lower sign at eastern/northern boundaries.

In the case of a zero normal gradient condition one obtains from (3.4.252)–(3.4.254):

$$\begin{aligned} B^O_{kj,i:i-1} &= \mp\frac{1}{2}u^F_{kji}(1\mp s_{kji})\frac{\Delta t_{3D}}{\Delta x_{1;j,i:i-1}} \\ D^O_{kj,i:i-1} &= \pm\frac{1}{2}u^F_{kji}(1\pm s_{kji})\psi^n_{kj,i:i-1}\frac{\Delta t_{3D}}{\Delta x_{1;j,i:i-1}} \end{aligned} \tag{3.4.271}$$

at U-boundaries, and

$$\begin{aligned} B^O_{k,j:j-1,i} &= \mp\frac{1}{2}v^F_{kji}(1\mp s_{kji})\frac{\Delta t_{3D}}{\Delta x_{2;j:j-1}} \\ D^O_{k,j:j-1,i} &= \pm\frac{1}{2}v^F_{kji}(1\pm s_{kji})\psi^n_{k,j:j-1,i}\frac{\Delta t_{3D}}{\Delta x_{2;j:j-1}} \end{aligned} \tag{3.4.272}$$

at V-boundaries.

f) other explicit terms

The other explicit terms (without the advection/diffusion fluxes at open boundaries) are given by

$$D^e_k = (\beta(\psi^n)_k + \mathcal{D}_h(\psi^n)_k)\Delta t_{3D} \tag{3.4.273}$$

in the absence of advection, or

$$\begin{aligned} D^e_k = \Big(&\beta(\psi^n)_k - \mathcal{A}_{hx}(\psi^n)_k - \mathcal{A}_{hy}(\psi^n)_k + \mathcal{M}_x(\psi^n)_k \\ &+ \mathcal{M}_y(\psi^n)_k + \mathcal{M}_z(\psi^n)_k + \mathcal{D}_{hx}(\psi^n)_k + \mathcal{D}_{hy}(\psi^n)_k\Big)\Delta t_{3D} \end{aligned} \tag{3.4.274}$$

in the case that no operator splitting is applied, or

$$D^e_k = (\beta(\psi^n)_k + \mathcal{M}_z(\psi^n)_k)\Delta t_{3D} \tag{3.4.275}$$

if operator splitting is applied.

g) solution of the linear system

The linear system (3.4.257) is solved using the algorithm given by (3.4.188)–(3.4.190).

4.4.9 Biological and sediment modules

Most of the numerical aspects of the biological and sediment parts of the program are related to the solution of the transport equations for the state variables, which is described in the previous sections. A few particular features are discussed below.

a) vertical sinking rate

To avoid that vertical sinking becomes disabled in the biological and sediment modules when advection is switched off (IADVS = 0), an additional switch IADVWB is implemented. This switch becomes only effective when IADVS = 0, and allows to run the program in e.g. 1-D applications, including vertical sinking. Allowed values are:

0 : vertical sinking disabled

1 : upwind scheme

2 : central scheme

3 : TVD scheme using the superbee limiter as a weighting function between the upwind and the central scheme

4: as the previous case now using the monotonic limiter.

b) source/sink terms

Although the numerical schemes for advection (except for the Lax-Wendroff scheme when IADVS or IADVWB equals 2) and diffusion should prevent the creation of negative concentrations, this problem may occur exceptionally through the source and sink terms. The following limiting conditions are therefore implemented

- The nitrate uptake in the microplankton nitrogen and nitrate equations is limited by the available amount of nitrogen

$$ {}^{NO}uB \leq {}^{NO}S/\Delta t_{3D} \tag{3.4.276}$$

- If the sum of the losses in the ammonium equation due to ammonium uptake and nitrification is larger than the available amount of ammonium, i.e.

$$ {}^{NH}uB + {}^{NH}r.{}^{NH}S > {}^{NH}S/\Delta t_{3D} \tag{3.4.277}$$

  then the following conditions are imposed

$$\begin{array}{lllll} \text{if} & {}^{NH}uB > {}^{NH}S/\Delta t_{3D} & \Longrightarrow & {}^{NH}uB = {}^{NH}S/\Delta t_{3D} & \text{and no nitrification} \\ \text{if} & {}^{NH}uB \leq {}^{NH}S/\Delta t_{3D} & \Longrightarrow & {}^{NH}r.{}^{NH}S = {}^{NH}S/\Delta t_{3D} - {}^{NH}uB & \end{array} \tag{3.4.278}$$

c) fluff layer

After solving the transport equations for the biological state variables and inorganic sediment, the fluff layer amounts of iSPM, microplankton and detrital carbon and nitrogen are updated. This is performed in the following steps:

- The resuspension rate is calculated using the formulation given by equations (3.2.72)–(3.2.77).
- The net amount of matter lost from the fluff and returned to the water column due to sinking and resuspension, is determined for each of the five fractions

$$\Delta f_l(\psi) = (E(\psi) + w^{\psi}_{s,b}\psi_b)\Delta t_{3D} \tag{3.4.279}$$

  where ψ stands either for iSPM, microplankton carbon or nitrogen, detrital carbon or nitrogen, and $w^{\psi}_{s,b}$, ψ_b are the (negative) sinking velocity and concentration at the bottom grid cell. To prevent negative amounts in the fluff layer or negative concentrations at the bottom cell $\Delta f_l(\psi)$ is limited by

$$-\psi_{bot}\Delta x_{3;1ji} \leq \Delta f_l(\psi) \leq f_l(\psi) \tag{3.4.280}$$

  where ψ_{bot} is the concentration of ψ at the bottom grid cell.
- The fluff layer amount and the water column concentrations at the bottom cell are updated[11]

$$\begin{aligned} f_l^{n+1}(\psi)_{ji} &= f_l^n(\psi)_{ji} - \Delta f_l(\psi)_{ji} \\ A_{1ji} &\equiv A_{1ji} + \Delta f_l(A)_{ji}/\Delta x_{3;1ji} \\ B_{1ji} &\equiv B_{1ji} + \Delta f_l(B)_{ji}/\Delta x_{3;1ji} \\ N_{1ji} &\equiv N_{1ji} + \Delta f_l(N)_{ji}/\Delta x_{3;1ji} \\ C_{1ji} &\equiv C_{1ji} + \Delta f_l(C)_{ji}/\Delta x_{3;1ji} \\ M_{1ji} &\equiv M_{1ji} + \Delta f_l(M)_{ji}/\Delta x_{3;1ji} \end{aligned} \tag{3.4.281}$$

- Microplankton is converted to detritus at a rate of 10% per day

$$\begin{aligned} f_l(B)_{ji} &\equiv f_l(B)_{ji} - 0.1\frac{\Delta t_{3D}}{86400}f_l(B)_{ji} \\ f_l(C)_{ji} &\equiv f_l(C)_{ji} + 0.1\frac{\Delta t_{3D}}{86400}f_l(B)_{ji} \\ f_l(N)_{ji} &\equiv f_l(N)_{ji} - 0.1\frac{\Delta t_{3D}}{86400}f_l(N)_{ji} \\ f_l(M)_{ji} &\equiv f_l(M)_{ji} + 0.1\frac{\Delta t_{3D}}{86400}f_l(N)_{ji} \end{aligned} \tag{3.4.282}$$

- Microplankton and detritus is drawn into the consolidated sediment at a rate of 10% per day

$$f_l(\psi)_{ji} \equiv f_l(\psi)_{ji} - 0.1\frac{\Delta t_{3D}}{86400}f_l(\psi)_{ji} \tag{3.4.283}$$

  where ψ stands for B, N, C and M.

[11]Note that A, B, C should not be confused with the matrices of the tridiagonal system in Section III-4.4.8.

- The amount of detrital carbon in the fluff layer and the bottom concentration of oxygen are updated for the loss due to remineralisation

$$\begin{aligned} f_l(C)_{ji} &\equiv f_l(C)_{ji} - (^Cr)_{ji} f_l(C)_{ji}\Delta t_{3D} \\ O_{1ji} &\equiv O_{1ji} - {}^Oq^C.(^Cr)_{ji} f_l(C)_{ji}\Delta t_{3D}/\Delta x_{3;1ji} \end{aligned} \tag{3.4.284}$$

- The loss of detrital nitrogen in the fluff layer due to remineralisation is added to the bottom ammonium concentration

$$\begin{aligned} f_l(M)_{ji} &\equiv f_l(M)_{ji} - (^Mr)_{ji} f_l(M)_{ji}\Delta t_{3D} \\ {}^{NH}S_{1ji} &\equiv {}^{NH}S_{1ji} + (^Mr)_{ji} f_l(M)_{ji}\Delta t_{3D}/\Delta x_{3;1ji} \end{aligned} \tag{3.4.285}$$

In each of the previous steps care is taken that neither the fluff layer amounts nor the bottom concentrations can become negative.

4.4.10 Optics

The heat absorption term in the temperature equation (3.1.28) is determined as follows (where the j- and i-indices are mostly omitted).

- The attenuation coefficient k_2 is evaluated at the cell centre using (3.1.159):

$$k_{2;k} = k_{20}^w - \epsilon^S S_k \tag{3.4.286}$$

- The infrared and short-wave component of irradiance are computed at the W-nodes

$$\begin{aligned} I^w_{1;N_z} &= R_1 Q_{sol;ji} \quad , \quad I^w_{2;N_z} = (1-R_1)Q_{sol;ji} \\ I^w_{1;k} &= I^w_{1;k+1}e^{-k_1\Delta x_{3;k}} \quad , \quad I^w_{2;k} = I^w_{2;k+1}R^*_{2;k}e^{-k_{2;k}\Delta x_{3;k}} \\ I^w_{1;1} &= 0 \quad , \quad I^w_{2;1} = 0 \end{aligned} \tag{3.4.287}$$

  where $k = N_z - 1, \cdots, 2$ and the factor for hyperexponential attenuation, defined by (3.1.160), is discretised using

$$\begin{aligned} R^*_{2;k} &= R_2^{\Delta x_{3;k}/\Delta_{opt}} && \text{if} \quad H_{ji}(1-\sigma_k) \leq \Delta_{opt} \\ R^*_{2;k} &= 1 && \text{if} \quad H_{ji}(1-\sigma_{k+1}) \geq \Delta_{opt} \\ R^*_{2;k} &= R_2^{(1-H_{ji}(1-\sigma_{k+1})/\Delta_{opt})} && \text{otherwise} \end{aligned} \tag{3.4.288}$$

  Note that the bottom values of I_1 and I_2 are set to zero to prevent absorption of heat at the sea bed.

- The absorption term is then obtained by

$$\Big(\frac{1}{J\rho_0 c_p}\frac{\partial I}{\partial \tilde{x}_3}\Big)_k = \frac{I^w_{1;k+1} + I^w_{2;k+1} - I^w_{1;k} - I^w_{2;k}}{\rho_0 c_p \Delta x_{3;k}} \tag{3.4.289}$$

Photosynthetically active radiation (PAR) is obtained in a similar way.

- The attenuation coefficient k_d is evaluated at the cell centre using (3.2.6):

$$k_{d;k} = k^w_{20} - \epsilon^S S_k + \epsilon^A A_k + \epsilon^C C_k + \epsilon^X X_k \tag{3.4.290}$$

- PAR is first computed at the W-nodes

$$\begin{aligned} I^w_{p;N_z} &= (1-R_1)Q_{sol;ji} \\ I^w_{p;k} &= I^w_{p;k+1}R^*_{2;k}e^{-k_{d;k}\Delta x_{3;k}} \end{aligned} \tag{3.4.291}$$

  where $R^*_{2;k}$ is determined by (3.4.288).

- The value of PAR at the cell centre is then given by

$$I^c_{p;k} = \frac{1}{2}(I^w_{p;k} + I^w_{p;k+1}) \tag{3.4.292}$$

- In the program this value is further averaged over the past 24 hours. This gives

$$I^{c;n+1}_{p;k}\Big|_{av} = I^{c;n}_{p;k}\Big|_{av}(1 - \frac{\Delta t_{3D}}{86400}) + \frac{\Delta t_{3D}}{86400}I^c_{p;k} \tag{3.4.293}$$

4.5 Turbulence transport equations and closure schemes

The numerical procedure for solving the transport equations for the turbulence variables k, kl and ε is mainly similar to the ones given in Section III-4.4, except that that the scalar ψ is now taken at the W-nodes and the vertical sinking rate term is absent.

4.5.1 Time discretisation

The transport equations are solved with or without the operator splitting method and corrector terms as described in Section III-4.4.1 except that

- The source terms are taken explicitly whereas the sink terms are discretised quasi-implicitly to ensure positivity (Patankar, 1980). This is further discussed in Section III-4.5.3.

- In the absence of vertical sinking $W = J\tilde{w}$.

- Since advection and horizontal diffusion of turbulence quantities is considered of minor importance, a model switch IAHDHT has been introduced with the following values:

0 : Advective and horizontal diffusion terms are set to zero in the turbulence transport equations (only) even when IADVS $\neq 0$, i.e.

$$\mathcal{A}_h(\psi) = \mathcal{A}_v(\psi) = \mathcal{D}_h(\psi) = 0 \tag{3.4.294}$$

1 : Advection and horizontal diffusion is enabled when IADVS $\neq 0$. The schemes for advection and horizontal diffusion are selected with the switches IADVS and IODIF.

4.5.2 Spatial discretisation

The following changes are made with respect to the spatial discretisation of advection and diffusion, described in Sections III-4.4.2–4.4.6.

- Quantities previously evaluated at the U-nodes are now taken at the X-nodes. This means in particular that

$$(F_x^u, D_x^u, u^F, J^u, \lambda_H^u) \quad \longrightarrow \quad (F_x^X, D_x^X, u^{F;X}, J^X, \lambda_H^X) \tag{3.4.295}$$

- Quantities previously evaluated at the V-nodes are now taken at the Y-nodes. This means in particular that

$$(F_y^v, D_y^v, v^F, J^v, \lambda_H^v) \quad \longrightarrow \quad (F_y^Y, D_y^Y, v^{F;Y}, J^Y, \lambda_H^Y) \tag{3.4.296}$$

- Quantities previously evaluated at the W-nodes are now taken at the cell centres. This means in particular that

$$(F_z^w, D_z^w, W, J^w, \lambda_T^{\psi;w}) \quad \longrightarrow \quad (F_z^c, D_z^c, W^c, J^c, \lambda_T^{\psi;c}) \tag{3.4.297}$$

  where λ_T^ψ denotes the vertical diffusion coefficient $\nu_T/\sigma_k + \nu_b$ used in the k- and kl-equations (3.1.114), (3.1.125) or the coefficient $\nu_T/\sigma_\varepsilon + \nu_b$ in the ε-equation (3.1.128). Since the cell-centered vertical fluxes $F_{z;k}^c$ and $D_{z;k}^c$ must have the same vertical index as the quantity ψ_k at the W-node above the cell centre, the vertical index at the cell centre is increased by one and varies between 2 at the bottom and $N_z + 1$ at the surface cell.

- Quantities previously evaluated at the centre are now taken at the W-nodes.

Apart from the transport equations the calculations in the turbulence scheme, implemented in the program, are mostly straightforward. Almost no interpolation is required except for the squared shear and buoyancy frequencies M^2 and N^2 defined by respectively (3.1.91) and (3.1.90):

- M^2 is spatially discretised using

$$\begin{aligned}\left(M_{kji}^w\right)^2 = \Big(\frac{1}{2\Delta^w x_{3;kji}}\Big)^2\big[&(u_{kji} - u_{k-1,ji} + u_{kj,i+1} - u_{k-1,j,i+1})^2 \\ &+ (v_{kji} - v_{k-1,ji} + v_{k,j+1,i} - v_{k-1,j+1,i})^2\big]\end{aligned} \tag{3.4.298}$$

- To avoid spurious numerical oscillations N^2 is spatially discretised by averaging over the neighbouring cells in the horizontal (except at solid and open boundaries):

$$\left(N^w_{kji}\right)^2 = \frac{1}{6}(2\tilde{N}^2_{kji} + \tilde{N}^2_{k,j-1,i} + \tilde{N}^2_{k,j+1,i} + \tilde{N}^2_{kj,i-1} + \tilde{N}^2_{kj,i+1}) \qquad (3.4.299)$$

where

$$\tilde{N}^2_{kji} = \frac{g}{2\Delta x_{3;kji}}\Big((\beta_{T;k-1,ji} + \beta_{T;kji})(T_{kji} - T_{k-1,ji}) - (\beta_{S;k-1,ji} + \beta_{S;kji})(S_{kji} - S_{k-1,ji})\Big) \qquad (3.4.300)$$

is the unfiltered squared frequency at the W-node.

4.5.3 Source and sink terms

The source terms in the k-, kl- and ε-equation are discretised explicitly. This gives

$$\mathcal{P}(k) = \nu^n_T M^2 + \lambda^n_T H(-N^2) \qquad (3.4.301)$$

$$\mathcal{P}(kl) = \frac{1}{2}E_1 l^n(\nu^n_T M^2 + \lambda^n_T H(-N^2)) \qquad (3.4.302)$$

$$\mathcal{P}(\varepsilon) = c_{1\varepsilon}\frac{\varepsilon^n}{k^n}(\nu^n_T M^2 + \lambda^n_T H(-N^2)) \qquad (3.4.303)$$

where $H(x)$ is the Heaviside function ($H(x) = 0$ for $x < 0$ and $= x$ otherwise).

The sink terms are discretised using the quasi-implicit forms

$$\mathcal{S}(k) = \lambda^n_T H(N^2)\frac{k^{n+1}}{k^n} + \varepsilon^n\frac{k^{n+1}}{k^n} \qquad (3.4.304)$$

$$\mathcal{S}(kl) = \frac{1}{2}E_1\lambda^n_T H(N^2)\frac{k^{n+1}l^{n+1}}{k^n} + \varepsilon_0\tilde{W}\frac{(k^n)^{1/2}k^{n+1}l^{n+1}}{2l^n} \qquad (3.4.305)$$

$$\mathcal{S}(\varepsilon) = c_{1\varepsilon}c_{3\varepsilon}\lambda^n_T H(N^2)\frac{\varepsilon^{n+1}}{k^n} + c_{2\varepsilon}\frac{\varepsilon^{n+1}\varepsilon^n}{k^n} \qquad (3.4.306)$$

where $\tilde{W}$ is the wall proximity function defined by (3.1.126).

4.5.4 Boundary conditions

The open boundary conditions (in the case that horizontal advection is applied) are as given in Section III-4.4.7 except that a zero normal gradient condition is considered for all turbulence variables at open boundaries, i.e.

$$F^X_{x;i} = \frac{1}{2}J_{i:i-1}u^{F;X}_i\Big((1 \pm s_i)\psi^n_{i:i-1} + (1 \mp s_i)\psi^{n+1}_{i:i-1}\Big) \qquad (3.4.307)$$

$$F^Y_{y;j} = \frac{1}{2}J_{j:j-1}v^{F;Y}_j\Big((1 \pm s_j)\psi^n_{j:j-1} + (1 \mp s_j)\psi^{n+1}_{j:j-1}\Big) \qquad (3.4.308)$$

$$D^X_{z;i} = 0\,,\; D^Y_{y;j} = 0 \qquad (3.4.309)$$

4.5.5 Tridiagonal matrix

Although the formulation is similar to the one described in Section III-4.4.8, there are a few differences such as the form of the surface and bottom boundary conditions. All equations are therefore given in detail. The tridiagonal matrix new consists of N_z+1 equations

$$\begin{aligned} B_{1ji}\psi_{1ji}^{n+1} + C_{1ji}\psi_{2ji}^{n+1} &= D_{1ji} \\ A_{kji}\psi_{k-1,ji}^{n+1} + B_{kji}\psi_{kji}^{n+1} + C_{kji}\psi_{k+1,ji}^{n+1} &= D_{kji} \\ A_{N_z+1,ji}\psi_{N_z,ji}^{n+1} + B_{N_z+1,ji}\psi_{N_z+1,ji}^{n+1} &= D_{N_z+1,ji} \end{aligned} \tag{3.4.310}$$

where $2 \le k \le N_z$.

a) time-derivative

The contributions of the time derivative are given by

$$A_k^t = 0\,,\; B_k^t = 1\,,\; C_k^t = 0 \tag{3.4.311}$$

for $2 \le k \le N_z$, and

$$D_k^t = \psi_k^n \quad \text{or} \quad D_k^t = \frac{J_k^{w;n}}{J_k^{w;n+1}}\psi_k^n \tag{3.4.312}$$

depending on whether the transport equation is solved with ($\mathsf{IAHDHT} = 1$, $\mathsf{IADVS} > 0$) or without ($\mathsf{IAHDHT} = 0$ or $\mathsf{IADVS} = 0$) the corrector terms.

b) vertical advection

Vertical advection is applied if $\mathsf{IAHDHT} = 1$ and $\mathsf{IADVS} > 0$. The contributions from the fluxes below the k-level are then given by

$$\begin{aligned} A_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1+f_k^c) \\ B_k^{a-} &= -\frac{1}{2}\theta_a c_k^-(1-f_k^c) \\ D_k^{a-} &= \frac{1}{2}(1-\theta_a)c_k^-[(1+f_k^c)\psi_{k-1}^n + (1-f_k^c)\psi_k^n] \end{aligned} \tag{3.4.313}$$

where $2 \le k \le N_z$, and

$$f_k^c = (1-\Omega(r_k^c))s_k \tag{3.4.314}$$

$$c_k^- = \frac{\Delta t_{3D} W_k^c}{\Delta^w x_{3;k}} \tag{3.4.315}$$

Note that r_k^c is calculated using the value ψ^n at the old time level.

The terms arising from the fluxes above the k-level, are

$$\begin{aligned} B_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1+f_{k+1}^c) \\ C_k^{a+} &= \frac{1}{2}\theta_a c_k^+(1-f_{k+1}^c) \\ D_k^{a+} &= -\frac{1}{2}(1-\theta_a)c_k^+[(1+f_{k+1}^c)\psi_k^n + (1-f_{k+1}^c)\psi_{k+1}^n] \end{aligned} \tag{3.4.316}$$

where $2 \le k \le N_z$ and

$$c_k^+ = \frac{\Delta t_{3D} W_{k+1}^c}{\Delta^w x_{3;k}} \tag{3.4.317}$$

c) vertical diffusion

The contributions, obtained from the fluxes below the k-level, are

$$\begin{aligned} A_k^{d-} &= -\theta_v \frac{\Delta t_{3D}\lambda_{T;k}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k}} \\ B_k^{d-} &= \theta_v \frac{\Delta t_{3D}\lambda_{T;k}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k}} \\ D_k^{d-} &= -(1-\theta_v)\frac{\Delta t_{3D}\lambda_{T;k}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k}}(\psi_k^n - \psi_{k-1}^n) \end{aligned} \tag{3.4.318}$$

where $2 \le k \le N_z$.

The terms, taken from the flux above the k-level, are

$$\begin{aligned} B_k^{d+} &= \theta_v \frac{\Delta t_{3D}\lambda_{T;k+1}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k+1}} \\ C_k^{d+} &= -\theta_v \frac{\Delta t_{3D}\lambda_{T;k+1}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k+1}} \\ D_k^{d+} &= (1-\theta_v)\frac{\Delta t_{3D}\lambda_{T;k+1}^{\psi;c}}{\Delta^w x_{3;k}\Delta x_{3;k+1}}(\psi_{k+1}^n - \psi_k^n) \end{aligned} \tag{3.4.319}$$

where $2 \le k \le N_z$.

d) surface and bottom boundary conditions

All turbulence variables use a Dirichlet type condition at the surface and bottom as given by (3.1.180) and (3.1.190). The conditions are applied either at the surface and bottom itself or one grid distance away from those boundaries. Each case is presented separately.

- k-equation

$$A_{N_z+1}^C = 0\,,\; B_{N_z+1}^C = 1,\,,\; D_{N_z+1}^C = u_{*s}^2/\varepsilon_0^{2/3} \tag{3.4.320}$$

$$B_1^C = 1\,,\; C_1^C = 0\,,\; D_1^C = u_{*b}^2/\varepsilon_0^{2/3} \tag{3.4.321}$$

- kl-equation
  - The values of kl at the surface and bottom are set to zero. Hence,

$$A_{N_z+1} = 0\,,\; B_{N_z+1} = 1\,,\; D_{N_z+1} = 0 \tag{3.4.322}$$

$$B_1 = 1\,,\; C_1 = 0\,,\; D_1 = 0 \tag{3.4.323}$$

  - The conditions are applied one grid distance below the surface and above the bottom

$$A_{N_z} = 0\,,\; B_{N_z} = 1\,,\; C_{N_z} = 0\,,\; D_{N_z} = u_{*s}^2 l_{2;N_z}/\varepsilon_0^{2/3} \tag{3.4.324}$$

$$A_2 = 0\,,\; B_2 = 1\,,\; C_2 = 0\,,\; D_2 = u_{*b}^2 l_{1;2}/\varepsilon_0^{2/3} \tag{3.4.325}$$

- ε-equation
  - The values of ε at the surface and bottom level are set to zero as given by (3.4.322)–(3.4.323).
  - The conditions are applied one grid distance below the surface and above the bottom

$$A_{N_z} = 0\,,\; B_{N_z} = 1\,,\; C_{N_z} = 0\,,\; D_{N_z} = k_{N_z}^{3/2}\varepsilon_0/l_{2;N_z} \tag{3.4.326}$$

$$A_2 = 0\,,\; B_2 = 1\,,\; C_2 = 0\,,\; D_2 = k_2^{3/2}\varepsilon_0/l_{1;2} \tag{3.4.327}$$

The length scales l_1 and l_2, defined by (3.1.130), are taken at respectively the vertical levels 2 and N_z:

$$l_{1;2ji} = \kappa(\sigma_2 H_{ji} + z_{0b})\,,\; l_{2;N_z,ji} = \kappa(H_{ji} - \sigma_{N_z} H_{ji} + z_{0s}) \tag{3.4.328}$$

e) open boundary conditions

When horizontal advection is enabled (IAHDHT $= 1$, IADVS > 0), a zero gradient condition is applied so that the contributions are analogous to (3.4.271)–(3.4.272).

f) source, sink and other explicit terms

The contributions are the following

$$B_k^s = \Delta t_{3D}\mathcal{S}(\psi^n)_k/\psi_k^{n+1} \tag{3.4.329}$$

and either

$$D_k^e = (\mathcal{P}(\psi^n)_k + \mathcal{D}_h(\psi^n)_k)\Delta t_{3D} \tag{3.4.330}$$

in the absence of advection, or

$$\begin{aligned} D_k^e = \Big(&\mathcal{P}(\psi^n)_k - \mathcal{A}_{hx}(\psi^n)_k - \mathcal{A}_{hy}(\psi^n)_k + \mathcal{M}_x(\psi^n)_k \\ &+ \mathcal{M}_y(\psi^n)_k + \mathcal{M}_z(\psi^n)_k + \mathcal{D}_{hx}(\psi^n)_k + \mathcal{D}_{hy}(\psi^n)_k\Big)\Delta t_{3D} \end{aligned} \tag{3.4.331}$$

in the case that no operator splitting is applied, or

$$D_k^e = (\mathcal{P}(\psi^n)_k + \mathcal{M}_z(\psi^n)_k)\Delta t_{3D} \tag{3.4.332}$$

if operator splitting is applied.

g) solution of the linear system

The linear system (3.4.310) is solved using the algorithm given by (3.4.188)–(3.4.190).

4.5.6 Implementation in the program

The aim of the turbulence module is to update the values of the eddy coefficients ν_T and λ_T. They are evaluated in the program prior to the solution of the momentum and all other scalar transport equations so that ν_T and λ_T are taken at the old time step.

- If IOPTK = 0, ν_T and λ_T are uniform and are obviously not updated.
- If IOPTK = 1, the coefficients are determined by one of the formulations described in Section III-1.2.1.
- If IOPTK = 2, the procedure consists of the following steps given below.

1. The turbulence quantities k, l ε are determined using a formulation which depends in the first place on the value of the switch NTRANS.
    - NTRANS = 0:
        - l is determined using a formulation selected by ILENG.
        - k is obtained either from (3.1.136)–(3.1.137) if ISTPAR = 1 or (3.1.140) if ISTPAR = 2.
        - ε is calculated using (3.1.116).
    - NTRANS = 1:
        - k is computed by solving the k-equation (3.1.114).
        - l is determined using a formulation selected by ILENG.
        - ε is calculated using (3.1.116).
        - If ILIM = 1, a limiting condition is applied for k and l if ITCPAR = 1 or for k and ε if ITCPAR = 2.
    - NTRANS = 2:
        - k is computed by solving the k-equation (3.1.114).
        - l is determined by solving the kl-equation (3.1.125) if ITCPAR = 1 or from (3.1.116) if ITCPAR = 2.
        - ε is calculated by solving the ε-equation (3.1.128) if ITCPAR = 2 or from (3.1.116) if ITCPAR = 1.

- If ILIM = 1, a limiting condition is applied for k and l if ITCPAR = 1 or for k and ε if ITCPAR = 2.

2. The eddy coefficients are updated as follows:

   - The stability coefficients are computed using the expressions (3.1.118)–(3.1.121) and (3.1.122)–(3.1.123) depending on the values of the switches ITCPAR and ISTPAR.
   - ν_T and λ_T are evaluated using either (3.1.115) if ITCPAR = 1 or (3.1.117) if ITCPAR = 2.

4.6 Spherical coordinates

There are no fundamental differences between the numerical solutions of the model equations in Cartesian and spherical coordinates. The procedures described in the previous sections remain valid except for a few changes listed below.

4.6.1 Model grid

- The x_1-coordinate is replaced by the longitude λ, the x_2-coordinate by the latitude ϕ. For convenience, λ and ϕ will be denoted by x_1 and x_2 in the equations below.
- The grid spacings are defined in meters (as in the Cartesian case) using

$$\Delta x_{1;ji} = R\cos\phi^c_j\Delta\lambda_i\,,\ \Delta x_{2;j} = R\Delta\phi_j \tag{3.4.333}$$

where

$$\phi^c_j = \frac{1}{2}(\phi_j + \phi_{j+1}) \tag{3.4.334}$$

is the latitude at the cell centers. Contrary to the Cartesian formulation, Δx_1 now varies also in the Y-direction.

4.6.2 Momentum equations

- The Coriolis frequency f in the momentum equations is replaced by $2\Omega\sin\phi^c_j$ at the U-nodes and by $2\Omega\sin\phi_j$ [12] at the V-nodes where $\Omega = 2\pi/86164$ is the angular frequency of the Earth's rotation.
- The Y-derivative terms in the 2-D continuity equation (3.4.12) and in the discretised equations (3.4.23) and (3.4.24) for the transformed and physical vertical current are replaced by respectively

$$\frac{\Delta^c_y(\cos\phi^v\overline{V}^m)}{\cos\phi^c\Delta x_2} \tag{3.4.335}$$

[12] Note that $\phi^v_j = \phi_j$, $\phi^u_j = \phi^c_j$.

$$-\frac{1}{\cos\phi^c\Delta x_2}\Delta_y^c\Big[\cos\phi^v J_k^{v;n+1}(v_k^F - \frac{\overline{V}^F}{H^{v;n+1}})\Big] \tag{3.4.336}$$

$$\frac{\Delta_y^c(\cos\phi^v J_k^{v;n+1}v_k^{n+1}x_{3;k}^{v;n+1})}{\cos\phi^c\Delta x_2} \tag{3.4.337}$$

- An extra advective term appears on the right hand side of the 3-D and 2-D momentum equations.
  - The term $\tan\phi^u u^n v^{u;n}/R$ is added to the right hand side of (3.4.9), (3.4.27), (3.4.30).
  - The term $-\tan\phi^v(u^{v;n})^2/R$ is added to the right hand side of (3.4.10), (3.4.35), (3.4.40).
  - The term $\tan\phi^u\overline{U}^m\overline{V}^{u;m}/(RH^{u;n})$ is added to the right hand side of (3.4.13).
  - The term $-\tan\phi^v(\overline{U}^{v;m})^2/(RH^{v;n})$ is added to the right hand side of (3.4.14).

- The advective operator $\mathcal{A}_{hy}$ in the expressions (3.4.44) and (3.4.46) now takes the form

$$\mathcal{A}_{hy}(u) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(F_{yx}) \tag{3.4.338}$$

$$\mathcal{A}_{hy}(v) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(F_{yy}) \tag{3.4.339}$$

  where F_{yx} and F_{yy}, defined in Section III-4.3.3, are multiplied by respectively $\cos\phi^v$ and $\cos\phi^c$. The spatially discretised forms of (3.4.338)–(3.4.339) are still given by (3.4.72) and (3.4.57) except that
  - $\Delta^u x_{2;j}$ is multiplied by $\cos\phi_j^u$ in (3.4.72).
  - $\Delta x_{2;j}^v$ is multiplied by $\cos\phi_j^v$ in (3.4.57).
  - $F_{yx;ji}^{W;v}$ and $F_{yx;ji}^{E;v}$ are multiplied by $\cos\phi_j^v$ in (3.4.66) and (3.4.70).
  - $F_{yy;ji}^c$ is multiplied by $\cos\phi^c$ in (3.4.58).

  The advective fluxes in the 2-D momentum equations are adapted similarly using the prescriptions given in Section III-4.3.4.

- In view of the definitions (3.1.57)–(3.1.58) for the horizontal shear stresses, the expressions (3.4.89) and (3.4.91) for horizontal diffusion contain an extra term so that

$$D_{xx;kji}^c = 2J_{kji}\nu_{H;kji}^c\Big(\frac{\Delta_x^c u_{kji}}{\Delta x_{1;ji}} - \frac{\tan\phi_j^c}{R}v_{kji}^c\Big) \tag{3.4.340}$$

$$D_{yx;kji}^{V;u} = J_{kji}^u\nu_{H;kji}^u\Big(\frac{\Delta_x^u v_{kji}^c}{\Delta^u x_1} + \frac{\tan\phi_j^u}{R}u_{kji}\Big) \tag{3.4.341}$$

  The corresponding fluxes for the 2-D mode are adapted similarly using the prescriptions given in Section III-4.3.6.

- An extra horizontal diffusion term appears on the right hand side of the 3-D and 2-D momentum equations.

  - The following term is added to the right hand side of (3.4.9), (3.4.27), (3.4.30):

$$-\frac{\tan\phi^u_j}{R}\tau^u_{\phi\lambda;kji} = -\frac{\tan\phi^u_j}{4R}\Big(\tilde{D}^{U;v}_{yx;kji}+\tilde{D}^{U;v}_{yx;k,j+1,i}+\tilde{D}^{U;v}_{yx;kj,i-1}+\tilde{D}^{U;v}_{yx;k,j+1,i-1}+4\tilde{D}^{V;u}_{yx;kji}\Big) \qquad (3.4.342)$$

  where $\tilde{D}^{U;v}_{yx}$ and $\tilde{D}^{V;u}_{yx}$ are defined by (3.4.90) and (3.4.341) but without the J-term.

  - The following term is added to the right hand side of (3.4.10), (3.4.36), (3.4.39):

$$\frac{\tan\phi^v_j}{R}\tau^v_{\lambda\lambda;kji} = \frac{\tan\phi^v_j}{2R}(\tilde{D}^c_{xx;k,j-1,i}+\tilde{D}^c_{xx;kji}) \qquad (3.4.343)$$

  where $\tilde{D}^c_{xx}$ is defined by (3.4.340) but without the J-term.

  - The terms

$$-\frac{\tan\phi^u_j}{R}\overline{\tau^u_{\phi\lambda;kji}} \quad \text{and} \quad \frac{\tan\phi^v_j}{R}\overline{\tau^v_{\lambda\lambda;kji}}$$

  are added to respectively (3.4.13) and (3.4.14). They are obtained from (3.4.342)–(3.4.343) after replacing (u,v,ν_H) by ($\overline{U}/H$,$\overline{V}/H$,$\overline{\nu_H}$).

- The diffusion operators $\mathcal{D}_{yx}$ and $\mathcal{D}_{yy}$ now take the form

$$\mathcal{D}_{yx}(u,v) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(D^U_{yx}) + \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(D^V_{yx}) \qquad (3.4.344)$$

$$\mathcal{D}_{yy}(v) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(D_{yy}) \qquad (3.4.345)$$

where D^U_{yx}, D^V_{yx} and D_{yy}, defined by (3.4.90), (3.4.341) and (3.4.95) in Section III-4.3.5, are multiplied by respectively $\cos\phi^v$, $\cos\phi^u$ and $\cos\phi^c$. The spatially discretised forms of (3.4.344) and (3.4.345) are given by (3.4.93), (3.4.94) and (3.4.98) except that

  - The denominators of (3.4.93)–(3.4.94) are multiplied by $\cos\phi^u_j$.
  - The denominator of (3.4.98) is multiplied by $\cos\phi^v_j$.
  - $D^{U;v}_{yx;ji}$ is multiplied by $\cos\phi^v$.
  - $D^{V;u}_{yx;ji}$ is multiplied by $\cos\phi^u$.
  - $D^c_{yy;ji}$ is multiplied by $\cos\phi^c$.

Similar changes are made for the horizontal diffusion terms in the 2-D momentum equations using the prescriptions given in Section III-4.3.6.

- The depth integrals (3.4.83)–(3.4.84) for advection and (3.4.110)–(3.4.111) for diffusion are calculated by

$$\begin{aligned}\overline{A}^h_{\lambda;ji} &= \sum_{k=1}^{N_z}[J^u_{kji}(\mathcal{A}_{hx}(u)_{kji} + \mathcal{A}_{hy}(u)_{kji} - \frac{\tan\phi^u_j}{R}u_{kji}v^u_{kji})] \\ &\quad - \overline{\mathcal{A}_{hx}}(\overline{U})_{ji} - -\overline{\mathcal{A}_{hy}}(\overline{U})_{ji} + \frac{\tan\phi^u_j}{R}\frac{\overline{U}_{ji}\overline{V}^u_{ji}}{H^u_{ji}} \qquad (3.4.346)\end{aligned}$$

$$\begin{aligned}\overline{A}^h_{\phi;ji} &= \sum_{k=1}^{N_z}[J^v_{kji}(\mathcal{A}_{hx}(v)_{kji} + \mathcal{A}_{hy}(v)_{kji} + \frac{\tan\phi^v_j}{R}(u^v_{kji})^2)] \\ &\quad - \overline{\mathcal{A}_{hx}}(\overline{V})_{ji} - -\overline{\mathcal{A}_{hy}}(\overline{V})_{ji} - \frac{\tan\phi^v_j}{R}(\overline{U}^v_{ji})^2 \qquad (3.4.347)\end{aligned}$$

$$\begin{aligned}\overline{D}^h_{\lambda;ji} &= \sum_{k=1}^{N_z}[J^u_{kji}(\mathcal{D}_{xx}(u)_{kji} + \mathcal{D}_{yx}(u,v)_{kji} - \frac{\tan\phi^u_j}{R}\tau^u_{\phi\lambda;kji})] \\ &\quad - \overline{\mathcal{D}_{xx}}(\overline{U})_{ji} - \overline{\mathcal{D}_{yx}}(\overline{U},\overline{V})_{ji} + \frac{\tan\phi^u_j}{R}\overline{\tau_{\phi\lambda}}^u_{;ji} \qquad (3.4.348)\end{aligned}$$

$$\begin{aligned}\overline{D}^h_{\phi;ji} &= \sum_{k=1}^{N_z}[J^v_{kji}(\mathcal{D}_{xy}(u,v)_{kji} + \mathcal{D}_{yy}(v)_{kji} + \frac{\tan\phi^v_j}{R}\tau^v_{\lambda\lambda;kji})] \\ &\quad - \overline{\mathcal{D}_{xy}}(\overline{U},\overline{V})_{ji} - \overline{\mathcal{D}_{yy}}(\overline{V})_{ji} - \frac{\tan\phi^v_j}{R}\overline{\tau_{\lambda\lambda}}^v_{;ji} \qquad (3.4.349)\end{aligned}$$

- The expressions (3.4.105)–(3.4.107) for the strain rate D_T (omitting the k-index) are now given by

$$\begin{aligned}\big(D^c_{T;ji}\big)^2 &= \Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} - \frac{\tan\phi^c_j}{R}v^c_{ji}\Big)^2 + \Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}}\Big)^2 \\ &+ \frac{1}{8}\Big(\frac{\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j+1,i}}{\Delta^v x_{2;j+1}} + \frac{\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} + \frac{\Delta^u_x v^c_{j,i+1}}{\Delta^u x_{1;j,i+1}} + 2\frac{\tan\phi^c_j}{R}u^c_{ji}\Big)^2 \qquad (3.4.350)\end{aligned}$$

$$\begin{aligned}\big(D^u_{T;ji}\big)^2 &= \frac{1}{4}\Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} + \frac{\Delta^c_x u_{j,i-1}}{\Delta x_{1;j,i-1}} - 2\frac{\tan\phi^u_j}{R}v^u_{ji}\Big)^2 + \frac{1}{4}\Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}} + \frac{\Delta^c_y v_{j,i-1}}{\Delta x_{2;j}}\Big)^2 \\ &+ \frac{1}{32}\Big(\frac{\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j,i-1}}{\Delta^v x_{2;j}} + \frac{\Delta^v_y u^c_{j+1,i}}{\Delta^v x_{2;j+1}} + \frac{\Delta^v_y u^c_{j+1,i-1}}{\Delta^v x_{2;j+1}} + \frac{4\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} + \frac{4\tan\phi^u_j}{R}u_{ji}\Big)^2 \\ &\qquad (3.4.351)\end{aligned}$$

$$\begin{aligned}\big(D^v_{T;ji}\big)^2 &= \frac{1}{4}\Big(\frac{\Delta^c_x u_{ji}}{\Delta x_{1;ji}} + \frac{\Delta^c_x u_{j-1,i}}{\Delta x_{1;j-1,i}} - 2\frac{\tan\phi^v_j}{R}v_{ji}\Big)^2 + \frac{1}{4}\Big(\frac{\Delta^c_y v_{ji}}{\Delta x_{2;j}} + \frac{\Delta^c_y v_{j-1,i}}{\Delta x_{2;j-1}}\Big)^2 \\ &+ \frac{1}{32}\Big(\frac{4\Delta^v_y u^c_{ji}}{\Delta^v x_{2;j}} + \frac{\Delta^u_x v^c_{ji}}{\Delta^u x_{1;ji}} + \frac{\Delta^u_x v^c_{j-1,i}}{\Delta^u x_{1;j-1,i}} + \frac{\Delta^u_x v^c_{j,i+1}}{\Delta^u x_{1;j,i+1}} + \frac{\Delta^u_x v^c_{j-1,i+1}}{\Delta^u x_{1;j-1,i+1}} + \frac{4\tan\phi^v_j}{R}u^v_{ji}\Big)^2 \\ &\qquad (3.4.352)\end{aligned}$$

- The zero gradient condition (3.4.150) for v is rewritten as

$$v_{kji} = \frac{v_{k,j+1:j-1,i}\cos\phi^v_{j+1:j-1}J^v_{k,j+1:j-1,i}}{\cos\phi^v_j J_{k,j:j-1,i}} + \frac{\cos\phi^v_j\overline{V}_{ji} - \sum_{k=1}^{N_z} v_{k,j+1:j-1,i}\cos\phi^v_{j+1:j-1}J^v_{k,j+1:j-1,i}}{\cos\phi^v_j H_{j:j-1,i}} \tag{3.4.353}$$

- The open boundary conditions for the advective and diffusion fluxes are as given in Section III-4.3.10d except that
  - All fluxes at V-nodes are now multiplied by $\cos\phi^v_j$.
  - A zero normal gradient condition is applied for the diffusion fluxes, used in (3.4.342), i.e.

$$\tilde{D}^{U;v}_{yx;ji} = \tilde{D}^{U;v}_{yx;j+1:j-1,i} \tag{3.4.354}$$

$$\tilde{D}^{V;u}_{yx;ji} = \tilde{D}^{V;u}_{yx;j,i+1:i-1} \tag{3.4.355}$$

- Equations (3.4.163) and (3.4.164) for the outgoing Riemann characteristics are modified as follows:
  - The second term on the right hand side of (3.4.163) now becomes

$$\pm c^n_{i:i-1}\frac{\Delta^c_y(\cos\phi^v\overline{V}_{i:i-1})}{\cos\phi^c\Delta x_2} \tag{3.4.356}$$

  - The following extra term is added to the right hand side of (3.4.164):

$$\mp c^n_{j:j-1}\frac{\tan\phi^v_j}{R}\overline{V}_j \tag{3.4.357}$$

4.6.3 Scalar transport equations

- The operator for advection in the X-direction is now defined by

$$\mathcal{A}_{hy}(\psi) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}(F_y) \tag{3.4.358}$$

where F_y, defined by (3.4.218), is multiplied by $\cos\phi^v_j$. The denominator in (3.4.217) is then multiplied by $\cos\phi^c_j$.

- The operator for diffusion in the Y-direction is now defined by

$$\mathcal{D}_{hy}(\psi) = \frac{1}{JR\cos\phi}\frac{\partial}{\partial\phi}D_y \tag{3.4.359}$$

where D_y, defined by (3.4.228), is multiplied by $\cos\phi^v_j$. The denominator in (3.4.227) is then multiplied by $\cos\phi^c_j$.

- The definition (3.4.199) for the corrector operator $\mathcal{M}_y$ is now given by

$$\mathcal{M}_y(\psi) = \frac{\psi}{JR\cos\phi}\frac{\partial}{\partial\phi}(J\cos\phi\, v^F) \tag{3.4.360}$$

  and its discretised form (3.4.244) by

$$\mathcal{M}_y(\psi)_{kji} = \frac{\psi_{kji}(\cos\phi^v_{j+1}J^v_{k,j+1,i}v^F_{k,j+1,i} - \cos\phi^v_j J^v_{kji}v^F_{kji})}{\cos\phi^c_j J_{kji}\Delta x_{2;j}} \tag{3.4.361}$$

- The open boundary conditions for the advective and diffusive fluxes are the same as in Section III-4.4.7c except that the flux $F^v_{y:j}$ in (3.4.249) and (3.4.253) is now multiplied by $\cos\phi^v_j$.

4.7 One-dimensional application

When the program switch IGRDIM is set to 1, the program is used as a point model in the vertical. The basic equations (see Section III-1.1.4) now take a much simpler form since all advective (except the vertical sinking term) and horizontal diffusion terms are neglected. The momentum equations (3.1.80)–(3.1.81) are integrated without mode-splitting along the following steps:

- The sea surface elevation and the pressure gradient are updated at the new time level t^{n+1} with the aid of the expressions (3.1.83)–(3.1.85).
- The vertical grid spacing and the Jacobian J are updated using

$$\Delta x^{n+1}_{3;k} = J^{n+1}_k = \sigma_k(h+\zeta^{n+1}) \tag{3.4.362}$$

- Predicted values are obtained for u, v by integrating the momentum equations in time

$$\begin{aligned}\frac{u^p-u^n}{\Delta t_{3D}} &= fv^{u;n:p} + \theta_v\mathcal{D}_v(u^p) + (1-\theta_v)\mathcal{D}_v(u^n) - g\Big(\frac{\partial\zeta}{\partial x_1}\Big)^{n+1} + Q_1\\ \frac{v^p-v^n}{\Delta t_{3D}} &= -fu^{v;p:n} + \theta_v\mathcal{D}_v(v^p) + (1-\theta_v)\mathcal{D}_v(v^n) - g\Big(\frac{\partial\zeta}{\partial x_2}\Big)^{n+1} + Q_2\end{aligned} \tag{3.4.363}$$

  The vertical diffusion terms are discretised using the forms given in Section III-4.3.8.

- A corrector step is applied yielding the values at the new time

$$u^{n+1} = \frac{J^n}{J^{n+1}}u^p\,,\ v^{n+1} = \frac{J^n}{J^{n+1}}v^p \tag{3.4.364}$$

- Although the depth-integrated currents $\overline{U}$, $\overline{V}$ are no longer required for the internal calculations in the program, they are still updated at each time step by integrating u, v over the vertical.

- The scalar transport equation (3.1.82) is integrated in time using the form (3.4.193) without the horizontal diffusion term. The vertical diffusion and sinking rate term are discretised using the formulations described in Sections III-4.4.4–4.4.5.

4.8 Lagrangian particle module

4.8.1 Particle coordinates

Each particle n is represented by six coordinates:

- i_n, j_n, k_n: the indices of the centre of the grid cell where the particle is currently located

- x'_{1n}, x'_{2n}, σ'_n: the particle's coordinates (in σ-space) relative to the cell centre

The coordinates (x_{1n},x_{2n},x_{3n}) of the particle with respect to the Cartesian reference frame are then given by

$$\begin{aligned} x_{1n} &= x_{1;i_n} + \frac{1}{2}\Delta x_{1;j_ni_n} + x'_{1n} \\ x_{2n} &= x_{2;j_n} + \frac{1}{2}\Delta x_{2;j_n} + x'_{2n} \\ x_{3n} &= H(j_n,i_n)(\sigma_{k_n} + \frac{1}{2}\Delta\sigma_{k_n} + \sigma'_n) - h(j_n,i_n) \end{aligned} \tag{3.4.365}$$

where ($x_{1;i_n}$,$x_{2;j_n}$) are the coordinates of the corner of the cell where the particle is located (see Section III-4.2.3). In the case of a spherical coordinate system (x'_{1n},x'_{2n}) are replaced by (λ'_n,ϕ'_n) and (x_{1n},x_{2n}) by (λ_n,ϕ_n). All calculations within the particle module are performed internally using a local Cartesian frame with respect to the cell centre. The relative coordinates (x'_{1n},x'_{2n}) are obtained by

$$x'_{1n} = \lambda'_n \frac{\Delta x_{1;j_ni_n}}{\lambda_{i_n+1} - \lambda_{i_n}} \; , \; x'_{2n} = \phi'_n \frac{\Delta x_{2;j_n}}{\phi_{j_n+1} - \phi_{j_n}} \tag{3.4.366}$$

where

$$\Delta x_{1;j_ni_n} = R\cos\phi^c_{j_n}\Delta\lambda_{i_n} \; , \; \Delta x_{2;j_n} = R\Delta\phi_{j_n} \tag{3.4.367}$$

When all particle positions are updated for the new time t^{n+1}, the horizontal coordinates are converted back to decimal degrees.

4.8.2 General procedures

The sequence of operations for the update of the particle positions at a new time step is as follows:

1. The positions are first updated for advection. The transition from the old to the new coordinates is indicated by an additional superscript ^{a}, i.e.

$$(i_n^n, j_n^n, k_n^n, x_{1n}'^n, x_{2n}'^n, \sigma_n'^n) \quad \longrightarrow \quad (i_n^a, j_n^a, k_n^a, x_{1n}'^a, x_{2n}'^a, \sigma_n'^a)$$

2. The X-coordinates are updated for diffusion in the X-direction or

$$(i_n^a, x_{1n}'^a) \quad \longrightarrow \quad (i_n^{n+1}, x_{1n}'^{n+1})$$

3. The Y-coordinates are updated for diffusion in the Y-direction or

$$(j_n^a, x_{2n}'^a) \quad \longrightarrow \quad (j_n^{n+1}, x_{2n}'^{n+1})$$

4. The Z-coordinates are updated for vertical diffusion or

$$(k_n^a, \sigma_n'^a) \quad \longrightarrow \quad (k_n^{n+1}, \sigma_n'^{n+1})$$

5. New particles are generated at open sea boundaries during inflow when an external concentration ρ_{spm}^e is specified.

6. New particles are generated at river boundaries when a river discharge Q_d is specified.

Each step is separately discussed in the subsections below.

4.8.3 Transport by advection

The following basic assumptions are made:

- The components of the current within a cell are only function of their own direction, i.e.

$$u(x_1)\,,\; v(x_2)\,,\; W(\sigma) \tag{3.4.368}$$

  where $W = J\tilde{w}$ is the transformed vertical current.

- The gradient of the current is uniform within a grid cell

$$u(x_{1s}) = u_w + x_{1s}\frac{\partial u}{\partial x_1}\,,\; v(x_{2s}) = v_s + x_{2s}\frac{\partial v}{\partial x_2}\,,\; W(\sigma_s) = W_b + \sigma_s\frac{\partial W}{\partial \sigma} \tag{3.4.369}$$

  where u_w, v_s, W_b are respectively the values of u at the western, v at the southern and W at the bottom cell face, and x_{1s}, x_{2s}, σ_s the distances to the western, southern and bottom faces.

Assume then that x^i_{1s} and x^e_{1s} are the values of x_{1s} for a particle at time t and $t+\Delta t_{3D}$. The advected distance in the X-direction is then determined by integrating the equation

$$dx_{1s} = u(x_{1s})dt = (u_w + x_{1s}u'_x)dt \tag{3.4.370}$$

where

$$u'_x = \frac{\partial u}{\partial x_1} \tag{3.4.371}$$

This gives

$$\int_{x^i_{1s}}^{x^e_{1s}} \frac{dx_{1s}}{u_w + x_{1s}u'_x} = \int_t^{t+\Delta t_{3D}} dt \tag{3.4.372}$$

Hence

$$u'_x\Delta t_{3D} = \ln\Big(\frac{u_w + x^e_{1s}u'_x}{u_w + x^i_{1s}u'_x}\Big) \tag{3.4.373}$$

so that

$$x^e_{1s} = \frac{(u_w + x^i_{1s}u'_x)e^{u'_x\Delta t_{3D}} - u_w}{u'_x} \tag{3.4.374}$$

The distance traveled by the particle is then given by

$$\Delta x_{1;adv} = x^e_{1s} - x^i_{1s} = \Big(\frac{u_w}{u'_x} + x^i_{1s}\Big)\Big(e^{u'_x\Delta t_{3D}} - 1\Big) \tag{3.4.375}$$

The advected distance in the Y- and vertical directions are obtained in a similar way:

$$\Delta x_{2;adv} = \Big(\frac{v_s}{v'_y} + x^i_{2s}\Big)\Big(e^{v'_y\Delta t_{3D}} - 1\Big) \tag{3.4.376}$$

$$\Delta\sigma_{adv} = \Big(\frac{W_b}{W'_\sigma} + \sigma^i_s\Big)\Big(e^{W'_\sigma\Delta t_{3D}/H} - 1\Big) \tag{3.4.377}$$

where

$$v'_y = \frac{\partial v}{\partial x_2}\,,\; W'_\sigma = \frac{\partial W}{\partial \sigma} \tag{3.4.378}$$

and (x^i_{2s},σ^i_s) are the initial coordinates of the particle in the Y- and vertical (σ-) direction.

The procedure for updating the position of a particle by advection is then as follows:

1. The initial "rest" time Δt_{rest} is set to Δt_{3D}.
2. The advected distances during Δt_{rest} in the X-, Y- and Z-directions are determined using (3.4.375), (3.4.376), (3.4.377).
3. If no boundary crossing occurs, the particle's relative coordinates are updated for advection at the same cell, i.e.

$$x'^a_{1n} = x'^n_{1n} + \Delta x_{1;adv}\,,\; x'^a_{2n} = x'^n_{2n} + \Delta x_{2;adv}\,,\; \sigma'^a_n = \sigma'_n + \Delta\sigma_{adv} \tag{3.4.379}$$

$$(i^a_n, j^a_n, k^a_n) \;=\; (i^n_n, j^n_n, k^n_n) \tag{3.4.380}$$

4. If the particle traversed one or more boundary faces, the following operations are performed:

   - A time step Δt^x_{min} is defined as the time required to reach the western ($\Delta x_{1;adv} < 0$) or eastern ($\Delta x_{1;adv} > 0$) boundary of the cell, which is determined from (3.4.373):
$$\Delta t^x_{min} = \frac{1}{u'_x} \ln\Big(\frac{\Delta x_{1;edge} u'_x}{u_w + x^i_{1s} u'_x} + 1\Big) \tag{3.4.381}$$
     where $\Delta x_{1;edge}$ represents $\pm$ the distance to the corresponding western or eastern cell boundary. Similar time steps Δt^y_{min}, Δt^z_{min} are defined in the Y- and Z-direction:
$$\Delta t^y_{min} = \frac{1}{v'_y} \ln\Big(\frac{\Delta x_{2;edge} v'_y}{v_s + x^i_{2s} v'_y} + 1\Big) \tag{3.4.382}$$
$$\Delta t^z_{min} = \frac{H}{W'_\sigma} \ln\Big(\frac{\Delta \sigma_{edge} W'_\sigma}{W_b + \sigma^i_s W'_\sigma} + 1\Big) \tag{3.4.383}$$
     where $\Delta x_{2;edge}$, $\Delta \sigma_{edge}$ are $\pm$ the distances to the southern or northern and bottom or top cell boundaries.

   - A minimum time step Δt_{min} is defined as the time required for the first crossing of a boundary and is equal to either Δt^x_{min}, Δt^y_{min} or Δt^z_{min} depending on which boundary face is first attained by the particle.

   - The particle's coordinates are reset at the boundary face where the first crossing occurs. If this takes place, for example, at the western or eastern boundary, the new coordinates are
$$(i^a_n, j^a_n, k^a_n) = (i^n_n \mp 1, j^n_n, k^n_n) \tag{3.4.384}$$
     and
$$(x'^a_{1n}, x'^a_{2n}, \sigma'^a_n) = (\pm \Delta x_{1;i_n \mp 1}, x'^n_{2n} + \Delta x_{2;min}, \sigma'^n_n + \Delta \sigma_{min}) \tag{3.4.385}$$
     where the upper (lower) sign applies at a western (eastern) boundary and $\Delta x_{2;min}$ and $\Delta \sigma_{min}$ are the advected distances in the Y- and σ-direction obtained from (3.4.376), (3.4.377) after replacing Δt_{3D} by Δt_{min}. A similar procedure is applied when the particle first reaches a southern/northern or bottom/top boundary cell.

   - A new rest time $\Delta t_{rest} - \Delta t_{min}$ is defined and the calculations proceed again from step 2 until either Δt_{rest} becomes zero or the particle no longer crosses a boundary.

4.8.4 Transport by diffusion

a) horizontal diffusion

The particle's position for diffusion in the X-direction is obtained as follows:

- A diffused distance is defined by

$$\Delta x_{1;dif} = \Delta t_{3D} u'_{max;k_n j_n i_n}(2\gamma_{x;n} - 1) \tag{3.4.386}$$

  where u'_{max} is defined by (3.3.5) and $\gamma_{x;n}$ is a random number between 0 and 1.

- If no boundary crossing occurs, the particle's relative coordinate is taken within the same cell, i.e.

$$x'^{n+1}_{1n} = x'^{a}_{1n} + \Delta x_{1;dif}\ ,\ i^{n+1}_n = i^a_n \tag{3.4.387}$$

- If the particle traverses a boundary face, the new coordinates in the X-direction are

$$i^{n+1}_n = i^a_n - 1\ ,\ x'^{n+1}_{1n} = x'^{a}_{1n} + \Delta x_{1;dif} + \Delta^u x_{1;j^a_n i^a_n} \tag{3.4.388}$$

  at western, and

$$i^{n+1}_n = i^a_n + 1\ ,\ x'^{n+1}_{1n} = x'^{a}_{1n} + \Delta x_{1;dif} - \Delta^u x_{1;j^a_n, i^a_n+1} \tag{3.4.389}$$

  at eastern boundaries.

A similar procedure is followed for diffusion in the Y-direction. Important to note here is that the algorithm assumes that the particle cannot be diffused over a distance larger than one grid distance. The following upper limit for the horizontal diffusion coefficient λ^C_H can then be derived from (3.3.5):

$$\lambda^C_H < \frac{\Delta h^2_{min}}{6\Delta t_{3D}} \tag{3.4.390}$$

where Δh_{min} is the minimum horizontal grid spacing in the X- and Y-direction.

b) vertical diffusion

A different procedure is taken for vertical diffusion where the particle is allowed to traverse more than one bottom or top boundary face. The method is the following:

- A diffused distance is defined by

$$\Delta\sigma_{dif} = \Delta t_{3D} w'_{max;k_n j_n i_n}(2\gamma_{z;n} - 1)/H_{j_n i_n} \tag{3.4.391}$$

  where w'_{max} is defined by (3.3.5) and $\gamma_{z;n}$ a random number between 0 and 1.

- If no boundary crossing occurs, the particle's relative coordinate is taken within the same cell, i.e.

$$\sigma'^{n+1}_n = \sigma'^{a}_n + \Delta\sigma_{dif}\ ,\ k^{n+1}_n = k^a_n \tag{3.4.392}$$

- When one or more boundary faces are traversed, the new coordinates in the vertical direction are determined as follows:

- If $\Delta\sigma_{dif} > 0$, one has

$$k_n^{n+1} = k_n^a + K \, , \; \sigma_n'^{n+1} = \sigma_n'^a + \Delta\sigma_{dif} - \sum_{k=1}^{K} \Delta\sigma_{k_n^a+k} \tag{3.4.393}$$

where K is the lowest number for which

$$-\frac{1}{2}\Delta\sigma_{k+K} \leq \sigma_n'^a + \Delta\sigma_{dif} - \sum_{k=1}^{K} \Delta\sigma_{k_n^a+k} \leq \frac{1}{2}\Delta\sigma_{k+K} \tag{3.4.394}$$

- If $\Delta\sigma_{dif} < 0$, one has

$$k_n^{n+1} = k_n^a - K \, , \; \sigma_n'^{n+1} = \sigma_n'^a + \Delta\sigma_{dif} + \sum_{k=1}^{K} \Delta\sigma_{k_n^a-k} \tag{3.4.395}$$

where K is the lowest number for which

$$-\frac{1}{2}\Delta\sigma_{k-K} \leq \sigma_n'^a + \Delta\sigma_{dif} - \sum_{k=1}^{K} \Delta\sigma_{k_n^a-k} \leq \frac{1}{2}\Delta\sigma_{k-K} \tag{3.4.396}$$

4.8.5 Boundary conditions

a) surface and bottom

- The particle is not allowed to jump out of the surface or to sink into the bottom by advection, in accordance with (3.3.6).
- A reflective condition is considered for vertical advection. This is implemented as follows:
  - If $\Delta\sigma_{dif} > 0$ and $\Delta\sigma_{dif} > \Delta\sigma_{sur}$ where $\Delta\sigma_{sur}$ is the distance of the particle's initial position to the surface in σ-coordinates, the particle is first positioned at the surface and then diffused downwards over a distance $\Delta\sigma_{dif} - \Delta\sigma_{sur}$ using the procedures given in Section III-4.8.4b.
  - If $\Delta\sigma_{dif} < 0$ and $|\Delta\sigma_{dif}| > \Delta\sigma_{bot}$ where $\Delta\sigma_{bot}$ is the distance of the particle's initial position to the sea bed in σ-coordinates, the particle is first positioned at the bottom and then diffused upwards over a distance $|\Delta\sigma_{dif}| - \Delta\sigma_{bot}$ using the procedures given in Section III-4.8.4b.

b) land boundaries

- Particles crossing a land boundary are lost to the system and are no longer updated.

- As for vertical diffusion, the particles are reflected backwards inside the computational domain by horizontal diffusion at closed boundaries. In the case of diffusion in the X-direction this gives

$$i_n^{n+1} = i_n^a \, , \; x_{1n}'^{n+1} = -\Delta x_{1;dif} \mp \Delta x_{1;j_n^a i_n^a} - x_{1n}'^a \tag{3.4.397}$$

  where the upper (lower) sign applies at western (eastern) boundaries. Backward reflection at southern or northern boundaries by diffusion in the Y-direction is treated similarly.

c) open sea boundaries

- Particles crossing an open sea boundary are lost to the system and no longer updated.
- A reflective condition is used for horizontal diffusion (see Section III-4.8.5b).
- New particle input is provided during inflow if an external (depth-independent) concentration $\rho^e_{spm}(x_1, x_2)$ of suspended matter (in ton/m^3) is specified by the user[13]. The procedure is the following:
  - At each (wet) grid cell (k,j,i) adjacent to an open sea boundary, the amount of suspended matter M_{tr} is calculated which enters the cell during inflow. This is given by

$$M_{tr;kji} = \max(\pm \rho^e_{spm;kji} u_{kj,i:i+1} H_{ji} \Delta x_{2;j} \Delta\sigma_k \Delta t_{3D}, 0) \tag{3.4.398}$$

    at U-boundaries, and

$$M_{tr;kji} = \max(\pm \rho^e_{spm;kji} v_{k,j:j+1,i} H_{ji} \Delta x_{1;ji} \Delta\sigma_k \Delta t_{3D}, 0) \tag{3.4.399}$$

    at V-boundaries, where the upper (lower) sign applies at western/southern (eastern/northern) boundaries.
  - The new material is distributed into a series of particles each having the same mass m_{sea}. The number of new particles is determined by

$$L_{new;kji} = \text{int}\Big(\frac{M_{tr;kji} + M_{rest;kji}}{m_{sea}} + \frac{1}{2}\Big) \tag{3.4.400}$$

    where "int" is the integer rounding function.
  - The remaining mass fraction M_{rest} is set to zero at the initial time $t = 0$ and updated at each time step using

$$M_{rest} \equiv M_{rest} + M_{tr} - m_{sea} L_{new} \tag{3.4.401}$$

[13]Note the although ρ^e_{spm} is given here as a time-independent function, the program allows to provide time-dependent input at selected time intervals. This is further discussed in Section IV-2.8.

– The horizontal relative coordinates of the new particles are set to

$$x'_{1n} = \mp\frac{1}{2}\Delta x_{1;ji} + \Delta t_{3D} u_{kj,i:i+1}(\frac{1}{2} \mp \frac{1}{2} \pm \gamma_{x;n}) \, , \, x'_{2n} = \Delta x_{2;j}(\gamma_{y;n} - \frac{1}{2}) \quad (3.4.402)$$

at U-boundaries, and

$$x'_{1n} = \Delta x_{1;ji}(\gamma_{x;n} - \frac{1}{2}) \, , \, x'_{2n} = \mp\frac{1}{2}\Delta x_{2;j} + \Delta t_{3D} v_{k,j:j+1,i}(\frac{1}{2} \mp \frac{1}{2} \pm \gamma_{y;n}) \quad (3.4.403)$$

at V-boundaries, while the vertical coordinates are given by

$$\sigma'_n = \Delta\sigma_k(\gamma_{z;n} - \frac{1}{2}) \quad (3.4.404)$$

The random numbers $\gamma_{x;n}$, $\gamma_{y;n}$, $\gamma_{z;n}$ are between 0 and 1. The first relation in (3.4.402) is obtained as follows. The particle is first positioned, prior to advection, at random inside the exterior cell (assumed to have the same dimensions) adjacent to the boundary. This gives $\Delta x_{1;ji}/2 + (\gamma_{x;n} - 1)\Delta t_{3D} u_{kji}$ at western and $-\Delta x_{1;ji}/2 - \gamma_{x;n}\Delta t_{3D} u_{kj,i+1}$ at eastern boundaries. The advected distance $\Delta t_{3D} u_{kj,i:i+1}$ is then applied and the additional term $\mp\Delta x_{1;ji}$ is added to reposition the particle inside the new (interior) cell. The three contributions are added giving x'_{1n}. The second relation in (3.4.403) is obtained similarly.

- Note that the method assumes that the criterion (3.4.8) is valid.

d) river open boundaries

- Particles crossing a river boundary are lost to the system.
- Particles are reflected backwards by the action of horizontal diffusion (see Section III-4.8.5b).
- New particle input can be provided if a (depth-independent) river discharge $Q_d(x_1, x_2)$ of suspended matter (in ton/s) is specified by the user[14]. The procedure is the following:

  - At each (wet) grid cell (k,j,i) adjacent to a river boundary, the amount of suspended matter M_{tr}, entering the cell during the interval Δt_{3D}, is calculated by

$$M_{tr;kji} = Q_{d;ji}\Delta\sigma_k\Delta t_{3D} \quad (3.4.405)$$

  - The new material is distributed into a series of new particles each having the same mass m_{riv}. The number of new particles is determined by

$$L_{new;kji} = \mathrm{int}\Big(\frac{M_{tr;kji} + M_{rest;kji}}{m_{riv}} + \frac{1}{2}\Big) \quad (3.4.406)$$

[14]Note that, although Q_d is given as a time-independent function, the program allows to provide time-dependent input at selected time intervals. This is further discussed in Section IV-2.8.

– The remaining mass fraction M_{rest} is set to zero at the initial time $t = 0$ and updated at each time step using

$$M_{rest} \equiv M_{rest} + M_{tr} - m_{riv} L_{new} \tag{3.4.407}$$

– The relative coordinates of the new particles are set to

$$x'_{1n} = (\gamma_{x;n} - \frac{1}{2})\Delta x_{1;ji}\ ,\ x'_{2n} = (\gamma_{y;n} - \frac{1}{2})\Delta x_{2;j}\ ,\ x'_{3n} = (\gamma_{z;n} - \frac{1}{2})\Delta \sigma_k \tag{3.4.408}$$

where the random numbers $\gamma_{x;n}$, $\gamma_{y;n}$, $\gamma_{z;n}$ are between 0 and 1.

4.8.6 Program utilities

The following utilities are implemented in the particle module.

- Each particle has an attached label, which is attributed as follows:
  - The labels of the particles which are inside the domain at the initial $t = 0$, are stored in the user-defined array LST.
  - The user-defined arrays LSTOBU, LSTOBV are used in the program to represent the labels of the particles entering the computational domain at open sea and river boundaries. Different values can be selected at different boundary locations.
  - The label of a particle leaving the domain at an open sea or land boundary is set to zero which means that its position is no longer updated. Its coordinates, however, remain stored in the appropriate program arrays.

  The user-defined arrays LST, LSTOBU, LSTOBV may take any integer value, except zero. The use of labels is of interest if a distinction needs to be made between particles entering at a specific open sea or river boundary or between particles initially distributed in certain areas of the computational domain. See Section IV-2.5.3 and IV-2.6.1 for further details.

- Each particle, inside the domain at the initial time, has a mass which is stored in the user-defined array ST. The masses of the particles entering the domain through open sea and river boundaries, are given by the user-defined parameters m_{sea} and m_{riv}.

- After the update of the particle positions at the new time level t^{n+1}, the volume concentration ρ_{spm} of suspended material is calculated by summing the masses of the particles inside each grid cell and dividing the total mass by the cell volume. The result is stored into the 3-D array SPMCON which is of no relevance for the internal calculations within the program but may be of interest for user-defined output.

- The amount of suspended matter entering or leaving the computational domain is stored into respectively the 3-D arrays STIN and STOUT. These arrays are not relevant for the internal calculations within the program but may be useful for user-defined output. See Section IV-2.6.4 for further details.

4.9 General solution procedures

The update of the transported quantities in the program (currents, scalars and particles) can be summarised as follows.

1. Update the horizontal diffusion coefficients at the old time (ν_H^n,λ_H^n,$\overline{\nu_H}^n$).
2. Update the baroclinic pressure gradient at the old time (Q_1^n,Q_2^n,$\overline{Q}_1^n$,$\overline{Q}_2^n$).
3. Evaluate the eddy coefficients at the old time (ν_T^n,λ_T^n).
4. Solve the 3-D momentum equations for the predictor step (u^p,v^p).
5. Solve the 2-D equation set M_t times using the 2-D time step with the 2-D continuity equation solved before the equations for the depth-integrated currents (ζ^{n+1},$\overline{U}^{n+1}$,$\overline{V}^{n+1}$).
6. Update the horizontal currents at the new time using the corrector step (u^{n+1},v^{n+1}).
7. Integrate the 3-D continuity equation ($(J\tilde{w})^{n+1}$).
8. Solve the salinity equation (S^{n+1}).
9. Solve the temperature equation (T^{n+1}).
10. Update the density ρ and the expansion coefficients (ρ^{n+1},β_T^{n+1},β_S^{n+1}).
11. Update the biological concentrations (B^{n+1},X^{n+1},N^{n+1},C^{n+1},M^{n+1},${}^{NO}S^{n+1}$,${}^{NH}S^{n+1}$, O^{n+1},Z_N^{n+1}).
12. Solve the transport equations for contaminants (C_i^{n+1}).
13. Solve the transport equation for inorganic sediment (A^{n+1}).
14. Update the fluff layer and bottom concentrations in the biological and sediment modules ($f_l^{n+1}(\psi)$,A_{bot}^{n+1},B_{bot}^{n+1},N_{bot}^{n+1},C_{bot}^{n+1},M_{bot}^{n+1}).
15. Update the particle positions in the Lagrangian module (i_n^{n+1},j_n^{n+1},k_n^{n+1}, $x_{1n}^{\prime n+1}$,$x_{2n}^{\prime n+1}$,$\sigma_n^{\prime n+1}$).

More details are found in the description of the main program MAINPROG (see Section V-1.2) and in the flow chart given in Figure 5.1.1.

4.10 Model switches

A number of switches, listed in Table 3.4.5, are available in the program to select different numerical schemes. A full description of all program switches is given in Section IV-2.2.1.

Table 3.4.5: Program switches to select numerical schemes.

IGTRH	A Cartesian or a spherical grid is selected if this switch is set to 0/1.
IGRDIM	Selects the dimensions of the numerical grid. 1 : 1-D application 3 : 3-D application
IOPT2	The 2-D mode calculations are disabled/enabled if this switch is set to 0/1.
IOPT3	The solution of the 3-D horizontal momentum and continuity equation is disabled/enabled if this switch is set to 0/1.
IADVC	Selects the advection scheme for momentum. 0 : horizontal and vertical advection disabled 1 : upwind scheme for horizontal and vertical advection 2 : Lax-Wendroff scheme for horizontal, central scheme for vertical advection 3 : TVD scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical 4 : as the previous case now using the monotonic limiter
IADVS	Selects the advection scheme for scalars. 0 : horizontal and vertical advection disabled 1 : upwind scheme for horizontal and vertical advection 2 : Lax-Wendroff scheme for horizontal, central scheme for vertical advection 3 : TVD scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical 4 : as the previous case now using the monotonic limiter

Table 3.4.5: continued.

IODIF	Selects the scheme for horizontal diffusion (momentum and scalars). 0 : horizontal diffusion disabled 1 : uniform diffusion coefficients for momentum and scalars 2 : diffusion coefficients proportional to the deformation tensor as given by (3.1.151)–(3.1.152) or (3.1.153)–(3.1.154)
IADVWB	Selects the vertical advection scheme for vertical sinking if IADVS $= 0$. 0 : vertical sinking disabled 1 : upwind scheme 2 : central scheme 3 : TVD scheme using the superbee limiter as a weighting function between the upwind and the central scheme 4 : as the previous case now using the monotonic limiter
IAHDHT	Advection and horizontal diffusion is disabled/enabled in the turbulence transport equations if this switch is set to 0/1.

Chapter 5

Future Developments

The aim of this chapter is to describe a few ideas for improvement of the COHERENS model which may be implemented in future versions of the program. Some of these are already in phase of development while others are or may be planned within future projects. Users are encouraged to provide suggestions for improvement or to make their own updates available for COHERENS users.

5.1 General aspects

5.1.1 FORTRAN language

The program is written in FORTRAN 77. Although this language has been a standard for scientific computing for several years, it has a number of known shortcomings, such as the non-allowance of variable expressions in the declaration of array dimensions. The more recent FORTRAN 90 standard, now available on most modern machines, offers several advantages compared to the older FORTRAN 77 language and allows for a more efficient programming. A few specific features of FORTRAN 90 are noted:

- New forms of program units can be defined in the form of modules, which are more powerful than the subroutine and function subprograms and the INCLUDE files used in FORTRAN 77.
- The possibility to perform operations on whole arrays or to specify arrays or array sections as arguments of intrinsic functions.
- A powerful tool of FORTRAN 90 is that arrays can be declared with non-constant dimensions or that the dimension values can be redefined ("allocatable arrays"), allowing a more efficient use of computer memory.
- The use of keywords in procedure calls.
- Names may be longer than 6 characters and the FORTRAN character set has been extended.

Tests showed that COHERENS can be compiled with a FORTRAN 90 compiler. However, it is clear that a complete restructuring of the code becomes necessary to make an optimum use of the new features offered by FORTRAN 90.

5.1.2 Model grid and numerics

- An advantage of the σ-transformation is that it allows an easy representation of the surface and bottom boundaries. This property is particularly useful in tidal shelf seas. However, problems may occur with a σ-grid at shelf edges with strong bathymetric slopes or for resolving shallow plume or thermocline layers over a sloping bottom. An elegant solution, proposed by Deleersnijder and Beckers (1992), consists in defining different σ-grids over two or more multiple layers in the vertical. The method has been applied by e.g. Beckers (1991) in the western Mediterrranean and by Marsaleix et al. (1998) for the Rhône plume. A second known, but still unresolved problem with the σ-coordinate system, already discussed in Section III-1.1.1b, is the generation of significant errors via the pressure gradient term in the presence of a steep bottom.

- A prominent feature of coastal seas is the alternate drying and wetting of grid cells in nearshore areas. A drying/wetting algorithm is not foreseen in the current version of the program but may be implemented in a future update of COHERENS.

- A problem which may arise when applying the model to an area using several components of the model (currents, temperature, salinity, biology, ...), is the necessity to represent different processes, each having its own length or time scale. To resolve both large scale (general circulation, tides) as well as small scale (fronts, eddies) phenomena within the same model run, a grid size must be selected over the whole area in accordance with the smallest scale. Since the grid size also limits the time step through the CFL condition, this procedure is computationally inefficient. A commonly employed solution is to make use of nested grids (e.g. Oey and Chen, 1992a) whereby a coarse grid is taken for the whole area and a fine resolution over a number of selected subdomains. The model equations are then solved on each of these subgrids using different grid sizes and time steps. Values of the variables at the open boundaries of the nested grids are obtained from the coarse grid solution. The external and internal solutions are matched by applying a "sponge-layer" formulation over a subdomain near the nest boundaries.

- A specific feature of integrated models is that, besides different length scales, the model has to resolve different time scales which may widely differ. The time scales, inherent to biological processes, are, for example, usually much longer than the ones related to the physics. In the current version of COHERENS the same time step is applied for all 3-D (physical and biological) quantities, which severely limits the size (i.e. number of state variables) in the biological module. A possible solution would be to implement "nesting" techniques in time with a larger time step for the biology compared to the physics. To circumvent numerical stability problems new numerical schemes have to be implemented using implicit formulations for horizontal advective terms (see e.g. Hirsch 1988, 1990 for an overview).

- The current version of COHERENS allows to set up the program either for three-dimensional or for one-dimensional applications. An additional feature would be to implement an option to run the model in a two-dimensional form using depth-integrated transport equations. No extra coding is required for the circulation part of the program, except for the bottom stress formulation, since the 2-D momentum and continuity equations are already solved by the mode-splitting technique. An additional transport module has to be created for solving the transport equations of 2-D scalars advected by the 2-D horizontal current. A 2-D formulation of the program is of interest for applications where CPU time is of importance while vertical effects can be neglected.

- Although a variety of advection schemes are available from the literature, only a few are implemented in the program, mainly to reduce the computational overhead. New ones can be considered in a future release. For example, the "Piecewise Parabolic Method (PPM)" scheme (Colella and Woodward, 1984) is monotone (in contrast to Lax-Wendroff) while test studies (e.g. James, 1996) showed that the scheme has a very low numerical diffusion. The disadvantage is a larger CPU time, compared to the TVD scheme.

5.1.3 Boundary conditions

The following improvements for the open boundary conditions are planned in future updates:

- Although simple in form, the zero normal gradient condition has the disadvantage of being fully reflective at open boundaries. An alternative, which requires only a small programming effort, is to implement a wave-like condition allowing disturbances to propagate in or out of the domain, such as Orlanski's (1976) radiation condition. A recent overview of different open boundary conditions is found in Jensen (1998).

- A widely-used formulation is the flow relaxation scheme (Martinsen and Engedahl, 1987; Røed and Cooper, 1987). The method defines a relaxation zone near the boundaries of the domain where the internal solution is allowed to relax towards the external value specified at the open boundary. The procedure is particularly useful when fine-scale grids are embedded within a larger coarse-grid domain (see above).

5.1.4 Model output

- Time series, harmonic and time-averaged output is performed in the program by writing the same amount of data at each output time to a specified data file. Output data can more easily be accessed if the data are written as a time series of records each having the same size. This can be performed in FORTRAN by opening the data files for DIRECT access and specifying the record length.

- The current version of the program allows to write data either in ASCII or in unformatted binary format. The output data can then be converted into netCDF format with the interface program **netcdfint**. The same data are then written in two formats which is expensive in CPU time and disk space. This can be avoided by integrating the interface program within the COHERENS model itself.

5.2 Physics

The "core" part of the physical model consists of the Navier-Stokes and continuity equations, and the equations of temperature and salinity. The basic assumptions of vertical hydrostatic equilibrium and the Boussinesq approximation can be considered as valid for most oceanographic applications and no changes are anticipated in that respect. The "weak" parts of the physics are the parameterisation of turbulence and the formulation of the surface fluxes. A few items which may be considered in future developments, are listed below.

- As already mentioned in Section III-1.6.1c, surface waves have a non-negligible impact on turbulence. A formulation has been proposed by Craig and Banner (1994) using a surface flux of turbulence energy proportional to u_{*s}^3 and a surface roughness length. An alternative, currently under study, is the inclusion of an extra source term in the turbulence energy equation.

- An additional source of turbulence in stratified layers is the mixing produced by the shear and breaking of internal waves. An adequate scheme is still lacking. Formulations have been proposed using a coupled wave-turbulence model (Anis and Moum, 1995; Uittenbogaard, 1996), a limiting condition for the mixing length (Hassid and Galperin, 1983; Luyten and Rippeth, 1997), the mixing length formulation of Bougeault and André (1986) or an empirical relation (Large et al., 1994; Kantha and Clayson, 1994).

- Most turbulence schemes (including those implemented in COHERENS) assume the same value for the vertical diffusion coefficient of temperature and salinity despite the more rapid diffusion of temperature relative to salinity. Double diffusion effects are of importance when a vertically stable density gradient occurs as a result of the combination of an unstable (stable) salinity with a stable (unstable) temperature gradient. A formulation is proposed by Large et al. (1994) but needs further testing.

- Further improvements are expected in the surface flux formulation of momentum and heat, such as the influence of surface waves on the drag coefficient (e.g. Janssen, 1991), the Monin-Obukhov formulation for the exchange coefficients (see Section III-1.6.4). The formulae for the calculation of solar irradiance (Section III-1.6.5) use a number of empirical relations which need further validation.

- The parameterisation of horizontal sub-grid scale processes is still poorly understood. Since horizontal diffusion is mainly intended for the damping of (eventual) numerically unstable disturbances or other computational noise, new formulations with an improved numerical performance (e.g. Heathershaw et al., 1994; Deleersnijder and Wolanski, 1990) may be implemented in the future.

- Alternative forms for the equation of state can easily be implemented.

5.3 Biology and sediments

5.3.1 Pelagic biology

The microplankton-detritus model was designed in 1990 (Tett 1990a) to provide simple biology in a water quality model. It has compartments for microplankton (phytoplankton and pelagic bacteria and protozoa) and detritus (non-living particulate organic matter together with associated bacteria and protozoa). Each compartment is specified by amounts of carbon and nitrogen, the latter element being recycled through compartments for dissolved ammonium and nitrate. Photosynthesis and respiration produce or consume dissolved oxygen. Improvements to the algorithms and parameter values for the microplankton compartment, in particular, were made during the subsequent 8 years (Tett and Grenz 1994; Tett et al. 1993; Tett and Walne 1995), culminating in the version included in COHERENS (Tett 1998).

Several developments are under way or anticipated.

1. The present model evaluates microplankton growth as a threshold function of internal nutrient abundance and 24-hour mean irradiance, and so cannot adequately respond to a day-night cycle of light. We plan to investigate simple ways of parameterising a diurnal response with the aid of an additional variable that will accumulate photosynthetically fixed energy and employ it in growth.

2. The microplankton algorithms assume a constant ratio of microheterotrophs to microautotrophs. Observations at sea show that this ratio changes seasonally. Tett and Wilson (1999) show that a microplankton model with the heterotroph ratio η treated as a forcing variable can simulate seasonal cycles of chlorophyll as well as a more detailed microbial loop model. A two-microplankton model allows η to change dynamically, as two microplankton compartments compete for nutrients. In recently developed versions of the two-microplankton model, microplankton "1" represents a diatom-dominated spring association, with relatively few bacteria and protozoa; microplankton "2" represents a summer microbial loop community with a high heterotroph biomass and nitrogen recycling (Smith and Tett 1996; Tett and Smith 1997). Including a "diatomy" microplankton requires additional state variables to represent the cycling of silica through dissolved, diatomaceous, and detrital, phases.

3. Unlike some microbial loop models (Fasham et al. 1990), the microplankton-detritus model does not explicitly include dissolved organic matter (DOM). Labile DOM is recycled implicitly within the microplankton compartment; refractory DOM contributes implicitly to the optical properties of seawater by way of a relationship between "background" diffuse attenuation and salinity. It seems desirable to include a more explicit description of DOM dynamics.

4. In shallow waters, such as those of the North Sea, the microplankton model generates detritus as a result of simulated defecation by mesozooplankton grazing on microplankton, and by death of microplankton in the sea-bed fluff layer followed by resuspension. In deep water columns the second process is inapplicable, and mesozooplankton grazing is inadequate to generate observed amounts of non-planktonic particulate organic matter. To take account of this, we have recently extended a 2-microplankton model to include a simple parameterisation of shear-induced aggregation (Hill 1992; Jackson 1990), leading to the formation of rapidly sinking phytodetritus (Wild-Allen and Lampitt 1998). Further development of the aggregation and sinking rate algorithms are desirable. In addition, there is a need to improve the parameterisation of the mineralisation of phytodetritus and mesozooplankton-produced detritus. Future versions of COHERENS could include both these features, which greatly improve estimates of sedimenting organic material.

5. The microplankton-detritus model "closes" the nitrogen cycle at the level of mesozooplankton by imposing their grazing pressure as a forcing function. This is simple to implement, and aids simulation stability, but it does require a good time series of observations of mesozooplankton abundance. The model could be developed by allowing the part of mesozooplankton nitrogen that is grazed, but not subsequently defecated or excreted, to contribute to a semi-dynamic zooplankton nitrogen pool which in turn influences grazing pressure. By "semi-dynamic", it is meant that the mesozooplankton equation would include a loss term which would mimicking predation but would also compel the zooplankton nitrogen pool to relax towards observed zooplankton abundances, whenever these were available.

6. The existing two-microplankton model represent diatoms and small flagellates but not large autotrophic dinoflagellates such as Ceratium spp. or Gyrodinium aureolum, or the colonial prymnesiophyte Phaeocystis pouchetii. Ceratium are common in frontal and stratified waters of the North Sea during late Summer; Phaeocystis is abundant in mixed coastal waters in Spring (Tett et al., 1993). One means for dealing with these organisms, which warrant special treatment, would be to add extra variables to a 2-microplankton model: in the case of Gyrodinium, for example, an extra compartment would represent the carbon and nitrogen of this singular dinoflagellate, which would compete for nutrients and light with the mixed populations of "diatomy" and "flagellatey" microplanktons. The characteristic features of Gyrodinium (and some other economically important dinoflagellates) are strong vertical migration and resistance to grazing by protozoans or mesozooplankton (Tett, 1987b).

7. Finally, atmospheric input of nitrate (by way of acid rain) is important in some coastal seas. It may be desirable to allow this input in future versions of COHERENS.

5.3.2 Sea bed processes

The part of the COHERENS Eulerian model dealing with suspended and deposited particulate material, contains algorithms for:

- the sinking of fine inorganic suspended particulate material (iSPM), and of organic SPM which is microplankton and detritus, and its deposition to a sea-bed "fluff" layer;
- the resuspension of SPM from this layer as a result of stress due to near-bed flows;
- the conversion of fluff-layer microplankton to detritus, and the daily loss of a proportion of both, representing biological transfer to the consolidated sediment;
- the mineralisation of both fluff-layer organic components according to the parameterisation of the biological model;
- the simulated input of dissolved nutrients from the consolidated sea-bed to the lower water column, notionally representing benthic remineralisation, but in fact allowing deep water nutrient concentrations to relax slowly towards winter concentrations.

The last point makes clear the break in the nitrogen cycle; the nitrogen of microplankton and detritus notionally transferred to the consolidated sediment, does not necessarily return in the benthic nutrient flux. In addition, the fluff layer parameterisation may be too simple to describe, adequately, the range of resuspension conditions over a water body such as the North Sea. Two classes of improvement are foreseen.

1. Desirable improvements to the representation of physical processes involved in the transport of particles include: (a) the possibility of resuspension of consolidated sediment under extreme conditions; (b) improved fluff-layer resuspension; (c) the use of several populations of iSPM, with different optical properties and sinking speed; and (d) the introduction of sources of iSPM, for example from coastal erosion.

2. Descriptions of biological and related chemical processes might be improved in two ways. The first would involve adding a realistic but simple model for microbial diagenesis (nutrient remineralisation, oxygen demand, and denitrification) in the upper part of the consolidated sediment. We have in mind a thick-layer version of the "early diagenesis" model of Soetaert et al. (1996). The second would involve a simple macrobenthic model (Chardy and Dauvin 1992) to drive water and particulate exchange between the benthic boundary and fluff layers and the consolidated sediment.

5.3.3 Optical model

The present optical model divides solar radiation into two components: long-wave radiation, which is absorbed at the first grid-point, and PAR, which can penetrate further into the simulated sea. Light and heat absorption are computed from irradiances at the top and bottom of a layer, using attenuation coefficients influenced by the concentration of particulates.

It is desirable to introduce more realistic descriptions of submarine optics, including explicit representation of scattering and absorption by water, dissolved organics, and several types of iSPM and organic SPM. PAR should be divided amongst several (4–6) wavebands, to take account of the differences in spectral properties of different microplankters, detritus and iSPM (Kratzer et al. 1999). Such improvements will allow the simulation of sea surface reflection coefficients at several wavelengths and so allow model results to be compared with data from remote or in-water ocean colour sensors (Bowers et al. 1999).

5.4 Contaminant modules

The particle module, in its present form, can be used either to trace the evolution of individual particles or as the "Lagrangian" alternative of the Eulerian contaminant module. The following improvements can be made:

- The module is extended as an alternative of the Eulerian sediment module in COHERENS. Resuspension is performed in the same way using the "fluff" layer approach while deposition is implemented by defining an additional array of particle sinking velocities.

- The present version assumes that the masses of particles are constant in time. This can easily be extended by taking account of an exponential decay factor so that the module can be applied for the tracing of radioactive particles or substances.

The contaminant module allows input from rivers and across open sea boundaries. Input through surface fluxes can be taken into account if this input takes the general form given by equation (3.1.181).

Part IV

User Manual

Chapter 1

Structure of the Program

1.1 Layout of the program

A schematic structure of the COHERENS program is presented in Figures 4.1.1 and 4.1.2. The program has four major components: physics, biology, sediments and contaminants which are described below. The FORTRAN names of the corresponding modules are indicated in the text. The diagrams assume that all interactions between the modules takes place so that the program is used in its full capacity. Note that some of the links shown by arrows are suppressed if the corresponding switch has a non-zero value (see Section IV-2.2.1).

1.1.1 Physical model

The core of the physical part consists of a series of modules for the currents, a turbulence module (VEDDY1 if IOPTK = 1 or VEDDY2 if IOPTK = 2) and separate modules for salinity (SALT) and temperature (HEAT). The currents are updated at each time step using the following procedure (see Section III-4.3.1):

- A predicted velocity is obtained by solving the 3-D momentum equations (CRRNT3P).
- The 2-D momentum equations (CRRNT2) and the 2-D continuity equations (CONTNY) are solved for a number of times determined by the counter IC3D.
- The corrector method is applied to determine the values of $(u,\, v)$ at the new time step (CRRNT3C).
- The 3-D continuity equation is solved for the (transformed) vertical velocity (WCALC).

The turbulence module supplies the values of the eddy coefficients ν_T and λ_T needed to solve the transport equations for currents, temperature and salinity. The interaction is two way since turbulence depends on the vertical gradients of the current and the density. The currents, temperature, salinity and turbulence interact in a second way through the density module (DENSTY) which evaluates the baroclinic pressure gradient and the expansion coefficients β_S and β_T needed to evaluate the buoyancy production term in the turbulence energy models.

The central part of the whole program is the advection-diffusion module which has, for convenience, been split up into four different units:

- UCALC to solve the u-equation at U-nodes
- VCALC to solve the v-equation at V-nodes
- TRANSPC to solve a general advection-diffusion equation at cell centres
- TRANSPW to solve a general advection-diffusion equation at W-nodes

Boundary conditions are applied using different types of input:

- The open boundary values of the depth integrated current ($\overline{U}$, $\overline{V}$) are determined in BOUNDC. Input data are supplied in BC2IN (2-D mode) and HBCIN (3-D physics) and analogous routines for the biology, sediments, contaminants and the Lagrangian particle module.
- The surface fluxes of momentum, heat and salinity are evaluated in SURFLX using the meteorological data provided in METIN.
- Bottom stress is calculated in BSTRES. If wave-current interaction is applied, surface wave data are supplied in WAVIN.

Two optical modules are implemented in the program: IRRAD which evaluates the absorption of solar heat in the water column for the physics and BIOPT which evaluates photosynthetically active radiation (PAR) for the biology. Surface solar irradiance may either be supplied directly through input (METIN) or by the radiation formulae incorporated in the program (SOLRAD).

1.1.2 Biological model

The main biological routine (BIOLGY) evaluates the source and sink terms for water column biology (e.g. for microplankton carbon and nitrogen, detritus carbon and nitrogen, and dissolved nitrate, ammonium and oxygen). A schematic presentation of the biological model is shown in Figure 3.2.1. Most calculations of source and sink terms involve the evaluation of rates by calling a series of functions, of which the most important is GROWTH. This determines the microplankton specific growth rate as a function of the microplankton nitrogen to carbon ratio and photosynthetically active radiation. Finally, BIOLGY calls the advection-diffusion module TRANSPC for each biological state variable.

1.1.3 Sediment model

The sediment module deals with the (Eulerian) advection-diffusion of inorganic sediment (SEDEUL) and with processes related to the fluff layer (FLUFF). SEDEUL is the equivalent, for inorganic suspended particulate material, of BIOLGY. The main difference is that inorganic particles are conservative within the water column so that no source/sink terms need to be evaluated. FLUFF deals with biological processes in the fluff layer. SEDEUL and BIOLGY link with FLUFF at the bottom grid cell where FLUFF changes the concentrations as a result of resuspension from the fluff layer to the water column and a sinking from the water column to the fluff layer. In the case of organic particles (microplankton and detritus) there is an additional loss which is to be identified with the transfer to the consolidated sediment.

1.1.4 Contaminant model

The Eulerian contaminant component of the program (MASS) solves a number of advection-diffusion equations (TRANSPC) for a given number of passive contaminants. Initial concentrations are defined by the user (DEFICS). Open boundary input is supplied through a data file (CBCIN).

The Lagrangian particle transport module (SEDLAG) determines the trajectories of a given number of particles (see Figure 4.1.2). The initial positions are defined by the user (DEFICS). The position of each particle is updated at a new time step by calculating the distance travelled by the particle in the X-, Y- and Z-direction through advection and diffusion (MC3D). New particles can be created at river (MCROB) and open sea (MCOSB) boundaries. Open boundary input is supplied from a data file (PBCIN). The two contaminant models are linked to the physics via the current and turbulence modules. Only passive contaminants are considered so that there is no feedback to the physics (e.g. damping of turbulence, attenuation of light).

1.2 Program modules

The source code is spread over a number of files with the suffix *.f* each containing one or several FORTRAN (sub)programs. They are located in the **SOURCE** directory except for the file *defmod.f* created by the user in the **PROJECT** directory and the example file *readdat.f* in **/tests/examples**. As explained in Chapter IV-3 program output is defined by the user in *defout.f*, *defanal.f*, *defavr.f*, *defpar.f* and *print.f*. Note again that the versions of these files in **SOURCE** are empty and have no real significance for the program. A full list of all source code files and subprograms is given in Table 4.1.1. For a detailed description of each module the reader is referred to the Reference Manual.

