SPEAKING OF MATHEMATICS

Brian Hayes

A reprint from

American Scientist

the magazine of Sigma Xi, the Scientific Research Society

Volume 84, Number 2
March–April, 1996
pages 110–113

SPEAKING OF MATHEMATICS

Brian Hayes

Computers, and networks of computers, have opened up a new channel of communication for people who are blind or visually impaired. Much information stored and transmitted in electronic form—including vast streams of text that flow over the Internet—can be made accessible to the visually impaired reader through a computer equipped with a nonvisual output device. This device commonly takes the form of a text-to-speech transducer that reads aloud the content of a display screen. Compared with other media such as tape-recorded books and Braille publications, computerized sources of information offer the important advantages of immediacy and independence: Visually disabled readers get the information as soon as anyone else does, and they can get at it without assistance.

Text-to-speech systems work reasonably well with plain, linear prose, like that found in most newspaper articles and electronic-mail messages. But what about the visually impaired student of mathematics or computer science or engineering, whose reading matter may well include more-challenging constructs? Take a look at this equation:

$$\lim_{x\to\infty}\int_0^x e^{-y^2}\,dy = \frac{\sqrt{\pi}}{2}$$

Now try to visualize it without seeing it. A text-to-speech system is likely to stumble badly when trying to read such an expression. For one thing, the usual left-to-right convention of English fails here; in some places within the equation the natural sequence is from top to bottom, and elsewhere from bottom to top. Furthermore, reading the symbols in *any* fixed sequence yields a phonetic string so long that you've forgotten the beginning by the time you reach the end.

The challenges of teaching a computer to read mathematics aloud are taken up in a thoughtful series of papers by T. V. Raman, and in a software system called ASTER, which Raman developed while he was a doctoral candidate at Cornell University. ASTER performs audio formatting and rendering of mathematical notation, and it allows the listener to browse actively through complex mathematical expressions and other forms of structured text. Systems like ASTER are obviously important to the specific community that needs them most, but they have a wider significance as well. ASTER illustrates some subtle principles about how best to encode and present information, principles of value to everyone.

Brian Hayes is a former editor of American Scientist. *Address: 211 Dacian Avenue, Durham, NC 27701. Internet: bhayes@amsci.org.*

A Two-Dimensional Language

Mathematics is an unusual language. In ordinary languages speech is primary, and the written form is a later addition, devised as a way of recording what is said. Hence there is a reasonably direct, one-to-one mapping between written and spoken language. Of particular importance, both forms are one-dimensional, at least in their surface structure: One word follows another in a definite sequence. When reading aloud from a written text, you don't even need to know the meaning of the words; you need only know the rules of pronunciation and prosody. That's what makes text-to-speech systems possible; if a computer had to understand a sentence before it could read it aloud, the listener would have a long wait.

Mathematical notation is different. Mathematics is first of all a written language, with a few speech conventions imposed on it after the fact. When mathematicians talk shop, they do so at the blackboard; in a more formal setting, a mathematician giving a talk comes equipped with transparencies to project. Perhaps the clearest sign that mathematical notation evolved initially as a system of writing rather than speaking is its reliance on the two dimensions of the written page. Superscripts and subscripts, to cite a commonplace example, derive their meaning from their position above or below the baseline, a concept that has no immediate oral counterpart. And it would be hard to imagine anything more patently two-dimensional than the notation for a matrix. If ordinary language is linear, then mathematical writing is planar.

The two-dimensional nature of mathematical notation leads to awkwardness in at least one other context besides spoken communication. The context is that of writing equations on a computer. A computer text file is strictly one-dimensional; it is a sequence of characters, typically represented as eight-bit bytes, with no higher-level structure. In the computer's memory you cannot place one symbol "above" another, nor can you arrange to shift one byte a little below the baseline. It's an ironic situation: The most mathematical of machines cannot accommodate the language of mathematics.

Among the various remedies for this problem, the most thoroughgoing and also the most widely adopted is the TEX formatting language, developed by Donald E. Knuth of Stanford University. TEX was extended by Leslie Lamport of the Digital Equipment Corporation to create a somewhat higher-level language called LATEX. (The names are pronounced *tech* and *lah-tech*. As for their odd typographical treatment, there is a rationale for it, but the reader might be forgiven for suspecting that the original purpose was to show off what the systems are capable of doing.)

TEX linearizes the two-dimensional layout of an equation. Here is the LATEX encoding of the equation given on the opposite page:

```
$$\lim_{x \to \infty}\int_0^x e^{-y^2}\dy =
\frac{\sqrt{\pi}}{2}$$
```

Terms that begin with a backslash are "control sequences," most of which are easy to figure out. For instance, the sequence \infty generates the sign ∞, \int is the integral sign $\int$, and \frac makes a fraction out of the two groups of symbols that follow it. The underscore and caret (_ and ^) designate subscripts and superscripts respectively. Of course no one would want to read mathematics in this form, but raw TEX is not meant for human consumption. It is processed by a computer program that renders it in a more palatable form on a display screen or on the printed page.

ASTER is also a computer program that accepts TEX notation as input and produces a rendering as output, but the rendering is audible rather than visual. The program does not simply read out the TEX code literally; a rendering that began "dollar dollar backslash lim underscore left-bracket x …" would be incomprehensible. What is needed is an approximation to the oral rendering that would be given by a mathematically knowledgeable human reader, perhaps something like: "The limit, as *x* goes to infinity, of the integral, from *y* equals zero to *x,* of *e* to the minus *y* squared, *dy,* equals the square root of pi, over 2." (I have had a hard time punctuating this sentence, because of the unusual pattern of pauses that readers employ to indicate the grouping of mathematical expressions. Perhaps this difficulty is another sign of the tenuous connection between written and spoken mathematics.)

The Making of Tools

The name ASTER stands for Audio System for Technical Readings, but it is no coincidence that Raman has a guide dog named Aster, "a big, friendly black Labrador." The typographical treatment of the term ASTER is obviously modeled on that of TEX and LATEX, but how is it rendered audibly? How does ASTER say ASTER? By speaking the word with a dog's bark in the background. Raman adds a disclaimer: "The bark is that of a generic dog. Aster is too well trained to bark, and could not therefore be recorded."

There is a demonstration of ASTER on the World Wide Web (see the bibliography below for the URL), where mathematical expressions are rendered graphically as well as in TEX and in sound files that reproduce ASTER's output. The Web site also supplies copies of Raman's papers and his dissertation. And the dissertation is also distributed by Recordings for the Blind in a version read by ASTER itself—the first computer-spoken book to be made available in this way.

The dissertation includes a biographical sketch. "T. V. Raman was born and raised in Pune, India. He was partially sighted (sufficient to be able to read and write) until he was 14. Thereafter, he learned with the help of his brother, who spent a great deal of time as his first reader/tutor.… Raman received his B.A. in Mathematics at Nowrosjee Wadia College in Pune and his Masters in Math and Computer Science at the Indian Institute of Technology, Bombay. For his final-year project, he developed CONGRATS, a program that allowed the user