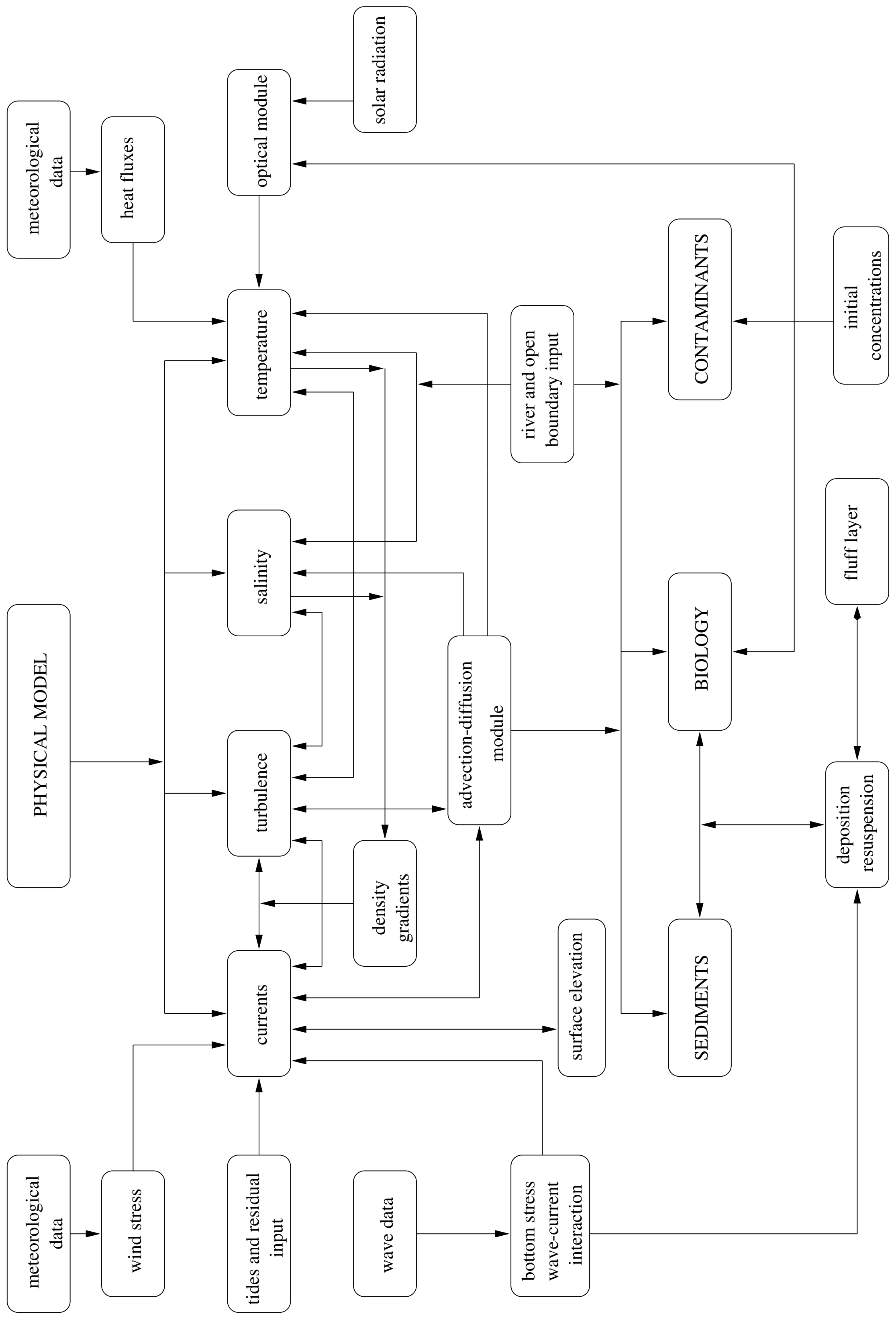


Figure 4.1.1: Schematic structure of the physical, biological and sediment parts of the program.

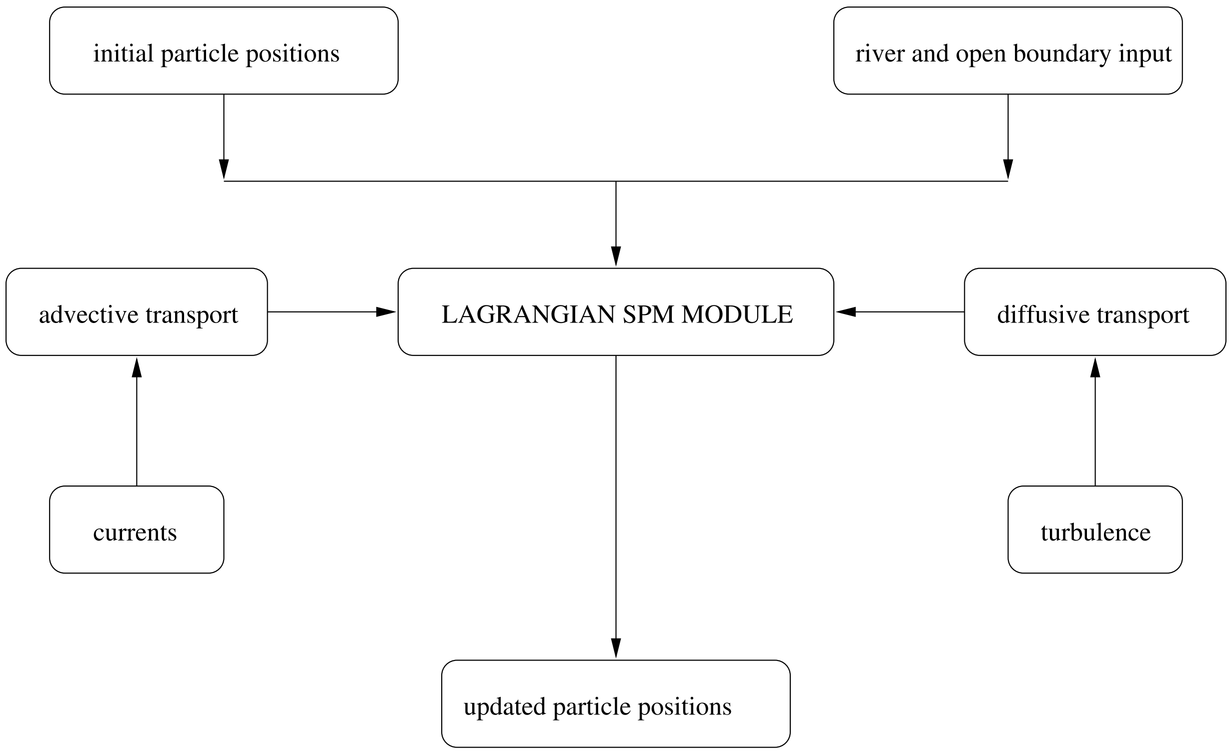

initial particle positions
river and open boundary input
advective transport
LAGRANGIAN SPM MODULE
diffusive transport
currents
turbulence
updated particle positions

Figure 4.1.2: Schematic structure of the Lagrangian particle module.

Table 4.1.1: List of all source code files and subprograms.

file	subprogram	type[a]	purpose
advdis.f			evaluates the horizontal (scalars) and vertical (momentum and scalars) advective and diffusive transport terms
	XADVDIS	S	evaluates the X-component of the advective-diffusive transport terms for scalars
	YADVDIS	S	evaluates the Y-component of the advective-diffusive transport terms for scalars
	ZADVDIS	S	evaluates the vertical advective and diffusive transport terms for scalars and momentum
analys.f			performs harmonic analysis
	ANALYS	S	performs harmonic analysis (main program unit)
	ANALYS0	S	initialises the arrays for harmonic analysis
	ANALYS1	S	updates the arrays for harmonic analysis
	ANALYS2	S	evaluates all harmonic parameters (including tidal ellipses) and writes the data to the appropriate output files
	LUDCMP	S	decomposes a matrix as the product of a lower and an upper triangular matrix using partial pivoting
	LUBKSB2	S	solves a linear system of equations using LU-decomposition for a “2-D” solution vector
	LUBKSB3	S	solves a linear system of equations using LU-decomposition for a “3-D” solution vector
bcsin.f			reads open boundary input
	BCSIN	S	reads the *obc*-file with specifications of the open boundary conditions
	BC2IN	S	reads the *2bc*-file with the open boundary data for the 2-D mode
	HBCIN	S	reads the *hbc*-file with the open boundary data for the physics
	BBCIN	S	reads the *bbc*-file with the open boundary data for the biological and sediment modules
	PBCIN	S	reads the *pbc*-file with the open boundary data for the Lagrangian particle module
	CBCIN	S	reads the *cbc*-file with the open boundary data for the contaminant module
biolgy.f	BIOLGY	S	solves the transport equations for the biological state variables
	BIOPT	S	evaluates PAR

[a]M: main program, S: subroutine, R: real function, I:integer function, C:character function

Table 4.1.1: continued.

file	subprogram	type	purpose
biolgy_in.f			series of functions used to evaluate different processes in the biological and sediment modules
	TEMPCH	R	temperature factor for biological rates
	QUOTA	R	ratio of microplankton nitrogen to carbon
	QDET	R	ratio of detrital nitrogen to detrital carbon
	UPTAKENH	R	microplankton biomass related ammonium uptake rate
	UPTAKENO	R	microplankton biomass related nitrate uptake rate
	GROWTH	R	microplankton growth rate
	RESPC	R	detrital carbon mineralisation rate
	RESPN	R	detrital nitrogen mineralisation rate
	AMMOXID	R	ammonium nitrification rate
	BSINKING	R	microplankton sinking rate
	DSINKING	R	detrital sinking rate
	SSINKING	R	sediment sinking rate
boundc.f	BOUNDC	S	evaluates the depth-integrated current at the open boundaries
bstres.f	BSTRES	S	evaluates the bottom stress
contny.f	CONTNY	S	solves the continuity equation (3.1.33) or (3.1.60) for the surface elevation
crrnt2.f			circulation module (2-D mode)
	CRRNT2	S	solves the depth-integrated momentum equation
	UDCALC	S	solves the $\overline{U}$-equation (3.1.34) or (3.1.61)
	VDCALC	S	solves the $\overline{V}$-equation (3.1.35) or (3.1.62)
crrnt3.f			circulation module (3-D mode)
	CRRNT3P	S	solves the horizontal momentum equations for the predicted velocity
	CRRNT3C	S	corrects the predicted velocity and applies the open boundary conditions for the horizontal current
	CRRNT1C	S	corrects the predicted velocity, calculates the depth-integrated velocity and evaluates the current at the open boundaries in the case of a 1-D application
	UCALC	S	solves the u-equation (3.1.25) or (3.1.50)
	VCALC	S	solves the v-equation (3.1.26) or (3.1.51)
	WCALC	S	solves the 3-D continuity equation (3.1.46) or (3.1.68) for the transformed velocity $J\tilde{w}$
	WPCALC	S	evaluates the physical vertical velocity w using (3.1.47) or (3.1.69)

Table 4.1.1: continued.

file	subprogram	type	purpose
defanal.f			defines the variables for harmonic output (user-defined)
	ANALPT3D	S	defines the 3-D quantities for harmonic analysis
	ANALPT2D	S	defines the 2-D quantities for harmonic analysis
	ANAL3DVAR	S	names, descriptions and units of the 3-D quantities for harmonic analysis
	ANAL2DVAR	S	names, descriptions and units of the 2-D quantities for harmonic analysis
	ANALFR	C	names of the analysed frequencies
defavr.f			defines the variables for time-averaged output (user-defined)
	AVRPT3D	S	defines the 3-D quantities for time-averaged output
	AVRPT2D	S	defines the 2-D quantities for time-averaged output
	AVR3DVAR	S	names, descriptions and units of the 3-D quantities for time-averaged output
	AVR2DVAR	S	names, descriptions and units of the 2-D quantities for time-averaged output
defmod.f			initialises the program (user-defined)
	DEFCON	S	defines the model constants and switches
	DEFGRID	S	defines the grid arrays
	DEFOB2D	S	defines the open boundary conditions and the (time-invariant) data for the 2-D mode (*obc*- and *2bc*-files)
	DEFOB3D	S	defines the open boundary conditions and the (time-invariant) open boundary data for all 3-D quantities (*obc*-, *hbc*-, *bbc*-, *pbc*- and *cbc*-files)
	DEFICS	S	defines the initial conditions for the *hin*-, *bin*-, *pin*- and *cin*-files
	WRFCDAT	S	writes the meteorological and wave data to the *met*- and *wav*-files
	WROBDAT	S	writes the open boundary data to the *2bc*-, *hbc*-, *bbc*-, *pbc*- and *cbc*-files at selected time intervals
defout.f			defines the variables for time series output (user-defined)
	OUTPT3D	S	defines the 3-D quantities for time series output
	OUTPT2D	S	defines the 2-D quantities for time series output
	OUT3DVAR	S	names, descriptions and units of the 3-D quantities for time series output
	OUT2DVAR	S	names, descriptions and units of the 2-D quantities for time series output

Table 4.1.1: continued.

file	subprogram	type	purpose
defpar.f	PARPT3D	S	defines particle output (user-defined)
densty.f	DENSTY	S	updates the baroclinic pressure gradient
	DENSTY1	S	updates the sea surface elevation and the pressure gradient in the case of a 1-D application using the formulation (3.1.83)–(3.1.85)
	SEARHO	S	evaluates the density ρ, the buoyancy b and the expansion coefficients β_S, β_T
dissip.f	DISSIP	S	solves the ε-equation (3.1.128)
	DISLEN	S	evaluates ε as function of l using (3.1.116)
hadvdis.f			evaluates the horizontal advective and diffusive terms in the 3-D and 2-D momentum equations
	XHAD3DU	S	evaluates the x_1-derivatives of the horizontal advective and diffusive fluxes in the u-equation (3.1.25) or (3.1.50) and in the depth integrals (3.1.42), (3.1.44) or (3.1.64), (3.1.66)
	YHAD3DU	S	evaluates the x_2-derivatives of the horizontal advective and diffusive fluxes in the u-equation (3.1.25) or (3.1.50) and in the depth integrals (3.1.42), (3.1.44) or (3.1.64), (3.1.66)
	XHAD3DV	S	evaluates the x_1-derivatives of the horizontal advective and diffusive fluxes in the v-equation (3.1.26) or (3.1.51) and in the depth integrals (3.1.43), (3.1.45) or (3.1.65), (3.1.67)
	YHAD3DV	S	evaluates the x_2-derivatives of the horizontal advective and diffusive fluxes in the v-equation (3.1.26) or (3.1.51) and in the depth integrals (3.1.43), (3.1.45) or (3.1.65), (3.1.67)
	HAD2DU	S	evaluates the advective and diffusive terms for the depth-integrated current in $\overline{U}$-equation (3.1.34) or (3.1.61)
	HAD2DV	S	evaluates the advective and diffusive terms for the depth-integrated current in $\overline{V}$-equation (3.1.35) or (3.1.62)
heat.f	HEAT	S	solves the temperature equation (3.1.28) or (3.1.53)
	IRRAD	S	evaluates the radiant heat absorption term in the temperature equation

Table 4.1.1: continued.

file	subprogram	type	purpose
heddy.f	HEDDY	S	evaluates the horizontal diffusion coefficients ν_H and λ_H
inicon.f			initialises model constants
	INICON	S	universal constants and constants for the turbulence module(s) except those defined in the *con*-file
	INIBIO	S	universal and derived parameters for the biological module
	RDCON	S	reads the *con*-file with model parameters and output specifiers
initc.f	INITC	S	reads model grid and initial conditions, initialises model arrays
integr.f			time-averaged output
	INTEGR	S	time-averaged output (main program unit)
	INTEGR0	S	initialises the arrays for time averaging
	INTEGR1	S	updates the arrays for time averaging
	INTEGR2	S	writes the time-averaged data to the appropriate output files
mainprog.f	MAINPROG	M	main program (COHERENS)
mass.f	MASS	S	solves the contaminant transport equations
metin.f			surface fluxes module
	METIN	S	reads the *met*-file with the meteorological forcing data
	SURFLX	S	evaluates the surface fluxes of momentum, heat and salinity using (3.1.168)–(3.1.179)
	CD	R	evaluates the surface drag coefficient C_D^s
	CHARNO	R	solves Charnock's relation (3.1.222) for the surface drag coefficient C_D^s by iteration
	CE	R	evaluates the exchange coefficient C_E (and C_H)
	FLUXCO	S	evaluates C_D^s and C_E at fixed intervals of wind speed and air-sea temperature difference by solving equations (3.1.252)–(3.1.254) iteratively
	HUMID	S	evaluates the specific humidity q and the vapour pressure pressure e using (3.1.176)–(3.1.177)
	SOLRAD	S	evaluates the short-wave radiation flux at the sea surface using the theory described in Section III-1.6.5

Table 4.1.1: continued.

file	subprogram	type	purpose
netcdfint.f			graphical interface program
	NETCDFINT	M	graphical interface program (main program unit)
	CDF2D	S	write 2-D output data in netCDF format
	CDF3D	S	write 3-D output data in netCDF format
	ERRCHECK	S	error handler for the graphical interface program
outpa.f	OUTPA	S	used by the main program to write the *hou-*, *bou-*, *pou-*, *cou*-files with the final conditions and by the preprocessor to write the *hin-*, *bin-*, *pin-*, *cin*-files with the initial conditions
	INPA	S	reads the initial conditions (*hin-*, *bin-*, *pin-*, *cin*-files)
outpt.f	OUTPT	S	writes time series output to the appropriate data files
preproc.f			preprocessor program (PREPROC)
	PREPROC	M	preprocessor program (main program unit)
	DEFAULT	S	defines default values for model parameters and arrays
	READIN	S	reads output specifiers from standard input (file *defmod.par* by default)
	DEFGRID1	S	defines the grid arrays in the case of a 1-D application
	INCTUR	S	initialises the turbulence arrays by default
	WRCON	S	writes the *con*-file with model parameters and output specifiers
	BCSOUT	S	writes the *obc*-file with the specifications of the open boundary conditions and the files with the open boundary data (*2bc-*, *hbc-*, *bbc-*, *pbc-*, *cbc*-files) if the corresponding counter is zero
print.f	PRINT	S	writes user-defined output
readdat.f			examples routines for reading user-defined output
	RDANAL	S	reading harmonic output (residuals, amplitudes, phases)
	RDAVR	S	reading time-averaged output
	RDELL	S	reading harmonic output (tidal ellipse parameters)
	RDOUT	S	reading time series output
	RDPAR	S	reading particle output
salt.f	SALT	S	solves the salinity equation (3.1.29) or (3.1.54)

Table 4.1.1: continued.

file	subprogram	type	purpose
sedeul.f			Eulerian sediment and fluff layer modules
	SEDEUL	S	solves the transport equation for the inorganic sediment concentration
	FLUFF	S	fluff layer processes (deposition and resuspension, biological interactions, losses to the consolidated sediment)
sedlag.f			particle module
	SEDLAG	S	particle module (main program unit)
	MCROB	S	generates particle input at river open boundaries
	MCOSB	S	generates particle input at open sea boundaries
	MC3D	S	updates the particle positions by advection and diffusion inside the computational domain
	STAUCH	S	updates the total number of particles inside the computational domain
	CONSPM	S	converts the particle distribution into a volume concentration
	OUTPAR	S	writes the positions of selected particles to the appropriate data file
thomv.f	THOMV	S	solves a tridiagonal system of linear equations
tkleng.f	TKLENG	S	solves the kl-equation (3.1.125)
	TLENDIS	S	evaluates l as function of ε using (3.1.116)
tleng.f	TLENG	S	determines the turbulence mixing length l using one of the algebraic mixing length formulations
transh.f	TRANSH	S	evaluates the horizontal grid spacings, the Coriolis frequency and other work space arrays
transp.f			advection-diffusion module
	TRANSPC	S	solves an advection-diffusion equation of the form (3.1.70) for scalar quantities located at cell centres
	TRANSPW	S	solves an advection-diffusion equation of the form (3.1.70) for scalar quantities located at W-nodes
transv.f	TRANSV	S	updates the vertical grid spacings (which determine the Jacobian J)
turben.f	TURBEN	S	solves the k-equation (3.1.114)
	PRODUC	S	evaluates the shear and buoyancy production terms in the transport equation for turbulence energy

Table 4.1.1: continued.

file	subprogram	type	purpose
veddy.f	VEDDY1	S	updates the eddy coefficients ν_T and λ_T using one of the algebraic formulations described in Section III-1.2.1
	VEDDY2	S	updates the eddy coefficients ν_T and λ_T using one of the turbulence schemes described in Section III-1.2.2
veddy_in.f			series of functions used in the turbulence modules
	BUOFR2	R	evaluates the squared buoyancy frequency N^2
	SHFR2	R	evaluates the squared shear frequency M^2
	FNTURF	R	evaluates the quantity k/l^2 in the zero-equation turbulence closure models using either (3.1.136) or (3.1.140)
	FNSMU	R	evaluates the stability coefficient S_m or S_u
	FNSHB	R	evaluates the stability coefficient S_h or S_b
	RICH	R	evaluates the Richardson number Ri
	FVEDMA	R	evaluates the factor $f_m(Ri)$ in the Munk-Anderson formulation (3.1.95)–(3.1.98)
	GVEDMA	R	evaluates the factor $g_m(Ri)$ in the Munk-Anderson formulation (3.1.95)–(3.1.98)
	ZLMAX	R	evaluates the upper limit (3.1.141) for the mixing length
	DISMIN	R	evaluates the lower limit (3.1.143) for the dissipation rate ε
wavin.f			wave-current interaction module
	WAVIN	S	reads the *wav*-file with the surface wave data
	WAVCUR	S	determines the bottom friction coefficient C_D^b using the wave-current interaction theory described in Section III-1.6.6
	WAVNUM	S	solves the linear dispersion relation (3.1.274) for the wave number
Dates.f			date/time routines
	NDIFF	S	calculates the number of seconds and time steps between two dates
	NEWTIM	S	updates the current date and time at a new time step
	DAYNUM	R	calculates the current day number of the year
	IDATES	S	determines the current date and time
	IGREG	S	converts a Julian date into the Gregorian calendar date
	IJUL	S	converts the Gregorian calendar date into the Julian date
	IDAADD	S	determines the new Julian date after adding a number of days to an initial Julian date
	LEAP	I	determines whether a given year is a leap year

Table 4.1.1: continued.

file	subprogram	type	purpose
Errors.f			error routines
	ERROR	S	stops execution of the program with an error message
	ERRMOD	S	checks whether model parameters are within the allowed range
	ERRGRD	S	checks whether grid parameters and arrays are within the allowed range and are mutually compatible
	ERRBCS	S	checks whether the arrays for the open boundary conditions are within the allowed range
	ERRICS	S	checks whether the initial parameters for the particle module are within the allowed range
Inout.f			I/O routines
	OPENF	S	opens a file for input or output
	CLOSEF	S	closes a file
	RDARR	S	reads a real array in COHERENS format
	RDARI	S	reads an integer array in COHERENS format
	WRARR	S	writes a real array in COHERENS format
	WRARI	S	reads an integer array in COHERENS format
	WROUT	S	writes an "Information"-file
	WRGRD	S	writes the *grd*-file with the grid parameters and arrays
	RDGRD	S	reads the *grd*-file with the grid parameters and arrays
	IGETFUN	I	returns the next available unit number for opening a file
Intp_in.f			series of interpolation routines (see Section V-1.12 of the Reference Manual)
Upwnd_in.f	FNLIM	R	computes the limiting function for the TVD scheme
Util.f	ZERO	S	sets all elements of a real 3-D array to zero
	IZERO	S	sets all elements of an integer 3-D array to zero
	CZERO	S	initialises a character array with blanks
	COMPLX	S	returns the amplitude and the phase of a complex number
	RANDOM	R	random generator
	RAN3	R	random generator
	LSQFIT	S	performs a linear regression analysis

1.3 Programming techniques

1.3.1 Portability

The program is written in FORTRAN 77 and has been checked against the ANSI Standard for FORTRAN 77 (American National Standard Programming Language FORTRAN, 1978). There are only two exceptions which can be considered as standard on most modern FORTRAN compilers:

1. The length of constant, variable, program, subroutine and function names sometimes exceeds the limit of 6 characters imposed by the ANSI standard. Symbolic names in the program are never longer than 10 characters and are only composed of letters and digits excluding special special characters such as "_" or ".".

2. INCLUDE statements are used for the inclusion of COMMON block statements (see below).

The ANSI requirement that all source lines — excluding comment lines — cannot extend beyond column 72, has been implemented.

To improve the portability of the code the following programming practices have been adopted.

- The order in which operators are evaluated in arithmetic expressions can differ on different machines, especially when optimisation options are used by the compiler. Significant rounding errors may result. This has been prevented as much as possible by the inclusion of parentheses at appropriate places to reduce rounding errors or by imposing lower limits to prevent division by a number which is too small.

- Rounding errors may occur by the conversion of REAL to INTEGER data types or vice versa. These problems are eliminated in the program (except in very unusual circumstances) provided that REAL and INTEGER data occupy four bytes of computer memory (see below).

- In standard FORTRAN variables either in the form of scalars or array elements must be defined before they can appear on the right of an assignment or in a WRITE statement. Scalar and array variables which are transferred to the main program via an input file, are initialised by a default value (usually taken to be zero) in subroutine DEFAULT. All other scalars and arrays referenced in a COMMON block statement and which may possibly be used in an assignment or WRITE statement are initialised in subroutine INITC of the main program. The same practice is adopted for local arrays except that array elements which are of no use to the program are not always defined. This prevents a secondary problem which may occur on compilers who save the last values of local variables after execution of the RETURN statement in a subprogram. If a variable (scalar or array) needs to be saved in a subprogram, its name appears in a SAVE statement in the declaration part of the subprogram.

- Care is taken that all 2-D and 3-D arrays defined on the model grid and referenced in a COMMON block have zero values on dry cells or closed interfaces.

- The names of constants, variables and external functions have been selected according to the default rules for implied integer and real typing. This means that the names of INTEGER data start with one of the letters [I-N] and the names of REAL data either with [A-H] or [O-Z]. Although this is in principle not necessary, all constants, variables and external functions are declared explicitly mainly for debugging purposes.

- In agreement with the rules of standard FORTRAN the statements in a (sub)program have the following order

  ```
  PROGRAM, SUBROUTINE or FUNCTION statement
  INCLUDE statements
  LOGICAL statements
  CHARACTER statements
  INTEGER type declarations (arrays, scalars, functions)
  REAL type declarations (arrays, scalars, functions)
  SAVE statements
  DATA statements
  executable statements
  ....
  STOP (main program) or RETURN (subprogram)
  FORMAT statements
  END statement
  ```

  If a subroutine or function is defined with arguments, the data types of the arguments are first declared followed by the types of the local variables. A series of statements may appear between the RETURN (STOP) and END statement. This has been implemented in a few program units which contain error traps. The part of the code beyond the RETURN (STOP) is only accessed by redirection of the program through a GOTO statement if an error is detected. The last statement line then calls subroutine ERROR which stops execution of the program.

- The program has been tested using default sizes of REAL, INTEGER and LOGICAL data which are machine-dependent. On most compilers REAL and INTEGER have a size of one word (4 bytes). The program could be made more portable by using a size declaration with a "$\star$"-format but this is unfortunately not allowed by the ANSI standard. It is recommended, if possible, to compile the program with a one-word size for REAL and INTEGER data. This may avoid problems which may occur in converting INTEGER data into REAL especially in the date/time routines NDIFF and NEWTIM (see below). The use of DOUBLE PRECISION REAL or INTEGER data has been avoided in the program except in the random generator function RANDOM.

- As is further explained in Section IV-4.1 the number of files which can be simultaneously connected to a program during its execution is limited. The maximum number is machine-dependent. On some machines this maximum can be increased by setting an appropriate environment variable. This limit is represented in the program by the parameter MAXFILES (see Section IV-2.2.5). Execution stops with the message

  ```
  Too many files connected to the program ...
  ```

  if the program attempts to open more files than allowed by the value of MAXFILES. Unit numbers must also be situated within a specific machine-dependent range. Before opening a file the program calls the function IGETFUN which returns the next available unit number (starting from number 22) for file connection. The program stops with the error message

  ```
  Invalid value for logical unit number ...
  ```

  if the unit number has a value not allowed by the processor.

- Intrinsic functions are always represented by their generic names. The program automatically determines the type of the result depending on the type of the argument.

- Comment lines are indicated by the character “C” in the first column.

- Continuation lines are indicated by a number in the sixth column.

1.3.2 COMMON blocks

The **SOURCE** directory contains besides the source files with the suffix *.f*, a series of files with the suffix *.inc*. The latter files are only composed of type declarations and COMMON statements (with exception of *param.inc*) which are incorporated in the program by an appropriate INCLUDE statement. A list of all *.inc*-files with a description of their purpose is given in Table 4.1.2. A detailed description of all COMMON block variables can be found in Chapter V-2 of the Reference Manual. To avoid compiling or optimisation problems on certain types of machines all the variables referenced in a COMMON statement belong to the same data type. For example, three COMMON blocks are defined in *anal.inc*: CANAL for CHARACTER data, IANAL for INTEGER data and RANAL for REAL data.

1.3.3 Layout of the arrays

The model uses an Arakawa C-grid with a σ-transformation in the vertical. This means that quantities related to the X-component of the current are evaluated at the U-nodes located at the centres of the lateral interfaces perpendicular to the X-direction, quantities related to the Y-component of the current are evaluated at the V-nodes located at the centres of the lateral interfaces perpendicular to the Y-direction and the transformed vertical velocity $J\tilde{w}$

Table 4.1.2: List of INCLUDE files

file	purpose
anal.inc	parameters and arrays for harmonic analysis
biopar.inc	model parameters for the biology
bounds.inc	arrays with the open boundary conditions and data
concn.inc	arrays for temperature, salinity, density, buoyancy, expansion coefficients β_S and β_T, and contaminant concentrations
consts.inc	universal constants
count.inc	counters
crrnts.inc	2-D and 3-D current fields and associated arrays
depths.inc	mean water depths
elev.inc	sea surface elevations
errorv.inc	workspace variables for error coding
grid.inc	grid parameters and arrays
met.inc	meteorological forcing data arrays
ncdfint.inc	parameters for the graphical interface program **netcdfint**
netcdf.inc	INCLUDE file from the netCDF library with parameters and declarations for netCDF routines
optics.inc	optical arrays
out.inc	output specifiers and parameters
param.inc	PARAMETER statement used for dimensioning of the program arrays (see Section IV-2.1)
partur.inc	turbulence switches
phycon.inc	physical model parameters
plnktn.inc	biological state variables and grazing pressures
runp.inc	model switches (except those for turbulence) and TITLE of the simulation
sedmnt.inc	arrays for the sediment and Lagrangian particle modules (e.g. concentrations, particle positions)
sedpar.inc	model parameters for the sediment and Lagrangian particle module
stress.inc	parameters and arrays related to the bottom stress and the surface fluxes of momentum, heat and salinity
tavout.inc	arrays for time-averaged output
tide.inc	tidal frequencies and initial phases
time.inc	date/time parameters and time steps
tsout.inc	arrays for time series output
turbke.inc	turbulence arrays
turcon.inc	turbulence model parameters
visc.inc	horizontal and vertical eddy viscosity and diffusion coefficients
waves.inc	surface wave data
worksp.inc	empty workspace arrays used as dummy arguments in subroutine calls

is evaluated at the W-nodes located at the centres of the horizontal interfaces. Turbulence quantities such as the eddy coefficients ν_T and λ_T, turbulence energy k, mixing length l, dissipation rate ε are evaluated at the W-nodes. All other scalar quantities (salinity, temperature, biological state variables, sediment and contaminant concentrations, ...) are specified at the cell centres. The components of the surface stress and wind velocity and all 2-D scalar quantities are determined at the centres of the horizontal grid. The components of all other 2-D vectors such as $(\overline{U}, \overline{V})$, (τ_{bx}, τ_{by}), ... are given at the U- or V-nodes of the horizontal grid. For more details see Section III-4.2.1 and Figure 3.4.1.

Most DO loops in the program are organised such that the leftmost index of an array varies the fastest[1]. This is illustrated by the following typical code example representing a series of calculations to be performed on all interior grid points:

```
      DO 110 I=1,NC
      DO 110 J=1,NR
         IF (NWD(J,I).EQ.1) THEN
            DO 111 K=1,NZ
               ...
               Q(K,J,I) = ...
 111        CONTINUE
         ENDIF
 110  CONTINUE
```

where NWD only has the value of 1 on wet grid cells and Q is a 3-D array evaluated at the centre of the cell.

The dimensions of an array defined on the model grid are then given by:

NZ, NR, NC+1 for a 3-D quantity evaluated at the U-nodes

NZ, NR+1, NC for a 3-D quantity evaluated at the V-nodes

NZ+1, NR, NC for a 3-D quantity evaluated at the W-nodes

NZ+1, NR, NC+1 for a 3-D quantity evaluated at the X-nodes

NZ+1, NR+1, NC for a 3-D quantity evaluated at the Y-nodes

NZ, NR, NC for a 3-D quantity evaluated at the cell centres

NR,NC+1 for a 2-D quantity evaluated at the U-nodes

NR+1,NC for a 2-D quantity evaluated at the V-nodes

NR,NC for a 2-D quantity evaluated at the cell centres.

For more details see Section III-4.2.2.

[1]Tests showed that this method saves CPU time on machines using scalar optimisation but may be less efficient on vector machines.

1.3.4 Program exits

There are three possible ways to end the execution of the program.

- The program stops by the STOP command encountered at the end of the main program unit of PREPROC or COHERENS. No detectable error occurred. This can be verified by inspecting the last line of the "log"-file which should read:

  ```
  Preprocessor terminated
  ```

  or

  ```
  Main program terminated
  ```

- The error code implemented in the program detects an error. The program is exited by the STOP statement in subroutine ERROR. One or more error messages are issued.

- The program stops due to to a run-time error detected by the machine or crashes by a system failure (no disk space available, power cutoff, ...). In the former case the type of error is usually explained by a displayed error message. A frequent problem is the occurrence on some machines of a "floating underflow" condition in the turbulence module of the program. If this should occur, it is recommended to instruct the compiler to ignore underflows and to replace the underflowed values by a "0".

1.3.5 Internal documentation

The program modules are internally documented following most of the programming standards of the "DOCTOR" system (Gibson, 1982). All subroutines contain comments concerning their purpose, last update, author, the name of the calling program and any accessed subroutines and a reference to the appropriate section(s) of the User Documentation. Local variables are documented with their name, type, purpose and unit. A similar procedure is followed for the COMMON variables defined in the *.inc*-files.

Chapter 2

Preparing Model Input

To initialise the model a series of scalar and array parameters must or may be defined by the user. Firstly, a file *param.inc* which has the form of a FORTRAN PARAMETER statement, must be supplied. Secondly, the structure of the program requires that the user creates a FORTRAN file *defmod.f* containing the following seven subroutines:

DEFCON, DEFGRID, DEFOB2D, DEFOB3D, DEFICS, WRFCDAT, WROBDAT

which are called successively by the preprocessor **preproc**.

The purpose of this chapter is to

- provide guidelines to write these FORTRAN files
- explain the meaning of all input parameters
- provide guidelines for selecting appropriate values for model parameters

Specific instructions for initialising a 1-D application are given in Section IV-2.9. All scalar and array parameters which can be defined in *param.inc* and *defmod.f* are listed in a series of tables at the end of the chapter.

2.1 File *param.inc*

The constants defined in this PARAMETER statement are integers used in the program for the dimensioning of arrays and must always be defined by the user. The parameters are further listed in Table 4.2.1.

NC

The number of grid cells in the X-direction. The C-grid requires a minimum value of 2 (even for 1-D simulations). Note that NC must be set to 2 in the case of a 1-D application (IGRDIM = 1).

NR

The number of grid cells in the Y-direction. The C-grid requires a minimum value of 2 (even for 1-D simulations). Note that NR must be set to 2 in the case of a 1-D application (IGRDIM = 1).

NZ

The number of grid cells in the vertical. Minimum value is 2 although a larger value is required for a realistic simulation.

NOBU

The number of open boundary points which are located at a U-node. A zero value is allowed. Note that all boundaries are open in the case of a 1-D application (IGRDIM = 1) so that NOBU must be set to 4 in that case.

NOBV

The number of open boundary points which are located at a V-node. A zero value is allowed. Note that all boundaries are open in the case of a 1-D application (IGRDIM = 1) so that NOBV must be set to 4 in that case.

NVPROF

The maximum number of distinct vertical profiles supplied by the user at all open boundary locations. If NOBU = NOBV = 0, NVPROF can be set to 0. Maximum value is NOBU + NOBV. More details are given in Section IV-2.5.

NCON

The number of tidal constituents. A zero value is allowed.

NCONTO

The number of frequencies for which harmonic analysis is applied. Its value may be different from NCON. To prevent that an error occurs during compilation NCONTO must always be strictly positive. If the switch IOUTT is set to 0, harmonic analysis is not performed and it is recommended to set NCONTO equal to 1. The maximum number of frequencies for which output can be written by the program, equals 9 so that NCONTO should not exceed this value.

NCONC

The number of concentrations used in the (Eulerian) contaminant module. A zero value is allowed. Note that contaminant transport is disabled if the switch IOPTC equals 0 in which case NCONC should be zero.

MAXNOP

The maximum allowed number of particles which can be used in the Lagrangian suspended particle module. A zero value is recommended if the switch IOPTP = 0. A sufficiently large value should be selected since the total number of particles inside the computational domain may increase or decrease during the execution of the program. As explained in Section III-4.8.5 new particles may be created at the open sea and river boundaries whereas particles which reach the bottom or are advected across a land or open sea boundary are lost to the system. The number of particles inside the domain at each instant is given by the program variable NUMP. An error occurs and program execution stops if NUMP exceeds MAXNOP.

NOUTMAX
The maximum allowed number of 2-D and 3-D output fields which can be selected for time series output. This number must be positive and equal to or larger than

```
MAX(2*N2VECO+N2SCALO,3*N3VECO+N3SCALO)
```

where N2VECO, N2SCALO, N3VECO, N3SCALO are defined in the parameter input file *defmod.par* and represent respectively the number of 2-D vector, 2-D scalar, 3-D vector and 3-D scalar output fields defined in *defout.f.* Note that time series output is disabled if the switch IOUTS $= 0$ in which case NOUTMAX should be set to its minimum value of 1.

NANALMAX
The maximum allowed number of 2-D and 3-D output fields on which harmonic analysis is performed. To prevent an error occurring during compilation the allowed minimum value is 1. This number must be positive and equal to or larger than

```
MAX(2*N2VECA+N2SCALA,3*N3VECA+N3SCALA)
```

where N2VECA, N2SCALA, N3VECA, N3SCALA are defined in the parameter input file *defmod.par* and represent respectively the number of 2-D vector, 2-D scalar, 3-D vector and 3-D scalar output fields defined in *defanal.f.* Note that harmonic analysis is disabled if the switch IOUTT $= 0$ in which case NANALMAX should be set to 1.

NAVRMAX
The maximum allowed number of 2-D and 3-D output fields which can be selected for time-averaged output. This number must be positive and equal to or larger than

```
MAX(2*N2VECM+N2SCALM,3*N3VECM+N3SCALM)
```

where N2VECM, N2SCALM, N3VECM, N3SCALM are defined in the parameter input file *defmod.par* and represent respectively the number of 2-D vector, 2-D scalar, 3-D vector and 3-D scalar output fields defined in *defavr.f.* Note that time-averaged output is disabled if the switch IOUTF $= 0$ in which case NAVRMAX should be set to its minimum value of 1.

MAXOUT

The maximum allowed number of output files used for time series output. This number must be situated between 1 and 9 and equal to or larger than the parameter NARROUT defined in the parameter input file *defmod.par* and which represents the "actual" number of time series output files. Note that time series output is disabled if the switch IOUTS = 0 in which case MAXOUT should be set to its minimum value of 1.

MAXAVR

The maximum allowed number of output files for time-averaged output. This number must be situated between 1 and 9 and equal to or larger than the parameter NARRAVR defined in the parameter input file *defmod.par* and which represents the "actual" number of time-averaged output files. Note that time averaging is disabled if the switch IOUTF = 0 in which case MAXAVR should be set to its minimum value of 1.

Some of the above parameters may take the value of zero. If the upper bound of an array dimension in the program is given by one of these parameters, its lower bound is set to zero, e.g. IOBU(0:NOBU) to prevent an error occurring during compilation. Unless explicitly stated in the text below, an array element with a zero index has no further significance for the program.

2.2 Subroutine DEFCON

The aim of this routine is to provide values for a series of "control" parameters (e.g. dates, time steps, counters, switches, file parameters and model parameters for the physics, biological, sediment and particle modules). All switches and parameters are listed in Table 4.2.2 and 4.2.3 with their FORTRAN name, data type, default value and a short description of their purpose. A detailed description is given below.

2.2.1 Switches

There are 33 switches available in the program. They are all of data type INTEGER and can be divided into the following categories:

- switches to activate or deactivate a particular module or to select a specific formulation
- switches to select the type of the grid
- switches to select a numerical scheme for advection or horizontal diffusion
- switches to select a particular type of boundary condition
- switches to select a turbulence model

- switches to select the type of output

All program switches are listed in Table 4.2.2. The first column lists the FORTRAN name, the second the allowed values, the third the default value. A short description of its purpose is given in the fourth column. Each program switches is described below in alphabetical order.

IADVC

Selects the type of advection scheme for the momentum equations.

0 : Horizontal and vertical advection is disabled.

1 : Horizontal and vertical advective fluxes are determined with the first order upwind scheme.

2 : Advective fluxes are determined with the Lax-Wendroff scheme in the horizontal and the central scheme in the vertical.

3 : The advective fluxes are evaluated with the TVD scheme using a weighted average between the upwind and Lax-Wendroff fluxes in the horizontal and between the upwind and central fluxes in the vertical. The weight factor is given by the superbee limiting function.

4 : The same meaning as the value 3 except that the weight factor is given by the monotonic limiting function.

The value 2 should not be used except for demonstration purposes (e.g. intercomparison study of advection schemes). The upwind scheme is computationally less expensive. The TVD scheme is recommended for the simulation of areas with strong horizontal shear such as river fronts, meandering and horizontal eddies, at the expense of a larger CPU time. Note that IADVC is set to 0 by the program in the case of a 1-D application (IGRDIM $= 1$).

IADVS

Selects the type of advection scheme for solving all scalar transport equations.

0 : Horizontal and vertical advection is disabled.

1 : Horizontal and vertical advective fluxes are determined with the first order upwind scheme.

2 : Advective fluxes are determined with the Lax-Wendroff scheme in the horizontal and the central scheme in the vertical.

3 : The advective fluxes are evaluated with the TVD scheme using a weighted average between the upwind and Lax-Wendroff fluxes in the horizontal and between the upwind and central fluxes in the vertical. The weight factor is given by the superbee limiting function.

4 : The same meaning as the value 3 except that the weight factor is given by the monotonic limiting function.

The value 2 should not be used except for demonstration purposes (e.g. intercomparison study of advection schemes). The upwind scheme is computationally less expensive but has the disadvantage of producing large numerical diffusion. The TVD scheme is recommended for the simulation of areas with strong frontal gradients. Advection is only enabled in the turbulence transport equations if both IAHDHT $= 1$ and IADVS > 0. Note that IADVS is set to 0 by the program in the case of a 1-D application (IGRDIM $= 1$).

IADVWB

Selects the type of advection scheme for the vertical sinking term in the biological and sediment transport equations if IADVS equals 0.

0 : Vertical advection is disabled.

1 : Vertical advective fluxes are determined with the first order upwind scheme.

2 : Vertical advective fluxes are determined with the central scheme.

3 : The vertical advective fluxes are evaluated with the TVD scheme using a weighted average between the upwind and central schemes and the superbee limiting function as weight factor.

4 : The same meaning as the value 3 except that the weight factor is given by the monotonic limiting function.

The purpose of introducing this switch is to make allowance for the advection of biological state variables by an appropriate sinking velocity when the model is used for a 1-D simulation (IGRDIM $= 1$). This switch is only used by the program when IADVS $= 0$ in which case the horizontal and vertical advective terms due to the physical current are neglected. Note that if IADVS is non-zero, all advective terms are evaluated using the scheme selected by IADVS and the value of IADVWB is no longer relevant for the program.

IAHDHT

Disables or enables the evaluation of the advective and horizontal diffusion terms in the turbulence transport equations (3.1.114), (3.1.125), (3.1.128).

0 : Advection and horizontal diffusion are disabled in the turbulence equations.

1 : Advection and horizontal diffusion are enabled in the turbulence equations.

Advection and horizontal diffusion of turbulence can be neglected in most applications so that the default value of 0 should normally not be changed. Note that advection is only enabled in the program if the switch IADVS is non-zero. In the same way horizontal diffusion is only enabled in the turbulence equations if IODIF is non-zero. Note also that IAHDHT is set to 0 in the case of a 1-D application (IGRDIM $= 1$).

IBSTR

Selects the formulation for the bottom stress.

0 : A zero bottom stress is used.

1 : The bottom stress is evaluated using the quadratic friction law (3.1.185).

2 : The bottom stress is evaluated using the linear friction law (3.1.186).

The default value of 1 should not be changed except for specific test simulations.

IDRAG

Selects the formulation for the neutral surface drag coefficient C_D^s used in the surface boundary condition (3.1.168) for horizontal momentum.

0 : constant value given by (3.1.218)

1 : Large and Pond (1991) formulation (3.1.219)

2 : Smith and Banke (1975) formulation (3.1.220)

3 : Geernaert et al. (1996) formulation (3.1.221)

4 : Charnock (1955) formulation (3.1.222)

If IDRAG is non-zero, C_D^s is expressed as function of the wind speed $|U_{10}|$. An additional dependence on the air-sea temperature difference can be included as well by taking ITDIF equal to 1. Note that IDRAG is not used by the program if IOPTM $= 0$ or 1.

IFLUFF

Disables or enables the "fluff-layer" component of the biological module.

0 : Sinking and resuspension of microplankton and detrital concentrations are disabled.

1 : Sinking and resuspension of microplankton and detrital concentrations are enabled.

If IFLUFF $= 1$, the switch IOPTS must also be set to 1. Note that the "fluff-layer" cannot be disabled in the sediment module unless IOPTS $= 0$.

IGRDIM

Selects the dimensions of the numerical grid.

1 : 1-D application as described in Section III-1.1.4

3 : 3-D application

Note that, if IGRDIM = 1, a number of model parameters and arrays must have appropriate values. See Section IV-2.9 for details.

IGTRH

Selects the type of the numerical grid and the coordinate system.

0 : Cartesian grid and coordinates.

1 : Spherical grid and coordinates.

ILENG

Selects the formulation for the turbulence mixing length if IOPTK = 2.

1 : parabolic law (3.1.131).

2 : "quasi-parabolic" law (3.1.132)

3 : "Xing" formulation given by (3.1.131) and (3.1.133)

4 : "Blackadar" formulation (3.1.134)–(3.1.135)

If NTRANS = 2, ILENG is only used by the preprocessor to initialise the mixing length at the start of the program but has no further relevance to the main program.

ILIM

Disables or enables the use of limiting conditions for turbulence variables if IOPTK = 2.

0 : Limiting conditions are disabled.

1 : Limiting conditions are enabled.

IODIF

Selects the type of scheme for horizontal diffusion in all (momentum and scalar) transport equations.

0 : Horizontal diffusion is disabled.

1 : Horizontal diffusion is enabled. Uniform values are taken for the diffusion coefficients ν_H and λ_H.

2 : Horizontal diffusion is enabled. The diffusion coefficients ν_H and λ_H are evaluated using the formulation (3.1.151) or (3.1.153).

Note that horizontal diffusion of turbulence is only enabled if both IAHDHT = 1 and IODIF > 0.

IOPTB

Disables or enables the activation of the biological module.

0 : The biological module is not activated.

1 : The biological module is activated.

Note that if IOPTB = 1, solar irradiance must be available to the program. This means that IOPTM must be larger than 1.

IOPTC

Disables or enables the activation of the contaminant module.

0 : The contaminant module is not activated.

1 : The contaminant module is activated but without horizontal and vertical diffusion.

2 : The contaminant module is activated with horizontal and vertical diffusion enabled.

IOPTD

Selects the equation of state and the formulation for the expansion coefficients β_S and β_T.

0 : The density takes the uniform value ρ_0; β_S and β_T are set to 0.

1 : The density is calculated from the linear equation of state (3.1.162). Uniform values are taken for β_S and β_T.

2 : The density and the coefficients β_S and β_T are evaluated using the general equation of state, as given by (3.1.163)–(3.1.166).

Note that all stratification effects are cancelled in the momentum equations if IOPTD = 0.

IOPTHE

Selects the type of solution for the temperature equation.

0 : The temperature field is initialised at the start of the program but the temperature equation is not solved.

1 : The temperature is solved assuming that all solar radiation is absorbed at the sea surface. This means that in the surface boundary condition (3.1.170) $Q_s = -Q_{nsol} + Q_{rad}$ and the absorption term $\partial I/\partial\tilde{x}_3$ is taken to be zero in the temperature equation (3.1.28) or (3.1.53).

2 : The temperature equation is solved with solar radiation absorbed within a surface layer using the theory described in Section III-1.4. This means that in the surface boundary condition (3.1.170) $Q_s = -Q_{nsol}$ and the absorption term $\partial I/\partial\tilde{x}_3$ is retained in the temperature equation (3.1.28) or (3.1.53).

If IOPTHE = 2, a series of optical parameters (see below) need to be defined. When the program is applied to study the evolution of a seasonal thermocline, the temperature equation should not be solved with IOPTHE = 1 since this may largely underpredict the depth of the thermocline and overestimates the sea surface temperature. The only advantage is that this value avoids the definition of the optical parameters.

IOPTK

Determines the general type of the turbulence scheme.

0 : The eddy coefficients ν_T and λ_T are uniform in space and time.

1 : The eddy coefficients ν_T and λ_T are determined using one of the algebraic formulations described in Section III-1.2.1. A further selection is made with the switch ITFORM.

2 : The eddy coefficients ν_T and λ_T are determined using one of the turbulence closure schemes discussed in Section III-1.2.2. A further selection is made with the switches NTRANS, ITCPAR, ISTPAR, ILENG, ILIM, IAHDHT.

To suppress vertical diffusion of momentum, temperature, salinity and sediment concentrations completely one has to set IOPTK $= 0$, VISMOL $= 0.0$ ($\nu_T = 0.0$), DIFMOL $= 0.0$ ($\lambda_T = 0.0$), IOPTM $= 0$ (zero surface stress) and IBSTR $= 0$ (zero bottom stress). Note that this is not possible for the biological parameters due to the presence of non-zero fluxes at the surface and the bottom. Vertical diffusion is only enabled in the Lagrangian particle module if IOPTP $= 2$ and in the contaminant module if IOPTC $= 2$.

IOPTM

Determines the type of meteorological forcing.

0 : Surface stress, heat fluxes, salinity fluxes and solar radiation are set to zero.

1 : Surface stress is constant in space and time. Heat fluxes, salinity fluxes and solar radiation are set to zero. The uniform components of the surface stress are supplied by the user.

2 : Surface stress, heat fluxes, salinity fluxes and solar radiation are uniform in space but non-uniform in time. Surface stress, heat fluxes and salinity fluxes are evaluated using (3.1.168) and (3.1.170)–(3.1.179), solar radiation by the formulation described in Section III-1.6.5. Spatially uniform values for the two wind components U_{10} and V_{10}, the air temperature T_a, the relative humidity RH, the cloud coverage f_c and either the evaporation minus precipitation rate $E_{vap} - R_{pr}$ (IOPTSA $= 0, 1$) or the precipitation rate R_{pr} (IOPTSA $= 2$) are supplied by the user at time intervals selected by ICMET.

3 : Surface stress, heat fluxes, salinity fluxes and solar radiation are non-uniform in space and time. Surface stress, heat fluxes and salinity fluxes are evaluated using (3.1.168) and (3.1.170)–(3.1.179), solar radiation by the formulation described in Section III-1.6.5. Arrays of values for the two wind components U_{10} and V_{10}, the atmospheric pressure P_a,the air temperature T_a, the relative humidity

RH, the cloud coverage f_c and either the evaporation minus precipitation rate $E_{vap} - R_{pr}$ (IOPTSA $= 0, 1$) or the precipitation rate R_{pr} (IOPTSA $= 2$) are supplied by the user at each horizontal grid point and at time intervals selected by ICMET.

4 : Surface stress, heat fluxes, salinity fluxes and solar radiation are uniform in space but non-uniform in time. The surface stress is evaluated using (3.1.168). Spatially uniform values for the two wind components U_{10} and V_{10}, the non-solar heat flux, solar radiation and the evaporation minus precipitation rate $E_{vap} - R_{pr}$ are supplied by the user at time intervals selected by ICMET.

5 : Surface stress, heat fluxes, salinity fluxes and solar radiation are non-uniform in space and time. The surface stress is evaluated using (3.1.168). Arrays of values for the two wind components U_{10} and V_{10}, the atmospheric pressure P_a, the non-solar heat flux, solar radiation and the evaporation minus precipitation rate $E_{vap} - R_{pr}$ are supplied by the user at each horizontal grid point and at time intervals selected by ICMET.

If IOPTB $= 1$, the solar heat flux at the surface must be evaluated even when IOPTHE $= 0$. This means that IOPTM must be larger than 1 in that case. Options 4 and 5 have been implemented to permit the user to make use of an alternative formulation for the heat fluxes or to incorporate "in situ" measured values. The forcing data, whose number depends on the value of IOPTM, must always be supplied even when they are of no use to the program. In that case a zero value must be substituted. For example, if the salinity equation is not solved (IOPTSA $= 0$) or solved with a zero surface flux (IOPTSA $= 1$), a zero value for the salinity flux must be substituted if IOPTM is larger than 1. Similarly, the atmospheric pressure P_a needed to evaluate the barotropic pressure gradient term in the momentum equations, can be set to zero in many applications but must be supplied if IOPTM $= 3$ or 5.

IOPTP

Selects the type of solution for Lagrangian particle transport.

0 : The Lagrangian particle module is not activated.

1 : The Lagrangian particle module is activated but without horizontal and vertical diffusion.

2 : The Lagrangian particle module is activated with horizontal and vertical diffusion enabled.

IOPTS

Disables or enables the activation of the sediment module.

0 : The sediment module is not activated.

1 : The sediment module is activated.

Note that if IOPTS = 0, the resuspension of biological material is no longer included in the program even when IFLUFF = 1.

IOPTSA

Selects the type of solution for the salinity equation.

0 : The salinity field is initialised at the start of the program but the salinity equation is not solved.

1 : The salinity equation is solved with the surface flux evaluated by (3.1.179) and the evaporation minus precipitation rate supplied by the user.

2 : The salinity equation is solved with the surface flux evaluated using (3.1.179) and the precipitation rate supplied by the user.

If IOPTSA = 2, IOPTM must be equal to 2 or 3.

IOPTW

Disables or enables wave-current interaction and selects the form of the wave input.

0 : The wave-current interaction module is not activated.

1 : The wave-current interaction module is activated. Wave height h_s and period T_w are uniform in space and time, and are supplied by the user.

2 : The wave-current interaction module is activated. Wave height h_s and period T_w are uniform in space and non-uniform in time. They are supplied by the user at time intervals selected by ICWAV.

3 : The wave-current interaction module is activated. Wave height h_s and period T_w are non-uniform in space and time. Arrays of values are supplied by the user at each horizontal grid point and at time intervals selected by ICWAV.

IOPT2

Disables or enables the solution of the depth-averaged continuity equation (3.1.33) or (3.1.60) and the 2-D momentum equations (3.1.34)–(3.1.35) or (3.1.61)–(3.1.62).

0 : The depth-averaged continuity and momentum equations are not solved (mode-splitting disabled).

1 : The depth-averaged continuity and momentum equations are solved (mode-splitting enabled).

The value 0 should only be used for:

- an application without currents (such as the **batch** test case) in which case the switch IOPT3 has to be set to 0 as well
- a 1-D application (IGRDIM = 1) to disable the mode-splitting

- special applications where the (3-D) currents are specified externally (test case **front**) or are stationary (test case **cones**)

IOPT3

Disables or enables the solution of the 3-D momentum equations (3.1.25)–(3.1.26) or (3.1.50)–(3.1.51) for the horizontal currents (u,v) and the continuity equation (3.1.46) or (3.1.68) for the transformed vertical current $J\tilde{w}$.

0 : The 3-D momentum and continuity equations are not solved.

1 : The 3-D momentum and continuity equations are solved.

The value 0 should only be used for:

- an application without currents (such as the **batch** test case) in which case the switch IOPT2 has to be set to 0 as well
- stationary problems such as the test case **cones**

Note that, if IOPT3 $= 0$, the 3-D current is initialised at the start of the program but not updated in time.

IOUTF

Disables or enables time-averaged output.

0 : Time-averaged output is disabled.

1 : Time-averaged output is enabled.

Time-averaged output is further discussed in Sections IV-3.1 and IV-3.4.

IOUTP

Disables or enables the output of the particle positions from the Lagrangian transport module.

0 : Particle output is disabled.

1 : Particle output is enabled.

Particle output is further discussed in Sections IV-3.1 and IV-3.5.

IOUTS

Disables or enables time series output.

0 : Time series output is disabled.

1 : Time series output is enabled.

Time series output is further discussed in Sections IV-3.1 and IV-3.2.

IOUTT

Disables or enables harmonic analysis and selects the type of output.

0 : Harmonic analysis is not performed. No output is written.

1 : Harmonic analysis is performed. Harmonically analysed values of the quantities defined by the user are written to the appropriate output files.

2 : Harmonic analysis is performed. Harmonically analysed values of the quantities defined by the user and the tidal ellipse parameters are written to the appropriate output files.

Harmonic output is further discussed in Sections IV-3.1 and IV-3.3.

ISTPAR

Selects whether the stability functions in the turbulence closure are evaluated as function of the stability parameter or the Richardson number.

1 : The stability functions are expressed in terms of the stability parameter using either (3.1.118)–(3.1.119) if ITCPAR $= 1$ or using (3.1.120)–(3.1.121) if ITCPAR $=$ 2.

2 : The stability functions are expressed in terms of the Richardson number using either (3.1.122) if ITCPAR $= 1$ or using (3.1.123) if ITCPAR $= 2$.

This switch is only used if IOPTK $= 2$.

ITCPAR

Selects the type of the turbulence scheme if IOPTK $= 2$.

1 : The $k-l$ formulation is selected with ν_T, λ_T given by (3.1.115) and the stability functions evaluated either by (3.1.118)–(3.1.119) if ISTPAR $= 1$ or by (3.1.122) if ISTPAR $= 2$. The mixing length is evaluated either by a formulation selected by ILENG if NTRANS $= 0, 1$ or by solving the kl-equation (3.1.125) if NTRANS $= 2$.

2 : The $k-\varepsilon$ formulation is selected with ν_T, λ_T given by (3.1.117) and the stability functions evaluated either by (3.1.120)–(3.1.121) if ISTPAR $= 1$ or by (3.1.123) if ISTPAR $= 2$. If NTRANS $= 0$ or 1, ε is determined from (3.1.116) with the mixing length evaluated using a formulation selected by ILENG. If NTRANS $= 2$, the ε-equation (3.1.128) is solved.

ITDIF

Determines whether the surface drag and exchange coefficients C_D^s, C_E and C_H are evaluated as function of the air-sea temperature difference ΔT.

0 : C_D^s is determined using one of the formulations selected by IDRAG. C_E and C_H have the uniform values given by (3.1.217).

1 : The coefficients are determined as function of ΔT and wind speed using the theory described in Section III-1.6.4.

Note that the value of ITDIF is only relevant to the program is IOPTM = 2 or 3.

ITFORM

Selects the type of the turbulence scheme if IOPTK = 1.

1 : Pacanowski-Philander formulations (3.1.86)–(3.1.88), (3.1.92)

2 : Munk-Anderson formulation (3.1.95)–(3.1.99)

3 : flow-dependent formulation(3.1.101)–(3.1.102) using (3.1.108) for the flow factor α

4 : flow-dependent formulation (3.1.101)–(3.1.102) using (3.1.109) for the flow factor α

5 : flow-dependent formulation(3.1.101)–(3.1.102) using (3.1.110) for the flow factor α

NTRANS

Selects the number of transport equations in the turbulence scheme if IOPTK = 2.

0 : The zero-equation model is selected with k determined by (3.1.136)–(3.1.137) if ISTPAR = 1 or by (3.1.140) if ISTPAR = 2 and l determined by a formulation selected by ILENG.

1 : The one-equation model is selected with k obtained from the k-equation (3.1.114) and l determined by a formulation selected by ILENG.

2 : The two-equation model is selected with k obtained from the k-equation (3.1.114), l from the kl-equation (3.1.125) if ITCPAR = 1 and ε from the ε-equation (3.1.128) if ITCPAR = 2.

2.2.2 Date/time parameters

IBDATE

Start date of the simulation which must be given in the I8.8 format MMDDHHMM (month-day-hour-minute).

IEDATE

End date of the simulation which must be given in the I8.8 format MMDDHHMM (month-day-hour-minute).

IBYEAR

Start year of the simulation.

IEYEAR

End year of the simulation.

Note that, if solar radiation is calculated by the program, time must be expressed in GMT.

2.2.3 Time steps

DELT

The program uses a series of time steps which are expressed (using a series of counters defined below) as multiples of the fundamental time step Δt_{2D} for the 2-D mode. The FORTRAN name of Δt_{2D} is DELT which must always be defined by the user (in seconds). The time step must then be chosen such that the criterion (3.4.6) is satisfied. This condition is checked by the program and execution stops with an error message if the criterion is not satisfied. More stringent stability conditions are discussed in Deleersnijder et al. (1997).

2.2.4 Counters

There are eigth counters available in the program, represented by an integer number. Each counter has a corresponding time step obtained by multiplying its value with the fundamental time step DELT.

IC3D

The corresponding time step DEL3 = IC3D*DELT is used for updating all 3-D vector and scalar quantities (currents, temperature, salinity, turbulence, biological quantities, sediments, contaminants and particle transport). A proper choice of IC3D is essential to minimise the execution time of the program. The 3-D time step is limited by the conditions (3.4.8) and, to a lesser extent, the criterion (3.4.7). Both conditions are less stringent than the the 2-D mode criterion (3.4.6). The CFL-condition (3.4.8) gives the following upper limit

$$\mathsf{IC3D} < \Delta h_{min}/(\Delta t_{2D} U_{max}) \tag{4.2.1}$$

where U_{max} is a characteristic maximum value for the horizontal current and Δh_{min} the smallest horizontal grid spacing. In practice, it is advised to select values for DELT and IC3D well below the limits imposed by (3.4.6) and (4.2.1). If for example a horizontal grid spacing is chosen of 1 km, $U_{max} \sim$ 1m/s and h_{max} = 20 m, a reasonable choice would be to take DELT between 15–30 s and IC3D between 20–30. Note that the criterion (4.2.1) is not checked by the program. The value of IC3D must obviously be strictly positive. For 1-D simulations one has to set IC3D equal to 1. By default, IC3D equals 1.

ICMET

This counter is used by the program to read input from the meteorological data file. Instructions how to write this file are given in Section IV-2.7. If for example DELT = 30 s and meteorological input is provided at hourly intervals, then ICMET = 120. ICMET must only be defined with a value greater than 0 if IOPTM is larger than 1.

ICWAV

This counter is used by the program to read input from the wave data file. Instructions how to write this file are given in Section IV-2.7. ICWAV must only be defined with a value greater than 0 if IOPTW is greater than 1.

IC2BC

The open boundary conditions for the 2-D mode require the specification of the harmonic function F_{har} having the form given by equation (3.1.198). When IC2BC equals 0 (default value), the harmonic parameters A_0, A_n and φ_n are taken as time-independent and only need to be supplied at the initial time. If IC2BC is non-zero, new values are read by the program for all open boundary locations at time intervals selected by IC2BC.

ICHBC

The open boundary conditions for the 3-D horizontal current, temperature and salinity (see Section III-1.6.3b) require the specification of external arrays at each open boundary point (see Section IV-2.5). If the counter ICHBC equals 0 (default value), these arrays are considered independent of time and are supplied at the initial time. If ICHBC is non-zero, new values are read by the program at time intervals selected by ICHBC.

ICBBC

The open boundary conditions for the biological and sediment concentrations require the specification for each quantity of vertical arrays at each open boundary point (see Section IV-2.5). If the counter ICBBC equals 0 (default value), these arrays are taken as time-independent and are given at the initial time. If ICBBC is non-zero, new values are supplied by the program at time intervals selected by ICBBC.

ICCBC

The open boundary conditions for the contaminant distributions require the specification for each contaminant of vertical arrays at each open boundary point (see Section IV-2.5). If the counter ICCBC equals 0 (default value), these arrays are taken as independent of time and must only be given at the initial time. When ICCBC has a non-zero value, new values are supplied by the program at time intervals selected by ICCBC.

ICPBC

The open boundary conditions for the (Lagrangian) suspended particle module require the specification of an ambient concentration at open sea and/or a river discharge at river boundaries (see Section IV-2.5.3). When the counter ICPBC takes a zero value (default), the open boundary input is taken as time-independent and only needs to be supplied at the initial time. If ICPBC is non-zero, the program reads new input values at time intervals selected by ICPBC.

2.2.5 File parameters

HEADERS

The LOGICAL parameter HEADERS selects whether "header" information will be included in the files written in ASCII-format ('A'-files). By default, HEADERS is .FALSE..

.TRUE. : Arrays are written to ASCII-formatted output files with extra header lines which make it easier to trace the array indices of each output data value. This option may consume a considerable amoumt of disk space in the case of large arrays and should only be used to check the contents of the data file.

.FALSE. : Extra header information is disabled. Note that ASCII-formatted output files are written with a header line containing the FORTRAN name, description and unit of each output array.

GRDFORM

The preprocessor writes a file with grid parameters and arrays (see Section IV-2.3). The format of this file is selected by the parameter GRDFORM. Default is ASCII-format.

'A' : ASCII-format

'U' : unformatted (binary)

ICRFORM

The preprocessor creates a series of files with initial values for 2-D and 3-D arrays (Section IV-4.1.1), written in ASCII-format by default. These files are used to initialise the main program. After execution of the main program a series of output files is created of the same form which can be used to restart the program. The format of these input and output files is selected by the parameter ICRFORM.

'A' : ASCII-format

'U' : unformatted (binary)

BCSFORM

The preprocessor creates a series of files with the open boundary data (Section IV-4.1.1), written in ASCII-format by default. The format of these files is selected by the parameter BCSFORM.

'A' : ASCII-format

'U' : unformatted (binary)

METFORM

By default, input/output of meteorological data is performed using the ASCII-format. To increase the speed of the I/O operations and to reduce the size of large data files, especially when IOPTM = 3 or 5, this format can be changed using the parameter METFORM which may take the following values:

'A' : ASCII-format

'U' : unformatted (binary)

WAVFORM

By default, input/output of wave data is performed using the ASCII-format. To increase the speed of I/O operations and to reduce the size of large data files, especially if IOPTW = 3, this format can be changed using the parameter WAVFORM which may take the following values:

'A' : ASCII-format

'U' : unformatted (binary)

MAXFILES

The maximum number of file connections which can be open at the same time during program execution. Its value is machine-dependent. Default value is 256.

OUTTIT

By default the input and output files have the same prefix given by the parameter TITLE (see Section IV-3.1). The prefix of the output files (time series, harmonic, time-averaged and particle output) can be changed by defining the parameter OUTTIT which must be composed of 6 characters (without blanks).

2.2.6 Reference values

DLAREF

Reference latitude in decimal degrees. In the case of a Cartesian grid this parameter is needed to evaluate the spatially uniform Coriolis frequency and is used in the formulation for the surface flux of solar radiation (see Section III-1.6.5). Its value must always be defined if IGTRH = 0. No value needs to be supplied if IGTRH = 1. Limiting values are -90^0 and $+90^0$.

DLOREF

Reference longitude in decimal degrees having a positive (negative) value in the eastern (western) hemisphere. This parameter is only used in the formulation for the solar radiation flux (see Section III-1.6.5). Its value must always be defined if IGTRH = 0 and IOPTM equals 2 or 3. In all other cases no value needs to be supplied. Limiting values are -180^0 and $+180^0$.

DEPUN

Uniform mean water depth (m) which must be defined by the user if IGRDIM = 1 (1-D simulation). If IGRDIM = 3 (default value), the mean water depths are given by the array DEP defined in subroutine DEFGRID (see Section IV-2.3 below).

R0REF

The reference density ρ_0 (kg/m^3) used to normalise the surface fluxes and bottom stress, in the linear equation of state (3.1.162), expression (3.1.16) for the buoyancy b and the formulation (3.1.168) for the surface stress. Its precise value is (relatively) unimportant.

SREF

The reference salinity S_0 (PSU) used to initialise the salinity field by default and in the linear equation of state (3.1.162). Note that, if the salinity equation is not solved (IOPTSA $= 0$) and initialised by default, a precise value of SREF may be important for the evaluation of the diffuse attenuation coefficient k_2 (see equation (3.1.159)).

TREF

The reference temperature T_0 (^{0}C) used to initialise the temperature field by default and in the linear equation of state (3.1.162). Note that TREF also appears in the Monin-Obukhov formulation for the surface fluxes (ITDIF $= 1$) as discussed in Section III-1.6.4.

2.2.7 Expansion coefficients

SBETUN

The uniform expansion coefficient for salinity β_S (PSU^{-1}) used when IOPTD $= 1$. If IOPTD $= 2$, this parameter is calculated by the program as function of S and T using (3.1.165) and (3.1.166), whereas β_S is set to zero if IOPTD $= 0$.

TBETUN

The uniform expansion coefficient for temperature β_T used when IOPTD $= 1$. If IOPTD $= 2$, this parameter is calculated by the program as function of S and T using (3.1.164) and (3.1.166), whereas β_T is set to zero if IOPTD $= 0$.

2.2.8 Optical parameters

ATCF1UN

The diffuse attenuation coefficient k_1 for the infrared part of solar irradiance determined by equation (3.1.156).

ATCF2UN

The fresh water value k_{20}^w of the diffuse attenuation coefficient used in the expressions (3.1.159) for the physical coefficient k_2 and (3.2.6) for the biological coefficient k_d.

R1OPTUN

The uniform fraction R_1 of infrared radiation in the expression (3.1.155) for solar irradiance.

R2OPTUN

The uniform extra attenuation factor R_2 which measures the hyper-exponential fraction of PAR decay in (3.1.160).

EPSSAL

The slope parameter ϵ^S in equation (3.1.159) relating the attenuation coefficient k_2 to salinity.

HEXPUN

The thickness Δ_{opt} of the optical layer for hyper-exponential PAR decay (see equation(3.1.160)).

Note that the optical parameters only need to defined if IOPTHE $= 2$ or IOPTB $= 1$.

2.2.9 Background mixing coefficients

VISMOL

The uniform background eddy viscosity coefficient ν_b appearing in (3.1.95), (3.1.101), (3.1.115) and (3.1.117). Note that if IOPTK $= 0$, $\nu_T = \nu_b$.

DIFMOL

The uniform background eddy diffusion coefficient λ_b appearing in (3.1.96), (3.1.102), (3.1.115) and (3.1.117). Note that if IOPTK $= 0$, $\lambda_T = \lambda_b$.

2.2.10 Horizontal diffusion parameters

HEDVUN

Represents the uniform horizontal diffusion coefficient ν_H (m^2/2) for momentum if IODIF $= 1$.

HEDDUN

Represents either the uniform horizontal diffusion coefficient λ_H (m^2/s) for all scalars if IODIF $= 1$.

CM0

The parameter C_{m0} in the shear-dependent formulation (3.1.151) or (3.1.153) for the horizontal diffusion coefficient ν_H for momentum. This parameter is only used when IODIF $= 2$.

CS0

The parameter C_{s0} in the shear-dependent formulation (3.1.151) or (3.1.153) for the horizontal diffusion coefficient λ_H for scalars. This parameter is only used when IODIF $= 2$.

2.2.11 Bottom friction coefficient

CDLIN

The uniform bottom friction coefficient k_{lin} (m/s) used in the bottom stress formulation (3.1.186) if IBSTR $= 2$.

CDZ0UN

Spatially uniform value for the bottom roughness length z_0 (m) as used in the expression (3.1.187) for the quadratic bottom friction coefficient C_D^b. The parameter z_0 can alternatively be defined in the form of a spatially varying array in subroutine DEFGRID. Note that CDZ0UN needs only to be defined if IBSTR $= 1$ and if either the array CDZ0 is not defined by the user in subroutine DEFGRID or the model is run for a 1-D application (IGRDIM $= 1$).

2.2.12 Meteorological and wave forcing parameters

FSUN

The uniform X-component of the surface stress (m^2/s^2) normalised with the density ρ_0, used when IOPTM $= 1$. If IOPTM $= 0$, FSUN is set to zero.

GSUN

The uniform Y-component of the surface stress (m^2/s^2) normalised with the density ρ_0, used when IOPTM $= 1$. If IOPTM $= 0$, GSUN is set to zero.

HSUN

The uniform significant wave height h_s (m) used in the wave-current interaction module if IOPTW $= 1$.

TWUN

The uniform wave period T_w (s) used in the wave-current interaction module if IOPTW $= 1$.

2.2.13 Turbulence model parameters

The turbulence schemes implemented in the program involve a large series of parameters. They are all set by default. Some default values can be changed by the user in subroutine DEFCON, which are all listed in Table 4.2.3. A full discussion of all turbulence schemes implemented in the program is given in Section III-1.2. For a full list of all turbulence model parameters see Table 5.2.7.

2.2.14 Model parameters for the biology

An extensive set of parameters is used in the biological model. The default values, listed in Table 4.2.3, can be changed by the user in subroutine DEFCON. Their meaning is explained in Chapter III-2 and the tables therein. For a full list of all biological model parameters see Table 5.2.9.

2.2.15 Model parameters for the sediment module

The sediment module involves two parameters for the determination of the resuspension rate and the uniform vertical sinking velocity.

ALPHAS
: The parameter α_s (g m^{-2} s^{-1}) used to determine the resuspension rate in the sediment and biological modules (see equation (3.2.72)).

RNSED
: The exponential factor n_s used to determine the resuspension rate in the sediment and biological modules (see equation (3.2.72)).

WSED1
: The uniform sinking velocity w_s^A (m/s) for inorganic sediment. (This parameter normally depends on the particle size.)

2.2.16 Model parameters for the Lagrangian particle module

STLSOL
: Mass m_{riv} (ton) of the particles which enter the domain through a river open boundary.

STSSOL
: Mass m_{sea} (ton) of the particles which enter the domain through an open sea boundary.

HEDPUN
: Uniform horizontal diffusion coefficient λ_H^C (m^2/s) for particle transport. Note that horizontal diffusion is only enabled in the particle module if IOPTP $= 2$.

The particle masses in the Lagrangian module are expressed by default in tons. A different unit can be selected in which case the parameters STLSOL, STSSOL and the array ST with the particle masses, defined in Section IV-2.6.1, must have the same units.

2.3 Subroutine DEFGRID

In this routine the user specifies the area of the simulation. A series of grid parameters and arrays must be defined here. They are listed in Table 4.2.4 with their FORTRAN name, data type, array dimensions, unit and a short description of their purpose. Unless stated otherwise all default values are zero. In the case of a 1-D application (IGRDIM $= 1$), the program sets most grid arrays by default in routine DEFGRID1 so that there is usually no need for defining subroutine DEFGRID. For more details see Section IV-2.9 and the description of DEFGRID1 in Section V-1.9.

GX0(NC+1)
: An array with the x_1-coordinates of the cell corners as shown in Figure 3.4.1. The coordinates are either expressed in meters if IGTRH $= 0$ or in decimal degrees longitude if IGTRH $= 1$. The array must be given in ascending order so that GX0(1) denotes the x_1-coordinate of the lower and upper left corners and GX0(NC+1) the x_1-coordinate of the lower and upper right corners of the domain. Note that, if IGTRH $= 1$, the values are constrained between -180^0 and $+180^0$.

GY0(NR+1)
: An array with the x_2-coordinates of the cell corners as shown in Figure 3.4.1. The coordinates are either expressed in meters if IGTRH $= 0$ or in decimal degrees latitude if IGTRH $= 1$. The array must be given in ascending order so that GY0(1) denotes the x_2-coordinate of the lower left and lower right corners and GY0(NR+1) the x_2-coordinate of the upper left and upper right corners of the domain. Note that, if IGTRH $= 1$, the values are constrained between -90^0 and $+90^0$.

GZ0(NZ+1)
: An array with the values of the σ-coordinate at the vertical W-nodes (see Figure 3.4.1) which must be given in ascending order. Although this is not explicitly required by the program, the bottom and surface values GZ0(1) and GZ0(NZ+1) are usually set to 0 and 1 respectively. If GZ0 is not defined by the user, the program selects a uniform grid by default so that

```
GZ0(K) = (K-1)/REAL(NZ) for K=1,NZ+1
```

DEP(NR,NC)
: An horizontal array representing the mean water depths h (m) at the cell centres which must always be defined by the user (unless IGRDIM $= 1$). The array values are usually obtained from a bathymetric input file with a user-defined format. It is recommended to set DEP(J,I) $= 0$ for land areas. Since an alternative drying or wetting of grid cells is not implemented in the model, the program will inevitably crash due to a division by zero, when the total water depth $H = h + \zeta$ becomes zero on a wet cell. This may occur for negative surface elevations induced by tides or wind. In that case it is highly recommended to adopt a minimum water depth on all wet cells.

NWD(NR,NC)
: An horizontal integer array of cell pointers evaluated at the cell centres which must always be supplied by the user. The array informs the program about the location of the wet and dry areas inside the simulated domain. Allowed values are

0 : dry cell (land area)

1 : wet cell (sea area)

Note that the values of all program arrays are set to zero by the program on dry cells. To avoid division by a zero water depth land areas are automatically disregarded in the calculations. It is clear that DEP(J,I) must be strictly positive if NWD(J,I) = 1.

NPIX(NR,NC+1)

An horizontal integer array of cell face pointers evaluated at the U-nodes which must always be supplied by the user. The array informs the program about the locations of land, open sea and river boundaries at U-nodes. Allowed values are

0 : dry interface

1 : wet interior interface not located at an open boundary

2 : open sea boundary interface

3 : river boundary interface

The following constraints apply depending on the value of NPIX(J,I):

0 : The cell face is at the western or the eastern boundary of the computational domain or at least one adjacent cell is wet so that either I = 1, or I = NC+1, or NWD(J,I-1) = 0, or NWD(J,I) = 0.

1 : The cell face is at the intersection of two (interior) wet cells so that 1 < I < NC+1, NWD(J,I-1) = 1 and NWD(J,I) = 1.

2 : The cell face is either at the western or the eastern boundary of the computational domain and the adjacent interior cell is wet so that I = 1 with NWD(J,1) = 1, or I = NC+1 with NWD(J,NC) = 1.

3 : The cell face is either at the western or the eastern boundary of the computational domain and the adjacent interior cell is wet, or the cell face is at a land/sea boundary so that either I = 1 with NWD(J,1) = 1, or I = NC+1 with NWD(J,NC) = 1, or NWD(J,I-1) = 0 and NWD(J,I) = 1, or NWD(J,I-1) = 1 and NWD(J,I) = 0.

NPIY(NR+1,NC)

An horizontal integer array of cell face pointers evaluated at the V-nodes which must always be supplied by the user. The array informs the program about the locations of land, open sea and river boundaries at V-nodes. Allowed values are

0 : dry interface

1 : wet interior interface not located at an open boundary

2 : open sea boundary interface

3 : river boundary interface

The following constraints apply depending on the value of NPIY(J,I):

0 : The cell face is at the southern or the northern boundary of the computational domain or at least one adjacent cell is wet so that either J = 1, or J = NR+1, or NWD(J-1,I) = 0, or NWD(J,I) = 0.

1 : The cell face is at the intersection of two (interior) wet cells so that 1 < J < NR+1, NWD(J-1,I) = 1 and NWD(J,I) = 1.

2 : The cell face is either at the southern or the northern boundary of the computational domain and the adjacent interior cell is wet so that J = 1 with NWD(1,I) = 1, or J = NR+1 with NWD(NR,I) = 1.

3 : The cell face is either at the southern or the northern boundary of the computional domain and the adjacent interior cell is wet, or the cell face is at a land/sea boundary so that either J = 1 with NWD(1,I) = 1, or J = NR+1 with NWD(NR,I) = 1, or NWD(J-1,I) = 0 and NWD(J,I) = 1, or NWD(J-1,I) = 1 and NWD(J,I) = 0.

IOBU(0:NOBU)
An array with the X-indices of the U-nodes located at an open sea or river boundary. At open sea boundaries only the values 1 and NC+1 are allowed. For river boundaries the values may vary between 1 and NC+1.

JOBU(0:NOBU)
An array with the Y-indices of the U-nodes located at an open sea or river boundary. Allowed values are between 1 and NR.

IOBV(0:NOBV)
An array with the X-indices of the V-nodes located at an open sea or river boundary. Allowed values are between 1 and NC.

JOBV(0:NOBV)
An array with the Y-indices of the V-nodes located at an open sea or river boundary. At open sea boundaries only the values 1 and NR+1 are allowed. For river boundaries the values may vary between 1 and NR+1.

CDZ0(NR,NC)
A horizontal array with the bottom roughness lengths z_0 evaluated at the cell centres which must be supplied by the user if IBSTR = 1 (quadratic friction law) and the parameter CDZ0UN is not defined in DEFCON. Unless a value different from 1 is selected for IBSTR, the array must be strictly positive on wet cells. If a horizontally uniform value is taken for z_0, the user must define the parameter CDZ0UN in DEFCON and the array CDZ0 does not need to be supplied by the user.

For more details about the model grid and the location of the variables see Section III-4.2.

2.4 Subroutine DEFOB2D

As explained in Sections III-1.6.3a and III-4.3.11b a number of arrays must be defined by the user to inform the program which type of open boundary condition is used for the 2-D mode at all open boundary locations and to provide the necessary data values. A complete list is given in Table IV-4.2.5. Note that some of these arrays have a different meaning in the case of a 1-D application. This is further discussed in Section IV-2.9.

2.4.1 Type of conditions

The open boundary conditions for the 2-D mode are based upon the method of Riemann characteristics. The depth-integrated current is determined as the average of the outgoing and incoming Riemann variable. The former is computed by solving the discretised equation (3.4.163) at the western/eastern and (3.4.164) at the southern/northern boundaries, whereas the latter is determined using an harmonic expansion of the form (3.4.167) at western/eastern and (3.4.168) at southern/northern boundaries. As explained in Section III-4.3.11b, the meaning of the harmonic function depends on the type of boundary condition selected by the arrays ITYPOBU and ITYPOBV.

ITYPOBU(0:NOBU)
: Selects the type of condition used at U-open boundaries. Allowed values are

0 : The incoming Riemann characteristic is set to zero which means that the input values for the depth-integrated current $\overline{U}_{in}$ and the surface elevation ζ_{in} are taken to be zero.

1 : The harmonic function is defined by the first equation of (3.4.170) and is written as a function of both $\overline{U}_{in}$ and ζ_{in}.

2 : A zero normal gradient condition is considered. No input data are required.

3 : The harmonic function is defined by the first equation of (3.4.172) and is written as a function of ζ_{in} only.

4 : The harmonic function is defined by the first equation of (3.4.174) and is written as a function of $\overline{U}_{in}$ only.

ITYPOBV(0:NOBV)
: Selects the type of condition used at V-open boundaries. Allowed values are

0 : The incoming Riemann characteristic is set to zero which means that the input values for the depth-integrated current $\overline{V}_{in}$ and the surface elevation ζ_{in} are taken to be zero.

1 : The harmonic function is defined by the second equation of (3.4.170) and is written as a function of both $\overline{V}_{in}$ and ζ_{in}.

2 : A zero normal gradient condition is considered. No input data are required.

3 : The harmonic function is defined by the second equation of (3.4.172) and is written as a function of ζ_{in} only.

4 : The harmonic function is defined by the second equation of (3.4.174) and is written as a function of $\overline{V}_{in}$ only.

2.4.2 Frequencies and initial phases

SIGMA(0:NCON)
: The tidal forcing frequencies ω_n used in the harmonic expansion (3.4.167) or (3.4.168).

PHAS0(0:NCON)
: The initial phases φ_{n0} appearing in the harmonic function (3.4.167) or (3.4.168). This array is updated by the program at each time step. Its values at the end of the program execution are then given by $\omega_n T_s + \varphi_{n0}$ (modulo 2π) where T_s is the total time of the simulation, and are stored into the '*hou*' output file (see Section IV-4.1) so that they can be reused for a possible restart of the program. This array may alternatively be defined in subroutine DEFICS.

ANALSIG(0:NCONTO)
: The frequencies ω_n (see equation (3.1.287)) used for harmonic analysis. This array must be defined if IOUTT equals 1 or 2.

2.4.3 Amplitudes and phases

The residual part and the amplitudes of the forcing are determined by the four arrays described below. They only need to be defined in subroutine DEFOB2D when the data are time-independent, i.e. IC2BC = 0. In that case the arrays are automatically written to the appropriate *2bc*-file by the preprocessor. Time-dependent data must be supplied and written to the *2bc*-file by the user in subroutine WROBDAT (see Section IV-2.8). Note that the arrays have a different meaning in the case of a 1-D application (IGRDIM = 1) (see Section IV-2.9 for further details).

AMPOBU(0:NOBU,0:NCON)
: The values of the residual part A_0 (given by the vector AMPOBU(1:NOBU,0)) and the amplitudes A_k (given by AMPOBU(1:NOBU,1:NCON)) in meters appearing in the harmonic function F^u_{har} defined by (3.4.167), for the open boundary points located at U-nodes.

AMPOBV(0:NOBV,0:NCON)
: The values of the residual part A_0 (given by the vector AMPOBV(1:NOBV,0)) and the amplitudes A_k (given by AMPOBV(1:NOBV,1:NCON)) in meters appearing in the harmonic function F^v_{har} defined by (3.4.168), for the open boundary points located at V-nodes.

PHAOBU(0:NOBU,0:NCON)
: The values of the phases φ_k in radians (given by PHAOBU(1:NOBU,1:NCON)) in the harmonic function F^u_{har} defined by (3.4.167), for the open boundary points located at U-nodes.

PHAOBV(0:NOBV,0:NCON)
: The values of the phases φ_k in radians (given by PHAOBV(1:NOBV,1:NCON)) in the harmonic function F^v_{har} defined by (3.4.168), for the open boundary points located at V-nodes.

2.5 Subroutine DEFOB3D

As explained in Section III-1.6.3b the open boundary conditions for a scalar quantity and the horizontal current allow to define either a scalar or a gradient condition at each vertical grid point located at an open boundary. This means in practice that the program has to be informed which type of condition will be used at all grid points in the vertical situated at open boundary locations. In the case of a scalar condition an external value ψ_e must be provided in addition. This is implemented in the program as follows. At each open boundary location a vertical array has to be defined for all the scalar quantities which are updated at each 3-D time step by solving a transport equation of the form (3.1.70), and for the velocity deviations u', v'. The elements of the array are denoted by ψ_0, ψ_1, $\ldots$, ψ_{N_z} where N_z denotes the number of grid points in the vertical represented by the parameter NZ and ψ either represents a scalar or one of the velocity deviations u' or v'. If $\psi_k \leq \psi_0$ a zero gradient condition is applied at level K whereas ψ_k denotes an external value of ψ if $\psi_k > \psi_0$. The default value of ψ_0 is "-99" which is lower than the actual value of all scalars used in the program or of any realistic value of u' and v'. (With exception of the temperature the scalars represent concentrations which must remain non-negative so that this default value should not be changed by the user). The procedure must in principle be repeated for each open boundary point. The storage of all these data into a series of data files may consume a large amount of disk space. The number of data values can be reduced in the program by limiting the number of "distinct" vertical arrays since in many applications the same array can be used at different open boundary points. The maximum number of "distinct" arrays is represented in the program by the parameter NVPROF defined in the file *param.inc* (see Section IV-2.1). If the user prefers to apply different conditions at all open boundary points, the parameter NVPROF must then be set to the total number of open boundary points given by NOBU+NOBV. If the same array is used at all boundaries, one must set NVPROF $= 1$.

2.5.1 Array numbers

To inform the program which arrays are applied to which open boundary point the following two INTEGER arrays must be supplied:

IVPOBU(0:NOBU)
: An array representing the number of the profile to be used for open boundaries located at U-nodes. Values are between 1 and NVPROF. The array elements must obviously be ordered in the same way as the boundary grid index arrays IOBU and JOBU, i.e. the profile number IVPOBU(II) applies at the boundary point with indices IOBU(II), JOBU(II). If NVPROF $= 0$, the program assumes that all boundaries are closed and the array IVPOBU must not be defined.

IVPOBV(0:NOBV)
: An array representing the number of the profile to be used for open boundaries located at V-nodes. Values are between 1 and NVPROF. The array elements must obviously be ordered in the same way as the boundary grid index arrays IOBV and JOBV, i.e. the profile number IVPOBV(JJ) applies at the boundary point with indices IOBV(JJ), JOBV(JJ). If NVPROF $= 0$, the program assumes that all boundaries are closed and the array IVPOBV must not be defined.

2.5.2 Open boundary data

The array elements ψ_0, ψ_1, ..., ψ_{N_z} are stored into a "PSIVP"-array where "PSI" stands for e.g. salinity "S", temperature "T", sediment concentration "SEDC1", microplankton carbon biomass "P2B", ... With exception of the contaminant distributions these are 2-dimensional arrays of the form PSIVP(0:NZ,0:NVPROF). The arrays are applied in the program as follows. Let M, N be array number indices between 1 and NVPROF, II the index of a U-open boundary point (between 1 and NOBU) and JJ the index of a V-open boundary point (between 1 and NOBV) such that IVPOBU(II) = M and IVPOBV(JJ) = N.

- If PSIVP(K,M) is greater than PSIVP(0,M), the advective and diffusive fluxes at the U-open boundary location are evaluated by taking PSIVP(K,M) as an external value. Similarly, if PSIVP(K,N) is greater than PSIVP(0,N), the advective and diffusive fluxes at the V-open boundary location are evaluated by taking PSIVP(K,N) as an external value.

- If PSIVP(K,M) is lower than or equal to PSIVP(0,M), the advective and diffusive fluxes at the U-open boundary are evaluated with a zero gradient condition which means that the exterior value is set to the nearest interior value of ψ (calculated by the program). Similarly, a zero gradient condition is applied at the V-open boundary if PSIVP(K,N) is lower than or equal to PSIVP(0,N).

Assume for example that the salinity "S" is prescribed at a river open boundary by a two-layer stratification with a fixed fresh water value of 10 PSU in the surface layer and a zero gradient condition in the bottom layer. The surface layer depth is set to 10 m. Assume further that the river mouth is located at a western boundary with index II and the open boundary number is given by M so that IVPOBU(II) = M. The corresponding open boundary array is defined, e.g., as follows

```
      I = IOBU(II)
      J = JOBU(II)
      SVP(0,M) = -99.0
      DO 101 K=1,NZ
         ZSUR = (1.0-0.5*(GZ0(K)+GZ0(K+1))*DEP(J,I)
         IF (ZSUR.LT.10.0) THEN
            SVP(K,M) = 10.0
         ELSE
            SVP(K,M) = -100.0
         ENDIF
 101  CONTINUE
```

(Note that the water depth is evaluated at the centre of the nearest wet cell. This means that DEP(J,I) must be replaced by DEP(J,I-1) at an eastern boundary).

The procedure is slightly different in the contaminant module where the open boundary arrays are stored into a 3-dimensional array of the form CONVP(0:NZ,0:NVPROF,0:NCONC) where NCONC denotes the total number of contaminants defined in the file *param.inc* (see Section IV-2.1). The method is the same as previously. The third dimension only serves to distinguish between the different contaminant distributions..

In analogy with the "PSIVP"-arrays for 3-D scalar quantities, the open boundary conditions for the horizontal current (see Section III-1.6.3b) require the specification of an array of the form UVP(0:NZ,0:NVPROF). The array is applied in the following way. Let II and JJ the indices of two open boundary points situated respectively at a U- and a V-node. If the profile numbers IVPOBU(II) and IVPOBV(JJ) are denoted by M and N, then

- If UVP(K,M) is greater than UVP(0,M), the value of the velocity deviation u'_0 (see equation (3.1.210)) at level K is given by UVP(K,M). Similarly, if UVP(K,N) is greater than UVP(0,N), the value of the velocity deviation v'_0 at level K equals UVP(K,N).
- If UVP(K,M) is lower than or equal to UVP(0,M), the normal gradient of Ju' at level K is set to zero as given by equation (3.1.208). Similarly, the normal gradient of Jv' or $Jv'\cos\phi$ is set to zero as given by equation (3.1.209), if UVP(K,N) is lower than or equal to UVP(0,N).

Compared to the open boundary conditions for scalars the following two important restrictions apply:

- No mixed scalar and zero gradient conditions can be applied for the currrents in the vertical. This means that all the elements of the array (UVP(K,M),K=1,NZ) must either be lower than (or equal to) UVP(0,M) for a zero gradient or larger than UVP(0,M) for a scalar condition.
- By their definition the depth integrals of u'_0 and v'_0 must be zero. Using the notations of Chapter III-4 this gives

$$\sum_{k=1}^{N_z} J^u_k u'_{0;k} = 0 \quad \text{or} \quad \sum_{k=1}^{N_z} J^v_k v'_{0;k} = 0 \tag{4.2.2}$$

This means that the sum of the vertical array elements of (UVP(K,M),K=1,NZ) must be zero in the case of a scalar condition.

A full list of all types of arrays which can de defined in the program as input at the open boundaries, is given in Table 4.2.6. They can be divided into four categories: physical arrays, arrays for biology and sediments, contaminant arrays and arrays for Lagrangian transport. Note that they only need to be defined in subroutine DEFOB3D if the corresponding counter is zero in which case the data are automatically written by the program to the appropriate data files. This means that the physical input is defined here if ICHBC = 0, the biological and sediment input if ICBBC = 0 and the contaminant input if ICCBC = 0. Time-dependent open boundary input must be supplied and written to the appropriate data files by the user in subroutine WROBDAT (see Section IV-2.8) at time intervals selected by the appropriate counter.

By default, a zero gradient condition is defined for all scalars and for u', v'. This takes the form:

```
      DO 100 IVP=1,NVPROF
         PSIVP(0,IVP) = -99.0
         DO 101 K=1,NZ
            PSIVP(K,IVP) = -100.0
 101     CONTINUE
 100  CONTINUE
```

where "PSIVP" now stands for any of the arrays listed in Table 4.2.6 whose FORTRAN name ends with "VP". The user then only needs to change the array values corresponding to open boundary locations where a scalar condition will be applied.

2.5.3 Open boundary conditions for the Lagrangian module

Each particle has an attached label which does not change during the execution of the program and which allows to follow its position. The suspended material entering the domain through an open boundary is distributed into a new series of particles. A new label must be attributed for this newly formed particles. The program allows to define a label for each open boundary point so that the particle's origin can be traced. The information is stored into the two following user-defined INTEGER arrays.

LSTOBU(0:NOBU)
: An array with the labels of the particles entering the computational domain during inflow at open boundary points located at U-nodes.

LSTOBV(0:NOBV)
: An array with the labels of the particles entering the computational domain during inflow at open boundary points located at V-nodes.

The open boundary input is defined with the aid of the following arrays:

QSTOBU(0:NOBU)
: An array which determines the input along open boundaries at U-nodes. At open sea boundaries array values represent the ambient concentration of suspended material in ton/m^3 just outside the grid. At river boundaries the array values represent the discharge of suspended material in ton/s.

QSTOBV(0:NOBV)
: An array which determines the input along open boundaries at V-nodes. At open sea boundaries array values represent the ambient concentration of suspended material in ton/m^3 just outside the grid. At river boundaries the array values represent the discharge of suspended material in ton/s.

If the arrays QSTOBU and QSTOBV are not defined, the open boundary input is set to zero by default. Note that the arrays QSTOBU and QSTOBV only need to be defined in subroutine DEFOB3D if the counter ICPBC $= 0$ in which case the arrays are automatically written to the appropriate data file. Time-dependent input must be supplied and written to the appropriate data file by the user in subroutine WROBDAT (see Section IV-2.8) at time intervals selected by ICPBC.

The default unit of mass for QSTOBU, QSTOBV is ton. As stated in Section IV-2.2.16 above, a different unit of mass may be selected in the Lagrangian module. Care should be taken that the same unit is used throughout (parameters STLSOL, STSSOL and the array ST).

2.6 Subroutine DEFICS

Initial conditions are defined in this subroutine. The preprocessor stores a series of arrays into a number of input files which are read by the main program at the startup time ($t = 0$). These arrays are continuously updated during program execution. All arrays are listed in Table 4.2.7. At the end of the program ($t = T_s$) the updated values are stored again into a series of output files having the same structure and format as the corresponding input files. The output files can then be used to restart the program at $t = T_s$. Restarting the program is further discussed in Section IV-5.4. The arrays can be divided into four categories:

- arrays which must always be initialised by the user if the appropriate switch is non-zero
- arrays which may be initialised if the user prefers to change the default values
- arrays for which changing the default values is not recommended
- arrays whose default values cannot be changed by the user

2.6.1 Arrays which must always be initialised

- The biological concentrations P2B, P2C, P2M, P2N, P2NHS, P2NOS, P2O must always be initialised if IOPTB $= 1$.
- The inorganic suspended matter concentration SEDC1 must always be initialised if IOPTS $= 1$.
- The contaminant distributions CONCN must always be initialised if IOPTC $= 1, 2$.

A series of particle arrays must always be initialised if IOPTP is non-zero. The initial positions are determined by six arrays. The first three are INTEGER arrays with the cell indices of the particle positions. The next three are the particle coordinates with respect to the cell centre.

IT(0:MAXNOP)
: The X-indices of the centres of the cells where the particles are located. Allowed values are between 1 and NC.

JT(0:MAXNOP)
: The Y-indices of the centres of the cells where the particles are located. Allowed values are between 1 and NR.

KT(0:MAXNOP)
: The Z-indices of the centres of the cells where the particles are located. Allowed values are between 1 and NZ.

XT(0:MAXNOP)
: The x_1-coordinates of the particles either in m or in decimal degrees longitude with respect to the cell centre. Allowed values are between $-0.5\Delta x_{1;ji}$ and $+0.5\Delta x_{1;ji}$ where $\Delta x_{1;ji}$ is the grid spacing of the cell in the x_1-direction.

YT(0:MAXNOP)
: The x_2-coordinates of the particles either in m or in decimal degrees latitude with respect to the cell centre. Allowed values are between $-0.5\Delta x_{2;j}$ and $+0.5\Delta x_{2;j}$ where $\Delta x_{2;j}$ is the grid spacing of the cell in the x_2-direction.

ZT(0:MAXNOP)
: The vertical σ-coordinates of the particles with respect to the cell centre. Allowed values are between $-0.5\Delta\sigma_k$ and $+0.5\sigma_k$ where σ_k is the vertical grid spacing of the cell in σ-coordinates.

Using the notations of Section III-4.8.1 the arrays IT, JT, KT correspond to the particle coordinates i_n, j_n, k_n and the arrays XT, YT, ZT to the particle's relative coordinates x'_{1n}, x'_{2n}, σ'_n.

Two additional particle attributes must be defined:

LST(0:MAXNOP)
: An INTEGER array representing the labels of the particles. Values should be non-zero. If a particle exits at an open or solid horizontal boundary, its label is set to zero and its position is no longer updated.

ST(0:MAXNOP)
: An array representing the masses of the particles in ton. Default unit is ton. As stated in Section IV-2.2.16 a different unit may be selected provided that the same unit is taken for all parameters (STLSOL, STSSOL) in the Lagrangian module.

2.6.2 Arrays which may be initialised

S(NZ,NR,NC)
: Represents the salinity field, initialised by default to a spatially uniform value given by the reference salinity SREF.

T(NZ,NR,NC)
: Represents the temperature field, initialised by default to a spatially uniform value given by the reference temperature TREF.

PHAS0(0:NCON)
: The initial phases φ_{n0} appearing in the harmonic function F_{har} defined by (3.4.167) or (3.4.168). This array may alternatively be defined in subroutine DEFOB2D.

GRAZING(NR,NC,12)
: Monthly averaged values of the mesozooplankton grazing pressure in day^{-1}. If this array is not defined, the grazing pressure is taken as uniform in space and time and given by the value of the parameter GRAZUN.

FSED(NR,NC,5)
: The amount per m^2 of organic (microplankton and detritus) and inorganic material available in the fluff layer for resuspension. The units are the concentrations integrated over the bottom grid cell. Even when the array elements are initialised to zero (default), their values are updated during the execution of the program and stored at the end for an eventual restart. The type of material is given by the third array dimension:

1 : inorganic sediment (iSPM)

2 : microplankton carbon

3 : detrital carbon

4 : microplankton nitrogen

5 : detrital nitrogen

2.6.3 Arrays which should not be initialised

- The current arrays, the bottom stress, the sea surface elevation and the Riemann variables $R_\pm$ for the 2-D open boundary conditions are set to zero by default. If e.g. tidal forcing is applied at the open boundaries, it is recommended to run the program first for a number of tidal cycles using simplified forcing conditions (e.g. no wind forcing) which should give more realistic values at the end of the spin-up phase. This procedure is followed in the setup of the test cases **river** and **plume** and in the NOMADS and North Sea applications discussed in Chapters II-6 and II-7. Examples of applications where the currents can be initialised are the test cases **front** and **cones**.

- The default vertical grid spacing is obviously given by the mean water depth h times the vertical grid spacing $\Delta\sigma_k$ in σ-coordinates.

- If IOPTK $= 2$, the turbulence arrays are initialised by default using the method described in Section III-1.6.7 and some of the algebraic relations described in Section III-1.2.2. No initialisation is required if IOPTK $= 1$.

2.6.4 Arrays which cannot be initialised

These arrays, related to the microplankton and Lagrangian transport module, must be zero initially. However, non-zero values are automatically supplied at the end of the simulation for a restart of the program.

P2ZN(NZ,NR,NC)
: This array represents zooplankton nitrogen which acts as an accumulator of grazed microplankton and never decreases. Its values are stored at the end of the program for an eventual restart.

FLUFLOS(NR,NC,5)
: This array represents the amount of organic material per m^2 (microplankton and detrital carbon and nitrogen) lost from the fluff layer to the consolidated sediment since the initial time $t = 0$. The array values are saved at the end of the simulation for an eventual restart. The type of material is given by the third array dimension.

2 :	microplankton carbon
3 :	detrital carbon
4 :	microplankton nitrogen
5 :	detrital nitrogen

PAR(NZ,NR,NC)
: This array represents photosynthetically active radiation which is saved at the end of the program since PAR is calculated as an average over the past 24 hours.

STIN(NZ,NR,NC)
: If a particle enters the computational domain via a cell with index (K,J,I), its mass is added to the value of STIN(K,J,I).

STOUT(NZ,NR,NC)
: If a particle leaves the computational domain or reaches a land boundary via a cell with index (K,J,I), its mass is added to the value of STOUT(K,J,I).

During an inflow condition a new series of particles is created. Their number is obtained by dividing the amount of suspended material entering the domain during one time step by the particle's mass given by the parameter STSSOL (open sea boundaries) or STLSOL (river boundaries). The remaining mass fraction is stored by the program and will be added to the amount of matter entering during the next time step. The following arrays are used

RMASSU(0:NOBU)
: Remaining mass fraction M_{rest} at U-open boundaries.

RMASSV(0:NOBV)
: Remaining mass fraction M_{rest} at V-open boundaries.

The unit of the arrays STIN, STOUT, RMASSU, RMASSV is the same as the one selected by the user for the parameters STLSOL, STSSOL and the array LST. Default is ton.

2.7 Subroutine WRFCDAT

This routine is called by the preprocessor only if either IOPTM or IOPTW is larger than 1. In the former case meteorological data must be supplied by the user for each meteorological time step defined by the counter ICMET. In the same way wave data must be given for each wave time step defined by the counter ICWAV when IOPTW is larger than 1. The data must be written using one of the forms given in the generic *defmod.f* in the **/examples** subdirectory. If IOPTM $= 3, 5$ or IOPTW $= 3$, the data are in the form of horizontal arrays. The output is automatically written in the appropriate format by calling the WRARR subroutine, described in Section V-1.10.1 of the Reference Manual, with the appropriate arguments. The data files are already connected to the program via the unit numbers IOMET and IOWAV given by the last argument of WRARR. The user is not allowed to open or close these files. The input data are usually obtained from available input files(s) with a user-defined format. Note that either the array EVAPR representing the evaporation minus precipitation rate or the array RAIN2 representing the precipitation rate must be supplied depending on the value of IOPTSA. If IOPTM $= 2, 4$ or IOPTW $= 2$, only one value needs to be given for each forcing parameter at each time step. The FORTRAN name of these uniform parameters in the example is obtained by adding "UN" to the corresponding array name. All arrays are listed in Table 4.2.8. Note that each data parameter must be measured in the appropriate unit.

The following scalars or arrays must be supplied for the meteorological forcing in WRFCDAT at time intervals given by ICMET*DELT depending on the value of the switch IOPTM:

0 : no data need to be supplied

1 : no data need to be supplied

2 : one value for the wind components (WINDU2UN, WINDV2UN), air temperature (SAT2UN), relative humidity (HUM2UN), cloud coverage (CLOUD2UN) and either the evaporation minus precipitation rate (EVAPRUN) or the precipitation rate (RAIN2UN)

3 : array values for the wind components (WINDU2, WINDV2), atmospheric pressure (P2), air temperature (SAT2), relative humidity (HUM2), cloud coverage (CLOUD2) and either the evaporation minus precipitation rate (EVAPR) or the precipitation rate (RAIN2).

4 : one value for the wind components (WINDU2UN, WINDV2UN), the non-solar heat flux (QNSOLUN), the surface solar heat flux (QSOLUN) and the evaporation minus precipiation rate (EVAPRUN)

5 : array values for the wind components (WINDU2, WINDV2), the atmospheric pressure (P2), the non-solar heat flux (QNSOL), the surface solar heat flux (QSOL) and the evaporation minus precipitation rate (EVAPR)

The following surface wave data (scalars or arrays) must be supplied in WRFCDAT at time intervals given by ICWAV*DELT depending on the value of the switch IOPTW:

0 : no data need to be supplied

1 : no data need to be supplied

2 : one value for the wave heigth (HSUN) and the wave period (TWUN)

3 : array values for the wave height (HS) and the wave period (TW)

2.8 Subroutine WROBDAT

This routine is called by the preprocessor if at least one of the counters IC2BC, ICHBC, ICBBC, ICPBC or ICCBC is non-zero. When a counter is non-zero, a series of open boundary input data must be supplied by the user for each corresponding time step. A recommended form is given in the generic *defmod.f* file in **/examples**. The output is automatically written in the appropriate format by calling the WRARR subroutine, described in Section V-1.10.1 of the Reference Manual, with the appropriate arguments. The data files are already connected to the program via the unit numbers IO2BC, IOHBC, IOBBC, IOPBC and IOCBC. The user is not allowed to open or close these files. The input data are usually obtained from available input file(s) with a user-defined format. The following data need to be supplied for each appropriate time step:

- The arrays AMPOBU, PHAOBU, AMPOBV, PHAOBV when IC2BC > 0.
- The physical input arrays UVP, SVP, TVP when ICHBC > 0.
- The input arrays P2BVP, P2CVP, P2MVP, P2NVP, P2NHSVP, P2NOSVP for the biology and SEDC1VP for the sediment module when ICBBC > 0 and IOPTB or IOPTS equals 1.
- The open boundary arrays QSTOBU and QSTOBV when ICPBC > 0 and IOPTP > 0.
- The contaminant open boundary array CONVP if ICCBC > 0 and IOPTC > 0.

2.9 Initialising a 1-D application

If the program is initialised for a 1-D application, the switch IGRDIM has to be set to 1. The following restrictions and modifications apply:

- file *param.inc*:
  - The parameters NC and NR must be set to their minimum values, i.e. NC = NR = 2. Although this forces the program to run four times slower compared to a point model in the vertical, this restriction is implied by the choice of the C-grid. The program stops with an error message if these conditions are not satisfied.
  - All boundaries are open so that NOBU = NOBV = 4. The program stops with an error message if these conditions are not satisfied.
  - The parameter NVPROF can be set to zero since no open boundary arrays are required by the program.
- A number of switches are automatically set to 0, if IGRDIM = 1:
  - The grid must be Cartesian so that IGTRH = 0.
  - Advection of momentum and scalars must be disabled, i.e. IADVS = IADVC = 0. However, the evaluations of the vertical sinking term in the biological and sediment modules can be enabled using the switch IADVWB.
  - Horizontal diffusion must be disabled, i.e. IODIF = 0.
  - No mode-splitting is applied so that IOPT2 = 0.
  - The particle module must be disabled so that IOPTP = IOUTP = 0.
- The following counters are automatically set by the program if IGRDIM = 1:

  - IC2BC, ICHBC, ICBBC, ICPBC, ICCBC are set to 0 since no open boundary conditions are applied.
  - IC3D is set to 1 since the mode-splitting is disabled so that all calculations are performed using the same time step DELT.

- Model parameters:
  - The parameter DEPUN (mean water depth) must be defined in DEFCON. The program stops with an error message if this is not the case.
  - The parameter CDZ0UN (bottom roughness length) must be defined in DEFCON if IBSTR = 1.

- All grid arrays are automatically set by the program if IGRDIM = 1. Only if the user wants to implement an irregular grid spacing in the vertical, the array GZ0 can be defined in DEFGRID. In all other cases subroutine DEFGRID may be taken as empty.

- The arrays ITYPOBU and ITYPOBV are not used by the program and do not need to be defined.

- Important to note is that the arrays AMPOBU, AMPOBV, PHAOBU and PHAOBV now have a different purpose.

  AMPOBU(0,1:NCON)
  : The values of the amplitudes ζ_n in the expansion (3.1.83) for the sea surface elevation.

  AMPOBU(1,0:NCON)
  : The values of the residual part F_0 (given by AMPOBU(1,0)) and the amplitudes F_n (given by AMPOBU(1,1:NCON)) in the expansion (3.1.84) for the x_1-component of the pressure gradient.

  AMPOBV(1,0:NCON)
  : The values of the residual part G_0 (given by AMPOBV(1,0)) and the amplitudes G_n (given by AMPOBV(1,1:NCON)) in the expansion (3.1.85) for the x_2-component of the pressure gradient.

  PHAOBU(0,1:NCON)
  : The values of the phases φ_{ns} (radians) in the expansion (3.1.83) for the sea surface elevation.

  PHAOBU(1,1:NCON)
  : The values of the phases φ_{nx} (radians) in the expansion (3.1.84) for the x_1-component of the pressure gradient.

  PHAOBV(1,1:NCON)
  : The values of the phases φ_{ny} (radians) in the expansion (3.1.85) for the x_2-component of the pressure gradient.

All the other elements of the previous arrays have no real significance for the program and must therefore not be defined.

- The initial phases φ_n and the tidal frequencies ω_n are represented by the arrays PHASE0 and SIGMA as in the 3-D case.
- Subroutine DEFOB3D must be empty since no boundary conditions are applied.
- All arrays defined in DEFICS must be horizontally homogeneous.
- Subroutine WRFCDAT is written in the same way as in the 3-D case, except that the switch IOPTM should not take the value 3 or 5 and the switch IOPTW should be different from 3.
- Subroutine WROBDAT must be empty since no boundary conditions are applied.

Examples of 1-D applications are the test cases **pycno, csnsp, batch** and **csbio**.

2.10 Tables of model input parameters and arrays

Table 4.2.1: Parameters defined in *param.inc*.

name	default	purpose
NC	none	number of grid cells in the X-direction
NR	none	number of grid cells in the Y-direction
NZ	none	number of grid cells in the vertical direction
NOBU	0	number of open boundary points in the X-direction
NOBV	0	number of open boundary points in the Y-direction
NVPROF	0	maximum number of open boundary profiles
NCON	0	number of tidal frequencies
NCONTO	1	number of frequencies for harmonic analysis (between 1 and 9)
NCONC	0	number of contaminant distributions
MAXNOP	0	maximum allowed number of particles in the Lagrangian module
NOUTMAX	5	maximum number of 2-D or 3-D fields for time series output (minimum value of 1)
NANALMAX	1	maximum number of 2-D or 3-D fields for harmonic analysis (minimum value of 1)
NAVRMAX	5	maximum number of 2-D or 3-D fields for time-averaged output (minimum value of 1)
MAXOUT	1	maximum number of time series output files (minimum value of 1)
MAXAVR	1	maximum number of time-averaged output files (minimum value of 1)

Table 4.2.2: Program switches defined in DEFCON.

name	values	default	purpose
switches to activate/deactivate a particular module or to select a specific formulation			
IFLUFF	0, 1	1	fluff layer module for biology
IOPTB	0, 1	0	biological module
IOPTC	0, 1, 2	0	contaminant module
IOPTD	0, 1, 2	1	selects the equation of state
IOPTHE	0, 1, 2	0	heat equation module
IOPTK	0, 1, 2	2	turbulence module
IOPTM	0, 1, 2, 3, 4, 5	0	meteorological and surface fluxes modules
IOPTP	0, 1, 2	0	module for Lagrangian particle transport
IOPTS	0, 1	0	sediment module
IOPTSA	0, 1, 2	0	salinity equation module
IOPTW	0, 1, 2, 3	0	wave-current interaction module
IOPT2	0, 1	1	module for the 2-D mode equations
IOPT3	0, 1	1	3-D momentum module
grid			
IGRDIM	1, 3	3	switch to select a 1-D/3-D application
IGTRH	0, 1	0	switch to select a Cartesian/spherical grid
advection and horizontal diffusion			
IADVC	0, 1, 2, 3, 4	3	advection scheme for momentum
IADVS	0, 1, 2, 3, 4	3	advection scheme for scalar quantities
IADVWB	0, 1, 2, 3, 4	3	advection scheme for the vertical sinking rate term in the biological and sediment equations (only effective if IADVS = 0)
IODIF	0, 1, 2	0	scheme for horizontal diffusion
boundary conditions			
IBSTR	0, 1, 2	1	bottom stress formulation
IDRAG	0, 1, 2, 3, 4	0	formulation for the surface drag coefficient
ITDIF	0, 1	0	formulation for the exchange coefficients in the surface fluxes module

Table 4.2.2: continued.

name	values	default	purpose
turbulence model			
IAHDHT	0, 1	0	disables/enables advection and horizontal diffusion of turbulence variables
ILENG	1, 2, 3, 4	4	type of mixing length formulation
ILIM	0, 1	1	limiting conditions
ISTPAR	1, 2	1	form of the stability functions
ITCPAR	1, 2	2	type of the second turbulence variable
ITFORM	1, 2, 3, 4, 5	3	type of algebraic formulation
NTRANS	0, 1, 2	1	number of transport equations
output			
IOUTF	0, 1	0	time-averaged output
IOUTP	0, 1	0	particle output
IOUTS	0, 1	1	time series output
IOUTT	0, 1, 2	0	harmonic analysis

Table 4.2.3: Model constants defined in DEFCON.

name	type	default	purpose
Date/time parameters			
IBDATE	INTEGER	none	start date
IEDATE	INTEGER	none	end date
IBYEAR	INTEGER	1998	start year
IEYEAR	INTEGER	1998	end year
Time steps			
DELT	REAL	none	time step Δt_{2D} for the 2-D mode
Counters			
IC3D	INTEGER	1	counter for 3-D mode
ICMET	INTEGER	0	counter for meteorological input
ICWAV	INTEGER	0	counter for wave input
IC2BC	INTEGER	0	counter for open boundary input (2-D mode)
ICHBC	INTEGER	0	counter for open boundary input (3-D current, temperature, salinity)
ICBBC	INTEGER	0	counter for open boundary input (biology and sediments)
ICCBC	INTEGER	0	counter for open boundary input (contaminants)
ICPBC	INTEGER	0	counter for open boundary input (Lagrangian particle module)
File parameters			
HEADERS	LOGICAL	.FALSE.	selects header information in ASCII-formatted output files ('A'-files)
GRDFORM	CHARACTER*1	'A'	output format of the file containing grid parameters and arrays
ICRFORM	CHARACTER*1	'A'	output format of the files containing the initial or final conditions
BCSFORM	CHARACTER*1	'A'	output format of the files containing the open boundary data
METFORM	CHARACTER*1	'A'	output format of the meteorological data file
WAVFORM	CHARACTER*1	'A'	output format of the wave data file
MAXFILES	INTEGER	256	maximum allowed number of file connections
OUTTIT	CHARACTER*6	TITLE	first 6 characters of time series, harmonic, time-averaged and particle output files

Table 4.2.3: continued.

name	type	default	purpose
Reference values			
DLAREF	REAL	0.0 degrees	reference latitude
DLOREF	REAL	0.0 degrees	reference longitude
DEPUN	REAL	0.0 m	uniform mean water depth (IGRDIM $= 1$)
R0REF	REAL	1025.0 kg/m^3	reference density ρ_0
SREF	REAL	33.0 PSU	reference salinity S_0
TREF	REAL	12.0 ^{0}C	reference temperature T_0
Expansion coefficients			
SBETUN	REAL	7.6×10^{-4}PSU^{-1}	uniform expansion coefficient β_S for salinity (IOPTD $= 1$)
TBETUN	REAL	1.8×10^{-4} ^{0}C^{-1}	uniform expansion coefficient β_T for temperature (IOPTD $= 1$)
Optical parameters			
ATCF1UN	REAL	10.0 m^{-1}	uniform diffuse attenuation coefficient k_1 for infrared radiation
ATCF2UN	REAL	2.06 m^{-1}	uniform diffuse attenuation coefficient k^w_{20} (non-infrared radiation) for pure seawater
R1OPTUN	REAL	0.54	uniform infrared fraction R_1 of solar irradiance
R2OPTUN	REAL	0.4	exponential fraction R_2 of hyper-exponential PAR decay
EPSSAL	REAL	0.05714 PSU^{-1}m^{-1}	slope parameter ϵ^S in the linear relationships (3.1.159) and (3.2.6).
HEXPUN	REAL	7.0 m	thickness Δ_{opt} of the surface layer for hyper-exponential PAR decay
Background mixing coefficients			
VISMOL	REAL	10^{-6} m^2/s	uniform background eddy viscosity
DIFMOL	REAL	10^{-6} m^2/s	uniform background eddy diffusivity
Horizontal diffusion parameters			
HEDVUN	REAL	0.0 m^2/s	uniform horizontal diffusion coefficient ν_H for momentum (IODIF $= 1$)
HEDDUN	REAL	0.0 m^2/s	uniform horizontal diffusion coefficient λ_H for scalars (IODIF $= 1$)
CM0	REAL	0.1	coefficient C_{m0} in the “Smagorinsky” formulation (3.1.151) or (3.1.153) for ν_H (IODIF $= 2$)
CS0	REAL	0.1	coefficient C_{s0} in the “Smagorinsky” formulation (3.1.151) or (3.1.153) for λ_H (IODIF $= 2$)

Table 4.2.3: continued.

name	type	default	purpose
Bottom friction coefficients			
CDLIN	REAL	0.0 m/s	linear bottom friction coefficient (IBSTR $= 2$)
CDZ0UN	REAL	0.0 m	uniform bottom roughness length (IBSTR $= 1$)
Meteorological forcing			
FSUN	REAL	0.0 $\mathrm{m}^2/\mathrm{s}^2$	uniform X-component of the surface stress divided by ρ_0 (IOPTM $= 1$)
GSUN	REAL	0.0 $\mathrm{m}^2/\mathrm{s}^2$	uniform Y-component of the surface stress divided by ρ_0 (IOPTM $= 1$)
HSUN	REAL	0.0 m	uniform wave height (IOPTW $= 1$)
TWUN	REAL	0.0 s	uniform wave period (IOPTW $= 1$)
Turbulence model parameters			
Richardson number dependent formulations (IOPTK $= 1$)			
PPA	REAL	5.0	constant α_p in the Paconowski-Philander expression (3.1.88)
PPN	REAL	2.0	exponent n_p in the Paconowski-Philander expression (3.1.86)
PPVED0	REAL	$10^{-2}\mathrm{m}^2/\mathrm{s}$	constant ν_{0p} in the Paconowski-Philander expression (3.1.86)
PPVISBG	REAL	$10^{-4}\mathrm{m}^2/\mathrm{s}$	background coefficient ν_{bp} in the Paconowski-Philander expression (3.1.86)
PPDIFBG	REAL	$10^{-5}\mathrm{m}^2/\mathrm{s}$	background coefficient λ_{bp} in the Paconowski-Philander expression (3.1.87)
AMVED0	REAL	0.06 m^2/s	constant ν_{0m} in the Munk-Anderson relations (3.1.95)–(3.1.96)
AMA	REAL	10.0	constant α_m in the Munk-Anderson expression (3.1.97)
AMB	REAL	3.33	constant β_m in the Munk-Anderson expression (3.1.98)
AMN1	REAL	0.5	exponent n_1 in the Munk-Anderson expression (3.1.97)
AMN2	REAL	1.5	exponent n_2 in the Munk-Anderson expression (3.1.98)
RMAXLAT	REAL	4.0	value of the parameter λ_{max} in (3.1.99) (Munk-Anderson formulation)
RMAXNUT	REAL	3.0	value for the parameter ν_{max} in (3.1.92) and (3.1.99)

Table 4.2.3: continued.

name	type	default	purpose
flow-dependent formulations (IOPTK $= 1$, ITFORM $= 3, 4, 5$)			
ADK1	REAL	2.5×10^{-3}	constant K_1 in the relations (3.1.108) and (3.1.110)
ADK2	REAL	2.0×10^{-5}	constant K_2 in the relation (3.1.109)
ADCNU	REAL	2.0	constant C_ν which measures the thickness of the bottom layer in equation (3.1.111)
ADLAM	REAL	0.0	constant λ_* relating the surface friction velocity to the eddy coefficient ν_w in the relation (3.1.106)
ADD1	REAL	0.0	parameter δ_1 for the vertical profile (3.1.103) used in the flow-dependent formulations
ADD2	REAL	0.0	parameter δ_2 for the vertical profile (3.1.103) used in the flow-dependent formulations
ADR1	REAL	1.0	parameter r_1 for the vertical profile (3.1.103) used in the flow-dependent formulations
ADR2	REAL	1.0	parameter r_2 for the vertical profile (3.1.103) used in the flow-dependent formulations
mixing length formulations (IOPTK $= 2$)			
ABLAC	REAL	0.2	the constant α_1 in the expression (3.1.135) for the asymptotic mixing length ("Blackadar" formulation)
BXING	REAL	-2.0	the exponential factor β_1 in the mixing length expression (3.1.133) ("Xing" formulation)
Z0BOT	REAL	0.0 m	bottom roughness length scale z_{0b} as used in (3.1.130), (3.1.126) and (3.1.133)
Z0SUR	REAL	0.0 m	surface roughness length scale z_{0s} as used in (3.1.130) and (3.1.126)
limiting conditions (IOPTK $= 2$)			
TKEMIN	REAL	10^{-6}J/kg	minimum value k_{min} for turbulence energy in the case that limiting conditions are enabled (ILIM $= 1$)

Table 4.2.3: continued.

name	type	default	purpose
Model parameters for the biology			
autotrophs			
PQHI	REAL	40.0 nmol C μE^{-1}	photosynthetic quantum yield Φ
AXQN	REAL	2.0 mg Chl (mmol N)$^{-1}$	chlorophyll to nitrogen ratio ${}^{X}q_{a}^{N}$
AR0	REAL	0.05 day^{-1}	basal respiration rate r_{0_a}
AB	REAL	0.5	slope of respiration/growth relationship b_a
AQMIN	REAL	0.05 mmol N (mmol C)$^{-1}$	minimum nitrogen to carbon quota Q_{min_a}
AQMAX	REAL	0.2 mmol N (mmol C)$^{-1}$	maximum nitrogen to carbon quota Q_{max_a}
AMUMAX	REAL	3.0 day^{-1}	maximum nutrient controlled growth rate μ_{max_a} at 20^{0}C
ANOUMAX	REAL	0.5 mmol N (mmol C)$^{-1}$ day^{-1}	maximum nitrate uptake ${}^{NO}U_{max_a}$ at 20^{0}C
ANHUMAX	REAL	1.5 mmol N (mmol C)$^{-1}$ day^{-1}	maximum ammonium uptake rate ${}^{NH}u_{max_a}$ at 20^{0}C
AKNOS	REAL	0.32 mmol N m^{-3}	half-saturation constant for nitrate uptake k_{NOS}
AKNHS	REAL	0.24 mmol N m^{-3}	half-saturation constant for ammonium uptake k_{NHS}
AKIN	REAL	0.5 mmol N m^{-3}	coefficient for inhibition of nitrate uptake due to ammonium k_{in}
heterotrophs			
ETA	REAL	0.3	ratio η of heterotroph to microplankton biomass
HQ	REAL	0.18 mmol N (mmol C)$^{-1}$	nitrogen to carbon quota q_h
HR0	REAL	0.02 day^{-1}	basal respiration rate r_{0_h}
HB	REAL	1.5	slope of respiration/growth relationship b_h
zooplankton grazing			
GAMMA	REAL	0.8	assimilated proportion γ of grazed microplankton carbon and nitrogen
EX	REAL	0.5	fraction e of nitrogen excreted as ammonium
GRAZUN	REAL	0.05 day^{-1}	uniform value (space and time) for the grazing pressure if the array GRAZING is not defined in subroutine DEFICS (see Table 4.2.7)

Table 4.2.3: continued.

name	type	default	purpose
optical parameters			
EPSSED	REAL	0.1 m^2/g	diffuse PAR attenuation cross section ϵ^A for iSPM
EPSDET	REAL	0.002 m^2 (mmol C)$^{-1}$	diffuse (PAR) attenuation cross section ϵ^C for detrital carbon
EPSBIO	REAL	0.016 m^2 (mg Chl)$^{-1}$	diffuse (PAR) attenuation cross section ϵ^X for chlorophyll
detritus parameters			
RMRMAX	REAL	0.08 day^{-1}	maximum relative rate of detrital nitrogen remineralisation $^Mr_{max}$ at 20^0C
CRMAX	REAL	0.06 day^{-1}	maximum detrital carbon remineralisation rate $^Cr_{max}$ at 20^0C
COKS	REAL	10.0 mmol O m^{-3}	half-saturation constant for nitrogen-dependent detrital carbon respiration $O_{1/2}$
QMCMIN	REAL	0.09 mmol N (mmol C)$^{-1}$	minimum of detrital nitrogen to detrital carbon quota $^Mq^C_{min}$
temperature growth factor			
Q10	REAL	0.07 (^{0}C)$^{-1}$	growth rate coefficient q_T in temperature growth factor
REFTEMP	REAL	20.0 ^{0}C	reference temperature T_r in temperature growth factor
dissolved nutrients and oxygen parameters			
QOB	REAL	1.0 mmol O (mmol C)$^{-1}$	photosynthetic and respiratory quotient for microplankton carbon growth $^Oq^B$ and detrital respiration $^Oq^C$
QON	REAL	2.0 mmol O (mmol N)$^{-1}$	photosynthetic and respiratory quotient for nitrate uptake $^Oq^{NO}$ and nitrification $^Oq^{NH}$
RNHMAX	REAL	0.1 day^{-1}	maximum rate of nitrification $^{NH}r_{max}$ at 20^0C
CUO	REAL	30.0 mmol O m^{-3}	half-saturation constant $O_{1/2,nit}$ for oxygen used in nitrification

Table 4.2.3: continued.

name	type	default	purpose
sinking velocities			
WSDET	REAL	-5.0 m/day	uniform vertical sinking velocity $w_{s_{det}}$ for detritus
WSMAX	REAL	-5.0 m/day	maximum vertical sinking velocity $w_{s_{max}}$ for microplankton
WSMIN	REAL	-0.5 m/day	minimum vertical sinking velocity $w_{s_{min}}$ for microplankton
bottom and surface fluxes			
FRATE	REAL	0.02 day^{-1}	relaxation rate F_{br} for bottom ammonium and nitrate fluxes
BNOS	REAL	5.0 mmol N m^{-3}	assumed nitrate concentration $^{NO}S_b^0$ in the consolidated sediment for benthic flux
BNHS	REAL	1.0 mmol N m^{-3}	assumed ammonium concentration $^{NH}S_b^0$ in the consolidated sediment for benthic flux
CWIND	REAL	5×10^{-7} s/m	factor k_w to determine surface transfer coefficient for oxygen
ASAT	REAL	2.74 × 10^{-3} m^3 (mmol O)$^{-1}$	parameter a_{sT} to determine air oxygen concentration
BSAT	REAL	7.8×10^{-5} m^3 (mmol O)$^{-1}$ (^{0}C^{-1})	parameter b_{sT} to determine air oxygen concentration
Model parameters for the sediment module			
ALPHAS	REAL	0.0025 g m^{-2} s^{-1}	parameter α_s in the formulation (3.2.72) for the resuspension rate
RNSED	REAL	3.0	exponential factor n_s in the formulation (3.2.72) for the resuspension rate
WSED1	REAL	-0.002 m/s	uniform sinking velocity w_s^A for inorganic sediment
Model parameters for the Lagrangian particle module			
STLSOL	REAL	100.0 ton	mass m_{riv} of the particles which enter the domain through a river open boundary
STSSOL	REAL	100.0 ton	mass m_{sea} of the particles which enter the domain through an open sea boundary
HEDPUN	REAL	0.0 m^2/s	uniform horizontal diffusion coefficient used in the particle module when IOPTP = 2

Table 4.2.4: Grid parameters defined in DEFGRID.

name	type	dimensions	unit	purpose
GX0	REAL	NC+1	m or degrees	x_1-coordinate ($x_{1;i}$) or longitude (λ_i) of the cell corners
GY0	REAL	NR+1	m or degrees	x_2-coordinate ($x_{2;j}$) or latitude (ϕ_j) of the cell corners
GZ0	REAL	NZ+1	—	values of the vertical σ-coordinate at the W-nodes
DEP	REAL	NR,NC	m	array of mean water depths
NWD	INTEGER	NR,NC	—	pointer array evaluated at the cell centres
NPIX	INTEGER	NR,NC+1	—	pointer array evaluated at the U-nodes
NPIY	INTEGER	NR+1,NC	—	pointer array evaluated at the V-nodes
IOBU	INTEGER	0:NOBU	—	X-indices of the cell faces at U-open boundaries
JOBU	INTEGER	0:NOBU	—	Y-indices of the cell faces at U-open boundaries
IOBV	INTEGER	0:NOBV	—	X-indices of the cell faces at V-open boundaries
JOBV	INTEGER	0:NOBV	—	Y-indices of the cell faces at V-open boundaries
CDZ0	REAL	NR,NC	m	array of values for the bottom roughness length z_0 (IBSTR $= 1$)

Table 4.2.5: Open boundary conditions for the 2-D mode defined in DEFOB2D and WROBDAT.

name	type	dimensions	unit	purpose
ITYPOBU	INTEGER	0:NOBU	—	type of condition at U-open boundaries
ITYPOBV	INTEGER	0:NOBV	—	type of condition at V-open boundaries
SIGMA	REAL	0:NCON	rad/s	tidal forcing frequencies
PHAS0	REAL	0:NCON	rad	the initial phases φ_{n0}
ANALSIG	REAL	0:NCONTO	rad/s	frequencies for harmonic analysis
AMPOBU[a]	REAL	0:NOBU,0:NCON	m	the residual part A_0 and the amplitudes A_n of the forcing at U-open boundaries
AMPOBV[a]	REAL	0:NOBV,0:NCON	m	the residual part A_0 and the amplitudes A_n of the forcing at V-open boundaries
PHAOBU[a]	REAL	0:NOBU,0:NCON	rad	the phases φ_n of the tidal forcing at the U-open boundaries
PHAOBV[a]	REAL	0:NOBV,0:NCON	rad	the phases φ_n of the tidal forcing at the V-open boundaries

[a]This array has a different meaning in the case of a 1-D application. See Section IV-2.9 for details.

Table 4.2.6: Open boundary conditions and arrays for 3-D quantities defined in DEFOB3D and WROBDAT.

name	type	dimensions	unit	purpose
parameter arrays				
IVPOBU	INTEGER	0:NOBU	—	profile number at U-open boundaries
IVPOBV	INTEGER	0:NOBV	—	profile number at V-open boundaries
LSTOBU	INTEGER	0:NOBU	—	particle labels at U-open boundaries
LSTOBV	INTEGER	0:NOBV	—	particle labels at V-open boundaries
open boundary arrays for physical input				
SVP	REAL	0:NZ,0:NVPROF	PSU	salinity
TVP	REAL	0:NZ,0:NVPROF	^{0}C	temperature
UVP	REAL	0:NZ,0:NVPROF	m/s	currents
open boundary arrays for biological and sediment input				
P2BVP	REAL	0:NZ,0:NVPROF	mmol C/m^3	microplankton carbon biomass B
P2CVP	REAL	0:NZ,0:NVPROF	mmol C/m^3	detrital carbon C
P2MVP	REAL	0:NZ,0:NVPROF	mmol N/m^3	detrital nitrogen M
P2NVP	REAL	0:NZ,0:NVPROF	mmol N/m^3	microplankton nitrogen biomass N
P2NHSVP	REAL	0:NZ,0:NVPROF	mmol N/m^3	ammonium ^{NH}S
P2NOSVP	REAL	0:NZ,0:NVPROF	mmol N/m^3	nitrate ^{NO}S
SEDC1VP	REAL	0:NZ,0:NVPROF	g/m^3	suspended sediment A
open boundary arrays for contaminant input				
CONVP	REAL	0:NZ,0:NVPROF,0:NCONC	g/m^3	contaminant concentrations
input of suspended material for Lagrangian transport				
QSTOBU	REAL	0:NOBU	ton/m^3 or ton/s	input at U-open boundaries
QSTOBV	REAL	0:NOBV	ton/m^3 or ton/s	input at V-open boundaries

Table 4.2.7: Initial conditions used as input for the main program.

name	type	dimensions	unit	purpose
currents				
UD2	REAL	NR,NC+1	m^2/s	depth-integrated current $\overline{U}$
VD2	REAL	NR+1,NC	m^2/s	depth-integrated current $\overline{V}$
U2	REAL	NZ,NR,NC+1	m/s	horizontal current u
V2	REAL	NZ,NR+1,NC	m/s	horizontal current v
W2	REAL	NZ+1,NR,NC	m/s	transformed vertical velocity $J\tilde{w}$
U2F	REAL	NZ,NR,NC+1	m/s	"filtered" horizontal current u^F
V2F	REAL	NZ,NR+1,NC	m/s	"filtered" horizontal current v^F
bottom stress				
FB	REAL	NR,NC+1	m^2/s^2	bottom stress component τ_{b1} or $\tau_{b\lambda}$ at U-nodes divided by ρ_0
GB	REAL	NR+1,NC	m^2/s^2	bottom stress component τ_{b1} or $\tau_{b\phi}$ at V-nodes divided by ρ_0
surface elevation, grid spacing				
ZETA2	REAL	NR,NC	m	surface elevation ζ
GZ2	REAL	NZ,NR,NC	m	vertical grid spacing Δx_3 (or Jacobian J)
salinity, temperature				
S	REAL	NZ,NR,NC	PSU	salinity S
T	REAL	NZ,NR,NC	^0C	temperature T
open boundary arrays[a]				
R1OBU	REAL	0:NOBU	m^2/s	incoming characteristic $R^u_\pm$ at U-open boundaries
R2OBU	REAL	0:NOBU	m^2/s	outgoing characteristic $R^u_\mp$ at U-open boundaries
R1OBV	REAL	0:NOBV	m^2/s	incoming characteristic $R^v_\pm$ at V-open boundaries
R2OBV	REAL	0:NOBV	m^2/s	outgoing characteristic $R^v_\mp$ at V-open boundaries
initial phases				
PHAS0	REAL	0:NCON	radians	initial phases φ_{n0}

[a]The upper sign applies at western or southern boundaries, the lower sign at eastern or northern boundaries.

Table 4.2.7: continued.

name	type	dimensions	unit	purpose
turbulence arrays				
VEDDYV	REAL	NZ+1,NR,NC	m^2/s	eddy viscosity ν_T
VEDDYD	REAL	NZ+1,NR,NC	m^2/s	eddy diffusivity λ_T
VEDDYK	REAL	NZ+1,NR,NC	m^2/s	vertical diffusion coefficient ν_T/σ_k in the k- and kl-equation
VEDDYE	REAL	NZ+1,NR,NC	m^2/s	vertical diffusion coefficient ν_T/σ_ε in the ε-equation
SMU	REAL	NZ+1,NR,NC	—	stability coefficient S_m or S_u
SHB	REAL	NZ+1,NR,NC	—	stability coefficient S_h or S_b
TKEW	REAL	NZ+1,NR,NC	J/kg	turbulence energy k
ZLW	REAL	NZ+1,NR,NC	m	mixing length l
DISSW	REAL	NZ+1,NR,NC	W/kg	dissipation rate ε
biological and sediment arrays				
P2B	REAL	NZ,NR,NC	mmol C m^{-3}	microplankton carbon biomass B
P2C	REAL	NZ,NR,NC	mmol C m^{-3}	detrital carbon C
P2M	REAL	NZ,NR,NC	mmol N m^{-3}	detrital nitrogen M
P2N	REAL	NZ,NR,NC	mmol N m^{-3}	microplankton nitrogen biomass N
P2NHS	REAL	NZ,NR,NC	mmol N m^{-3}	ammonium ${}^{NH}S$
P2NOS	REAL	NZ,NR,NC	mmol N m^{-3}	nitrate ${}^{NO}S$
P2O	REAL	NZ,NR,NC	mmol O m^{-3}	dissolved oxygen O
P2ZN	REAL	NZ,NR,NC	mmol N m^{-3}	zooplankton nitrogen Z_N
PAR	REAL	NZ,NR,NC	W/m^2	photosynthetically active radiation
GRAZING	REAL	NR,NC,12	day^{-1}	grazing pressure
SEDC1	REAL	NZ,NR,NC	g/m^3	suspended sediment A
FSED	REAL	NR,NC,5	amount m^{-2}	fluff layer amounts
FLUFLOS	REAL	NR,NC,5	amount m^{-2}	amount of organic material lost to the consolidated sediment since the initial time $t = 0$

Table 4.2.7: continued.

name	type	dimensions	unit	purpose
Lagrangian particle transport				
IT	INTEGER	0:MAXNOP	—	X-indices of the cells where the particles are located
JT	INTEGER	0:MAXNOP	—	Y-indices of the cells where the particles are located
KT	INTEGER	0:MAXNOP	—	Z-indices of the cells where the particles are located
XT	REAL	0:MAXNOP	m or degrees	x'_{1n}-coordinates of the particles with respect to the cell centre
YT	REAL	0:MAXNOP	m or degrees	x'_{2n}-coordinates of the particles with respect to the cell centre
ZT	REAL	0:MAXNOP	—	σ'_n-coordinates of the particles with respect to the cell centre
LST	INTEGER	0:MAXNOP	—	labels of the particles inside the domain
ST	REAL	0:MAXNOP	ton	masses of the particles
STIN	REAL	NZ,NR,NC	ton	amount of suspended matter which entered the domain
STOUT	REAL	NZ,NR,NC	ton	amount of suspended matter which left the domain
RMASSU	REAL	0:NOBU	ton	remaining mass fraction at U-open boundaries
RMASSV	REAL	0:NOBV	ton	remaining mass fraction at V-open boundaries
contaminants				
CONCN	REAL	NZ,NR,NC,0:NCONC	g/m^3	contaminant concentrations

Table 4.2.8: Meteorological and wave forcing data defined in WRFCDAT.

name	type	dimensions	unit	purpose
meteorological data				
WINDU2	REAL	NR,NC	m/s	X-component of the wind vector at 10 m height
WINDV2	REAL	NR,NC	m/s	Y-component of the wind vector at 10 m height
P2	REAL	NR,NC	N/m^2	atmospheric pressure
SAT2	REAL	NR,NC	^0C	air temperature
HUM2	REAL	NR,NC	—	relative humidity between 0 and 1
CLOUD2	REAL	NR,NC	—	fractional cloud cover between 0 and 1
EVAPR	REAL	NR,NC	$kg/m^2/s$	evaporation minus precipitation rate
RAIN2	REAL	NR,NC	$kg/m^2/s$	precipitation rate
QNSOL	REAL	NR,NC	W/m^2	non-solar heat flux Q_{nsol} (positive upwards)
QSOL	REAL	NR,NC	W/m^2	surface solar heat flux Q_{rad} (always positive)
wave data				
HS	REAL	NR,NC	m	significant wave height
TW	REAL	NR,NC	s	mean wave period

Chapter 3

Preparing Model Output

This chapter explains how model output can be defined by the user. The aim is to provide guidelines to create

- the parameter file *defmod.par* where the output specifiers are defined (output grid locations and time steps, file formats, number of output variables, period for harmonic analysis and time averaging)
- the files *defout.f*, *defanal.f*, *defavr.f* and *defpar.f* where the program variables for output are defined
- the file *print.f* where the user can define other output data specific for a particular application

3.1 File *defmod.par*

This file is composed of a number of lines on which several parameters are defined. No generic example is available but examples can be found in all **/tests** subdirectories. The names and order in which each parameter needs to be supplied, are given below. The user has to replace the FORTRAN names by the appropriate data values which are either of type CHARACTER or INTEGER. The line number before the ":" and the ":" do not appear in the actual file.

```
1 : TITLE
A1: NARROUT
A2: N2VECO N2SCALO N3VECO N3SCALO
A3: ARRFORM(IARR)
A4: LIMOUT(1,1,IARR) LIMOUT(2,1,IARR) LIMOUT(3,1,IARR)
A5: LIMOUT(1,2,IARR) LIMOUT(2,2,IARR) LIMOUT(3,2,IARR)
A6: LIMOUT(1,3,IARR) LIMOUT(2,3,IARR) LIMOUT(3,3,IARR)
A7: LIMOUT(1,4,IARR) LIMOUT(2,4,IARR) LIMOUT(3,4,IARR)
B1:
```

```
B2: N2VECA N2SCALA N3VECA N3SCALA
B3: ANALFORM
B4: LIMANAL(1,1) LIMANAL(2,1) LIMANAL(3,1)
B5: LIMANAL(1,2) LIMANAL(2,2) LIMANAL(3,2)
B6: LIMANAL(1,3) LIMANAL(2,3) LIMANAL(3,3)
B7: LIMANAL(1,4) LIMANAL(2,4) LIMANAL(3,4)
C1:
C2: NARRAVR
C3: N2VECM N2SCALM N3VECM N3SCALM
C4: AVRFORM(IARR)
C5: LIMAVR(1,1,IARR) LIMAVR(2,1,IARR) LIMAVR(3,1,IARR)
C6: LIMAVR(1,2,IARR) LIMAVR(2,2,IARR) LIMAVR(3,2,IARR)
C7: LIMAVR(1,3,IARR) LIMAVR(2,3,IARR) LIMAVR(3,3,IARR)
C8: LIMAVR(1,4,IARR) LIMAVR(2,4,IARR) LIMAVR(3,4,IARR)
D1:
D2: NOPOUT
D3: PARFORM
D4: LIMPOUT(1) LIMPOUT(2) LIMPOUT(3)
```

- Lines A1–A7 need only to be present if IOUTS = 1.
- Lines B1–B7 need only to be present if IOUTT = 1 or 2.
- Lines C1–C8 need only to be present if IOUTF = 1.
- Lines D1–D4 need only to be present if IOUTP = 1.
- Lines B1, C1 and D1 can be used as comment lines but they need to be there (if the appropriate switch is set).
- Lines A3-A7 are repeated NARROUT times, i.e. for IARR = 1 to NARROUT.
- Lines C4-C8 are repeated NARRAVR times, i.e. for IARR = 1 to NARRAVR.

The string variable TITLE is read using "A6" format, the other string variables ARRFORM, ANALFORM, AVRFORM and PARFORM using "A1" format. All other parameter are of type INTEGER and read using free format.

The input parameters have the following meaning for the program.

TITLE
The title of the simulation which is always composed of 6 characters.

NARROUT
The number of files for time series output.

N2VECO
: The number of 2-D vector fields for time series output as defined in the file *defout.f.*

N2SCALO
: The number of 2-D scalar fields for time series output as defined in the file *defout.f.*

N3VECO
: The number of 3-D vector fields for time series output as defined in the file *defout.f.*

N3SCALO
: The number of 3-D scalar fields for time series output as defined in the file *defout.f.*

ARRFORM(NARROUT)
: The format of the time series output files:

  'A' : ASCII-format

  'U' : unformatted (binary)

LIMOUT(3,4,NARROUT)
: Determines the grid locations and the time instants for time series output. If (K,J,I) or (J,I) are the spatial indices of a 3-D or a 2-D array, NT the time counter representing the number of 2-D time steps elapsed since the start of the simulation and IARR the number of the output file, output is produced at the grid points and time levels given by the following expression (where the first number after the equal sing denotes the start value, the second the end value and the third the step value):

```
I = LIMOUT(1,1,IARR), LIMOUT(2,1,IARR), LIMOUT(3,1,IARR)
J = LIMOUT(1,2,IARR), LIMOUT(2,2,IARR), LIMOUT(3,2,IARR)
K = LIMOUT(1,3,IARR), LIMOUT(2,3,IARR), LIMOUT(3,3,IARR)
NT= LIMOUT(1,4,IARR), LIMOUT(2,4,IARR), LIMOUT(3,4,IARR)
```

  The number of times that times series output is produced is then given by

```
(LIMOUT(2,4,IARR)-LIMOUT(1,4,IARR))/LIMOUT(3,4,IARR) + 1
```

  Although this is not explicitly required by the program, it is recommended to use a value for the time step parameter LIMOUT(3,4,IARR) which is a multiple of the counter IC3D. Note also that the difference between an end and a start value LIMOUT(2,*,IARR)-LIMOUT(1,*,IARR) must be an integer multiple of the corresponding step value LIMOUT(3,*,IARR).

N2VECA
: The number of 2-D vector fields for harmonic analysis as defined in the file *defanal.f.* As explained in Section IV-3.3.2, this parameter must be non-zero if IOUTT $= 2$.

N2SCALA

The number of 2-D scalar fields for harmonic analysis as defined in the file *defanal.f*.

N3VECA

The number of 3-D vector fields for harmonic analysis as defined in the file *defanal.f*. As explained in Section IV-3.3.1, this parameter must be non-zero if IOUTT = 2.

N3SCALA

The number of 3-D scalar fields for harmonic analysis as defined in the file *defanal.f*.

ANALFORM

The format of the harmonic output files:

'A' : ASCII-format

'U' : unformatted (binary)

LIMANAL(3,4)

Determines the grid locations, the time instants, the period T and the central time t_c for harmonic analysis and output. The grid locations are specified on the lines B4–B6 which have the same meaning as the corresponding ones in the array LIMOUT for time series output. Line B7 has the following meaning. The harmonic period T is defined as the product of the index LIMANAL(3,4) with the 2-D time step DELT. Two instants t_1 and t_2, measured relative to the initial startup time of the program, are defined by the time counters LIMANAL(1,4) and LIMANAL(2,4) multiplied by DELT. Harmonic analysis is then performed for each of the time intervals (t_1, t_1+T), (t_1+T, t_1+2T), ..., (t_2-T, t_2). The corresponding central times are then given by $(t_1+T/2$,$t_1+3T/2$,...,$t_2-T/2)$. The results of the analysis are written to the appropriate output files for each central time, i.e. at

```
NT = LIMANAL(1,4)+LIMANAL(3,4)/2, LIMANAL(2,4)-LIMANAL(3,4)/2,
     LIMANAL(3,4)
```

where NT is the number of time steps elapsed since the start of the simulation and the numbers after the equal sign represent respectively the start, end and step value of the time index NT. The number of times that harmonically analysed output is produced is then given by

```
(LIMANAL(2,4)-LIMANAL(1,4))/LIMANAL(3,4)
```

The time counters LIMANAL(1,4) and LIMANAL(2,4) must be integer multiples of the counter IC3D. As explained in Section III-1.7 the period T contains an even number of 3-D time steps so that LIMANAL(3,4) must be an even multiple of IC3D. In analogy with the array LIMOUT the difference between an end and a start value LIMANAL(2,*)-LIMANAL(1,*) must be an integer multiple of the corresponding step value LIMANAL(3,*). Note that the phases of each harmonic component are determined with respect to the corresponding central time t_c.

NARRAVR
The number of files for time-averaged output.

N2VECM
The number of 2-D vector fields for time-averaged output as defined in the file *defavr.f*.

N2SCALM
The number of 2-D scalar fields for time-averaged output as defined in the file *defavr.f*.

N3VECM
The number of 3-D vector fields for time-averaged output as defined in the file *defavr.f*.

N3SCALM
The number of 3-D scalar fields for time-averaged output as defined in the file *defavr.f*.

AVRFORM(NARRAVR)
The format of the time-averaged output files:

'A' : ASCII-format

'U' : unformatted (binary)

LIMAVR(3,4,NARRAVR)
Determines the grid locations, the time instants and the period T for time averaging. The grid locations are specified on the lines C5–C7 which have the same meaning as the corresponding ones in the array LIMOUT for time series output. Line C8 has the following meaning. The period T for time averaging is defined as the product of the index LIMAVR(3,4,IARR) with the 2-D time step DELT. Two instants t_1 and t_2, measured relative to the initial startup time of the program, are defined by the time counters LIMAVR(1,4,IARR) and LIMAVR(2,4,IARR) multiplied by DELT. Time averaging is then performed for each of the time intervals (t_1, t_1+T), (t_1+T, t_1+2T), $\ldots$, (t_2-T, t_2). Time-averaged results are written to the appropriate output files for each central time, i.e. at

```
NT = LIMAVR(1,4,IARR)+LIMAVR(3,4,IARR)/2,
     LIMAVR(2,4,IARR)-LIMAVR(3,4,IARR)/2, LIMAVR(3,4,IARR)
```

where NT is the number of time steps elapsed since the start of the simulation and the numbers after the equal sign represent respectively the start, end and step value of the time index NT. The number of times that time-averaged output is produced is then given by

```
(LIMAVR(2,4,IARR)-LIMAVR(1,4,IARR))/LIMAVR(3,4,IARR)
```

The time counters LIMAVR(1,4,IARR), LIMAVR(2,4,IARR), LIMAVR(3,4,IARR) must be integer multiples of the counter IC3D. In analogy with the array LIMOUT the difference between an end and a start value LIMAVR(2,*,IARR)-LIMAVR(1,*,IARR) must be an integer multiple of the corresponding step value LIMAVR(3,*,IARR).

NOPOUT
: The total number of output particle trajectories.

PARFORM
: The format of the output file with the particle trajectories:

  'A' : ASCII-format

  'U' : unformatted (binary)

LIMPOUT(3)
: Determines the time instants for the output of the particle positions. This occurs when the time counter NT has the values

```
NT = LIMPOUT(1), LIMPOUT(2), LIMPOUT(3)
```

where the first number after the equal sign denotes the start value, the second the end value and the the third the step value. The number of output times is then given by

```
(LIMPOUT(2)-LIMPOUT(1))/LIMPOUT(3) + 1
```

All elements of LIMPOUT must be integer multiples of the counter IC3D. As before the difference LIMPOUT(2)-LIMPOUT(1) must be a multiple of the step value LIMPOUT(3).

3.2 File *defout.f*

To define the variables for time series output the user must create a FORTRAN file *defout.f* containing the following four FORTRAN subroutines:

- OUTPT3D where the 3-D quantities for time series output are defined
- OUTPT2D where the 2-D quantities for time series output are defined
- OUT3DVAR where the names, descriptions and units of all 3-D output quantities are defined
- OUT2DVAR where the names, descriptions and units of all 2-D output quantities are defined

Each output data file has an associated "Information" file containing the names, descriptions and units of all variables and other useful information which can be used in a post-processor program to provide e.g. plotting titles and axis labels (see Section IV-4.2.2a for details). All quantities must be defined at the cell centres. This means in particular that quantities evaluated at U-, V- or W-nodes must be interpolated along the X-, Y- or Z-direction. The file must always be supplied by the user if time series output is selected, i.e. IOUTS = 1. The empty default file in the **SOURCE** directory is only needed to prevent that an error occurs during compilation if IOUTS = 0 and no file is specified by the user.

3.2.1 Subroutine OUTPT3D

In this routine the 3-D quantities for time series output are defined and stored into the array OUT3D. The variables must be defined in the following order:

- 3-D vector fields with their X-, Y- and Z-components giving a total number of 3*N3VECO quantities
- 3-D scalar fields with a total number given by N3SCALO

Note that in the example file the current is the only 3-D output vector field, i.e. N3VECO = 1. The array W2PHYS denotes the vertical velocity w in the non-transformed coordinate system as defined by equation (3.1.47) or (3.1.69). If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must appear[1]. Output variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION. It is recommended to set all array values to zero on dry cells.

3.2.2 Subroutine OUTPT2D

In this routine the 2-D quantities for time series output are defined and stored into the array OUT2D. The variables must be defined in the following order:

- 2-D vector fields with their X-, Y- and Z-components giving a total number of 2*N2VECO quantities
- 2-D scalar fields with a total number given by N2SCALO

It is preferred to define the sea surface elevation ZETA2 as the first 2-D scalar quantity which can be used in a postprocessor program to plot the sea surface elevation and to determine the total water depth. Note that in the example the depth-averaged current is the only 2-D output vector field, i.e. N2VECO = 1. If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must be present[1]. Output variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION such as the function H2ATC, representing the total depth H, in the example. It is recommended to set all array values to zero on dry cells.

[1]A list of all program variables, referenced in a COMMON block, with their FORTRAN name and the name of the corresponding INCLUDE file is given in Chapter V-2 of the Reference Manual.

3.2.3 Subroutine OUT3DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES3
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 3-D output variables (e.g. "U2" for the horizontal current u).

DESCS3
: An array of strings composed of maximum 30 characters representing descriptions of the 3-D output variables (e.g. "U-velocity").

UNITS3
: An array of strings composed of maximum 16 characters representing the units of the 3-D output variables (e.g. "m/s").

3.2.4 Subroutine OUT2DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES2
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 2-D output variables (e.g. "ZETA2" for the surface elevation).

DESCS2
: An array of strings composed of maximum 30 characters representing descriptions of the 2-D output variables (e.g. "Sea surface elevation").

UNITS2
: An array of strings composed of maximum 16 characters representing the units of the 2-D output variables (e.g. "m").

3.3 File *defanal.f*

The program variables on which harmonic analysis is performed, are defined in the file *defanal.f* which must be created by the user. This file contains the following five FORTRAN subroutines and functions:

- ANALPT3D where the 3-D quantities for harmonic analysis are defined
- ANALPT2D where the 2-D quantities for harmonic analysis are defined
- ANAL3DVAR where the names, descriptions and units of all 3-D quantities on which harmonic analysis is performed, are defined
- ANAL2DVAR where the names, descriptions and units of all 2-D quantities on which harmonic analysis is performed, are defined

- the CHARACTER*3 FUNCTION ANALFR(N) where N takes a value between 1 and NCONTO

Each output data file has an associated "Information" file containing the names, descriptions and units of all variables and other useful information which can be used in a post-processor program to provide e.g. plotting titles and axis labels (see Section IV-4.2.2a for details). All quantities must be defined at the cell centres. This means in particular that quantities evaluated at U-, V- or W-nodes must be interpolated along the X-, Y- or Z-direction. The file must always be supplied by the user if harmonic analysis is selected, i.e. IOUTT = 1 or 2. The empty default file in the **SOURCE** directory is only needed to prevent that an error occurs during compilation if IOUTT = 0 and no file is specified by the user.

3.3.1 Subroutine ANALPT3D

In this routine the 3-D quantities for harmonic analysis are defined and stored into the array ANAL3D. The variables must be defined in the following order:

- 3-D vector fields with their X-, Y- and Z-components giving a total number of 3*N3VECA quantities
- 3-D scalar fields with a total number given by N3SCALA

Important to note is that N3VECA must be non-zero if IOUTT = 2. In that case the current vector must be the first vector field (as in the example) which is used in the program to derive the tidal ellipse parameters. If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must appear[1]. Variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION. It is recommended to set all array values to zero on dry cells.

3.3.2 Subroutine ANALPT2D

In this routine the 2-D quantities for harmonic analysis are defined and stored into the array ANAL2D. The variables must be defined in the following order:

- 2-D vector fields with their X- and Y-components giving a total number of 2*N2VECA quantities
- 2-D scalar fields with a total number given by N2SCALA

Important to note is that N2VECA must be non-zero if IOUTT = 2. In that case the depth-averaged current must be the first vector field (as in the example) which is used in the program to derive the tidal ellipse parameters. If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must be present[1]. Variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION. It is recommended to set all array values to zero on dry cells.

3.3.3 Subroutine ANAL3DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES3
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 3-D variables for harmonic output.

DESCS3
: An array of strings composed of maximum 17 characters representing descriptions of the 3-D variables for harmonic output.

UNITS3
: An array of strings composed of maximum 16 characters representing the units of the 3-D variables for harmonic output.

3.3.4 Subroutine ANAL2DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES2
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 2-D variables for harmonic output.

DESCS2
: An array of strings composed of maximum 17 characters representing descriptions of the 2-D variables for harmonic output.

UNITS2
: An array of strings composed of maximum 16 characters representing the units of the 2-D variables for harmonic output.

3.3.5 Function ANALFR

A series of strings composed of maximum 2 characters representing the name of the analysed frequencies must be supplied in a DATA statement as shown in the example file where "M2" and "S2" are assumed to be the first frequencies. The purpose is to create a series of strings with a full description of each output variable, e.g. "M2-amplitude U-velocity". They are written to the appropriate "Information" files. Note that the number of data in the DATA statement must be equal to the value of the parameter NCONTO.

3.4 File *defavr.f*

The variables for time-averaged output are defined in the file *defavr.f* which must be created by the user. The file contains the following four FORTRAN subroutines:

- AVRPT3D where the 3-D quantities for time-averaged output are defined
- AVRPT2D where the 2-D quantities for time-averaged output are defined
- AVR3DVAR where the names, descriptions and units of all 3-D quantities for time-averaged output are defined
- AVR2DVAR where the names, descriptions and units of all 2-D quantities for time-averaged output are defined

Each output data file has an associated "Information" file containing the names, descriptions and units of all variables and other useful information which can be used in a post-processor program to provide e.g. plotting titles and axis labels (see Section IV-4.2.2a for details). All quantities must be defined at the cell centres. This means in particular that quantities evaluated at U-, V- or W-nodes must be interpolated along the X-, Y- or Z-direction. The file must always be supplied by the user if time-averaged output is selected, i.e. IOUTF = 1. The empty default file in the **SOURCE** directory is only needed to prevent that an error occurs during compilation if IOUTF = 0 and no file is specified by the user.

3.4.1 Subroutine AVRPT3D

In this routine the 3-D quantities for time-averaged output are defined and stored into the array OUT3D. The variables must be defined in the following order:

- 3-D vector fields with their X-, Y- and Z-components giving a total number of 3*N3VECM quantities
- 3-D scalar fields with a total number given by N3SCALM

Note that in the example file the current is the only 3-D output vector field, i.e. N3VECM = 1. If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must appear[1]. Output variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION. It is recommended to set all array values to zero on dry cells.

3.4.2 Subroutine AVRPT2D

In this routine the 2-D quantities for time-averaged output are defined and stored into the array OUT2D. The variables must be defined in the following order:

- 2-D vector fields with their X-, Y- and Z-components giving a total number of 2*N2VECM quantities
- 2-D scalar fields with a total number given by N2SCALM

Note that in the example file the depth-averaged current is the only 2-D output vector field, i.e. N2VECM = 1. If a variable is referenced in a COMMON block, the appropriate INCLUDE statement must be present[1]. Output variables can also be defined using a CALL to an existing or user-created SUBROUTINE or FUNCTION. It is recommended to set all array values to zero on dry cells.

3.4.3 Subroutine AVR3DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES3
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 3-D output variables.

DESCS3
: An array of strings composed of maximum 30 characters representing descriptions of the 3-D output variables.

UNITS3
: An array of strings composed of maximum 16 characters representing the units of the 3-D output variables.

3.4.4 Subroutine AVR2DVAR

The following string arrays which are written to the "Information" file, are defined here:

NAMES2
: An array of strings composed of maximum 8 characters representing the FORTRAN names of the 2-D output variables.

DESCS2
: An array of strings composed of maximum 30 characters representing descriptions of the 2-D output variables.

UNITS2
: An array of strings composed of maximum 16 characters representing the units of the 2-D output variables.

3.5 File *defpar.f*

This file contains the subroutine PARPT3D where the particle coordinates for the output trajectories are defined. The total number of output particles is given by the parameter NOPOUT defined in the input file *defmod.par*. The particle coordinates are defined in the program with respect to the centre of the grid box where the particle is located. The coordinates of a particle "N" with respect to the Cartesian or spherical reference frame are obtained as follows. Let II = IT(N), JJ = JT(N), KK = KT(N) the indices of the grid cell containing the particle. The "relative" coordinates with respect to the cell centre are given by XT(N), YT(N), ZT(N). The "absolute" coordinates are then:

```
0.5*(GX0(II)+GX0(II+1)) + XT(N)
0.5*(GY0(JJ)+GY0(JJ+1)) + YT(N)
(0.5*(GZ0(KK)+GZ0(KK+1))+ZT(N))*H2ATC(JJ,II)
```

where H2ATC is the total water depth and (GX0, GY0) are expressed in Cartesian coordinates if IGTRH = 0 and in spherical coordinates (longitude and latitude in decimal degrees) if IGTRH = 1. The absolute coordinates are stored into the array PAR3D. This array is written to one output file which has an associated "Information" file containing useful information for a postprocessor program (see Section IV-4.2.2b for details).

If a large number of particles is considered in the simulation, the number of output trajectories can be limited by selecting particles with a given label which is stored in the array LST. The label of a particle does not change during program execution except that the label is set to zero if the particle either exits the domain or crosses a land boundary. Other useful output trajectories such as for the centre of mass may be defined here by the user.

Interesting to note here is that the program evaluates, at each time step, the averaged SPM concentration over each grid cell by summing the masses of all particles inside the cell and dividing by the cell volume. The result is stored into the array SPMCON(NZ,NR,NC) which can be sent to output using one of the procedures discussed in the previous sections.

3.6 File *print.f*

The procedures described in the previous sections enable the program to write output in a "standard" format. In some cases it may be more useful to define output with a different format. This may occur for example if time series output is requested at a number of stations spread irregularly over the domain or at unequal time intervals, or if output data are required along a transect which is not parallel to the grid axes. The program therefore offers the possibility to define other forms of output. This performed by creating the file *print.f* containing the subroutine PRINT in the **PROJECT** directory. The PRINT routine is called by the main program at each 3-D time step. If no file is supplied by the user, the program automatically compiles with the empty default version in the **SOURCE** directory. The user is responsible for its contents except that it must obviously contain the following lines

```
SUBROUTINE PRINT
...
RETURN
```

The PRINT has been used to define the output parameters for the test case simulations. Example *print.f*-files can be found in the **/tests** subdirectories containing the source code files of the test cases.

Chapter 4

Input/Output Operations

The aim of this chapter is to describe

- all input and output files produced by the program
- the input and output routines

4.1 I/O files

An overview of all the input and output files connected to the preprocessor and the main program is presented schematically in Figure 4.4.1 and can be summarised as follows:

- The preprocessor reads input from the parameter file *defmod.par* and from a series of files supplied by the user.
- The preprocessor writes a series of output files (described in Section IV-4.1.1) with the model parameters, initial conditions, the data at the open boundaries and the surface forcing.
- These output files serve as input for the main program.
- The main program writes a series of files to be used for an eventual restart (described in Section IV-4.1.1) and a series of output data files (described in Section IV-4.1.2) whose number depends on the specifications given by the user in *defmod.par* and optionally a series of output files created by the user.
- Both the preprocessor and the main program produce a “log”-file with run-time information.

A listing of all the file units connected to the preprocessor and the main program is given in respectively Table 4.4.1 and 4.4.2. Note that units 1–21 are reserved internally by the program and cannot be accessed by the user. The unit numbers starting from the value 22 are used by the main program for the output data files, and are available for the input of user data in the FORTRAN file *defmod.f* or for user output in subroutine PRINT. Unit 0 is reserved for error and warning messages which are further discussed in Section IV-5.3.

The value of a UNIT specifier in an OPEN statement and the number of files which can be connected simultaneously to a FORTRAN program are limited. These limits are machine-dependent. The second one is represented in the program by the parameter MAXFILES. Files are opened in the program by calling the OPENF routine. Before the file is opened, the routine increments the number of opened files by 1 and verifies that the unit number exists and the number of opened files does not exceed MAXFILES. The program stops with an error message if this is not the case. Each time a file connection is closed which is performed in the program by calling the routine CLOSEF, the number of opened files is decremented by 1. It is therefore recommended to make use of OPENF and CLOSEF to open or close user data files. The utility function IGETFUN is further provided in the program which returns the next accessible unit number (above the value of 21) available for file connection. Further details concerning OPENF, CLOSEF and IGETFUN are found in Section V-1.10.1 of the Reference Manual.

4.1.1 Input files for the main program

This includes the file boxes in Figure 4.4.1 between "PREPROCESSOR" and "COHERENS" and the four boxes with the "final conditions". The file names are composed of 11 characters:

- characters 1–6: TITLE of the simulation
- character 7: "." (dot)
- characters 8–10: the file description (further denoted by "file-desc") referring to the contents of the file. Values are given below.
- character 11: file format given by 'A' for ASCII or 'U' for unformatted binary

The "file-desc" is given by one of the following 3-character strings.

con

Contains the model switches and constants listed in Table 4.2.2 and 4.2.3, the TITLE of the run and the output specifiers as defined in *defmod.par*. The format is 'A'. The file is always written with header information permitting the user to check the settings of the parameters.

grd

The grid parameters and arrays listed in Table 4.2.4. Format is determined by the value of GRDFORM.

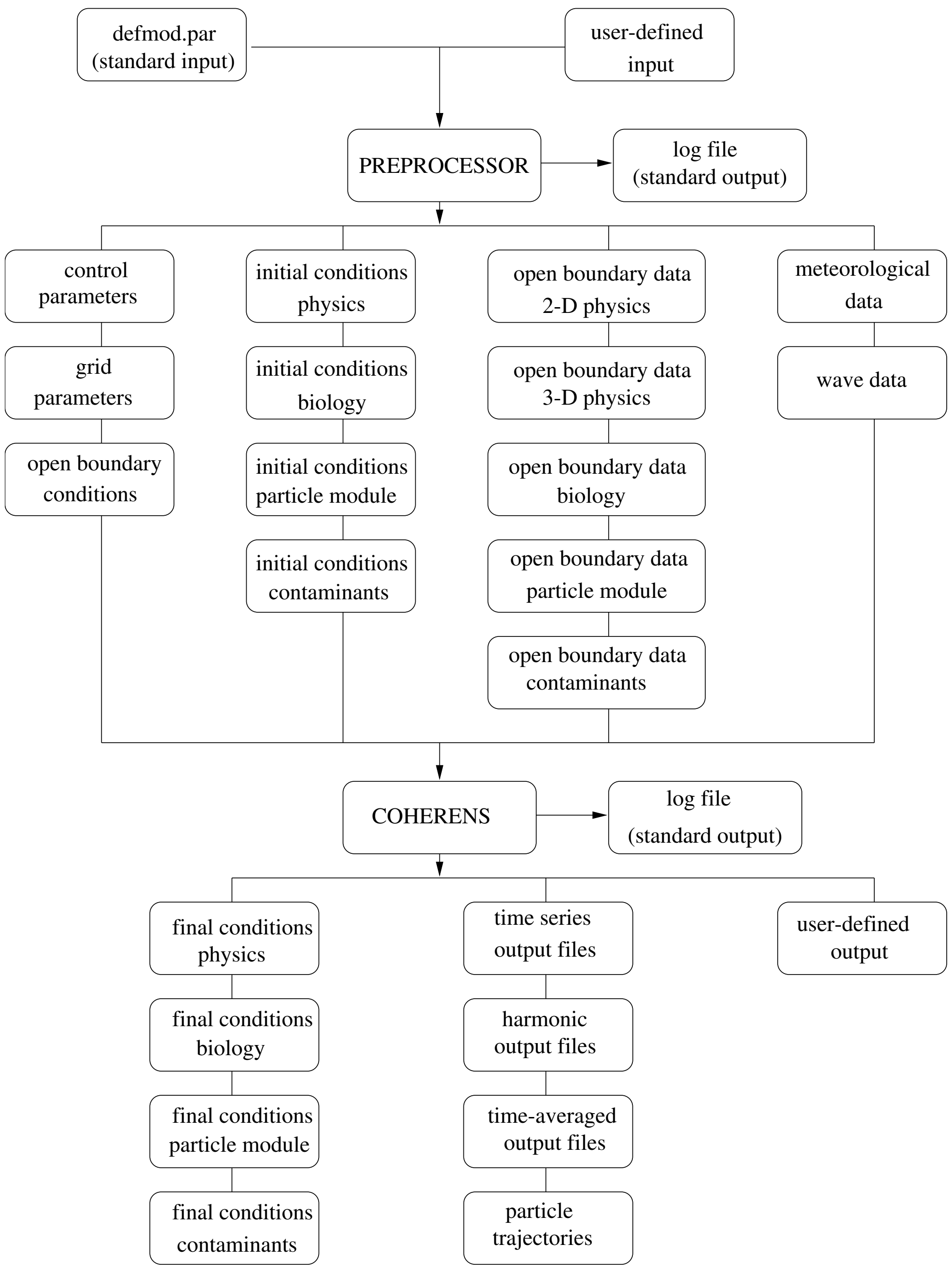


Figure 4.4.1: Schematic structure of all input and output files used in the program.

Table 4.4.1: File units connected to the preprocessor **preproc**.

unit	name	file-desc	format	I/O[a]	purpose
0	standard error	—	'A'	O	error messages
1	IOGRD	*grd*	'A' or 'U'	O	grid parameters
2	IOMET	*met*	'A' or 'U'	O	meteorological data
3	IOWAV	*wav*	'A' or 'U'	O	wave data
4	IOOBC	*obc*	'A'	O	open boundary conditions
5	standard input	*defmod.par*	'A'	I	TITLE and output specifiers
6	standard output	*log-file*	'A'	O	log-file
11	IOHOU	*hin*	'A' or 'U'	O	initial conditions (physics)
12	IOBOU	*bin*	'A' or 'U'	O	initial conditions (biology and sediments)
13	IOPOU	*pin*	'A' or 'U'	O	initial conditions (particle module)
14	IOCOU	*cin*	'A' or 'U'	O	initial conditions (contaminants)
15	IO2BC	*2bc*	'A' or 'U'	O	open boundary data (2-D physics)
16	IOHBC	*hbc*	'A' or 'U'	O	open boundary data (3-D physics)
17	IOBBC	*bbc*	'A' or 'U'	O	open boundary data (biology and sediments)
18	IOPBC	*pbc*	'A' or 'U'	O	open boundary data (particle module)
19	IOCBC	*cbc*	'A' or 'U'	O	open boundary data (contaminants)
20	—	*con*	'A'	O	switches and model parameters
>21	—	—	—	I	user-defined input

[a] I means input, O output.

Table 4.4.2: File units connected to the main program **coherens**.

unit	name	file-desc	format	I/O[a]	purpose
0	standard error	—	'A'	O	error messages
1	IOGRD	*grd*	'A' or 'U'	I	grid parameters
2	IOMET	*met*	'A' or 'U'	I	meteorological data
3	IOWAV	*wav*	'A' or 'U'	I	wave data
4	IOOBC	*obc*	'A'	I	open boundary conditions
5	IOCON	*con*	'A'	I	switches and model parameters
6	standard output	*log_file*	'A'	O	log file
7	IOHIN	*hin*	'A' or 'U'	I	initial conditions (physics)
8	IOBIN	*bin*	'A' or 'U'	I	initial conditions (biology and sediments)
9	IOPIN	*pin*	'A' or 'U'	I	initial conditions (particle module)
10	IOCIN	*cin*	'A' or 'U'	I	initial conditions (contaminants)
11	IOHOU	*hou*	'A' or 'U'	O	final conditions (physics)
12	IOBOU	*bou*	'A' or 'U'	O	final conditions (biology and sediments)
13	IOPOU	*pou*	'A' or 'U'	O	final conditions (particle module)
14	IOCOU	*cou*	'A' or 'U'	O	final conditions (contaminants)
15	IO2BC	*2bc*	'A' or 'U'	I	open boundary data (2-D physics)
16	IOHBC	*hbc*	'A' or 'U'	I	open boundary data (3-D physics)
17	IOBBC	*bbc*	'A' or 'U'	I	open boundary data (biology and sediments)
18	IOPBC	*pbc*	'A' or 'U'	I	open boundary data (particle module)
19	IOCBC	*cbc*	'A' or 'U'	I	open boundary data (contaminants)
20	reserved for future use				
21	—		'A'	O	information files
>21	—	*out*	'A' or 'U'	O	time series output
	—	*res*	'A' or 'U'	O	harmonic output (residuals)
	—	*amp*	'A' or 'U'	O	harmonic output (amplitudes)
	—	*pha*	'A' or 'U'	O	harmonic output (phases)
	—	*ell*	'A' or 'U'	O	harmonic output (tidal ellipses)
	—	*avr*	'A' or 'U'	O	time-averaged output
	—	*par*	'A' or 'U'	O	particle trajectories
	—	—	—	O	output defined in PRINT

[a] I means input, O output.

obc

The specifications of the open boundary conditions: the frequencies SIGMA and ANALSIG, the arrays ITYPOBU, ITYPOBV, IVPOBU and IVPOBV giving the type of conditions and the arrays LSTOBU, LSTOBV for the particle module. Format is 'A'.

hin

The arrays with the initial conditions for the physics as listed in Table 4.2.7 (currents, bottom stress, surface elevation, grid spacing, salinity, temperature, Riemann invariants, initial phases, turbulence arrays). Format is determined by the value of ICRFORM.

bin

The arrays with the initial conditions for the biology and the sediments as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

pin

The arrays with the initial conditions for Lagrangian particle transport as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

cin

The arrays with the initial conditions for contaminants as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

2bc

The open boundary arrays AMPOBU, PHAOBU, AMPOBV, PHAOBV for the 2-D mode. Format is determined by the value of BCSFORM.

hbc

The open boundary arrays for the physics as listed in Table 4.2.6. Format is determined by the value of BCSFORM.

bbc

The open boundary arrays for the biology and the sediments as listed in Table 4.2.6. Format is determined by the value of BCSFORM.

pbc

The open boundary arrays for the Lagrangian transport as listed in Table 4.2.6. Format is determined by the value of BCSFORM.

cbc

The open boundary arrays for the contaminants as listed in Table 4.2.6. Format is determined by the value of BCSFORM.

met

The meteorological data as supplied in subroutine WRFCDAT. Format is determined by the value of METFORM.

wav
: The wave data as supplied in subroutine WROBDAT. Format is determined by the value of WAVFORM.

hou
: The arrays with the final conditions for the physics as listed in Table 4.2.7 (currents, bottom stress, surface elevation, grid spacing, salinity, temperature, Riemann invariants, initial phases, turbulence arrays). Format is determined by the value of ICRFORM.

bou
: The arrays with the final conditions for the biology and the sediments as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

pou
: The arrays with the final conditions for Lagrangian particle transport as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

cou
: The arrays with the final conditions for contaminants as listed in Table 4.2.7. Format is determined by the value of ICRFORM.

All the files listed above are created by the program. However, the biological and sediment arrays are only written if either IOPTB or IOPTS equals 1, the particle arrays if IOPTP is non-zero, the contaminant arrays if IOPTC is non-zero, the meteorological data in the *met*-file if IOPTM is larger than 1 and the wave data in the *wav*-file if IOPTW is larger than 1.

4.1.2 Output data files

This includes the four boxes in Figure 4.4.1 with time series, harmonic, time-averaged and particle output. The file names are composed of 14 characters:

- characters 1–6: the parameter OUTTIT defined in subroutine DEFCON or the TITLE of the simulation if OUTTIT is not defined by the user
- character 7: "_" (underscore)
- character 8: the output file number which varies between 1 and NARROUT for time series output, 1 and NCONTO for harmonic output, 1 and NARRAVR for time-averaged output and equals 1 for particle output.
- character 9: "." (dot)
- characters 10–12: the "files-desc" which refers to the type of output. Possible values are given below.

- character 13: this has one of the following values

  ‘I’ : Information file in ‘A’-format describing the characteristics of the output data.

  ‘2’ : The file contains 2-D output data as defined in subroutines OUTPT2D, ANALPT2D, AVRPT2D.

  ‘3’ : The file contains 3-D output data as defined in subroutines OUTPT3D, ANALPT3D, AVRPT3D or particle output as defined in subroutine PARPT3D.

- character 14: file format given by ‘A’ for ASCII or ‘U’ for unformatted binary

The “file-desc” is given by one of the following 3-character strings.

out

Indicates time series output. The format of the data files is determined by the corresponding value of ARRFORM(IARR) where IARR is the file number.

res

Indicates output of residual values derived from harmonic analysis. The format of the data files is determined by the value of ANALFORM. Note that the file number always equals 1 in this case.

amp

Indicates output of amplitude values derived from harmonic analysis. The format of the data files is determined by the value of ANALFORM. The file number N indicates that the file contains the data for harmonic frequency ANALSIG(N).

pha

Indicates output of phase values in radians (with respect to the central time t_c) derived from harmonic analysis. The format of the data files is determined by the value of ANALFORM. The file number N indicates that the file contains the data for the harmonic frequency ANALSIG(N).

ell

Indicates output of tidal ellipse parameters derived form harmonic analysis. The format of the data files is determined by the value of ANALFORM. The file number N indicates that the file contains the data for harmonic frequency ANALSIG(N).

avr

Indicates time-averaged output. The format of the data files is determined by the corresponding value of AVRFORM(IARR) where IARR is the file number.

par

Indicates output of particle trajectories. The format of the data file is determined by the value of PARFORM.

4.2 Reading model data

4.2.1 I/O routines

Output data are mostly written in the program in a specific COHERENS-format with the aid of subroutines WRARR and WRARI. WRARR writes an array of data type REAL with a maximum of 3 dimensions using either "A" or "U" format and with or without header information. WRARI has the same purpose now for INTEGER arrays. Input arrays, written by WRARR and WRARI, are read with the aid of subroutines RDARR and RDARI. The four I/O routines are described in Section V-1.10.1.

For example, the *hin*-file with the initial conditions for the physics is written by the preprocessor by calling WRARR for each of the arrays listed in the first part of Table 4.2.7 (UD2, ..., DISSW). It is important to note that the meteorological and/or wave data and the open boundary data arrays must be stored into the corresponding files using the procedure shown by the generic routines WRFCDAT and WROBDAT in the **/examples** subdirectory. The only exceptions are the *con*- and *obc*-files which are written with extra header information allowing to check the settings of user-defined parameters. Note also that the *grd*-file starts with extra header lines.

Time series, harmonic, time-averaged and particle output are performed in respectively subroutines OUTPT, ANALYS2, INTEGR2 and OUTPAR further described in Section V-1.10.3 of the Reference Manual. The routines RDOUT, RDANAL, RDELL, RDAVR and RDPAR listed in the file *readdat.f* in **/examples**, show how the output data can be read and stored into temporary arrays which may be used for plotting the data. The meaning of the arguments is explained in Section V-1.10.5 of the the Reference Manual. It is important to note that these routines are only "examples" showing how the data are stored in the output files. Saving all data in temporary arrays may consume a substantial amount of computer memory whereas in practice a limited amount of data are needed for plotting. The routines are only provided as a guideline for the user how to read model output. The actual postprocessor program has to be written by the user depending on the type of graphical software. An alternative way for reading user-defined output is to make use of the associated "Information" file, described in the next subsection. Some example code is given in subsection IV-4.2.3 below.

A complete list of all routines, related to model input or output, is given in Table 4.4.3.

4.2.2 Information files

a) time series, harmonic and time-averaged output

The 2-D and 3-D output data files with user-defined output have an associated Information file. The general form of this file in the case of time series, harmonic and time-averaged output is as follows[1]

[1]Note that the line number before the ":" and the ":" do not appear in the actual file.

Table 4.4.3: Modules related to model I/O.

routine	file	purpose
Reading/writing arrays in COHERENS-format		
RDARI	*Inout.f*	read an INTEGER array in COHERENS-format
RDARR	*Inout.f*	read a REAL array in COHERENS-format
WRARI	*Inout.f*	write an INTEGER array in COHERENS-format
WRARR	*Inout.f*	write a REAL array in COHERENS-format
Reading model input		
BBCIN	*bcsin.f*	read the *bbc*-file with open boundary input for the biological and sediment modules
BCSIN	*bcsin.f*	read the *obc*-file with the types of open boundary conditions
BC2IN	*bcsin.f*	read the *2bc*-file with open boundary input for the 2-D mode
CBCIN	*bcsin.f*	read the *cbc*-file with open boundary input for the contaminant module
HBCIN	*bcsin.f*	read the *hbc*-file with open boundary input for the physics
INPA	*outpa.f*	read the file(s) with the initial conditions (*hin*, *bin*, *pin*, *cin*)
METIN	*metin.f*	read the *met*-file with the meteorological data
PBCIN	*bcsin.f*	read the *pbc*-file with open boundary input for the particle module
RDCON	*inicon.f*	read the *con*-file with model switches and parameters
RDGRD	*Inout.f*	read the *grd*-file with the grid arrays
READIN	*preproc.f*	read the parameter file (*defmod.par* by default) with output specifiers
WAVIN	*wavin.f*	read the *wav*-file with the surface wave data
Writing model input		
BCSOUT	*preproc.f*	write the *obc*-file with the types of open boundary conditions and all time-independent open boundary data (*2bc*, *hbc*, *bbc*, *pbc*, *cbc*-files)
OUTPA	*outpa.f*	write the file(s) with the initial conditions (*hin*, *bin*, *pin*, *cin*) or the file(s) with the final conditions (*hou*, *bou*, *pou*, *cou*)
WRCON	*preproc.f*	write the *con*-file with model switches and parameters
WRFCDAT	*defmod.f*	write the *met*-file with the meteorological data and the *wav*-file with the surface wave data
WRGRD	*Inout.f*	write the *grd*-file with the grid arrays
WROBDAT	*defmod.f*	write the time-dependent open boundary input (*2bc*-, *hbc*-, *bbc*-, *pbc*-, *cbc*-files)

Table 4.4.3: continued.

routine	file	purpose
Writing user-defined output		
ANALYS2	*analys.f*	write the data files (*res*, *amp*, *pha*, *ell*) with harmonic output
INTEGR2	*integr.f*	write the data file (*avr*) with time-averaged output
OUTPAR	*sedlag.f*	write the data file (*par*) with the particle trajectories (including the associated Information file)
OUTPT	*outpt.f*	write the data files (*out*) with time series output
PRINT	*print.f*	write (non-standard) data output
WROUT	*Inout.f*	write the Information file (except for particle output)
Reading user-defined output (example routines)		
RDANAL	*readdat.f*	read harmonic output (*res*, *amp*, *pha*-files)
RDAVR	*readdat.f*	read time-averaged output (*avr*-files)
RDELL	*readdat.f*	read tidal ellipse output (*ell*-files)
RDOUT	*readdat.f*	read time series output (*out*-files)
RDPAR	*readdat.f*	read particle output (*par*-file)

```
1  : Header line
2  : OUTTIT
3  : OUTFORM
4  : HEADERS
5  : Header line
6  : GRDFIL
7  : Header line
8  : GRDFORM
9  : NC NR NZ NSTEP
10 : IGTRH
11 : LIMRES(1,1) LIMRES(2,1) LIMRES(3,1)
12 : LIMRES(1,2) LIMRES(2,2) LIMRES(3,2)
13 : LIMRES(1,3) LIMRES(2,3) LIMRES(3,3)
14 : LIMRES(1,4) LIMRES(2,4) LIMRES(3,4)
15 : N2VEC
15a: NAME2D DESC2D UNIT2D
16 : N2SCAL
16a: NAME2D DESC2D UNIT2D
17 : N3VEC
17a: NAME3D DESC3D UNIT3D
18 : N3SCAL
18a: NAME3D DESC3D UNIT3D
19 : Header line
20 : OUTFIL2D
21 : Header line
22 : OUTFIL3D
23 : DELX
24 : DELY
25 : DELT
```

- Lines 15a are repeated 2*N2VEC times and are absent if N2VEC = 0.
- Lines 16a are repeated N2SCAL times and are absent if N2SCAL = 0.
- Lines 17a are repeated 3*N3VEC times and are absent if N3VEC = 0.
- Lines 18a are repeated N3SCAL times and are absent if N3SCAL = 0.
- Lines 19 and 20 are absent if N2VEC= N2SCAL = 0.
- Lines 21 and 22 are absent if N3VEC= N3SCAL = 0.

The comment lines are read by skipping one line. The parameters are read using the following formats:

- OUTTIT: '(12X,A6)'
- OUTFORM: '(22X,A1)'
- HEADERS: '(21X,L1)'
- GRDFIL, OUTFIL2D, OUTFIL3D: '(A)'
- GRDFORM: '(A1)'
- NC NR NZ NSTEP: '(16X,3(1X,I6),1X,I10)'
- IGTRH: '(12X,I1)'
- LIMRES: '(3(7X,3(1X,I6),/),7X,3(1X,I10))'
- N2VEC, N2SCAL, N3VEC, N3SCAL: '(23X,I2)'
- NAME2D DESC2D UNIT2D: '(A8,1X,A30,1X,A16)'
- NAME3D DESC3D UNIT3D: '(A8,1X,A30,1X,A16)'
- DELX, DELY, DELT: '(7X,1PE14.7)'

The parameters have the following meaning.

OUTTIT
: First 6 characters of the output data file.

OUTFORM
: Format of the data files ('A' or 'U').

HEADERS
: If OUTFORM equals 'A' and HEADERS is .TRUE., the output files contain extra header information.

GRDFIL
: Name of the *grd*-file (including the full pathname).

GRDFORM
: Format of the *grd*-file ('A' or 'U').

NC, NR, NZ
: Number of grid cells in the X-, Y- and Z-direction.

NSTEP
: Total number of (2-D) time steps in the simulation.

IGTRH
: Type of the grid: 0 means Cartesian, 1 spherical.

LIMRES(3,4)

Determines the grid locations and the time instants of the data. If (K, J, I) or (J, I) are the spatial indices of a 3-D or 2-D data array and NT the time counter representing the number of 2-D time steps elapsed since the start of the simulation, the output data are obtained at the following grid locations and time levels (where the first number after the equal sign denotes the start value, the second the end value and the third the step value):

```
I = LIMRES(1,1),LIMRES(2,1),LIMRES(3,1)
J = LIMRES(1,2),LIMRES(2,2),LIMRES(3,2)
K = LIMRES(1,3),LIMRES(2,3),LIMRES(3,3)
NT= LIMRES(1,4),LIMRES(2,4),LIMRES(3,4)
```

N2VEC

Number of 2-D output vector fields.

N2SCAL

Number of 2-D output scalar fields.

N3VEC

Number of 3-D output vector fields.

N3SCAL

Number of 3-D output scalar fields.

NAME2D, DESC2D, UNIT2D

Names (8 characters), descriptions (30 characters) and units (16 characters) of the 2-D output fields (including both the X- and Y-component of 2-D vectors).

NAME3D, DESC3D, UNIT3D

Names (8 characters), descriptions (30 characters) and units (16 characters) of the 3-D output fields (including both the X-, Y- and Z-component of 3-D vectors).

OUTFIL2D

Name of the 2-D output data file (including the full pathname).

OUTFIL3D

Name of the 3-D output data file (including the full pathname).

DELX, DELY

(Uniform) grid spacings in the X- and Y-direction as given by GX0(2)-GX0(1) and GY0(2)-GY0(1). In the case of a non-uniform grid, the grid coordinates at the cell corners (arrays GX0 and GY0) are obtained by reading in the *grd*-file. Note that, by default, a uniform grid is selected in the vertical. When a non-uniform vertical grid is taken, the array GZ0 has to be read from the *grd*-file.

DELT
: Time step (s) for the 2-D mode. This parameter is needed to convert the time level (NT) into an actual time (NT*DELT) in seconds.

b) particle output

The general form of the Information file for particle output is as follows[2]

```
1  : Header line
2  : OUTTIT
3  : OUTFORM
4  : HEADERS
5  : Header line
6  : GRDFIL
7  : Header line
8  : GRDFORM
9  : NC NR NZ NSTEP
10 : IGTRH
11 : LIMRES(1) LIMRES(2) LIMRES(3)
12 : NOPOUT
13 : Header line
14 : OUTFIL
15 : DELT
```

The header lines are read by skipping one line. The parameters are read using the following format:

- OUTTIT: '(12X,A6)'
- OUTFORM: '(22X,A1)'
- HEADERS: '(21X,L1)'
- GRDFIL, OUTFIL: '(A)'
- GRDFORM: '(A1)'
- NC NR NZ NSTEP: '(16X,3(1X,I6),1X,I10)'
- IGTRH: '(12X,I1)'
- LIMRES: '(7X,3(1X,I10))'
- NOPOUT: '(29X,I6)'
- DELT: '(7X,1PE14.7)'

[2]The line number before the ":" and the ":" do not appear in the actual file.

The parameters have the following meaning.

OUTTIT
: First 6 characters of the output data file.

OUTFORM
: Format of the data files ('A' or 'U').

HEADERS
: If OUTFORM equals 'A' and HEADERS is .TRUE., the output files contain extra header information.

GRDFIL
: Name of the *grd*-file (including the full pathname).

GRDFORM
: Format of the *grd*-file ('A' or 'U').

NC, NR, NZ
: Number of grid cells in the X-, Y- and Z-direction.

NSTEP
: Total number of (2-D) time steps in the simulation.

IGTRH
: Type of the grid: 0 means Cartesian, 1 spherical.

LIMRES(3)
: Determines the output times of the particle positions. Output data are then given at the following time levels (where the first number after the equal sign denotes the start value, the second the end value and the third the step value):

```
NT = LIMRES(1),LIMRES(2),LIMRES(3)
```

NOPOUT
: Number of particles in the data files.

DELT
: Time step (s) for the 2-D mode. This parameter is needed to convert the time level (NT) into an actual time (NT*DELT) in seconds.

4.2.3 Reading user-defined output

The parameters contained in the Information file allow to read the associated data files. Three types of output can be distinguished: 3-D output, 2-D output and particle output. An example code is given below for each case. A more formal treatment can be found in the routines listed in the *readdat.f* (**/examples** subdirectory).

a) reading 3-D output data

The example code below reads all model output and stores the values into the array DATA3D.

```
C     ---open data file
      IUNIT = 22
      IF (OUTFORM.EQ.'A') THEN
         OPEN (IUNIT,FILE=OUTFIL3D,STATUS='OLD')
      ELSEIF (OUTFORM.EQ.'U') THEN
         OPEN (IUNIT,FILE=OUTFIL3D,FORM='UNFORMATTED',STATUS='OLD')
      ENDIF

C     ---dimensions of the output data array
      NZMAX = (LIMRES(2,3)-LIMRES(1,3))/LIMRES(3,3) + 1
      NRMAX = (LIMRES(2,2)-LIMRES(1,2))/LIMRES(3,2) + 1
      NCMAX = (LIMRES(2,1)-LIMRES(1,1))/LIMRES(3,1) + 1
      NTMAX = (LIMRES(2,4)-LIMRES(1,4))/LIMRES(3,4) + 1
      N3RES = 3*N3VEC + N3SCAL

C     ---read data
      DO 100 NT=LIMRES(1,4),LIMRES(2,4),LIMRES(3,4)
      DO 100 I3VAR=1,N3RES
         NTSUB = (NT-LIMRES(1,4))/LIMRES(3,4) + 1
C        --'A'-format
         IF (OUTFORM.EQ.'A') THEN
            READ (IUNIT,*)
C           --with headers
            IF (HEADERS) THEN
               DO 110 I=1,NCMAX
                  READ (IUNIT,*)
                  DO 111 J=1,NRMAX
                     READ (IUNIT,*)
                     READ (IUNIT,'(100(1PE14.7,1X))')
     1                     (DATA3D(K,J,I,NTSUB,I3VAR),K=1,NZMAX)
 111              CONTINUE
 110           CONTINUE
C           --without headers
            ELSE
               READ (IUNIT,'(100(1PE14.7,1X))')
     1              (((DATA3D(K,J,I,NTSUB,I3VAR),K=1,NZMAX),
     2                J=1,NRMAX),I=1,NCMAX)
            ENDIF
C        --'U'-format
```

```
         ELSEIF(OUTFORM.EQ.'U') THEN
            READ (IUNIT)
     1           (((DATA3D(K,J,I,NTSUB,I3VAR),K=1,NZMAX),
     2              J=1,NRMAX),I=1,NCMAX)
         ENDIF
 100  CONTINUE
```

It is clear that the dimensions of the 5-D array DATA3D must be larger than or equal to NZMAX, NRMAX, NCMAX, NTMAX, N3RES. The type of field variable, the grid location and the output time, corresponding to the array element DATA3D(K,J,I,NT,I3VAR) are obtained as follows:

- I3VAR denotes the number of the field variable as defined in OUTPT3D (time series), ANALPT3D (harmonic output) or AVRPT3D (time-averaged output). When the data file contain values of the (3-D) tidal ellipse parameters (*ell*-file), I3VAR represents one of the following fields:

  1 : major axis A (m/s)

  2 : minor axis B (m/s)

  3 : ellipticity ε

  4 : inclination Θ (radians)

  5 : elliptic phase Φ (radians)

  6 : magnitude of the cyclonic current $|S_+|$ (m/s)

  7 : magnitude of the anticyclonic current $|S_-|$ (m/s)

- The data value is spatially located at the centre of the cell on the model grid with indices (KK,JJ,II) determined by

```
KK = LIMRES(1,3) + (K-1)*LIMRES(3,3)
JJ = LIMRES(1,2) + (J-1)*LIMRES(3,2)
II = LIMRES(1,1) + (I-1)*LIMRES(3,1)
```

- The corresponding time of output (in seconds since the start of the simulation) is given by

```
DELT*(LIMRES(1,4)+(NT-1)*LIMRES(3,4))
```

b) reading 2-D output data

The procedure is similar to the previous one but given here for clarity. The example code below reads all model output and stores the values into the array DATA2D.

```
C     ---open data file
      IUNIT = 23
      IF (OUTFORM.EQ.'A') THEN
         OPEN (IUNIT,FILE=OUTFIL2D,STATUS='OLD')
      ELSEIF (OUTFORM.EQ.'U') THEN
         OPEN (IUNIT,FILE=OUTFIL2D,FORM='UNFORMATTED',STATUS='OLD')
      ENDIF

C     ---dimensions of the output data array
      NRMAX = (LIMRES(2,2)-LIMRES(1,2))/LIMRES(3,2) + 1
      NCMAX = (LIMRES(2,1)-LIMRES(1,1))/LIMRES(3,1) + 1
      NTMAX = (LIMRES(2,4)-LIMRES(1,4))/LIMRES(3,4) + 1
      N2RES = 2*N2VEC + N2SCAL

C     ---read data
      DO 100 NT=LIMRES(1,4),LIMRES(2,4),LIMRES(3,4)
      DO 100 I2VAR=1,N2RES
         NTSUB = (NT-LIMRES(1,4))/LIMRES(3,4) + 1
C        --'A-'format
         IF (OUTFORM.EQ.'A') THEN
            READ (IUNIT,*)
C           --with headers
            IF (HEADERS) THEN
               READ (IUNIT,*)
               DO 110 I=1,NCMAX
                  READ (IUNIT,*)
                  DO 111 J=1,NRMAX
                     READ (IUNIT,*)
                     READ (IUNIT,'(1PE14.7,1X)') DATA2D(J,I,NTSUB,I2VAR)
 111              CONTINUE
 110           CONTINUE
C           --without headers
            ELSE
               READ (IUNIT,'(100(1PE14.7,1X))')
     1              ((DATA2D(J,I,NTSUB,I2VAR),J=1,NRMAX),I=1,NCMAX)
            ENDIF
C        --'U'-format
         ELSEIF(OUTFORM.EQ.'U') THEN
            READ (IUNIT)
     1           ((DATA2D(J,I,NTSUB,I2VAR),J=1,NRMAX),I=1,NCMAX)
         ENDIF
 100  CONTINUE
```

It is clear that the dimensions of the 4-D array DATA2D must be larger than or equal to NRMAX, NCMAX, NTMAX, N2RES. The type of field variable, the grid location and the output time, corresponding to the array element DATA2D(J,I,NT,I2VAR) are obtained as follows:

- I2VAR denotes the number of the field variable as defined in OUTPT2D (time series), ANALPT2D (harmonic output) or AVRPT2D (time-averaged output). When the data file contain values of the (2-D) tidal ellipse parameters (*ell*-file), I2VAR represents one of the following fields:

  1 : major axis A (m/s)

  2 : minor axis B (m/s)

  3 : ellipticity ε

  4 : inclination Θ (radians)

  5 : elliptic phase Φ (radians)

  6 : magnitude of the cyclonic current $|S_+|$ (m/s)

  7 : magnitude of the anticyclonic current $|S_-|$ (m/s)

- The data value is spatially located at the centre of the (2-D) cell on the model grid with indices (JJ,II) determined by

```
JJ = LIMRES(1,2) + (J-1)*LIMRES(3,2)
II = LIMRES(1,1) + (I-1)*LIMRES(3,1)
```

- The corresponding time of output (in seconds since the start of the simulation) is given by

```
DELT*(LIMRES(1,4)+(NT-1)*LIMRES(3,4))
```

c) reading particle output

The code below reads all model output and stores the particle positions into the array PARDAT.

```
C     ---open data file
      IUNIT = 24
      IF (OUTFORM.EQ.'A') THEN
         OPEN (IUNIT,FILE=OUTFIL,STATUS='OLD')
      ELSEIF (OUTFORM.EQ.'U') THEN
         OPEN (IUNIT,FILE=OUTFIL,FORM='UNFORMATTED',STATUS='OLD')
      ENDIF
```

```
C     ---number of time series
      NTMAX = (LIMRES(2)-LIMRES(1))/LIMRES(3) + 1

C     ---read data
      DO 100 NT=LIMRES(1),LIMRES(2),LIMRES(3)
         NTSUB = (NT-LIMRES(1))/LIMRES(3) + 1
C        ---'A'-format
         IF (OUTFORM.EQ.'A') THEN
            IF (HEADERS) READ (IUNIT,*)
            DO 110 IVAR=1,NOPOUT
               READ (IUNIT,'(3(1PE14.7,1X))')
     1              (PARDAT(L,IVAR,NTSUB),L=1,3)
 110        CONTINUE
C        ---'U'-format
         ELSEIF (OUTFORM.EQ.'U') THEN
            READ (IUNIT) (PARDAT(L,IVAR,NTSUB),L=1,3),IVAR=1,NOPOUT)
         ENDIF
 100  CONTINUE
```

The dimensions of the 3-D array PARDAT must be larger than or equal to 3, NOPOUT, NTMAX. The particle number, its geographical position and the output time, associated with the array element PARDAT(L,IVAR,NT) are obtained as follows:

- IVAR denotes the number of the particle as defined in PARPT3D.
- The array element represents the X-, Y- or Z-coordinate of the particle with number IVAR if L equals respectively 1, 2, or 3. Note that (X,Y) are either given in meters (Cartesian case) or as longitude and latitude in decimal degrees (spherical case).
- The corresponding time of output (in seconds since the start of the simulation) is given by

```
DELT*(LIMRES(1,4)+(NT-1)*LIMRES(3,4))
```

Chapter 5

Additional Issues

The following topics are further discussed:

- the startup files **csetup** and **vsetup**
- compiling the program
- error and warning messsages
- restarting the program
- changing the source code

5.1 The startup shell scripts

There are two shell scripts **csetup** and **vsetup** (located in the **UTIL** directory) available which enable to execute one of the test cases or a user-defined application from a "work" directory on the user's system. They are executed with one argument. The script **csetup** performs a series of actions in the following order:

- links the "project" directory $**COHERENS/tests/test_name** to **PROJECT** where **test_name** is the value of the argument
- links the "source" directory $**COHERENS/source** to **SOURCE**
- links the "utilities" directory $**COHERENS/utilities** to **UTIL**
- copies all the files of the **SOURCE** directory to the current ("work") directory
- copies all the files of the **PROJECT** directory to the "work" directory
- copies all the files of the **UTIL** directory to the "work" directory

The alternative startup script **vsetup** performs the actions:

- creates the links to **PROJECT**, **SOURCE** and **UTIL** as in the previous case
- copies the files of the **PROJECT** directory to the current ("work") directory
- copies the *Makefile* from the **SOURCE** to the current directory

The script **csetup** has the advantage that it is portable on all UNIX machines. The second one avoids copying of all the source code but can only be used if the macro VPATH is available in the UNIX **make** utility.

5.2 Compilation of the source code

The preprocessor and main programs are compiled with the **make** command available on UNIX systems. By executing **make** the compilation proceeds following the instructions given in the file *Makefile*:

1. All FORTRAN source files with the suffix **.f* (except *preproc.f* and *defmod.f*) are compiled separately without linking. A series of object files with a suffix **.o* is created.
2. The object files are linked to produce the executable program **coherens**.
3. The FORTRAN files *preproc.f* and *defmod.f* are compiled without linking.
4. The object files are linked to produce the executable program **preproc**.

If different source files (suffix *.f* or *.inc*) with the same name reside both in the **PROJECT** and in the **SOURCE** directory, the compilation is performed with the **PROJECT** file. This has the advantage that the changes in the source code can more easily be performed (see Section IV-5.5). The following comments are to be made:

- If a program application is installed using the startup file **vsetup**, the following lines must be present in *Makefile*:

  ```
  VPATH = PROJECT:SOURCE
  IFLAGS = -I. -IPROJECT -ISOURCE
  ```

- The default version of *Makefile* assumes that FORTRAN programs are compiled with **f77** command. If this command has a different name, e.g. **fcomp**, on the user's machine, the line `FC = f77` must be replaced by `FC = fcomp`.
- The default names of the preprocessor and the main program are **preproc** and **coherens**. New names can be given by changing the values of the variables PRECOM and MAINCOM in *Makefile*.

- Most FORTRAN compilers provide extra compiler options for debugging to create an optimised code, for parallel execution of the program, ... These options are mostly machine-dependent and can be included using the variable FOPT. For example, to run the program with the optimisation option **-O3**, one has to set `FOPT = -O3`.

- Some machines allow to use inlining of functions and subroutines. This means that the compiler substitutes the source code of the inlined subprogram at the source line where the CALL statement is made, so that additional optimisations can be performed. If this option is available, the following source files ending with a *_in.f* are recommended for inlining: *Intp_in.f*, *Upwnd_in.f*, *biolgy_in.f*, *veddy_in.f*.

- Specific instructions for compiling the program on PCs using LINUX or Microsoft Fortran are given in Section II-1.2.

The graphical interface program **netcdfint** is compiled following the instructions given in the file *MakePost*. For details see Section I-4.3.2.

5.3 Error and warning messages

Two types of messages may be written by the program to standard error output ("stderr"). The first are warning messages and do not stop execution of the program. Such messages usually mean that the user has assigned a "strange" value for some model parameter, which may (but not always) produce erroneous results. If for example the 1-D test case **pycno** is run, the following warning message is issued:

```
WARNING : 2-D current calculations disabled
```

due to the fact that the switch IOPT2 is set to 0. This choice is allowed for this particular application but not in general.

The program does stop execution when one or more error messages appear on the screen or in the file where "stderr" is redirected. Each error statement consists of a full line followed by one or more intended lines which try to explain the source of the error. The routine where the error occurred, is shown on the third last line. The second last line displays the type of the error which caused the termination of the program. The program traps 6 types of errors:

Type 1 : `Not possible to OPEN/CLOSE file`

Type 2 : `Read Error, End of File or file does not exist`

Type 3 : `Wrong input value(s)`

Type 4 : `Write error`

Type 5 : `Invalid initial values for model parameters or variables`

Type 6 : `Invalid value(s) for variable(s) at run time`

The last line always displays the message:

```
PROGRAM TERMINATED ABNORMALLY
```

To verify the validity of model input parameters the preprocessor and the main program call the following subroutines:

ERRMOD : checks whether model parameters are within the allowed range (e.g. switches) or have the correct format (e.g. Date/Time parameters)

ERRGRD : checks whether the grid parameters and arrays have valid values and are mutually compatible

ERRBCS : checks the validity of the parameter arrays related to the location and the type of the open boundary conditions

ERRICS : checks the validity of the initial arrays supplied for the Langrangian particle module

If non-allowed or unusual values have been assigned by the user, the error statements or warning messages may provide some help to locate the source of the problem and eventually to provide a solution.

5.4 Restarting the program

As discussed in Section IV-4.1.1 the program writes a series of files at the end of a simulation which can be used for a restart, which allows to split up a simulation into several runs. This procedure is recommended for long runs to avoid that CPU time is lost due to a computer crash or even a power cutoff, but is also useful to spin-up the program for a particular application. A suggested procedure is the following.

1. Prepare and run the program for an initial simulation.

   - Initialise the program using the setup file (say) *defmod_0.f*:

     ```
     cp defmod_0.f defmod.f
     make
     preproc < defmod.par > prelog
     ```

   - Run the main program for the first time time (assuming that mytest is the TITLE of the run

     ```
     coherens < mytest.conA > runlog
     ```

2. Create a new setup file (say) *defmod_1.f* for the next simulation

- The start date/year is given by the end date/year of the previous run, i.e. IBDATE and IBYEAR in *defmod_1.f* must be equal to IEDATE and IEYEAR in *defmod_0.f*.
- Define new initial conditions in DEFICS by reading the files with the final conditions from the previous run:
  - Open the *hou*-file for input:
    ```
    CALL OPENF(IOHIN,TITLE(1:6)//'.hou'//ICRFORM,'IN',ICRFORM)
    ```
  - Open the *bou*-file for input (if necessary):
    ```
    CALL OPENF(IOBIN,TITLE(1:6)//'.bou'//ICRFORM,'IN',ICRFORM)
    ```
  - Open the *pou*-file for input (if necessary):
    ```
    CALL OPENF(IOPIN,TITLE(1:6)//'.pou'//ICRFORM,'IN',ICRFORM)
    ```
  - Open the *cou*-file for input (if necessary):
    ```
    CALL OPENF(IOCIN,TITLE(1:6)//'.cou'//ICRFORM,'IN',ICRFORM)
    ```
  - Read the initial conditions:
    ```
    CALL INPA
    ```
  - Redefine new initial conditions if necessary (e.g. new initial temperature field).
- The user is free to change other settings of the program (switches or model parameters) or to supply new open boundary input and meteorological data for the new run.

3. Prepare and run the program for the second simulation

```
cp defmod_1.f defmod.f
make
preproc < defmod.par > prelog
coherens < mytest.conA > runlog
```

Steps 2 and 3 can be repeated as many times as needed.

Examples of the previous procedure are the test cases **river** and **plume**. During an initial spin-up run (using the file *defmod_0.f*) the current field is initialised for four tidal cycles without updating the salinity field (IOPTSA = 0) and without discharge of fresh water in the **plume** case. In the setup file *defmod_1.f* for the final run the salinity module is switched on (IOPTSA = 1) and the final data from the previous run are used as initial conditions following the procedure described above. In the **river** test case a new initial salinity field (as given by (2.4.1) and shown in Figure 2.4.1a) is defined while in the **plume** test the release of fresh water is enabled at the river mouth.

A similar procedure is followed in the North Sea case study discussed in Chapter II-7. The program is first initialised for three days without temperature, salinity, biology and sediments. The final run is split up into 12 simulations each covering a period of

30 days. The temperature and salinity fields, the biological and sediment concentrations are initialised in the setup file of the first run. The other runs are set up using the final condition files from the previous run with the aid of the procedure described above.

5.5 Changing the source code

If the user prefers to change the code in one of the the FORTRAN source files, the following procedure has to be followed:

- copy the appropriate source file form the **SOURCE** to the **PROJECT** directory
- edit the file in the **PROJECT** directory
- (re)compile the program

As mentioned in Section IV-5.2 the program automatically compiles with the edited file. Note that if the program application has already been installed with **csetup** or **vsetup**, the startup file must be executed again.

It is clear that changing the source code is a potentially dangerous procedure since this may produce unexpected side effects on the other parts of the program. A few cases where alterations of the program may be useful are listed below.

5.5.1 Default values

The default values for most model input parameters, either in scalar or in array form, are assigned in subroutine DEFAULT located in the file *preproc.f.* This routine is called by the preprocessor before DEFCON. A new default value can easily be substituted by changing the appropriate assignment statement.

5.5.2 Model parameters not defined in DEFCON

The turbulence module contains a large number of parameters. Some of them are tuning parameters, listed in Table 4.2.3 which can be redefined by the user in subroutine DEFCON. Other parameters, all related to the turbulence closure schemes discussed in Section III-1.2.2, have been calibrated from experimental data or are derived from theoretical considerations. They are defined in subroutine INICON. If a user, experienced in turbulence modelling, prefers to redefine one of these parameters (which is not recommended), the new value has to be substituted by editing the code in INICON, located in the file *inicon.f.* A list of the turbulence parameters, defined in INICON is given in Table 5.1.2.

5.5.3 Program switches

Although this is not the case in general, the range of values for some program switches can be extended without unexpected consequences to the program. If for example the user prefers to add a new relation for the surface drag coefficient selected by the additional value 5 for the switch IDRAG, the following procedure should be followed:

- edit the source code in function CD, located in the file *metin.f*, by inserting the lines

  ```
  ELSEIF (IDRAG.EQ.5) THEN
    .... (new formulation)
  ```

- to avoid that an error trap occurs, replace the following lines in subroutine ERRMOD in *Errors.f*

  ```
  IF (IDRAG.LT.0.OR.IDRAG.GT.4) THEN
     ...
     WRITE (0,8102) '0 1 2 3 4'
  ENDIF
  ```

  by

  ```
  IF (IDRAG.LT.0.OR.IDRAG.GT.5) THEN
     ...
     WRITE (0,8102) '0 1 2 3 4 5'
  ENDIF
  ```

Part V

Reference Manual

Chapter 1

Module Descriptions

1.1 Introduction

This chapter describes each program unit (main programs, subroutines and functions). The aim is to provide some guidance for the experienced user concerning the structure of each program unit and to show the links between the different units. The latter is illustrated with the aid of a few flow charts and indicated by mentioning the calling program and externals in each module description.

This chapter is organised into the following sections and subsections each describing a series of modules in alphabetical order related to a specific aspect of the COHERENS program.

- Main program units (Section V-1.2):
  MAINPROG, NETCDFINT, PREPROC

- Physical modules (Section V-1.3):
  CONTNY, CRRNT1C, CRRNT2, CRRNT3C, CRRNT3P, DENSTY, DENSTY1, HEAT, HEDDY, IRRAD, SALT, SEARHO, SOLRAD, UCALC, UDCALC, VCALC, VDCALC, WCALC, WPCALC

- Turbulence modules (Section V-1.4):
  BUOFR2, DISLEN, DISMIN, DISSIP, FNSHB, FNSMU, FNTURF, FVEDMA, GVEDMA, PRODUC, RICH, SHFR2, TKLENG, TLENDIS, TLENG, TURBEN, VEDDY1, VEDDY2, ZLMAX

- Biological and sediment modules (Section V-1.5):
  AMMOXID, BIOLGY, BIOPT, BSINKING, DSINKING, FLUFF, GROWTH, QDET, QUOTA, RESPC, RESPN, SEDEUL, SSINKING, TEMPCH, UPTAKENH, UPTAKENO

- Contaminant modules (Section V-1.6):
  CONSPM, MASS, MCOSB, MCROB, MC3D, SEDLAG, STAUCH

- Modules related to boundary conditions (Section V-1.7):
  BOUNDC, BSTRES, CD, CE, CHARNO, FLUXCO, HUMID, SURFLX, WAVCUR, WAVNUM
- Numerical routines (Section V-1.8):
  FNLIM, HAD2DU, HAD2DV, TRANSH, TRANSPC, TRANSPW, TRANSV, XADVDIS, XHAD3DU, XHAD3DV, YADVDIS, YHAD3DU, YHAD3DV, ZADVDIS
- Modules related to model setup (Section V-1.9):
  DEFAULT, DEFCON, DEFGRID, DEFGRID1, DEFICS, DEFOB2D, DEFOB3D, INCTUR, INIBIO, INICON, INITC, WRFCDAT, WROBDAT
- Modules performing file operations (Section V-1.10.1):
  CLOSEF, IGETFUN, OPENF, RDARI, RDARR, WRARI, WRARR
- Input/output routines for model setup (Section V-1.10.2):
  BBCIN, BCSIN, BCSOUT, BC2IN, CBCIN, HBCIN, INPA, METIN, OUTPA, PBCIN, RDCON, RDGRD, READIN, WAVIN, WRCON, WRGRD
- Routines for user-defined output (Section V-1.10.3):
  ANALFR, ANALPT2D, ANALPT3D, ANALYS, ANALYS0, ANALYS1, ANALYS2, ANAL2DVAR, ANAL3DVAR, AVRPT2D, AVRPT3D, AVR2DVAR, AVR3DVAR, INTEGR, INTEGR0, INTEGR1, INTEGR2, OUTPAR, OUTPT, OUTPT2D, OUTPT3D, OUT2DVAR, OUT3DVAR, PARPT3D, PRINT, WROUT
- Routines for the graphical interface (Section V-1.10.4):
  CDF2D, CDF3D, ERRCHECK
- Example routines for reading user-defined output (Section V-1.10.5):
  RDANAL, RDAVR, RDELL, RDOUT, RDPAR
- Utility routines (Section V-1.11):
  COMPLX, CZERO, DAYNUM, ERRBCS, ERRGRD, ERRICS, ERRMOD, ERROR, IDAADD, IDATES, IGREG, IJUL, IZERO, LEAP, LSQFIT, LUBKSB2, LUBKSB3, LUDCMP, NDIFF, NEWTIM, RANDOM, RAN3, THOMV, ZERO
- Interpolation functions (Section V-1.12)

1.2 Main programs

MAINPROG

File

mainprog.f

Type

main program

Purpose

main COHERENS program

Externals

ANALYS, BBCIN, BCSIN, BC2IN, BIOLGY, BSTRES, CBCIN, CONTNY, CRRNT1C, CRRNT2, CRRNT3C, CRRNT3P, DENSTY, DENSTY1, ERRMOD, HBCIN, HEAT, HEDDY, INITC, INTEGR, MASS, METIN, NEWTIM, OUTPA, OUTPAR, OUTPT, PBCIN, PRINT, RDCON, SALT, SEARHO, SEDEUL, SEDLAG, TRANSV, VEDDY1, VEDDY2, WAVIN, WAVNUM, WCALC

Description

The main program of COHERENS mainly consists in calling a number of routines (given in parentheses).

- Read the *con*-file with the values of the switches and parameters defined in DEFCON or set by default in DEFAULT and the parameters defined in the file *defmod.par* (including the TITLE of the run and output specifiers) and open the files for data input (RDCON).
- Check whether some of these input parameters are within the allowed range (ERRMOD).
- Evaluate time steps for the 3-D mode, meteorological and wave input.
- Start a series of initialisation procedures:
  - Initialise a series of parameters and arrays and read the initial conditions by calling INITC.
  - Read open boundary conditions, open boundary and surface forcing data by calling BCSIN, METIN, WAVIN and evaluate the surface fluxes at the initial time.
  - Initialise bottom stress (BSTRES).
  - Initialise density, buoyancy and expansion coefficients (IOPTD = 2) by calling SEARHO.
  - Write output for the initial time (OUTPT, ANALYS, INTEGR, OUTPAR, PRINT) if necessary.
- Start the time loop. Time is increased by the 2-D mode time step DELT after each iteration.
  - Update current date and time (NEWTIM).
  - Read meteorological data and update the surface fluxes at the selected times (METIN).
  - Read surface wave and open boundary data at the selected times (WAVIN, BC2IN, HBCIN, BBCIN, PBCIN, CBCIN).

- Update the horizontal diffusion coefficients (HEDDY), the pressure gradient in the 1-D case (DENSTY1), the baroclinic pressure gradient in the 3-D case (DENSTY), the turbulence arrays (VEDDY1 or VEDDY2) and the horizontal current at each predictor time step (CRRNT3P).
- Solve the depth-integrated continuity (CONTNY) and momentum (CRRNT2) equations at each 2-D time step.
- Update a series of field arrays at each corrector (3-D) time step in the following order: vertical grid spacing (TRANSV), horizontal current (CRRNT1C or CRRNT3C), vertical current (WCALC), salinity (SALT), temperature (HEAT), density, buoyancy and expansion coefficients (SEARHO), biological state variables (BIOLGY), Eulerian contaminant concentrations (MASS), inorganic sediment concentration (including an update of the fluff layer, SEDEUL), particle positions in the Lagrangian module (SEDLAG).
- Update the fields for harmonic analysis and time averaging at each 3-D time step (ANALYS, INTEGR).
- Write output (time series, harmonic, time-averaged, particle positions) at the appropriate time step (OUTPT, ANALYS, INTEGR, OUTPAR).
- Call PRINT for user-defined output.

• Write the files with the final conditions (OUTPA).

• Stop the program.

A flow chart of MAINPROG is given in Figure 5.1.1.

Remarks

Most calls in MAINPROG are only executed at selected values of the time step counter NT and for specific values of model switches.

NETCDFINT

File

netcdfint.f

Type

main program

Purpose

graphics interface program

Externals

CDF2D, CDF3D, CLOSEF, IGETFUN, OPENF, RDGRD

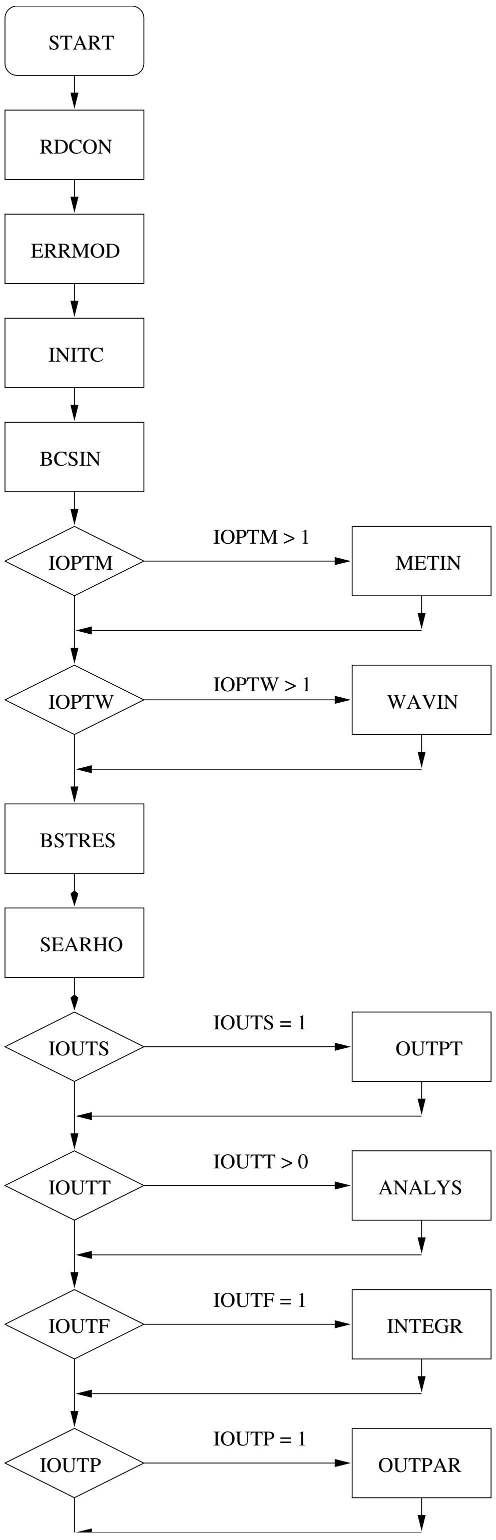


Figure 5.1.1: Flow chart of the main COHERENS program MAINPROG.

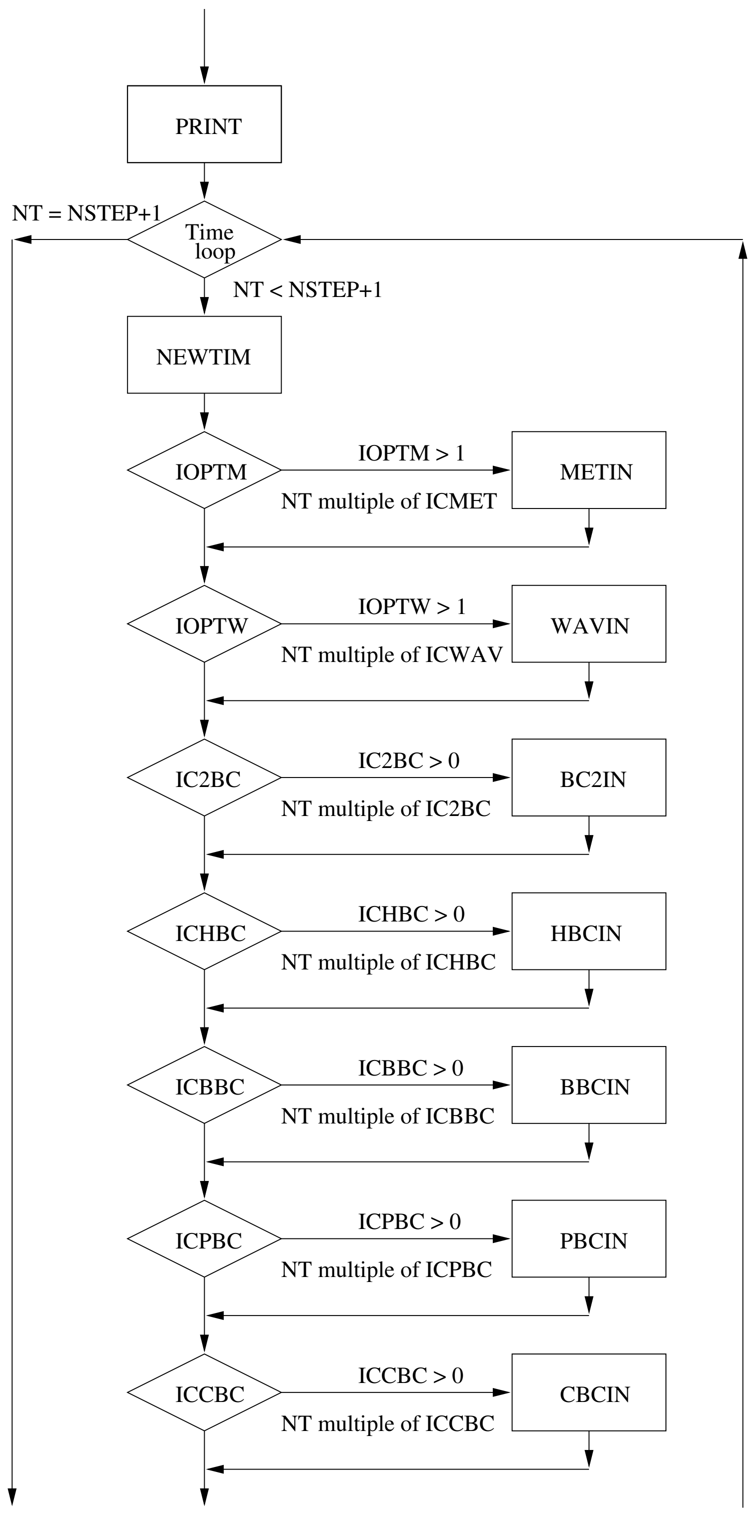


Figure 5.1.1: continued.

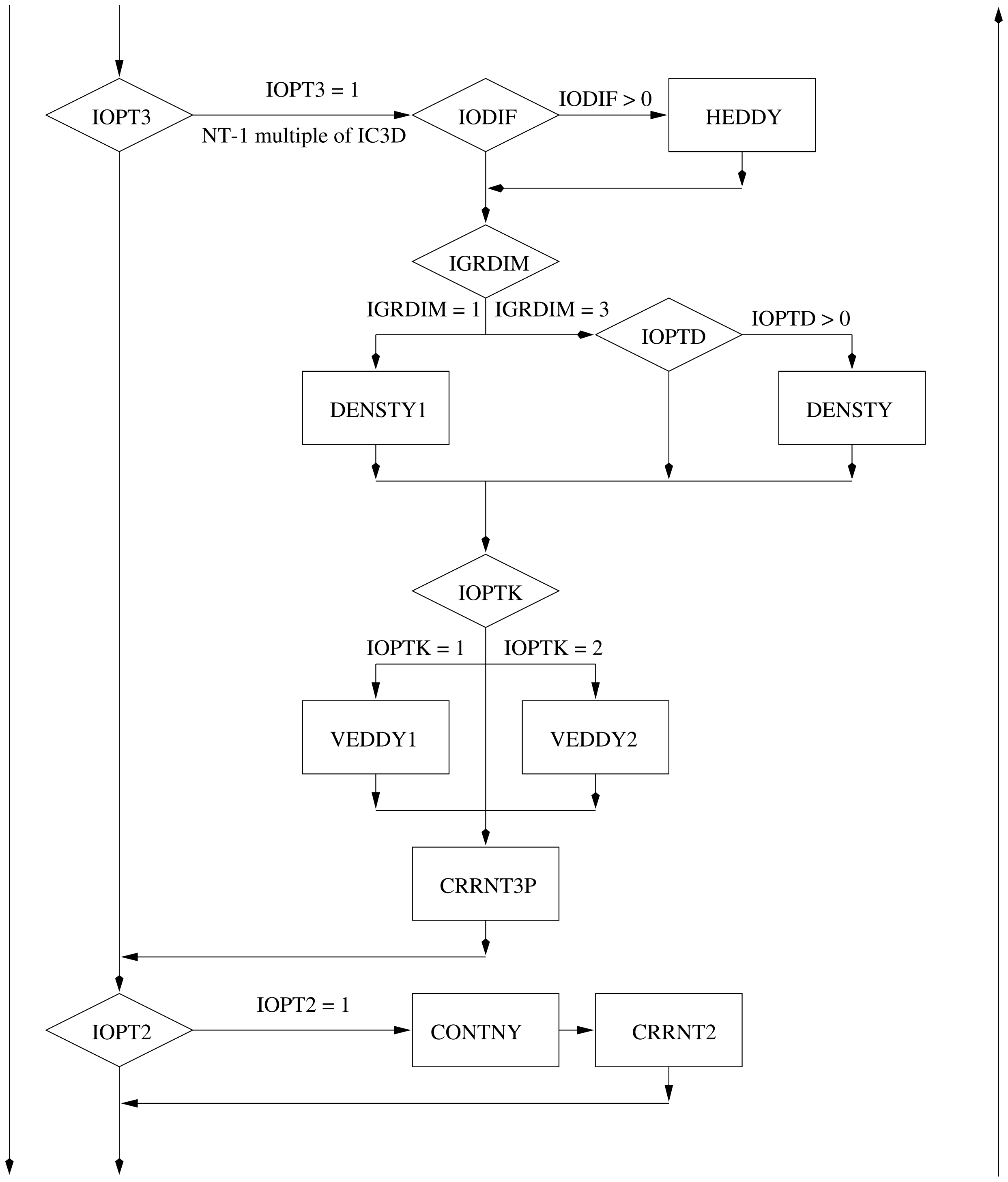


Figure 5.1.1: continued.

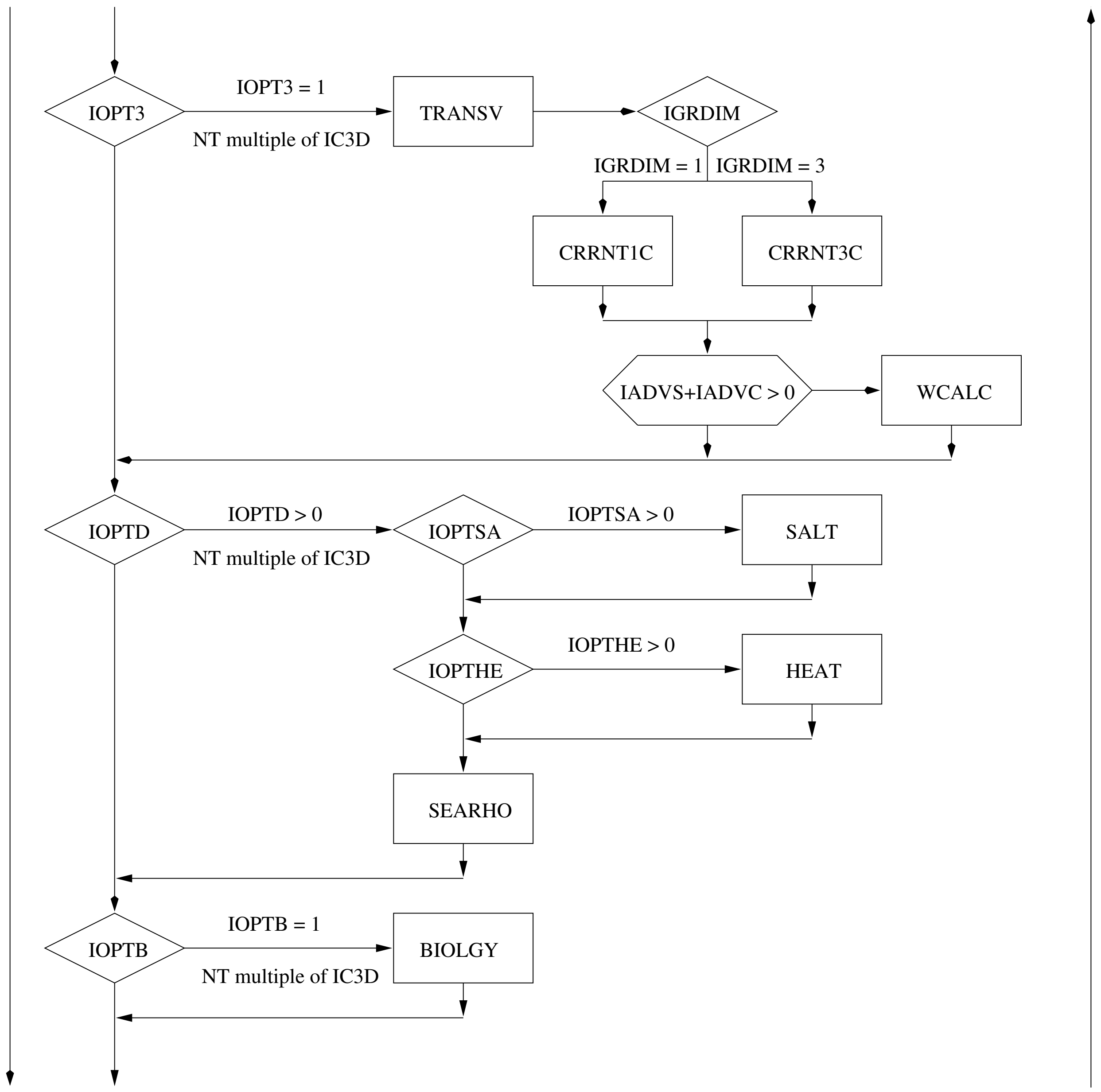


Figure 5.1.1: continued.

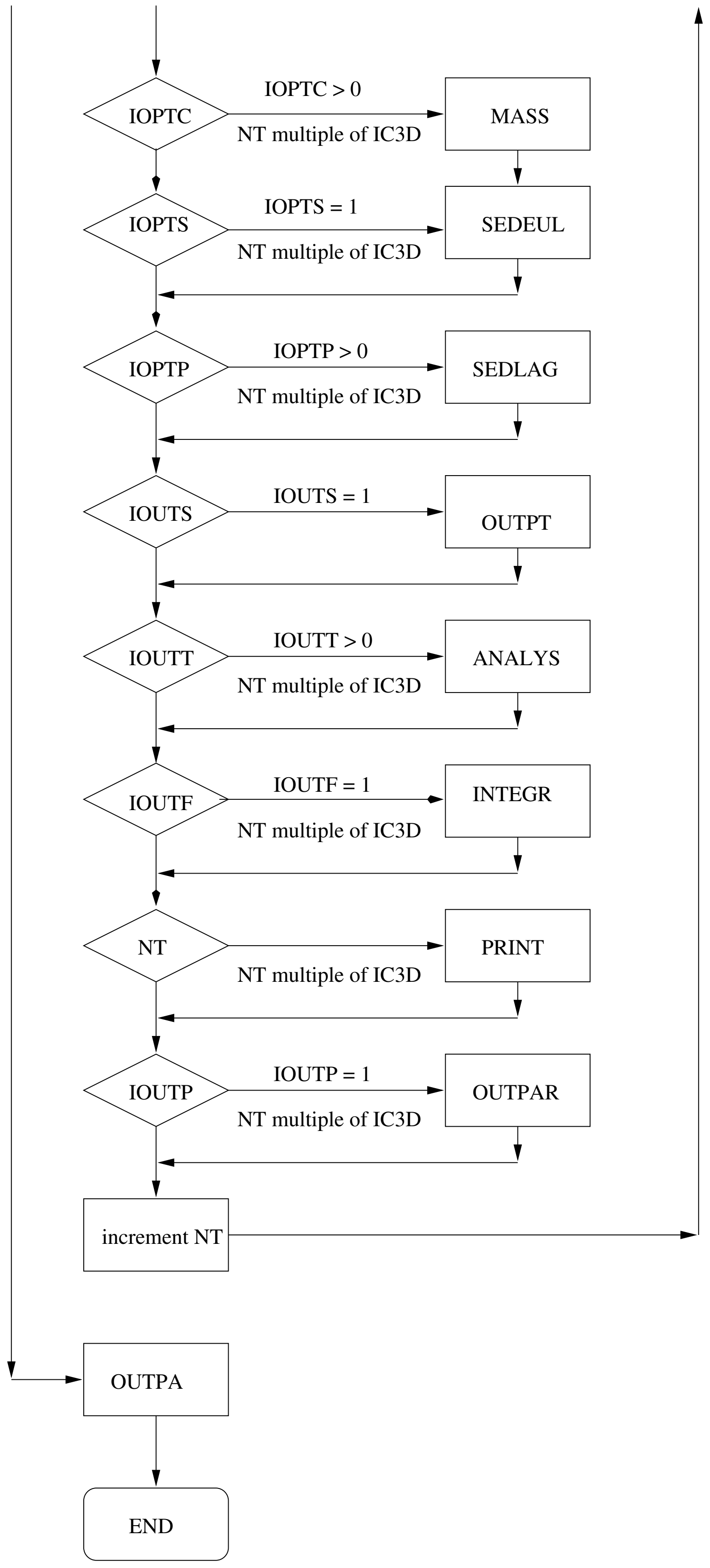


Figure 5.1.1: continued.

Description

The interface program converts model output into "netCDF" format:

- Read the number of (2-D and/or 3-D) files which are to be converted into netCDF format, and read the name of the first file (first 12 characters) from standard input. Default name of the input file is *defpost.par*.
- Read the output specifiers from the Information file.
- Check names and descriptions of field variables for compatibility with netCDF and FERRET conventions.
- Set the dimensions of the output data.
- Define units for horizontal and time coordinates. Horizontal grid units are either meters, kilometers or degrees (spherical case). Time is measured either in seconds, hours or days.
- Read the *grd*-file with the model grid arrays by calling RDGRD.
- Read the data from the 2-D output file (if non-empty) and store the results into an equivalent netCDF file by calling CDF2D. The name of the latter file is obtained by adding *.cdf* to the name of the model data file.
- Read the data from the 3-D output file (if non-empty) and store the results into an equivalent netCDF file by calling CDF3D. The name of the latter file is obtained by adding *.cdf* to the name of the model data file.
- Read the name of the next data file (if any) from standard input and proceed as previous.

Remarks

- The number of netCDF files created by the program is given by the number on the first line of the input parameter file (*defpost.par* by default) times 2 minus (eventually) empty data files.
- The program must be compiled with the netCDF library.

PREPROC

File

preproc.f

Type

main program

Purpose

preprocessor program which defines all input for COHERENS

Externals

BCSOUT, DEFAULT, DEFCON, DEFGRID, DEFGRID1, DEFICS, DEFOB2D, DEFOB3D, ERRICS, ERRMOD, ERROR, INCTUR, INICON, NDIFF, OUTPA, READIN, SEARHO, STAUCH, TRANSH, TRANSV, WRCON, WRFCDAT, WRGRD, WROBDAT

Description

- Define/write switches and model parameters:
  - read TITLE of the run from standard input (file *defmod.par*)
  - set default values by calling DEFAULT
  - set user-defined values by calling DEFCON
  - initialise date
  - reset parameters in the case of a 1-D application
  - initialise bottom roughness length and grazing pressures by default
  - read output parameters from standard input by calling READIN
  - check parameter ranges (ERRMOD)
  - write *con*-file (WRCON)
- Initialise model grid (DEFGRID or DEFGRID1) and grid spacings (TRANSH, TRANSV), write *grd*-file (WRGRD).
- Calculate the total number of 2-D time steps (NDIFF).
- Write the data files for meteorological (*met*) and wave (*wav*) input (WRFCDAT).
- Define boundary conditions and data for the 3-D (DEFOB3D) and 2-D (DEFOB2D) mode.
- Write open boundary data to the appropriate files (BCSOUT, WROBDAT).
- Initialise salinity and temperature by default.
- Define initial conditions (DEFICS) and update the vertical grid spacings (TRANSV).
- Initialise density, buoyancy and the expansion coefficients for temperature and salinity (SEARHO).
- Initialise turbulence fields (INCTUR).
- Initialise the total number of particles (STAUCH) and check the initial conditions for the particle module (ERRICS).
- Write the initial condition files (OUTPA).
- Stop the program.

A flow chart of PREPROC is given in Figure 5.1.2.

Remarks

- WRFCDAT is only called when either IOPTM or IOPTW is larger than 1.
- WROBDAT is only called when one of the counters IC2BC, ICHBC, ICBBC, ICPBC or ICCBC is non-zero.

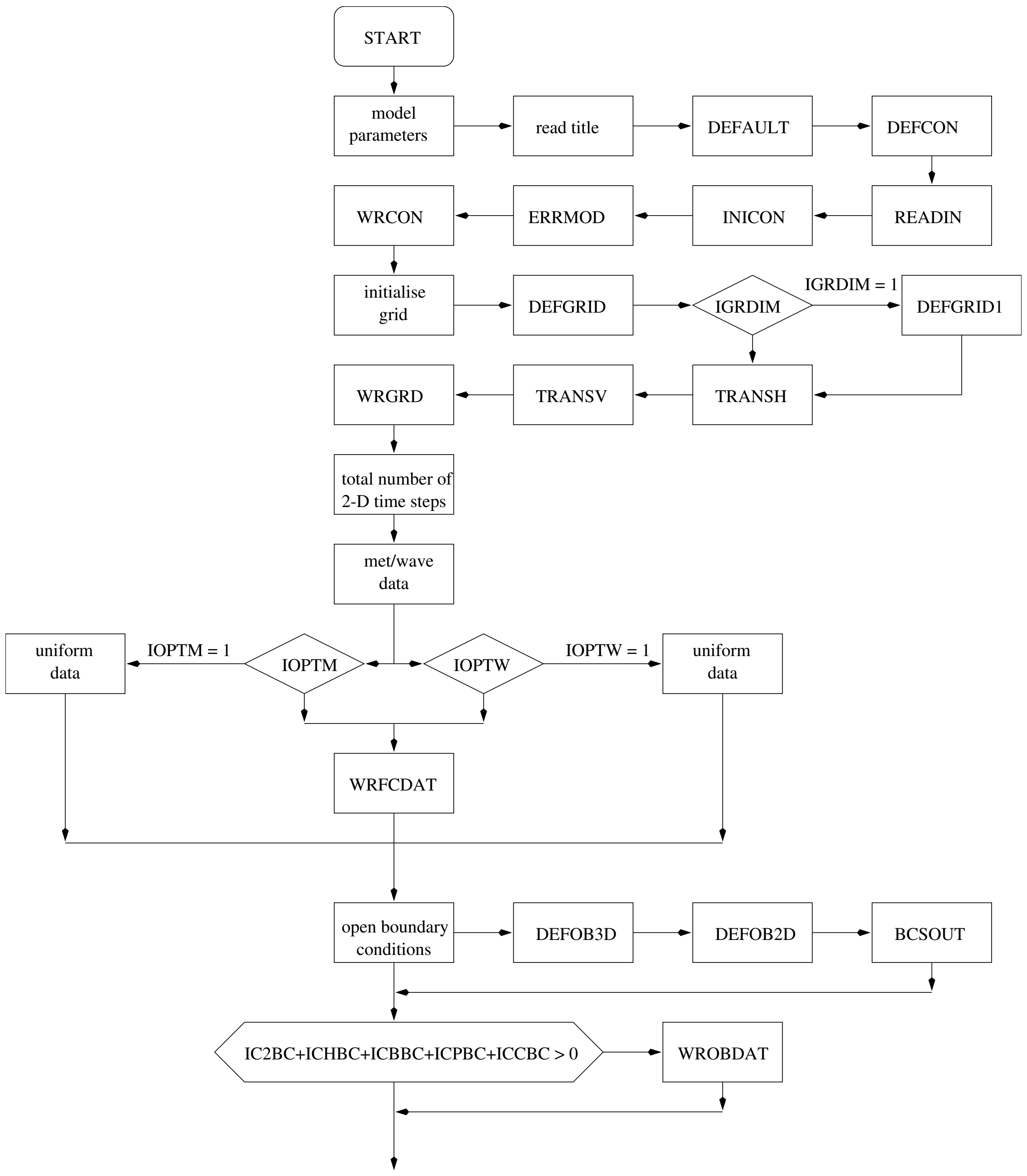


Figure 5.1.2: Flow chart of the preprocessor program PREPROC.

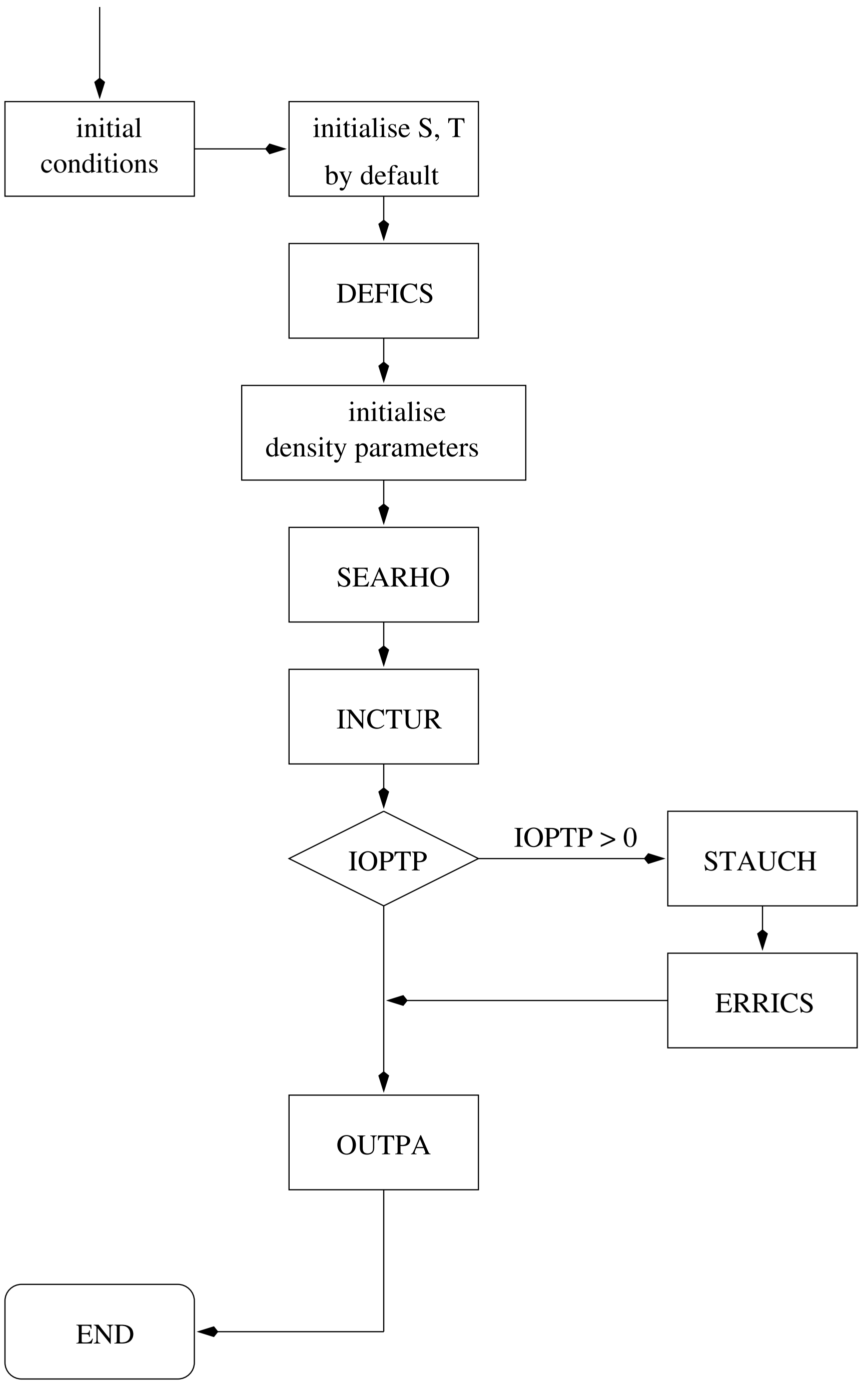


Figure 5.1.2: continued.

1.3 Physics

CONTNY

File
: *contny.f*

Type
: subroutine

Purpose
: solve the 2-D continuity equation

Calling program
: MAINPROG

Externals
: ERROR

Arguments
: none

Description

- Checks the CFL condition (3.4.6) on first call and stops the program if this condition is not satisfied.
- Solve equation (3.1.33) or (3.1.60) for the surface elevation ζ (using the discretised forms (3.4.12) and (3.4.335)).

Remarks
: The routine is called at all 2-D time steps if IOPT2 $=1$.

CRRNT1C

File
: *crrnt3.f*

Type
: subroutine

Purpose
: update the horizontal current at the corrector step in the case of a 1-D model application

Calling program
MAINPROG

Externals
BSTRES

Arguments
none

Description

- Update the horizontal current at the new baroclinic time t^{n+1} at all interior points using the discretised forms (3.4.363).
- Evaluate the depth-integrated current at all interior points.
- Update the current at the open boundaries by taking the value at the nearest interior point. The same precedure is taken for the depth-integrated current.
- Update the bottom stress using the new current values.

Remarks

- The procedure is such that u ($\overline{U}$) and v ($\overline{V}$) have the same values at respectively all U- and V-nodes.
- The routine is called at all 3-D (corrector) steps if IOPT3 $= 1$ and IGRDIM $= 1$.

CRRNT2

File
crrnt2.f

Type
subroutine

Purpose
solve the 2-D momentum equations

Calling program
MAINPROG

Externals
BOUNDC, HAD2DU, HAD2DV, UDCALC, VDCALC

Arguments
none

Description

This routine updates the depth-integrated current ($\overline{U}$, $\overline{V}$) using a number of subroutine calls:

- Evaluate the 2-D advective and diffusion terms for the depth-integrated current at the old barotropic time t^m (HAD2DU and HAD2DV).
- Solve the 2-D momentum equations (3.1.34)–(3.1.35) or (3.1.61)–(3.1.62) for the new baroclinic time t^{m+1} using the discretised forms (3.4.13)–(3.4.14) at all (wet) interior U- or V-nodes by calling UDCALC and VDCALC.
- Update the values of $\overline{U}$ and $\overline{V}$ at the open boundaries and at the new time step by calling BOUNDC.
- Update the values of ($\overline{U}^F$, $\overline{V}^F$) representing the values of ($\overline{U}$, $\overline{V}$) averaged over one baroclinic (3-D) time step (see equation (3.4.18)) and store the results in the arrays UD2F, VD2F.

Remarks

- The $\overline{U}$-equation is solved before the $\overline{V}$-equation at all odd (2-D) time steps, whereas the reverse occurs ($\overline{V}$ before $\overline{U}$) at all even (2-D) time steps.
- The routine is called at all 2-D time steps if IOPT2 $= 1$.
- For further details about the numerical procedures see Section III-4.3.

CRRNT3C

File

crrnt3.f

Type

subroutine

Purpose

update the horizontal current at the corrector time step in the case of a 3-D application

Calling program

MAINPROG

Externals

BSTRES

Arguments

none

Description

- Update the horizontal current (u,v) at the new baroclinic time t^{n+1} and evaluate the "filtered" advective velocities (u^F,v^F), at all interior points using the discretised forms (3.4.19)–(3.4.20) and (3.4.21)–(3.4.22).
- Update (u,v) and (u^F,v^F) at the open boundaries using either one of the boundary conditions (3.1.208)–(3.1.209) or (3.1.210) and the numerical procedures described in Section III-4.3.10c (including equation (3.4.353) in the spherical case).
- Update the bottom stress using the new current values.

Remarks

- The routine is called at all 3-D (corrector) time steps if IOPT3 = 1 and IGRDIM = 3.
- For further details about the numerical procedures see Section III-4.3.

CRRNT3P

File
  crrnt3.f

Type
  subroutine

Purpose
  update the horizontal current at the predictor time step

Calling program
  MAINPROG

Externals
  BSTRES, UCALC, VCALC

Arguments
  none

Description

- Store the old values of the surface elevation ζ, the vertical grid spacing (or Jacobian J) and the horizontal current into the arrays ZETA1, GZ1, U1, V1.
- Initialise the values of $\overline{U}^F$ and $\overline{V}^F$ (represented by the arrays UD2F, VD2F) to zero for a new series of 2-D mode calculations.

- Solve the horizontal momentum equations (3.1.25)–(3.1.26) or (3.1.50)–(3.1.51) at all (wet) interior U- or V-nodes by calling UCALC and VCALC.
- Evaluate the depth-integrated currents $\overline{U}^p$,$\overline{V}^p$ at the predictor time step using (3.4.11).
- Update the bottom stress and the friction coefficient k_b^p at the predictor step for a new series of 2-D mode calculations by calling BSTRES.

Remarks

- The u-equation is solved before the v-equation at all odd (3-D) time steps whereas the reverse occurs (v before u) at all even (3-D) time steps.
- The routine is called at all 3-D (predictor) time steps if IOPT3 $= 1$.
- For further details about the numerical procedures see Section III-4.3.

DENSTY

File
: *densty.f*

Type
: subroutine

Purpose
: evaluate the baroclinic pressure gradient in the case of a 3-D application

Calling program
: MAINPROG

Externals
: none

Arguments
: none

Local variables

RPRESS:	reduced pressure q_d ($\mathrm{m}^2/\mathrm{s}^2$)
UQDEN1:	the term $\partial(Jq_d)/\partial x_1$ on the right hand side of (3.4.140)
UQDEN2:	the factor G_1 discretised using (3.4.143)
VQDEN1:	the term $\partial(Jq_d)/\partial x_2$ on the right hand side of (3.4.141)
VQDEN2:	the factor G_2 discretised using (3.4.144)

Description

- Integrate the hydrostatic equilibrium equation (3.1.27) or (3.1.52) for the normalised reduced pressure q_d using (3.4.139).
- Evaluate the baroclinic pressure gradient components Q_1 or Q_λ at the interior U-nodes and Q_2 or Q_ϕ at the interior V-nodes using the numerical procedures described in Section III-4.3.9. Results are stored into the arrays UQDEN, VQDEN.
- Evaluate the depth integrals ($\overline{Q}_1$, $\overline{Q}_2$) or ($\overline{Q}_\lambda$, $\overline{Q}_\phi$) using (3.4.147). Results are stored in the arrays UDQDEN, VDQDEN.

Remarks

This routine is always called at each predictor time step if IOPTD > 0 and IGRDIM $= 3$.

DENSTY1

File

densty.f

Type

subroutine

Purpose

evaluate the pressure gradient in the case of a 1-D application

Calling program

MAINPROG

Externals

none

Arguments

none

Description

- Update the phases $\omega_n t + \varphi_{n0}$ in the harmonic expansions (3.1.84)–(3.1.85). The values are calculated modulo 2π to avoid rounding errors at large times and stored into the array PHASE0. This array is saved at the end of the program (*hou*-file) and can be used for an eventual restart.
- Update the surface elevation using the expansion (3.1.83).
- Update the pressure gradient in the X-direction using the expansion (3.1.84).

- Update the pressure gradient in the Y-direction using the expansion (3.1.85).

Remarks

This routine is called at each time step if IGRDIM = 1.

HEAT

File

heat.f

Type

subroutine

Purpose

solve the temperature equation

Calling program

MAINPROG

Externals

IRRAD, TRANSPC, ZERO

Arguments

none

Description

The routine updates the temperature field (see Figure 5.1.3 for a general flow chart).

- Set the sink terms to zero on first call.
- Evaluate the surface fluxes using the known values for the solar (IOPTHE = 1 only) and non-solar heat fluxes.
- Evaluate the amount of heat absorbed within the water column (array QHEAT) by calling IRRAD if IOPTHE = 2.
- Evaluate the source (solar absorption) term if IOPTHE = 2 by dividing QHEAT through $\rho_0 c_p$.
- Solve the temperature equation (3.1.28) or (3.1.53) by calling TRANSPC.

Remarks

- The routine is called at all 3-D time steps when IOPTHE > 0.
- The surface heat flux is evaluated differently if IOPTHE = 1 or 2 (see Section III-1.6.1b for details).

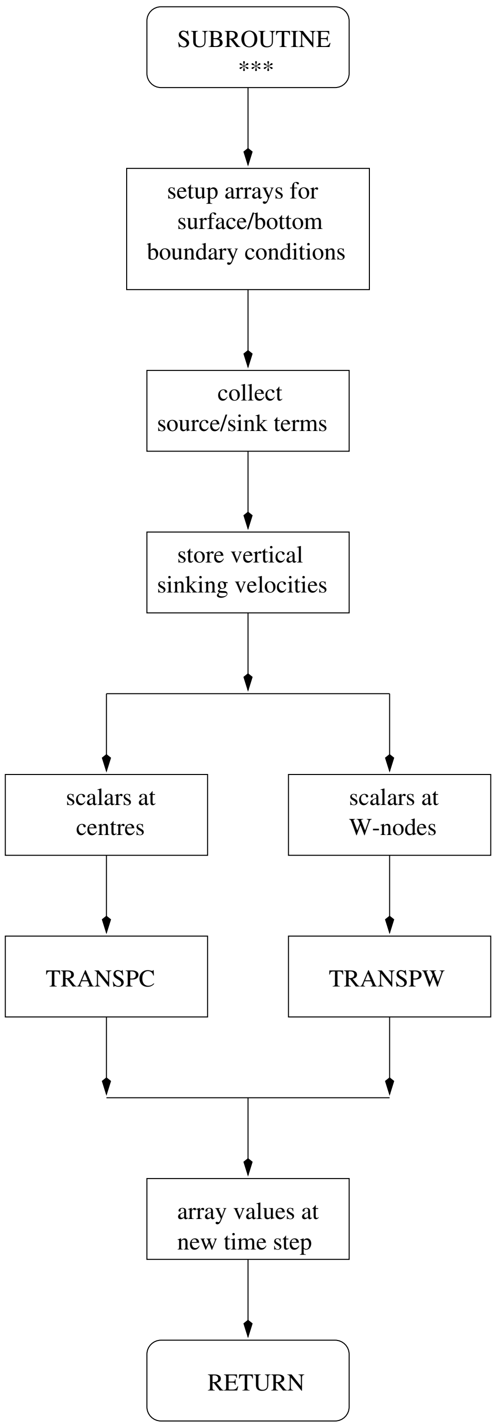

SUBROUTINE
***
setup arrays for surface/bottom boundary conditions
collect source/sink terms
store vertical sinking velocities
scalars at centres
scalars at W-nodes
TRANSPC
TRANSPW
array values at new time step
RETURN

Figure 5.1.3: General procedure for updating a scalar transport variable.

HEDDY

File

heddy.f

Type

subroutine

Purpose

evaluate the horizontal diffusion coefficients

Calling program

MAINPROG

Externals

none

Arguments

none

Local variables

DUDXC:	the x_1-derivative of u at the cell centres
DVDYC:	the x_2-derivative of v at the cell centres
DUDYV:	the x_2-derivative of u at the V-nodes
DVDXU:	the x_1-derivative of v at the U-nodes

Description

The routine updates the values of the horizontal diffusion coefficients ν_H, λ_H and the depth-integrated value $\overline{\nu_H}$ of the former coefficient as given by (3.1.40) at each 3-D time step. The formulation depends on the value of the switch IODIF. If IODIF equals 1, ν_H and λ_H are spatially uniform. Their values are initialised in subroutine INITC at the start of the program. If IODIF $= 2$, the coefficients are evaluated using the formulation (3.1.151)–(3.1.152) or (3.1.153)–(3.1.154). The following arrays are then updated:

- HEDDYVC: the value of ν_H at the cell centres using (3.4.102) and either (3.4.105) or (3.4.350).
- HEDDYDC: the value of λ_H at the cell centres using (3.4.229) and either (3.4.105) or (3.4.350).
- HEDDYVU: the value of ν_H at the U-nodes using (3.4.103) and either (3.4.106) or (3.4.351).
- HEDDYDU: the value of λ_H at the U-nodes using (3.4.230) and either (3.4.106) or (3.4.351).

- HEDDYVV: the value of ν_H at the V-nodes using (3.4.104) and either (3.4.107) or (3.4.352).
- HEDDYDV: the value of λ_H at the V-nodes using (3.4.231) and either (3.4.107) or (3.4.352).
- DHEDDYVC: the value of $\overline{\nu_H}$ at the cell centre using the first relation (3.4.108).
- DHEDDYVU: the value of $\overline{\nu_H}$ at the U-nodes using the second relation (3.4.108).
- DHEDDYVV: the value of $\overline{\nu_H}$ at the V-nodes using the third relation (3.4.108).

The horizontal derivatives of the currents u and v, used in the evaluation of the strain rate D_T, are first stored in a series of local arrays.

Remarks

Note that $\overline{\nu_H}$ needs to be updated as well if IODIF $= 1$.

IRRAD

File

heat.f

Type

subroutine

Purpose

heat absorption term in the temperature equation

Calling program

HEAT

Externals

none

Arguments

none

Local variables

RADLO1: value of $I^w_{1;k}$

RADLO2: value of $I^w_{2;k}$

RADUP1: value of $I^w_{1;k+1}$

RADUP2: value of $I^w_{2;k+1}$

R2EXP: the factor R^*_2 obtained from (3.4.288)

Description

- Update the diffusive attenuation coefficient k_2 using (3.4.286).
- The infrared and short-wave components of solar irradiance are determined at the W-nodes using the iterative scheme (3.4.287)–(3.4.288).
- Solar absorption is then evaluated using (3.4.289).

Remarks

Subroutine IRRAD is called from the HEAT module if IOPTHE $= 2$.

SALT

File

salt.f

Type

subroutine

Purpose

solve the salinity equation

Calling program

MAINPROG

Externals

TRANSPC, ZERO

Arguments

none

Description

The routine updates the salinity field (see Figure 5.1.3 for a general flow chart).

- Set all source/sink terms to zero on first call.
- Solve the salinity equation (3.1.29) or (3.1.54) by calling TRANSPC.

Remarks

The SALT module is called at all 3-D time steps when IOPTSA > 0.

SEARHO

File

densty.f

Type

subroutine

Purpose

evaluate the density, the buoyancy and the salinity and temperature expansion coefficients using the linear or the general equation of state of seawater

Reference

UNESCO, 1981

Calling program

MAINPROG, PREPROC

Externals

none

Arguments

none

Description

- The density ρ is computed either using the linear (IOPTD $= 1$) or the general (IOPTD $= 2$) equation of state.
- The buoyancy b is obtained from (3.4.138).
- The expansion coefficients β_S and β_T are computed using the general equation of state (equations (3.1.164)–(3.1.166)) if IOPTD $= 2$.

Remarks

- Note that, in the case of a linear equation of state, b is calculated using

$$b = g\Big(\beta_S(S - S_0) - \beta_T(T - T_0)\Big) \tag{5.1.1}$$

  obtained by combining (3.1.16) and (3.1.162) to avoid rounding errors in the computation.
- SEARHO is called at $t = 0$ and at each 3-D time step if IOPTD > 0.

SOLRAD

File
: *metin.f*

Type
: subroutine

Purpose
: evaluate the solar radiation Q_{sol} incident on the sea surface

Calling program
: METIN

Externals
: DAYNUM, IDATES, LEAP

Arguments
: none

Local variables

- AAL: water vapor and ozon absorption coefficient A_α (=0.09)
- ALB: sea surface albedo A_s (=0.06)
- DEC: the sun's declination $\delta_\odot$ in degrees from (3.1.259)
- GAMF: the sun's altitude $\gamma_\odot$ from (3.1.258)
- GAMMAX: the solar altitude at noon $\gamma_{\odot,max}$ in degrees
- HA: the sun's hour angle $H_\odot$ in radians from (3.1.261)
- HR: time of the day t_h in hours (GMT)
- OM: optical air mass m_o from (3.1.268)
- ORB: orbital correction factor from (3.1.256)
- QCS: clear sky value Q_{cs} of solar radiation at the sea surface from (3.1.270)
- QD: direct solar radiation Q_{dir} at the sea surface from (3.1.264)
- QDIF: diffuse sky radiation Q_{dif} from (3.1.269)
- Q0: solar flux Q_s at the top of the atmosphere from (3.1.255)
- ROT: optical thickness δ_R from (3.1.266)
- SOLC: solar constant Q_0 (=1367 W/m^2)
- TE: equation of time TE from (3.1.262)
- TTF: Linke's factor from (3.1.267)

Description

The value of the surface solar irradiance is calculated along the lines given in Section III–1.6.5. Note that angles (except the solar altitude at noon $\gamma_{\odot,max}$) and hours are converted to radians.

Remarks

SOLRAD is called from the surface forcing module only when IOPTM = 2 or 3 and either the temperature (IOPTHE $>$ 0) or the biological module (IOPTB = 1) are activated.

UCALC

File

crrnt3.f

Type

subroutine

Purpose

solve the u-momentum equation (3.1.25) or (3.1.50) for the predictor time step

Calling program

CRRNT3P

Externals

THOMV, XHAD3DU, YHAD3DU, ZADVDIS, ZERO

Arguments

none

Local variables

ULHSA: A-vector in the tridiagonal equation system (u_{k-1}-terms)

ULHSB: B-vector in the tridiagonal equation system (u_k-terms)

ULHSC: C-vector in the tridiagonal equation system (u_{k+1}-terms)

URHS: D-vector in the tridiagonal equation system (explicit terms)

U2A: value of u_A (step A)

U2B: value of u_B (step B)

Description

The numerical solution of the u-equation in UCALC is displayed schematically in Figure 5.1.4 and proceeds along the lines given in Section III-4.3.

- Initialise arrays.
- Interpolate arrays at the U-nodes: vertical velocity, eddy viscosity, vertical grid spacing, X-component of surface stress.
- The depth-integrated terms $\overline{D}_1^h - \overline{A}_1^h$ or $\overline{D}_\lambda^h - \overline{A}_\lambda^h$ in the 2-D momentum equations are evaluated using the discretised forms (3.4.83), (3.4.110) in the Cartesian or (3.4.346), (3.4.348) in the spherical case. This is performed as follows:
  - The integral $\overline{\mathcal{D}_{xx}(u)} - \overline{\mathcal{A}_{hx}(u)}$ is stored into the array XINTU by calling XHAD3DU.
  - The integral $\overline{\mathcal{D}_{yx}(u,v)} - \overline{\mathcal{A}_{hy}(u)}$, including (eventually) an additional spherical term, is stored into the array YINTU by calling YHAD3DU.
  - The term $-\overline{\mathcal{A}_{hx}}(\overline{U}) - \overline{\mathcal{A}_{hy}}(\overline{U})$ including (eventually) an additional spherical term, is stored into the array UAH2D by calling HAD2DU.
  - The term $\overline{\mathcal{D}_{xx}}(\overline{U}) + \overline{\mathcal{D}_{yx}}(\overline{U}, \overline{V})$ including (eventually) an additional spherical term, is stored into the array UDH2D by calling HAD2DU.
  - The integral $\overline{D}_1^h - \overline{A}_1^h$ or $\overline{D}_\lambda^h - \overline{A}_\lambda^h$ is obtained by making the sum XINTU + YINTU + UAH2D - UDH2D and stored into the array UADHDEV.

The remainder of the procedure depends on the value of the advection switch IADVC. If IADVC $= 0$ (no advection, including the 1-D case) or 1 (upwind scheme), the predicted value u^p is obtained as follows:

- The explicit advective and diffusion terms are stored in URHS from the previous calls to XHAD3DU and YHAD3DU (see above).
- Initialise the arrays of the triangular matrix and assemble inertia terms.
- Add Coriolis force, barotropic and baroclinic pressure gradient to URHS.
- Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal matrix system by calling ZADVDIS.
- Solve the tridiagonal system (THOMV) for u^p.
- Exit.

As explained in Section III-4.3, the numerical solution uses the operator splitting method if IADVC equals 2 (Lax-Wendroff) or 3, 4 (TVD scheme):

- Solve (3.4.26) for $u_A^{n+1/3}$ with the terms on the right determined from a call to XHAD3DU.
- Solve (3.4.27) for $u_A^{n+2/3}$ with the terms on the right determined from a call to YHAD3DU.
- Solve (3.4.28) for u_A^p:
  - Initialise the arrays of the triangular matrix and assemble inertia terms.
  - Add Coriolis force, barotropic and baroclinic pressure gradient to URHS.
  - Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal matrix system by calling ZADVDIS.

  - Solve the tridiagonal system (THOMV) for u_A^p.
- Solve (3.4.29) for $u_B^{n+1/3}$ using the previous procedure.
- Solve (3.4.30) for $u_B^{n+2/3}$ with the terms on the right determined from a call to YHAD3DU.
- Solve (3.4.31) for u_B^p with the terms on the right determined from a call to XHAD3DU.
- Obtain u^p by averaging over the steps A and B (see equation (3.4.32)).
- Exit.

Remarks

- UCALC is only called if IOPT3 $= 1$.
- See Section III-4.3 for further details about the numerical procedures.

UDCALC

File

crrnt2.f

Type

subroutine

Purpose

solve the 2-D u-momentum equation (3.1.34) or (3.1.61)

Calling program

CRRNT2

Externals

none

Arguments

none

Description

The module solves the $\overline{U}$-momentum equation with the aid of the discretised form (3.4.13):

- Collect inertia terms.
- Add Coriolis force, barotropic pressure gradient and surface stress on the right hand side.

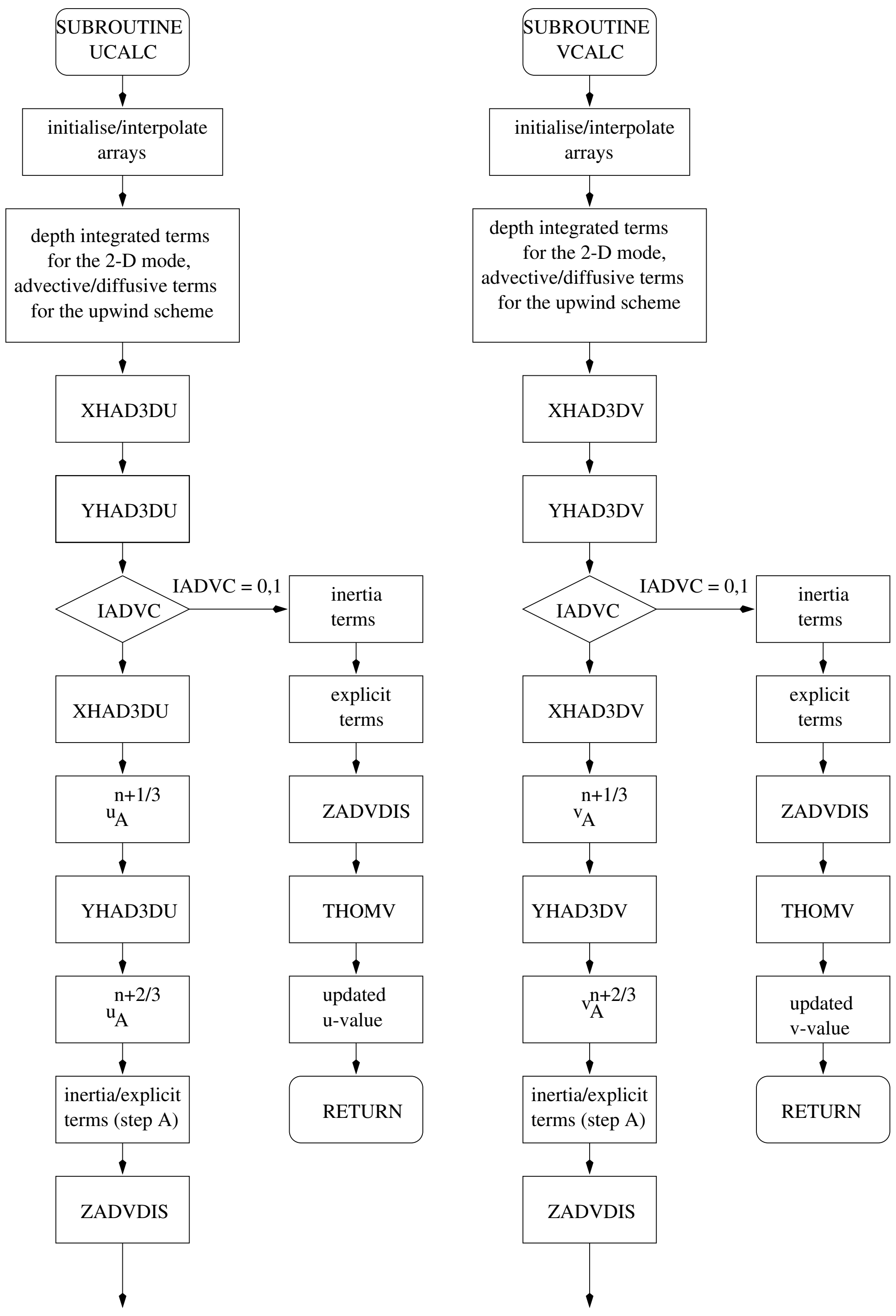


Figure 5.1.4: Flow chart of subroutines UCALC and VCALC.

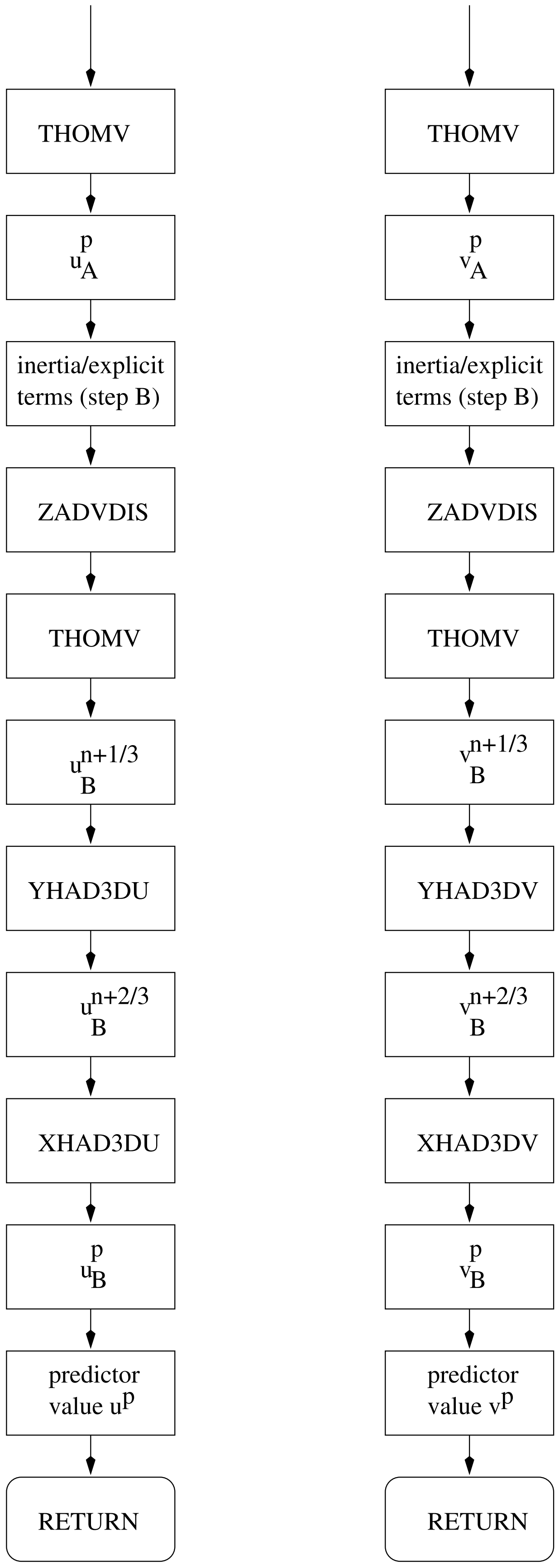


Figure 5.1.4: continued.

- Add bottom stress components to the left and right hand side.
- Add the depth integrals $\overline{D}_1^h - \overline{A}_1^h$ or $\overline{D}_\lambda^h - \overline{A}_\lambda^h$ (array UADHDEV), the 2-D advective terms (array UAH2D) and the 2-D diffusion terms (array UDH2D) to the right hand side.
- Add the baroclinic pressure gradient to the right hand side (array UDQDEN).
- Update $\overline{U}$ at the new barotropic time t^{m+1}.

Remarks

- The $\overline{U}$-equation is solved at each 2-D time step.
- The routine is only called when IOPT2 $= 1$.
- See Section III-4.3 for further details about the numerical procedures.

VCALC

File

crrnt3.f

Type

subroutine

Purpose

solve the v-momentum equation (3.1.26) or (3.1.51) for the predictor time step

Calling program

CRRNT3P

Externals

THOMV, XHAD3DV, YHAD3DV, ZADVDIS, ZERO

Arguments

none

Local variables

VLHSA: A-vector in the tridiagonal equation system (v_{k-1}-terms)

VLHSB: B-vector in the tridiagonal equation system (v_k-terms)

VLHSC: C-vector in the tridiagonal equation system (v_{k+1}-terms)

VRHS: D-vector in the tridiagonal equation system (explicit terms)

V2A: value of v_A (step A)

V2B: value of v_B (step B)

Description

The numerical solution of the v-equation in VCALC is displayed schematically in Figure 5.1.4 and proceeds along the lines given in Section III-4.3.

- Initialise arrays.
- Interpolate arrays at the V-nodes: vertical velocity, eddy viscosity, vertical grid spacing, Y-component of surface stress.
- The depth-integrated terms $\overline{D}^h_2 - \overline{A}^h_2$ or $\overline{D}^h_\phi - \overline{A}^h_\phi$ in the 2-D momentum equations are evaluated using the discretised forms (3.4.84), (3.4.111) in the Cartesian or (3.4.347), (3.4.349) in the spherical case. This is performed as follows:
  - The integral $\overline{\mathcal{D}_{xy}(u,v)} - \overline{\mathcal{A}_{hx}(v)}$ including (eventually) an additional spherical term, is stored into the array XINTV by calling XHAD3DV.
  - The integral $\overline{\mathcal{D}_{yy}(v)} - \overline{\mathcal{A}_{hy}(v)}$, is stored into the array YINTV by calling YHAD3DV.
  - The term $-\overline{\mathcal{A}_{hx}}(\overline{V}) - \overline{\mathcal{A}_{hy}}(\overline{V})$ including (eventually) an additional spherical term, is stored into the array VAH2D by calling HAD2DV.
  - The term $\overline{\mathcal{D}_{xy}}(\overline{U},\overline{V}) + \overline{\mathcal{D}_{yy}}(\overline{V})$ including (eventually) an additional spherical term, is stored into the array VH2D by calling HAD2DV.
  - The integral $\overline{D}^h_2 - \overline{A}^h_2$ or $\overline{D}^h_\phi - \overline{A}^h_\phi$ is obtained by making the sum XINTV + YINTV + VAH2D - VDH2D and stored into the array VADHDEV.

The remainder of the procedure depends on the value of the advection switch IADVC. If IADVC $=0$ (no advection, including the 1-D case) or 1 (upwind scheme), the predicted value v^p is obtained as follows:

- The explicit advective and diffusion terms are stored in VRHS from the previous calls to XHAD3DV and YHAD3DV (see above).
- Initialise the arrays of the triangular matrix and assemble inertia terms.
- Add Coriolis force, barotropic and baroclinic pressure gradient to VRHS.
- Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal matrix system by calling ZADVDIS.
- Solve the tridiagonal system (THOMV) for v^p.
- Exit.

As explained in Section III-4.3, the numerical solution uses the operator splitting method if IADVC equals 2 (Lax-Wendroff) or 3, 4 (TVD scheme):

- Solve (3.4.35) for $v_A^{n+1/3}$ with the terms on the right determined from a call to XHAD3DV.
- Solve (3.4.36) for $v_A^{n+2/3}$ with the terms on the right determined from a call to YHAD3DV.
- Solve (3.4.37) for v_A^p:

- Initialise the arrays of the triangular matrix and assemble inertia terms.
- Add Coriolis force, barotropic and baroclinic pressure gradient to VRHS.
- Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal matrix system by calling ZADVDIS.
- Solve the tridiagonal system (THOMV) for v_A^p.

- Solve (3.4.38) for $v_B^{n+1/3}$ using the previous procedure.
- Solve (3.4.39) for $v_B^{n+2/3}$ with the terms on the right determined from a call to YHAD3DV.
- Solve (3.4.40) for v_B^p with the terms on the right determined from a call to XHAD3DV.
- Obtain v^p by averaging over the steps A and B (see equation (3.4.41)).
- Exit.

Remarks

- VCALC is only called if IOPT3 $= 1$.
- See Section III-4.3 for further details about the numerical procedures.

VDCALC

File

crrnt2.f

Type

subroutine

Purpose

solve the 2-D v-momentum equation (3.1.35) or (3.1.62)

Calling program

CRRNT2

Externals

none

Arguments

none

Description

The module solves the $\overline{V}$-momentum equation with the aid of the discretised form (3.4.14):

- Collect inertia terms.
- Add Coriolis force, barotropic pressure gradient and surface stress on the right hand side.
- Add bottom stress components to the left and right hand side.
- Add the depth integrals $\overline{D}_2^h - \overline{A}_2^h$ or $\overline{D}_\phi^h - \overline{A}_\phi^h$ (array VADHDEV), the 2-D advective terms (array VAH2D) and the 2-D diffusion terms (array VDH2D) to the right hand side.
- Add the baroclinic pressure gradient to the right hand side (array VDQDEN).
- Update $\overline{V}$ at the new barotropic time t^{m+1}.

Remarks

- The $\overline{V}$-equation is solved at each 2-D time step.
- The routine is only called when IOPT2 $= 1$.
- See Section III-4.3 for further details about the numerical procedures.

WCALC

File
: *crrnt3.f*

Type
: subroutine

Purpose
: solve (3.1.46) or (3.1.68) for the transformed vertical velocity $J\tilde{w}$

Calling program
: MAINPROG

Externals
: WPCALC

Arguments
: none

Description
: The procedure uses the discretised forms (3.4.23) and (3.4.336). Note that (u,v) are replaced by the advective velocities (u^F,v^F).

Remarks

The routine is not called if vertical advection is disabled in the momentum and scalar transport equations (IADVC = IADVS = 0).

WPCALC

File

crrnt3.f

Type

subroutine

Purpose

solve (3.1.47) or (3.1.69) for the physical vertical velocity w

Calling program

WCALC

Externals

none

Arguments

none

Local variables

U2JZ: value of $J_k^{u;n+1}u_k^{n+1}x_{3;k}^{u;n+1}$

V2JZ: value of $J_k^{v;n+1}v_k^{n+1}x_{3;k}^{v;n+1}$ (eventually multiplied by $\cos\phi^v$)

W2JZ: value of $(J\tilde{w})_k^{n+1}x_{3;k}^{w;n+1}$

ZJOLD: value of $J_k^n x_{3;k}^{c;n}$

ZJNEW: value of $J_k^{n+1}x_{3;k}^{c;n+1}$

Description

The procedure uses the discretised forms (3.4.24) and (3.4.337). The velocity w is stored into the array W2PHYS. Note that w is evaluated at the cell centre.

Remarks

- The internal calculations in the program only use the transformed vertical velocity $J\tilde{w}$. The purpose of WPCALC is to provide the physical velocity as a useful quantity for user-defined output.
- WPCALC is not called if vertical advection is disabled in the momentum and scalar transport equations (IADVC = IADVS = 0).

1.4 Turbulence

BUOFR2

File

  veddy_in.f

Type

  real function

Purpose

  calculate the squared buoyancy frequency N^2 (s^{-2})

Calling program

  DISMIN, FNTURF, INCTUR, PRODUC, RICH, VEDDY2, ZLMAX

Externals

  none

Arguments

  K,J,I: cell indices at W-node

Description

  Evaluates the squared buoyancy frequency N^2, defined by (3.1.90), using the discretisation (3.4.299)–(3.4.300). To avoid spurious numerical oscillations its value is averaged over the neighbouring cells in the horizontal (excluding solid and open boundaries).

Remarks

  The vertical index K must be between 2 and NZ.

DISLEN

File

  dissip.f

Type

  subroutine

Purpose

  evaluate the dissipation rate ε ($k-l$ theory)

Calling program

  INCTUR, VEDDY2

Externals
: none

Arguments
: none

Description
: Evaluate ε as a function of k and l using (3.1.116).

Remarks
: The dissipation rate is only evaluated when IOPTK $= 2$.

DISMIN

File
: *veddy_in.f*

Type
: real function

Purpose
: evaluate the minimum dissipation rate ($k - \varepsilon$ theory)

Calling program
: VEDDY2

Externals
: BUOFR2

Arguments
: K,J,I: cell indices at W-node

Description
: Evaluates the minimum dissipation rate using the first equation of (3.1.143) and the relation $\varepsilon_{min} = \alpha_{N,max}^{-1/2}$.

Remarks

- DISMIN is only called when IOPTK $= 2$, NTRANS $\neq 0$, ILIM $= 1$ and ITCPAR $=$ 2.
- The vertical index K must be between 2 and NZ.

DISSIP

File
: *dissip.f*

Type
: subroutine

Purpose
: solve the ε-equation

Calling program
: VEDDY2

Externals
: TRANSPW

Arguments
: none

Description
: The routine solves equation (3.1.128) for the dissipation rate ε (a general flow chart is shown in Figure 5.1.3).

- Apply the surface and bottom boundary conditions (3.1.180) and (3.1.190). The results are stored into temporary arrays used as arguments in the call to TRANSPW.
- Evaluate the source and sink terms appearing on the right hand side of (3.1.128) using the discretisations (3.4.303) and (3.4.306).
- Solve (3.1.128) by calling TRANSPW.

Remarks
: The ε-equation is only solved if IOPTK $= 2$, NTRANS $= 2$ and ITCPAR $= 2$.

FNSHB

File
: *veddy_in.f*

Type
: real function

Purpose
: evaluate the stability function S_h or S_b

Calling program

INCTUR, VEDDY2

Externals

GVEDMA

Arguments

K,J,I: cell indices at W-node

X: the stability parameter G_h or α_N

Description

The function calculates the stability coefficient for scalars using one of the following available formulations:

- S_h using (3.1.118) if ITCPAR $= 1$, ISTPAR $= 1$
- S_h using (3.1.122) if ITCPAR $= 1$, ISTPAR $= 2$
- S_b using (3.1.120) if ITCPAR $= 2$, ISTPAR $= 1$
- S_b using (3.1.123) if ITCPAR $= 2$, ISTPAR $= 2$

Remarks

- FNSHB is only called when IOPTK $= 2$.
- The vertical index K must be between 2 and NZ.

FNSMU

File

veddy_in.f

Type

real function

Purpose

evaluate the stability function S_m or S_u

Calling program

INCTUR, VEDDY2

Externals

FVEDMA

Arguments

K,J,I: cell indices at W-node

X: the stability parameter G_h or α_N

Description

The function calculates the stability coefficient for momentum using one of the following available formulations:

- S_m using (3.1.118) if ITCPAR = 1, ISTPAR = 1
- S_m using (3.1.122) if ITCPAR = 1, ISTPAR = 2
- S_u using (3.1.120) if ITCPAR = 2, ISTPAR = 1
- S_u using (3.1.123) if ITCPAR = 2, ISTPAR = 2

Remarks

- FNSMU is only called when IOPTK = 2.
- The vertical index K must be between 2 and NZ.

FNTURF

File

veddy_in.f

Type

real function

Purpose

evaluate the quantity k/l^2 (s^{-2}) used in the zero-equation turbulence model

Calling program

VEDDY2

Externals

BUOFR2, FVEDMA, GVEDMA, SHFR2

Arguments

K,J,I: cell indices at W-node

Description

The function calculates k/l^2 for the zero equation turbulence scheme (see equations (3.1.136)–(3.1.140) and Section III-1.2.2b for details). The formulation depends on the values of the switches ISTPAR and ITCPAR. The second degree equation (3.1.136) is solved using an algorithm described in Press et al. (1989).

Remarks

- FNTURF is only called when IOPTK $= 2$ and NTRANS $= 0$.
- The vertical index K must be between 2 and NZ.

FVEDMA

File
: *veddy_in.f*

Type
: real function

Purpose
: evaluate $f_m(Ri)$ in the Munk-Anderson turbulence formulation

Calling program
: FNSMU, FNTURF, VEDDY1

Externals
: RICH

Arguments
: K,J,I: cell indices at W-node

Description
: The function calculates the factor f_m defined by (3.1.97) taking account of the limiting condition (3.1.99).

Remarks

- The function is called when IOPTK $= 1$ and ITFORM $= 2, 3, 4, 5$ or when IOPTK $= 2$ and ISTPAR $= 2$.
- The vertical index K must be between 2 and NZ.

GVEDMA

File
: *veddy_in.f*

Type
: real function

Purpose
: evaluate $g_m(Ri)$ in the Munk-Anderson turbulence formulation

Calling program
: FNSHB, FNTURF, VEDDY1

Externals
: RICH

Arguments
: K,J,I: cell indices at W-node

Description
: The function calculates the factor g_m defined by (3.1.98) taking account of the limiting condition (3.1.99).

Remarks

- The function is called when IOPTK $= 1$ and ITFORM $= 2, 3, 4, 5$ or when IOPTK $= 2$ and ISTPAR $= 2$.
- The vertical index K must be between 2 and NZ.

PRODUC

File
: *turben.f*

Type
: subroutine

Purpose
: evaluate the shear and buoyancy terms on the right hand side of the turbulence equation (3.1.114)

Calling program
: TURBEN

Externals
: BUOFR2, SHFR2

Arguments
: none

Description

- Evaluate the shear production term $\nu_T M^2$ and store the result into the array SHPROD.
- Evaluate the buoyancy source or sink term $-\lambda_T N^2$ and store the result into the array BUPROD.

Remarks

BUPROD is only called if IOPTK $= 2$.

RICH

File

veddy_in.f

Type

real function

Purpose

evaluate the Richardson number Ri

Calling program

FVEDMA, GVEDMA, VEDDY1

Externals

BUOFR2, SHFR2

Arguments

K,J,I: cell indices at W-node

Description

The Richardson number is defined by (3.1.89). An upper limit is imposed to prevent floating overflows when M^2 is small.

Remarks

The vertical index K must be between 2 and NZ.

SHFR2

File
: *veddy_in.f*

Type
: real function

Purpose
: calculate the squared shear frequency M^2 (s^{-2})

Calling program
: FNTURF, PRODUC, RICH

Externals
: none

Arguments
: K,J,I: cell indices at W-node

Description
: Evaluates the squared shear frequency M^2, defined by (3.1.91), using the discretised form (3.4.298).

Remarks
: The vertical index K must be between 2 and NZ.

TKLENG

File
: *tkleng.f*

Type
: subroutine

Purpose
: solve the kl-equation

Calling program
: VEDDY2

Externals
: TRANSPW

Arguments

none

Local variables

TKEZL: old value of kl

WPF: wall proximity function $\tilde{W}$ given by (3.1.126)

ZL1: bottom length scale l_1 defined by (3.1.130)

ZL2: surface length scale l_2 defined by (3.1.130)

Description

The routine solves equation (3.1.125) for the product kl (see Figure 5.1.3 for a general flow chart).

- Store old values of kl.
- Apply the surface and bottom boundary conditions (3.1.180) and (3.1.190). The results are stored into temporary arrays used as arguments in the call to TRANSPW.
- Evaluate the source and sink terms appearing on the right of (3.1.125) using the discretisations (3.4.302) and (3.4.305).
- Solve (3.1.125) by calling TRANSPW.
- Update the mixing length by dividing the updated value of kl by k.

Remarks

The kl-equation is only solved if IOPTK $= 2$, NTRANS $= 2$ and ITCPAR $= 1$.

TLENDIS

File

tkleng.f

Type

subroutine

Purpose

evaluate mixing length l ($k - \varepsilon$ theory)

Calling program

VEDDY2

Externals

none

Arguments

none

Description

Evaluate l as a function of k and ε using (3.1.116).

Remarks

- Although the value of l is not required for the internal calculations in the program when the $k - \varepsilon$ theory is selected, it may be a useful quantity for user-defined output.
- The mixing length is only evaluated when IOPTK $= 2$.

TLENG

File

tleng.f

Type

subroutine

Purpose

calculate l using one of the available algebraic mixing length formulations

Calling program

INCTUR, VEDDY2

Externals

none

Arguments

none

Local variables

ZLASYM: the asymptotic mixing length l_a, defined by (3.1.135)

ZL1: bottom length scale defined by (3.1.130) or (3.1.133)

ZL2: surface length scale defined by (3.1.130)

Description

The formulation depends on the value of the switch ILENG.

1: parabolic law (3.1.131)

2: “quasi-parabolic” law (3.1.132)

3: “Xing” formulation (3.1.133)

4: “Blackadar” formulation (3.1.134)–(3.1.135)

Remarks

TLENG is only called in the main program when IOPTK $= 2$ and NTRANS $= 0, 1$.

TURBEN

File

turben.f

Type

subroutine

Purpose

solve the turbulence energy equation

Calling program

VEDDY2

Externals

PRODUC, TRANSPW

Arguments

none

Description

The routine solves equation (3.1.114) for the turbulence energy k (see Figure 5.1.3 for a general flow chart).

- Store old values of k (to be used eventually in subroutines TKLENG or DISSIP).
- Apply the surface and bottom boundary conditions (3.1.180) and (3.1.190). The results are stored into temporary arrays used as arguments in the call to TRANSPW.
- Evaluate the source and sink terms on the right of (3.1.114) by calling PRODUC and using the discretisations (3.4.301) and (3.4.304).
- Solve (3.1.114) by calling TRANSPW.
- To avoid floating overflow in subsequent calculations k is set to zero if k is below a small lower limit (taken to be as 10^{-20} J/kg).

Remarks

The k-equation is solved when IOPTK $= 2$ and NTRANS $= 1, 2$.

VEDDY1

File

veddy.f

Type

subroutine

Purpose

evaluate eddy coefficients (algebraic formulations)

Calling program

MAINPROG

Externals

FVEDMA, GVEDMA, RICH

Arguments

none

Description

The routine calculates the eddy viscosity and diffusion coefficients ν_T and λ_T using one of the algebraic formulations described in Section III-1.2.1.

- Initialise the array $\Phi(\sigma)$ on first call. The result is stored in the local array PHIZ.
- Evaluation of ν_T and λ_T depending on the value of the switch ITFORM.
  1 : Pacanowski-Philander formulation (3.1.86)–(3.1.88) including the limiting condition (3.1.92).
  2 : Munk-Anderson relations (3.1.95)–(3.1.98) including the limiting condition (3.1.99).
  3 : Flow-dependent formulation (3.1.101)–(3.1.102) with α determined by (3.1.108).
  4 : Flow-dependent formulation (3.1.101)–(3.1.102) with α determined by (3.1.109).
  5 : Flow-dependent formulation (3.1.101)–(3.1.102) with α determined by (3.1.110).

Remarks

VEDDY1 is called at each 3-D time step if IOPTK $= 1$.

VEDDY2

File

veddy.f

Type

subroutine

Purpose

evaluate turbulence arrays and vertical diffusion coefficients (turbulence closure schemes)

Calling program

MAINPROG

Externals

BUOFR2, DISLEN, DISMIN, DISSIP, FNSHB, FNSMU, FNTURF, TKLENG, TLENDIS, TLENG, TURBEN, ZLMAX

Arguments

none

Local variables

GHAN: stability parameter G_h or α_N defined by either (3.1.119) or (3.1.121)

Description

The routine determines the turbulence parameters k, l and ε, the eddy coefficients ν_T, λ_T and the diffusion coefficients appearing in the k-, kl- and ε-equation using one of the turbulence closures described in Section III-1.2.2. A flow chart of VEDDY2 is presented in Figure 5.1.5.

- Turbulent energy is determined by calling FNTURF (NTRANS $= 0$) or TURBEN (NTRANS $= 1$ or 2).
- The mixing length is obtained by calling TLENG (NTRANS $= 0$ or 1), TKLENG (NTRANS $= 2$, ITCPAR $= 1$) or TLENDIS (NTRANS $= 2$, ITCPAR $= 2$).
- Dissipation rate is obtained by calling DISLEN (NTRANS $= 0, 1$ or NTRANS $= 2$, ITCPAR $= 1$) or DISSIP (NTRANS $= 2$, ITCPAR $= 2$).
- If ILIM $= 1$ and NTRANS > 0, limiting conditions are applied for k, l if ITCPAR $= 1$ and for k, ε if ITCPAR $= 2$.
- The updated values of k, l and ε are stored into the arrays TKEW, ZLW and DISSW.
- The stability coefficients S_m, S_h or S_u, S_b are determined by calling FNSMU and FNSHB. The results are stored into the arrays SMU (S_m or S_u) and SHB (S_h or S_b).
- The eddy coefficients ν_T and λ_T are evaluated either using (3.1.115) for the $k-l$ (ITCPAR $= 1$) formulation or (3.1.117) for the $k-\varepsilon$ (ITCPAR $= 2$) formulation. The results are stored into the arrays VEDDYV and VEDDYD.
- The vertical diffusion coefficient appearing in the k- and kl-equation is stored into VEDDYK, the one appearing in the ε-equation into VEDDYE.

Remarks

- The stability parameter (G_h or α_N) has a lower and an upper limit. The former applies for unstable stratification and is given by (3.1.146). The latter occurs for strong stable stratification and is given by (3.1.142) if ILIM = 1, or is set to a high value in INICON if ILIM = 0 to prevent machine overflow in the calculation of the stability coefficients.
- VEDDY2 is called at each 3-D time step when IOPTK = 2.

ZLMAX

File

veddy_in.f

Type

real function

Purpose

evaluation of the maximum mixing length ($k - l$ theory)

Calling program

VEDDY2

Externals

BUOFR2

Arguments

K,J,I: cell indices at W-node

Description

Evaluates the maximum mixing length using (3.1.141) with c obtained from (3.1.142).

Remarks

- ZLMAX is only called when IOPTK = 2, NTRANS $\neq$ 0, ILIM = 1 and ITCPAR = 1.
- The vertical index K must be between 2 and NZ.

1.5 Biology and sediments

AMMOXID

File

biolgy_in.f

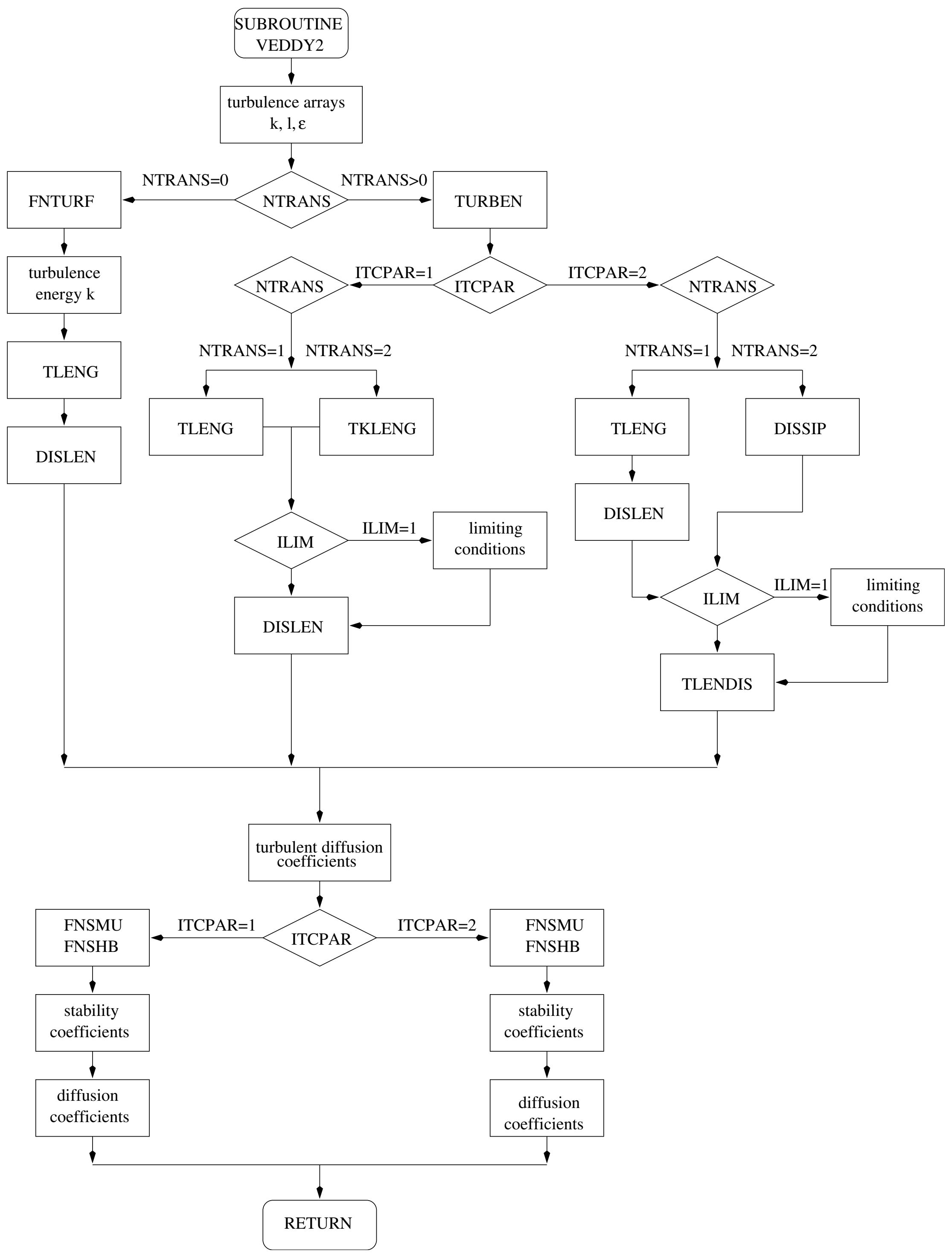


Figure 5.1.5: Flow chart of subroutine VEDDY2.

Type

real function

Purpose

calculate the ammonium nitrification rate ${}^{NH}r$ (s^{-1})

Calling program

BIOLGY

Externals

none

Arguments

TF: temperature growth factor $f(T)$

XO: dissolved oxygen concentration (mmol O m^{-3})

XNHS: ammonium concentration (mmol N m^{-3})

Local variables

P2NHSMIN: minimum value for ammonium (see below) (mmol N m^{-3})

Description

The oxidation rate of ammonium to nitrite is calculated using (3.2.69).

Remarks

- Nitrification appears as a source term in the nitrate and as a sink term in the ammonium and oxygen transport equations.
- The nitrification rate is set to zero when ammonium drops below the small (lower) limit P2NHSMIN.
- Note that days are converted into seconds.

BIOLGY

File

biolgy.f

Type

subroutine

Purpose

update the concentrations of all biological state variables

Calling program

MAINPROG

Externals

AMMOXID, BIOPT, BSINKING, DSINKING, GROWTH, IDATES, QDET, QUOTA, RESPC, RESPN, TEMPCH, TRANSPC, UPTAKENH, UPTAKENO, ZERO

Arguments

none

Local variables

BNHS2: assumed ammonium concentration in the sediment which is set to ${}^{NH}S_b^0$ (mmol N m^{-3})

BNOS2: assumed nitrate concentration in the sediment which is set to ${}^{NO}S_b^0$ (mmol N m^{-3})

BTRAN: transfer coefficient $F_{br}H$ for the benthic flux (m/s)

CR: detrital carbon remineralisation rate ${}^{C}r$ (s^{-1})

GR: microplankton growth rate (s^{-1})

GRAZ: grazing pressure (s^{-1})

OAIR: air concentration $O_{air}(T_s)$ from (3.2.82) for the surface oxygen flux (mmol O m^{-3})

PISTON: transfer coefficient C_O from (3.2.81) for the surface oxygen flux (m/s)

Q: nitrogen quota

QD: detrital nitrogen quota

QMOD: temporary variable representing $Q - q_h\eta$

RMR: detrital nitrogen remineralisation rate ${}^{M}r$ (s^{-1})

RNH: nitrification rate (s^{-1})

TF: temperature growth factor $f(T)$ from (3.2.21)

UNH: ammonium uptake rate ${}^{NH}u$ (s^{-1})

UNO: nitrate uptake rate ${}^{NO}u$ (s^{-1})

WIND: wind speed $|U_{10}|$ (m/s)

WSB: total vertical advective velocity for microplankton (m/s)

WSC: total vertical velocity for detritus (m/s)

Description

The routine solves the biological transport equations along the lines shown in Figure 5.1.3 and updates the chlorophyll concentration.

- Initialise the source terms and vertical velocities to zero.

- Evaluate PAR by calling BIOPT.
- Evaluate the explicit (source/sink) terms (as given by the $\beta(\psi)$ in the general transport equation (3.2.3)–(3.2.4)). Function calls are given in parentheses:
  - update grazing pressure and evaluate temperature growth factor (TEMPCH)
  - nitrogen quota Q (QUOTA) and detrital nitrogen quota ${}^{M}q^{C}$ (QDET)
  - grazing pressure terms
  - microplankton growth terms (GROWTH)
  - detrital carbon remineralisation rate terms (RESPC)
  - detrital nitrogen remineralisation rate terms (RESPN)
  - nitrate uptake terms (UPTAKENO)
  - ammonium uptake terms (UPTAKENH)
  - ammonium nitrification terms (AMMOXID)
- Surface/bottom boundary conditions:
  - The surface boundary condition (3.2.80) for oxygen has the form given by the general expression (3.1.181). The transfer coefficient C_O and the external value O_{air}, given by respectively (3.2.81) and (3.2.82) are stored into local arrays used as arguments in the call to TRANSPC.
  - The bottom boundary condition (3.2.78) for nitrate and (3.2.79) for ammonium have the form given by the general expression (3.1.191). The transfer coefficient $F_{br}H$ and the external values ${}^{NO}S_b^0$, ${}^{NH}S_b^0$ are stored into local arrays used as arguments in the call to TRANSPC.
  - All other surface and bottom fluxes are set to zero. The corresponding arguments in the call to TRANSPC are the zero arrays (ZEROS1, ..., ZEROS6).
- The (total) vertical velocity is obtained using the following steps:
  - Evaluate the vertical sinking rate for microplankton ($w_s^B = w_s^N$), as given by (3.2.71), and detritus ($w_{s_{det}}$) by calling BSINKING and DSINKING.
  - Add the physical (transformed) vertical velocity $J\tilde{w}$.
- Solve the transport equations for microplankton carbon and nitrogen, detrital nitrogen and carbon, ammonium, nitrate, oxygen and zooplankton nitrogen by calling TRANSPC for each state variable.
- Update the chlorophyll concentration using (3.2.12).

Remarks

- Limiting conditions are imposed for the uptake of nitrate and ammonium and for nitrification to prevent negative concentrations. See Section III-4.4.9b for details.
- If horizontal advection is disabled (IADVS $=0$), vertical advection remains enabled if IADVWB >0 but without the physical velocity $J\tilde{w}$. On the other hand, the biological sinking rates (w_s^{ψ}) are set to zero if IFLUFF $=0$.

- BIOLGY is called at each 3-D time step if IOPTB = 1.

BIOPT

File
: *biolgy.f*

Type
: subroutine

Purpose
: evaluate the diffuse attenuation coefficient k_d for PAR, and PAR

Calling program
: BIOLGY

Externals
: none

Arguments
: none

Local variables

RADLO:	value of $I^w_{p;k}$
RADUP:	value of $I^w_{p;k+1}$
R2EXP:	the factor R^*_2 obtained from (3.4.288)

Description

- Update the diffuse attenuation coefficient k_d for PAR using (3.4.290).
- PAR is first computed at the W-nodes above and below the cell centre using the iterative scheme (3.4.291).
- The value of PAR at the cell centre is obtained by taking the average of the two values and a further average over the past 24 hours (see equations (3.4.292)–(3.4.293)).

Remarks
: BIOPT is called at each 3-D time step if IOPTB = 1.

BSINKING

File
: *biolgy_in.f*

Type
: real function

Purpose
: evaluate the vertical sinking rate for microplankton (m/s)

Calling program
: BIOLGY, FLUFF

Externals
: none

Arguments
: XQ: nitrogen quota Q

Description
: Calculates the microplankton sinking velocity $w_s^B = w_s^N$ as a function of the nitrogen quota Q with the aid of (3.2.71).

Remarks

- BSINKING is only called when IFLUFF = 1.
- Note that days are converted into seconds.

DSINKING

File
: *biolgy_in.f*

Type
: real function

Purpose
: evaluate the vertical sinking rate for detritus (m/s)

Calling program
: BIOLGY, FLUFF

Externals
: none

Arguments
: none

Description
: The vertical sinking velocity for detritus is given by the uniform value $w_{s_{det}}$.

Remarks

- DSINKING is only called when IFLUFF = 1.
- Note that days are converted into seconds.

FLUFF

File
: *sedeul.f*

Type
: subroutine

Purpose
: update the fluff layer amounts and the bottom concentrations in the water column

Calling program
: SEDEUL

Externals
: BSINKING, DSINKING, QUOTA, RESPC, RESPN, SSINKING, TEMPCH

Arguments
: none

Local variables

BST100: bottom stress (normalised by ρ_0) at 1 m above the sea bed (m^2/s^2)

CRBOT: amount of detrital carbon lost by remineralisation during the previous 3-D time step (mmol C m^{-2} s^{-1})

DCONV: amount of microplankton (carbon or nitrogen) converted into detritus during the previous 3-D time step

DFSED: net amount of material lost from the fluff layer to the water column due to deposition and resuspension during the previous 3-D time step and for the different fractions (amount m^{-2})

EROS: resuspension rate $E(\psi)$ for the different fractions (amount m^{-2} s^{-1})

Q: nitrogen quota

QDBOT: detrital nitrogen quota

RMRBOT: amount of detrital nitrogen lost by remineralisation during the previous 3-D time step (mmol N m^{-2} s^{-1})

TF: temperature growth factor $f(T)$

U100: u-current at 1 m height above the sea bed (m/s)

V100: v-current at 1 m height above the sea bed (m/s)

WCBOT: bottom value of the detrital sinking rate (m/s)

WC1BOT: bottom value of the iSPM sinking rate (m/s)

WSBOT: bottom value of the microplankton sinking rate (m/s)

Description

The module evaluates all fluff layer processes and updates the bottom concentrations of inorganic sediment and biological concentrations.

- Evaluate the bottom stress at 1 m height above the sea bed by interpolating the current components at the reference height.
- Store the bottom concentrations of iSPM, microplankton and detrital carbon and nitrogen, and the bottom sinking rates.
- Calculate the resuspension rate $E(\psi)$ for the five fluff layer amounts (iSPM, microplankton and detrital carbon, microplankton and detrital nitrogen) using the formulation described in Section III-2.8.3. The results are stored into the array EROS.
- Evaluate the net amount of material lost to the fluff layer during the previous (3-D) time step. Gains arise from the deposition due to the sinking of water column material, losses from the resuspension of fluff layer amounts into the water column. The net amounts are stored into the local array DFSED. To prevent negative amounts in the fluff layer, or negative concentrations at the bottom grid cell, DFSED is limited by the conditions (3.4.280).
- Update the water column concentrations (at the bottom grid cell) of the five fluff layer quantities.
- Update the fluff layer amounts:
  - Substract the net losses due to deposition and resuspension.
  - Convert microplankton into detritus by assuming a 10% daily conversion rate.
  - Substract losses of microplankton and detritus to the consolidated sediment assuming a 10% loss per day.

- The amounts of detrital carbon and nitrogen in the fluff layer are further reduced by the process of remineralisation. The corresponding bottom concentrations of oxygen and ammonium are updated. For each process it is checked that the new amounts in the fluff do not become negative.

Remarks

- For more details about the numerical procedures see Section III-4.4.9c.
- FLUFF is called at each 3-D time step if IOPTS = 1.
- The biological part of the fluff layer is only executed when IFLUFF = 1.

GROWTH

File

biolgy_in.f

Type

real function

Purpose

evaluate the microplankton growth rate μ (s^{-1})

Calling program

BIOLGY

Externals

none

Arguments

TF: temperature growth factor $f(T)$

XQ: nitrogen quota Q

QMOD: the factor $Q - Q_h\eta$ appearing in (3.2.29)

XPAR: PAR ($\mathrm{W/m^2}$)

Local variables

CHLCRATIO: ratio χ of chlorophyll to microplankton carbon (mg Chl (mmol $C)^{-1}$)

GLIGHT: light-controlled growth rate $\mu_1(I_p)$ (s^{-1})

GNUTRIENT: nutrient-controlled growth rate $\mu_2(Q)$ (s^{-1})

Description

- Evaluate the nutrient-controlled growth rate $\mu_2(Q)$ using (3.2.39) and (3.2.41). The maximum growth rate μ_{max}, given by the parameter GRMAX, is previously defined in INIBIO.
- Evaluate the light-controlled growth rate $\mu_1(I_p)$ using (3.2.28) with χ defined by (3.2.29), r_0 by (3.2.30) and b by (3.2.31). The respiration rate r_0 and slope b represented by RB0 and RMU, are previously defined in INIBIO. Note that $\mu_1(I_p)$ may become negative, when microplankton respiration exceeds autotroph photosynthesis.
- Determine the microplankton growth rate as the minimum value of μ_1 and μ_2 (see equation (3.2.22)).

Remarks

- The microplankton growth rate term appears as a source or sink term in the microplankton carbon and oxygen transport equations.
- Note that days are converted into seconds.

QDET

File

biolgy_in.f

Type

real function

Purpose

evaluate the detrital nitrogen quota ${}^{M}q^{C}$

Calling program

BIOLGY

Externals

none

Arguments

K,J,I: cell indices at cell centre

Description

The detrital nitrogen quota ${}^{M}q^{C}$ is defined by the ratio of detrital nitrogen to detrital carbon (see equation (3.2.64)). A minimum value of ${}^{M}q^{C}_{min}$ is imposed.

Remarks

$^Mq^C$ is used for the evaluation of the detrital carbon and detrital nitrogen remineralisation rates (functions RESPC and RESPN).

QUOTA

File

biolgy_in.f

Type

real function

Purpose

evaluate the nitrogen quota Q

Calling program

BIOLGY, FLUFF

Externals

none

Arguments

K,J,I: cell indices at cell centre

Description

The nitrogen quota Q is defined by the ratio of microplankton nitrogen to microplankton carbon (see equation (3.2.11)). A minimum value Q_{min} is imposed. This is reprented by the parameter QMIN, previously defined in INIBIO.

Remarks

Q is used for the evaluation of the microplankton (nutrient) growth rate in GROWTH, the nitrate and ammonium uptake rates in UPTAKENO and UPTAKENH and the microplankton sinking rate in BSINKING.

RESPC

File

biolgy_in.f

Type

real function

Purpose

calculate the detrital carbon remineralisation rate ${}^{C}r$ (s^{-1})

Calling program

BIOLGY, FLUFF

Externals

none

Arguments

TF: temperature growth factor $f(T)$

XQDET: detrital nitrogen quota ${}^{M}q^{C}$

XO: dissolved oxygen concentration (mmol O m^{-3})

Description

The detrital carbon remineralisation rate ${}^{C}r$ is defined by (3.2.62) and (3.2.65). ${}^{C}r$ is set to zero if O drops below a small lower limit OMIN (defined in INIBIO).

Remarks

- Detrital carbon remineralisation appears as a sink term in the detrital carbon and oxygen transport equations.
- Note that days are converted into seconds.

RESPN

File

biolgy_in.f

Type

real function

Purpose

calculate the detrital nitrogen remineralisation rate ${}^{M}r$ (s^{-1})

Calling program

BIOLGY, FLUFF

Externals

none

Arguments

TF: temperature growth factor $f(T)$

XQDET: detrital nitrogen quota ${}^{M}q^{C}$

Description

The detrital nitrogen remineralisation rate is defined by (3.2.63).

Remarks

- Detrital nitrogen remineralisation appears as a sink term in the detrital nitrogen and as a source term in the ammonium transport equations.
- Note that days are converted into seconds.

SEDEUL

File

sedeul.f

Type

subroutine

Purpose

solve the transport equation for inorganic sediment A

Calling program

MAINPROG

Externals

FLUFF, SSINKING, TRANSPC, ZERO

Arguments

none

Description

The routine updates the iSPM concentration A as shown in Figure 5.1.3.

- Set all source/sink terms to zero on first call.
- Surface and bottom fluxes are set to zero (represented by the zero argument arrays ZEROS1, ..., ZEROS6 in the call to TRANSPC).
- Evaluate the (total) vertical velocity by calling SSINKING and add the physical (transformed) vertical velocity $J\tilde{w}$.
- Solve the iSPM transport equation by calling TRANSPC.

- Evaluate the fluff layer processes for iSPM and biology by calling FLUFF.

Remarks

- If horizontal advection is disabled (IADVS = 0), vertical advection remains enabled if IADVWB > 0 but without the physical velocity $J\tilde{w}$.
- SEDEUL is called at each 3-D time step if IOPTS $= 1$.
- Since FLUFF is called from SEDEUL, the (biological) fluff layer processes can only be evaluated when IOPTS $= 1$.

SSINKING

File
: *biolgy_in.f*

Type
: real function

Purpose
: evaluate the vertical sinking rate for inorganic sediment (m/s)

Calling program
: FLUFF, SEDEUL

Externals
: none

Arguments
: none

Description
: The vertical sinking velocity for inorganic sediment is given by the uniform value w_s^A.

TEMPCH

File
: *biolgy_in.f*

Type
: real function

Purpose
: evaluate the temperature growth factor $f(T)$

Calling program
: BIOLGY, FLUFF

Externals
: none

Arguments
: TEMP: temperature (^{0}C)

Description
: The temperature growth factor $f(T)$ is defined by (3.2.21).

Remarks
: The factor $f(T)$ is used in the expressions for the microplankton growth rate (GROWTH), the nitrate and ammonium uptake rates (UPTAKENO, UPTAKENH), the detrital carbon and nitrogen remineralisation rates (RESPC, RESPN), and the ammonium nitrification rate (AMMOXID).

UPTAKENH

File
: *biolgy_in.f*

Type
: real function

Purpose
: evaluate the ammonium uptake rate ${}^{NH}u$ (s^{-1})

Calling program
: BIOLGY

Externals
: none

Arguments
: TF: temperature growth factor $f(T)$

: QMOD: the factor $Q - Q_h\eta$ in (3.2.52)

: XNHS: ammonium concentration ${}^{NH}S$ (mmol N m^{-3})

Local variables

P2NHSMIN: minimum value for ammonium (see below) (mmol N m^{-3})

Description

The ammonium uptake rate ${}^{NH}u$ is defined by (3.2.55) with $f({}^{NH}S)$ and $f_{in2}(Q)$ given by respectively (3.2.51) and (3.2.52). Note that ${}^{NH}u$ can become negative when QMOD $>$ QMAXMOD. The factor $Q_{max} - q_h\eta$ and the maximum uptake rate ${}^{NH}u_{max}$ are represented by the parameters QMAXMOD and UMAXNH, defined in INIBIO.

Remarks

- The factor $f({}^{NH}S)$ is set to zero if ammonium drops below the (small) lower limit P2NHSMIN.
- The ammonium uptake term appears as a source (sink) term in the microplankton nitrogen and as a sink (source) term in the ammonium transport equation.
- Note that days are converted into seconds.

UPTAKENO

File

biolgy_in.f

Type

real function

Purpose

evaluate the nitrate uptake rate ${}^{NO}u$ (s^{-1})

Calling program

BIOLGY

Externals

none

Arguments

TF: temperature growth factor $f(T)$

QMOD: the factor $Q - q_h\eta$ in (3.2.47)

XNOS: nitrate concentration ${}^{NO}S$ (mmol N m^{-3})

XNHS: ammonium concentration ${}^{NH}S$ (mmol N m^{-3})

Local variables

P2NHSMIN:	minimum value for ammonium (see below) (mmol N m^{-3})
P2NOSMIN:	minimum value for nitrate (see below) (mmol N m^{-3})

Description

The nitrate uptake rate ${}^{NO}u$ is defined by (3.2.43) with $f({}^{NO}S)$, $f_{in}({}^{NH}S)$ and $f_{in1}(Q)$ defined by (3.2.45)–(3.2.47). The parameter $Q_{max} - q_h\eta$ and the maximum uptake rate ${}^{NO}u_{max}$ are represented by the parameters QMAXMOD and UMAXNO defined in INIBIO.

Remarks

- Ammonium inhibition of nitrate uptake (given by $f_{in}({}^{NH}S)$) only occurs when XNHS exceeds P2NHSMIN.
- Nitrate uptake is set to zero if XNOS is lower than P2NOSMIN.
- The nitrate uptake terms appears as a source term in the microplankton nitrogen and oxygen transport equations and as a sink term in the nitrate transport equation.
- Note that days are converted into seconds.

1.6 Contaminants

CONSPM

File

sedlag.f

Type

subroutine

Purpose

calculate contaminant concentration (g/m^3) in the Lagrangian particle module

Calling program

SEDLAG

Externals

ZERO

Arguments

none

Description

After each update of the particle positions in the Lagrangian module, the program converts the particle distribution into a contaminant concentration (g/m^3). The result is stored in the array SPMCON.

Remarks

- The array SPMCON has no relevance for the program itself but can be considered as a useful quantity for user-defined output since it is the equivalent of the CONCN array in the Eulerian contaminant module.
- A different scale factor has to be inserted into the code if a unit of mass is selected by the user, differing from the default one (ton).
- CONSPM is called from SEDLAG after each particle update.

MASS

File

mass.f

Type

subroutine

Purpose

solve the transport equation(s) for contaminant(s) (Eulerian module)

Calling program

MAINPROG

Externals

TRANSPC, ZERO

Arguments

none

Description

The module updates the contaminant concentrations by solving equations (3.3.9) for the distributions C_i. For a general flow chart see Figure 5.1.3.

- Set all source/sink terms to zero on first call.
- Surface/bottom fluxes are set to zero (represented by the zero argument arrays ZEROS1, ..., ZEROS6 in the call to TRANSPC).

- Disable vertical and horizontal diffusion if IOPTC = 1. The first is performed by setting the vertical diffusion coefficient equal to zero (represented by the zero argument array VEDCON in the call to TRANSPC), the second by using a zero value for the argument "JDIFS" in the call to TRANSPC which acts as a switch to disable/enable horizontal diffusion in the TRANSPC routine (see the description below of TRANSPC).
- Vertical and horizontal diffusion are enabled if IOPTC = 2 (and IODIF > 0).
- Update the concentrations for each contaminant array (1 to NCONC):
  - store the old values in a temporary array
  - update the distribution by calling TRANSPC using the temporary array as argument
  - store the updated values in the array CONCN

Remarks

- Note that, if IOPTC = 2, horizontal diffusion is only enabled when IODIF = 1 or 2.
- MASS is called at each 3-D time step if IOPTC = 1 or 2.

MCOSB

File

sedlag.f

Type

subroutine

Purpose

generate particle input at the open sea boundaries (Lagrangian module)

Calling program

SEDLAG

Externals

ERROR, RANDOM

Arguments

none

Local variables

LNEW: number of particles created at an open sea boundary cell by inflow during the previous (3-D) time step

QSTVOL: amount of suspended material transported into an open boundary cell during the previous time step (ton)

VOLTRP: amount of volume transported in or out of an open boundary cell during the previous time step (m^3)

XRAN: random number between 0 and 1

Description

New particles are generated when an inflow condition occurs at an open sea boundary, provided that an ambient (external) concentration (ton/m^3) has been initialised by the user via the arrays QSTOBU and QSTOBV (at open sea boundaries). The calculations are firstly performed at U-open sea boundaries (NPIX values of 2):

- Calculate the amount of volume ΔV_{tr} transported in or out of a grid cell (k,j,i) using (3.4.398):

$$\Delta V_{tr} = \pm u_{kj,i:i+1} H_{ji} \Delta x_{2;j} \Delta\sigma_k \Delta t_{3D} \tag{5.1.2}$$

  where the +(-) sign applies at western (eastern) boundaries so that ΔV_{tr} is positive for inflow and negative for outflow.

- If $\Delta V_{tr} \leq 0$, no particles are generated.
- If $\Delta V_{tr} > 0$, determine the amount of suspended material M_{tr} (ton) transported into the cell by multiplying ΔV_{tr} with the ambient concentration ρ^e_{spm}.
- The number of newly created particles is calculated by adding a remaining mass fraction RMASSU (from the previous time) to QSTVOL and dividing through the particle mass STSSOL (see equation (3.4.400)). The result is rounded to the nearest integer. After the division a fraction (lower than STSSOL) remains and is stored in the array RMASSU (see equation (3.4.401)) which will be used at the next time step when inflow occurs.
- An error occurs and program execution stops when the total number of particles inside the computational domain exceeds the maximum allowed value MAXNOP.
- Evaluate the coordinates of each new particle (see equations (3.4.402)–(3.4.404)):
  - The Y- and Z-coordinates within the cell (arrays YT and ZT) are set at random.
  - The X-coordinate is first set at random within an external grid cell (of the same dimensions) adjacent to the open boundary cell. An advective distance of $\pm u_{kj,i:i+1}\Delta t_{3D}$ (+ at western, - at eastern boundaries) is then applied yielding the particle's X-coordinate inside the domain after adding $\mp\Delta x_{1;ji}$.
  - The particle labels are given by the user-defined array LSTOBU. Note that different values can be assigned at different locations.

- The particle masses (m_{sea}) are set to the user-defined parameter STSSOL.
- The total mass of all created particles are stored into the array STIN.
- The total number of particles inside the computational domain (NUMP) is updated.

A similar procedure is taken at the V-open sea boundaries.

Remarks

- The numerical procedures at open sea boundaries are further discussed in Section III-4.8.5c.
- The arrays QSTOBU and QSTOBV are set to zero (no input) by default. The routine MCOSB is only effectively used by the program if non-zero values are supplied for these two arrays.
- It is assumed in the module that the distance traveled by a particle during a 3-D time step is smaller than one grid distance, so that the criterion (3.4.8) applies. No program error occurs, if these criteria are not satisfied, but the results may become less realistic.
- The arrays RMASSU and RMASSV are updated at each time step and are saved at the end of the program (*pou*-file) for an eventual restart.
- The array STIN accumulates the amount of suspended matter entering the domain and has no further relevance for the program, but may be a useful quantity for user-defined output. Its value is saved at the end of the program (*pou*-file) for an eventual restart.
- The program stops when the total number of particles exceeds MAXNOP. Since MAXNOP is used for the dimensioning of most particle arrays, its value cannot be changed during the program's execution. It is therefore recommended to assign a sufficiently large value for MAXNOP (file *param.inc*) if open boundary input is supplied by the user.
- Note that the horizontal particle coordinates inside the routine MCOSB are always expressed in meters for computational efficiency. In the case of spherical coordinates, the necessary conversions from and to degrees, given by (3.4.366)–(3.4.367), are performed in SEDLAG.

MCROB

File

sedlag.f

Type

subroutine

Purpose

generate particle input at river boundaries (Lagrangian module)

Calling program

SEDLAG

Externals

ERROR, RANDOM

Arguments

none

Local variables

LNEW: number of particles created at a river open boundary cell during the previous (3-D) time step

QSTDT: amount of suspended material discharged into a river open boundary cell during the previous (3-D) time step (ton)

XRAN: random number between 0 and 1

Description

New particles are generated by the discharge of suspended matter at river boundaries, provided that the arrays QSTOBU and QSTOBV, representing (at river boundaries) the discharge rate (ton/s) have been initialised by the user. The calculations are firstly performed at U-open boundaries (NPIX value of 3):

- Calculate the amount of suspended material QSTDT discharged into a boundary cell during the last 3-D time step using (3.4.405).
- The number of newly created particles is calculated by adding a remaining mass fraction RMASSU (from the previous time) to QSTDT and dividing through the particle mass STLSOL (see equation (3.4.406)). The result is rounded to the nearest integer. After division a fraction (lower than STLSOL) remains which is stored in the array RMASSU (see equation (3.4.407)) and will be used at the next time step.
- An error occurs and program execution stops when the total number of particles inside the computational domain exceeds the maximum allowed value MAXNOP.
- The X-, Y- and Z-coordinates are set at random within the boundary cell (see equation (3.4.408)).
- The particle labels are given by the user-defined array LSTOBU. Note that different values can be assigned at different locations.
- The particle masses (m_{riv}) are set to the user-defined parameter STLSOL.
- The total mass of all created particles are stored into the array STIN.
- The total number of particles inside the computational domain (NUMP) is updated.

The same procedure is taken at V-open (river) boundaries.

Remarks

- The numerical procedures at river boundaries are further discussed in Section III-4.8.5d.
- The arrays QSTOBU and QSTOBV are set to zero (no discharge) by default. The routine MCOSB is only effectively used by the program if non-zero values are supplied for these two arrays.
- The arrays RMASSU and RMASSV are updated at each time step and are saved at the end of the program (*pou*-file) for an eventual restart.
- The array STIN accumulates the amount of suspended matter entering the computational domain and has no further relevance for the program, but may be a useful quantity for user-defined output. Its value is saved at the end of the program (*pou*-file) for an eventual restart.
- The program stops when the total number of particles exceeds MAXNOP. Since MAXNOP is used for the dimensioning of most particle arrays, its value cannot be changed during the program's execution. It is therefore recommended to assign a sufficiently large value for MAXNOP (file *param.inc*) if open boundary input is supplied by the user.
- Note that the horizontal particle coordinates inside the routine MCOSB are always expressed in meters for computational efficiency. In the case of spherical coordinates, the necessary conversions from and to degrees, given by (3.4.366)–(3.4.367), are performed in SEDLAG.

MC3D

File

sedlag.f

Type

subroutine

Purpose

update the particle positions by advection and diffusion inside the computational domain

Calling program

SEDLAG

Externals

RANDOM

Arguments

none

Local variables

IBOT: counter indicating whether the particle is advected across the bottom cell face during Δt_{rest}

IEAST: counter indicating whether the particle is advected across the eastern cell face during Δt_{rest}

INORT: counter indicating whether the particle is advected across the northern cell face during Δt_{rest}

IOLD: particle cell coordinate i_n^n (old value) or i_n^a (after applying advection)

IRND: number of boundary faces crossed by the particle during Δt_{rest}

ISOUT: counter indicating whether the particle is advected across the southern cell face during Δt_{rest}

ITOP: counter indicating whether the particle is advected across the top cell face during Δt_{rest}

IWEST: counter indicating whether the particle is advected across the western cell face during Δt_{rest}

JOLD: particle cell coordinate j_n^n (old value) or j_n^a (after applying advection)

KOLD: particle cell coordinate k_n^n (old value) or k_n^a (after applying advection)

DTMIN: if a particle crosses at least one boundary face during Δt_{rest}, Δt_{min} is defined as the time (s) required for the first crossing of a boundary face

DTREST: time step Δt_{rest} (s) for advection which is first set to the 3-D time step Δt_{3D}; an amount Δt_{min} is substracted after each iteration

DTX: Δt^x_{min} as defined by (3.4.381)

DTY: Δt^y_{min} as defined by (3.4.382)

DTZ: Δt^z_{min} as defined by (3.4.383)

DUDX: u'_x as defined by (3.4.371)

DVDY: v'_y as defined by (3.4.378)

DWDZ: W'_σ/H as defined by (3.4.378)

XADV: advected distance (m) in the X-direction

XDIFF: diffused distance (m) in the X-direction

XEAST: distance of the particle to the eastern boundary cell face (m)

XEDGE: value of $\Delta x_{1;edge}$ as used in (3.4.381)

XOLD: particle coordinate x'^{n}_{1n} (old value)

XRAN: random number between 0 and 1

XWEST: distance of the particle to the western boundary cell face (m)

YADV: advected distance (m) in the Y-direction

YDIFF: diffused distance (m) in the Y-direction

YEDGE: value of $\Delta x_{2;edge}$ as used in (3.4.382)

YNORT: distance of the particle to the northern boundary cell face (m)

YOLD: particle coordinate x'^{n}_{2n} (old value)

YSOUT: distance of the particle to the southern boundary cell face (m)

ZADV: advected distance (σ-coordinates) in the Z-direction

ZBOT: distance of the particle to the bottom boundary cell face (in σ-coordinates)

ZDIFF: diffused distance (m) in the Z-direction

ZEDGE: value of $\Delta\sigma_{edge}$ as used in (3.4.383)

ZOLD: particle coordinate σ'^{n}_{n} (old value)

ZTOP: distance of the particle to the top boundary cell face (in σ-coordinates)

Description

A flow chart of MC3D is presented in Figure 5.1.6. The code of the routine is written in the form of a DO-loop. Particles outside the computational domain are automatically eliminated by making use of the array NUMPIN defined in STAUCH. The particle positions are first updated for advection. The procedure is the following:

- The rest time Δt_{rest} is first set to the 3-D time step Δt_{3D}.
- The advected distance in the X-, Y-, Z-directions during Δt_{rest} are calculated separately with the aid of (3.4.375)–(3.4.377). Note that particles are not allowed to cross the sea bed or the sea surface.
- If no boundary cell face is crossed by the particle, the particle's coordinates inside the cell are updated and the calculations proceed with the diffusion part.
- If a boundary crossing occurs, the program determines the total number of crossings, given by the parameter IRND (which must be smaller than or equal to 3), and sets a number of pointers (IEAST, IWEST, ...) to indicate which cell faces have been traversed by the particle (e.g. IEAST $= 1$ if the particle crossed the eastern boundary).
- The times Δt^{x}_{min}, Δt^{y}_{min}, Δt^{z}_{min} needed to reach an X-, Y- or Z- boundary are calculated using (3.4.381)–(3.4.383) and stored into respectively the local variables DTX, DTY and DTZ.

- The time Δt_{min} is then defined as the time after which the particle first crosses a boundary. The type of the boundary (West, East, South, North, top, bottom) is fixed by resetting the values of the pointers (e.g. ITOP = 1 and all other counters set to zero if the particle first crosses the top cell face).
- It is checked whether the first crossing takes place at an open sea or land boundary. If this is the case, the particle's label is set to zero, which means that its position will no longer be updated by the program, and its mass is stored in the array STOUT. The counter of the DO-loop is incremented by 1 and the program proceeds with the next particle inside the domain.
- The advected distances (from the original position) in the X-, Y- and Z-direction are recalculated using (3.4.375)–(3.4.377), now using the time step Δt_{min}. The particle coordinates are reset to the new position (at the cell boundary where the first crossing occurs).
- A new rest time is defined by $\Delta t_{rest} \equiv \Delta t_{rest} - \Delta t_{min}$.
- The whole procedure for advection is repeated until no crossings occur during Δt_{rest} or when Δt_{rest} becomes non-positive, in which case the program continues with the diffusion part.

The update of the particle's position for diffusion is first performed along the X-direction, next along the Y-direction and finally along the Z-direction.

- The diffused distance in the X-direction is given by (3.4.386). The maximum velocity fluctuation u'_{max}, stored in the array UTURB, is previously determined in SEDLAG. It is checked whether the particle crosses the western or eastern cell boundaries. If not, the particle's new coordinates are set within the old cell (see equation (3.4.387)). If a crossing occurs, the coordinates are determined within the new adjacent cell (see equations (3.4.388)–(3.4.389)). Since the particle is not allowed to cross a solid or open sea boundary by diffusion, it is assumed to be reflected backwards inside the domain at the boundaries. For more details see Sections III-4.8.4a and III-4.8.5b.
- The same procedure is applied for the diffusion in the Y- and Z-direction, except that in the latter case the particle is allowed to traverse more than one cell boundary in the vertical direction (see Sections III-4.8.4b and III-4.8.5a for details).
- The counter of the DO-loop is incremented by 1 and the program proceeds with the next particle inside the domain.

Remarks

- The exponential and logarithmic factors in (3.4.375)–(3.4.377) and (3.4.381)–(3.4.383) have the form $e^x - 1$ and $\ln(x+1)$. To avoid spurious results due to rounding for small values of x, the expressions are replaced by their linearised forms

$$e^x - 1 \rightarrow x \quad , \quad \ln(x+1) \rightarrow x \quad \text{if} \quad x \rightarrow 0 \tag{5.1.3}$$

- The array STOUT accumulates the amount of suspended matter exiting the computational domain and has no further relevance to the program, but may be useful for user-defined output. Its value is saved at the end of the program (*pou*-file) for an eventual restart.
- The vertical particle coordinate inside a cell (array ZT) is expressed in σ-coordinates. The effects of sea surface and bottom slopes (including vertical currents induced by the bathymetry) are therefore implicitly taken into account.
- Note that the horizontal particle coordinates inside the routine MC3D are always expressed in meters for computational efficiency. In the case of spherical coordinates, the necessary conversions from and to degrees, given by (3.4.366)–(3.4.367), are performed in SEDLAG.
- The diffusion part of the calculations is only performed when IOPTP $= 2$.
- In the horizontal diffusion part it is assumed that the particle cannot cross more than one cell boundary. This condition is satisfied when $u'_{max}\Delta t_{3D}/\Delta x_1$ and $v'_{max}\Delta t_{3D}/\Delta x_2$ are smaller than 1. As shown in Section III-4.8.4a, this condition is always satisfied if the horizontal diffusion coefficient λ_H^C is limited by the criterion (3.4.390). The limiting condition is checked at the start of the program and λ_H^C is set to its maximum value if necessary.

SEDLAG

File

sedlag.f

Type

subroutine

Purpose

the Lagrangian particle module

Calling program

MAINPROG

Externals

CONSPM, MCOSB, MCROB, MC3D, RANDOM, STAUCH

Arguments

none

Description

This is the main unit of the program's Lagrangian particle module. The routine mainly consists of a number of subroutine calls (given in parentheses).

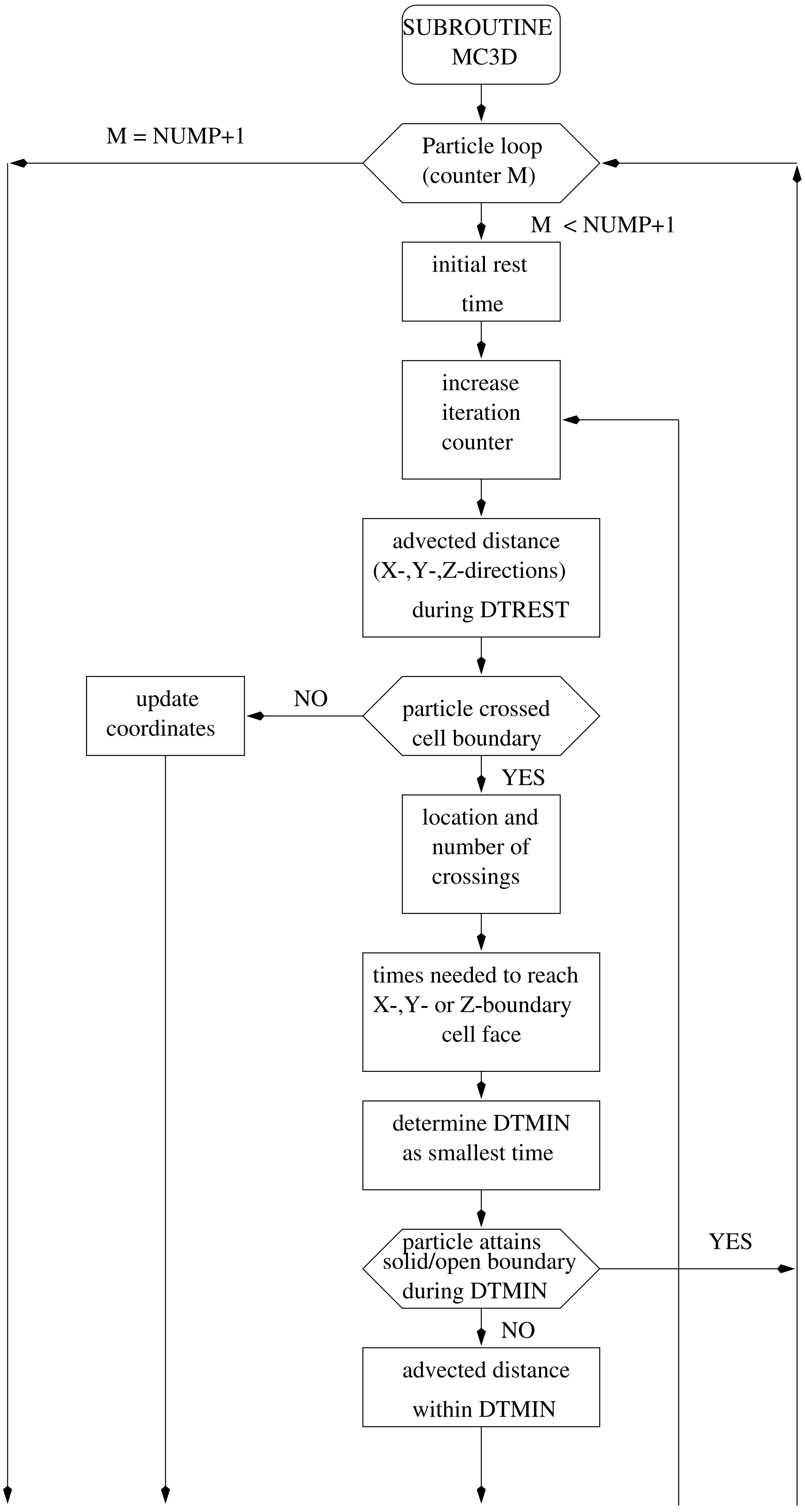

SUBROUTINE MC3D
Particle loop (counter M)
M = NUMP+1
M < NUMP+1
initial rest time
increase iteration counter
advected distance (X-,Y-,Z-directions) during DTREST
particle crossed cell boundary
NO
update coordinates
YES
location and number of crossings
times needed to reach X-,Y- or Z-boundary cell face
determine DTMIN as smallest time
particle attains solid/open boundary during DTMIN
YES
NO
advected distance within DTMIN

Figure 5.1.6: Flow chart of subroutine MC3D.

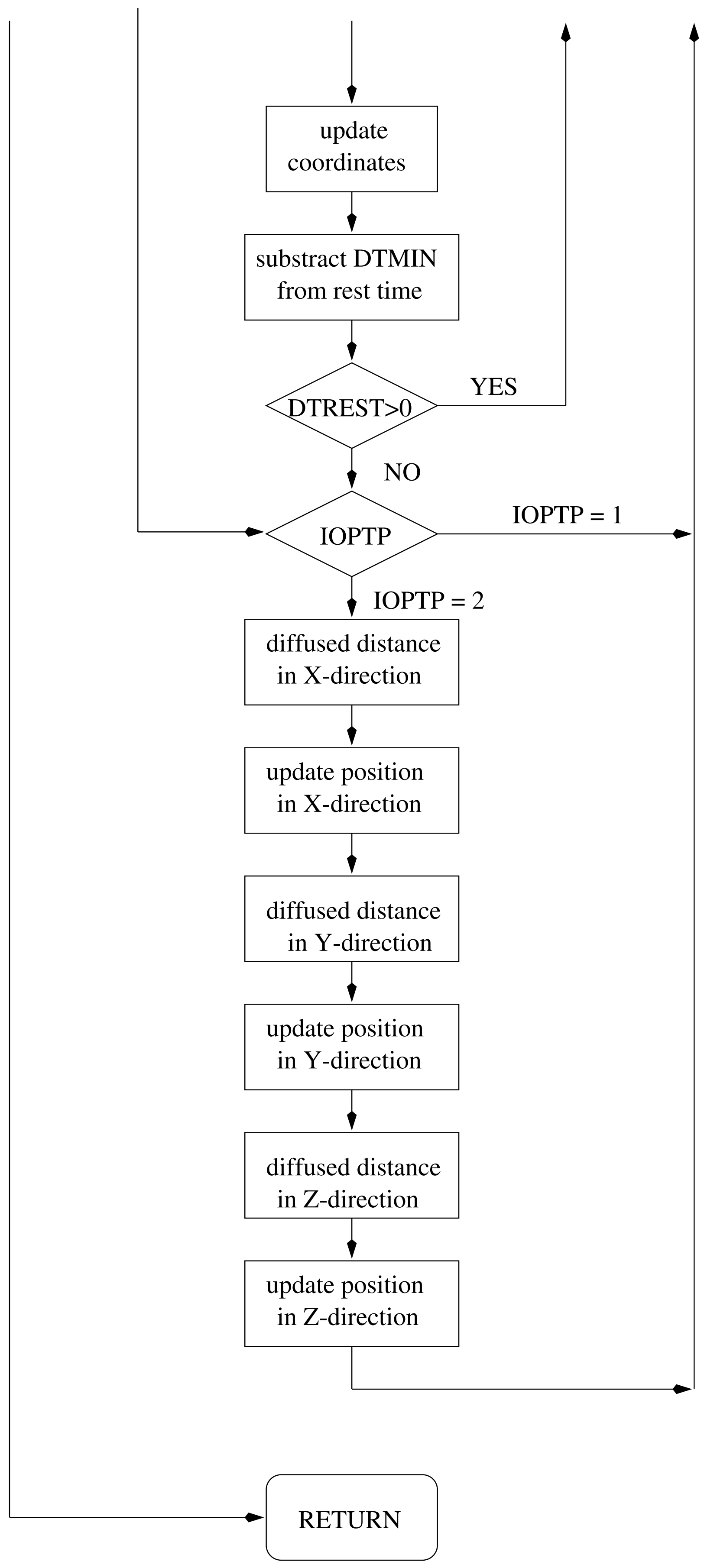

update coordinates
substract DTMIN from rest time
DTREST>0
YES
NO
IOPTP
IOPTP = 1
IOPTP = 2
diffused distance in X-direction
update position in X-direction
diffused distance in Y-direction
update position in Y-direction
diffused distance in Z-direction
update position in Z-direction
RETURN

Figure 5.1.6: continued.

- Initialise the array NUMPIN and the number of particles NUMP inside the computational domain (STAUCH) on first call.
- Convert the relative horizontal particle coordinates x'_{1n} and x'_{2n} (arrays XT and YT) from degrees to meters in the case of spherical coordinates using (3.4.366)–(3.4.367).
- Evaluate the maximum turbulent velocities (u'_{max},v'_{max},w'_{max}) using (3.3.5) and store the results into the arrays UTURB, VTURB, WTURB.
- Update the particle positions for advection and diffusion (MC3D).
- Generate river input (MCROB).
- Generate open sea boundary input (MCOSB).
- Update the array NUMPIN and the number of particles inside the computational domain (STAUCH).
- Convert the particle distribution into a volume density of suspended matter in g/m^3 (CONSPM).
- Convert the relative horizontal particle coordinates x'_{1n} and x'_{2n} (arrays XT and YT) back from meters to degrees in the case of spherical coordinates.

Remarks
SEDLAG is called at each 3-D time step when IOPTP = 1 or 2.

STAUCH

File
sedlag.f

Type
subroutine

Purpose
update the total number and indices of the particles inside the computational domain

Calling program
INITC, PREPROC, SEDLAG

Externals
IZERO

Arguments
none

Description

The routine updates the following parameters:

- The array NUMPIN containing the numbers of the particles inside the domain. This is performed by checking the label of each particle, stored in the array LST. Particles which have left the domain at a land or open sea boundary, are considered as lost and their position is no longer updated. If this occurs, the particle's label is set to zero. The array NUMPIN therefore only contains the indices of those particles whose labels are non-zero.
- The parameter NUMP representing the total number of particles inside the domain.

Remarks

The array NUMPIN is only used for computational efficiency in the Lagrangian modules.

1.7 Boundary conditions

BOUNDC

File

boundc.f

Type

subroutine

Purpose

apply the boundary conditions for the 2-D mode

Calling program

CRRNT2

Externals

none

Arguments

none

Local variables [1]

CRAD: barotropic wave speed $\pm c^n_{ji}$

CRAD0: barotropic wave speed $\pm\tilde{c}_{ji}$ using mean water depths

DUU: the difference $\Delta^c_x\overline{U}_{j:j-1}$

DVV: the difference $\Delta^c_y\overline{V}_{i:i-1}$ or $\Delta^c_y(\cos\phi^v\overline{V}_{i:i-1})$

[1]The upper sign applies at western/southern and the lower sign at eastern/northern boundaries.

DXX: grid spacing $\Delta x_{1;i:i-1}$ or $\Delta x_{1;j:j-1}$

DYY: grid spacing Δx_2, $\cos\phi^c\Delta x_2$ or $\Delta x_{2;j:j-1}$

OBAH: all advection and diffusion terms on the right hand side of (3.4.163) or (3.4.164)

OBCOR: the Coriolis term on the right hand side of (3.4.163) or (3.4.164)

OBDRAG: the bottom stress term on the right hand side of (3.4.163) or (3.4.164)

OBDR2X: the first term on the right hand side of (3.4.163)

OBDR2Y: the first term on the right hand side of (3.4.164)

OBDUDX: the second term on the right hand side of (3.4.164)

OBDVDY: the second term on the right hand side of (3.4.163) using the form (3.4.356) in the spherical case

OBFS: the surface stress term on the right hand side of (3.4.163)

OBGS: the surface stress term on the right hand side of (3.4.164)

OBPR: the atmospheric pressure term on the right hand side of (3.4.163) or (3.4.164)

OBQDEN: the baroclinic pressure gradient term on the right hand side of (3.4.163) or (3.4.164)

OBSPH: the extra spherical term on the right hand side of (3.4.164) as given by (3.4.357)

Description

- Update the phases $\omega_n t + \varphi_{n0}$ in the harmonic expansion (3.1.198). The values are calculated as modulo 2π to avoid rounding errors at large times and stored in the array PHASE0. This array is saved at the end of the program (*hou*-file) and can be used for an eventual restart.
- Evaluate $\overline{U}$ at the open boundaries.
  - Solve (3.1.194) or (3.1.196) for the outgoing Riemann variable R^u_- (western boundary) or R^u_+ (eastern boundary) using the discretised forms (3.4.163) and (3.4.356) (spherical case). The updated value of the outgoing Riemann variable is stored in the array R2OBU.
  - Calculate the incoming variable R^u_+ (western boundary) or R^u_- (eastern boundary) using one of the procedures outlined in Section III-4.3.11b. The formulation depends on the value of the array ITYPOBU at the specific location.

    0: The incoming Riemann variable is set to zero.

    1: Formulation (3.4.169)–(3.4.170).

    2: Zero gradient condition as discussed in Section III-4.3.11b.

    3: Formulation (3.4.171)–(3.4.172).

  4: Formulation (3.4.173)–(3.4.174).

- The incoming Riemann variable is stored in the array R1OBU.
- Determine $\overline{U}$ by taking the average of R^u_+ and R^u_- (see equation (3.1.199)).

• Evaluate $\overline{V}$ at the open boundaries.

- Solve (3.1.195) or (3.1.197) for the outgoing Riemann variable R^v_- (southern boundary) or R^v_+ (northern boundary) using the discretised forms (3.4.164) and (3.4.357) (spherical case). The updated value of the outgoing Riemann variable is stored in the array R2OBV.
- Calculate the incoming variable R^v_+ (southern boundary) or R^v_- (northern boundary) using one of the procedures outlined in Section III-4.3.11b. The formulation depends on the value of the array ITYPOBV at the specific location.

  0: The incoming Riemann variable is set to zero.

  1: Formulation (3.4.169)–(3.4.170).

  2: Zero gradient condition as discussed in Section III-4.3.11b.

  3: Formulation (3.4.171)–(3.4.172).

  4: Formulation (3.4.173)–(3.4.174).
- The incoming Riemann variable is stored in the array R1OBV.
- Determine $\overline{V}$ by taking the average of R^v_+ and R^v_- (see equation (3.1.199)).

Remarks

• The harmonic expansion F_{har} (see equation (3.1.198) or (3.4.167)–(3.4.168)) is evaluated using the arrays AMPOBU, PHAOBU (U-boundaries) and AMPOBV, PHAOBV (V-boundaries). See Section IV-2.4 for details.

• See Section III-4.3.11b for further details about the numerical procedures.

• BOUNDC is called at each 2-D time step if IOPT2 = 1.

BSTRES

File

bstres.f

Type

subroutine

Purpose

bottom stress calculations

Calling program
    CRRNT3C, CRRNT3P, INCTUR, MAINPROG

Externals
    WAVCUR

Arguments
    none

Description

- Initialise the quadratic bottom friction coefficient at the bottom grid cell (C_D^b) and at a reference height of 1 m (C_{100}) as a function of the roughness length z_0 on the first call. The results are stored into the arrays CDB and CDB100.
- Update C_D^b and C_{100} in the case that wave-current interaction is included in the simulation (IOPTW > 0).
- Evaluate the friction coefficient k_b (see equation (3.4.15)) at the U- and V-nodes. The results are stored in the arrays FBK and GBK.
- Evaluate the bottom stress components $(\tau_{b1}, \tau_{b2})/\rho_0$ at respectively the U- and the V-nodes and the magnitude of the bottom stress (normalised by ρ_0) at the cell centre. The results are stored into the arrays -FB, -GB and BSTOT.

CD

File
    metin.f

Type
    real function

Purpose
    calculate the surface drag coefficient C_D^s

Calling program
    SURFLX

Externals
    CHARNO

Arguments
    WIND: magnitude of the wind vector $|U_{10}|$ (m/s)
    TDIF: air-sea temperature difference (^{0}C)

DRAG: array used for the bilinear interpolation procedure

ITYPE: the same meaning as the switch ITDIF

Description

The formulation depends on the value of the switch ITYPE (=ITDIF).

- If ITYPE = 0, C_D^s only depends on the wind speed. A further selection is made with the switch IDRAG.
  0: constant value (0.0013)
  1: Large and Pond (1981)
  2: Smith and Banke (1975)
  3: Geernaert et al. (1986)
  4: Charnock's relation (1955)
- If ITYPE = 1, C_D^s is a function of the wind speed WIND and the air-sea temperature difference TDIF (see Section III-1.6.4 for details). Instead of solving the complicated system (3.1.249), (3.1.252)–(3.1.254), the drag coefficient is obtained by a bilinear interpolation from previously determined values (calculated in FLUXCO) which are stored in the array DRAG. This array contains the values for $C_D^s(U, \Delta T)$ for $0 \leq U \leq 30$m/s and -5^0C $\leq \Delta T \leq 5^0$C at intervals of 2.5 m/s and 1^0C.

Remarks

- Charnock's relation is solved by calling the function CHARNO.
- Arguments TDIF and DRAG are only used if ITYPE = 1.

CE

File

metin.f

Type

real function

Purpose

calculate the surface exchange coefficient C_E (=C_H)

Calling program

SURFLX

Externals

none

Arguments

WIND: magnitude of the wind vector $|U_{10}|$ (m/s)

TDIF: air-sea temperature difference (^{0}C)

EXCH: array used for the bilinear interpolation procedure

ITYPE: the same meaning as the switch ITDIF

Description

The formulation depends on the value of the switch ITYPE (=ITDIF).

- If ITYPE = 0, C_E is set to a constant value (0.00113).
- If ITYPE = 1, C_E is a function of the wind speed WIND and the air-sea temperature difference TDIF (see Section III-1.6.4 for details). Instead of solving the complicated system (3.1.249), (3.1.252)–(3.1.254), the exchange coefficient is obtained by a bilinear interpolation from previously determined values (calculated in FLUXCO) stored in the array EXCH. This array contains the values for $C_E(U, \Delta T)$ for $0 \leq U \leq 30$m/s and $-5^0\mathrm{C} \leq \Delta T \leq 5^0\mathrm{C}$ at intervals of 2.5 m/s and 1^0C.

Remarks

Arguments TDIF and DRAG are only used when ITYPE = 1.

CHARNO

File

metin.f

Type

real function

Purpose

solve Charnock's relation

Calling program

CD

Externals

none

Arguments

WIND: magnitude of the wind vector (m/s)

Local variables

ALCHAR: Charnock's constant a (=0.014)

EPS: accuracy of the algorithm (=10^{-6})

Description

The function solves equation (3.1.222) for C_D^s by a Newton-Raphson iteration procedure.

Remarks

CHARNO is called when IDRAG = 4.

FLUXCO

File

metin.f

Type

subroutine

Purpose

calculate the surface drag coefficient C_D^s and exchange coefficient C_E (=C_H) at fixed intervals of wind speed $|U_{10}|$ and air-sea temperature difference ΔT

Calling program

SURFLX

Externals

CD, CE, ERROR, HUMID

Arguments

DRAG: array with the values of C_D^s at fixed intervals of $|U_{10}|$ and ΔT

EXCH: array with the values of C_E at fixed intervals of $|U_{10}|$ and ΔT

Local variables

ALPHA: constant α in (3.1.233)

AVAP: vapour pressure e_a at air temperature

BETA: constant β in (3.1.233)

CDN: neutral value C_{DN} of the surface drag coefficient C_D^s

CEN: neutral value C_{EN} of the surface exchange coefficient C_E

PHIH: value of ϕ_h from (3.1.234)

PHIM: value of ϕ_m from (3.1.233)

PSIH: value of ψ_h from (3.1.241)

PSIM: value of ψ_m from (3.1.240)

QA: specific humidity q_a at air temperature

QS: specific humidity q_s at sea surface temperature

RHREF: reference value for the specific humidity RH (=0.8)

SVAP: vapour pressure e_s at sea surface temperature

TAIR: air temperature (^{0}C)

TDIF: air-sea temperature difference (^{0}C)

TKELV: air temperature in degrees Kelvin (K)

TSEA: sea surface temperature (^{0}C)

TVIR: virtual temperature from (3.1.232) (K)

WIND: wind speed $|U_{10}|$ (m/s)

Description

The non-linear system (3.1.249), (3.1.252)–(3.1.254) with ψ_m and ψ_h given by (3.1.240) and (3.1.241) is solved using an Newton-Raphson iteration scheme for C_D^s and C_E and selected values of $|U_{10}|$ and ΔT. The results are stored in the arrays DRAG and EXCH. The values of $|U_{10}|$ are between 0 and 30 m/s at intervals of 2.5 m/s, those of ΔT between -5 and 5^{0}C at intervals of 1^{0}C.

Remarks

- The following simplifications are considered:
  - C_H is set to C_E.
  - The sea temperature T_s is set to the reference value T_0 (parameter TREF) so that $T_a = T_s + \Delta T$.
  - The relative humidity is set to a reference value of 80%.
- FLUXCO is only called at the initial time if ITDIF = 1. The arrays DRAG and EXCH are passed to the functions CD and CE where C_D^s and C_E are calculated at each subsequent meteorological time step.

HUMID

File

metin.f

Type

subroutine

Purpose
: calculate the vapour pressure e and the specific humidity q

Calling program
: FLUXCO, SURFLX

Externals
: none

Arguments
: T: temperature (^{0}C)
: RH: relative humidity
: PVAP: vapour pressure (mb)
: Q: humidity

Description
: The vapour pressure is calculated using (3.1.177), the humidity by (3.1.176).

Remarks
: HUMID is called two times: firstly for the air values q_a, e_a and secondly for the sea surface values q_s, e_s.

SURFLX

File
: *metin.f*

Type
: subroutine

Purpose
: evaluate surface stress, non-solar heat flux and salinity flux at the sea surface

Calling program
: METIN

Externals
: CD, CE, FLUXCO, HUMID

Arguments
: none

Local variables

- AVAP: vapour pressure e_a at air temperature (mbar)
- CDS: surface drag coefficient C_D^s
- CES: surface exchange coefficient C_E and C_H
- CPAIR: specific heat c_{pa} of air at constant pressure from (3.1.174) (J kg^{-1} (^{0}C)$^{-1}$)
- EVAP: evaporation rate E_{vap} (kg m^{-2} s^{-1})
- HVAP: latent heat of vaporization L_ν from (3.1.175) (J/kg)
- QA: specific humidity q_a at air temperature
- QLAT: latent heat flux Q_{la} from (3.1.172) (W/m^2)
- QNLW: long-wave radiation flux Q_{lw} from (3.1.178) (W/m^2)
- QS: specific humidity q_s at sea surface temperature
- QSEN: sensible heat flux Q_{sen} from (3.1.173) (W/m^2)
- RHOA: air density ρ_a (=1.2 kg/m^3)
- SVAP: vapour pressure e_s at sea surface temperature (mbar)
- TKELV: sea surface temperature in degrees Kelvin (K)
- WIND: wind speed $|U_{10}|$ (m/s)

Description

- Call FLUXCO if SURFLX is first called and ITDIF $= 1$.
- Evaluate the surface stress components τ_{s1},τ_{s2} (normalised by ρ_0) using (3.1.168) and store the results into the arrays FS and GS. The drag coefficient C_D^s is obtained by calling CD. Store the total surface stress into SSTOT (normalised by ρ_0).
- Obtain the non-solar heat flux after evaluation of the latent, sensible and long-wave radiation fluxes using the formulation described in Section III-1.6.1b. The exchange coefficient C_E is obtained by calling CE, the vapour pressures e_a,e_s and the specific humidities q_a,q_s by calling HUMID.
- Evaluate the salinity flux using (3.1.179). Note the different formulation if IOPTSA $= 1$ or 2. In the former case the evaporation minus precipitation rate is given by a user supplied array whereas in the latter case E_{vap} is determined by the program with R_{pr} supplied by the user.

Remarks

- The evaluation of the heat and salinity fluxes depend on the values of the switches IOPTHE, IOPTSA and IOPTM.
- SURFLX is called at each meteorological time step if IOPTM > 1.

WAVCUR

File
: *wavin.f*

Type
: subroutine

Purpose
: evaluate the bottom drag coefficient C_D^b using wave-current interaction theory

Reference
: Signell et al. (1990), Davies and Lawrence (1994, 1995)

Calling program
: BSTRES

Externals
: none

Arguments
: none

Local variables

AB: near bottom excursion amplitude A_b (m)

APBROU: apparent bottom roughness k_{bc} (m)

BCUR: bottom current u_c (m/s)

BETA: exponential factor β given by (3.1.283)

CSTAR: friction velocity u_{*b} (m/s)

CWSTAR: combined current-wave friction velocity u_{*cw} (m/s)

FWAVE: wave friction factor f_w

PHBROU: physical bottom roughness length k_b (m)

UWAVE: near-bottom wave orbital velocity U_w (m/s)

WFREQ: wave frequency ω_w (rad/s)

WSTAR: wave friction velocity U_{*w} (m/s)

Description

The routine follows the procedure described in Section III-1.6.6:

- Initialise k_{bc} by the physical roughness k_b on first call. This value will be used as a first guess in (3.1.281).
- Evaluate ω_w, U_w using (3.1.272)–(3.1.273), A_b and k_b.
- Evaluate f_w using the formulation (3.1.277).

- Determine U_{*w} from (3.1.276).
- Calculate C_D^b from (3.1.281) using the value of k_{bc} obtained at the previous call. The result is stored in the array CDZ0.
- Calculate C_{100} using the second relation (3.2.74) with z_0 replaced by $k_{bc}/30$. The result is stored in the array CDB100.
- Determine u_{*b} from (3.1.279), u_{*cw} from (3.1.278), β from (3.1.283).
- Update k_{bc} using (3.1.282) which will be used in (3.1.281) on the next call.

Remarks

The routine is only called when IOPTW > 0.

WAVNUM

File

wavin.f

Type

subroutine

Purpose

solve the dispersion relation (3.1.274) for the wave number k_w

Calling program

MAINPROG, WAVIN

Externals

ERROR

Arguments

none

Description

Instead of solving (3.1.274) by iteration, sufficiently accurate values are obtained using the following more direct algorithm:

$$k_w \simeq \omega_w \sqrt{\frac{x+\alpha}{gH}} \tag{5.1.4}$$

with

$$x = \frac{\omega_w^2 H}{g} \tag{5.1.5}$$

$$\alpha = \Big[1 + x(0.6667 + x(0.3556 + \beta x))\Big]^{-1} \tag{5.1.6}$$

$$\beta = 0.1608 + x(0.06321 + x(0.02744 + 0.01x)) \tag{5.1.7}$$

1.8 Numerical routines

FNLIM

File
: *Upwnd_in.f*

Type
: real function

Purpose
: evaluate the weight factor $\Omega(r)$ for the TVD scheme

Calling program
: HAD2DU, HAD2DV, XADVDIS, XHAD3DU, XHAD3DV, YADVDIS, YHAD3DU, YHAD3DV, ZADVDIS

Externals
: none

Arguments

- ADV2: Lax-Wendroff (horizontal) or central (vertical) advective flux at the point where the total advective flux is evaluated
- ADV1: as ADV2 now for the upwind advective flux
- ADV2UP: Lax-Wendroff (horizontal) or central (vertical) advective flux evaluated at the location one grid distance away in the upwind direction from the point where the total advective flux is evaluated
- ADV1UP: as ADV2UP now for the upwind advective flux
- IFNLIM: type of limiting function (superbee or monotonic)

Local variables

- R: argument r of the weight function Ω

Description

- The parameter r is calculated as follows:
  - Evaluate the difference ADV2UP - ADV1UP between the Lax-Wendroff (central) and upwind fluxes in the upwind direction.
  - Evaluate the difference ADV2 - ADV1 between the Lax-Wendroff (central) and upwind fluxes at the grid point where the advective flux is taken.
  - Divide the first difference by the second.
- The weight factor $\Omega(r)$ is determined either using the superbee formulation (3.4.54) if FNLIM $= 3$ or the monotonic formulation (3.4.55) if FNLIM $= 4$.

Remarks

- To prevent a division by zero, $\Omega(r)$ is set to zero (giving the upwind scheme) if the absolute value of ADV2 - ADV1 drops below a (small) lower limit. The conditions (3.4.159)–(3.4.162) and (3.4.255)–(3.4.256) are automatically satisfied since then ADV2 = ADV1 = 0.
- To illustrate the meaning of the first four arguments the following example is given. Assume that a quantity is advected in the X-direction and the advective flux is evaluated at the grid point (K,J,I). The fluxes ADV2 and ADV1 then represent the fluxes at (K,J,I) whereas ADV2UP and ADV1UP represent the fluxes either at (K,J,I-1) if the advective velocity (u) is positive at (K,J,I), or at (K,J,I+1) if the advective flux is negative.
- FNLIM is only called if IADVC or IADVS equals 3 or 4.
- For more details see Sections III-4.3 and III-4.4.

HAD2DU

File

hadvdis.f

Type

subroutine

Purpose

evaluate the advection and diffusion terms in the $\overline{U}$-equation (3.1.34) or (3.1.61)

Calling program

CRRNT2, UCALC

Externals

FNLIM, ZERO

Arguments

none

Local variables

ADVFLC: 2-D equivalent of the advective flux F^c_{xx}

ADVFLE: 2-D equivalent of the advective flux $F^{E;v}_{yx}$

ADVFLW: 2-D equivalent of the advective flux $F^{W;v}_{yx}$

ADVFLWC: 2-D equivalent of the Lax-Wendroff flux F^c_{lw}

ADVFLWE: 2-D equivalent of the Lax-Wendroff flux $F_{lw}^{E;v}$

ADVFLWW: 2-D equivalent of the Lax-Wendroff flux $F_{lw}^{W;v}$

ADVFUPC: 2-D equivalent of the upwind flux F_{up}^{c}

ADVFUPE: 2-D equivalent of the upwind flux $F_{up}^{E;v}$

ADVFUPW: 2-D equivalent of the upwind flux $F_{up}^{W;v}$

XDIFLU: 2-D equivalent of the diffusion flux D_{xx}^{c}

XDIFLV: 2-D equivalent of the diffusion flux $D_{yx}^{V;u}$

YDIFLU: 2-D equivalent of the diffusion flux $D_{yx}^{U;v}$

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

The advection and diffusion terms are given by respectively $\overline{\mathcal{A}_{hx}}(\overline{U}) + \overline{\mathcal{A}_{hy}}(\overline{U})$ (plus an additional term in the spherical case) and $\overline{\mathcal{D}_{xx}}(\overline{U}) + \overline{\mathcal{D}_{yx}}(\overline{U}, \overline{V})$ (plus an additional term in the spherical case). They are calculated separately and stored into respectively the arrays UAH2D and UDH2D at the end of the routine.

- Step 1: advective fluxes
  - Initialise the advective and diffusion fluxes to zero.
  - Determine the advective flux $\overline{U}\,\overline{U}/H$ at the cell centre (with the advected current interpolated at the centre) by first evaluating the upwind and Lax-Wendroff fluxes and then applying the 2-D equivalent of (3.4.48). Results are stored in ADVFLC.
  - The advective flux $\overline{V}\,\overline{U}/H$ is determined at the V-node in two ways. Firstly, the upwind and Lax-Wendroff fluxes are evaluated using values for the advected quantity $\overline{U}$ at the U-nodes West of the V-node (points W_1 and W_2 in Figure 5.1.7a). The fluxes, derived from the 2-D equivalent of (3.4.66), are stored in the local array ADVFLW. The same procedure is repeated now using $\overline{U}$-values East of the V-node (points E_1 and E_2 in Figure 5.1.7a). Results are stored in ADVFLE.
  - The open boundary values of ADVFLW and ADVFLE are obtained using the upwind scheme and a zero gradient condition in the case of inflow.
- Step 2: diffusion fluxes
  - The stress $\overline{\tau_{11}}$ or $\overline{\tau_{\lambda\lambda}}$ is evaluated at the cell centre. The $\tan\phi$-term in (3.1.57) is added to $\overline{\tau_{\lambda\lambda}}$. The result is stored into the local array XDIFLU.
  - The Y-derivative term in $\overline{\tau_{21}}$ or $\overline{\tau_{\phi\lambda}}$ is evaluated at the V-nodes using $\overline{U}$-values interpolated at the centre and is stored in YDIFLU.
  - The X-derivative term in $\overline{\tau_{21}}$ or $\overline{\tau_{\phi\lambda}}$ is evaluated at the U-nodes using $\overline{V}$-values interpolated at the centre and stored in XDIFLV. The $\tan\phi$-term is added to the array in the spherical case.

  - Open boundary values of YDIFLU and XDIFLV are set to their nearest interior values.

- Step 3: advection terms at interior U-nodes
  - The X-derivative term is determined by taking the difference of the fluxes stored in ADVFLC.
  - The Y-derivative term is derived by taking the difference of the fluxes at the corner nodes C_1 and C_2 (see Figure 5.1.7b). The corner fluxes at C_2 (C_1) are given by the average of the "East"-flux at E_{f2} (E_{f1}) and the "West"-flux at W_{f2} (W_{f1}).
  - The $\tan\phi$-term is added in the spherical case.
  - The sum of all advective terms is stored in UAH2D.

- Step 4: diffusion terms at interior U-nodes
  - The X-derivative of $\overline{\tau_{11}}$ or $\overline{\tau_{\lambda\lambda}}$ is determined by taking the difference of the fluxes stored in XDIFLU.
  - The Y-derivative of $\overline{\tau_{21}}$ or $\cos\phi\,\overline{\tau_{\phi\lambda}}$ is obtained by interpolating YDIFLU at the corners and differencing, and by differencing XDIFLV over two grid cells in the Y-direction (using a special procedure near the edges of the computational domain).
  - The sum of all diffusion terms is stored in UDH2D.
  - In the spherical case, the term $-\tan\phi\,\overline{\tau_{\phi\lambda}}/R$ is added to UDH2D.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the fluxes at the centres are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the cell centre is only one-half grid distance away from an open/solid boundary and the advective current is pointing away from that boundary. The same procedure is applied for the western and eastern fluxes one grid distance away from an open/solid boundary.
- The arrays SPHCUR and SPHCURV represent the value of $\tan\phi/R$ in the spherical case and are set to zero in the Cartesian case (IGTRH $= 0$).
- The arrays COSPHI and COSPHIV represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- Advective terms are calculated if IADVC > 0, diffusion terms if IODIF > 0.
- For more details about the numerical procedures see Section III-4.3.

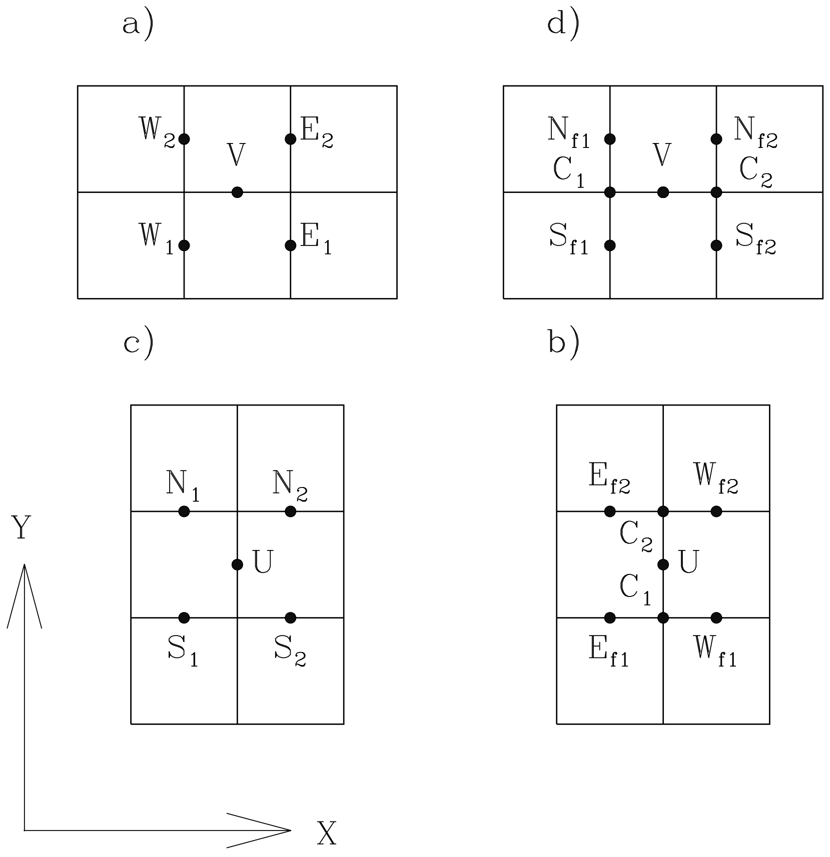


Figure 5.1.7: The diagrams illustrate how the cross-stream advective fluxes and terms are evaluated in the horizontal momentum equations. See text for details.

HAD2DV

File
: *hadvdis.f*

Type
: subroutine

Purpose
: evaluate the advection and diffusion terms in the $\overline{V}$-equation (3.1.35) or (3.1.62)

Calling program
: CRRNT2, VCALC

Externals
: FNLIM, ZERO

Arguments
: none

Local variables
: ADVFLC: 2-D equivalent of the advective flux F^c_{yy}
: ADVFLN: 2-D equivalent of the advective flux $F^{N;u}_{xy}$
: ADVFLS: 2-D equivalent of the advective flux $F^{S;u}_{xy}$
: ADVFLWC: 2-D equivalent of the Lax-Wendroff flux F^c_{lw}
: ADVFLWN: 2-D equivalent of the Lax-Wendroff flux $F^{N;u}_{lw}$
: ADVFLWS: 2-D equivalent of the Lax-Wendroff flux $F^{S;u}_{lw}$
: ADVFUPC: 2-D equivalent of the upwind flux F^c_{up}
: ADVFUPN: 2-D equivalent of the upwind flux $F^{N;u}_{up}$
: ADVFUPS: 2-D equivalent of the upwind flux $F^{S;u}_{up}$
: XDIFLU: 2-D equivalent of the diffusion flux D^c_{xx}
: XDIFLV: 2-D equivalent of the diffusion flux $D^{V;u}_{xy}$
: YDIFLU: 2-D equivalent of the diffusion flux $D^{U;v}_{xy}$
: YDIFLV: 2-D equivalent of the diffusion flux D^c_{yy}
: CFL: CFL-number
: PSI: weight factor $\Omega(r)$

Description
: The advection and diffusion terms are given by respectively $\overline{\mathcal{A}_{hx}}(\overline{V}) + \overline{\mathcal{A}_{hy}}(\overline{V})$ (plus an additional term in the spherical case) and $\overline{\mathcal{D}_{xy}}(\overline{U},\overline{V}) + \overline{\mathcal{D}_{yy}}(\overline{V})$ (plus an additional term in the spherical case). They are calculated separately and stored into respectively the arrays VAH2D and VDH2D at the end of the routine.

- Step 1: advective fluxes
  - Initialise the advective and diffusion fluxes to zero.
  - Determine the advective flux $\overline{V}\,\overline{V}/H$ at the cell centre (with the advected current interpolated at the centre) by first evaluating the upwind and Lax-Wendroff fluxes and then applying the 2-D equivalent of (3.4.58). Results are stored in ADVFLC.
  - The advective flux $\overline{U}\,\overline{V}/H$ is determined at the U-node in two ways. Firstly, the upwind and Lax-Wendroff fluxes are evaluated using values for the advected quantity $\overline{V}$ at the V-nodes South of the U-node (points S_1 and S_2 in Figure 5.1.7c). The fluxes, derived from the 2-D equivalent of (3.4.76), are stored in the local array ADVFLS. The same procedure is repeated now using $\overline{V}$-values North of the U-node (points N_1 and N_2 in Figure 5.1.7c). Results are stored in ADVFLN.
  - The open boundary values of ADVFLS and ADVFLN are obtained using the upwind scheme and a zero gradient condition in the case of inflow.
- Step 2: diffusion fluxes
  - The stress $\overline{\tau_{22}}$ or $\overline{\tau_{\phi\phi}}$ is evaluated at the cell centre and stored into the local array YDIFLV.
  - The X-derivative term in $\overline{\tau_{12}}$ or $\overline{\tau_{\lambda\phi}}$ is evaluated at the U-nodes using $\overline{V}$-values interpolated at the centre and is stored in XDIFLV. The $\tan\phi$-term is added to the array in the spherical case.
  - The Y-derivative term in $\overline{\tau_{12}}$ or $\overline{\tau_{\lambda\phi}}$ is evaluated at the V-nodes using $\overline{U}$-values interpolated at the centre and stored in YDIFLU.
  - In the spherical case, the X-derivative in $\overline{\tau_{\lambda\lambda}}$ is evaluated at the cell centre and stored in XDIFLU.
  - Open boundary values of XDIFLV and YDIFLU are set to their nearest interior values.
- Step 3: advection terms at interior V-nodes
  - The Y-derivative term is determined by taking the difference of the fluxes stored in ADVFLC.
  - The X-derivative term is derived by taking the difference of the fluxes at the corner nodes C_1 and C_2 (see Figure 5.1.7d). The corner fluxes at C_2 (C_1) are given by the average of the "North"-flux at N_{f2} (N_{f1}) and the "South"-flux at S_{f2} (S_{f1}).
  - The $\tan\phi$-term is added in the spherical case.
  - The sum of all advective terms is stored in VAH2D.
- Step 4: diffusion terms at interior V-nodes
  - The Y-derivative of $\overline{\tau_{22}}$ or $\cos\phi\,\overline{\tau_{\phi\phi}}$ is determined by taking the difference of the fluxes stored in YDIFLV.

- The X-derivative of $\overline{\tau_{12}}$ or $\overline{\tau_{\lambda\phi}}$ is obtained by interpolating XDIFLV at the corners and differencing, and by differencing YDIFLU over two grid cells in the X-direction (using a special procedure near the edges of the computational domain).
- The sum of all diffusion terms is stored in VDH2D.
- In the spherical case, $\overline{\tau_{\lambda\lambda}} \tan\phi/R$ is added to VDH2D.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the fluxes at the centres are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the cell centre is only one-half grid distance away from an open/solid boundary and the advective current is pointing away from that boundary. The same procedure is applied for the southern and northern fluxes one grid distance away from an open/solid boundary.
- The arrays SPHCUR and SPHCURV represent the value of $\tan\phi/R$ in the spherical case and are set to zero in the Cartesian case (IGTRH $= 0$).
- The arrays COSPHI and COSPHIV represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- Advective terms are calculated if IADVC > 0, diffusion terms if IODIF > 0.
- For more details about the numerical procedures see Section III-4.3.

TRANSH

File

transh.f

Type

subroutine

Purpose

evaluate the horizontal grid spacings, geographical coordinates, Coriolis frequency and other work space arrays

Calling program

INITC, PREPROC

Externals

none

Description

- The horizontal grid spacings (m) are determined by

$$\Delta x_{1;i} = x_{1;i+1} - x_{1;i}\,,\ \Delta x_{2;j} = x_{2;j+1} - x_{2;j} \tag{5.1.8}$$

  in Cartesian or

$$\Delta x_{1;ji} = R\cos\phi_j^c(\lambda_{i+1} - \lambda_i)\,,\ \Delta x_{2;j} = R(\phi_{j+1} - \phi_j) \tag{5.1.9}$$

  in spherical coordinates, where R denotes the Earth's radius, $x_{1;i}$ or λ_i are the X-coordinates of the cell corners stored into the array GX0 and $x_{2;j}$ or ϕ_j the Y-coordinates stored in GY0. The grid spacings are stored in GX2 and GY2. Note that ϕ and λ in (5.1.9) are converted in the program to radians.

- Evaluate the longitude and latitude at the cell centres (used in subroutine SOLRAD). The values are stored in DLON and DLAT. In the Cartesian case, longitude and latitude have the spatially uniform values DLAREF and DLOREF.

- Evaluate the Coriolis frequency using $f = 2\Omega\sin\phi$ where $\Omega = 4\pi/86164$ is the frequency of the Earth's rotation. The results are stored in CORIOL (U-nodes and cell centres) and CORIOLV (V-nodes). In the spherical case, ϕ equals the reference value DLAREF and f is spatially uniform.

- For computational efficiency the term $\tan\phi/R$ is stored into the array SPHCUR (U-nodes and cell centres) and SPHCURV (V-nodes). The arrays are set to zero in the Cartesian case. In this way, all the extra terms which appear in the spherical formulation of the model equations are automatically cancelled if IGTRH $= 0$.

- For computational efficiency (evaluation of advection and diffusion terms in the momentum and scalar transport equations) the term $\cos\phi$ is stored into the array COSPHI (U-nodes and cell centres) and COSPHIV (V-nodes). The arrays are set to 1 in the Cartesian case.

TRANSPC

File

  transp.f

Type

  subroutine

Purpose

  solve the general transport equation (3.1.70) for a cell-centered scalar quantity ψ

Calling program

BIOLGY, HEAT, MASS, SALT, SEDEUL

Externals

THOMV, XADVDIS, YADVDIS, ZADVDIS, ZERO

Arguments

PHI: value ψ^n (old time) on input, value ψ^{n+1} (new time) on output (unit is e.g. Q which may be ^{0}C, PSU or some concentration)

VEDPHI: vertical diffusion coefficient at W-nodes (m^2/s)

SOURCE: the sum of all source and sink terms in the transport equation (3.1.70), as given by $\beta(\psi) = \mathcal{P}(\psi) - \mathcal{S}(\psi)$

SINK: zero array (see first remark below)

CALL1: logical flag to determine whether TRANSPC is called for the first time from the calling routine

ISUR: switch to determine the type of surface boundary condition (see description of ZADVDIS for details)

IBOT: switch to determine the type of bottom boundary condition (see description of ZADVDIS for details)

JADVHS: switch to select the type of scheme for horizontal advection

0: no advection
1: upwind
2: Lax-Wendroff
3: TVD with superbee limiting function
4: TVD with monotonic limiting function

JADVWS: switch to select the type of scheme for vertical advection

0: no advection
1: upwind
2: central
3: TVD with superbee limiting function
4: TVD with monotonic limiting function

JDIFS: switch to select the type of scheme for horizontal diffusion

0: no diffusion
1: uniform coefficients
2: Smagorinsky formulation

IVPU: the same meaning as the array IVPOBU (see Section IV-2.5.1)

IVPV: the same meaning as the array IVPOBV (see Section IV-2.5.1)

PHIVP: "PSIVP"-array (Q) to be used at open boundaries (see Section IV-2.5.2)

W2PHI: vertical current (m/s) including sinking velocities at the W-nodes, for vertical advection (m/s)

SFLUX: parameter F^{ψ}_{s0} in the surface boundary condition (3.1.181), representing the surface flux (Q m/s)

STRAN: parameter F^{ψ}_{s1} in the surface boundary condition (3.1.181), representing the transfer coefficient (m/s)

SPHI: parameter ψ_{se} in the surface boundary condition (3.1.181), representing the external value (Q)

BFLUX: parameter F^{ψ}_{b0} in the bottom boundary condition (3.1.191), representing the bottom flux (Q m/s)

BTRAN: parameter F^{ψ}_{b1} in the bottom boundary condition (3.1.191), representing the transfer coefficient (m/s)

BPHI: parameter ψ_{be} in the bottom boundary condition (3.1.191), representing the external value (Q)

Local variables

PHIA: value of ψ_A (step A)

PHIB: value of ψ_B (step B)

PLHSA: A-vector in the tridiagonal system (ψ_{k-1}-terms)

PLHSB: B-vector in the tridiagonal system (ψ_k-terms)

PLHSC: C-vector in the tridiagonal system (ψ_{k+1}-terms)

PRHS: D-vector in the tridiagonal system (explicit terms)

UCORR: corrector term $\mathcal{M}_x(\psi)$

VCORR: corrector term $\mathcal{M}_y(\psi)$

WCORR: corrector term $\mathcal{M}_z(\psi)$

Description

The numerical solution of the scalar advection-diffusion equation is displayed schematically in Figure 5.1.8 and proceeds along the lines described in Section III-4.4.

- Set ψ to zero at all dry cells on first call.
- Initialise the arrays PHIA and PHIB.
- Evaluate the corrector terms using (3.4.243)–(3.4.245) if advection is applied (JADVHS > 0).

The solution procedure depends on the values of the switch JADVHS. If JADVHS equals 0 (no advection, or 1-D appplication) or 1 (upwind scheme), the updated value ψ^{n+1} at the new time is obtained as follows:

- Initialise the arrays of the triangular matrix and assemble the inertia terms using (3.4.258)–(3.4.259).
- Add the corrector terms $\mathcal{M}_x$, $\mathcal{M}_y$ and $\mathcal{M}_z$ to PRHS if JADVHS $= 1$
- Add the advective and diffusion terms in the X-direction to the arrays PLHSB and PRHS by calling XADVDIS.
- Add the advective and diffusion terms in the Y-direction to the arrays PLHSB and PRHS by calling YADVDIS.
- Add the SOURCE array to PRHS and the (zero) SINK array to PLHSB.
- Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal system by calling ZADVDIS.
- Solve the tridiagonal system (THOMV) for ψ^{n+1}.
- Exit

The numerical procedure uses the operator splitting method if JADVHS is larger than 1.

- Assemble the inertia terms (first A-step) and add the corrector term $\mathcal{M}_x$ to PRHS.
- Solve (3.4.202) for $\psi_A^{n+1/3}$ with the advective/diffusion terms on the right determined from a call to XADVDIS.
- Assemble the inertia terms (second A-step) and add the corrector term $\mathcal{M}_y$ to PRHS.
- Solve (3.4.203) for $\psi_A^{n+2/3}$ with the advective/diffusion terms on the right determined from a call to YADVDIS.
- Solve (3.4.204) for ψ_A^{n+1}:
  - Initialise the arrays of the triangular matrix and assemble the inertia terms using (3.4.258)–(3.4.259).
  - Add the corrector term $\mathcal{M}_z$ to PRHS.
  - Add the SOURCE array to PRHS and the (zero) SINK array to PLHSB.
  - Add the vertical advection and diffusion (implicit/explicit) terms to the appropriate components of the tridiagonal system by calling ZADVDIS.
  - Solve the tridiagonal system (THOMV) for ψ_A^{n+1}.
- Solve (3.4.205) for $\psi_B^{n+1/3}$ using the previous procedure.
- Assemble the inertia terms (second B-step) and add the corrector term $\mathcal{M}_y$ to PRHS.
- Solve (3.4.206) for $\psi_B^{n+2/3}$ with the advective/diffusion terms on the right determined from a call to YADVDIS.

- Assemble the inertia terms (third B-step) and add the corrector term $\mathcal{M}_x$ to PRHS.
- Solve (3.4.207) for ψ_B^{n+1} with the advective/diffusion terms on the right determined from a call to XADVDIS.
- Evaluate ψ^{n+1} as the average between ψ_A^{n+1} and ψ_B^{n+1} (see equation (3.4.208)).

Remarks

- The source and sink terms in the transport equation (3.1.70) for a cell-centered scalar quantity can be evaluated in two ways within TRANSPC. In the first one all source/sink terms, given $\beta(\psi) = \mathcal{P}(\psi) - \mathcal{S}(\psi)$, are taken explicitly and stored into the argument SOURCE while the SINK array is set to zero. In the second one only the (positive) terms $\mathcal{P}(\psi)$ are stored into SOURCE and taken explicitly whereas the sink terms are evaluated quasi-implicitly in which case the array SINK contains the values of $\mathcal{S}(\psi)/\psi$. For the reasons, explained in the introduction of Section III-4.4, only the first approach is considered in the current version of the program.
- Important to note is that all horizontal and vertical advective terms, except the sinking rate term for the biological and sediment modules, are cancelled if JADVHS $= 0$. In that case, the argument W2PHI only contains the values of the vertical sinking velocity (i.e. excluding the "physical" velocity $J\tilde{w}$). The vertical sinking term is then evaluated using the advection scheme selected by JADVWS (equal to the value of the switch IADVWB).
- The argument W2PHI represents the sum of the transformed velocity $J\tilde{w}$ (if JADVHS > 0) and (eventually) an additional velocity representing the sinking of suspended material. Note that only the first part (array W2) is used in the evaluation of the corrector term $\mathcal{M}_z$.
- The arrays COSPHI and COSPHIV, used in the evaluation of $\mathcal{M}_y(\psi)$, represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- For more details about the numerical procedures see Section III-4.4.

TRANSPW

File

transp.f

Type

subroutine

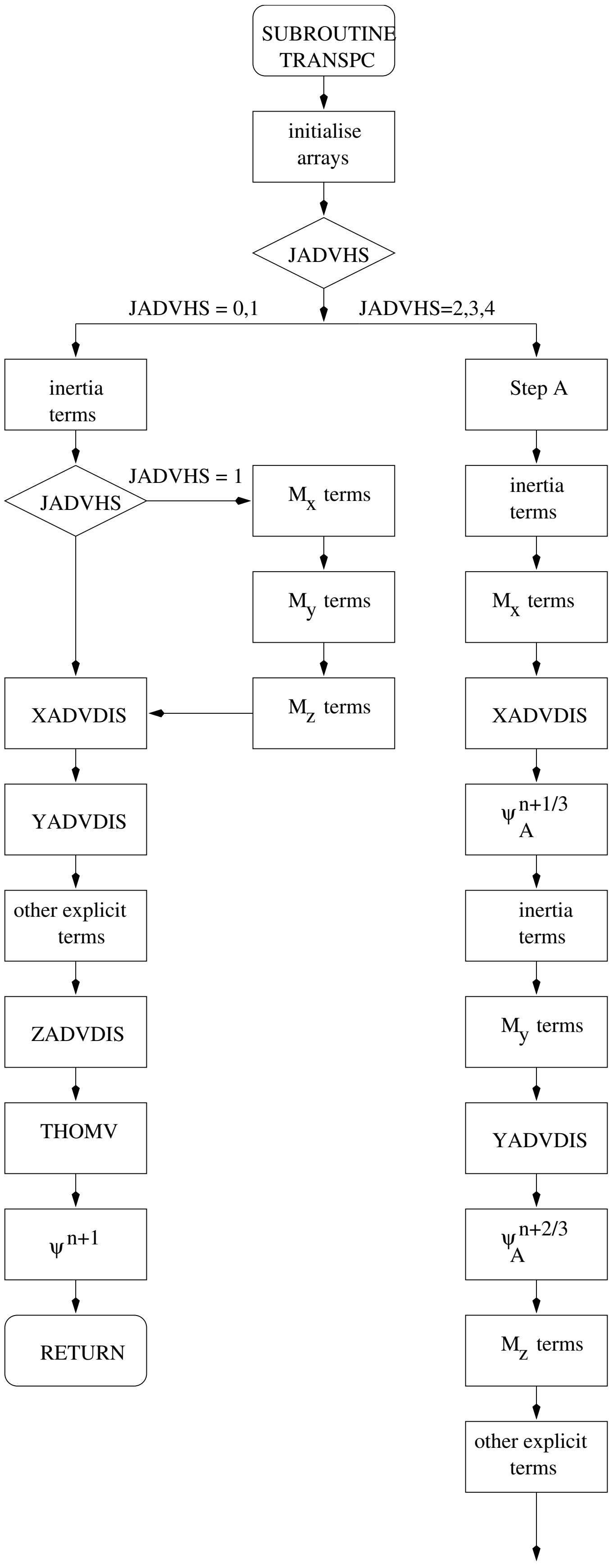


Figure 5.1.8: Flow chart of subroutine TRANSPC.

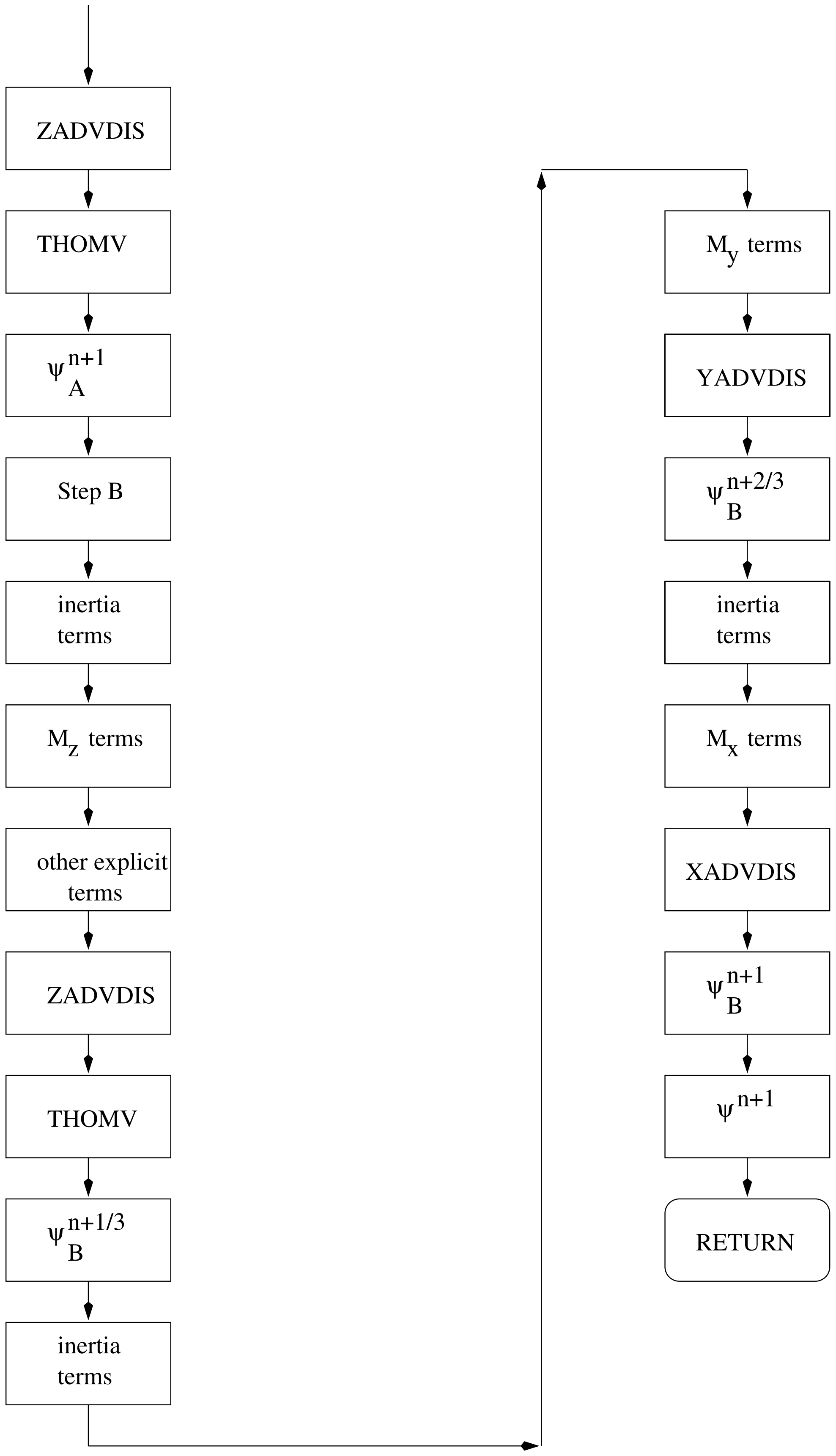


Figure 5.1.8: continued.

Purpose

solve the general transport equation (3.1.70) for a scalar quantity ψ evaluated at the W-nodes

Calling program

DISSIP, TKLENG, TURBEN

Externals

ERROR, THOMV, XADVDIS, YADVDIS, ZADVDIS, ZERO

Arguments

PHI: old value of ψ (W-nodes) at t^n on input, new value of ψ at t^{n+1} on output (unit is e.g. Q)

VEDW: vertical diffusion coefficient at W-nodes (m^2/s)

SOURCE: the source term $\mathcal{P}(\psi)$ (W-nodes) in the transport equation (3.1.70)

SINK: the sink term $\mathcal{S}(\psi)$ (W-nodes) in the transport equation (3.1.70) divided by ψ, i.e. $\mathcal{S}(\psi)/\psi$ (s^{-1})

CALL1: logical flag to determine whether TRANSPW is called for the first time from the calling routine

ISUR: switch to determine the type of surface boundary condition (see description of ZADVDIS for details)

IBOT: switch to determine the type of bottom boundary condition (see description of ZADVDIS for details)

JADVHS: switch to select the type of scheme for horizontal advection

0: no advection
1: upwind
2: Lax-Wendroff
3: TVD with superbee limiting function
4: TVD with monotonic limiting function

JADVWS: switch to select the type of scheme for vertical advection

0: no advection
1: upwind
2: central
3: TVD with superbee limiting function
4: TVD with monotonic limiting function

JDIFS: switch to select the type of scheme for horizontal diffusion

0: no diffusion
1: uniform coefficients

2: Smagorinsky formulation

IVPU: the same meaning as the array IVPOBU (see Section IV-2.5.1)

IVPV: the same meaning as the array IVPOBV (see Section IV-2.5.1)

PHIVP: "PSIVP"-array (Q) to be used at open boundaries (see Section IV-2.5.2)

SFLUX: parameter F^{ψ}_{s0} in the surface boundary condition (3.1.181), representing the surface flux (Q m/s)

STRAN: parameter F^{ψ}_{s1} in the surface boundary condition (3.1.181), representing the transfer coefficient (m/s)

SPHI: parameter ψ_{se} in the surface boundary condition (3.1.181), representing the external value (Q)

BFLUX: parameter F^{ψ}_{b0} in the bottom boundary condition (3.1.191), representing the bottom flux (Q m/s)

BTRAN: parameter F^{ψ}_{b1} in the bottom boundary condition (3.1.191), representing the transfer coefficient (m/s)

BPHI: parameter ψ_{be} in the bottom boundary condition (3.1.191), representing the external value (Q)

Local variables

GZ2PHI: vertical grid spacing at the W-nodes

HEDPHIU: horizontal diffusion coefficient λ_H at the X-nodes

HEDPHIV: horizontal diffusion coefficient λ_H at the Y-nodes

PHIA: value of ψ_A (step A)

PHIB: value of ψ_B (step B)

PLHSA: A-vector in the tridiagonal system (ψ_{k-1}-terms)

PLHSB: B-vector in the tridiagonal system (ψ_k-terms)

PLHSC: C-vector in the tridiagonal system (ψ_{k+1}-terms)

PRHS: D-vector in the tridiagonal system (explicit terms)

UCORR: corrector term $\mathcal{M}_x(\psi)$

U2PHI: advecting current u^F at the X-nodes

VCORR: corrector term $\mathcal{M}_y(\psi)$

VEDPHI: vertical diffusion coefficient λ^{ψ}_T at the cell centres

V2PHI: advecting current v^F at the Y-nodes

WCORR: corrector term $\mathcal{M}_z(\psi)$

W2PHI: vertical current $J\tilde{w}$ at the cell centres

Description

The numerical solution proceeds in (almost) the same way as in routine TRANSPC. The following differences are noted:

- Due to the staggering of the variables on the vertical grid, a number of arrays are interpolated:
  - horizontal advective velocity u^F and horizontal diffusion coefficient λ_H (for the fluxes in the X-direction) at the X-nodes
  - horizontal advective velocity v^F and horizontal diffusion coefficient λ_H (for fluxes in the Y-direction) at the Y-nodes
  - vertical current $J\tilde{\omega}$ and vertical diffusion coefficient λ_T^ψ at the cell centres
  - vertical grid spacing Δx_3 at the W-nodes
- The source terms $\mathcal{P}(\psi)$ (stored in the array SOURCE) are taken explicitly whereas the sink terms $\mathcal{S}(\psi)$ (stored in the array SINK after division by ψ) are evaluated quasi-implicitly.
- Note that, for computational efficiency, cell-centered variables in TRANSPW have the same vertical index as the quantities at the W-node above the grid point. This means that their vertical index is increased by 1 and varies between 2 (cell nearest to the bottom) and NZ+1 (cell nearest to the surface).

Remarks

- Since ψ, $\mathcal{P}(\psi)$ and $\mathcal{S}(\psi)$ in (3.1.70) are non-negative by definition, SOURCE and SINK must always be non-negative. $\mathcal{P}(\psi)$ is evaluated explicitly and added to PRHS. $\mathcal{S}(\psi)$ is evaluated quasi-implicitly and added to PLHSB after division by ψ.
- Important to note is that the Neumann surface or bottom conditions are not implemented in the current version of TRANSPW. This means that ISUR and IBOT may only take the values 1 or 2 and that the arguments SFLUX, STRAN, BFLUX and BTRAN are not used by the routine (see description of ZADVDIS below for further details).
- Note that all horizontal and vertical advective terms are cancelled if JADVHS $= 0$.
- Although JADVHS and JADVWS are separate arguments in the call to TRANSPW, they must have the same value.
- The arrays COSPHI and COSPHIV, used in the evaluation of $\mathcal{M}_y(\psi)$, represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- For more details about the numerical procedures see Section III-4.5.

TRANSV

File
: *transv.f*

Type
: subroutine

Purpose
: update the vertical grid spacing

Calling program
: INITC, MAINPROG, PREPROC

Externals
: none

Arguments
: none

Description
: The routine updates the vertical grid spacings given by $\Delta x_3 = H\Delta\sigma$ using the latest update of the surface elevation and stores the result in the array GZ2.

Remarks
: Note that the vertical grid spacing GZ2 equals the Jacobian J defined by (3.1.21).

XADVDIS

File
: *advdis.f*

Type
: subroutine

Purpose
: evaluate the advective and diffusion terms along the X-direction in the transport equation of a scalar quantity ψ

Calling program
: TRANSPC, TRANSPW

Externals
: FNLIM, ZERO

Arguments

- PHI: scalar quantity ψ
- HEDPHI: horizontal diffusion coefficient λ_H at the U- or X-nodal points where the fluxes are determined
- UADV: advective current u^F at the U- or X-nodal points where the fluxes are determined (m/s)
- GZ: vertical grid spacing at the points where ψ is evaluated (m)
- NZPHI: vertical dimension of ψ (NZ or NZ+1)
- IVPU: the same meaning as the array IVPOBU (see Section IV-2.5.1)
- PHIVP: "PSIVP"-array to be used at open boundaries (see Section IV-2.5.2)
- JADVS: switch to select the type of scheme for horizontal advection
  - 0: no advection
  - 1: upwind
  - 2: Lax-Wendroff
  - 3: TVD with superbee limiting function
  - 4: TVD with monotonic limiting function
- JDIFS: switch to select the type of scheme for horizontal diffusion
  - 0: no diffusion
  - 1: uniform coefficients
  - 2: Smagorinsky formulation
- SIMP: B-vector in the tridiagonal equation system (ψ_k-terms)
- SEXP: D-vector in the tridiagonal equation system (explicit terms)

Local variables

- ADVFL: advective flux F_x^u or F_x^X
- ADVFLW: Lax-Wendroff flux F_{lw}^u or F_{lw}^X
- ADVFUP: upwind flux F_{up}^u or F_{up}^X
- DIFFL: diffusion flux D_x^u or D_x^X
- CFL: CFL-number
- PSI: weight factor $\Omega(r)$

Description

It should be remarked first that the fluxes are calculated at the U-nodes if XADVDIS is called from TRANSPC and at the X-nodes if XADVDIS is called from TRANSPW.

- Initialise the advective and diffusion fluxes to zero.
- Evaluate the upwind and Lax-Wendroff fluxes at all interior U- or X-nodes using (3.4.213) and (3.4.214).

- Evaluate the advective flux using (3.4.212).
- Determine the advective fluxes at U- or X-open boundaries using the discretised form (3.4.248) or (3.4.252). The type of condition (scalar or zero gradient) and the external profile ψ_{ext} are previously stored in the array PHIVP (see Section IV-2.5.2 for details). The results are added to the arrays SIMP and SEXP using the prescriptions (3.4.269) or (3.4.271).
- Evaluate the horizontal diffusion flux at all interior U- or X-nodes using (3.4.226).
- Determine the diffusion fluxes at U- or X-open boundaries using the discretised form (3.4.250) or (3.4.254). The type of condition (scalar or zero gradient) and the external profile ψ_{ext} are previously stored in the array PHIVP (see Section IV-2.5.2 for details). The results are, in the case of a scalar condition, added to the arrays SIMP and SEXP using the prescription (3.4.269).
- The advective term is obtained by differencing the advective fluxes in the X-direction using (3.4.211). The result, multiplied by $-\Delta t_{3D}$, is added to the array SEXP.
- The diffusion term is obtained by differencing the diffusion fluxes in the X-direction using (3.4.225). The result, multiplied by Δt_{3D}, is added to the array SEXP.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that if an interior U- or X-node is only one grid distance away from an open/solid boundary and the advective current is directed away from that open boundary, $\Omega(r)$ is set to zero (upwind scheme) in FNLIM (as imposed by (3.4.255)).
- Advective terms are calculated if JADVS > 0, diffusion terms if JDIFS > 0.
- All terms are evaluated explicitly except for the advective flux terms at open boundaries which are taken implicitly.
- For more details about the numerical procedures see Section III-4.4.

XHAD3DU

File

hadvdis.f

Type

subroutine

Purpose

evaluate the advective and diffusion terms $\mathcal{A}_{hx}(u)$, $\mathcal{D}_{xx}(u)$ along the X-direction in the u-momentum equation (3.1.25) or (3.1.50) and the depth integral of $\mathcal{D}_{xx}(u) - \mathcal{A}_{hx}(u)$ as used in the discretised expressions (3.4.110) (or (3.4.348)) and (3.4.83) (or (3.4.346)) for the integral quantity $\overline{D}^h_1 - \overline{A}^h_1$ (or $\overline{D}^h_\lambda - \overline{A}^h_\lambda$)

Calling program

UCALC

Externals

FNLIM, ZERO

Arguments

U2PHI: u-current (quantity to be advected/diffused) at the U-nodes (m/s)

UADV: advective u-velocity at the U-nodes (m/s)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

XINTU: the depth integral of $\mathcal{D}_{xx}(u) - \mathcal{A}_{hx}(u)$

JDIFC: switch to select the type of scheme for horizontal diffusion

0: no diffusion

1: uniform coefficients

2: Smagorinsky formulation

Local variables

ADVFL: advective flux F^c_{xx}

ADVFLW: Lax-Wendroff flux F^c_{lw}

ADVFUP: upwind flux F^c_{up}

XDIFLUJ: diffusion flux D^c_{xx}

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

- Initialise the advective and diffusion fluxes to zero.
- Evaluate the upwind and Lax-Wendroff fluxes F^c_{up} and F^c_{lw} at the cell centres using (3.4.49) and (3.4.50).
- Evaluate the advective flux F^c_{xx} using (3.4.48).
- Evaluate the diffusion flux D^c_{xx} at the cell centres using either (3.4.89) or (3.4.340).

- The advective and diffusion terms $\mathcal{A}_{hx}(u)$ and $\mathcal{D}_{xx}(u)$ are obtained by differencing the advective and diffusion fluxes in the X-direction using the discretisations (3.4.47) and (3.4.92). The result, multiplied by $\pm\Delta t_{3D}$, is added to the array SEXP.
- The quantity $\mathcal{D}_{xx}(u) - \mathcal{A}_{hx}(u)$ is integrated vertically and stored in XINTU.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the fluxes are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the cell centre is only one-half grid distance away from an open/solid boundary and the advective current is pointing away from that boundary (as imposed by (3.4.159)).
- The arrays SPHCUR and SPHCURV represent the value of $\tan\phi/R$ in the spherical case and are set to zero in the Cartesian case (IGTRH $= 0$).
- Advective terms are calculated if IADVC > 0, diffusion terms if JDIFC > 0.
- For more details about the numerical procedures see Section III-4.3.

XHAD3DV

File

hadvdis.f

Type

subroutine

Purpose

evaluate the advective and diffusion terms $\mathcal{A}_{hx}(v)$, $\mathcal{D}_{xy}(u, v)$ (including additional terms in the spherical case) along the X-direction in the v-momentum equation (3.1.26) or (3.1.51) and the depth integral of $\mathcal{D}_{xy}(u, v) - \mathcal{A}_{hx}(v)$ (including the extra terms in the spherical case) as used in the discretised expressions (3.4.111) (or (3.4.349)) and (3.4.84) (or (3.4.347)) for the integral quantity $\overline{D}_2^h - \overline{A}_2^h$ (or $\overline{D}_\phi^h - \overline{A}_\phi^h$)

Calling program

VCALC

Externals

FNLIM, ZERO

Arguments

V2PHI: v-current (quantity to be advected/diffused) at the V-nodes (m/s)

UADV: advective u-velocity at the U-nodes (m/s)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

XINTV: the depth integral of $\mathcal{D}_{xy}(u,v) - \mathcal{A}_{hx}(v)$ (including the extra terms in the spherical case)

JDIFC: switch to select the type of scheme for horizontal diffusion

0: no diffusion

1: uniform coefficients

2: Smagorinsky formulation

Local variables

ADVFLN: advective flux F^N_{xy}

ADVFLS: advective flux F^S_{xy}

ADVFLWN: Lax-Wendroff flux F^N_{lw}

ADVFLWS: Lax-Wendroff flux F^S_{lw}

ADVFUPN: upwind flux F^N_{up}

ADVFUPS: upwind flux F^S_{up}

XDIFLU: diffusion flux $\tilde{D}^c_{xx}$ (spherical case only)

XDIFLVJ: diffusive flux $D^{V;u}_{xy}$

YDIFLUJ: diffusive flux $D^{U;v}_{xy}$

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

- Initialise the advective and diffusion fluxes to zero.
- Step 1: advective fluxes
  - The advective flux uv is determined at the U-nodes in two ways. Firstly, the upwind and Lax-Wendroff fluxes are evaluated using values for the advected quantity v at the V-nodes South of the U-node (points S_1 and S_2 in Figure 5.1.7c) as given by (3.4.73) and (3.4.74). The flux F^S_{xy}, derived from (3.4.76), is stored in the local array ADVFLS. The same procedure is repeated now using v-values North of the U-node (points N_1 and N_2 in Figure 5.1.7c). The corresponding expressions are (3.4.78), (3.4.79) and (3.4.80). The advective flux F^N_{xy} is stored in ADVFLN.
  - The open boundary values of ADVFLS and ADVFLN are obtained using the upwind scheme and a zero gradient condition in the case of inflow, as given by (3.4.155) and (3.4.156).
- Step 2: diffusion fluxes

- The flux D^V_{xy}, given by either (3.4.91) or (3.4.341) (in the spherical case) is evaluated at the U-nodes using v-values interpolated at the centre. The result is stored in XDIFLVJ.
- The flux D^U_{xy}, given by (3.4.90), is evaluated at the V-nodes using u-values interpolated at the centre. The result is stored in YDIFLUJ.
- In the spherical case, the flux D^c_{xx}, given by (3.4.89) (without the J-term) is evaluated at the cell centre and stored in XDIFLU.
- Open boundary values of XDIFLVJ and YDIFLUJ are set to their nearest interior values, using the prescriptions (3.4.157) and (3.4.158).

• Step 3: advection terms at interior U-nodes
  - The X-derivative term is derived by taking the difference of the fluxes at the corner nodes C_1 and C_2 (see Figure 5.1.7d). The corner fluxes at C_2 (C_1) are given by the average of the "South"-flux at S_{f2} (S_{f1}) and the "North"-flux at N_{f2} (N_{f1}). See equation (3.4.82) for details.
  - The term $u^2 \tan\phi/R$ is added in the spherical case.
  - The advective term $-\mathcal{A}_{hx}(v)$, times Δt_{3D}, is added to SEXP.
  - The advective term (with the minus sign) is integrated vertically and stored into XINTV.

• Step 4: diffusion terms at interior V-nodes
  - The X-derivative of τ_{12} ($\tau_{\lambda\phi}$) is obtained by interpolating XDIFLVJ at the corners and differencing, and by differencing YDIFLUJ over two grid cells in the X-direction (using a special procedure near the edges of the computational domain). See equations (3.4.99) and (3.4.100) for details.
  - In the spherical case, the term $-2\nu_H v \tan\phi\,/R$ is added to D^c_{xx}. The result, multiplied by Δt_{3D}, is added to the diffusion term.
  - The diffusion term $\mathcal{D}_{xy}(u,v)$, times Δt_{3D}, is added to SEXP.
  - The diffusion term is integrated vertically and added to XINTV.

Remarks

• The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the southern and northern fluxes are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the U-node is only one grid distance away from an open/solid boundary and the advective current is pointing away from that boundary (as imposed by (3.4.162)).
• The arrays SPHCUR and SPHCURV represent the value of $\tan\phi/R$ in the spherical case and are set to zero in the Cartesian case (IGTRH = 0).
• Advective terms are calculated if IADVC > 0, diffusive terms if JDIFC > 0.
• For more details about the numerical procedures see Section III-4.3.

YADVDIS

File
: *advdis.f*

Type
: subroutine

Purpose
: evaluate the advective and diffusion terms along the Y-direction in the transport equation of a scalar quantity ψ

Calling program
: TRANSPC, TRANSPW

Externals
: FNLIM, ZERO

Arguments

PHI: scalar quantity ψ

HEDPHI: horizontal diffusion coefficient λ_H at the V-or Y-nodal points where the fluxes are determined

VADV: advective current v^F at the V-or Y-nodal points where the fluxes are determined (m/s)

GZ: vertical grid spacing at the points where ψ is evaluated (m)

NZPHI: vertical dimension of ψ (NZ or NZ+1)

IVPV: the same meaning as the array IVPOBV (see Section IV-2.5.1)

PHIVP: "PSIVP"-array to be used at open boundaries (see Section IV-2.5.2)

JADVS: switch to select the type of scheme for horizontal advection

- 0: no advection
- 1: upwind
- 2: Lax-Wendroff
- 3: TVD with superbee limiting function
- 4: TVD with monotonic limiting function

JDIFS: switch to select the type of scheme for horizontal diffusion

- 0: no diffusion
- 1: uniform coefficients
- 2: Smagorinsky formulation

SIMP: B-vector in the tridiagonal equation system (ψ_k-terms)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

Local variables

ADVFL: advective flux F_y^v or F_y^Y

ADVFLW: Lax-Wendroff flux F_{lw}^v or F_{lw}^Y

ADVFUP: upwind flux F_{up}^v or F_{up}^Y

DIFFL: diffusion flux D_y^v or D_y^Y

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

It should be remarked first that the fluxes are calculated at the V-nodes if YADVDIS is called from TRANSPC and at the Y-nodes if YADVDIS is called from TRANSPW.

- Initialise the advective and diffusion fluxes to zero.
- Evaluate the upwind and Lax-Wendroff fluxes at all interior V- or Y-nodes using (3.4.219) and (3.4.220).
- Evaluate the advective flux using (3.4.218).
- Determine the advective fluxes at V- or Y-open boundaries using the discretised form (3.4.249) or (3.4.253). The type of condition (scalar or zero gradient) and the external profile ψ_{ext} are previously stored in the array PHIVP (see Section IV-2.5.2 for details). The results are added to the arrays SIMP and SEXP using the prescriptions (3.4.270) or (3.4.272).
- Evaluate the horizontal diffusion flux at all interior V- or Y-nodes using (3.4.228).
- Determine the diffusion fluxes at V- or Y-open boundaries using the discretised form (3.4.251) or (3.4.254). The type of condition (scalar or zero gradient) and the external profile ψ_{ext} are previously stored in the array PHIVP (see Section IV-2.5.2 for details). The results are, in the case of a scalar condition, added to the arrays SIMP and SEXP using the prescription (3.4.270).
- The advective term is obtained by differencing the advective fluxes in the Y-direction using (3.4.217). The result, multiplied by $-\Delta t_{3D}$, is added to the array SEXP.
- The diffusion term is obtained by differencing the diffusion fluxes in the Y-direction using (3.4.227). The result, multiplied by Δt_{3D}, is added to the array SEXP.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that if an interior V- or Y-node is only one grid distance away from an open/solid boundary and the advective current is directed away from that open boundary, $\Omega(r)$ is set to zero (upwind scheme) in FNLIM (as imposed by (3.4.256)).
- Advective terms are calculated if JADVS > 0, diffusion terms if JDIFS > 0.

- All terms are evaluated explicitly except for the advective flux terms at open boundaries which are taken implicitly.
- The arrays COSPHI and COSPHIV represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- For more details about the numerical procedures see Section III-4.4.

YHAD3DU

File

hadvdis.f

Type

subroutine

Purpose

evaluate the advective and diffusion terms $\mathcal{A}_{hy}(u)$, $\mathcal{D}_{yx}(u,v)$ (including additional terms in the spherical case) along the Y-direction in the u-momentum equation (3.1.25) or (3.1.50) and the depth integral of $\mathcal{D}_{yx}(u,v)-\mathcal{A}_{hy}(u)$ (including extra terms in the spherical case) as used in the discretised equations (3.4.110) (or (3.4.348)) and (3.4.83) (or (3.4.346)) for the integral quantity $\overline{D}_1^h - \overline{A}_1^h$ (or $\overline{D}_\lambda^h - \overline{A}_\lambda^h$)

Calling program

UCALC

Externals

FNLIM, ZERO

Arguments

U2PHI: u-current (quantity to be advected/diffused) at the U-nodes (m/s)

VADV: advective v-velocity at the V-nodes (m/s)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

YINTU: the depth integral of $\mathcal{D}_{yx}(u,v) - \mathcal{A}_{hy}(u)$ (including extra terms in the spherical case)

JDIFC: switch to select the type of scheme for horizontal diffusion

0: no diffusion

1: uniform coefficients

2: Smagorinsky formulation

Local variables

ADVFLE: advective flux F^E_{yx}

ADVFLW: advective flux F^W_{yx}

ADVFLWE: Lax-Wendroff flux F^E_{lw}

ADVFLWW: Lax-Wendroff flux F^W_{lw}

ADVFUPE: upwind flux F^E_{up}

ADVFUPW: upwind flux F^W_{up}

XDIFLV: diffusion flux $\tilde{D}^{V;u}_{yx}$ (spherical case only)

XDIFLVJ: diffusive flux $D^{V;u}_{yx}$

YDIFLU: diffusion flux $\tilde{D}^{U;v}_{yx}$ (spherical case only)

YDIFLUJ: diffusive flux $D^{U;v}_{yx}$

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

- Initialise the advective and diffusion fluxes to zero.
- Step 1: advective fluxes
  - The advective flux vu is determined at the V-nodes in two ways. Firstly, the upwind and Lax-Wendroff fluxes are evaluated using values for the advected quantity u at the U-nodes West of the V-node (points W_1 and W_2 in Figure 5.1.7a), as given by (3.4.63) and (3.4.64). The flux F^W_{yx}, derived from (3.4.66), is stored in the local array ADVFLW. The same procedure is repeated now using u-values East of the V-node (points E_1 and E_2 in Figure 5.1.7a). The corresponding expressions are (3.4.68), (3.4.69) and (3.4.70). The advective flux F^E_{yx} is stored in ADVFLE.
  - The open boundary values of ADVFLW and ADVFLE are obtained using the upwind scheme and a zero gradient condition in the case of inflow, as given by (3.4.153) and (3.4.154).
- Step 2: diffusion fluxes
  - The flux D^U_{yx} given by (3.4.90), is evaluated at the V-nodes using u-values interpolated at the centre. The result is stored in YDIFLUJ. The flux $\tilde{D}^U_{yx}$ (the same as D^U_{yx} but without the J-term) is stored in YDIFLU (spherical case).
  - The flux D^V_{yx}, given by (3.4.91) (or (3.4.341) in the spherical case), is evaluated at the U-nodes using v-values interpolated at the centre. The result is stored in XDIFLVJ. The flux $\tilde{D}^V_{yx}$ (the same as D^V_{yx} but without the J-term) is stored in XDIFLV (spherical case).

  - Open boundary values of YDIFLUJ, XDIFLVJ and YDIFLU, XDIFLV are set to their nearest interior values, using the prescriptions (3.4.157)–(3.4.158) and (3.4.354)–(3.4.355).

- Step 3: advection terms at interior U-nodes
  - The Y-derivative term is derived by taking the difference of the fluxes at the corner nodes C_1 and C_2 (see Figure 5.1.7b). The corner fluxes at C_2 (C_1) are given by the average of the "West"-flux at W_{f2} (W_{f1}) and the "East"-flux at E_{f2} (E_{f1}). See equation (3.4.72) for details.
  - The term $-uv\tan\phi/R$ is added in the spherical case.
  - The advective term $-\mathcal{A}_{yx}(u)$, times Δt_{3D}, is added to SEXP.
  - The advective term is integrated vertically (with the minus sign) and stored into YINTU.
- Step 4: diffusion terms at interior U-nodes
  - The Y-derivative of τ_{21} ($\tau_{\phi\lambda}$) is obtained by interpolating YDIFLUJ at the corners and differencing, and by differencing XDIFLVJ over two grid cells in the Y-direction (using a special procedure near the edges of the computational domain). See equations (3.4.93) and (3.4.94) for details.
  - The term (3.4.342) is added to the diffusion term in the spherical case.
  - The diffusion term $\mathcal{D}_{yx}(u,v)$, times Δt_{3D}, is added to SEXP.
  - The diffusion term is integrated vertically and added to YINTU.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the western and eastern fluxes are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the V-node is only one grid distance away from an open/solid boundary and the advective current is pointing away from that boundary (as imposed by (3.4.161)).
- The arrays SPHCUR and SPHCURV represent the value of $\tan\phi/R$ in the spherical case and are set to zero in the Cartesian case (IGTRH = 0).
- The arrays COSPHI and COSPHIV represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH = 0).
- Advective terms are calculated if IADVC > 0, diffusive terms if JDIFC > 0.
- For more details about the numerical procedures see Section III-4.3.

YHAD3DV

File

hadvdis.f

Type

subroutine

Purpose

evaluate the advective and diffusion terms $\mathcal{A}_{hy}(v)$, $\mathcal{D}_{yy}(v)$ along the Y-direction in the v-momentum equation (3.1.26) or (3.1.51) and the depth integral of $\mathcal{D}_{yy}(v)-\mathcal{A}_{hy}(v)$ as used in the discretised expressions (3.4.111) (or (3.4.349)) and (3.4.84) (or (3.4.347)) for the integral quantity $\overline{D}_2^h - \overline{A}_2^h$ (or $\overline{D}_\phi^h - \overline{A}_\phi^h$)

Calling program

VCALC

Externals

FNLIM, ZERO

Arguments

V2PHI: v-current (quantity to be advected/diffused) at the V-nodes (m/s)

VADV: advective v-velocity at the V-nodes (m/s)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

YINTV: the depth integral of $\mathcal{D}_{yy}(v) - \mathcal{A}_y(v)$

JDIFC: switch to select the type of scheme for horizontal diffusion

0: no diffusion

1: uniform coefficients

2: Smagorinsky formulation

Local variables

ADVFL: advective flux F^c_{yy}

ADVFLW: Lax-Wendroff flux F^c_{lw}

ADVFUP: upwind flux F^c_{up}

YDIFLVJ: diffusion flux D^c_{yy}

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

- Initialise the advective and diffusion fluxes to zero.
- Evaluate the upwind and Lax-Wendroff fluxes F^c_{up} and F^c_{lw} at the cell centres using (3.4.59) and (3.4.60).
- Evaluate the advective flux F^c_{yy} using (3.4.58).
- Evaluate the diffusion flux D^c_{yy} at the cell centres using (3.4.95).

- The advective and diffusion terms $\mathcal{A}_{hy}(v)$ and $\mathcal{D}_{yy}(v)$ are obtained by differencing the advective and diffusion fluxes in the Y-direction using the discretisations (3.4.57) and (3.4.98). The result, multiplied by Δt_{3D}, is added to the array SEXP.
- The quantity $\mathcal{D}_{yy}(v) - \mathcal{A}_{hy}(v)$ is integrated vertically and stored into YINTV.

Remarks

- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the fluxes are obtained with the upwind scheme ($\Omega(r) = 0$ in FNLIM) if the cell centre is only one-half grid distance away from an open/solid boundary and the advective current is pointing away from that boundary (as imposed by (3.4.160)).
- The arrays COSPHI and COSPHIV, represent the value of $\cos\phi$ in the case of a spherical grid and are set to 1 in the Cartesian case (IGTRH $= 0$).
- Advective terms are calculated if IADVC > 0, diffusion terms if JDIFC > 0.
- For more details about the numerical procedures see Section III-4.3.

ZADVDIS

File

advdis.f

Type

subroutine

Purpose

evaluate the vertical advective and diffusion terms in the transport equation (3.1.70) of a scalar quantity, the u-equation (3.1.25) or (3.1.50) and the v-equation (3.1.26) or (3.1.51)

Calling program

TRANSPC, TRANSPW, UCALC, VCALC

Externals

FNLIM, ZERO

Arguments

PHI: scalar quantity (e.g. temperature, biological concentration or turbulence variable), u-current or v-current further denoted by ψ (unit is e.g. Q which may be m/s, ^{0}C, PSU or some concentration)

VEDPHI: vertical diffusion coefficient at the points where the fluxes are determined (m^2/s)

WADV: vertical advective current (including sinking velocities) at the points where the fluxes are determined (m/s)

GZ: vertical grid spacing around the points where ψ is evaluated (m)

IBLOCK: pointer array to eliminate dry cells or interfaces from the calculations (equal to NWD if ZADVDIS is called from TRANSPC or TRANSPW, to NPIX if called from UCALC and to NPIY if called from VCALC)

ISUR: type of surface boundary condition

- 0 : Neumann condition of the form (3.1.181)
- 1 : Dirichlet condition at the surface
- 2 : Dirichlet condition at one grid distance below the surface

IBOT: type of bottom boundary condition

- 0 : Neumann condition of the form (3.1.191)
- 1 : Dirichlet condition at the bottom
- 2 : Dirichlet condition at one grid distance above the sea bed

NZPHI: vertical dimension of the array ψ (NZ or NZ+1)

NRPHI: Y-dimension of the array ψ (NR or NR+1)

NCPHI: X-dimension of the array ψ (NC or NC+1)

SFLUX: parameter F^{ψ}_{s0} in the surface boundary condition (3.1.181), representing the surface flux (Q m/s)

STRAN: parameter F^{ψ}_{s1} in the surface boundary condition (3.1.181), representing the transfer coefficient (m/s)

SPHI: parameter ψ_{se} in the surface boundary condition (3.1.181), representing the external value (Q)

BFLUX: parameter F^{ψ}_{b0} in the bottom boundary condition (3.1.191), representing the bottom flux (Q m/s)

BTRAN: parameter F^{ψ}_{b1} in the bottom boundary condition (3.1.191), representing the transfer coefficient (m/s)

BPHI: parameter ψ_{be} in the bottom boundary condition (3.1.191), representing the external value (Q)

JADVW: switch to select the type of scheme for vertical advection

- 0: no advection
- 1: upwind
- 2: central
- 3: TVD with superbee limiting function

4: TVD with monotonic limiting function

SIMPA: A-vector in the tridiagonal equation system (ψ_{k-1}-terms)

SIMPB: B-vector in the tridiagonal equation system (ψ_k-terms)

SIMPC: C-vector in the tridiagonal equation system (ψ_{k+1}-terms)

SEXP: D-vector in the tridiagonal equation system (explicit terms)

Local variables

ADVUP: upwind flux F_{up} (used only for the evaluation of the weight factor $\Omega(r)$)

ADVCE: central flux F_{ce} (used only for the evaluation of the weight factor $\Omega(r)$)

DTHIMP: implicity factor for vertical diffusion θ_v (0: fully explicit, 1: fully implicit)

THIMP: implicity factor for vertical advection θ_a (0: fully explicit, 1: fully implicit)

CFL: CFL-number

PSI: weight factor $\Omega(r)$

Description

The quantity ψ is evaluated at the cell centers, U-nodes, V-nodes or W-nodes if ZADVDIS is called from respectively TRANSPC, UCALC, VCALC or TRANSPW. The fluxes are evaluated vertically between the points where ψ is located. A schematic presentation of the vertical indexing system is shown in Figure 5.1.9.

- Step 1: vertical advection term
  - Initialise the advective fluxes to zero.
  - Evaluate the upwind and Lax-Wendroff fluxes at the old time using a formulation which reduces to (3.4.116)–(3.4.117) (u-current), (3.4.123)–(3.4.124) (v-current), (3.4.236)–(3.4.237) (scalar ψ).
  - Calculate the weight factor $\Omega(r)$ at each vertical level using old values of ψ.
  - As indicated in Section III-4.3.12b, III-4.4.8b and III-4.5.5b the vertical advection term $\mathcal{A}_v(\psi)$ or $\tilde{\mathcal{A}}_v(\psi)$ is composed of a part related to the flux, e.g. F^a_k, below the ψ_k-level and a part related to the flux F^a_{k+1} above the ψ_k-level. The corresponding terms are added to the arrays SIMPA (A-vector), SIMPB (B-vector), SIMPC (C-vector), SEXP (D-vector) of the tridiagonal matrix using the expressions (3.4.178) and (3.4.181) (u- or v-current), (3.4.260) and (3.4.263) (scalar ψ at the centre), (3.4.313) and (3.4.316) (scalar ψ at the W-node).
- Step 2: vertical diffusion terms
  - A similar procedure is applied for the vertical diffusion term $\mathcal{D}_v(\psi)$, which is composed of part related to the flux, e.g. F^d_k, below the ψ_k-level and a part related to the flux F^d_{k+1} above the ψ_k-level (see Section III-4.3.12c, III-4.4.8c and III-4.5.5c). The corresponding terms are added to the arrays SIMPA (A-vector), SIMPB (B-vector), SIMPC (C-vector), SEXP (D-vector)

of the tridiagonal matrix using the expressions (3.4.183) and (3.4.184) (u- or v-current), (3.4.265) and (3.4.266) (scalar ψ at the centre), (3.4.318) and (3.4.319) (scalar ψ at the W-node).

- The surface boundary condition depends on the value of ISUR:
  - 0: Neumann condition of the form (3.1.181) where the $\psi_{s,i}$-term is taken implicitly and added to SIMPB, and the other terms are added to SEXP, as given by the second relation of (3.4.185) (u- or v-current) and by (3.4.267) (scalar ψ at the centre).
  - 1: Dirichlet condition with a prescribed surface value (SPHI) which is stored in SEXP, as given by (3.4.320) (k-equation).
  - 2: Dirichlet condition with a prescribed value (SPHI) one grid distance below the sea surface which is stored in SEXP, as given by (3.4.324) (kl-equation) and (3.4.326) (ε-equation). The surface value is set to zero (see equation (3.4.322)).
- The bottom boundary condition depends on the value of IBOT:
  - 0: Neumann condition of the form (3.1.191) where the $\psi_{b,i}$-term is taken implicitly and added to SIMPB, and the other terms are added to SEXP, as given by the first relation (3.4.185) (u- or v-current) and by (3.4.268) (scalar ψ at the centre).
  - 1: Dirichlet condition with a prescribed bottom value (BPHI) which is stored in SEXP, as given by (3.4.321) (k-equation).
  - 2: Dirichlet condition with a prescribed bottom value (BPHI) one grid distance above the sea bed which is stored in SEXP, as given by (3.4.325) (kl-equation) and (3.4.327) (ε-equation). The bottom value is set to zero (see equation (3.4.323)).

Remarks

- The implicity factor θ_a is set to 0.501 giving a semi-implicit formulation for vertical advection. Vertical diffusion is evaluated fully implicitly ($\theta_v = 1$). The values of the corresponding FORTRAN variables THIMP and DTHIMP can only be changed in ZADVDIS.
- The surface and bottom values of the advective fluxes are taken to be zero in agreement with the conditions (3.1.169) and (3.1.188). Sinking into the bottom of organic material in the biological and sediment modules is taken into account in the FLUFF routine.
- The values ISUR = 2 and IBOT = 2 are only used for the solution of the kl- and ε-equations in the turbulence module.
- The weight factor $\Omega(r)$ is obtained by calling FNLIM. Note that the upwind scheme is used at the first grid point below (above) the surface (bottom) when the advective velocity is directed downwards (upwards).

- Vertical advection is only evaluated when JADVW > 0.
- For more details about the numerical procedures see Sections III-4.3, 4.4, 4.5.

1.9 Model setup

DEFAULT

File
: *preproc.f*

Type
: subroutine

Purpose
: default values for model setup

Calling program
: PREPROC

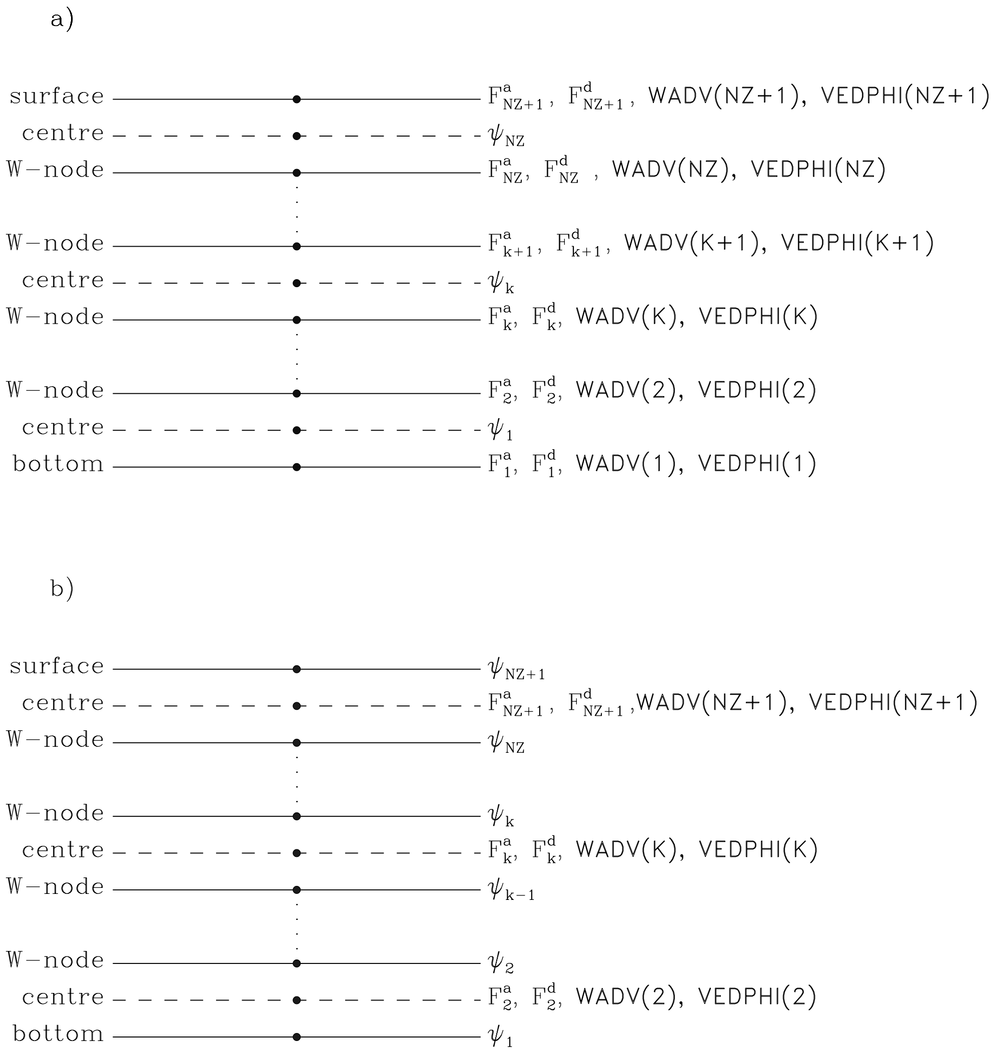
Externals
: IZERO, ZERO

Arguments
: none

Description
: Default values are set for:
  - all model switches and parameters which can be defined in DEFCON
  - the array GZ0 defining the (uniform) vertical location of the σ-grid
  - all open boundary ("PSIVP") arrays which are set for a normal gradient condition

  All other arrays, which can possibly be used for model input, are initialised to zero. The reason is that, according to standard FORTRAN, a variable (scalar or array) must be defined before it can be used in a WRITE statement.

Remarks
: - DEFAULT is called by PREPROC before any of the user-defined routines described in Chapter IV-2.
  - All parameters, defined in DEFAULT, can be redefined by the user.



a)
surface
centre
W−node
W−node
centre
W−node
W−node
centre
bottom
F^a_{NZ+1}, F^d_{NZ+1}, WADV(NZ+1), VEDPHI(NZ+1)
ψ_{NZ}
F^a_{NZ}, F^d_{NZ}, WADV(NZ), VEDPHI(NZ)
F^a_{k+1}, F^d_{k+1}, WADV(K+1), VEDPHI(K+1)
ψ_k
F^a_k, F^d_k, WADV(K), VEDPHI(K)
F^a_2, F^d_2, WADV(2), VEDPHI(2)
ψ_1
F^a_1, F^d_1, WADV(1), VEDPHI(1)
b)
surface
centre
W−node
W−node
centre
W−node
W−node
centre
bottom
ψ_{NZ+1}
F^a_{NZ+1}, F^d_{NZ+1},WADV(NZ+1), VEDPHI(NZ+1)
ψ_{NZ}
ψ_k
F^a_k, F^d_k, WADV(K), VEDPHI(K)
ψ_{k-1}
ψ_2
F^a_2, F^d_2, WADV(2), VEDPHI(2)
ψ_1

Figure 5.1.9: Vertical location of the advective/diffusion fluxes, the advective velocity and the vertical diffusion coefficient in module ZADVDIS for the case that ψ is evaluated at the cell centre (a) or at the W-nodes (b).

- Note that, for technical reasons, some default values (e.g. for salinity, temperature and grazing pressures) are set outside the routine in the main program unit of PREPROC.

DEFCON

File
: *defmod.f*

Type
: subroutine

Purpose
: define parameters for model setup (user-defined)

Calling program
: PREPROC

Externals
: none

Arguments
: none

Description
: The following parameters can be defined here by the user:
  - switches
  - date/time parameters
  - time step (2-D) and counters
  - file parameters
  - model parameters for the physics and the biological, sediment and particle modules

  For more details see Section IV-2.2.

Remarks
: - Note that some parameters (e.g. time step, start and end date, ...) need always to be defined so that DEFCON must always be non-empty.
  - The parameters in DEFCON, are written to the *con*-file.
  - A generic example of DEFCON is found in the **/examples** directory.

DEFGRID

File

defmod.f

Type

subroutine

Purpose

define grid arrays (user-defined)

Calling program

PREPROC

Externals

none

Arguments

none

Description

The following arrays can be defined here by the user:

- grid coordinates (X-, Y- and Z-direction)
- bathymetry
- pointer arrays (cell centre, U- and V-nodes)
- grid indices of the open boundary locations
- array of roughness lengths

For more details see Section IV-2.3.

Remarks

- Note that most arrays must always be defined by the user if IGRDIM = 3 in which case DEFGRID must be non-empty.
- The arrays, defined in DEFGRID, are written to the *grd*-file.
- A generic example of DEFGRID is found in the **/examples** directory.
- In the case of a 1-D application (IGRDIM = 1) all arrays, except for the vertical grid coordinates (array GZ0), are set by the program in DEFGRID1 so that DEFGRID is empty except when a (non-default) non-uniform vertical grid is selected by the user. Note that the parameters DEPUN and CDZ0UN (if IBSTR = 1) must be defined in DEFCON for a 1-D application.

DEFGRID1

File
: *preproc.f*

Type
: subroutine

Purpose
: define grid arrays in the case of a 1-D application

Calling program
: PREPROC

Externals
: none

Arguments
: none

Description
: The following grid arrays are set by the program for a 1-D application in the following way:

- The horizontal resolution is set by the program to an arbitrary value of 1 km which has no effect on the program, i.e.

$$x_{1;i} = 1000(i-1)\,,\; x_{2;j} = 1000(j-1) \tag{5.1.10}$$

- The array of mean water depths DEP is set to the value of DEPUN, defined in DEFCON.
- All grid cells are wet.
- All interior cell faces are wet.
- All domain boundaries are open.

Remarks

- If the user prefers to use an irregular grid spacing in the vertical, the array GZ0 must be defined in DEFGRID.
- The uniform water depth DEPUN must be defined in DEFCON.
- In the case of a quadratic friction law (IBSTR = 1), the bottom roughness length (parameter CDZ0UN) must be defined in DEFCON.
- DEFGRID1 is only called if IGRDIM = 1.

DEFICS

File
defmod.f

Type
subroutine

Purpose
define initial conditions (user-defined)

Calling program
PREPROC

Externals
none

Arguments
none

Description
The initial values of the following quantities can be defined here by the user:

- salinity and temperature
- the initial phases (PHASE0) for the tidal forcing
- biological state variables (except chlorophyll) and grazing pressure
- inorganic sediment
- contaminant distributions (Eulerian module)
- particle positions, labels and masses for the Lagrangian particle module

Note that the previous initial arrays, except salinity and temperature, have zero default values and must be initialised if the corresponding module is activated.

Remarks

- The user may, in principle, define a series of additional arrays (listed in Table 4.2.7) which are written by the preprocessor to the *hin*-, *bin*-, *cin*- and *pin*-files. Initialising these arrays is in most cases not recommended and even not allowed in some cases. For more details see Section IV-2.6.
- If the preprocessor is run to reinitialise the program for a restart, all fields can be initialised from the previous run by reading in the *hou*-, *bou*-, *cou*- and *pou*-files. See Section IV-5.4 for details.
- Although this may occur in some applications (e.g. file *defmod_0.f* in the **river** and **plume** test cases), DEFICS is usually non-empty.
- The array PHASE0 can alternatively be defined in DEFOB2D.

- A generic example of DEFICS is found in the **/examples** directory.

DEFOB2D

File
: *defmod.f*

Type
: subroutine

Purpose
: define open boundary conditions and (time-independent) data for the 2-D mode (user-defined)

Calling program
: PREPROC

Externals
: none

Arguments
: none

Description
: The following arrays can be defined here by the user:
  - type of condition at U- and V-nodes (ITYPOBU, ITYPOBV)
  - the frequencies for the forcing (SIGMA) and harmonic analysis (ANALSIG)
  - initial phases (PHASE0) for the tidal forcing
  - residuals, amplitudes and phases for the harmonic expansion at U-nodes (AMPOBU, PHAOBU) and V-nodes (AMPOBV, PHAOBV)

  For more details see Section IV-2.4.

Remarks
- Important to note is that the residuals, amplitudes and phases are only defined in DEFOB2D if the counter IC2BC $= 0$, in which case they are considered as uniform in time. The arrays (ITYPOBU, ITYPOBV), the frequencies, the initial phases and the (time-independent) residuals, amplitudes and phases are written to the appropriate output files in BCSOUT. If IC2BC $\neq 0$, the residuals, amplitudes and phases must be defined and written to the output data file in WROBDAT.
- Note that all arrays are zero by default.

- The routine DEFOB2D can be empty in which case there is no tidal forcing and no open boundary input. It is obvious that the routine is of no relevance to the program if all model boundaries are closed (NOBU = 0, NOBV = 0).
- The array PHASE0 can alternatively be defined in DEFICS.
- Note that the arrays (AMPOBU, PHAOBU) and (AMPOBV, PHAOBV) have a different meaning in the case of a 1-D application. See Section IV-2.9 for details.
- A generic example of DEFOB2D is found in the **/examples** directory.

DEFOB3D

File

defmod.f

Type

subroutine

Purpose

define open boundary conditions and (time-independent) data for 3-D quantities (user-defined)

Calling program

PREPROC

Externals

none

Arguments

none

Description

The following arrays can be defined here by the user:

- arrays giving the profile number to be used at U-nodes (IVPOBU) and V-nodes (IVPOBV)
- profile ("PSIVP") arrays (physics, biology, sediment, contaminants) providing either data input at the open boundaries or instructing the program to use a zero normal gradient condition
- labels of the particles generated at U-nodes (LSTOBU) and V-nodes (LSTOBV)
- open boundary input for the Lagrangian particle module at U-nodes (QSTOBU) and V-nodes (QSTOBV)

For more details see Section IV-2.5.

Remarks

- The profile arrays and the arrays QSTOBU, QSTOBV only need to be defined here if the corresponding module is activated and the corresponding counter is zero (time-independent data input). If the corresponding counter (ICHBC, ICBBC, ICPBC, ICCBC) is non-zero, the associated profile arrays must be defined in WROBDAT.
- The arrays (IVPOBU, IVPOBV), (LSTOBU, LSTOBV) and the (time-independent) profile arrays are written to the appropriate output files in BCSOUT.
- The profile arrays are set to a zero gradient condition by default. All other arrays have zero default values.
- Routine DEFOB3D can only be empty if all model boundaries are closed (NOBU = 0, NOBV = 0). If the user prefers a zero gradient condition at all open boundaries it is sufficient to set IVPOBU and IVPOBV to 1 and to set NVPROF (defined in *param.inc*) to 1 (or any positive value).
- A generic example of DEFOB3D is found in the **/examples** directory.

INCTUR

File
: *preproc.f*

Type
: subroutine

Purpose
: initial conditions for the turbulence module (default)

Calling program
: PREPROC

Externals
: BSTRES, BUOFR2, DISLEN, FNSHB, FNSMU, METIN, TLENG

Arguments
: none

Local variables
: TKEWMIN: minimum value for turbulence energy (=10^{-6} J/kg)
: GHAN: stability parameter G_h or α_N defined by either (3.1.119) or (3.1.121)

Description

- Check first whether the turbulence field arrays have already been initialised (in DEFICS). This may occur if the preprocessor is run to reinitialise the program for a restart and the turbulence arrays are defined by reading the *hou*-file in DEFICS. The criterion is that the array VEDDYV has at least one non-zero element in which case the routine is exited without further initialisation.
- Initialise surface and bottom stress by calling METIN and BSTRES.
- If IOPTK $= 0$, the eddy coefficients VEDDYV and VEDDYD are initialised to their background values VISMOL and DIFMOL and the routine is exited.
- Initialise turbulence energy using (3.1.286). A minimum value of 10^{-6} J/kg is imposed.
- Initialise the mixing length and the dissipation rate by calling TLENG and DISLEN.
- The stability coefficients S_m, S_h or S_u, S_b are determined by calling FNSMU and FNSHB. The results are stored into the arrays SMU (S_m or S_u) and SHB (S_h or S_b).
- The eddy coefficients ν_T and λ_T are evaluated either using (3.1.115) for the $k-l$ (ITCPAR $= 1$) formulation or (3.1.117) for the $k-\varepsilon$ (ITCPAR $= 2$) formulation. The results are stored in the arrays VEDDYV and VEDDYD.
- The vertical diffusion coefficient appearing in the k- and kl-equations is stored in VEDDYK, the one appearing in the ε-equation in VEDDYE.

Remarks

- It is clear that the initial profile for turbulence energy is only a crude estimate. A more realistic profile should depend at least on the shear of the horizontal current which is not available in general since the current is set to zero by default.
- The stability parameter (G_h or α_N) has a lower and upper limit. See the remarks following the description of VEDDY2 for more details.
- Although INCTUR is always called by the preprocessor, the routine has no relevance for the program if IOPTK $= 1$.
- All initial turbulence fields are written by the preprocessor to the *hin*-file.

INIBIO

File

inicon.f

Type

subroutine

Purpose

define model parameters (universal and derived) for the biological module

Calling program

INITC

Externals

none

Arguments

none

Description

A number of standard parameters are defined and a series of parameters are derived from a more fundamental set. See Table 5.1.1 for details.

Remarks

The values of these parameters can only be changed in INIBIO and cannot be reset by the user in DEFCON.

INICON

File

inicon.f

Type

subroutine

Purpose

define model parameters for the physics

Calling program

INITC, PREPROC

Externals

none

Arguments

none

Table 5.1.1: Model parameters for the biology defined in subroutine INIBIO.

name	value	unit	purpose
constants			
UCK1	0.0864	(mmol s)(nmol day)$^{-1}$	factor to convert (nmol day) into (mmol s)
EJK2	4.15	μE/J	factor to convert J into μE
BTRE	0.42	mmol C m^{-3}	grazing threshold for microplankton carbon below which no grazing occurs
OMIN	0.1	mmol O m^{-3}	minimum value for oxygen used in some of the biological transport equations
derived parameters			
ALPHAL	0.23	(mmol C) (mg Chl)$^{-1}$ day^{-1} W^{-1} m^2	photosynthetic efficiency α given by $\epsilon^X\Phi$ times the conversion factors UCK1 and EJK2 (see Section III-2.4.2a for details)
GRMAX	2.33	day^{-1}	maximum growth rate μ_{max} for nutrient controlled growth at 20^{0}C given by (3.2.40)
QMIN	0.09	mmol N (mmol C)$^{-1}$	minimum value Q_{min} for the nitrogen to carbon quota given by (3.2.38)
QMAX	0.19	mmol N (mmol C)$^{-1}$	maximum value Q_{max} for the nitrogen to carbon quota given by (3.2.48)
RB0	0.04	day^{-1}	basal respiration rate r_0 given by (3.2.30)
RMU	1.18		respiration slope b given by (3.2.31)
UMAXNH	1.05	mmol N (mmol C)$^{-1}$ day^{-1}	maximum ammonium uptake rate $^{NH}u_{max}$ at 20^{0}C given by (3.2.50)
UMAXNO	0.35	mmol N (mmol C)$^{-1}$ day^{-1}	maximum nitrate uptake rate $^{NO}u_{max}$ at 20^{0}C given (3.2.44)
QMAXMOD	0.14	mmol N (mmol C)$^{-1}$	the factor $Q_{max} - q_h\eta$ in the nutrient uptake expressions (3.2.47) and (3.2.52)
XQN	1.4	mg Chl (mmol N)$^{-1}$	ratio of chlorophyll to microplankton nitrogen given by $^Xq^N = {}^Xq^N_a(1-\eta)$

Description

A number of parameters, mainly related to the turbulence schemes described in Section III-1.2.2, are defined. A full list is given in Table 5.1.2.

Remarks

- Note that some turbulence parameters are derived from a more fundamental set.
- The values of the above parameters can only be changed in INICON and cannot be reset by the user in DEFCON.

INITC

File

initc.f

Type

subroutine

Purpose

initialise parameters and arrays, set up the initial conditions for the model run

Calling program

MAINPROG

Externals

ERRICS, INIBIO, INICON, INPA, NDIFF, RDGRD, STAUCH, TRANSH, TRANSV, ZERO

Arguments

none

Description

- Define model parameters for the physics and biology (except those already defined in DEFAULT or DEFCON) by calling INICON and INIBIO.
- Read the *grd*-file by calling RDGRD.
- Evaluate horizontal grid spacings, grid coordinates and Coriolis frequency by calling TRANSH.
- Initialise vertical grid spacings by calling TRANSV.
- Evaluate CFL limit for the 2-D mode (see equation (3.4.6))
- Calculate the total number of time steps, given by the parameter NSTEP, by calling NDIFF and initialise date and time.

Table 5.1.2: Model parameters defined in subroutine INICON.

name	value	purpose
physical parameters		
CKAR	0.4	von Kármán constant
CP	3987.5	specific heat of seawater at constant pressure J kg^{-1} $(^0C)^{-1}$)
REARTH	6371×10^3	radius of the Earth (m)
turbulence parameters		
CLEV2(4)	—	the coefficients c_{21}, c_{22}, c_{23}, c_{24} in the zeo-equation model given either by (3.1.138) if ITCPAR = 1 or (3.1.139) if ITCPAR = 2
CLEV25(6)	—	the coefficients in the expressions (3.1.118) (ITCPAR = 1) or (3.1.120) (ITCPAR = 2) for the stability functions
CMU		value of the parameter $\varepsilon_0^{4/3}$ equal to 0.096 if ITCPAR = 1 or 0.108 if ITCPAR = 2
EPS0	0.172	the parameter ε_0, in the $k-l$ theory (ITCPAR = 1), defined by the first relation (3.1.124)
	0.188	the parameter ε_0, in the $k-\varepsilon$ theory (ITCPAR = 2), defined by the second relation (3.1.124)
SIGK	1.96	the constant σ_k in the diffusion term of the turbulence energy equation (3.1.114) for the $k-l$ formulation
	1.0	as previous now for the $k-\varepsilon$ formulation
E1	1.8	the constant E_1 in the kl-equation (3.1.125)
E2	1.33	the parameter E_2 in the kl-equation (3.1.125) derived from the relation (3.1.127)
GAMAX	0.56	the maximum value $G_{h,max}$ for the stability parameter G_h, derived from (3.1.142) with $c = 0.75$, when limiting conditions are applied
	6.903	the maximum value $\alpha_{N,max}$, derived from (3.1.142) with $c = 0.495$, for the stability parameter α_N when limiting conditions are applied
GAMIN	-0.046	the minimum value $G_{h,min}$ for the stability parameter as given by (3.1.146)
	-1.725	the minimum value $\alpha_{N,min}$ for the stability parameter as given by (3.1.146)
SIGE	1.3	the constant σ_ε in the diffusion term of the ε-equation (3.1.128)

Table 5.1.2: continued.

name	value	purpose
C1E	1.55	the parameter in the shear production term of the ε-equation (3.1.128) derived from the relation (3.1.129)
C2E	1.92	the constant $c_{2\varepsilon}$ in the dissipation term of the ε-equation (3.1.128)
C3E1	0.2	the constant $c_{3\varepsilon}$ in the buoyancy term of the ε-equation (3.1.128) in the case of stable stratification
C3E2	1.0	the constant $c_{3\varepsilon}$ in the buoyancy term of the ε-equation (3.1.128) in the case of unstable stratification

- Read the file(s) containing the initial conditions by calling INPA.
- Initialise the total number and indices of the particles inside the computational domain by calling STAUCH.
- Check the initial conditions and arrays in the particle module for possible errors by calling ERRICS.
- A number of model parameters are spatially uniform but are still internally represented by a 2-D or 3-D array. In some cases the spatial dependence of these parameters depends on the value of a model switch (surface stress components if IOPTM = 1; wave parameters h_s and T_w if IOPTW = 1; horizontal diffusion coefficients if IODIF = 1; density ρ if IOPTD = 0; expansion coefficients β_S and β_T if IOPTD = 0 or 1; eddy coefficients ν_T and λ_T if IOPTK = 0). In other cases (such as some optical parameters) this is implemented in view of future extensions of the program. All these spatially uniform arrays are defined in INITC.
- To make the code compatible with the rules of standard FORTRAN, all elements of arrays, referenced in a COMMON statement, must be initialised before they are used on the right hand side of an assignment statement. The following procedure is followed:
  - All arrays are set to zero on dry cells or solid interfaces.
  - Arrays whose initial values are of no relevance, are set to zero.
- The profile arrays giving the open boundary conditions for turbulence, oxygen and zooplankton nitrogen are set to a normal gradient condition. These conditions cannot be changed by the user in DEFOB3D or WROBDAT.

Remarks

The routine INITC only initialises the arrays appearing in COMMON blocks. A similar procedure is mostly adopted for local arrays in other routines except when it is clear that no conflict can arise with the standard FORTRAN rules.

WRFCDAT

File
: *defmod.f*

Type
: subroutine

Purpose
: prepare and write meteorological/wave input for the surface forcing (user-defined)

Calling program
: PREPROC

Externals
: WRARR

Arguments
: none

Description
: Meteorological data must be defined if IOPTM > 1 and written to the *met*-file at time intervals given by the counter ICMET. Wave data are to be defined if IOPTW > 1 and written to the *wav*-file at time intervals given by the counter ICWAV. The type of the data depends on the values of the switches IOPTM and IOPTW. The data arrays must be written in COHERENS format using the WRARR routine. For more details see Section IV-2.7. A list of all possible input data is found in Table 4.2.8.

Remarks

- WRFCDAT is only called if either IOPTM > 1 or IOPTW > 1.
- The data file for meteorological input is read by the main program in routine METIN, the one for wave input in WAVIN. If the user prefers to use a different format or to read the data in the main program directly from other data file(s), the METIN or WAVIN modules need to be adapted.
- WRFCDAT can be empty if IOPTM or IOPTW are both lower than 2, or the user prefers to read in the data using a modified version of METIN or WAVIN.
- A generic example of WRFCDAT is found in the **/examples** directory.

WROBDAT

File
: *defmod.f*

Type
: subroutine

Purpose
: prepare and write (time-dependent) open boundary input (user-defined)

Calling program
: PREPROC

Externals
: WRARR

Arguments
: none

Description
: Open boundary input is supplied by the user at intervals selected by an appropriate counter. The data must be written in the appropriate format using the WRARR routine. The following input can be provided:

- 2-D physics (counter IC2BC): residuals, amplitudes and phases, given by the arrays AMPOBU, AMPOBV, PHAOBU and PHAOBV (*2bc*-file)
- 3-D physics (counter ICHBC): profile arrays for salinity, temperature and horizontal current (*hbc*-file)
- biology (counter ICBBC): profile arrays for biological state variables and inorganic sediment as given in Table 4.2.6 (*bbc*-file)
- contaminants (counter ICCBC): profile array for contaminants (*cbc*-file)
- particle module (counter ICPBC): arrays QSTOBU and QSTOBV (*pbc*-file)

For more details see Section IV-2.8.

Remarks

- If a counter is zero, the corresponding open boundary data are defined in DEFOB3D.
- WROBDAT is only called if one of the five counters IC2BC, ICHBC, ICBBC, ICCBC, ICPBC is non-zero.
- The data files are read by the main program in BC2IN (2-D physics), HBCIN (3-D physics), BBCIN (biology and sediments), CBCIN (contaminants) and PBCIN (particle module). If the user prefers to use a different format or to read the data in the main program directly from other data file(s), the corresponding input module need to be adapted.

- WROBDAT can be empty if all five counters are zero, or the user prefers to read in the data using a modified version of the input routine(s) (file *bcsin.f*) in the main program.
- A generic example of WROBDAT is found in the **/examples** directory.

1.10 Input/output routines

1.10.1 File operations

CLOSEF

File
 Inout.f

Type
 subroutine

Purpose
 close a file connection

Calling program
 various

Externals
 ERROR

Arguments
 JUNIT: unit of the connected file
 JFORM: file format ('A' or 'U')

Description
- Check first whether a file connection exists on unit JUNIT.
- Close the file on unit JUNIT.
- Decrement the number of opened files (parameter NOPENF) by 1.

Remarks
- It is recommended to use CLOSEF for closing a file opened by OPENF.
- An error message is issued and the program stops if no file is connected.

IGETFUN

File
: *Inout.f*

Type
: integer function

Purpose
: return the next available unit number for file connection

Calling program
: various

Externals
: ERROR

Arguments
: none

Description
: The function returns the lowest unit number, higher than 21, which is not yet connected to a file.

Remarks

- It is recommended to call IGETFUN before opening a file with OPENF. This allows to use all the available unit numbers in a more efficient way.
- An error message is issued and the program stops if the unit number is not accessible for a file connection.

OPENF

File
: *Inout.f*

Type
: subroutine

Purpose
: open a file connection

Calling program
: various

Externals

ERROR

Arguments

JUNIT: file unit

JNAME: file name

JTYPE: file type

- 'IN': input only
- 'OUT': output only
- 'TEMP': temporary file

JFORM: file format

- 'A': ASCII
- 'U': unformatted binary

Description

- Inquire whether the file exists if JTYPE='IN'.
- Increment the number of opened files (parameter NOPENF) by 1.
- The file is opened with the following specifiers:
  - UNIT: the value of JUNIT
  - FILE: the file name JNAME
  - FORM: 'FORMATTED' if JFORM='A', 'UNFORMATTED' if JFORM='U'
  - STATUS: 'OLD' if JTYPE='IN', UNKNOWN if JTYPE='OUT', 'SCRATCH' if JTYPE='TEMP'

Remarks

- It is recommended to open user-defined files with OPENF. The unit number JUNIT can be obtained by first calling IGETFUN.
- The program stops with an error message if an input file does not exists, the number of opened files is larger than the maximum allowed value given by the parameter MAXFILES, or any other error occurs in the OPEN statement.

RDARI

File

Inout.f

Type

subroutine

Purpose

read an integer array in COHERENS format

Calling program

various

Externals

ERROR

Arguments

IX: integer array to be read

XNAME: FORTRAN name of the array IX

KDIM: Z-dimension of the array IX

JDIM: Y-dimension of the array IX

IDIM: X-dimension of the array IX

KMAX: maximum number of values to be read in the Z-direction

JMAX: maximum number of values to be read in the Y-direction

IMAX: maximum number of values to be read in the X-direction

INFORM: file format ('A' for ASCII, 'U' for unformatted binary)

HEADS: logical flag which is .TRUE. if the file contains extra header information

IUNIT: unit of the file connection

Description

The routine reads the array IX stored in the file connected to IUNIT, by a previous call to WRARI. Note that the arguments must have the same values as the corresponding ones in the previous WRARI call. For further information see the description of WRARI below.

Remarks

- If INFORM='A', the routine first reads a header line and checks whether the array name is equal to XNAME and the dimensions are equal to KMAX, JMAX, IMAX. The program stops with an error message if this is not the case, or if an error occurs during the execution of a READ statement (READ error or end of file condition), or if the file connection does not exist.
- RDARI can be used to read 3-D, 2-D or 1-D integer arrays (e.g. KDIM = KMAX = 1 for a 2-D horizontal array).

- The argument HEADS is always set in the program to the user-defined parameter HEADERS which is .FALSE. by default. Note that HEADS is not used for binary input data.

RDARR

File
: *Inout.f*

Type
: subroutine

Purpose
: read a real array in COHERENS format

Calling program
: various

Externals
: ERROR

Arguments

X: real array to be read

XNAME: FORTRAN name of the array X

KDIM: Z-dimension of the array X

JDIM: Y-dimension of the array X

IDIM: X-dimension of the array X

KMAX: maximum number of values to be read in the Z-direction

JMAX: maximum number of values to be read in the Y-direction

IMAX: maximum number of values to be read in the X-direction

INFORM: file format ('A' for ASCII, 'U' for unformatted binary)

HEADS: logical flag which is .TRUE. if the file contains extra header information

IUNIT: unit of the file connection

Description
: The routine reads the array X stored in the file connected to IUNIT, by a previous call to WRARR. Note that the arguments must have the same values as the corresponding ones in the previous WRARR call. For further information see the description of WRARR below.

Remarks

- If INFORM='A', the routine first reads a header line and checks whether the array name is equal to XNAME and the dimensions are equal to KMAX, JMAX, IMAX. The program stops with an error message if this is not the case, or if an error occurs during the execution of a READ statement (READ error or end of file condition), or if the file connection does not exist.
- RDARR can be used to read 3-D, 2-D or 1-D real arrays (e.g. KDIM = KMAX = 1 for a 2-D horizontal array).
- The argument HEADS is always set in the program to the user-defined parameter HEADERS which is .FALSE. by default. Note that HEADS is not used for binary input data.

WRARI

File

Inout.f

Type

subroutine

Purpose

write an integer array in COHERENS format

Calling program

various

Externals

ERROR

Arguments

IX: integer array to be written

KDIM: Z-dimension of the array IX

JDIM: Y-dimension of the array IX

IDIM: X-dimension of the array IX

KMAX: maximum number of values to be written in the Z-direction

JMAX: maximum number of values to be written in the Y-direction

IMAX: maximum number of values to be written in the X-direction

XNAME: FORTRAN name of the array IX (maximum of 8 characters)

XDESC: description of the variable IX (maximum of 30 characters)

XUNIT: unit of the variable IX (maximum of 16 characters)

OUTFORM: file format ('A' for ASCII, 'U' for unformatted binary)

HEADS: logical flag which is .TRUE. if the file is to be written with extra header information

IUNIT: unit of the file connection

Description

- Checks whether the file is opened and can be used for output.
- In the case of 'A'-format the file is written as follows:
  - A first header line is written with the values of XNAME, XDESC, XUNIT, IMAX, JMAX, KMAX.
  - If HEADS is .TRUE., the elements of IX are written using the format

```
      DO 100 I=1,IMAX
         ...
         DO 110 J=1,JMAX
            ...
            WRITE (IUNIT,'(100(I11,1X))') (IX(K,J,I),K=1,KMAX)
110      CONTINUE
100   CONTINUE
```

    Extra header lines are written with the values of the array indices I and J. Although this procedure may create large output files, it may be a useful practice for debugging purposes.
  - If HEADS is .FALSE., the array is written in the format:

```
      WRITE (IUNIT,'(100(I11,1X))')
     1     (((IX(K,J,I),K=1,KMAX),J=1,JMAX),I=1,IMAX)
```

- In the case of 'U'-format the array is written by

```
      WRITE (IUNIT)
     1     (((IX(K,J,I),K=1,KMAX),J=1,JMAX),I=1,IMAX)
```

Remarks

- Arrays written using WRARI can be read with the aid of the RDARI routine. Note that the arguments must be the same as the corresponding ones used in WRARI.
- Note that only the first IMAX elements are written in the X-direction, the first JMAX elements in the Y-direction and the first KMAX elements in the Z-direction.
- All arrays in the program are written in COHERENS format, except those in the *con*- and *obc*-files.

- The program stops with an error message if the file connection does not exist, the file is write protected or an error occurs during the execution of a WRITE statement.
- WRARI can be used to write 3-D, 2-D or 1-D integer arrays (e.g. KDIM = KMAX = 1 for a 2-D horizontal array).
- The argument HEADS is always set in the program to the user-defined parameter HEADERS which is .FALSE. by default. Note that HEADS is not used for binary output data.

WRARR

File
: *Inout.f*

Type
: subroutine

Purpose
: write a real array in COHERENS format

Calling program
: various

Externals
: ERROR

Arguments

X: real array to be written

KDIM: Z-dimension of the array X

JDIM: Y-dimension of the array X

IDIM: X-dimension of the array X

KMAX: maximum number of values to be written in the Z-direction

JMAX: maximum number of values to be written in the Y-direction

IMAX: maximum number of values to be written in the X-direction

XNAME: FORTRAN name of the array X (maximum of 8 characters)

XDESC: description of the variable X (maximum of 30 characters)

XUNIT: unit of the variable X (maximum of 16 characters)

OUTFORM: file format ('A' for ASCII, 'U' for unformatted binary)

HEADS: logical flag which is .TRUE. if the file is to be written with extra header information

IUNIT: unit of the file connection

Description

- Checks whether the file is opened and can be used for output.
- In the case of 'A'-format the file is written as follows:
  - A first header line is written with the values of XNAME, XDESC, XUNIT, IMAX, JMAX, KMAX.
  - If HEADS is .TRUE., the elements of X are written using the format:

```
      DO 100 I=1,IMAX
         ...
         DO 110 J=1,JMAX
            ...
            WRITE (IUNIT,'(100(1PE14.7,1X))') (X(K,J,I),K=1,KMAX)
110      CONTINUE
100   CONTINUE
```

    Extra header lines are written with the values of the array indices I and J. Although this procedure may create large output files, it may be a useful practice for debugging purposes.
  - If HEADS is .FALSE., the array is written in the format:

```
      WRITE (IUNIT,'(100(1PE14.7,1X))')
     1     (((X(K,J,I),K=1,KMAX),J=1,JMAX),I=1,IMAX)
```

- In the case of 'U'-format the array is written by

```
      WRITE (IUNIT)
     1     (((X(K,J,I),K=1,KMAX),J=1,JMAX),I=1,IMAX)
```

Remarks

- Arrays written using WRARR can be read with the aid of the RDARR routine. Note that the arguments must be the same as the corresponding ones used in WRARR.
- Note that only the first IMAX elements are written in the X-direction, the first JMAX elements in the Y-direction and the first KMAX elements in the Z-direction.
- All arrays in the program are written in COHERENS format, except those in the *con*- and *obc*-files.
- The program stops with an error message if the file connection does not exist, the file is write protected or an error occurs during the execution of a WRITE statement.

- WRARR can be used to write 3-D, 2-D or 1-D real arrays (e.g. KDIM = KMAX = 1 for a 2-D horizontal array).
- The argument HEADS is always set in the program to the user-defined parameter HEADERS which is .FALSE. by default. Note that HEADS is not used for binary output data.

1.10.2 Input/output routines for model setup

BBCIN

File
: *bcsin.f*

Type
: subroutine

Purpose
: read open boundary input for the biological and sediment modules

Calling program
: BCSIN, MAINPROG

Externals
: RDARR

Arguments
: none

Description
: The profile arrays are read from the *bbc*-file in the same order as they are stored by the BCSOUT or WROBDAT routine.

Remarks

- BBCIN is called at the initial time from BCSIN, and, if ICBBC > 0, called by MAINPROG at time intervals selected by ICBBC.
- BBCIN is only called when IOPTB $= 1$ or IOPTS $= 1$.
- The format of the data file is determined by the value of the parameter BCSFORM.

BCSIN

File
: *bcsin.f*

Type
: subroutine

Purpose
: read the open boundary conditions and data at the initial time, determine the orientation of the open boundary cell faces

Calling program
: MAINPROG

Externals
: BBCIN, BC2IN, CBCIN, ERRBCS, ERROR, HBCIN, PBCIN

Arguments
: none

Description

- Read the open boundary conditions from the *obc*-file:
  - frequencies: arrays SIGMA, ANALSIG
  - type of open boundary conditions: arrays ITYPOBU, ITYPOBV (2-D mode) and IVPOBU, IVPOBV (3-D mode)
  - particle labels: arrays LSTOBU, LSTOBV (IOPTP > 0)
- Check the open boundary conditions for possible errors by calling ERRBCS.
- Read open boundary input for the 2-D mode by calling BC2IN.
- Read open boundary input for the 3-D physics by calling HBCIN.
- Read open boundary input for the biology and sediments by calling BBCIN (IOPTB or IOPTS equal to 1).
- Read open boundary input for the particle module by calling PBCIN (IOPTP > 0).
- Read open boundary input for the contaminant module by calling CBCIN (IOPTC > 0).
- Evaluate the arrays WESTOB and SOUTOB at respectively U- and V-open boundaries. The former has the value .TRUE. (.FALSE.) at western (eastern) open boundary faces, while the latter has the value .TRUE. (.FALSE.) at southern (northern) open boundary faces.

Remarks

The *obc*-file is not in the usual COHERENS format. If an error occurs during reading, the program stops with an error message which mentions the line where the error occurred.

BCSOUT

File

preproc.f

Type

subroutine

Purpose

write the open boundary conditions and all (time-independent) open boundary data

Calling program

PREPROC

Externals

ERRBCS, ERROR, WRARR

Arguments

none

Description

- Check the open boundary conditions for possible errors by calling ERRBCS.
- Write the open boundary conditions to the *obc*-file:
  - frequencies: arrays SIGMA, ANALSIG
  - type of open boundary conditions: arrays ITYPOBU, ITYPOBV (2-D mode) and IVPOBU, IVPOBV (3-D mode)
  - particle labels: arrays LSTOBU, LSTOBV (IOPTP > 0)
- Write the open boundary input for the 2-D mode to the *2bc*-file if IC2BC $= 0$.
- Write the open boundary input for the 3-D physics to the *hbc*-file if ICHBC $= 0$.
- Write the open boundary input for the biology and sediments to the *bbc*-file if ICBBC $= 0$ and IOPTB $= 1$ or IOPTS $= 1$.
- Write the open boundary input for the particle module to the *pbc*-file if ICPBC $=$ 0 and IOPTP > 0.
- Write the open boundary input for the contaminants to the *cbc*-file if ICCBC $= 0$ and IOPTC > 0.

Remarks

- The *obc*-file is not in the usual COHERENS format. Note that it is always written in ‘A’-format.
- The format of the input data files is determined by the value of the parameter BCSFORM.

BC2IN

File

bcsin.f

Type

subroutine

Purpose

read open boundary input for the 2-D mode

Calling program

BCSIN, MAINPROG

Externals

RDARR

Arguments

none

Description

The residuals, amplitudes and phases (arrays AMPOBU, PHAOBU, AMPOBV, PHAOBV) are read from the *2bc*-file in the same order as they are stored by the BCSOUT or WROBDAT routine.

Remarks

- BC2IN is called at the initial time from BCSIN, and, if IC2BC > 0, called by MAINPROG at time intervals selected by IC2BC.
- The format of the data file is determined by the value of the parameter BCSFORM.

CBCIN

File
: *bcsin.f*

Type
: subroutine

Purpose
: read open boundary input for the contaminant module

Calling program
: BCSIN, MAINPROG

Externals
: RDARR

Arguments
: none

Description
: The profile array CONVP is read form the *cbc*-file in the same way as it is stored by the BCSOUT or WROBDAT routine.

Remarks

- CBCIN is called at the initial time from BCSIN, and, if ICCBC > 0, called by MAINPROG at time intervals selected by ICCBC.
- CBCIN is only called when IOPTC > 0.
- The format of the data file is determined by the value of the parameter BCSFORM.

HBCIN

File
: *bcsin.f*

Type
: subroutine

Purpose
: read open boundary input for the physics

Calling program
: BCSIN, MAINPROG

Externals

RDARR

Arguments

none

Description

The profile arrays are read form the *hbc*-file in the same order as they are stored by the BCSOUT or WROBDAT routines.

Remarks

- HBCIN is called at the initial time from BCSIN, and, if ICHBC > 0, called by MAINPROG at time intervals selected by ICHBC.
- The format of the files is determined by the value of the parameter BCSFORM.

INPA

File

outpa.f

Type

subroutine

Purpose

read the initial conditions file(s)

Calling program

INITC

Externals

CLOSEF, ERROR, RDARR

Arguments

none

Description

- Read the initial conditions for the physics from the *hin*-file.
- If IOPTB $= 1$ or IOPTS $= 1$, read the initial conditions for the biology and sediments from the *bin*-file.
- If IOPTP > 0, read the initial conditions for the particle module from the *pin*-file.

- If IOPTC > 0, read the initial conditions for the contaminant module from the *cin*-file.

For an overview of all arrays see Table 4.2.7.

Remarks

- INPA can be used in DEFICS to reinitialise the program for a restart.
- INPA reads the input arrays in the same order as they are written by OUTPA.
- The format of the files is determined by the value of the parameter ICRFORM.

METIN

File

metin.f

Type

subroutine

Purpose

read meteorological data, evaluate surface fluxes and solar irradiance

Calling program

INCTUR, METIN

Externals

ERROR, RDARR, SOLRAD, SURFLX

Arguments

none

Description

- Read the *met*-file with meteorological data in the same way as they are written in WRFCDAT. The type of the data depend of the value of the switch IOPTM.
- Although this not really necessary, all data are set to zero on dry cells.
- Evaluate the surface fluxes by calling SURFLX.
- Evaluate surface solar irradiance QSOL if necessary.

Remarks

- METIN is called if IOPTM > 1 at time intervals selected by the counter ICMET.
- Solar irradiance is only determined from SOLRAD if either the temperature or biological module is switched on and if IOPTM equals 2 or 3. If IOPTM equals 4 or 5, QSOL is already obtained from the *met*-data file.

- The format of the data file is determined by the value of the parameter METFORM.

OUTPA

File
: *outpa.f*

Type
: subroutine

Purpose
: write the initial conditions file(s)

Calling program
: MAINPROG, PREPROC

Externals
: ERROR, WRARR

Arguments
: none

Description

- Write the initial conditions for the physics to the *hou*-file.
- If IOPTB = 1 or IOPTS = 1, write the initial conditions for the biology and sediments to the *bou*-file.
- If IOPTP > 0, write the initial conditions for the particle module to the *pou*-file.
- If IOPTC > 0, write the initial conditions for the contaminant module to the *cou*-file.

For an overview of all arrays see Table 4.2.7.

Remarks

- The format of the files is determined by the value of the parameter ICRFORM.
- OUTPA is called at the end of the preprocessor and main program.

PBCIN

File

bcsin.f

Type

subroutine

Purpose

read open boundary input for the Lagrangian particle module

Calling program

BCSIN, MAINPROG

Externals

RDARR

Arguments

none

Description

The open boundary data arrays QSTOBU and QSTOBV are read from the *pbc*-file in the same way as they are stored by the BCSOUT or WROBDAT routines.

Remarks

- PBCIN is called at the initial time from BCSIN, and, if ICPBC > 0, called by MAINPROG at time intervals selected by ICPBC.
- PBCIN is only called when IOPTC > 0.
- The format of the data file is determined by the value of the parameter BCSFORM.

RDCON

File

inicon.f

Type

subroutine

Purpose

read the *con*-file with model switches and parameters

Calling program

MAINPROG

Externals

ERROR

Arguments

IOFLAG:	flag which is .TRUE. if the files for model input/output are to be opened

Description

Read the values of model switches and parameters including those defined in DEFCON or DEFAULT (see Table 4.2.2 and 4.2.3) from the *con*-file in the same way as they are written by WRCON.

- switches
- date/time parameters
- time step (2-D) and counters
- units, formats and names of the model I/O files
- file parameters
- model parameters for the physics and for the biological, sediment and particle modules

Read the parameters defined in *defmod.par*:

- specifiers for time series output if IOUTS $= 1$
- specifiers for harmonic analysis if IOUTT > 0
- specifiers for time-averaged output if IOUTF $= 1$
- specifiers for particle output if IOUTP > 0

Open the files for model input/output if IOFLAG is .TRUE. (see Section IV-4.1.1 for an overview).

Remarks

- The *con*-file is not written in the usual COHERENS-format. If an error occurs during reading, the program stops with an error message which mentions the line where the error occurred.
- The aim of the argument IOFLAG is to read in the *con*-file from a postprocessor program without opening the file connections.

RDGRD

File

Inout.f

Type
: subroutine

Purpose
: read grid arrays

Calling program
: INITC

Externals
: ERRGRD, ERROR, RDARI, RDARR

Arguments
: none

Description

- Read the grid parameters and arrays, defined in DEFGRID (or DEFGRID1), from the *grd*-file in the same way as they are stored by WRGRD. For an overview of the parameters and arrays see Table 4.2.4.
- Check the grid arrays (grid coordinates, pointers, open boundary locations) for possible errors by calling ERRGRD.

Remarks

- The format of the data file is determined by the value of the parameter GRDFORM.
- The head of the file contains 3 header lines if GRDFORM = 'A' and 1 header line if GRDFORM = 'U'.

READIN

File
: *preproc.f*

Type
: subroutine

Purpose
: read the *defmod.par* file

Calling program
: PREPROC

Externals
: ERROR

Arguments

none

Description

- Read the TITLE of the run.
- Read the output specifiers for time series output (number of output files and variables, file format, grid and time locations) if IOUTS $= 1$.
- Read parameters for harmonic analysis (number of variables, file format, grid and time locations, harmonic period) if IOUTT > 0.
- Read parameters for time-averaged output (number of output files and variables, file format, grid and time locations, period for averaging) if IOUTF $= 1$.
- Read parameters for particle output (number of particles, file format, time locations) if IOUTP $= 1$.

For more details see Section IV-3.1.

Remarks

The file *defmod.par* is read from standard input. If the file is not available, PREPROC reads from a file supplied by the user via standard input.

WAVIN

File

wavin.f

Type

subroutine

Purpose

read surface wave data

Calling program

MAINPROG

Externals

ERROR, RDARR, ZERO

Arguments

none

Description

- Read the *wav*-file with the surface wave data in the same way as they are written in WRFCDAT. The data are either spatially uniform or non-uniform depending on the value of the switch IOPTW.
- Solve the wave dispersion relation (3.1.274) by calling WAVNUM.

Remarks

- WAVIN is called if IOPTW > 1 at time intervals selected by the counter ICWAV.
- The format of the data file is determined by the value of the parameter WAVFORM.

WRCON

File

preproc.f

Type

subroutine

Purpose

write the *con*-file

Calling program

PREPROC

Externals

OPENF

Arguments

none

Description

Write the values of model switches and parameters including those defined in DEFCON or DEFAULT (see Table 4.2.2 and 4.2.3) and the output specifiers defined in *defmod.par* (see Section IV-3.1) to the *con-file*:

- TITLE of the run
- switches
- date/time parameters
- time step (2-D) and counters
- units, formats and names of model I/O files
- file parameters

- model parameters for the physics and for the biological, sediment and particle modules

Write the parameters defined in *defmod.par*:

- specifiers for time series output if IOUTS = 1
- specifiers for harmonic output if IOUTT > 0
- specifiers for time-averaged output if IOUTF > 0
- specifiers for particle output if IOUTP = 1

Remarks

- The *con*-file is not written in the usual COHERENS format.
- The file is written in 'A'-format with header information allowing the user to check the settings of all parameters before running the main program.

WRGRD

File
Inout.f

Type
subroutine

Purpose
write grid arrays

Calling program
PREPROC

Externals
ERRGRD, ERROR, WRARI, WRARR

Arguments
none

Description

- Check the grid arrays (grid coordinates, pointers, open boundary locations) for possible errors by calling ERRGRD.
- Write the grid parameters and arrays, defined in DEFGRID (or DEFGRID1) to the *grd*-file. For an overview of the parameters and arrays see Table 4.2.4.

Remarks

- The format of the data file is determined by the value of the parameter GRDFORM.
- The head of the file contains 3 header lines if GRDFORM = ‘A’ and 1 header line if GRDFORM = ‘U’.

1.10.3 Routines for user-defined output

ANALFR

File
: *defanal.f*

Type
: character function

Purpose
: names of the frequencies used in the harmonic analysis (user-defined)

Calling program
: ANALYS0

Externals
: none

Arguments
: N: index number of the harmonic frequency

Description
: This is a user-defined function. For more details see Section IV-3.3.5.

Remarks
: ANALFR is only called when IOUTT > 0.

ANALPT2D

File
: *defanal.f*

Type
: subroutine

Purpose
: define the 2-D variables for harmonic analysis and output (user-defined)

Calling program
: ANALYS1

Externals
: none

Arguments
: ANAL2D: storage array for the 2-D output variables

Description
: See Section IV-3.3.2 for details.

Remarks
: ANALPT2D is only called when IOUTT > 0.

ANALPT3D

File
: *defanal.f*

Type
: subroutine

Purpose
: define the 3-D variables for harmonic analysis and output (user-defined)

Calling program
: ANALYS1

Externals
: none

Arguments
: ANAL3D: storage array for the 3-D output variables

Description
: See Section IV-3.3.1 for details.

Remarks

ANALPT3D is only called when IOUTT > 0.

ANALYS

File

analys.f

Type

subroutine

Purpose

harmonic analysis for user-defined variables

Calling program

MAINPROG

Externals

ANALFR, ANALYS0, ANALYS1, ANALYS2, ANAL2DVAR, ANAL3DVAR, CLOSEF, CZERO, LUDCMP, WROUT

Arguments

none

Local variables

ELNAME2: string array with the FORTRAN names of the 2-D elliptic parameters

ELNAME3: string array with the FORTRAN names of the 3-D elliptic parameters

ELDESC: string array with the descriptions of the 2-D and 3-D elliptic parameters

ELUNIT: string array with the units of the 2-D and 3-D elliptic parameters

Description

This is the main unit of the harmonic analysis module. The routine mainly consists of a number of subroutine calls (given in parentheses).

- A number of operations are performed at the initial time:
  - Initialise all string arrays (CZERO).
  - Obtain names, descriptions and units of all 2-D and 3-D output variables (ANAL2DVAR, ANAL3DVAR).
  - Open output data files (*res*, *amp*, *pha*, *ell*) and write the information file for each data file (WROUT).

  - Evaluate the matrices X_{mn} and Y_{mn}, given by (3.1.296) and (3.1.297) of the linear equation systems (3.1.293) and (3.1.294). The results are stored into the local arrays A and B.
  - Write A and B as the product of a lower and upper triangular matrix (*LU*-decomposition) by calling LUDCMP.
  - Initialise the sums, given by (5.1.11) below, appearing on the right hand side of (3.1.295), (3.1.298), (3.1.299).
- Update the sums, given by equation (5.1.11) below, at each (3-D) time step (ANALYS1).
- The following operations are performed at the end of each period:
  - Calculate residuals, amplitudes, phases ($\mathsf{IOUTT} = 1, 2$) and elliptic parameters ($\mathsf{IOUTT} = 2$) and store the results into the appropriate output files (ANALYS2).
  - Initialise the sums (5.1.11) for a new period of analysis (ANALYS0).
  - Update the sums (5.1.11) for the first time step of the new period (ANALYS1).
- Close all data files if ANALYS is called for the last time.

Remarks

- See Section III-1.7 for more details about harmonic analysis.
- ANALYS is only called if $\mathsf{IOUTT} > 0$.

ANALYS0

File

analys.f

Type

subroutine

Purpose

initialise the arrays for harmonic analysis at the start of a new period

Calling program

ANALYS

Externals

none

Arguments

none

Description

- The arrays ANAL2DM, ANAL2DA, ANAL2DB (2-D variables) and ANAL3DM, ANAL3DA, ANAL3DB (3-D variables) representing the sums

$$\sum_{k=-M}^{M} F_k\,,\ \sum_{k=-M}^{M} F_k\cos(\omega_m k\Delta t)\,,\ \sum_{k=-M}^{M} F_k\sin(\omega_m k\Delta t) \qquad (5.1.11)$$

  in (3.1.295), (3.1.298) and (3.1.299) are set to zero.
- The counter k (represented by the parameter NTANAL) in the sums is set to $-M$. Note that there are $2M+1$ time steps within a period.

Remarks

- See Section III-1.7 for more details about harmonic analysis.
- ANALYS0 is only called if IOUTT > 0.

ANALYS1

File

analys.f

Type

subroutine

Purpose

update the arrays for harmonic analysis at each (3-D) time step

Calling program

ANALYS

Externals

ANALPT2D, ANALPT3D

Arguments

none

Description

- Extract the current values of F_k by calling ANALPT2D (2-D variables) and ANALPT3D (3-D variables).

- The new values are added to the sums (5.1.11) at the grid locations specified by LIMANAL and are stored into the arrays ANAL2DM, ANAL2DA, ANAL2DB for 2-D variables and ANAL3DM,ANAL3DA, ANAL3DB for 3-D variables.
- The counter k (represented by the parameter NTANAL) is incremented by 1.

Remarks

- See Section IV-3.1 for a description of the array LIMANAL.
- See Section III-1.7 for more details about harmonic analysis.
- ANALYS0 is only called if IOUTT > 0.

ANALYS2

File
: *analys.f*

Type
: subroutine

Purpose
: perform harmonic analysis and write data to the output files at the end of an harmonic period

Calling program
: ANALYS

Externals
: COMPLX, LUBKSB2, LUBKSB3, WRARR

Arguments
: none

Local variables

ANAL2DAMP: amplitudes of the 2-D variables

ANAL2DPHA: phases of the 2-D variables (rad)

ANAL2DRES: residuals of the 2-D variables

ANAL3DAMP: amplitudes of the 3-D variables

ANAL3DPHA: phases of the 3-D variables (rad)

ANAL3DRES: residuals of the 3-D variables

TELELL2: ellipticity of the 2-D tidal ellipses

TELINC2: inclination of the 2-D tidal ellipses (rad)

TELMAJ2: major axis of the 2-D tidal ellipses (m/s)

TELMIN2: minor axis of the 2-D tidal ellipses (m/s)

TELPHA2: elliptic phase of the 2-D tidal ellipses (rad)

TUMIN2: magnitude of the 2-D anticyclonic current (m/s)

TUPLUS2: magnitude of the 2-D cyclonic current (m/s)

TELELL3: ellipticity of the 3-D tidal ellipses

TELINC3: inclination of the 3-D tidal ellipses (rad)

TELMAJ3: major axis of the 3-D tidal ellipses (m/s)

TELMIN3: minor axis of the 3-D tidal ellipses (m/s)

TELPHA3: elliptic phase of the 3-D tidal ellipses (rad)

TUMIN3: magnitude of the 3-D anticyclonic current (m/s)

TUPLUS3: magnitude of the 3-D cyclonic current (m/s)

ELNAME2: string array with the FORTRAN names of the 2-D elliptic parameters

ELNAME3: string array with the FORTRAN names of the 3-D elliptic parameters

ELDESC: string array with the descriptions of the 2-D and 3-D elliptic parameters

ELUNIT: string array with the units of the 2-D and 3-D elliptic parameters

Description

- Solve the linear system (3.1.293) and (3.1.294) for all 2-D variables (LUBKSB2) and 3-D variables (LUBKSB3). The routines return the arrays ANAL2DA, ANAL2DB and ANAL3DA, ANAL3DB representing the coefficients a_n, b_n ($1 \leq n \leq N$) in the harmonic expansion (3.1.287) for respectively the 2-D and 3-D case.
- Calculate the residuals, amplitudes and phases at the grid locations specified by LIMANAL for all 2-D variables and frequencies using (3.1.295) and (3.1.302). The amplitudes and phases are obtained by calling COMPLX. The phases are evaluated in radians with respect to the central time t_c. The results are written to the *res* (residuals), *amp* (amplitudes) and *pha* (phases) output files. Note that there are different output files for different frequencies. The 2-D variables are stored into the data file in the same order as they are defined in ANALPT2D.
- Calculate the residuals, amplitudes and phases at the grid locations specified by LIMANAL for all 3-D variables and frequencies using (3.1.295) and (3.1.302). The amplitudes and phases are obtained by calling COMPLX. The phases are evaluated in radians with respect to the central time t_c. The results are written to the *res* (residuals), *amp* (amplitudes) and *pha* (phases) output files. Note that there are different output files for different frequencies. The 3-D variables are stored into the data file in the same order as they are defined in ANALPT3D.

- Evaluate and write the tidal ellipse parameters for the depth-averaged current if IOUTT $= 2$. The parameters are calculated at the grid locations specified by LIMANAL and for each frequency using the theory described in Section III-1.7.2. The values are written to the output files (one for each frequency) in the following order:

TELMAJ2:	major axis (m/s)
TELMIN2:	minor axis (m/s)
TELELL2:	ellipticity
TELINC2:	inclination Θ (radians)
TELPHA2:	elliptic phase Φ (radians)
TUPLUS2:	magnitude of the cyclonic current $\|S_+\|$ (m/s)
TUMIN2:	magnitude of the anticyclonic current $\|S_-\|$ (m/s)

- Evaluate and write the tidal ellipse parameters for the 3-D current if IOUTT $= 2$. The parameters are calculated at the grid locations specified by LIMANAL and for each frequency using the theory described in Section III-1.7.2. The values are written to the output files (one for each frequency) in the following order:

TELMAJ3:	major axis (m/s)
TELMIN3:	minor axis (m/s)
TELELL3:	ellipticity
TELINC3:	inclination Θ (radians)
TELPHA3:	elliptic phase Φ (radians)
TUPLUS3:	magnitude of the cyclonic current $\|S_+\|$ (m/s)
TUMIN3:	magnitude of the anticyclonic current $\|S_-\|$ (m/s)

Remarks

- There are separate output files for 2-D and 3-D variables and a separate file for each frequency. A description of the file names is given in Section IV-4.1.2.
- See Section IV-3.1 for a description of the array LIMANAL.
- The format of the output data files is determined by the value of the parameter ANALFORM.
- If IOUTT $= 2$, the u- and v-current must be the first and second variables defined in ANALPT3D, and the depth-averaged currents $\overline{U}/H$ and $\overline{V}/H$ must be the first and second variable defined in ANALPT2D.
- ANALYS2 is only called if IOUTT > 0.

ANAL2DVAR

File
: *defanal.f*

Type
: subroutine

Purpose
: define the names, descriptions and units of the 2-D variables for harmonic output (user-defined)

Calling program
: ANALYS

Externals
: none

Arguments
: NAMES2: names

  DESCS2: descriptions

  UNITS2: units

Description
: See Section IV-3.3.4 for details.

Remarks
: ANAL2DVAR is only called when IOUTT > 0.

ANAL3DVAR

File
: *defanal.f*

Type
: subroutine

Purpose
: define the names, descriptions and units of the 3-D variables for harmonic output (user-defined)

Calling program
: ANALYS

Externals
: none

Arguments

NAMES3: names

DESCS3: descriptions

UNITS3: units

Description
: See Section IV-3.3.3 for details.

Remarks
: ANAL3DVAR is only called when IOUTT > 0.

AVRPT2D

File
: *defavr.f*

Type
: subroutine

Purpose
: define the 2-D variables for time-averaged output (user-defined)

Calling program
: INTEGR1

Externals
: none

Arguments

OUT2D: storage array for the 2-D output variables

Description
: See Section IV-3.4.2 for details.

Remarks
: AVRPT2D is only called when IOUTF $= 1$.

AVRPT3D

File

defavr.f

Type

subroutine

Purpose

define the 3-D variables for time-averaged output (user-defined)

Calling program

INTEGR1

Externals

none

Arguments

OUT3D: storage array for the 3-D output variables

Description

See Section IV-3.4.1 for details.

Remarks

AVRPT3D is only called when IOUTF = 1.

AVR2DVAR

File

defavr.f

Type

subroutine

Purpose

define the names, descriptions and units of the 2-D variables for time-averaged output (user-defined)

Calling program

INTEGR

Externals

none

Arguments

NAMES2: names

DESCS2: descriptions

UNITS2: units

Description

See Section IV-3.4.4 for details.

Remarks

AVR2DVAR is only called when IOUTF = 1.

AVR3DVAR

File

defavr.f

Type

subroutine

Purpose

define the names, descriptions and units of the 3-D variables for time-averaged output (user-defined)

Calling program

INTEGR

Externals

none

Arguments

NAMES3: names

DESCS3: descriptions

UNITS3: units

Description

See Section IV-3.4.3 for details.

Remarks

AVR3DVAR is only called when IOUTF = 1.

INTEGR

File

integr.f

Type

subroutine

Purpose

time averaging for user-defined variables

Calling program

MAINPROG

Externals

AVR2DVAR, AVR3DVAR, CLOSEF, CZERO, INTEGR0, INTEGR1, INTEGR2, WROUT

Arguments

none

Description

This is the main unit of the module for time averaging. The routine mainly consists of a number of subroutine calls (given in parentheses).

- A number of operations is performed at the initial time:
  - Initialise all string arrays (CZERO).
  - Obtain names, descriptions and units of all 2-D and 3-D output variables (AVR2DVAR, AVR3DVAR).
  - Open the *avr* output data files and write the information file for each data file (WROUT). Note that the total number of 2-D and 3-D output data files is given by the parameter NARRAVR, defined in *defmod.par*.
  - Initialise the sum arrays AVR2D and AVR3D (INTEGR0).
- The time average of a quantity Q is defined as

$$\langle Q\rangle = \sum_{k=0}^{M} Q(t_i + k\Delta t)/(M+1) \tag{5.1.12}$$

  where t_i is the initial time and $M\Delta t$ the period for time averaging. After each period t_i is increased by $M\Delta t$ and a new averaged value is calculated. The first value of t_i, the period and the number of periods, and the grid locations for time averaging are determined by the values of the array LIMAVR defined in *defmod.par* (see Section IV-3.1 for details). Averaging is performed in the following steps:
  - Initialise the sums (5.1.12) to zero (INTEGR0) on first call.

  - Update the sums (5.1.12) at each (3-D) time step and for each variable defined in AVRPT2D and AVRPT3D by calling INTEGR1.
  - Save the averaged values at the end of the period to the output files (INTEGR2).
  - Reinitialise the sums (5.1.12) to zero at the end of a period (INTEGR0).
  - Update the sums (5.1.12) for the first time step of the new period (INTEGR1).
- The previous steps are repeated for each of the NARRAVR output files.
- Close all data files if INTEGR is called for the last time.

Remarks
INTEGR is only called if IOUTF = 1.

INTEGR0

File
integr.f

Type
subroutine

Purpose
initialise the sum arrays for time averaging at the start of a new period

Calling program
INTEGR

Externals
none

Arguments

IARX: output file number

Description
The sum arrays AVR2D, AVR3D are set to zero.

Remarks
INTEGR0 is only called if IOUTF = 1.

INTEGR1

File
: *integr.f*

Type
: subroutine

Purpose
: update the sum arrays for time averaging at each (3-D) time step

Calling program
: INTEGR

Externals
: AVRPT2D, AVRPT3D

Arguments
: IARX: output file number

Description

- Extract the current values of the 2-D and 3-D quantities for time-averaged output by calling AVRPT2D and AVRPT3D.
- Add the results to the sum arrays AVR2D and AVR3D for the grid locations specified by the array LIMAVR.

Remarks

- See Section IV-3.1 for a description of the array LIMAVR.
- INTEGR1 is only called if IOUTF = 1.

INTEGR2

File
: *integr.f*

Type
: subroutine

Purpose
: write time-averaged output

Calling program
: INTEGR

Externals

WRARR

Arguments

IARX: output file number

Description

- The routine first stores the time-averaged results for all 2-D output quantities at the grid locations specified by LIMAVR into the temporary array AVR2DSUB. The data are stored in the same order as they are defined in AVRPT2D. The array is written to the *avr* output file with number IARX.
- The routine then stores the time-averaged results for all 3-D output quantities at the grid locations specified by LIMAVR into the temporary array AVR3DSUB. The data are stored in the same order as they are defined in AVRPT3D. The array is written to the *avr* output file with number IARX.

Remarks

- See Section IV-3.1 for a description of the array LIMAVR.
- The format of the output data files is determined by the values of the array AVRFORM.
- See Section IV-4.1.2 for a description of the output file names.
- INTEGR2 is only called if IOUTF = 1.

OUTPAR

File

sedlag.f

Type

subroutine

Purpose

write particle output from the Lagrangian particle module

Calling program

MAINPROG

Externals

CLOSEF, ERROR, OPENF, PARPT3D

Arguments
  none

Description
- Write the information file at the initial time.
- Check whether particle output has to be produced.
- Obtain the coordinates of the output particles by calling PARPT3D.
- Write the particle data to the *par* data file.
- Close the data file if OUTPAR is called for the last time.

Remarks
- The format of the data file is determined by the value of the parameter PARFORM defined in *defmod.par*.
- The Information file for particle output is described in Section IV-4.2.2b.
- OUTPAR is only called if IOUTP = 1.

OUTPT

File
  outpt.f

Type
  subroutine

Purpose
  write time series output

Calling program
  MAINPROG

Externals
  CLOSEF, CZERO, OUTPT2D, OUTPT3D, OUT2DVAR, OUT3DVAR, WRARR, WROUT

Arguments
  none

Description
  The following operations are performed at the initial time:
- Initialise all string arrays (CZERO).

- Obtain the names, descriptions and units of all 2- and 3-D output variables (OUT2DVAR, OUT3DVAR).
- Open the *out* data files and write the information file for each data file (WROUT). Note that the total number of 2-D and 3-D output data files is given by the parameter NARROUT, defined in *defmod.par*.

Time series output is written as follows:

- Check if time series output has to be produced for the current time in the program.
- Write 2-D output:
  - Extract the current values of the 2-D quantities for time series output by calling OUTPT2D.
  - Store the results into the local array OUT2DSUB for the grid locations specified by the array LIMOUT. The array is stored into the corresponding *out*-file. Note that the data arrays are stored in the same order as they are defined in OUTPT2D.
  - Extract the current values of the 3-D quantities for time series output by calling OUTPT3D.
  - Store the results into the local array OUT3DSUB for the grid locations specified by the array LIMOUT. The array is stored into the corresponding *out*-file. Note that the data arrays are stored in the same order as they are defined in OUTPT3D.
- The previous steps are repeated for each of the NARROUT output files.
- Close all data files if OUTPT is called for the last time.

Remarks

- See Section IV-3.1 for a description of the array LIMOUT.
- The format of the output data files is determined by the values of the array ARRFORM.
- See Section IV-4.1.2 for a description of the output file names.
- OUTPT is only called if IOUTS = 1.

OUTPT2D

File

defout.f

Type

subroutine

Purpose
: define the 2-D variables for time series output (user-defined)

Calling program
: OUTPT

Externals
: none

Arguments
: OUT2D: storage array for the 2-D output variables

Description
: See Section IV-3.2.2 for details.

Remarks
: OUTPT2D is only called when IOUTS = 1.

OUTPT3D

File
: *defout.f*

Type
: subroutine

Purpose
: define the 3-D variables for time series output (user-defined)

Calling program
: OUTPT

Externals
: OUT3D: storage array for the 3-D output variables

Arguments
: none

Description
: See Section IV-3.2.1 for details.

Remarks
: OUTPT3D is only called when IOUTS = 1.

OUT2DVAR

File
: *defout.f*

Type
: subroutine

Purpose
: define the names, descriptions and units of the 2-D variables for time series output (user-defined)

Calling program
: OUTPT

Externals
: none

Arguments
: NAMES2: names
: DESCS2: descriptions
: UNITS2: units

Description
: See Section IV-3.2.4 for details.

Remarks
: OUT2DVAR is only called when IOUTS = 1.

OUT3DVAR

File
: *defout.f*

Type
: subroutine

Purpose
: define the names, descriptions and units of the 3-D variables for time series output (user-defined)

Calling program
: OUTPT

Externals
: none

Arguments

NAMES3: names

DESCS3: descriptions

UNITS3: units

Description

See Section IV-3.2.3 for details.

Remarks

OUT3DVAR is only called when IOUTS = 1.

PARPT3D

File

defpar.f

Type

subroutine

Purpose

define the coordinates of the output particles (user-defined)

Calling program

OUTPAR

Externals

none

Arguments

PAR3D: storage array for the particle coordinates

Description

See Section IV-3.5 for details.

Remarks

PARPT3D is only called when IOUTP = 1.

PRINT

File
: *print.f*

Type
: subroutine

Purpose
: define non-standard output (user-defined)

Calling program
: MAINPROG

Externals
: none

Arguments
: none

Description
: See Section IV-3.6 for details.

Remarks
: PRINT is called at each (3-D) time step.

WROUT

File
: *Inout.f*

Type
: subroutine

Purpose
: open and write information file for time series, harmonic or time-averaged output

Calling program
: ANALYS0, INTEGR0, OUTPT

Externals
: ERROR, IGETFUN, OPENF

Arguments

XNAME: (generic) name of the the data file

JUNIT2: unit number of the 2-D output file

JUNIT3: unit number of the 3-D output file

LIMX: a 3×4 integer array representing the spatial and temporal resolution of the output data (e.g. LIMOUT for time series output)

XNAME2: string array with the FORTRAN names of the 2-D output variables (maximum 8 characters)

XDESC2: string array with the descriptions of the 2-D output variables (maximum 30 characters)

XUNIT2: string array with the units of the 2-D output variables (maximum 16 characters)

XNAME3: string array with the FORTRAN names of the 3-D output variables (maximum 8 characters)

XDESC3: string array with the descriptions of the 3-D output variables (maximum 30 characters)

XUNIT3: string array with the units of the 3-D output variables (maximum 16 characters)

J2VEC: number of 2-D output vectors

J2SCAL: number of 2-D output scalars

J3VEC: number of 3-D output vectors

J3SCAL: number of 3-D output scalars

N2MAX: dimension of the arrays XNAME2, XDESC2, XUNIT2

N3MAX: dimension of the arrays XNAME3, XDESC3, XUNIT3

Description

- The routine opens a data file for 2-D output on unit JUNIT2 and a data file for 3-D output on unit JUNIT3.
- Separate information files are written for 2-D and 3-D output. The files contain the following information:
  - format and full pathname of the output data file and the grid (*grd*) file
  - name, descriptions and units of the output variables
  - spatial and temporal resolution of the output data

Remarks

- These information files are a useful utility for postprocessor and plotting programs.
- Information files are described in Section IV-4.2.2a.

1.10.4 Graphical interface routines

CDF2D

File
: *netcdfint.f*

Type
: subroutine

Purpose
: read 2-D model output and write data into netCDF format

Calling program
: NETCDFINT

Reference
: Rew et al. (1997)

Externals
: ERRCHECK, NF_CREATE, NF_DEF_DIM, NF_DEF_VAR, NF_ENDDEF, NF_PUT_ATT_REAL, NF_PUT_ATT_TEXT,NF_PUT_VAR_DOUBLE, NF_PUT_VARA_DOUBLE, NF_PUT_VARA_REAL, NF_PUT_VAR1_REAL

Description

- Enter define mode.
- Define dimensions and dimension IDs.
- Define variables (name, type, dimensions, ID):
  - bathymetric array (mean water depths)
  - 2-D data field variables
  - coordinate arrays
- Define attributes (long_name, units, _FillValue, ...) for the same arrays.
- Leave define mode.
- Write (spatial) coordinates.
- Write array of mean water depths.
- Read/write 2-D data:
  - read model data for each output time
  - write results to the netCDF file for each output time
  - write time coordinate to the netCDF file for each output time

- Close netCDF file.

Remarks

- The data have an "unlimited" time dimension (see netCDF manual for details).
- The vertical coordinate array only has one (zero) value, is only supplied for completeness and has no further relevance.
- The routine uses a number of routines supplied from the netCDF library.
- An error handler (ERRCHECK routine) is provided checking each netCDF routine call for (eventual) errors.

CDF3D

File

netcdfint.f

Type

subroutine

Purpose

read 3-D model output and write data into netCDF format

Calling program

NETCDFINT

Reference

Rew et al. (1997)

Externals

ERRCHECK, NF_CREATE, NF_DEF_DIM, NF_DEF_VAR, NF_ENDDEF,
NF_PUT_ATT_REAL, NF_PUT_ATT_TEXT,NF_PUT_VAR_DOUBLE,
NF_PUT_VARA_DOUBLE, NF_PUT_VARA_REAL, NF_PUT_VAR1_REAL

Description

- Enter define mode.
- Define dimensions and dimension IDs.
- Define variables (name, type, dimensions, ID):
  - bathymetric array (mean water depths)
  - 3-D data field variables
  - coordinate arrays

- Define attributes (long_name, units, _FillValue, ...) for the same arrays.
- Leave define mode.
- Write (spatial) coordinates.
- Write array of mean water depths.
- Read/write 3-D data:
  - read model data for each output time
  - write results to the netCDF file for each output time
  - write time coordinate to the netCDF file for each output time
- Close netCDF file.

Remarks

- The data have an "unlimited" time dimension (see netCDF manual for details).
- The vertical grid is expressed in σ-coordinates. The physical vertical grid coordinates (and grid spacings) have to be determined by the plotting program with the aid of the bathymetric array using (e.g.) $x_3 = h(\sigma - 1)$, $\Delta x_3 = h\Delta\sigma$.
- The routine uses a number of routines supplied from the netCDF library.
- An error handler (ERRCHECK routine) is provided checking each netCDF routine call for (eventual) errors.

ERRCHECK

File
: *netcdfint.f*

Type
: subroutine

Purpose
: error handler for netCDF routines

Calling program
: CDF2D, CDF3D

Reference
: Rew et al. (1997)

Externals
: NF_STRERROR

Arguments

IRET: error number returned by a netCDF function call

Description

If IRET is non-zero, an error message is issued (by calling NF_STRERROR) and the program stops execution.

Remarks

The function NF_STRERROR is supplied from the netCDF library.

1.10.5 Reading user-defined output

RDANAL

File

readdat.f

Type

subroutine

Purpose

read harmonic model output (residuals, amplitudes, phases)

Externals

none

Arguments

RES2D: storage array for 2-D output data (residuals)

RES3D: storage array for 3-D output data (residuals)

AMP2D: storage array for 2-D output data (amplitudes)

AMP3D: storage array for 3-D output data (amplitudes)

PHA2D: storage array for 2-D output data (phases)

PHA3D: storage array for 3-D output data (phases)

NZDAT: Z-dimension of the 3-D storage arrays

NRDAT: Y-dimension of the 2-D and 3-D storage arrays

NCDAT: X-dimension of the 2-D and 3-D storage arrays

NTDAT: time dimension of the 2-D and 3-D storage arrays

N2RESA: number of 2-D output fields

N3RESA: number of 3-D output fields

NCONTO: number of harmonic frequencies

ANALFORM: format of the data files (‘A’ or ‘U’)

LIMANAL: integer array specifying the spatial and temporal resolution of the output data

TITLE: the first 6 characters in the output file names

HEADERS: logical flag which must be .TRUE. if ANALFORM equals ‘A’ and the data files contain extra header information

Description

The example routine reads the *res*, *amp* and *pha* output files and stores the data into a series of arrays. Assume, for convenience, that TITLE equals *mytest* (first 6 characters in the output file name) and ANALFORM equals ‘A’ (last character in the output file name).

- Residual data:
  - Open the file *mytest_1.res2A* (2-D data).
  - Read the data and store the results into the 4-dimensional array RES2D(NRDAT,NCDAT,NTDAT,N2RESA).
  - Open the file *mytest_1.res3A* (3-D data).
  - Read the data and store the results into the 5-dimensional array RES3D(NZDAT,NRDAT,NCDAT,NTDAT,N3RESA).
- Amplitudes:
  - Open the first 2-D output file for the first frequency *mytest_1.amp2A*.
  - Read the data and store the results into the 5-dimensional array AMP2D(NRDAT,NCDAT,NTDAT,N2RESA,NCONTO).
  - Open the first 3-D output file for the first frequency *mytest_1.amp3A*.
  - Read the data and store the results into the 6-dimensional array AMP3D(NZDAT,NRDAT,NCDAT,NTDAT,N3RESA,NCONTO).
  - The previous steps are repeated with the file number (character 8 in the data file name) replaced by respectively 2, 3, ..., NCONTO. The data are stored in the same arrays.
- Phases:
  - Open the first 2-D output file for the first frequency *mytest_1.pha2A*.
  - Read the data and store the results into the 5-dimensional array PHA2D(NRDAT,NCDAT,NTDAT,N2RESA,NCONTO).
  - Open the first 3-D output file for the first frequency *mytest_1.pha3A*.

- Read the data and store the results into the 6-dimensional array PHA3D(NZDAT,NRDAT,NCDAT,NTDAT,N3RESA,NCONTO).
- The previous steps are repeated with the file number (character 8 in the data file name) replaced by respectively 2, 3, . . . , NCONTO. The data are stored in the same arrays.

Remarks

- The routine is of less practical interest and is only given to show how model data can be read from the output data files, and to offer a guideline for constructing postprocessor and plotting programs.
- The X-, Y-, Z- and time dimensions of the output data are determined as follows:

```
NCMAX = (LIMANAL(2,1)-LIMANAL(1,1))/LIMANAL(3,1) + 1
NRMAX = (LIMANAL(2,2)-LIMANAL(1,2))/LIMANAL(3,2) + 1
NZMAX = (LIMANAL(2,3)-LIMANAL(1,3))/LIMANAL(3,3) + 1
NTMAX = (LIMANAL(2,4)-LIMANAL(1,4))/LIMANAL(3,4)
```

  which must obviously be lower than or equal to the corresponding dimensions NCDAT, NRDAT, NZDAT, NTDAT of the storage arrays. The array LIMANAL is further discussed in Section IV-3.1.
- An alternative way for reading the data is to make use of the program's WRARR routine.

RDAVR

File

readdat.f

Type

subroutine

Purpose

read time-averaged model output

Externals

none

Arguments

AVR2D:	storage array for 2-D output data
AVR3D:	storage array for 3-D output data

NZDAT: Z-dimension of the 3-D storage arrays

NRDAT: Y-dimension of the 2-D and 3-D storage arrays

NCDAT: X-dimension of the 2-D and 3-D storage arrays

NTDAT: time dimension of the 2-D and 3-D storage arrays

N2RESM: number of 2-D output fields

N3RESM: number of 3-D output fields

NARRAVR: total number of 2-D and 3-D output files

AVRFORM: format of the data files ('A' or 'U')

LIMAVR: integer array specifying the spatial and temporal resolution of the output data

TITLE: the first 6 characters in the output file names

HEADERS: logical flag which must be .TRUE. if the data are written in ASCII format and the data files contain extra header information

Description

The example routine reads the *avr* output file(s) and stores the data into a series of arrays. Assume, for convenience, that TITLE equals *mytest* (first 6 characters in the output file name) and the array AVRFORM has the value 'A' (last character in the output file names).

- Open the first 2-D data file *mytest_1.avr2A*.
- Read the data and store the results into the 5-dimensional array AVR2D(NRDAT,NCDAT,NTDAT,N2RESM,NARRAVR).
- Open the first 3-D data file *mytest_1.avr3A*.
- Read the data and store the results into the 6-dimensional array AVR3D(NZDAT,NRDAT,NCDAT,NTDAT,N3RESM,NARRAVR).
- The previous steps are repeated with the file number (character 8 in the data file name) replaced by respectively 2, 3, . . . , NARRAVR. The data are stored in the same arrays.

Remarks

- The routine is of less practical interest and is only given to show how model data can be read from the output data files, and to offer a guideline for constructing postprocessor and plotting programs.
- The X-, Y-, Z- and time dimensions of the output data in the file with number IARR are determined as follows:

```
NCMAX = (LIMAVR(2,1,IARR)-LIMAVR(1,1,IARR))/LIMAVR(3,1,IARR) + 1
NRMAX = (LIMAVR(2,2,IARR)-LIMAVR(1,2,IARR))/LIMAVR(3,2,IARR) + 1
NZMAX = (LIMAVR(2,3,IARR)-LIMAVR(1,3,IARR))/LIMAVR(3,3,IARR) + 1
NTMAX = (LIMAVR(2,4,IARR)-LIMAVR(1,4,IARR))/LIMAVR(3,4,IARR)
```

which must obviously be lower than or equal to the corresponding dimensions NCDAT, NRDAT, NZDAT, NTDAT of the storage arrays. The array LIMAVR is further discussed in Section IV-3.1.

- An alternative way for reading the data is to make use of the program's WRARR routine.

RDELL

File
: *readdat.f*

Type
: subroutine

Purpose
: read harmonic model output (tidal ellipse parameters)

Externals
: none

Arguments

- ELL2D: storage array for 2-D output data
- ELL3D: storage array for 3-D output data
- NZDAT: Z-dimension of the 3-D storage arrays
- NRDAT: Y-dimension of the 2-D and 3-D storage arrays
- NCDAT: X-dimension of the 2-D and 3-D storage arrays
- NTDAT: time dimension of the 2-D and 3-D storage arrays
- NCONTO: number of harmonic frequencies
- ANALFORM: format of the data files ('A' or 'U')
- LIMANAL: integer array specifying the spatial and temporal resolution of the output data
- TITLE: the first 6 characters in the output file names
- HEADERS: logical flag which must be .TRUE. if ANALFORM equals 'A' and the data files contain extra header information

Description
: The example routine reads the *ell* output files and stores the data into a series of arrays. Assume, for convenience, that TITLE equals *mytest* (first 6 characters in the output file name) and ANALFORM equals 'A' (last character in the output file name).

- Open the first 2-D output file for the first frequency *mytest_1.ell2A*.
- Read the data and store the results into the 5-dimensional array ELL2D(NRDAT,NCDAT,NTDAT,7,NCONTO).
- Open the first 3-D output file for the first frequency *mytest_1.ell3A*.
- Read the data and store the results into the 6-dimensional array ELL3D(NZDAT,NRDAT,NCDAT,NTDAT,7,NCONTO).
- The previous steps are repeated with the file number (character 8 in the data file name) replaced by respectively 2, 3, ..., NCONTO. The data are stored in the same arrays.

Remarks

- The elliptic parameters are stored in the following order:

  1: major axis A (m/s)
  2: minor axis B (m/s)
  3: ellipticity ε
  4: inclination Θ (radians)
  5: elliptic phase Φ (radians)
  6: magnitude of the cyclonic current $|S_+|$ (m/s)
  7: magnitude of the anticyclonic current $|S_-|$ (m/s)

- The routine is of less practical interest and is only given to show how model data can be read from the output data files, and to offer a guideline for constructing postprocessor and plotting programs.
- The X-, Y-, Z- and time dimensions of the output data are determined as follows:

```
NCMAX = (LIMANAL(2,1)-LIMANAL(1,1))/LIMANAL(3,1) + 1
NRMAX = (LIMANAL(2,2)-LIMANAL(1,2))/LIMANAL(3,2) + 1
NZMAX = (LIMANAL(2,3)-LIMANAL(1,3))/LIMANAL(3,3) + 1
NTMAX = (LIMANAL(2,4)-LIMANAL(1,4))/LIMANAL(3,4)
```

  which must obviously be lower than or equal to the corresponding dimensions NCDAT, NRDAT, NZDAT, NTDAT of the storage arrays. The array LIMANAL is further discussed in Section IV-3.1.

- An alternative way for reading the data is to make use of the program's WRARR routine.

RDOUT

File

readdat.f

Type

subroutine

Purpose

read time series model output

Externals

none

Arguments

OUT2D: storage array for 2-D output data

OUT3D: storage array for 3-D output data

NZDAT: Z-dimension of the 3-D storage arrays

NRDAT: Y-dimension of the 2-D and 3-D storage arrays

NCDAT: X-dimension of the 2-D and 3-D storage arrays

NTDAT: time dimension of the 2-D and 3-D storage arrays

N2RESO: number of 2-D output fields

N3RESO: number of 3-D output fields

NARROUT: total number of 2-D and 3-D output files

ARRFORM: format of the data files ('A' or 'U')

LIMOUT: integer array specifying the spatial and temporal resolution of the output data

TITLE: the first 6 characters in the output file names

HEADERS: logical flag which must be .TRUE. if the data are written in ASCII format and the data files contain extra header information

Description

The example routine reads the *out* output file(s) and stores the data into a series of arrays. Assume, for convenience, that TITLE equals *mytest* (first 6 characters in the output file name) and the array ARRFORM has the value 'A' (last character in the output file names).

- Open the first 2-D data file *mytest_1.out2A*.
- Read the data and store the results into the 5-dimensional array OUT2D(NRDAT,NCDAT,NTDAT,N2RESO,NARROUT).
- Open the first 3-D data file *mytest_1.out3A*.

- Read the data and store the results into the 6-dimensional array OUT3D(NZDAT,NRDAT,NCDAT,NTDAT,N3RESO,NARROUT).
- The previous steps are repeated with the file number (character 8 in the data file name) replaced by respectively 2, 3, . . . , NARROUT. The data are stored in the same arrays.

Remarks

- The routine is of less practical interest and is only given to show how model data can be read from the output data files, and to offer a guideline for constructing postprocessor and plotting programs.
- The X-, Y-, Z- and time dimensions of the output data in the file with number IARR are determined as follows:

  ```
  NCMAX = (LIMOUT(2,1,IARR)-LIMOUT(1,1,IARR))/LIMOUT(3,1,IARR) + 1
  NRMAX = (LIMOUT(2,2,IARR)-LIMOUT(1,2,IARR))/LIMOUT(3,2,IARR) + 1
  NZMAX = (LIMOUT(2,3,IARR)-LIMOUT(1,3,IARR))/LIMOUT(3,3,IARR) + 1
  NTMAX = (LIMOUT(2,4,IARR)-LIMOUT(1,4,IARR))/LIMOUT(3,4,IARR) + 1
  ```

  which must obviously be lower than or equal to the corresponding dimensions NCDAT, NRDAT, NZDAT, NTDAT of the storage arrays. The array LIMOUT is further discussed in Section IV-3.1.
- An alternative way for reading the data is to make use of the program's WRARR routine.

RDPAR

File

readdat.f

Type

subroutine

Purpose

read model output from the particle module

Externals

none

Arguments

PAR3D: storage array

NOPOUT: total number of particles in the data file

NTMAX: time dimension of the storage array

PARFORM: format of the data file ('A' or 'U')

LIMPOUT: integer array specifying the temporal resolution of the output data

TITLE: the first 6 characters in the output file name

HEADERS: logical flag which must be .TRUE. if PARFORM equals .TRUE. in which case an extra header line is written at each time step

Description

The example routine reads the *par* output file and stores the data into the array PAR3D. Assume, for convenience, that TITLE equals *mytest* (first 6 characters in the output file name) and PARFORM equals 'A' (last character in the output file name).

- Open the data file *mytest_1.par3A*.
- Read the data and store the results into the 3-dimensional array PAR3D(3,NOPOUT,NTMAX).

Remarks

- The routine is only given to show how model data can be read from the output data file, and to offer a guideline for constructing postprocessor and plotting programs.
- The time dimension of the output data is determined as follows:

  ```
  NTMAX = (LIMPOUT(2)-LIMPOUT(1))/LIMPOUT(3) + 1
  ```

  The array LIMPOUT and the parameters NOPOUT and PARFORM are further described in Section IV-3.1.

1.11 Utility routines

COMPLX

File

Util.f

Type

subroutine

Purpose

amplitude and phase of the complex number $c_r + ic_i$

Calling program

ANALYS2

Externals
    none

Arguments
    CR: real part of the complex number
    CI: imaginary part of the complex number
    CAMP: magnitude of the complex number
    CPHAS: phase of the complex number

Description

- The routine writes the complex number in the form
$$c_r + c_i = Ae^{i\phi} \qquad (5.1.13)$$
where
$$A = \sqrt{c_r^2 + c_i^2}\,,\ \phi = \tan^{-1}(c_i/c_r) \qquad (5.1.14)$$
are the amplitude and the phase returned by the routine.
- If $c_r = a\cos\omega t$ and $c_i = b\sin\omega t$, the routine determines the amplitude A and phase ϕ in the harmonic expression:
$$a\cos\omega t + b\sin\omega t = A\cos(\omega t - \phi) \qquad (5.1.15)$$

Remarks
    The phase is returned in radians, given between 0 and 2π.

CZERO

File
    Util.f

Type
    subroutine

Purpose
    initialise a character array

Calling program
    various

Externals
    none

Arguments

CVAR: character array to be initialised

NX: dimension of the array CVAR

LENX: length of the array CVAR

Description

The elements of the array CVAR are filled with blanks.

DAYNUM

File

Dates.f

Type

real function

Purpose

return the day number of the year

Calling program

SOLRAD

Externals

IDATES, LEAP

Arguments

IYEARX: current year

IDATEX: current date in the format MMDDHHMM (month-day-hour-minute)

Description

The function returns the fractional day number. Lower and upper limit are 1 and either 366 or 367, depending on whether IYEAR is a leap year.

ERRBCS

File
: *Errors.f*

Type
: subroutine

Purpose
: check the boundary conditions for possible errors

Calling program
: BCSIN, BCSOUT

Externals
: ERROR

Arguments
: none

Description
: The following arrays are checked:
  - open boundary locations: IOBU, JOBU, IOBV, JOBV
  - the arrays ITYPOBU, ITYPOBV
  - the profile number arrays IVPOBU, IVPOBV

Remarks
: The program stops with one or more error messages if an error is detected.

ERRGRD

File
: *Errors.f*

Type
: subroutine

Purpose
: check the grid arrays for possible errors

Calling program
: RDGRD, WRGRD

Externals
    ERROR

Arguments
    none

Description
    The routine checks the grid and pointer arrays for non-allowed values.

Remarks
    The program stops with one or more error messages if an error is detected.

ERRICS

File
    Errors.f

Type
    subroutine

Purpose
    check the initial conditions for the Lagrangian particle module

Calling program
    INITC, PREPROC

Externals
    ERROR

Arguments
    none

Description

- Check whether the coordinate arrays are within the allowed range.
- A warning message is issued if the parameter HEDPUN does not satisfy the criterium (3.4.390), in which case HEDPUN is set to the maximum allowed value.

Remarks
    The program stops with one or more error messages if an error is detected.

ERRMOD

File
: *Errors.f*

Type
: subroutine

Purpose
: check model parameters

Calling program
: MAINPROG, PREPROC

Externals
: ERROR, IDATES, LEAP, NDIFF

Arguments
: none

Description
: A series of parameters defined in DEFCON, *param.inc* or *defmod.par* is checked for non-allowed values.

Remarks
: The program stops with one or more messages if an error is found, but continues when only a warning message is issued.

ERROR

File
: *Errors.f*

Type
: subroutine

Purpose
: stop the program with an error message if necessary

Calling program
: various

Externals
: none

Arguments

SUBNAM: name of the program unit where the error (eventually) occurred

ICODE: code identifying the type of the error

Description

- Check whether an error occurred (NERRS > 0).
- If an error occurred then:
  - Write error messages including the error type given by ICODE (see Section IV-5.3).
  - Stop the program.

IDAADD

File

Dates.f

Type

subroutine

Purpose

calculate a new Julian date after adding a number of days to an initial Julian date

Calling program

NEWTIM

Externals

LEAP

Arguments

JULDB: initial Julian date

INCR: number of days to be added

JULDE: new Julian date

Remarks

The Julian date is defined in the format year*1000+day where year is the current year minus 1900 and day the current day number (3 digits).

IDATES

File

Dates.f

Type

subroutine

Purpose

convert the date variable IDATEX into month, day, hour, minute

Calling program

various

Externals

none

Arguments

IDATEX: current date given in the format MMDDHHMM (month-day-hour-minute)

IMONTX: current month

IDAYX: current day

IHOURX: current hour

IMINX: current minute

IGREG

File

Dates.f

Type

subroutine

Purpose

convert a Julian date into the Gregorian calendar date

Calling program

NEWTIM

Externals

LEAP

Arguments

JULD: Julian date

IYEARX: Gregorian calendar year

IMONT: Gregorian calendar month

IDAYX: Gregorian calendar day

Remarks

The Julian date is defined in the format year*1000+day where year is the current year minus 1900 and day the current day number (3 digits).

IJUL

File

Dates.f

Type

subroutine

Purpose

convert a date on the Gregorian calendar into a Julian date

Calling program

NDIFF, NEWTIM

Externals

LEAP

Arguments

IYEARX: Gregorian calendar year

IMONTX: Gregorian calendar month

IDAYX: Gregorian calendar day

JULD: Julian date

Remarks

The Julian date is defined in the format year*1000+day where year is the current year minus 1900 and day the current day number (3 digits).

IZERO

File
: *Util.f*

Type
: subroutine

Purpose
: initialise an integer array

Calling program
: various

Externals
: none

Arguments
: IVAR: integer array to be initialised
: NK: Z-dimension of the array IX
: NJ: Y-dimension of the array IX
: NI: X-dimension of the array IX

Description
: All elements of IVAR are set to zero.

Remarks
: The routine can be used to initialise 3-D, 2-D or 1-D arrays (e.g. NK = 1 for a 2-D horizontal array).

LEAP

File
: *Dates.f*

Type
: integer function

Purpose
: check if a year is a leap year

Calling program
: DAYNUM, ERRMOD, IDAADD, IGREG, IJUL, SOLRAD

Externals
: none

Arguments
: IYEARX: current year

Description
: Returns 1 if IYEARX is a leap year on the Gregorian calendar, 0 otherwise.

LSQFIT

File
: *Util.f*

Type
: subroutine

Purpose
: fit a series of data points (X,Y) to the straight line Y = A+BX and calculate the linear regression coefficient

Reference
: Press et al. (1989)

Externals
: none

Arguments
: X: X-coordinates of the data points
: Y: Y-coordinates of the data points
: NDATMAX: dimension of the vectors X and Y
: NDAT: number of data points
: A: parameter A of the straight line (abscis of the point where the line cuts the X-axis)
: B: slope of the straight line
: CORR: linear correlation coefficient

Description
: See Press et al. (1989) for details.

LUBKSB2

File
: *analys.f*

Type
: subroutine

Purpose
: used in ANALYS2 to solve a linear equation system of the form (3.1.293) or (3.1.294) for all 2-D variables and at all grid locations specified by LIMANAL

Reference
: Press et al. (1989)

Calling program
: ANALYS2

Externals
: none

Arguments
- F: matrix of the linear system
- INDX: vector which stores the row permutation effected by partial pivoting (obtained by a previous call to LUDCMP)
- X: right hand side of the linear system on input, solution vector on output
- NJ: number of points in the Y-direction for which the linear system has to be solved
- NI: number of points in the X-direction for which the linear system has to be solved
- N: number of linear equations

Description
: See Press et al. (1989) for details about the algorithm. Note that the matrix F is decomposed into the sum of a lower and upper triangular matrix by a previous call to LUDCMP.

LUBKSB3

File
: *analys.f*

Type
: subroutine

Purpose

used in ANALYS2 to solve a linear equation system of the form (3.1.293) or (3.1.294) for all 3-D variables and at all grid locations specified by LIMANAL

Reference

Press et al. (1989)

Calling program

ANALYS2

Externals

none

Arguments

F: matrix of the linear system

INDX: vector which stores the row permutation effected by partial pivoting (obtained by a previous call to LUDCMP)

X: right hand side of the linear system on input, solution vector on output

NK: number of points in the Z-direction for which the linear system has to be solved

NJ: number of points in the Y-direction for which the linear system has to be solved

NI: number of points in the X-direction for which the linear system has to be solved

N: number of linear equations

Description

See Press et al. (1989) for details about the algorithm. Note that the matrix F is decomposed into the sum of a lower and upper triangular matrix by a previous call to LUDCMP.

LUDCMP

File

analys.f

Type

subroutine

Purpose

decompose the matrix F into an upper (U) and lower (L) triangular matrix

Reference

Press et al. (1989)

Calling program
ANALYS

Externals
none

Arguments
F: matrix to be decomposed on input, the sum of L and U on output

INDX: vector which stores the row permutation effected by partial pivoting (needed for a subsequent call to LUBKSB2 and LUBKSB3)

N: dimensions of the square matrix F

Description
See Press et al. (1989) for details about the algorithm.

NDIFF

File
Dates.f

Type
subroutine

Purpose
number of seconds and number of time steps between two dates

Calling program
ERRMOD, INITC, PREPROC

Externals
IDATES, IJUL

Arguments
NSECS: number of seconds between the two dates

NXSTEP: number of time steps between the two dates

DELTX: time step (s)

IYEAR1: start year

IDATE1: start date given in the format MMDDHHMM (month-day-hour-min)

IYEAR2: end year

IDATE2: end date given in the format MMDDHHMM (month-day-hour-min)

INFO: logical flag which is .TRUE. if an information message has to be produced on standard output

NEWTIM

File

Dates.f

Type

subroutine

Purpose

current year and date after adding a number of time steps to an initial date and year

Calling program

MAINPROG

Externals

IDAADD, IDATES, IGREG, IJUL

Arguments

IYEAR1: initial year

IDATE1: initial date given in the format MMDDHHMM (month-day-hour-minute)

NTX: number of time steps since the initial date/year

DELTX: time step (s)

IYEAR2: current year

IDATE2: current date given in the format MMDDHHMM (month-day-hour-minute)

HOUR: number of hours elapsed since the initial date/year

INFO: logical flag which is .TRUE. if an information message has to be produced on standard output

RANDOM

File

Util.f

Type

real function

Purpose

random generator returning a random number between 0 and 1 (for the particle module)

Calling program

MCOSB, MCROB, MC3D, SEDLAG

Externals

none

Arguments

IFLAG: the random generation is continued if IFLAG is zero, and is restarted if IFLAG is non-zero.

Local variables

IEND: number of iterations

Description

The routine uses the chaos equation

$$X = R(X - X^2) \tag{5.1.16}$$

and returns the random number $\cos^{-1}(2(x - 1/2))/\pi$ after IEND iterations, with $R = 4$ and iterations of X between 0 and 1.

Remarks

This is the only routine in the program using double precision variables.

RAN3

File

Util.f

Type

real function

Purpose

random generator returning a random number between 0 and 1

Reference

Press et al. (1989)

Externals
: none

Arguments
: ISEED: the random generation is continued if ISEED is zero or positive and is restarted if ISEED is negative

Description
: See Press et al. (1989) for details.

THOMV

File
: *thomv.f*

Type
: subroutine

Purpose
: solve a series of tridiagonal linear equation systems in the vertical at a series of horizontal grid points

Reference
: Press et al. (1989)

Calling program
: TRANSPC, TRANSPW, UCALC, VCALC

Externals
: none

Arguments
: A: A-vector of the tridiagonal matrix
: B: B-vector of the tridiagonal matrix
: C: C-vector of the tridiagonal matrix
: D: right hand side of the tridiagonal equation system
: F: solution matrix
: NK: number of equations in each linear system (equal to the vertical dimension of A, B, C, D, F)
: NJ: number of grid points in the Y-direction for which the linear system has to be solved (equal to the Y-dimension of A, B, C, D, F)

NI: number of grid points in the X-direction for which the linear system has to be solved (equal to the X-dimension of A, B, C, D, F)

Description

The routine solves a series of linear systems of the form:

$$\begin{aligned} B_{1ji}F_{1ji} + C_{1ji}F_{2ji} &= D_{1ji} \\ A_{kji}F_{k-1,ji} + B_{kji}F_{kji} + C_{kji}F_{k+1,ji} &= D_{kji} \\ A_{Nji}F_{N-1,ji} + B_{Nji}F_{Nji} &= D_{Nji} \end{aligned} \tag{5.1.17}$$

where $2 \leq k \leq N-1$ and $N =$ NK. The algorithm is given by equations (3.4.188)–(3.4.190) with N_z replaced by NK.

ZERO

File

Util.f

Type

subroutine

Purpose

initialise a real array

Calling program

various

Externals

none

Arguments

VAR: real array to be initialised

NK: Z-dimension of the array X

NJ: Y-dimension of the array X

NI: X-dimension of the array X

Description

All elements of VAR are set to zero.

Remarks

The routine can be used to initialise 3-D, 2-D or 1-D arrays (e.g. NK $= 1$ for a 2-D horizontal array).

1.12 Interpolation functions

A series of interpolation functions are implemented in the program. The general features are given below followed by a short description of each function.

File
: *Intp_in.f*

Type
: real function

Purpose
: interpolate a quantity "Q" at the cell centre, U-, V-, W-, X- or Y-node

Calling program
: various

Externals
: none

Arguments

K:	Z-index of the cell centre or node where Q is interpolated
J:	Y-index of the cell centre or node where Q is interpolated
I:	X-index of the cell centre or node where Q is interpolated

Description
: The type of grid location where Q is interpolated, is indicated by the last letter(s) of the function name:

C:	interpolation at the cell centre
U:	interpolation at the U-node
V:	interpolation at the V-node
W:	interpolation at the W-node
UX:	interpolation at the X-node
VX:	interpolation at the Y-node

Remarks

- The arguments I,J,K are subject to the following constraints:
  - interpolation at U-nodes: $2 \leq$ I $\leq$ NC
  - interpolation at V-nodes: $2 \leq$ J $\leq$ NR
  - interpolation at W-nodes: $2 \leq$ K $\leq$ NZ
  - interpolation at X-nodes: $2 \leq$ I $\leq$ NC, $2 \leq$ K $\leq$ NZ

  - interpolation at Y-nodes: $2 \leq$ J $\leq$ NR, $2 \leq$ K $\leq$ NZ

- The interpolation procedures are mostly straightforward. Only exceptions are the functions CUD2ATV, CU2ATV, CVD2ATU and CV2ATU, used for the calculation of the Coriolis force in the momentum equations, where cell faces at open or solid boundaries are excluded from the interpolation.

CUD2ATV(J,I)
: depth-integrated $\overline{U}$-current (m^2/s) at V-nodes (excluding solid/open boundaries)

CU2ATV(K,J,I)
: u-current (m/s) at V-nodes (excluding solid/open boundaries)

CVD2ATU(J,I)
: depth-integrated $\overline{V}$-current (m^2/s) at U-nodes (excluding solid/open boundaries)

CV2ATU(K,J,I)
: v-current (m/s) at U-nodes (excluding solid/open boundaries)

DEPATU(J,I)
: mean water depth at U-nodes (m)

DEPATV(J,I)
: mean water depth at V-nodes (m)

GX0C(I)
: X-coordinate of cell centre (m or longitude)

GX2U(J,I)
: grid spacing Δx_1 at U-node (m)

GX2V(J,I)
: grid spacing Δx_1 at V-node (m)

GY0C(J)
: Y-coordinate of cell centre (m or latitude)

GY2U(J)
: grid spacing Δx_2 at U-node (m)

GY2V(J)
: grid spacing Δx_2 at V-node (m)

GZSC(K)
: vertical grid spacing $\Delta\sigma$ at cell centre

GZ0C(K)
: σ-coordinate at cell centre

GZ1U(K,J,I)
: vertical grid spacing Δx_3 at U-node and old time step t^n (m)

GZ1V(K,J,I)
: vertical grid spacing Δx_3 at V-node and old time step t^n (m)

GZ1W(K,J,I)
: vertical grid spacing Δx_3 at W-node and old time step t^n (m)

GZ2U(K,J,I)
: vertical grid spacing Δx_3 at U-node and new time step t^{n+1} (m)

GZ2UX(K,J,I)
: vertical grid spacing Δx_3 at X-node and new time step t^{n+1} (m)

GZ2V(K,J,I)
: vertical grid spacing Δx_3 at V-node and new time step t^{n+1} (m)

GZ2VX(K,J,I)
: vertical grid spacing Δx_3 at Y-node and new time step t^{n+1} (m)

GZ2W(K,J,I)
: vertical grid spacing Δx_3 at W-node and new time step t^{n+1} (m)

H1ATC(J,I)
: total water depth at cell centre and old time step t^n (m)

H1ATU(J,I)
: total water depth at U-node and old time step t^n (m)

H1ATV(J,I)
: total water depth at V-node and old time step t^n (m)

H2ATC(J,I)
: total water depth at cell centre and new time step t^{n+1} (m)

H2ATU(J,I)
: total water depth at U-node and new time step t^{n+1} (m)

H2ATV(J,I)
: total water depth at V-node and new time step t^{n+1} (m)

UD2ATC(J,I)
: depth-integrated $\overline{U}$-current at cell centre (m^2/s)

UD2ATV(J,I)
: depth-integrated $\overline{U}$-current at V-node (m^2/s)

U1ATC(K,J,I)
: u-current at cell centre and old time step t^n (m/s)

U1ATV(K,J,I)
: u-current at V-node and old time step t^n (m/s)

U2ATC(K,J,I)
: u-current at cell centre and new time step t^{n+1} (m/s)

U2ATV(K,J,I)
: u-current at V-node and new time step t^{n+1} (m/s)

VD2ATC(J,I)
: depth-integrated $\overline{V}$-current at cell centre (m^2/s)

VD2ATU(J,I)
: depth-integrated $\overline{V}$-current at U-node (m^2/s)

V1ATC(K,J,I)
: v-current at cell centre and old time step t^n (m/s)

V1ATU(K,J,I)
: v-current at U-node and old time step t^n (m/s)

V2ATC(K,J,I)
: v-current at cell centre and new time step t^{n+1} (m/s)

V2ATU(K,J,I)
: v-current at U-node and new time step t^{n+1} (m/s)

W2ATC(K,J,I)
: vertical current $J\tilde{w}$ at cell centre (m/s)

Chapter 2

Program Variables and Constants

The aim of this chapter is to describe all global variables and constants in the COHERENS program, which are either referenced in a COMMON or defined in a PARAMETER statement. A series of tables is produced with the following columns:

- Name: the FORTRAN name of the variable or constant as used in the program, given in alphabetical order in each table.
- Dimensions: the array dimensions.
- Type: REAL, INTEGER, CHARACTER or LOGICAL.
- Range: the range of values allowed in the program (only in Table 5.2.1 describing the model switches).
- Symbol: the symbolic name used in the COHERENS documentation.
- Value: the value of a model constant used in the program which is in most cases the default value which may be reset by the user.
- Unit: the unit of the variable or model constant.
- Include: the prefix of the *inc*-file which contains the COMMON or PARAMETER statement where the variable or constant is referenced or defined.
- Defined: the program unit where the variable or constant is defined. Note that a model parameter, whose value is not set by the user in subroutine DEFCON, has a default value, defined by the program in DEFAULT.

A short description is given on each second line.

Table 5.2.1: Model switches.

Name	Type	Range	Value	Unit	Include	Defined
IADVC	INTEGER	0–4	3	—	*runp*	DEFCON
	Selects the type of advection scheme in the momentum equations. 0 : horizontal and vertical advection disabled 1 : upwind scheme 2 : Lax-Wendroff scheme in the horizontal, central scheme in the vertical 3 : TVD scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical 4 : as the previous case now using the monotonic limiter					
IADVS	INTEGER	0–4	3	—	*runp*	DEFCON
	Selects the type of advection scheme in the scalar transport equations. 0 : horizontal and vertical advection disabled 1 : upwind scheme 2 : Lax-Wendroff scheme in the horizontal, central scheme in the vertical 3 : TVD scheme using the superbee limiter as a weighting function between the upwind scheme and either the Lax-Wendroff scheme in the horizontal or the central scheme in the vertical 4 : as the previous case now using the monotonic limiter					
IADVWB	INTEGER	0–4	3	—	*runp*	DEFCON
	Selects the type of scheme for vertical advection in the biological and sediment modules if IADVS $= 0$. 0 : vertical sinking disabled 1 : upwind scheme 2 : central scheme 3 : TVD scheme using the superbee limiter as a weighting function between the upwind scheme and the central scheme 4 : as the previous case now using the monotonic limiter					
IAHDHT	INTEGER	0–1	0	—	*partur*	DEFCON
	Disables/enables the evaluation of the advective (if IADVS > 0) and horizontal diffusion (if IODIF > 0) terms in the turbulence transport equations. 0 : advection and horizontal diffusion disabled 1 : advection and horizontal diffusion enabled					
IBSTR	INTEGER	0–2	1	—	*runp*	DEFCON
	Selects the bottom stress formulation. 0 : zero bottom stress 1 : quadratic law (3.1.185) 2 : linear friction law (3.1.186)					

Table 5.2.1: continued.

Name	Type	Range	Value	Unit	Include	Defined
IDRAG	INTEGER	0–4	0	—	*runp*	DEFCON
	Selects the formulation for the (neutral) surface drag coefficient C_D^s. 0 : constant value (0.0013) 1 : Large and Pond (1981) formulation (3.1.219) 2 : Smith and Banke (1975) formulation (3.1.220) 3 : Geernaert et al. (1986) formulation (3.1.221) 4 : Charnock (1955) formulation (3.1.222)					
IFLUFF	INTEGER	0–1	1	—	*runp*	DEFCON
	Disables/enables the fluff layer in the biological module. 0 : fluff layer disabled 1 : fluff layer enabled					
IGRDIM	INTEGER	1, 3	3	—	*runp.inc*	DEFCON
	Selects the number of dimensions of the numerical grid. 1 : 1-D application in the vertical as described in Sections III-1.1.4 and IV-2.9 3 : 3-D application					
IGTRH	INTEGER	0–1	0	—	*runp*	DEFCON
	Selects the type of grid and coordinates. 0 : Cartesian grid and coordinates 1 : spherical grid and coordinates					
ILENG	INTEGER	1–4	4	—	*partur*	DEFCON
	Selects the formulation for the mixing length if IOPTK = 2. 1 : parabolic law (3.1.131) 2 : “quasi-parabolic” law (3.1.132) 3 : “Xing” formulation (3.1.131), (3.1.133) 4 : “Blackadar” formulation (3.1.134)–(3.1.135)					
ILIM	INTEGER	0–1	1	—	*partur*	DEFCON
	Disables/enables the use of limiting conditions in the turbulence module if IOPTK = 2. 0 : limiting conditions disabled 1 : limiting conditions enabled					
IODIF	INTEGER	0–2	0	—	*runp*	DEFCON
	Selects the type of scheme for horizontal diffusion. 0 : horizontal diffusion disabled 1 : using uniform values for the diffusion coefficients ν_H and λ_H 2 : diffusion coefficients ν_H and λ_H determined using the Smagorinsky formulation (3.1.151)–3.1.152) or (3.1.153)–(3.1.154)					

Table 5.2.1: continued.

Name	Type	Range	Value	Unit	Include	Defined
IOPTB	INTEGER	0–1	0	—	*runp*	DEFCON
	Disables/enables the biological module. 0 : biological module disabled 1 : biological module enabled					
IOPTC	INTEGER	0–2	0	—	*runp*	DEFCON
	Disables/enables the (Eulerian) contaminant module. 0 : contaminant module disabled 1 : contaminant module enabled without horizontal and vertical diffusion 2 : contaminant module enabled with horizontal (if IODIF > 0) and vertical diffusion					
IOPTD	INTEGER	0–2	1	—	*runp*	DEFCON
	Selects the equation of state and the formulation for the expansion coefficients β_S and β_T. 0 : uniform density, zero values for β_S and β_T 1 : linear equation of state, uniform values for β_S and β_T 2 : density and β_S, β_T determined from the general equation of state of seawater (UNESCO, 1981) as described in Section III–1.5					
IOPTHE	INTEGER	0–2	0	—	*runp*	DEFCON
	Disables/enables the temperature module. 0 : temperature module disabled 1 : temperature module enabled, all solar radiation absorbed at the sea surface (optical module disabled) 2 : temperature module enabled, solar radiation absorbed throughout the water column (optical module enabled)					
IOPTK	INTEGER	0–2	2	—	*runp*	DEFCON
	Selects the general type of turbulence scheme. 0 : uniform eddy coefficients ν_T and λ_T 1 : ν_T and λ_T determined using one of the algebraic formulations described in Section III–1.2.1 (further selected by ITFORM) 2 : ν_T and λ_T determined using one of the turbulence closure schemes described in Section III–1.2.2 (further selected by NTRANS, ITCPAR, ISTPAR, ILENG, ILIM, IAHDHT)					

Table 5.2.1: continued.

Name	Type	Range	Value	Unit	Include	Defined
IOPTM	INTEGER	0–5	0	—	*runp*	DEFCON
	Selects the type of meteorological forcing. 0 : zero surface fluxes for momentum, temperature and salinity; solar irradiance set to zero 1 : uniform surface stress, zero surface fluxes for temperature and salinity; solar irradiance set to zero 2 : meteorological forcing data (wind components, air temperature, relative humidity, cloud coverage and either the evaporation minus precipitation rate $E_{vap} - R_{pr}$ or the precipitation rate R_{pr}) supplied as uniform in space but non-uniform in time 3 : the previous forcing data plus atmospheric pressure now supplied as non-uniform in space and time 4 : meteorological forcing data (wind components, non-solar heat flux, surface solar irradiance and $E_{vap} - R_{pr}$) supplied as uniform in space but non-uniform in time 5 : the previous meteorological forcing data plus atmospheric pressure now supplied as non-uniform in space and time					
IOPTP	INTEGER	0–2	0	—	*runp*	DEFCON
	Disables/enables the Lagrangian particle module. 0 : particle module disabled 1 : particle module enabled without horizontal and vertical diffusion 2 : particle module enabled with horizontal and vertical diffusion					
IOPTS	INTEGER	0–1	0	—	*runp*	DEFCON
	Disables/enables the sediment module. 0 : sediment module disabled 1 : sediment module enabled					
IOPTSA	INTEGER	0–2	0	—	*runp*	DEFCON
	Disables/enables the salinity module. 0 : salinity module disabled 1 : salinity module enabled with the evaporation minus precipitation rate $E_{vap} - R_{pr}$ supplied by the user 2 : salinity module enabled with the precipitation rate R_{pr} supplied by the user					

Table 5.2.1: continued.

Name	Type	Range	Value	Unit	Include	Defined
IOPTW	INTEGER	0–3	0	—	*runp*	DEFCON
	Disables/enables the wave-current interaction module and selects the type of wave input. 0 : wave-current module disabled 1 : wave-current module enabled, wave data supplied as uniform in space and time 2 : wave-current module enabled, wave data supplied as uniform in space but non-uniform in time 3 : wave-current module enabled, wave data supplied as non-uniform in space and time					
IOPT2	INTEGER	0–1	1	—	*runp*	DEFCON
	Disables/enables the solution of the 2-D mode equations. 0 : mode splitting disabled 1 : mode splitting enabled					
IOPT3	INTEGER	0–1	1	—	*runp*	DEFCON
	Disables/enables the update of the 3-D horizontal and vertical current fields. 0 : update of the 3-D current disabled 1 : update of the 3-D current enabled					
IOUTF	INTEGER	0–1	0	—	*runp*	DEFCON
	Disables/enables time-averaged output. 0 : time-averaged output disabled 1 : time-averaged output enabled					
IOUTP	INTEGER	0–1	0	—	*runp*	DEFCON
	Disables/enables output of particle positions from the Lagrangian particle module. 0 : particle output disabled 1 : particle output enabled					
IOUTS	INTEGER	0–1	1	—	*runp*	DEFCON
	Disables/enables time series output. 0 : time series output enabled 1 : time series output disabled					
IOUTT	INTEGER	0–2	0	—	*runp*	DEFCON
	Disables/enables harmonic analysis. 0 : harmonic analysis disabled 1 : harmonic analysis enabled without calculation of tidal ellipse parameters 2 : harmonic analysis enabled including the calculation of tidal ellipse parameters					

Table 5.2.1: continued.

Name	Type	Range	Value	Unit	Include	Defined
ISTPAR	INTEGER	1–2	1	—	*partur*	DEFCON
	Selects whether the stability functions in the turbulence module (IOPTK $= 2$) are evaluated as a function of the stability parameter or the Richardson number. 1 : as a function of the stability parameter (equations (3.1.118)–(3.1.119) or (3.1.120)–(3.1.121)) 2 : as a function of the Richardson number (equations (3.1.122) or (3.1.123))					
ITCPAR	INTEGER	1–2	2	—	*partur*	DEFCON
	Selects either the $k - l$ or the $k - \varepsilon$ formulation in the turbulence module (IOPTK $= 2$). 1 : $k - l$ formulation with ν_T and λ_T evaluated using (3.1.115); the stability functions calculated using either (3.1.118)–(3.1.119) or (3.1.122); kl-equation (3.1.125) solved if NTRANS $= 2$ 2 : $k - \varepsilon$ formulation with ν_T and λ_T evaluated using (3.1.117); the stability functions calculated using either (3.1.120)–(3.1.121) or (3.1.123); ε-equation (3.1.128) solved if NTRANS $= 2$					
ITDIF	INTEGER	0–1	0	—	*runp*	DEFCON
	Selects whether stratification effects are included in the formulation for the surface drag and exchange coefficients C_D^s, C_E and C_H. 0 : stratification effects not included 1 : stratification effects included using the theory described in Section III-1.6.4					
ITFORM	INTEGER	1–5	3	—	*partur*	DEFCON
	Selects the type of (algebraic) turbulence scheme if IOPTK $= 1$. 1 : Pacanowski-Philander formulation (3.1.86)–(3.1.88) 2 : Munk-Anderson formulation (3.1.95)–(3.1.98) 3 : flow-dependent formulation (3.1.101)–(3.1.102) using (3.1.108) 4 : flow-dependent formulation (3.1.101)–(3.1.102) using (3.1.109) 5 : flow-dependent formulation (3.1.101)–(3.1.102) using (3.1.110)					
NTRANS	INTEGER	0–2	1	—	*partur*	DEFCON
	Selects the number of transport equations in the turbulence scheme if IOPTK $= 2$ 0 : zero-equation model using either (3.1.136)–(3.1.137) or (3.1.140) 1 : one-equation model solving the k-equation (3.1.114) 2 : two-equation model solving the k-equation (3.1.114) and either the kl-equation (3.1.125) if ITCPAR $= 1$ or the ε-equation (3.1.128) if ITCPAR $= 2$					

Table 5.2.2: Time parameters.

Name	Type	Symbol	Value	Unit	Include	Defined
DELM	REAL	—	—	s	*time*	MAINPROG
	Time step for new meteorological input.					
DELT	REAL	Δt_{2D}	—	s	*time*	DEFCON
	Time step for the 2-D (barotropic) mode.					
DELW	REAL	—	—	s	*time*	MAINPROG
	Time step for new surface wave input.					
DEL3	REAL	Δt_{3D}	—	s	*time*	MAINPROG
	Time step for the 3-D (baroclinic) mode, the update of all scalar quantities and the update of the particles in the Lagrangian module.					
DTMAX	REAL	—	—	s	*time*	INITC
	Maximum value for Δt_{2D} allowed by the CFL-stability criterion (3.4.6).					
HOUR	REAL	—	—	h	*time*	MAINPROG
	Number of hours elapsed since the start of the simulation.					
IBDATE	INTEGER	—	—	—	*time*	DEFCON
	Start date of the simulation in the format MMDDHHMM (month-day-hour-minute).					
IBYEAR	INTEGER	—	—	1998	*time*	DEFCON
	Start year of the simulation.					
ICBBC	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new biological input at open boundaries.					
ICCBC	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new contaminant input (Eulerian module) at open boundaries.					
ICHBC	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new physical input at open boundaries (temperature, salinity, 3-D horizontal currents).					
ICMET	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new meteorological input.					
ICPBC	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new contaminant input (Lagrangian module) at open boundaries.					
ICWAV	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new surface wave input.					
IC2BC	INTEGER	—	0	—	*count*	DEFCON
	Number of 2-D time steps for new 2-D mode input (residuals, amplitudes and phases) at open boundaries.					

Table 5.2.2: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
IC3D	INTEGER	M_t	1	—	*count*	DEFCON
	Number of 2-D time steps within one 3-D (baroclinic) time step.					
IDATE	INTEGER	—	—	—	*time*	MAINPROG
	Current date in the format MMDDHHMM (month-day-hour-minute).					
IEDATE	INTEGER	—	—	—	*time*	DEFCON
	End date of the simulation in the format MMDDHHMM (month-day-hour-minute).					
IEYEAR	INTEGER	—	—	1998	*time*	DEFCON
	End year of the simulation.					
IYEAR	INTEGER	—	—	—		MAINPROG
	Current year.					
NSTEP	INTEGER	M_{tot}	—	—	*time*	INITC
	Total number of 2-D time steps used in the simulation.					
NT	INTEGER	—	—	—	*time*	MAINPROG
	Time index representing the number of 2-D time steps elapsed since the start of the simulation.					

Table 5.2.3: Grid parameters.

Name	Type	Symbol	Value	Unit	Include	Defined
DEPUN	REAL	h	—	m	*depths*	DEFCON
	Uniform mean water depth used if IGRDIM $= 1$.					
DLAREF	REAL	ϕ	0	degrees	*grid*	DEFCON
	Reference latitude in decimal degrees (used only if IGTRH $= 0$).					
DLOREF	REAL	λ	0	degrees	*grid*	DEFCON
	Reference longitude in decimal degrees (used only if IGTRH $= 0$).					
NC	INTEGER	N_x	—	—	*param*	*param.inc*
	Number of grid cells in the X-direction.					
NOBU	INTEGER	—	—	—	*param*	*param.inc*
	Number of open boundaries at U-nodes.					
NOBV	INTEGER	—	—	—	*param*	*param.inc*
	Number of open boundaries at V-nodes.					
NR	INTEGER	N_y	—	—	*param*	*param.inc*
	Number of grid cells in the Y-direction.					
NZ	INTEGER	N_z	—		*param*	*param.inc*
	Number of grid cells in the vertical.					
REARTH	REAL	R	6371×10^3	m	*consts*	INICON
	Radius of the Earth.					

Table 5.2.4: Grid arrays.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
CORIOL	NR	REAL	—	rad/s	*grid*	TRANSH
	The Coriolis frequency f or $2\Omega\sin\phi$ at the cell centres and U-nodes.					
CORIOLV	NR+1	REAL	—	rad/s	*grid*	TRANSH
	The Coriolis frequency f or $2\Omega\sin\phi$ at the V-nodes.					
COSPHI	NR	REAL	—	—	*grid*	TRANSH
	The value of $\cos\phi$ at the cell centres and U-nodes in the case of a spherical grid or set to 1 in the Cartesian case.					
COSPHIV	NR+1	REAL	—	—	*grid*	TRANSH
	The value of $\cos\phi$ at the V-nodes in the case of a spherical grid or set to 1 in the Cartesian case.					
DEP	NR,NC	REAL	h	m	*depths*	DEFGRID
	Mean water depth.					
DLAT	NR	REAL	ϕ	degrees	*grid*	TRANSH
	Latitude at the cell centre in decimal degrees.					
DLON	NC	REAL	λ	degrees	*grid*	TRANSH
	Longitude at the cell centres in decimal degrees.					
GX0	NC+1	REAL	x_1, λ	m,degrees	*grid*	DEFGRID
	x_1-coordinate or longitude (decimal degrees) of the cell corners.					
GX2	NR,NC	REAL	—	m	*grid*	TRANSH
	Grid spacing in the X-direction at the cell centre given by Δx_1 in Cartesian or $R\cos\phi\Delta\lambda$ in spherical coordinates.					
GY0	NR+1	REAL	x_2, ϕ	m,degrees	*grid*	DEFGRID
	x_2-coordinate or latitude (decimal degrees) of the cell corners.					
GY2	NR	REAL	—	m	*grid*	TRANSH
	Grid spacing in the Y-direction at the cell centres given by Δx_2 in Cartesian or $R\Delta\phi$ in spherical coordinates.					
GZ0	NZ+1	REAL	σ	—	*grid*	DEFGRID
	Vertical σ-coordinate of the W-nodes given by $(k-1)/N_z$ with $k = 1, \cdots, N_z + 1$ by default.					
GZ1	NZ,NR,NC	REAL	Δx_3^n	m	*grid*	CRRNT3P
	Vertical grid spacing $\Delta x_3 = J$ at the cell centre and the old (baroclinic) time t^n.					
GZ2	NZ,NR,NC	REAL	Δx_3^{n+1}	m	*grid*	TRANSV
	Vertical grid spacing $\Delta x_3 = J$ at the cell centre and the new (baroclinic) time t^{n+1}.					
IOBU	0:NOBU	INTEGER	—	—	*grid*	DEFGRID
	The X-indices of the open boundary points at U-nodes.					
IOBV	0:NOBV	INTEGER	—	—	*grid*	DEFGRID
	The X-indices of the open boundary points at V-nodes.					
JOBU	0:NOBU	INTEGER	—	—	*grid*	DEFGRID
	The Y-indices of the open boundary points at U-nodes.					

Table 5.2.4: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
JOBV	0:NOBV	INTEGER	—	—	*grid*	DEFGRID
	The Y-indices of the open boundary points at V-nodes.					
NPIX	NR,NC+1	INTEGER	—	—	*grid*	DEFGRID
	Pointer array at the U-nodes to determine the type of boundary face. 0 : solid (land) interface 1 : wet interior face 2 : open sea boundary face 3 : river boundary face					
NPIY	NR+1,NC	INTEGER	—	—	*grid*	DEFGRID
	Pointer array at the V-nodes to determine the type of boundary face. 0 : solid (land) interface 1 : wet interior face 2 : open sea boundary face 3 : river boundary face					
NWD	NR,NC	INTEGER	—	—	*grid*	DEFGRID
	Pointer array at the cell centres to determine the type of grid cell. 0 : dry cell 1 : wet cell					
SOUTOB	0:NOBV	LOGICAL	—	—	*grid*	BCSIN
	Determines whether an open boundary at a V-node is located at the southern or northern face of the wet grid cell adjacent to the boundary. .TRUE.: southern boundary face .FALSE.: northern boundary face					
SPHCUR	NR	REAL	—	m^{-1}	*grid*	TRANSH
	Represents the value of $\tan\phi/R$ at the cell centres and U-nodes in the case of a spherical grid or set to zero in the Cartesian case.					
SPHCURV	NR+1	REAL	—	m^{-1}	*grid*	TRANSH
	Represents the value of $\tan\phi/R$ at the V-nodes in the case of a spherical grid or set to zero in the Cartesian case.					
WESTOB	0:NOBU	LOGICAL	—	—	*grid*	BCSIN
	Determines whether an open boundary at a U-node is located at the western or eastern face of the wet grid cell adjacent to the boundary. .TRUE.: western boundary face .FALSE.: eastern boundary face					

Table 5.2.5: Parameters for the physics.

Name	Type	Symbol	Value	Unit	Include	Defined
ATCF1UN	REAL	k_1	10.0	m^{-1}	*phycon*	DEFCON
	The optical attenuation coefficient k_1 for infrared radiation.					
ATCF2UN	REAL	k^w_{20}	2.06	m^{-1}	*phycon*	DEFCON
	The fresh water value of the optical attenuation coefficient k_2 for monochromatic PAR (see equation (3.1.159)).					
CKAR	REAL	κ	0.4	—	*phycon*	INICON
	von Kármán's constant.					
CONV	REAL	—	$\pi/180$	—	*consts*	*consts.inc*
	Conversion factor from degrees to radians.					
CP	REAL	c_p	3987.5	$\text{J kg}^{-1}\ (^0\text{C})^{-1}$	*phycon*	INICON
	The specific heat of seawater at constant pressure.					
EPSSAL	REAL	ϵ^S	0.05714	$\text{m}^{-1}\ (\text{PSU})^{-1}$	*phycon*	DEFCON
	Slope of the linear relationship between the attenuation coefficient k_2 and salinity (see equation (3.1.159)).					
G	REAL	g	9.81	m/s^2	*consts*	*consts.inc*
	Gravitational acceleration of the Earth.					
HEXPUN	REAL	Δ_{opt}	7.0	m	*phycon*	DEFCON
	Thickness of the optical surface layer for hyper-exponential PAR decay (see equation (3.1.160)).					
PI	REAL	π	3.141592741	—	*consts*	*consts.inc*
	The number π.					
R0REF	REAL	ρ_0	1025.0	kg/m^3	*phycon*	DEFCON
	The reference density ρ_0.					
R1OPTUN	REAL	R_1	0.54	—	*phycon*	DEFCON
	The infrared fraction of solar irradiance.					
R2OPTUN	REAL	R_2	0.4	—	*phycon*	DEFCON
	Near surface attenuation factor for hyperexponential decay (see equation (3.1.160)).					
SBETUN	REAL	β_S	7.6×10^{-4}	$(\text{PSU})^{-1}$	*phycon*	DEFCON
	Uniform expansion coefficient for salinity as used in the linear equation of state (3.1.162).					
SREF	REAL	S_0	33.0	PSU	*phycon*	DEFCON
	Reference salinity S_0.					
TBETUN	REAL	β_T	1.8×10^{-4}	$(^0\text{C})^{-1}$	*phycon*	DEFCON
	Uniform expansion coefficient for temperature as used in the linear equation of state (3.1.162).					
TREF	REAL	T_0	12.0	^0C	*phycon*	DEFCON
	Reference temperature T_0.					

Table 5.2.6: Physical arrays.

Name	Dimensions	Type	Symbol	Value	Unit	Include	Defined
ATCFCOR2	NZ,NR,NC	REAL	k_2	—	m^{-1}	*optics*	IRRAD
	The optical attenuation coefficient k_2 for non-infrared radiation in the physical model (see equation (3.1.159)).						
ATCF1	NR,NC	REAL	k_1	10.0	m^{-1}	*optics*	INITC
	The optical attenuation coefficient k_1 for infrared radiation.						
ATCF2	NR,NC	REAL	k^w_{20}	2.06	m^{-1}	*optics*	INITC
	The fresh water value of the optical attenuation coefficient k_2 for non-infrared radiation in the physical model (see equation (3.1.159)).						
BUOY	NZ,NR,NC	REAL	b	—	$\mathrm{m/s}^2$	*concn*	SEARHO
	Buoyancy b.						
HEXP	NR,NC	REAL	Δ_{opt}	7.0	m	*optics*	INITC
	Thickness of the optical surface layer for hyper-exponential PAR decay (see equation (3.1.160)).						
QHEAT	NZ,NR,NC	REAL	—	—	$\mathrm{W/m}^3$	*optics*	IRRAD
	Amount of heat absorbed per unit volume and time, given by $\partial I/\partial x_3$.						
RO	NZ,NR,NC	REAL	ρ	—	$\mathrm{kg/m}^3$	*concn*	SEARHO
	Density ρ.						
R1OPT	NR,NC	REAL	R_1	0.54	—	*optics*	INITC
	The infrared fraction of solar irradiance.						
R2OPT	NR,NC	REAL	R_2	0.4	—	*optics*	INITC
	Near surface attenuation factor for hyperexponential decay.						
S	NZ,NR,NC	REAL	S	—	PSU	*concn*	SALT
	Salinity S.						
SBET	NZ,NR,NC	REAL	β_S	—	$(\mathrm{PSU})^{-1}$	*concn*	SEARHO
	The expansion coefficient β_S for salinity.						
T	NZ,NR,NC	REAL	T	—	$^0\mathrm{C}$	*concn*	HEAT
	Temperature T.						
TBET	NZ,NR,NC	REAL	β_T	—	$(^0\mathrm{C})^{-1}$	*concn*	SEARHO
	The expansion coefficient β_T for temperature.						
UADHDEV	NR,NC+1	REAL	—	—	$\mathrm{m}^2/\mathrm{s}^2$	*crrnts*	UCALC
	The depth integral $\overline{D}^h_1 - \overline{A}^h_1$ or $\overline{D}^h_\lambda - \overline{A}^h_\lambda$ in the $\overline{U}$-momentum equation (3.1.34) or (3.1.61).						
UAH2D	NR,NC+1	REAL	—	—	$\mathrm{m}^2/\mathrm{s}^2$	*crrnts*	HAD2DU
	The advective term $\overline{\mathcal{A}_h}(\overline{U}^m)$ in the $\overline{U}$-momentum equation (3.4.13) (including an extra term in the spherical case).						

Table 5.2.6: continued.

Name	Dimensions	Type	Symbol	Value	Unit	Include	Defined
UDH2D	NR,NC+1	REAL	—	—	$\mathrm{m^2/s^2}$	*crrnts*	HAD2DU
	The horizontal diffusion term $\overline{\mathcal{D}_{xx}}(\overline{U}^m)+\overline{\mathcal{D}_{yx}}(\overline{U}^m,\overline{V}^m)$ in the $\overline{U}$-momentum equation (3.4.13) (including an extra term in the spherical case).						
UDP	NR,NC+1	REAL	$\overline{U}^p$	—	$\mathrm{m^2/s}$	*crrnts*	CRRNT3P
	The depth integral of the predicted current as given by (3.4.11).						
UDQDEN	NR,NC+1	REAL	$\overline{Q}_1$,$\overline{Q}_\lambda$	—	$\mathrm{m^2/s^2}$	*crrnts*	DENSTY
	The depth-integrated value of the baroclinic pressure gradient in the X-direction.						
UD2	NR,NC+1	REAL	$\overline{U}$	—	$\mathrm{m^2/s}$	*crrnts*	UDCALC
	The depth-integrated current $\overline{U}$.						
UD2F	NR,NC+1	REAL	$\overline{U}^F$	—	$\mathrm{m^2/s}$	*crrnts*	CRRNT2
	The value of $\overline{U}$ averaged over one baroclinic (3-D) time step as given by (3.4.18).						
UQDEN	NZ,NR,NC+1	REAL	Q_1,Q_λ	—	$\mathrm{m/s^2}$	*crrnts*	DENSTY
	The baroclinic pressure gradient in the X-direction.						
U1	NZ,NR,NC+1	REAL	u^n	—	m/s	*crrnts*	CRRNT3P
	The X-component of the current at the old time t^n.						
U2	NZ,NR,NC+1	REAL	u^{n+1}	—	m/s	*crrnts*	UCALC
	The X-component of the current at the new time t^{n+1}.						
U2F	NZ,NR,NC+1	REAL	u^F	—	m/s	*crrnts*	CRRNT3C
	The "filtered" advective velocity in the X-direction as defined by (3.4.21).						
VADHDEV	NR+1,NC	REAL	—	—	$\mathrm{m^2/s^2}$	*crrnts*	VCALC
	The depth integral $\overline{D}_2^h-\overline{A}_2^h$ or $\overline{D}_\phi^h-\overline{A}_\phi^h$ in the $\overline{V}$-momentum equation (3.1.35) or (3.1.62).						
VAH2D	NR+1,NC	REAL	—	—	$\mathrm{m^2/s^2}$	*crrnts*	HAD2DV
	The advective term $\overline{\mathcal{A}_h}(\overline{V}^m)$ in the $\overline{V}$-momentum equation (3.4.14) (including an extra term in the spherical case).						
VDH2D	NR+1,NC	REAL	—	—	$\mathrm{m^2/s^2}$	*crrnts*	HAD2DV
	The horizontal diffusion term $\overline{\mathcal{D}_{xy}}(\overline{U}^m,\overline{V}^m)+\overline{\mathcal{D}_{yy}}(\overline{V}^m)$ in the $\overline{V}$-momentum equation (3.4.14) (including an extra term in the spherical case).						
VDP	NR+1,NC	REAL	$\overline{V}^p$	—	$\mathrm{m^2/s}$	*crrnts*	CRRNT3P
	The depth integral of the predicted current as given by (3.4.11).						
VDQDEN	NR+1,NC	REAL	$\overline{Q}_2$,$\overline{Q}_\phi$	—	$\mathrm{m^2/s^2}$	*crrnts*	DENSTY
	The depth-integrated value of the baroclinic pressure gradient in the Y-direction.						
VD2	NR+1,NC	REAL	$\overline{V}$	—	$\mathrm{m^2/s}$	*crrnts*	VDCALC
	The depth-integrated current $\overline{V}$.						
VD2F	NR+1,NC	REAL	$\overline{V}^F$	—	$\mathrm{m^2/s}$	*crrnts*	CRRNT2
	The value of $\overline{V}$ averaged over one baroclinic (3-D) time step as given by (3.4.18).						
VQDEN	NZ,NR+1,NC	REAL	Q_2,Q_ϕ	—	$\mathrm{m/s^2}$	*crrnts*	DENSTY
	The baroclinic pressure gradient in the Y-direction.						

Table 5.2.6: continued.

Name	Dimensions	Type	Symbol	Value	Unit	Include	Defined
V1	NZ,NR+1,NC	REAL	v^n	—	m/s	*crrnts*	CRRNT3P
	The Y-component of the current at the old time t^n.						
V2	NZ,NR+1,NC	REAL	v^{n+1}	—	m/s	*crrnts*	VCALC
	The Y-component of the current at the new time t^{n+1}.						
V2F	NZ,NR+1,NC	REAL	v^F	—	m/s	*crrnts*	CRRNT3C
	The "filtered" advective velocity in the Y-direction as defined by (3.4.22).						
W2	NZ+1,NR,NC	REAL	$J\tilde{w}$	—	m/s	*crrnts*	WCALC
	The transformed vertical current $J\tilde{w}$.						
W2PHYS	NZ,NR,NC	REAL	w	—	m/s	*crrnts*	WPCALC
	The "physical" vertical current w at the cell centres.						
ZETA1	NR,NC	REAL	ζ^n	—	m	*elev*	CRRNT3P
	The sea surface elevation ζ at the old time t^n.						
ZETA2	NR,NC	REAL	ζ^{n+1}	—	m	*elev*	CONTNY
	The sea surface elevation ζ at the new time t^{n+1}.						

Table 5.2.7: Turbulence parameters.

Name	Type	Symbol	Value	Unit	Include	Defined
ABLAC	REAL	α_1	0.2	—	*turcon*	DEFCON
	Parameter α_1 in the “Blackadar” mixing length (3.1.135).					
ADCNU	REAL	C_ν	2.0	—	*turcon*	DEFCON
	Parameter which measures the thickness of the bottom boundary layer in (3.1.111).					
ADD1	REAL	δ_1	0.0	—	*turcon*	DEFCON
	Parameter δ_1 in the expression (3.1.103) for the vertical profile $\Phi(\sigma)$ used in the flow-dependent formulation (3.1.101)–(3.1.102).					
ADD2	REAL	δ_2	0.0	—	*turcon*	DEFCON
	Parameter δ_2 in the expression (3.1.103) for the vertical profile $\Phi(\sigma)$ used in the flow-dependent formulation (3.1.101)–(3.1.102).					
ADK1	REAL	K_1	2.5×10^{-3}	—	*turcon*	DEFCON
	Parameter K_1 in the expression (3.1.108) for the parameter α used in the flow-dependent formulation (3.1.101)–(3.1.102).					
ADK2	REAL	K_2	2×10^{-5}	—	*turcon*	DEFCON
	Parameter K_2 in the expression (3.1.109) for the parameter α used in the flow-dependent formulation (3.1.101)–(3.1.102).					
ADLAM	REAL	λ_*	0.0	m	*turcon*	DEFCON
	Parameter relating the surface friction velocity u_{*s} to the wind-induced eddy coefficient ν_w in (3.1.106).					
ADR1	REAL	r_1	1.0	—	*turcon*	DEFCON
	Parameter r_1 in the expression (3.1.103) for the vertical profile $\Phi(\sigma)$ used in the flow-dependent formulation (3.1.101)–(3.1.102).					
ADR2	REAL	r_2	1.0	—	*turcon*	DEFCON
	Parameter r_2 in the expression (3.1.103) for the vertical profile $\Phi(\sigma)$ used in the flow-dependent formulation (3.1.101)–(3.1.102).					
AMA	REAL	α_m	10.0	—	*turcon*	DEFCON
	Parameter α_m in the Munk-Anderson formulation (3.1.97).					
AMB	REAL	β_m	3.33	—	*turcon*	DEFCON
	Parameter β_m in the Munk-Anderson formulation (3.1.98).					
AMN1	REAL	n_1	0.5	—	*turcon*	DEFCON
	Exponent n_1 in the Munk-Anderson formulation (3.1.97).					
AMN2	REAL	n_2	1.5	—	*turcon*	DEFCON
	Exponent n_2 in the Munk-Anderson formulation (3.1.98).					

Table 5.2.7: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
AMVED0	REAL	ν_{0m}	0.06	m^2/s	*turcon*	DEFCON
	Neutral eddy coefficient in the Munk-Anderson formulation (3.1.95)–(3.1.96).					
BXING	REAL	β_1	-2.0	—	*turcon*	DEFCON
	Parameter β_1 in the “Xing” mixing length formulation (3.1.133).					
CMU	REAL	c_μ	—	—	*turcon*	INICON
	Value of the parameter $\varepsilon_0^{4/3}$.					
CM0	REAL	C_{m0}	0.1	—	*turcon*	DEFCON
	Coefficient C_{m0} in the Smagorinsky formulation (3.1.151) or (3.1.153) for the horizontal diffusion coefficient ν_H for momentum.					
CS0	REAL	C_{s0}	0.1	—	*turcon*	DEFCON
	Coefficient C_{s0} in the Smagorinsky formulation (3.1.151) or (3.1.153) for the horizontal diffusion coefficient λ_H for scalars.					
C1E	REAL	$c_{1\varepsilon}$	1.55	—	*turcon*	INICON
	Parameter $c_{1\varepsilon}$ in the ε-equation (3.1.128).					
C2E	REAL	$c_{2\varepsilon}$	1.92	—	*turcon*	INICON
	Parameter $c_{2\varepsilon}$ in the ε-equation (3.1.128).					
C3E1	REAL	$c_{3\varepsilon}$	0.2	—	*turcon*	INICON
	The parameter $c_{3\varepsilon}$ in the ε-equation (3.1.128) for the case of stable stratification.					
C3E2	REAL	$c_{3\varepsilon}$	1.0	—	*turcon*	INICON
	The parameter $c_{3\varepsilon}$ in the ε-equation (3.1.128) for the case of unstable stratification.					
DIFMOL	REAL	λ_b	10^{-6}	m^2/s	*turcon*	DEFCON
	Uniform background diffusion coefficient for scalars.					
EPS0	REAL	ε_0	—	—	*turcon*	INICON
	Parameter ε_0, defined by (3.1.124), used in the expression (3.1.116) for the dissipation rate ε. Value is 0.172 if ITCPAR = 1 and 0.188 if ITCPAR = 2.					
E1	REAL	E_1	1.8	—	*turcon*	INICON
	Parameter E_1 in the kl-equation (3.1.125).					
E2	REAL	E_2	1.33	—	*turcon*	INICON
	Parameter E_2 in the kl-equation (3.1.125).					
GAMAX	REAL	—	—	—	*turcon*	INICON
	Maximum value 0.560 for the stability parameter G_h if ITCPAR = 1 or maximum value 6.903 for the stability parameter α_N if ISTPAR = 2. The parameter is used when a limiting condition is applied for the mixing length (ILIM = 1). See Section III-1.2.2c for details.					
GAMIN	REAL	—	—	—	*turcon*	INICON
	Minimum value -0.046 for the stability parameter G_h if ITCPAR = 1 or minimum value -1.725 for the stabillity parameter α_N if ITCPAR = 2 (see equation (3.1.146)).					
HEDDUN	REAL	λ_H	0.0	m^2/s	*turcon*	DEFCON
	Uniform horizontal diffusion coefficient λ_H for scalars.					

Table 5.2.7: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
HEDVUN	REAL	ν_H	0.0	m^2/s	*turcon*	DEFCON
	Uniform horizontal diffusion coefficient ν_H for momentum.					
PPA	REAL	α_p	5.0	—	*turcon*	DEFCON
	Parameter α_p in the Pacanowski-Philander formulation (3.1.88).					
PPDIFBG	REAL	λ_{bp}	10^{-5}	m^2/s	*turcon*	DEFCON
	Background diffusion coefficient for scalars in the Pacanowski-Philander formulation (3.1.87).					
PPN	REAL	n_p	2.0	—	*turcon*	DEFCON
	Exponent n_p in the Paconowski-Philander formulation (3.1.86).					
PPVED0	REAL	ν_{0p}	0.01	m^2/s	*turcon*	DEFCON
	Neutral eddy coefficient in the Pacanowski-Philander formulation (3.1.86).					
PPVISBG	REAL	ν_{bp}	10^{-4}	m^2/s	*turcon*	DEFCON
	Background eddy coefficient for momentum in the Paconowski-Philander formulation (3.1.86).					
RMAXLAT	REAL	λ_{max}	4.0	—	*turcon*	DEFCON
	Maximum allowed value for the function g_m in the Munk-Anderson formulation (3.1.98).					
RMAXNUT	REAL	ν_{max}	3.0	—	*turcon*	DEFCON
	Maximum allowed value for the function f_m in the Munk-Anderson formulation (3.1.97) or for the function $f_p^{n_p}$ in (3.1.92).					
SIGE	REAL	σ_ε	1.3	—	*turcon*	INICON
	Parameter σ_ε in the vertical diffusion term of the ε-equation (3.1.128).					
SIGK	REAL	σ_k	—	—	*turcon*	INICON
	Parameter σ_k in the vertical diffusion term of the k-equation (3.1.114). Value is 1.96 if ITCPAR = 1 and 1.0 if ITCPAR = 2.					
TKEMIN	REAL	k_{min}	10^{-6}	J/kg	*turcon*	DEFCON
	Minimum value for turbulence energy in the case that limiting conditions are enabled (ILIM = 1).					
VISMOL	REAL	ν_b	10^{-6}	m^2/s	*turcon*	DEFCON
	Uniform background coefficient for momentum.					
Z0BOT	REAL	z_{0b}	0.0	m	*turcon*	DEFCON
	Bottom roughness length scale as used in (3.1.130), (3.1.126) and (3.1.133).					
Z0SUR	REAL	z_{0s}	0.0	m	*turcon*	DEFCON
	Surface roughness length scale as used in (3.1.130) and (3.1.126).					

Table 5.2.8: Turbulence arrays.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
BUPROD	NZ+1,NR,NC	REAL	—	W/kg	*turbke*	PRODUC
	Buoyancy source/sink term $-\lambda_T N^2$ in the turbulence transport equations.					
CLEV2	4	REAL	—	—	*turcon*	INICON
	Parameters (c_{21},c_{22},c_{23},c_{24}) in the zero-equation model as given by (3.1.138) or (3.1.139).					
CLEV25	6	REAL	—	—	*turcon*	INICON
	If ITCPAR = 1, the coefficients (0.556,2.18,20.4,53.1,0.699,17.3) in the expressions (3.1.118) for the stability functions (S_m,S_h). If ITCPAR = 2, the coefficients (0.108,0.0229,0.471,0.0275,0.177,0.403) in the expressions (3.1.120) for the stability functions (S_u,S_b).					
DHEDDYVC	NR,NC	REAL	$\overline{\nu_H}^c$	m^3/s	*visc*	HEDDY
	Depth-integrated horizontal diffusion coefficient for momentum at the cell centres.					
DHEDDYVU	NR,NC+1	REAL	$\overline{\nu_H}^u$	m^3/s	*visc*	HEDDY
	Depth-integrated horizontal diffusion coefficient for momentum at the U-nodes.					
DHEDDYVV	NR+1,NC	REAL	$\overline{\nu_H}^v$	m^3/s	*visc*	HEDDY
	Depth-integrated horizontal diffusion coefficient for momentum at the V-nodes.					
DISSW	NZ+1,NR,NC	REAL	ε	W/kg	*turbke*	DISSIP,DISLEN
	Turbulence dissipation rate ε.					
HEDDYDC	NZ,NR,NC	REAL	λ_H^c	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for scalars at the cell centres.					
HEDDYDU	NZ,NR,NC+1	REAL	λ_H^u	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for scalars at the U-nodes.					
HEDDYDV	NZ,NR+1,NC	REAL	λ_H^v	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for scalars at the V-nodes.					
HEDDYVC	NZ,NR,NC	REAL	ν_H^c	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for momentum at the cell centres.					
HEDDYVU	NZ,NR,NC+1	REAL	ν_H^u	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for momentum at the U-nodes.					
HEDDYVV	NZ,NR+1,NC	REAL	ν_H^v	m^2/s	*visc*	HEDDY,INITC
	Horizontal diffusion coefficient for momentum at the V-nodes.					
SHB	NZ+1,NR,NC	REAL	S_h,S_b	—	*turbke*	VEDDY2
	Stability coefficient S_h (ITCPAR = 1) or S_b (ITCPAR = 2) for scalars.					
SHPROD	NZ+1,NR,NC	REAL	—	W/kg	*turbke*	PRODUC
	Shear production term $\nu_T M^2$ in the turbulence transport equations.					

Table 5.2.8: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
SMU	NZ+1,NR,NC	REAL	S_m,S_u	—	*turbke*	VEDDY2
	Stability coefficient S_m (ITCPAR = 1) or S_u (ITCPAR = 2) for momentum.					
TKEOLD	NZ+1,NR,NC	REAL	k^n	J/kg	*turbke*	TURBEN
	Turbulence energy k at the old time level t^n.					
TKEW	NZ+1,NR,NC	REAL	k^{n+1}	J/kg	*turbke*	TURBEN,VEDDY2
	Turbulence energy k at the new time level t^{n+1}.					
VEDDYD	NZ+1,NR,NC	REAL	λ_T	m^2/s	*visc*	VEDDY1,VEDDY2
	Vertical diffusion coefficient for scalars.					
VEDDYE	NZ+1,NR,NC	REAL	—	m^2/s	*visc*	VEDDY2
	Vertical diffusion coefficient $\nu_T/\sigma_\varepsilon + \nu_b$ in the ε-equation (3.1.128).					
VEDDYK	NZ+1,NR,NC	REAL	—	m^2/s	*visc*	VEDDY2
	Vertical diffusion coefficient $\nu_T/\sigma_k + \nu_b$ in the k- and kl-equations (3.1.114) and (3.1.125).					
VEDDYV	NZ+1,NR,NC	REAL	ν_T	m^2/s	*visc*	VEDDY1,VEDDY2
	Vertical diffusion coefficient for momentum.					
ZLW	NZ+1,NR,NC	REAL	l	m	*turbke*	TKLENG,TLENG,TLENDIS
	Turbulence mixing length l.					

Table 5.2.9: Parameters for the biology.

Name	Type	Symbol	Value	Unit	Include	Defined
AB	REAL	b_a	0.5	—	*biopar*	DEFCON
	Slope of the respiration-growth relationship for autotrophs (see equation (3.2.31)).					
AKIN	REAL	k_{in}	0.5	mmol N m^{-3}	*biopar*	DEFCON
	Parameter representing the inhibition of nitrate uptake due to ammonium (see equation (3.2.46)).					
AKNHS	REAL	k_{NHS}	0.24	mmol N m^{-3}	*biopar*	DEFCON
	Half-saturation constant for ammonium uptake (see equation (3.2.51)).					
AKNOS	REAL	k_{NOS}	0.32	mmol N m^{-3}	*biopar*	DEFCON
	Half-saturation constant for nitrate uptake (see equation (3.2.45)).					
ALPHAL	REAL	α	0.23	mmol C (mg Chl)$^{-1}$day^{-1}(W/m^2)$^{-1}$	*biopar*	INIBIO
	Photosynthetic efficiency as used in the expression (3.2.28) for the light-controlled growth rate.					
AMUMAX	REAL	μ_{max_a}	3.0	day^{-1}	*biopar*	DEFCON
	Maximum value for the nutrient controlled growth rate of autotrophs at 20^{0}C (see equations (3.2.39)–(3.2.40)).					
ANHUMAX	REAL	${}^{NH}u_{max_a}$	1.5	mmol N (mmol C)$^{-1}$ day^{-1}	*biopar*	DEFCON
	Maximum value for the ammonium uptake rate of autotrophs at 20^{0}C (see equations (3.2.49)–(3.2.50)).					
ANOUMAX	REAL	${}^{NO}u_{max_a}$	0.5	mmol N (mmol C)$^{-1}$ day^{-1}	*biopar*	DEFCON
	Maximum value for the nitrate uptake rate of autotrophs at 20^{0}C (see equations (3.2.43)–(3.2.44)).					
AQMAX	REAL	Q_{max_a}	0.2	mmol N (mmol C)$^{-1}$	*biopar*	DEFCON
	Maximum value for the nitrogen to carbon quota (autotrophs).					
AQMIN	REAL	Q_{min_a}	0.05	mmol N (mmol C)$^{-1}$	*biopar*	DEFCON
	Minimum value for the nitrogen to carbon quota (autotrophs).					
AR0	REAL	r_{0_a}	0.05	day^{-1}	*biopar*	DEFCON
	Basal respiration rate for autotrophs (see equation (3.2.30)).					
ASAT	REAL	a_{sT}	2.74×10^{-3}	m^3 (mmol O)$^{-1}$	*biopar*	DEFCON
	Parameter used to determine the temperature dependence in the formulation (3.2.80)–(3.2.82)) for the surface oxygen flux.					
AXQN	REAL	${}^{X}q_a^N$	2.0	mg Chl (mmol N)$^{-1}$	*biopar*	DEFCON
	Chlorophyll to nitrogen ratio for autrotrophs (see equation (3.2.29)).					
BNHS	REAL	${}^{NH}S_b^0$	1.0	mmol N m^{-3}	*biopar*	DEFCON
	Assumed concentration of ammonium in the consolidated sediment used to calculate the benthic flux for ammonium (see equation (3.2.79)).					

Table 5.2.9: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
BNOS	REAL	${}^{NO}S_b^0$	5.0	mmol N m^{-3}	*biopar*	DEFCON
	Assumed concentration of nitrate in the consolidated sediment used to calculate the benthic flux for nitrate (see equation (3.2.78)).					
BSAT	REAL	b_{sT}	7.8×10^{-5}	m^3 (mmol O)$^{-1}$ ^{0}C^{-1}	*biopar*	DEFCON
	Parameter used to determine the temperature dependence in the formulation (3.2.80)–(3.2.82)) for the surface oxygen flux.					
BTRE	REAL	—	0.42	mmol C m^{-3}	*biopar*	INIBIO
	Threshold value for microplankton carbon below which the grazing pressure is set to zero in all biological equations.					
COKS	REAL	$O_{1/2}$	10.0	mmol O m^{-3}	*biopar*	DEFCON
	Half-saturation constant in the oxygen dependence of detrital carbon remineralisation (see equation (3.2.65)).					
CRMAX	REAL	${}^{C}r_{max}$	0.06	day^{-1}	*biopar*	DEFCON
	Maximum detrital carbon remineralisation rate at 20^{0}C (see equation (3.2.62)).					
CUO	REAL	$O_{1/2,nit}$	30.0	mmol O m^{-3}	*biopar*	DEFCON
	Half-saturation constant in the oxygen dependence of nitrification at 20^{0}C (see equation (3.2.69)).					
CWIND	REAL	k_w	5.0×10^{-7}	s/m	*biopar*	DEFCON
	Parameter used to determined the wind dependence in the formulation (3.2.80)–(3.2.82) for the surface oxygen flux.					
EJK2	REAL	—	4.15	μE/J	*biopar*	INIBIO
	Converts J into μE (needed to convert $\epsilon^X\Phi$ into typical units of α).					
EPSBIO	REAL	ϵ^X	0.016	m^2 (mg Chl)$^{-1}$	*biopar*	DEFCON
	Diffuse PAR attenuation cross-section for chlorophyll.					
EPSDET	REAL	ϵ^C	0.002	m^2 (mmol C)$^{-1}$	*biopar*	DEFCON
	Diffuse PAR attenuation cross-section for detrital carbon.					
EPSSED	REAL	ϵ^A	0.1	m^2/g	*biopar*	DEFCON
	Diffuse PAR attenuation cross-section for inorganic suspended matter.					
ETA	REAL	η	0.3	—	*biopar*	DEFCON
	Ratio of heterotroph to microplankton biomass (see equation (3.2.8)).					
EX	REAL	e	0.5	—	*biopar*	DEFCON
	Portion of the assimilated nitrogen excreted as ammonium.					
FRATE	REAL	F_{br}	0.02	day^{-1}	*biopar*	DEFCON
	Transfer rate in the bottom flux formulations (3.2.78)–(3.2.79) for nitrate and ammonium.					
GAMMA	REAL	γ	0.8	—	*biopar*	DEFCON
	Fraction of the grazed microplankton carbon and nitrogen assimilated by zooplankton.					

Table 5.2.9: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
GRAZUN	REAL	G	0.05	day^{-1}	*biopar*	DEFCON
	Uniform default grazing pressure used in the program if no grazing pressure is specified by the user.					
GRMAX	REAL	μ_{max}	2.33	day^{-1}	*biopar*	INIBIO
	Maximum value for the nutrient controlled growth rate at 20^0C as determined from (3.2.40).					
HB	REAL	b_h	1.5	—	*biopar*	DEFCON
	Slope of the respiration-growth relationship for heterotrophs (see equation (3.2.27)).					
HQ	REAL	q_h	0.18	mmol N (mmol C)$^{-1}$	*biopar*	DEFCON
	Nitrogen to carbon quota for heterotrophs.					
HR0	REAL	r_{0_h}	0.02	day^{-1}	*biopar*	DEFCON
	Basal respiration rate for heterotrophs (see equation (3.2.30)).					
OMIN	REAL	—	0.1	mmol O m^{-3}	*biopar*	INIBIO
	Minimum value for oxygen below which the nitrification rate (as given by the factor $f(O)$ in (3.2.62)) and all sink terms in the oxygen equation (see equation (3.2.68)) are set to zero.					
PQHI	REAL	Φ	40.0	nmol C μE^{-1}	*biopar*	DEFCON
	Photosynthetic quantum yield used for the calculation of the photosynthetic efficiency α.					
QMAX	REAL	Q_{max}	0.19	mmol N (mmol C)$^{-1}$	*biopar*	INIBIO
	Maximum value for the nitrogen to carbon quota given by (3.2.48) and used in the formulations for nutrient uptake and microplankton sinking rates.					
QMAXMOD	REAL	—	0.14	mmol N (mmol C)$^{-1}$	*biopar*	INIBIO
	The factor $Q_{max} - q_h\eta$ in the nutrient uptake expressions (3.2.47) and (3.2.52).					
QMCMIN	REAL	${}^{M}q^{C}_{min}$	0.09	mmol N (mmol C)$^{-1}$	*biopar*	DEFCON
	Minimum value for the detrital nitrogen quota as used in the expressions (3.2.62)–(3.2.63) for detrital remineralisation.					
QMIN	REAL	Q_{min}	0.09	mmol N (mmol C)$^{-1}$	*biopar*	INIBIO
	Minimum value for the nitrogen to carbon quota as given by (3.2.38) and used in the formulations for nutrient-controlled growth and microplankton sinking rates.					
QOB	REAL	${}^{O}q^{B}$,${}^{O}q^{C}$	1.0	mmol O (mmol C)$^{-1}$	*biopar*	DEFCON
	Photosynthetic and respiratory quotient for microplankton carbon growth and detrital respiration.					
QON	REAL	${}^{O}q^{NO}$,${}^{O}q^{NH}$	2.0	mmol O (mmol N)$^{-1}$	*biopar*	DEFCON
	Photosynthetic and respiratory quotient for nitrate uptake and nitrification.					

Table 5.2.9: continued.

Name	Type	Symbol	Value	Unit	Include	Defined
Q10	REAL	q_T	0.07	$(^0\text{C})^{-1}$	*biopar*	DEFCON
	Growth rate coefficient in the temperature growth factor (3.2.21).					
RB0	REAL	r_0	0.04	day^{-1}	*biopar*	INIBIO
	Basal respiration rate as given by equation (3.2.30).					
REFTEMP	REAL	T_r	20.0	^{0}C	*biopar*	DEFCON
	Reference temperature in the temperature growth factor (3.2.21).					
RMRMAX	REAL	${}^{M}r_{max}$	0.08	day^{-1}	*biopar*	DEFCON
	Maximum rate of detrital nitrogen remineralisation at 20^{0}C.					
RMU	REAL	b	1.18	—	*biopar*	INIBIO
	Respiration slope determined from (3.2.31).					
RNHMAX	REAL	${}^{NH}r_{max}$	0.1	day^{-1}	*biopar*	DEFCON
	Maximum rate of nitrification at 20^{0}C.					
UCK1	REAL	—	0.0864	(mmol s)(nmol day)$^{-1}$	*biopar*	INIBIO
	Converts (nmol day) into (mmol s) (needed to convert $\epsilon^X\Phi$ into typical units of α).					
UMAXNH	REAL	${}^{NH}u_{max}$	1.05	mmol N (mmol C)$^{-1}$ day^{-1}	*biopar*	INIBIO
	Maximum value for the ammonium uptake rate at 20^{0}C given by (3.2.50).					
UMAXNO	REAL	${}^{NO}u_{max}$	0.35	mmol N (mmol C)$^{-1}$ day^{-1}	*biopar*	INIBIO
	Maximum value for the nitrate uptake rate at 20^{0}C given by (3.2.44).					
WSDET	REAL	$w_{s_{det}}$	-5.0	m/day	*biopar*	DEFCON
	Uniform sinking rate for detritus.					
WSMAX	REAL	$w_{s_{max}}$	-5.0	m/day	*biopar*	DEFCON
	Maximum sinking rate for microplankton (see equation (3.2.71)).					
WSMIN	REAL	$w_{s_{min}}$	-0.5	m/day	*biopar*	DEFCON
	Minimum sinking rate for microplankton (see equation (3.2.71)).					
XQN	REAL	${}^{X}q^{N}$	1.4	mg Chl (mmol N)$^{-1}$	*biopar*	INIBIO
	Chlorophyll to nitrogen ratio given by ${}^{X}q^{N} = {}^{X}q^{N}_a(1-\eta)$.					

Table 5.2.10: Biological arrays.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
ATCFCORD	NZ,NR,NC	REAL	k_d	m^{-1}	*optics.inc*	BIOPT
	The optical attenuation coefficient for PAR in the biological module as given by equation (3.2.6).					
GRAZING	NR,NC,12	REAL	G	day^{-1}	*plnktn.inc*	DEFICS
	Monthly averaged values of grazing pressures.					
PAR	NZ,NR,NC	REAL	I_p	$\mathrm{W/m}^2$	*optics.inc*	BIOPT
	Photosynthetically active radiation (PAR).					
P2B	NZ,NR,NC	REAL	B	mmol C m^{-3}	*plnktn.inc*	BIOLGY
	Microplankton carbon concentration.					
P2C	NZ,NR,NC	REAL	C	mmol C m^{-3}	*plnktn.inc*	BIOLGY
	Detrital carbon concentration.					
P2M	NZ,NR,NC	REAL	M	mmol N m^{-3}	*plnktn.inc*	BIOLGY
	Detrital nitrogen concentration.					
P2N	NZ,NR,NC	REAL	N	mmol N m^{-3}	*plnktn.inc*	BIOLGY
	Microplankton nitrogen concentration.					
P2NHS	NZ,NR,NC	REAL	${}^{NH}S$	mmol N m^{-3}	*plnktn.inc*	BIOLGY
	Ammonium concentration.					
P2NOS	NZ,NR,NC	REAL	${}^{NO}S$	mmol N m^{-3}	*plnktn.inc*	BIOLGY
	Nitrate concentration.					
P2O	NZ,NR,NC	REAL	O	mmol O m^{-3}	*plnktn.inc*	BIOLGY
	Oxygen concentration.					
P2X	NZ,NR,NC	REAL	X	mg Chl m^{-3}	*plnktn.inc*	BIOLGY
	Chlorophyll concentration.					
P2ZN	NZ,NR,NC	REAL	Z_N	mmol N m^{-3}	*plnktn.inc*	BIOLGY
	Zooplankton nitrogen concentration.					

Table 5.2.11: Parameters and arrays for the sediment and fluff module.

Name	Dimensions	Type	Symbol	Value	Unit	Include	Defined
ALPHAS	—	REAL	α_s	0.0025	g m^{-2} s^{-1}	*sedpar*	DEFCON
	Parameter used in the formulation (3.2.72) for the resuspension rate.						
FLUFLOS	NR,NC,5	REAL	—	—	amount m^{-2}	*sedmnt*	FLUFF
	Amount of material lost by the fluff layer to the consolidated sediment since the initial time. The type is given by the value of the third array index: 2 : microplankton carbon (mmol C m^{-2}) 3 : detrital carbon (mmol C m^{-2}) 4 : microplankton nitrogen (mmol N m^{-2}) 5 : detrital nitrogen (mmol N m^{-2})						
FSED	NC,NR,5	REAL	f_l	—	amount m^{-2}	*sedmnt*	FLUFF
	Amount of material in the fluff layer. The type is given by the value of the third array index: 1 : inorganic sediment (g m^{-2}) 2 : microplankton carbon (mmol C m^{-2}) 3 : detrital carbon (mmol C m^{-2}) 4 : microplankton nitrogen (mmol N m^{-2}) 5 : detrital nitrogen (mmol N m^{-2})						
RNSED	—	REAL	n_s	3.0	—	*sedpar*	DEFCON
	Exponential factor in the formulation (3.2.72) for the resuspension rate.						
SEDC1	NZ,NR,NC	REAL	A	—	g/m^3	*sedmnt*	SEDEUL
	Concentration of inorganic suspended matter.						
WSED1	—	REAL	w_s^A	-0.002	m/s	*sedpar*	DEFCON
	Uniform sinking rate for inorganic suspended matter.						

Table 5.2.12: Parameters and arrays for the contaminant module.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
CONCN	NZ,NR,NC,0:NCONC	REAL	C_i	g/m^3	*concn*	MASS
	Contaminant concentrations.					
NCONC	—	REAL	—	—	*param*	*param.inc*
	Number of contaminant concentrations.					

Table 5.2.13: Parameters for the particle module.

Name	Type	Symbol	Value	Unit	Include	Defined
HEDPUN	REAL	λ_H^C	0.0	m^2/s	*sedpar*	DEFCON
	Uniform horizontal diffusion coefficient.					
MAXNOP	INTEGER	—	—	—	*param*	*param.inc*
	Maximum allowed number of particles during the simulation.					
NUMP	INTEGER	—	—	—	*sedmnt*	STAUCH
	Number of particles inside the computational domain.					
STLSOL	REAL	m_{riv}	100.0	ton	*sedpar*	DEFCON
	Masses of the particles entering at river boundaries.					
STSSOL	REAL	m_{sea}	100.0	ton	*sedpar*	DEFCON
	Masses of the particles entering at open sea boundaries.					

Table 5.2.14: Arrays for the particle module.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
HEDDYP	NR,NC	REAL	λ_H^C	m^2/s	*visc*	INITC
	Horizontal diffusion coefficient (taken be be uniform).					
IT	0:MAXNOP	INTEGER	i_n	—	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	The i-indices of the cell centres where the particles are located.					
JT	0:MAXNOP	INTEGER	j_n	—	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	The j-indices of the cell centres where the particles are located.					
KT	0:MAXNOP	INTEGER	k_n	—	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	The k-indices of the cell centres where the particles are located.					
LST	0:MAXNOP	INTEGER	—	—	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	Particle labels.					
NUMPIN	0:MAXNOP	INTEGER	—	—	*sedmnt*	STAUCH
	Indices of the particles currently inside the computational domain.					
SPMCON	NZ,NR,NC	REAL	ρ_{spm}	g/m^3	*sedmnt*	CONSPM
	Concentration of dissolved particulate matter.					
ST	0:MAXNOP	REAL	—	ton	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	Particle masses.					
STIN	NZ,NR,NC	REAL	—	ton	*sedmnt*	MCOSB,MCROB
	Total mass of the particles who entered the computational domain through open sea and river boundaries.					
STOUT	NZ,NR,NC	REAL	—	ton	*sedmnt*	MC3D
	Total mass of all particles who left the computational domain through land, open sea and river boundaries.					
UTURB	NZ,NR,NC	REAL	u'_{max}	m/s	*sedmnt*	SEDLAG
	Maximum turbulent fluctuation velocity in the X-direction given by (3.3.5).					
VTURB	NZ,NR,NC	REAL	v'_{max}	m/s	*sedmnt*	SEDLAG
	Maximum turbulent fluctuation velocity in the Y-direction given by (3.3.5).					
WTURB	NZ,NR,NC	REAL	w'_{max}	m/s	*sedmnt*	SEDLAG
	Maximum turbulent fluctuation velocity in the vertical direction given by (3.3.5).					
XT	0:MAXNOP	REAL	x'_{1n}	m,degrees	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	X-coordinates of the particles relative to the centre of the cell where they are currently located.					
YT	0:MAXNOP	REAL	x'_{2n}	m,degrees	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	Y-coordinates of the particles relative to the centre of the cell where they are currently located.					
ZT	0:MAXNOP	REAL	σ'_n	—	*sedmnt*	DEFICS,MC3D,MCOSB,MCROB
	σ-coordinates of the particles relative to the centre of the cell where they are currently located.					

Table 5.2.15: Parameters for the boundary conditions.

Name	Type	Symbol	Value	Unit	Include	Defined
CDLIN	REAL	k_{lin}	0.0	m/s	*stress*	DEFCON
	Linear bottom friction coefficient (see equation (3.1.186)).					
CDZ0UN	REAL	z_0	0.0	m	*stress*	DEFCON
	Uniform bottom roughness length (quadratic bottom friction law) which can be defined by the user in a 3-D application if z_0 is assumed to be horizontally homogeneous and which must be defined for a 1-D application (if IBSTR $= 1$).					
FSUN	REAL	τ_{s1}/ρ_0	0.0	$\mathrm{m}^2/\mathrm{s}^2$	*phycon*	DEFCON
	Uniform surface stress component in the X-direction (normalised by ρ_0) used when IOPTM $= 1$.					
GSUN	REAL	τ_{s2}/ρ_0	0.0	$\mathrm{m}^2/\mathrm{s}^2$	*phycon*	DEFCON
	Uniform surface stress component in the Y-direction (normalised by ρ_0) used when IOPTM $= 1$.					
HSUN	REAL	h_s	0.0	m	*phycon*	DEFCON
	Uniform significant wave height used in the wave-current interaction module when IOPTW $= 1$.					
NCON	INTEGER	N_T	—	—	*param*	*param.inc*
	Number of tidal constituents.					
NVPROF	INTEGER	—	—	—	*param*	*param.inc*
	Maximum number of “distinct” vertical vertical arrays used in the open boundary conditions for scalars (see Section IV-2.5).					
TWUN	REAL	T_w	0.0	s	*phycon*	DEFCON
	Uniform wave period used in the wave-current interaction module when IOPTW $= 1$.					

Table 5.2.16: Arrays for the boundary conditions.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
AMPOBU	0:NOBU,0:NCON	REAL	A_n	m	*bounds*	DEFOB2D, WROBDAT
	Residual and amplitudes in the harmonic expansion (3.1.198) used to determine the depth-integrated current $\overline{U}$ at U-open boundaries.					
AMPOBV	0:NOBV,0:NCON	REAL	A_n	m	*bounds*	DEFOB2D, WROBDAT
	Residual and amplitudes in the harmonic expansion (3.1.198) used to determine the depth-integrated current $\overline{V}$ at V-open boundaries.					
BSTOT	NR,NC	REAL	τ_b/ρ_0	$\mathrm{m}^2/\mathrm{s}^2$	*stress*	BSTRES
	Magnitude of the bottom stress (normalised by ρ_0).					
CDB	NR,NC	REAL	C_D^b	—	*stress*	BSTRES, WAVCUR
	Quadratic bottom friction coefficient at the bottom grid cell, given by (3.1.187) if IOPTW $= 0$ or by (3.1.281) if wave-current interaction is included.					
CDB100	NR,NC	REAL	C_{100}	—	*stress*	BSTRES, WAVCUR
	Quadratic bottom friction coefficient at 1 m height above the sea bed, given by (3.2.74).					
CDZ0	NR,NC	REAL	z_0	m	*stress*	DEFGRID, PREPROC
	Bottom roughness length used to evaluate the quadratic bottom friction coefficient.					
CLOUD2	NR,NC	REAL	f_c	—	*met*	WRFCDAT
	Fractional cloud cover between 0 and 1.					
CONVP	0:NZ,0:NVPROF,0:NCONC	REAL	—	$\mathrm{g/m}^3$	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for the contaminants C_i (Eulerian contaminant module).					
EVAPR	NR,NC	REAL	—	$\mathrm{kg\ m}^{-2}\mathrm{s}^{-1}$	*stress*	WRFCDAT
	Evaporation minus precipitation rate $E_{vap} - E_{pr}$.					
FB	NR,NC+1	REAL	$-\tau_{b1}/\rho_0$	$\mathrm{m}^2/\mathrm{s}^2$	*stress*	BSTRES
	(Minus) the bottom stress component in the X-direction (normalised by ρ_0).					
FBK	NR,NC+1	REAL	k_b^u	m/s	*stress*	BSTRES
	Value of the coefficient $C_d^b(u_b^2 + v_b^2)^{1/2}$ at the U-nodes as used in the discretised $\overline{U}$-equation (3.4.13) and in the bottom boundary condition for the u-current (see equation (3.4.185)).					
FS	NR,NC	REAL	τ_{s1}/ρ_0	$\mathrm{m}^2/\mathrm{s}^2$	*stress*	SURFLX
	Surface stress component in the X-direction (normalised by ρ_0).					

Table 5.2.16: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
GB	NR+1,NC	REAL	$-\tau_{b2}/\rho_0$	m^2/s^2	*stress*	BSTRES
	(Minus) the bottom stress component in the Y-direction (normalised by ρ_0).					
GBK	NR+1,NC	REAL	k_b^v	m/s	*stress*	BSTRES
	Value of the coefficient $C_d^b(u_b^2+v_b^2)^{1/2}$ at the V-nodes as used in the discretised $\overline{V}$-equation (3.4.14) and in the bottom boundary condition for the v-current (see equation (3.4.185)).					
GS	NR,NC	REAL	τ_{s2}/ρ_0	m^2/s^2	*stress*	SURFLX
	Surface stress component in the Y-direction (normalised by ρ_0).					
HS	NR,NC	REAL	h_s	m	*waves*	WRFCDAT
	Significant height of the surface waves.					
HUM2	NR,NC	REAL	RH	—	*met*	WRFCDAT
	Relative humidity (between 0 and 1).					
ITYPOBU	0:NOBU	INTEGER	—	—	*bounds*	DEFOB2D
	Type of open boundary condition for the depth-integrated current $\overline{U}$ at U-boundaries (see Section III-1.6.3a and III-4.3.11b for details). 0 : Incoming Riemann variable is set to zero. 1 : Incoming Riemann variable is determined using input data for $\overline{U}$ and ζ. 2 : Incoming Riemann variable is determined using a zero gradient condition. 3 : Incoming Riemann variable is determined using input data for ζ. 4 : Incoming Riemann variable is determined using input data for $\overline{U}$.					
ITYPOBV	0:NOBV	INTEGER	—	—	*bounds*	DEFOB2D
	Type of open boundary condition for the depth-integrated current $\overline{V}$ at V-boundaries (see Section III-1.6.3a and III-4.3.11b for details). 0 : Incoming Riemann variable is set to zero. 1 : Incoming Riemann variable is determined using input data for $\overline{V}$ and ζ. 2 : Incoming Riemann variable is determined using a zero gradient condition. 3 : Incoming Riemann variable is determined using input data for ζ. 4 : Incoming Riemann variable is determined using input data for $\overline{V}$.					
IVPOBU	0:NOBU	INTEGER	—	—	*bounds*	DEFOB3D
	Number of the profile used for the (3-D) open boundary conditions at U-nodes (see Section IV-2.5 for details).					
IVPOBV	0:NOBV	INTEGER	—	—	*bounds*	DEFOB3D
	Number of the profile used for the (3-D) open boundary conditions at V-nodes (see Section IV-2.5 for details).					
LSTOBU	0:NOBU	INTEGER	—	—	*bounds*	DEFOB3D
	Labels of the particles in the Lagrangian module entering at U-open boundaries.					
LSTOBV	0:NOBV	INTEGER	—	—	*bounds*	DEFOB3D
	Labels of the particles in the Lagrangian module entering at V-open boundaries.					
PHAOBU	0:NOBU,0:NCON	REAL	φ_n	rad	*bounds*	DEFOB2D, WROBDAT
	Phases φ_n in the harmonic expansion (3.1.198) used to determine the depth-integrated current $\overline{U}$ at U-open boundaries.					

Table 5.2.16: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
PHAOBV	0:NOBV,0:NCON	REAL	φ_n	rad	*bounds*	DEFOB2D, WROBDAT
	Phases φ_n in the harmonic expansion (3.1.198) used to determine the depth-integrated current $\overline{V}$ at V-open boundaries.					
PHASE0	0:NCON	REAL	φ_{n0}	rad	*tide*	DEFOB2D, DEFICS, BOUNDC
	The initial phases in the harmonic expansion (3.1.198) used for the open boundary conditions for the 2-D mode.					
P2	NR,NC	REAL	P_a	N/m^2	*met*	WROBDAT
	Atmospheric pressure.					
P2BVP	0:NZ,0:NVPROF	REAL	—	mmol C m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for microplankton carbon B.					
P2CVP	0:NZ,0:NVPROF	REAL	—	mmol C m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for detrital carbon C.					
P2MVP	0:NZ,0:NVPROF	REAL	—	mmol N m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for detrital nitrogen M.					
P2NHSVP	0:NZ,0:NVPROF	REAL	—	mmol N m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for ammonium ^{NH}S.					
P2NOSVP	0:NZ,0:NVPROF	REAL	—	mmol N m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for nitrate ^{NO}S.					
P2NVP	0:NZ,0:NVPROF	REAL	—	mmol N m^{-3}	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for microplankton nitrogen N.					
P2OVP	0:NZ,0:NVPROF	REAL	—	mmol O m^{-3}	*bounds*	INITC
	Vertical profile arrays specifying the open boundary conditions for oxygen O.					
P2ZNVP	0:NZ,0:NVPROF	REAL	—	mmol N m^{-3}	*bounds*	INITC
	Vertical profile arrays specifying the open boundary conditions for zooplankton nitrogen Z_N.					
QNSOL	NR,NC	REAL	Q_{nsol}	W/m^2	*stress*	WRFCDAT, SURFLX
	Non-solar heat flux, given by (3.1.171), at the sea surface.					
QSOL	NR,NC	REAL	Q_{rad}	W/m^2	*stress*	WRFCDAT, SOLRAD
	Solar heat flux at the sea surface.					
QSTOBU	0:NOBU	REAL	—	—	*bounds*	DEFOB3D, WROBDAT
	External concentration ρ^e_{spm} (in ton/m^3) or river discharge (in ton/s) of suspended matter at open sea or river boundaries used in the open boundary conditions at U-nodes within the Lagrangian particle module.					
QSTOBV	0:NOBV	REAL	—	—	*bounds*	DEFOB3D, WROBDAT
	External concentration ρ^e_{spm} (in ton/m^3) or river discharge (in ton/s) of suspended matter at open sea or river boundaries used in the open boundary conditions at V-nodes within the Lagrangian particle module.					

Table 5.2.16: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
RAIN2	NR,NC	REAL	R_{pr}	kg m^{-2} s^{-1}	*met*	WRFCDAT
	Precipitation rate.					
RMASSU	0:NOBU,NZ	REAL	M_{rest}	ton	*bounds*	MCOSB, MCROB
	Remaining mass fraction at U-open boundaries used in the formulation of the open boundary conditions within the Lagrangian particle module (see Sections III-4.8.5c–d for details).					
RMASSV	0:NOBV,NZ	REAL	M_{rest}	ton	*bounds*	MCOSB, MCROB
	Remaining mass fraction at V-open boundaries used in the formulation of the open boundary conditions within the Lagrangian particle module (see Sections III-4.8.5c–d for details).					
R1OBU	0:NOBU	REAL	$R^u_\pm$	m^2/s	*bounds*	BOUNDC
	Incoming Riemann variable R^u_+ at western or R^u_- at eastern U-open boundaries (see Section III-1.6.3a for details).					
R1OBV	0:NOBV	REAL	$R^v_\pm$	m^2/s	*bounds*	BOUNDC
	Incoming Riemann variable R^v_+ at southern or R^v_- at northern V-open boundaries (see Section III-1.6.3a for details).					
R2OBU	0:NOBU	REAL	$R^u_\pm$	m^2/s	*bounds*	BOUNDC
	Outgoing Riemann variable R^u_- at western, or R^u_+ at eastern U-open boundaries (see Section III-1.6.3a for details).					
R2OBV	0:NOBV	REAL	$R^v_\pm$	m^2/s	*bounds*	BOUNDC
	Outgoing Riemann variable R^v_- at southern or R^v_+ at northern V-open boundaries (see Section III-1.6.3a for details).					
SAT2	NR,NC	REAL	T_a	^{0}C	*met*	WRFCDAT
	Air temperature.					
SEDC1VP	0:NZ,0:NVPROF	REAL	—	g/m^3	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for inorganic suspended matter A.					
SIGMA	0:NCON	REAL	ω_n	rad/s	*tide*	DEFOB2D
	Frequencies of the tidal forcing used in the harmonic expansion (3.1.198).					
SSALFL	NR,NC	REAL	—	—	*met*	SURFLX
	Surface salinity flux in PSU m/s.					
SSTOT	NR,NC	REAL	τ_s/ρ_0	m^2/s^2	*stress*	SURFLX
	Magnitude of the surface stress (normalised by ρ_0).					
SST2	NR,NC	REAL	T_s	^{0}C	*met*	METIN
	Sea surface temperature.					
SVP	0:NZ,0:NVPROF	REAL	—	PSU	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for salinity S.					

Table 5.2.16: continued.

Name	Dimensions	Type	Symbol	Unit	Include	Defined
TURVP	0:NZ,0:NVPROF	REAL	—	—	*bounds*	INITC
	Vertical profile arrays specifying the open boundary conditions for turbulence variables.					
TVP	0:NZ,0:NVPROF	REAL	—	^{0}C	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for temperature T.					
TW	NR,NC	REAL	T_w	m	*waves*	WRFCDAT
	Period of the surface waves.					
UVP	0:NZ,0:NVPROF	REAL	—	m/s	*bounds*	DEFOB3D, WROBDAT
	Vertical profile arrays specifying the open boundary conditions for the horizontal currents u, v.					
WINDU2	NR,NC	REAL	U_{10}	m/s	*met*	WRFCDAT
	X-component of the wind vector at 10 m above the sea surface.					
WINDV2	NR,NC	REAL	V_{10}	m/s	*met*	WRFCDAT
	Y-component of the wind vector at 10 m above the sea surface.					
WNUM	NR,NC	REAL	k_w	m^{-1}	*waves*	WAVNUM
	Wave number of the surface waves as obtained from the dispersion relation (3.1.274).					

Table 5.2.17: Parameters for input/output.

Name	Type	Value	Include	Defined
ANALFORM	CHARACTER*1	—	*out*	*defmod.par*
	Format of the output files for harmonic analysis (‘A’ or ‘U’)			
BCSFORM	CHARACTER*1	‘A’	*out*	DEFCON
	Format of the open boundary data files (‘A’ or ‘U’)			
GRDFORM	CHARACTER*1	‘A’	*out*	DEFCON
	Format of the *grd*-file with the grid parameters and arrays (‘A’ or ‘U’)			
HEADERS	LOGICAL	.FALSE.	*out*	DEFCON
	Flag to determine whether files in ASCII (‘A’) format are written by the program with extra header information.			
ICRFORM	CHARACTER*1	‘A’	*out*	DEFCON
	Format of the files containing the initial or final conditions (‘A’ or ‘U’)			
IOBBC	INTEGER	17	*out*	*out.inc*
	File unit number for the *bbc*-file (open boundary data for the biological and sediment modules).			
IOBIN	INTEGER	8	*out*	*out.inc*
	File unit number for the *bin*-file (initial conditions for the biological and sediment modules).			
IOBOU	INTEGER	12	*out*	*out.inc*
	File unit number for the *bou*-file (final conditions for the biological and sediment modules.)			
IOCBC	INTEGER	19	*out*	*out.inc*
	File unit number for the *cbc*-file (open boundary data for the Eulerian contaminant module).			
IOCIN	INTEGER	10	*out*	*out.inc*
	File unit number for the *cin*-file (initial conditions for the Eulerian contaminant module).			
IOCON	INTEGER	5	*out*	*out.inc*
	File unit number for the *con*-file (switches and model parameters).			
IOCOU	INTEGER	14	*out*	*out.inc*
	File unit number for the *cou*-file (final conditions for the Eulerian contaminant module.)			
IOGRD	INTEGER	1	*out*	*out.inc*
	File unit number for the *grd*-file (grid parameters and arrays).			
IOHBC	INTEGER	16	*out*	*out.inc*
	File unit number for the *hbc*-file (open boundary data for the 3-D physics).			
IOHIN	INTEGER	7	*out*	*out.inc*
	File unit number for the *hin*-file (initial conditions for the physics).			

Table 5.2.17: continued.

Name	Type	Value	Include	Defined
IOHOU	INTEGER	11	*out*	*out.inc*
	File unit number for the *hou*-file (final conditions for the physics).			
IOMET	INTEGER	2	*out*	*out.inc*
	File unit number for the *met*-file (meteorological forcing data).			
IOOBC	INTEGER	4	*out*	*out.inc*
	File unit number for the *obc*-file (types of open boundary conditions).			
IOPBC	INTEGER	18	*out*	*out.inc*
	File unit number for the *pbc*-file (open boundary data for the Lagrangian particle module).			
IOPIN	INTEGER	9	*out*	*out.inc*
	File unit number for the *pin*-file (initial conditions for the Lagrangian particle module).			
IOPOU	INTEGER	13	*out*	*out.inc*
	File unit number for the *pou*-file (final conditions for the Lagrangian particle module).			
IOWAV	INTEGER	3	*out*	*out.inc*
	File unit number for the *wav*-file (surface wave data).			
IO2BC	INTEGER	15	*out*	*out.inc*
	File unit number for the *2bc*-file (open boundary data for the 2-D physics).			
IUNITP	INTEGER	—	*out*	OUTPAR
	File unit number for the *par* output file (particle output file).			
MAXAVR	INTEGER	—	*param*	*param.inc*
	Maximum allowed number of time-averaged output files.			
MAXFILES	INTEGER	256	*out*	DEFCON
	Maximum number of file connections which can be open simultaneously.			
MAXOUT	INTEGER	—	*param*	*param.inc*
	Maximum allowed number of time series output files.			
METFORM	CHARACTER*1	'A'	*out*	DEFCON
	Format of the meteorological data file ('A' or 'U').			
NANALMAX	INTEGER	—	*param*	*param.inc*
	Maximum allowed number of 2-D and 3-D output fields on which harmonic analysis is performed.			
NANALTOT	INTEGER	$2M+1$	*anal*	ANALYS
	Total number of 3-D time steps within one period of harmonic analysis.			
NARRAVR	INTEGER	—	*out*	*defmod.par*
	Number of time-averaged output files (which must be smaller than MAXAVR).			
NARROUT	INTEGER	—	*out*	*defmod.par*
	Number of time series output files (which must be smaller than MAXOUT).			
NAVRMAX	INTEGER	—	*param*	*param.inc*
	Maximum allowed number of 2-D and 3-D fields for time-averaged output.			
NCONTO	INTEGER	N_h	*param*	*param.inc*
	Number of frequencies used in the harmonic analysis.			

Table 5.2.17: continued.

Name	Type	Value	Include	Defined
NOPENF	INTEGER	—	*out*	CLOSEF, OPENF
	Number of files connected to the program.			
NOPOUT	INTEGER	—	*out*	*defmod.par*
	Number of particle trajectories selected for output.			
NOUTMAX	INTEGER		*param*	*param.inc*
	Maximum allowed number of 2-D and 3-D fields for time series output.			
NTANAL	INTEGER	—	*anal*	ANALYS0, ANALYS1
	Time step counter during one period of harmonic analysis. Its value is set to -(NANALTOT-1)/2 at the start of a new period, is incremented by 1 after each 3-D time step and attains the value of (NANALTOT-1)/2 at the end of the analysed period.			
N2ANAL	INTEGER	—	*out*	MAINPROG
	Number of 2-D fields on which harmonic analysis is performed.			
N2RESM	INTEGER	—	*out*	MAINPROG
	Number of 2-D fields for time-averaged output.			
N2RESO	INTEGER	—	*out*	MAINPROG
	Number of 2-D fields for time series output.			
N2SCALA	INTEGER	—	*out*	*defmod.par*
	Number of 2-D scalar fields on which harmonic analysis is performed.			
N2SCALM	INTEGER	—	*out*	*defmod.par*
	Number of 2-D scalar fields for time-averaged output.			
N2SCALO	INTEGER	—	*out*	*defmod.par*
	Number of 2-D scalar fields for time series output.			
N2VECA	INTEGER	—	*out*	*defmod.par*
	Number of 2-D vector fields on which harmonic analysis is performed.			
N2VECM	INTEGER	—	*out*	*defmod.par*
	Number of 2-D vector fields for time-averaged output.			
N2VECO	INTEGER	—	*out*	*defmod.par*
	Number of 2-D vector fields for time series output.			
N3ANAL	INTEGER	—	*out*	MAINPROG
	Number of 3-D fields on which harmonic analysis is performed.			
N3RESM	INTEGER	—	*out*	MAINPROG
	Number of 3-D fields for time-averaged output.			
N3RESO	INTEGER	—	*out*	MAINPROG
	Number of 3-D fields for time series output.			
N3SCALA	INTEGER	—	*out*	*defmod.par*
	Number of 3-D scalar fields on which harmonic analysis is performed.			
N3SCALM	INTEGER	—	*out*	*defmod.par*
	Number of 3-D scalar fields for time-averaged output.			

Table 5.2.17: continued.

Name	Type	Value	Include	Defined
N3SCALO	INTEGER	—	*out*	*defmod.par*
	Number of 3-D scalar fields for time series output.			
N3VECA	INTEGER	—	*out*	*defmod.par*
	Number of 3-D vector fields on which harmonic analysis is performed.			
N3VECM	INTEGER	—	*out*	*defmod.par*
	Number of 3-D vector fields for time-averaged output.			
N3VECO	INTEGER	—	*out*	*defmod.par*
	Number of 3-D vector fields for time series output.			
OUTFILE	CHARACTER*14	—	*out*	WRCON
	Generic form of all output data files.			
OUTTIT	CHARACTER*6	TITLE(1:6)	*out*	DEFCON
	First six characters of all output data files.			
PARFORM	CHARACTER*1	'A'	*out*	*defmod.par*
	Format of the *par* output file with particle trajectories ('A' or 'U').			
TITLE	CHARACTER*6	—	*runp*	*defmod.par*
	The title of the run (first six characters of all input files for the main program).			
WAVFORM	CHARACTER*1	'A'	*out*	DEFCON
	Format of the surface wave data file ('A' or 'U').			

Table 5.2.18: Arrays for input/output.

Name	Dimensions	Type	Include	Defined
A	NCONTO,NCONTO	REAL	*anal*	ANALYS0
	The matrix X_{mn}, used in the harmonic analysis, as defined by (3.1.296).			
ANALSIG	0:NCONTO	REAL	*out*	DEFOB2D
	Frequencies (rad/s) used in the harmonic analysis (as given by ω_n in the harmonic expansion (3.1.287)).			
ANAL2DA	NR,NC,NCONTO,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1, ANALYS2
	- The sum array R_m for all 2-D fields, as defined by (3.1.298), initialised in ANALYS0 and updated at each 3-D time step in ANALYS1. - The coefficients a_n (2-D fields) in the harmonic expansion (3.1.287) and calculated in ANALYS2.			
ANAL2DB	NR,NC,NCONTO,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1, ANALYS2
	- The sum array S_m for all 2-D fields, as defined by (3.1.299), initialised in ANALYS0 and updated at each 3-D time step in ANALYS1. - The coefficients b_n (2-D fields) in the harmonic expansion (3.1.287) and calculated in ANALYS2.			
ANAL2DDESC	NANALMAX	CHARACTER*30	*anal*	ANAL2DVAR
	Descriptions of the 2-D fields on which harmonic analysis is applied.			
ANAL2DM	NR,NC,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1
	The sum array $(2M+1)^{-1}\sum_{k=-M}^{M} F_k$ (see Section III-1.7.1) for all 2-D fields, initialised in ANALYS0 and updated at each 3-D time step in ANALYS1.			
ANAL2DNAME	NANALMAX	CHARACTER*8	*anal*	ANAL2DVAR
	Names of the 2-D fields on which harmonic analysis is applied.			
ANAL2DUNIT	NANALMAX	CHARACTER*16	*anal*	ANAL2DVAR
	Units of the 2-D fields on which harmonic analysis is applied.			
ANAL3DA	NZ,NR,NC,NCONTO,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1, ANALYS2
	- The sum array R_m for all 3-D fields, as defined by (3.1.298), initialised in ANALYS0 and updated at each 3-D time step in ANALYS1. - The coefficients a_n (3-D fields) in the harmonic expansion (3.1.287) and calculated in ANALYS2.			
ANAL3DB	NZ,NR,NC,NCONTO,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1, ANALYS2
	- The sum array S_m for all 3-D fields, as defined by (3.1.299), initialised in ANALYS0 and updated at each 3-D time step in ANALYS1. - The coefficients b_n (3-D fields) in the harmonic expansion (3.1.287) and calculated in ANALYS2.			
ANAL3DDESC	NANALMAX	CHARACTER*30	*anal*	ANAL3DVAR
	Descriptions of the 3-D fields on which harmonic analysis is applied.			

Table 5.2.18: continued.

Name	Dimensions	Type	Include	Defined
ANAL3DM	NZ,NR,NC,NANALMAX	REAL	*anal*	ANALYS0, ANALYS1
	The sum array $(2M+1)^{-1}\sum_{k=-M}^{M} F_k$ (see Section III-1.7.1) for all 3-D fields, initialised in ANALYS0 and updated at each 3-D time step in ANALYS1.			
ANAL3DNAME	NANALMAX	CHARACTER*8	*anal*	ANAL3DVAR
	Names of the 3-D fields on which harmonic analysis is applied.			
ANAL3DUNIT	NANALMAX	CHARACTER*16	*anal*	ANAL3DVAR
	Units of the 3-D fields on which harmonic analysis is applied.			
ARRFORM	MAXOUT	CHARACTER*1	*out*	*defmod.par*
	Format of the time series output data fields ('A' or 'U').			
AVRFORM	MAXAVR	CHARACTER*1	*out*	*defmod.par*
	Format of the time-averaged output data fields ('A' or 'U').			
AVR2D	NR,NC,NAVRMAX,MAXAVR	REAL	*tavout*	INTEGR0, INTEGR1
	Time-averaged values of all 2-D fields. This is represented by a sum which is initialised in INTEGR0 at the start of a new period and updated at each 3-D time step in INTEGR0.			
AVR2DDESC	NAVRMAX	CHARACTER*30	*tavout*	AVR2DVAR
	Descriptions of the 2-D fields for time-averaged output.			
AVR2DNAME	NAVRMAX	CHARACTER*8	*tavout*	AVR2DVAR
	Names of the 2-D fields for time-averaged output.			
AVR2DUNIT	NAVRMAX	CHARACTER*16	*tavout*	AVR2DVAR
	Units of the 2-D fields for time-averaged output.			
AVR3D	NZ,NR,NC,NAVRMAX,MAXAVR	REAL	*tavout*	INTEGR0, INTEGR1
	Time-averaged values of all 3-D fields. This is represented by a sum which is initialised in INTEGR0 at the start of a new period and updated at each 3-D time step in INTEGR1.			
AVR3DDESC	NAVRMAX	CHARACTER*30	*tavout*	AVR3DVAR
	Descriptions of the 3-D fields for time-averaged output.			
AVR3DNAME	NAVRMAX	CHARACTER*8	*tavout*	AVR3DVAR
	Names of the 3-D fields for time-averaged output.			
AVR3DUNIT	NAVRMAX	CHARACTER*16	*tavout*	AVR3DVAR
	Units of the 3-D fields for time-averaged output.			
B	NCONTO,NCONTO	REAL	*anal*	ANALYS
	The matrix Y_{mn}, used in the harmonic analysis, as defined by (3.1.297).			
CF	NCONTO	REAL	*anal*	ANALYS
	Temporary work space array used in the modules for harmonic analysis.			
INDXA	NCONTO	INTEGER	*anal*	LUDCMP
	Row permutation vector for the X-matrix (3.1.296) as part of the solution algorithm of the linear system (3.1.293) used in the harmonic analysis.			

Table 5.2.18: continued.

Name	Dimensions	Type	Include	Defined
INDXB	NCONTO	INTEGER	*anal*	LUDCMP
	Row permutation vector for the Y-matrix (3.1.297) as part of the solution algorithm of the linear system (3.1.294) used in the harmonic analysis.			
IUNIT2A	4,NCONTO	INTEGER	*anal*	ANALYS
	File units of the 2-D output files for harmonic analysis (*res*-, *amp*-, *pha*- and *ell*-files).			
IUNIT2M	MAXAVR	INTEGER	*tavout*	INTEGR
	File units of the 2-D time-averaged output files (*avr*-files).			
IUNIT2O	MAXOUT	INTEGER	*tsout*	OUTPT
	File units of the 2-D time series output files (*out*-files).			
IUNIT3A	4,NCONTO	INTEGER	*anal*	ANALYS
	File units of the 3-D output files for harmonic analysis (*res*-, *amp*-, *pha*- and *ell*-files).			
IUNIT3M	MAXAVR	INTEGER	*tavout*	INTEGR
	File units of the 3-D time-averaged output files (*avr*-files).			
IUNIT3O	MAXOUT	INTEGER	*tsout*	OUTPT
	File units of the 3-D time series output files (*out*-files).			
LIMANAL	3,4	INTEGER	*out*	*defmod.par*
	Array which determines the grid locations, the time instants, the period T and the central time t_c for harmonic analysis and output. See Section IV-3.1 for details.			
LIMAVR	3,4,MAXAVR	INTEGER	*out*	*defmod.par*
	Array which determines the grid locations, the time instants and the period T for time-averaged output. See Section IV-3.1 for details.			
LIMOUT	3,4,MAXOUT	INTEGER	*out*	*defmod.par*
	Array which determines the grid locations and the time instants for time series output. See Section IV-3.1 for details.			
LIMPOUT	3	INTEGER	*out*	*defmod.par*
	Array which determines the time instants for the output of particle positions. See Section IV-3.1 for details.			
OUT2DDESC	NOUTMAX	CHARACTER*30	*tsout*	OUT2DVAR
	Descriptions of the 2-D fields for time series output.			
OUT2DNAME	NOUTMAX	CHARACTER*8	*tsout*	OUT2DVAR
	Names of the 2-D fields for time series output.			
OUT2DUNIT	NOUTMAX	CHARACTER*16	*tsout*	OUT2DVAR
	Units of the 2-D fields for time series output.			
OUT3DDESC	NOUTMAX	CHARACTER*30	*tsout*	OUT3DVAR
	Descriptions of the 3-D fields for time series output.			
OUT3DNAME	NOUTMAX	CHARACTER*8	*tsout*	OUT3DVAR
	Names of the 3-D fields for time series output.			
OUT3DUNIT	NOUTMAX	CHARACTER*16	*tsout*	OUT3DVAR
	Units of the 3-D fields for time series output.			

References

Aksnes D.L., Ulvestad K.B., Baliño B.M., Berntsen J., Egge J.K. and Svendsen E., 1995. Ecological modelling in coastal waters: towards predictive physical-chemical-biological simulation models. Ophelia, **41**, 5–36.

Alldredge A.L., Gotschalk C., Passow U. and Riebesell U., 1995. Mass aggregation of diatom blooms: Insights from a mesocosm study. Deep Sea Research, **42**, 9–27.

Allen C.M., 1982. Numerical simulation of contaminant dispersion in estuary flows. Proceedings of the Royal Society of London, **A381**, 179–194.

American National Standard Programming Language FORTRAN, X3.9-1978, 1978. New York.

Anderson R.J. and Smith S.D., 1981. Evaporation coefficient for the sea surface from eddy flux measurements. Journal of Geophysical Research, **86**, 449–456.

Anis A. and Moum J.N., 1995. Surface wave-turbulence interactions: Scaling $\varepsilon(z)$ near the sea surface. Journal of Physical Oceanography, **25**, 2025–2045.

Arakawa A. and Suarez M.J., 1983. Vertical differencing of the primitive equations in sigma coordinates. Monthly Weather Review, **111**, 34–45.

Azam F., Fenchel T., Field J.G., Gray J.S., Meyer-Reil L.A. and Thingstad F., 1983. The ecological role of water-column microbes in the sea. Marine Ecology Progress Series, **10**, 257–263.

Backhaus J.O., 1985. A three-dimensional model for the simulation of shelf sea dynamics. Deutsche Hydrographische Zeitschrift, **38**, 165–187.

Baretta-Bekker J.G., Baretta J.W. and Rasmussen E.K., 1995. The microbial food web in the European Regional Seas Ecosystem model. Netherlands Journal of Sea Research, **33**, 363–379.

Bartsch J. and Coombs S.H., 1996. A numerical model of the dispersion of the blue whiting larvae (Micromesistius poutassou) in the eastern North Atlantic. ICES Council Meeting Papers, C.M.1996/S:40, 16 pp.

Baumert H., 1996. On the theory of photosynthesis and growth in phytoplankton. Part I: light limitation and constant temperature. Internationale Revue der gesamten Hydrobiologie, **81**, 109–139.

Baumert H., Burchard H. and Kleine E., 1997. On second-moment closures for marine turbulence: A review. Presented at the 29th Liège Colloquium on Marine Hydrodynamics. Turbulence Revisited, Liège, May 5–9 1997.

Beckers J.-M., 1991. Application of the GHER 3D general circulation model to the western Mediterranean. Journal of Marine Systems, **1**, 315–332.

Beckers J.M., 1992. La Méditerranée Occidentale: de la modélisation mathématique à la simulation numérique. Ph.D. Thesis, Université de Liège, Belgium, 342 pp.

Blackadar A.K., 1962. The vertical distribution of wind and turbulent exchange in a neutral atmosphere. Journal of Geophysical Research, **67**, 3095–3102.

Blanc T.V., 1985. Variation of bulk-derived surface flux, stability, and roughness results due to the use of different transfer coefficient schemes. Journal of Physical Oceanography, **15**, 650–669.

Blumberg A.F. and Herring H.J., 1987. Circulation modelling using orthogonal curvilinear coordinates. In: J.C.J. Nihoul and B.M. Jamart (Editors), Three-dimensional models of marine and estuarine dynamics. Elsevier, Amsterdam, pp. 55–88.

Blumberg A.F. and Mellor G.L., 1987. A description of a three-dimensional coastal ocean circulation model. In : N.S. Heaps (Editor), Three-dimensional Coastal Ocean Models. Coastal and Estuarine Sciences, Vol. 4, American Geophysical Union, Washington D.C., pp. 1–16.

Bougeault P. and André J.-C., 1986. On the stability of the third-order turbulence closure for the modeling of the stratocumulus-topped boundary layer. Journal of the Atmospheric Sciences, **43**, 1574–1581.

Bowden K.F., Fairbairn L.A. and Hughes P., 1959. The distribution of shearing stresses in a tidal current. Geophysical Journal of the Royal Astronomical Society, **2**, 288–305.

Bowers D. G., Kratzer S., Morrison J.R., Smith P.S.D., Tett P., Walne A.W. and Wild-Allen K., 1999. On the calibration and use of in situ ocean colour measurements for monitoring algal blooms. International Journal of Remote Sensing, in press.

Burchard H. and Baumert H., 1995. On the performance of a mixed-layer model based on the $k-\varepsilon$ turbulence closure. Journal of Geophysical Research, **100**, 8523–8540.

Businger J.A., Wyngaard J.C., Izumi Y. and Bradley E.F., 1971. Flux-profile relationships in the atmospheric surface layer. Journal of the Atmospheric Sciences, **28**, 181–189.

Caperon J. and Meyer J., 1972a. Nitrogen-limited growth of marine phytoplankton – I. Changes in population characteristics with steady-state growth rate. Deep Sea Research, **19**, 601–618.

Caperon J. and Meyer J., 1972b. Nitrogen-limited growth of marine phytoplankton – II. Uptake kinetics and their role in nutrient limited growth of phytoplankton. Deep Sea Research, **19**, 619–632.

Carpenter J.H., 1966. New measurements of oxygen solubility in pure and natural water. Limnology and Oceanography, **11**, 264–277.

Chao S.-Y., 1988. River-forced estuarine plumes. Journal of Physical Oceanography, **18**, 72–88.

Chao S.-Y. and Boicourt W.C, 1986. Onset of estuarine plumes. Journal of Physical Oceanography, **16**, 2137–2149.

Chardy P. and Dauvin J.-C., 1992. Carbon flows in a subtidal fine sand community from the western English Channel: a simulation analysis. Marine Ecology Progress Series, **81**, 147–161.

Charnock H., 1955. Wind stress on a water surface. Quarterly Journal Royal Meteorological Society, **81**, 639–640.

Charnock H., Dyer K.R., Huthnance J.M., Liss P.S., Simpson J.H. and Tett B., 1994. Understanding the North Sea System. The Royal Society, Chapman & Hall, 222 pp.

Cloern J.E., Grenz C. and Vidergar-Lucas L., 1995. An empirical model of the phytoplankton chlorophyll:carbon ratio — the conversion factor between productivity and growth rate. Limnology and Oceanography, **40**, 1313–1321.

Colella P. and Woodward P.R., 1984. The piecewise parabolic method (PPM) for gas-dynamical simulations. Journal of Computational Physics, **54**, 174–201.

Craig P.D., 1996. Velocity profiles and surface roughness under breaking waves. Journal of Geophysical Research, **101**, 1265–1277.

Craig P.D. and Banner M.L., 1994. Modeling wave-enhanced turbulence in the ocean surface layer. Journal of Physical Oceanography, **24**, 2546–2559.

Davies A.M., 1990. On extracting tidal current profiles from vertically integrated two-dimensional hydrodynamic models. Journal of Geophysical Research, **95**, 18317–18342.

Davies A.M., 1993. A bottom boundary layer-resolving three-dimensional tidal model : A sensitivity study of eddy viscosity formulation. Journal of Physical Oceanography, **23**, 1437–1453.

Davies A.M. and Jones J.E., 1992. A three-dimensional wind driven circulation model of the Celtic and Irish Seas. Continental Shelf Research, **12**, 159–188.

Davies A.M. and Lawrence J., 1994. Examining the influence of wind and wind wave turbulence on tidal currents, using a three-dimensional hydrodynamic model including wave-current interaction. Journal of Physical Oceanography, **24**, 2441–2460.

Davies A.M. and Lawrence J., 1995. Modeling the effect of wave-current interaction on the three-dimensional wind-driven circulation of the Eastern Irish Sea. Journal of Physical Oceanography, **25**, 29–45.

Davies A.M., Kwong S.C.M. and Flather R.A., 1997. Formulation of a variable-function three-dimensional model, with applications to the M_2 and M_4 tide on the North-West European continental shelf. Continental Shelf Research, **17**, 165–204.

Deleersnijder E.L., 1989. Upwelling and upsloping in three-dimensional marine models. Applied Mathematical Modelling, **13**, 462–467.

Deleersnijder E., 1992. Modélisation hydrodynamique tridimensionnelle de la circulation générale estivale de la région du Détroit de Bering. Ph.D. Thesis, Université Catholique de Louvain, Belgium, 189 pp.

Deleersnijder E., 1993. Numerical mass conservation in a free-surface sigma coordinate marine model with mode splitting. Journal of Marine Systems, **4**, 365–370.

Deleersnijder E. and Beckers J.-M., 1992. On the use of the σ-coordinate system in regions of large bathymetric variations. Journal of Marine Systems, **3**, 381–390.

Deleersnijder E. and Luyten P., 1994. On the practical advantages of the quasi-equilibrium version of the Mellor and Yamada level 2.5 turbulence closure applied to marine modelling. Applied Mathematical Modelling, **18**, 281–287.

Deleersnijder E. and Ruddick K.G., 1992. A generalized vertical coordinate for 3D marine models. Bulletin de la Société Royale des Sciences de Liège, **61**, 489–502.

Deleersnijder E. and Wolanski E., 1990. Du rôle de la dispersion horizontale de quantité de mouvement dans les modelès marins tridimensionnels. In: J.M. Crolet and P. Lesaint (Editors), Publications Mathématiques de Besançon; Proceedings of the "Journées Numériques de Besançon - Courants Océaniques", September 1990, Besançon, France, pp. 39–50.

Deleersnijder E., Norro A. and Wolanski E., 1992. A three-dimensional model of the water circulation around an island in shallow water. Continental Shelf Research, **12**, 891–906.

Deleersnijder E., Beckers J.M., Campin J.M., El Mohajir M., Fichefet T. and Luyten P., 1997. Some mathematical problems associated with the development and use of marine models. In: J.I. Diaz (Editor), The mathematics of models for climatology and environment. Springer Verlag, Heidelberg, 39–86.

Dogniaux R., 1984. Eclairement énergétique solaire direct diffus et global des surfaces orientées et inclinées. Partie I: Algorithmes et méthodologies. Miscellanea Série B No. 59, 46 pp.

Dogniaux R., 1985. Programme général de calcul des éclairements solaires énergétiques et lumineux des surfaces orientées et inclinées. Ciels clairs, couverts et variables. Miscellanea Série C No. 21, 37 pp.

Droop M.R., 1968. Vitamin B_{12} and marine ecology. IV. The kinetics of uptake, growth and inhibition in Monochrysis lutheri. Journal of the Marine Biological Association of the United Kingdom, **48**, 689–733.

Droop M.R., 1983. 25 Years of algal growth kinetics, a personal view. Botanica Marina, **26**, 99–112.

Droop M.R., 1985. Fluorescence and the light-nutrient interaction in Monochrysis. Journal of the Marine Biological Association of the United Kingdom, **65**, 221–237.

Droop M.R., Mickelson M.J., Scott J.M. and Turner M.F., 1982. Light and nutrient status of algal cells. Journal of the Marine Biological Association of the United Kingdom, **62**, 403–434.

Dyke P.P.G. and Davies A.M. (Editors), 1992. Mathematical models of the North Sea and surrounding continental shelf seas. Proceedings of JONSMOD 90, Birkenhead, U.K., 2–5 April 1990. Continental Shelf Research, **12**(1), 211 pp.

Edinger J.E., Brady D.K. and Gyer J.C., 1974. Heat exchange and transport in the environment. Cooling water discharge project report No. 14. Electronic Research Institute publication No. 74-049-00-3. Palo Alto.

Elliott A.J., 1986. Shear diffusion and the spread of oil in the surface layers of the North Sea. Deutsche Hydrographische Zeitschrift, **39**, 113–137.

Eppley R.W., 1972. Temperature and phytoplankton growth in the sea. United States Fisheries and Wildlife Service Bulletin, **70**, 1063–1085.

Fasham M.J.R., Ducklow H.W. and McKelvie S.M., 1990. A nitrogen-based model of plankton dynamics in the oceanic mixed layer. Journal of Marine Research, **48**, 591–639.

Fransz H.G., Mommaerts J.P. and Radach G., 1991. Ecological modelling of the North Sea. Netherlands Journal of Sea Research, **28**, 67–140.

Frey H., 1991. A three-dimensional, baroclinic shelf sea circulation model – 1. The turbulence closure scheme and the one-dimensional test model. Continental Shelf Research, **11**, 365–395.

Galperin B., Kantha L.H., Hassid S. and Rosati A., 1988. A quasi-equilibrium turbulent energy model for geophysical flows. Journal of the Atmospheric Sciences, **45**, 55–62.

Garvine R.W., 1974. Dynamics of small-scale oceanic fronts. Journal of Physical Oceanography, **4**, 557–569.

Gary J.M., 1973. Estimate of truncation error in transformed coordinate, primitive equation atmospheric models. Journal of the Atmospheric Sciences, **30**, 223–233.

Gaspar P., Grégoris Y. and Lefèvre J.-M., 1990. A simple eddy kinetic energy model for simulations of the oceanic vertical mixing : Tests at Station Papa and long-term upper ocean study site. Journal of Geophysical Research, **95**, 16179–16193.

Geernaert G.L., 1990. Bulk parameterizations for the wind stress and heat fluxes. In: G.L. Geernaert and W.J. Plant (Editors), Surface Waves and Fluxes, Vol. 1 – Current theory. Kluwer Academic Publishers, Dordrecht, pp. 91–172.

Geernaert G.L., Katsaros K.B. and Richter K., 1986. Variation of the drag coefficient and its dependence on sea state. Journal of Geophysical Research, **91**, 7667–7679.

Geider R.K., MacIntyre H.L. and Kana T.M., 1997. A dynamic model of phytoplankton growth and acclimation: responses of the balanced growth rate and the chlorophyll*a*:-carbon ratio to light, nutrient-limitation and temperature. Marine Ecology Progress Series, **148**, 187–200.

Gibson J.K., 1982. The DOCTOR system — a documentary oriented programming system. European Centre for Medium Range Weather Forecasts, Technical Memorandum No. 52, 17 pp.

Gill A.E., 1982. Atmosphere-Ocean Dynamics. International Geophysics Series, Vol.30. Academic Press, Orlando, 662 pp.

Glorioso P.D. and Davies A.M., 1995. The influence of eddy viscosity formulation, bottom topography, and wind wave effects upon the circulation of a shallow bay. Journal of Physical Oceanography, **25**, 1243–1264.

Godin G., 1972. The Analysis of Tides. Liverpool Univ. Press, 264 pp.

Gowen R.J., Tett P. and Wood B.J.B., 1983. Changes in the major dihydroporphyrin plankton pigments during the spring bloom of phytoplankton in two Scottish sea-lochs. Journal of the Marine Biological Association of the United Kingdom, **63**, 27–36.

Gradshteyn I.S. and Ryzhik I.M., 1981. Table of integrals, series and products. Academic Press, 1160 pp.

Grant W.D. and Madsen O.S., 1979. Combined wave and current interaction with a rough bottom. Journal of Geophysical Research, **84**, 1797–1808.

Gregg M.C., 1987. Diapycnal mixing in the thermocline : A review. Journal of Geophysical Research, **92**, 5249–5286.

Griffiths R.W. and Linden P.F., 1981. The stability of vortices in a rotating, stratified fluid. Journal of Fluid Mechanics, **105**, 283–316.

Haney R.L., 1991. On the pressure gradient force over steep topography in sigma coordinate ocean models. Journal of Physical Oceanography, **21**, 610–619.

Hankin S. and Denham M., 1996. FERRET User's Guide. Version 4.4. NOAA/PMEL-/TMAP, 235 pp. Available via http://ferret.wrc.noaa.gov/Ferret/ .

Hassid S. and Galperin B., 1983. A turbulent energy model for geophysical flows. Boundary Layer Meteorology, **26**, 397–412.

Heaps N.S., 1972. Estimation of density currents in the Liverpool Bay area of the Irish Sea. Geophysical Journal of the Royal Astronomical Society, **30**, 415–432.

Heathershaw A.D., 1981. Comparisons of measured and predicted sediment transport rates in tidal currents. Marine Geology, **42**, 75–104.

Heathershaw A.D., Small J. and Stretch C.E., 1994. Frictional formulations in numerical ocean models and their effect on simulated acoustic fields. Journal of Physical Oceanography, **24**, 273–297.

Hedstrom G.W., 1979. Nonreflecting boundary conditions for nonlinear hyperbolic systems. Journal of Computational Physics, **30**, 222–237.

Hill A.E., James I.D., Linden P.F., Matthews J.P., Prandle D., Simpson J.H., Gmitrowicz E.M., Smeed D.A., Lwiza K.M.M., Durazo R., Fox A.D. and Bowers D.G., 1994. Dynamics of tidal mixing fronts in the North Sea. In: H. Charnock, K.R. Dyer, J.M. Huthnance, P.S. Liss, J.H. Simpson and P.B. Tett (Editors), Understanding the North Sea System. The Royal Society, Chapman & Hall, pp. 53–68.

Hill P.S., 1992. Reconciling aggregation theory with observed vertical fluxes following phytoplankton blooms. Journal of Geophysical Research, **97**, 2295–2308.

Hirsch C., 1988. Numerical computation of internal and external flows. Volume 1: Fundamentals of numerical discretisation. Wiley, New York, 515 pp.

Hirsch C., 1990. Numerical computation of internal and external flows. Volume 2: Computational methods for inviscid and viscous flows. Wiley, New York, 691 pp.

Holt J.T. and James I.D., 1999. A simulation of the Southern North Sea in comparison with measurements from the North Sea Project. Part 1: Temperature. Continental Shelf Research, **19**, 1087–1112.

Hossain M.S. and Rodi W., 1982. A turbulence model for buoyant flows and its application to vertical buoyant jets. In : W. Rodi (Editor), Turbulent buoyant jets and plumes. HMT-Series, Vol. 6, Pergamon Press, Oxford, pp. 121–178.

Howarth M.J., Dyer K.R., Joint I.R., Hydes D.J., Purdie D.A., Edmunds H., Jones J.E., Lowry R.K., Moffatt T.J., Pomroy, A.J. and Proctor R., 1994. Seasonal cycles and their spatial variability. In: H. Charnock, K.R. Dyer et al. (Editors), Understanding the North Sea System. The Royal Society, Chapman & Hall, London, pp. 5–26.

Hunter J.R., 1987. The application of Lagrangian particle-tracking techniques to modelling of dispersion in the sea. In: J. Noye (Editor), Numerical modelling: application to marine systems. North-Holland, Amsterdam, The Netherlands, 257–269.

Hunter J.R., Craig P.D. and Phillips H.E., 1993. On the use of random-walk models with spatially varying diffusivity. Journal of Computational Physics, **106**, 366–376.

Huthnance J.M. (Editor), 1997. Processes in Regions of Freshwater Influence – PROFILE, MAS2-CT93-0054. Final Report. POL Internal Document No. 102, Proudman Oceanographic Laboratory, 95 pp.

Jackson G. A., 1990. A model of the formation of marine algal flocs by physical coagulation processes. Deep Sea Research, **37**, 1197–1211.

Jago C.F., Bale A.J., Green M.O., Howarth M.J., Jones S.E., McCave I.N., Milward G.E., Morris A.W., Rowden A.A. and Williams J.J., 1993. Resuspension processes and seston dynamics, southern North Sea. Philosophical Transactions of the Royal Society of London, **A 343**, 475–491.

James I.D., 1996. Advection schemes for shelf sea models. Journal of Marine Systems, **8**, 237–254.

Janjić Z.I., 1977. Pressure gradient force and advection scheme used for forecasting with steep and small scale topography. Contrib. Atmospheric Phys., **50**, 186–199.

Janssen P.A.E.M., 1991. Quasi-linear theory of wind-wave generation applied to wave forecasting. Journal of Physical Oceanography, **21**, 1631–1642.

Jensen T.G., 1998. Open boundary conditions in stratified ocean models. Journal of Marine Systems, **16**, 297–322.

Jerlov N.G., 1968. Optical Oceanography. Elsevier, 194 pp.

Jones S.E., Jago C.F. and Simpson J.H., 1995. Modelling suspended sediment dynamics in tidally mixed and periodically stratified waters : progress and pitfalls. In : C.B. Pattiaratchi (Editor), Mixing Processes in Estuaries and Coastal Seas. Coastal and Estuarine Studies, Vol. 41, American Geophysical Union, 315–338.

Kantha L.H. and Clayson C.A., 1994. An improved mixed layer model for geophysical applications. Journal of Geophysical Research, **99**, 25235–25266.

Kato H. and Phillips O.M., 1969. On the penetration of a turbulent layer into stratified fluid. Journal of Fluid Mechanics, **37**, 643–655.

Katsaros K.B., 1990. Parameterization schemes and models for estimating the surface radiation budget. In: G.L. Geernaert and W.J. Plant (Editors), Surface Waves and Fluxes, Vol. 2 – Remote sensing. Kluwer Academic Publishers, Dordrecht, pp. 339–368.

Kirk J.T.O., 1983. Light and photosynthesis in aquatic ecosystems. Cambridge University Press.

Kourafalou V.H., Oey L.-Y., Wang J.D. and Lee T.N., 1996. The fate of river discharge on the continental shelf - 1. Modeling the river plume and the inner shelf coastal current. Journal of Geophysical Research, **101**, 3415–3434.

Kranenburg C., 1983. The influence of side-wall friction on shear-stress driven entrainment experiments. Journal of Hydraulic Research, **21**, 99–117.

Kratzer S., Bowers D. and Tett P.B., 1999. The annual cycle of water colour and its interpretation in optical Case-2 waters. International Journal of Remote Sensing, in press.

Kundu P.K., 1981. Self-similarity in stress-driven entrainment experiments. Journal of Geophysical Research, **86**, 1979–1988.

Lancelot C. and Billen G., 1985. Carbon-nitrogen relationships in nutrient metablosm of coastal marine ecosystems. Advances in Aquatic Microbiology, **3**, 263–321.

Large W.G. and Pond S., 1981. Open ocean momentum flux measurements in moderate to strong winds. Journal of Physical Oceanography, **11**, 324–336.

Large W.G., McWilliams J.C. and Doney S.C., 1994. Oceanic vertical mixing: A review and a model with a nonlocal boundary layer parameterization. Reviews of Geophysics, **32**, 363–403.

Launder B.E. and Spalding D.B., 1974. The numerical computation of turbulent flows. Computer Methods in Applied Mechanics and Engineering, **3**, 269–289.

Laws E.A. and Bannister T.T., 1980. Nutrient- and light-limited growth of Thalassiosira fluviatilis in continuous culture, with implications for phytoplankton growth in the sea. Limnology and Oceanography, **25**, 457–473.

Levasseur M., Thompson P.A. and Harrison P.J., 1993. Physiological acclimation of marine phytoplankton to different nitrogen sources. Journal of Phycology, **29**, 587–595.

Liss P.S., 1988. Tracers of air-sea gas exchange. Philosophical Transactions of the Royal Society of London, **A325**, 93–103.

Luyten P.J., 1996. An analytical and numerical study of surface and bottom boundary layers with variable forcing and application to the North Sea. Journal of Marine Systems, **8**, 171–189.

Luyten P.J., 1997. Modelling physical processes of haline stratification in ROFIs with application to the Rhine outflow region. Journal of Marine Systems, **12**, 277–298.

Luyten P.J. (Editor), 1999. COHERENS – Dissemination and exploitation of a coupled hydrodynamical-ecological model for regional and shelf seas, MAS3-CT97-0088. Final Report. MUMM Internal Report, Management Unit of the Mathematical Models, 76 pp.

Luyten P.J. and De Mulder T., 1992. A module representing surface fluxes of momentum and heat. Technical Report No. 9 MAST-0050-C (MUMM), 30 pp.

Luyten P.J. and Rippeth T.P., 1997. A comparison study of turbulence closure schemes for tidal flows with application to the Irish Sea. Presented at the 29th Liège Colloquium on Marine Hydrodynamics. Turbulence Revisited, Liège, May 5–9 1997.

Luyten P.J., Deleersnijder E., Ozer J. and Ruddick K.G., 1996. Presentation of a family of turbulence closure models for stratified shallow water flows and preliminary application to the Rhine outflow region. Continental Shelf Research, **16**, 101–130.

Lynch D.R. and Davies A.M. (Editors), 1995. Quantitative skill assessment for coastal ocean models. Coastal and Estuarine Studies, Vol. 47, American Geophysical Union, Washington D.C., 510 pp.

Madec G., Chartier M., Delecluse P. and Crépon M., 1991. A three-dimensional numerical study of deep-water formation in the northwestern Mediterranean Sea. Journal of Physical Oceanography, **21**, 1349–1371.

Maier-Reimer E., 1980. On the formation of salt water wedges in estuaries. Lecture Notes on Coastal and Estuarine Studies, **1**, 91–101.

Maier-Reimer E. and Sündermann J., 1982. On Tracer Methods in Computational Hydrodynamics. Engineering Application of Computational Hydraulics, **1**, Pitman Advanced Publ. Program, Boston.

Mantoura R.F.C., Jeffrey S.W., Llewellyn C.A., Claustre H. and Morales C.E., 1997. Comparison between spectrophotometric, fluorometric and HPLC methods for chlorophyll analysis. In: S.W. Jeffrey, R.F.C. Mantoura and S.W. Wright (Editors), Phytoplankton Pigments in Oceanography. UNESCO, Paris, 361–380.

Marsaleix P., Estournel C., Kondrachoff V. and Véhil R., 1998. A numerical study of the Rhône river plume. Journal of Marine Systems, **14**, 99–115.

Martin P.J., 1985. Simulation of the mixed layer at OWS November and Papa with several models. Journal of Geophysical Research, **90**, 903–916.

Martinsen E.A. and Engedahl H., 1987. Implementation and testing of a lateral boundary scheme as an open boundary condition in a barotropic ocean model. Coastal Engineering, **11**, 603–627.

McCalpin J.D., 1994. A comparison of second-order and fourth-order pressure gradient algorithms in a σ-coordinate ocean model. International Journal for Numerical Methods in Fluids, **18**, 361–383.

Mellor G.L. and Blumberg A.F., 1985. Modeling vertical and horizontal diffusivities with the sigma coordinate system. Monthly Weather Review, **113**, 1379–1383.

Mellor G.L. and Yamada T., 1974. A hierarchy of turbulence closure models for planetary boundary layers. Journal of the Atmospheric Sciences, **31**, 1791–1806.

Mellor G.L. and Yamada T., 1982. Development of a turbulence closure model for geophysical fluid problems. Reviews of Geophysics and Space Physics, **20**, 851–875.

Mesinger F. and Arakawa A., 1976. Numerical methods used in atmospheric models. Global Atmospheric Research Programme (GARP) publication series No. 17, Volume 1, 64 pp.

Mesinger F. and Janjić Z.I., 1985. Problems and numerical methods of the incorporation of mountains in atmospheric models. Lectures in Applied Mathematics, **22**, 81–121.

Mills D.K. and Tett P.B., 1990. Use of a recording fluorometer for continuous measurement of phytoplankton concentration. SPIE Proceedings, **1269**, 106–115.

Mills D. K., Tett P. and Navarino G.F., 1994. The spring bloom in the south-western North Sea in 1989. Netherlands Journal of Sea Research, **33**, 65–80.

Mirbach K., 1997. Schwebstofftransport in Küstengewässern mit Frischwassereinfluss. Berichte aus dem Zentrum für Meeres- und Klimaforschung, Reihe B: Ozeanographie, No. 29, 82 pp.

Monin A. and Obukhov A., 1954. Basic turbulent mixing laws in the atmospheric near-surface layer. Trans. Geophys. Inst. Akad. Nauk USSR, **151**, 163–187.

Moum J.N., Caldwell D.R. and Paulson C.A., 1989. Mixing in the equatorial surface layer and thermocline. Journal of Geophysical Research, **94**, 2005–2021.

Munk W.H. and Anderson E.R., 1948. Notes on a theory of the thermocline. Journal of Marine Research, VII(3), 276–295.

Naimie C.E., Loder J.W. and Lynch D.R., 1994. Seasonal variation of the three-dimensional residual circulation on Georges Bank. Journal of Geophysical Research, **99**, 15967–15989.

Nihoul J.C.J. and Jamart B.M. (Editors), 1987. Three-dimensional models of marine and estuarine dynamics. Elsevier Oceanography Series, Vol. 45, 624 pp.

North Sea Task Force, 1993. North Sea Quality Status Report 1993. Oslo and Paris Commissions, London. Olsen and Olsen, Fredensborg, Denmark, 132 pp.

Oey L.-Y. and Chen P., 1992a. A nested-grid ocean model: With application to the simulation of meanders and eddies in the Norwegian coastal current. Journal of Geophysical Research, **97**, 20063–20086.

Oey L.-Y. and Chen P., 1992b. A model simulation of circulation in the Northeast Atlantic shelves and seas. Journal of Geophysical Research, **97**, 20087–20115.

Officer C.B., 1976. Physical Oceanography of Estuaries (and Associated Coastal Waters). John Wiley, New York, 465 pp.

Orlanski I., 1976. A simple boundary condition for unbounded hyperbolic flows. Journal of Computational Physics, **21**, 251–269.

Pacanowski R.C. and Philander S.G.H., 1981. Parameterization of vertical mixing in numerical models of tropical oceans. Journal of Physical Oceanography, **11**, 1443–1451.

Paffenhöfer G.-A., 1971. Grazing and ingestion rates of nauplii, copepods and adults of the marine planktonic copepod Calanus helgolandicus. Marine Biology, **11**, 286–298.

Paffenhöfer G.-A. and Harris R. P., 1976. Feeding, growth and reproduction of the marine planktonic copepod Pseudocalanys elongatus Boeck. Journal of the Marine Biological Association of the United Kingdom, **56**, 327–344.

Patankar S.V., 1980. Numerical heat transfer and fluid flow. Hemisphere, 197 pp.

Paulson C.A. and Simpson J.J., 1977. Irradiance measurements in the upper ocean. Journal of Physical Oceanography, **7**, 952–956.

Payne R.E., 1972. Albedo of the sea surface. Journal of the Atmospheric Sciences, **29**, 959–970.

Peters H., Gregg M.C. and Sanford T.B., 1995. Detail and scaling of turbulent overturns in the Pacific Equatorial Undercurrent. Journal of Geophysical Research, **100**, 18349–18368.

Peters H., Gregg M.C. and Toole J.M., 1988. On the parameterization of equatorial turbulence. Journal of Geophysical Research, **93**, 1199–1218.

Phillips N.A., 1957. A coordinate system having some special advantages for numerical forecasting. Journal of Meteorology, **14**, 184–185.

Prandle D., 1982. The vertical structure of tidal currents and other oscillatory flows. Continental Shelf Research, **1**, 191–207.

Press W.H., Flannery B.P., Teukolsky S.A. and Vetterling W.T., 1989. Numerical Recipes. The art of scientific computing. Cambridge University Press, Cambridge, 702 pp.

Price J.F., 1979. On the scaling of stress-driven entrainment experiments. Journal of Fluid Mechanics, **90**, 509–529.

Proctor R. (Editor), 1997. NOMADS – North Sea Model Advection-Dispersion study, MAS2-CT94-0105. Technical Report IV: Model Intercomparison. POL Internal Document No. 107, Proudman Oceanographic Laboratory, 143 pp.

Proctor R. and Davies A.M., 1996. A three dimensional hydrodynamic model of tides off the north-west coast of Scotland. Journal of Marine Systems, **7**, 43–66.

Proctor R., Flather R.A. and Elliott A.J., 1994. Modelling tides and surface drift in the Arabian Gulf – application to the Gulf oil spill. Continental Shelf Research, **14**, 531–545.

Puls W. and Sündermann J., 1993. Modelling the distribution of suspended matter and of lead in the North Sea. In: A.J. Mehta (Editor), Nearshore and estuarine cohesive sediment transport. American Geophysical Union, Washington D.C., pp. 520–540.

Reed R.K., 1977. On estimating insolation over the ocean. Journal of Physical Oceanography, **7**, 482–485.

Rew R., Davis G., Emmerson S. and Davies H., 1997. NetCDF User's Guide for FORTRAN. Version 3. Unidata Program Center, Boulder, Colorado, 146 pp. Available via http://www.unidata.ucar.edu/packages/netcdf/ .

Ridderinkhof H. and Zimmerman J.T.F., 1992. Chaotic stirring in a tidal system. Science, **258**, 1107–1111.

Riley, 1946. Factors controlling phytoplankton populations on Georges Bank. Journal of Marine Research, **6**, 54–73.

Rodi W., 1984. Turbulence models and their application in hydraulics. International Association for Hydraulic Research, 2nd edition, Delft, Netherlands, 104 pp.

Rodi W., 1987. Examples of calculation methods for flow and mixing in stratified fluids. Journal of Geophysical Research, **92**, 5305–5328.

Roe P.L., 1985. Some contributions to the modelling of discontinuous flows. In: Proceedings of 1983 AMS-SIAM summer seminar on large scale computing in fluid mechanics, Philadelphia. Lectures in Applied Mathematics, Vol. 22, pp. 163–193.

Røed L.P. and Cooper C.K., 1987. A study of various open boundary conditions for wind-forced barotropic numerical ocean models. In: J.C.J. Nihoul and B.M. Jamart (Editors), Three-dimensional models of marine and estuarine dynamics. Elsevier, Amsterdam, pp. 305–335.

Rosati A. and Miyakoda K., 1988. A general circulation model for upper ocean simulation. Journal of Physical Oceanography, **18**, 1601–1626.

Ruddick K.G., 1995. Modelling of coastal processes influenced by the freshwater discharge of the Rhine. Ph.D. Thesis, Univ. de Liège, Belgium, 247 pp.

Ruddick K.G., Deleersnijder E., Luyten P.J. and Ozer J., 1995. Haline stratification in the Rhine-Meuse freshwater plume: A three-dimensional model sensitivity analysis. Continental Shelf Research, **15**, 1597–1630.

Sakshaug E., Andresen K. and Kiefer D. A., 1989. A steady state description of growth and light absorption in the marine planktonic diatom Skeletonema costatum. Limnology and Oceanography, **34**, 198–205.

Semtner A.J. and Chervin R.M., 1988. A simulation of the global ocean circulation with residual eddies. Journal of Geophysical Research, **93**, 15502–15522.

Sharples J. and Tett P., 1994. Modelling the effect of physical variability on the midwater chlorophyll maximum. Journal of Marine Research, **52**, 219–238.

Sielecki A., 1968. An energy-conserving finite difference scheme for the storm surge equations. Monthly Weather Review, **96**, 150–156.

Signell R.P., Beardsley R.C., Graber H.C. and Capotondi A., 1990. Effect of wave-current interaction on wind-driven circulation in narrow, shallow embayments. Journal of Geophysical Research, **95**, 9671–9678.

Simpson J.H., Crawford W.R., Rippeth T.P., Campbell A.R. and Cheok J.V.S., 1996. The vertical structure of turbulent dissipation in shelf seas. Journal of Physical Oceanography, **26**, 1579–1590.

Smagorinsky J., 1963. General circulation experiments with the primitive equations – I. The basic experiment. Monthly Weather Review, **91**, 99–165.

Smith C.L. and Tett P.B., 1996. Development of one and two microplankton compartment models to simulate conditions on the Goban Spur (abstract). UK Oceanography 96, University of Wales, Bangor, 2–6 September 1996, Programme & Abstracts, p. 30.

Smith C.L. and Tett P., ms. Parameterisation of a numerical model containing one and two microplankton compartments for the Goban Spur region.

Smith S.D. and Banke E.G., 1975. Variation of the sea surface drag coefficient with windspeed. Quarterly Journal Meteorological Society, **101**, 665–673.

Soetaert K., Herman P.M.J. and Middelburg J.J., 1996. A model of early diagenetic processes from the shelf to abyssal depths. Geochimica et Cosmochimica Acta, **60**, 1019–1040.

Sosik H.M. and Mitchell B.G., 1991. Absorption fluorescence and quantum yield for growth in nitrogen limited Dunaliella tertiolecta. Limnology and Oceanography, **36**, 910–921.

Soulsby R.L., 1983. The bottom boundary layers of shelf seas. In: B. Johns (Editor), Physical Oceanography of Coastal and Shelf Seas. Elsevier Oceanography Series, Vol. 35, 189–266.

Spaulding M.L., 1988. A state of the art review of oil spill trajectory and fate modelling. Oil and Chemical Pollution, **4**, 39–55.

Stacey M.W. and Pond S., 1997. On the Mellor-Yamada turbulence closure scheme: the surface boundary condition for q^2. Journal of Physical Oceanography, **27**, 2081–2086.

Stacey M.W., Pond S. and Nowak Z.P., 1995. A numerical model of the circulation in Knight Inlet, British Columbia, Canada. Journal of Physical Oceanography, **25**, 1037–1062.

Steinhorn I., 1991. Salt flux and evaporation. Journal of Physical Oceanography, **21**, 1681–1683.

Stelling G.S. and Van Kester J.A., 1994. On the approximation of horizontal gradients in sigma coordinates for bathymetry with steep bottom slopes. International Journal for Numerical Methods in Fluids, **18**, 915–935.

Takacs L.L., 1985. A two-step scheme for the advection equation with minimized dissipation and dispersion errors. Monthly Weather Review, **113**, 1050–1065.

Tartinville B., Deleersnijder E., Lazure P., Proctor R., Ruddick K.G. and Uittenbogaard R.E., 1998. A coastal ocean model intercomparison study for a three-dimensional idealised test case. Applied Mathematical Modelling, **22**, 165–182.

Tett P., 1987a. Modelling the growth and distribution of marine microplankton. Society for General Microbiology Symposium, Cambridge University Press, **41**, 387–425.

Tett P., 1987b. The ecophysiology of exceptional blooms. Rapport et Proces-verbaux des Réunions, Conseil International pour l'Exploration de la Mer, **187**, 47–60.

Tett P., 1990a. A three layer vertical and microbiological processes model for shelf seas. Proudman Oceanographic Laboratory, Report no. 14, 85 pp.

Tett P., 1990b. The Photic Zone. In: P.J. Herring, A.K. Campbell, M. Whitfield and L. Maddock (Editors), Light and Life in the Sea. Cambridge University Press, Cambridge, U.K., pp. 59–87.

Tett P., 1998. Parameterising a microplankton model. Report, Napier University, Edinburgh, 54 pp.

Tett P. and Droop M.R., 1988. Cell quota models and planktonic primary production. In: J.W.T. Wimpenny (Editor), Handbook of Laboratory Model Systems for Microbial Ecosystems. CRC Press, Florida, 177–233.

Tett P. and Grenz C., 1994. Designing a simple microbiological-physical model for a coastal embayment. Vie et Milieu, **44**, 39–58.

Tett P. and Smith C., 1997. Modelling benthic-pelagic coupling in the North Sea. New Challenges for North Sea Research – 20 years after FLEX '76, Hamburg, 21-23 October 1996. Berichte aus dem Zentrum für Meeres- und Klimaforschung. Reihe Z: Interdisziplinäre Zentrumsberichte, **2**, 235–243.

Tett P. and Walne A., 1995. Observations and simulations of hydrography, nutrients and plankton in the southern North Sea. Ophelia, **42**, 371–416.

Tett P. and Wilson H., 1999. From biogeochemical to ecological models of marine microplankton. Journal of Marine Systems. In press.

Tett P., Heaney S.I. and Droop M.R., 1985. The Redfield ratio and phytoplankton growth rate. Journal of the Marine Biological Association of the United Kingdom, **65**, 487–504.

Tett P.B., Joint I.R., Purdie D.A., Baars M., Oosterhuis S., Daneri G., Hannah F., Mills D.K., Plummer D., Pomroy A.J., Walne A.W. and Witte H.J., 1993. Biological consequences of tidal stirring gradients in the North Sea. Philosophical Transactions of the Royal Society of London, **A 340**, 493–508.

Uittenbogaard R.E., 1996. Internal waves explain damping functions (abstract). MAST Workshop on Turbulence Modelling, August 8–10 1996, Bergen, Norway. 8 pp.

UNESCO, 1981. Tenth report of the joint panel on oceanographic tables and standards. UNESCO Technical Papers in Marine Science No. 36, UNESCO, Paris.

Van Damm G.C, 1994. The study of shear dispersion in tidal waters by applying discrete particle techniques. In: K.Beven, P.C. Chatwin and P. Millbank (Editors), Mixing and Transport in the environment. Wiley, New York, pp. 269–293.

van den Berg A.J., Ridderinkhof H., Riegman R., Ruardij P. and Lenhart H., 1996. Influence of variability in water transport on phytoplankton biomass and composition in the southern North Sea : a modelling approach (FYFY). Continental Shelf Research, **16**, 907–931.

Van Raaphorst W., Philippart C.J.M., Smit J.P.C., Dijkstra F.J. and Malschaert J.F.P., 1998. Distribution of suspended particulate matter in the North Sea as inferred from NOAA/AVHRR reflectance images and in situ observations. Journal of Sea Research, **39**, 197–215.

Varela R.A., Cruzado A. and Gabaldón J.E., 1995. Modelling primary production in the North Sea using the European Regional Seas Ecosystem Model. Netherlands Journal of Sea Research, **33**, 337–361.

Visser A.W., Souza A.J., Hessner K. and Simpson J.H., 1994. The effect of stratification on tidal current profiles in a region of freshwater influence. Oceanologica Acta, **17**, 369–381.

Vos R.J.and Schuttelaar M. 1995. RESTWAQ Data assessment, data-model integration and application to the southern North Sea. BCRS Report 95–19, Delft.

Vos R.J. and Schuttelaar M., 1997. An integrated data-model system to support monitoring and assessment of marine systems. In: Stel J.H. (Editor), Proceedings of the first EuroGOOS Conference, The Hague. Operational Oceanography. Elsevier Oceanography Series No. 62, pp 507–515.

Warrach K., 1998. Modelling the thermal stratification in the North Sea. Journal of Marine Systems, **14**, 151-165.

Wild-Allen K.A. and Lampitt R.S., 1998. Modelling and observation of marine snow on the continental slope (abstract). U.K. Oceanography 98, Southampton Oceanography Centre, Programme & Abstracts, p. 62.

Williams P.J.L., 1981. Incorporation of microheterotrophic processes into the classical paradigm of the plankton food web. Kieler Meeresforschungen, **5**, 1–28.

Xing J. and Davies A.M., 1996. Application of turbulence energy models to the computation of tidal currents and mixing intensities in shelf edge regions. Journal of Physical Oceanography, **26**, 417–447.

Yanenko N.N., 1971. The method of fractional steps. Springer Verlag, New York.

Zehr J.P., Falkowski P.G., Fowler J. and Capone D.G., 1988. Coupling between ammonium uptake and incorporation in a marine diatom: experiments with the short-lived radioisotope ^{13}N. Limnology and Oceanography, **33**, 518–527.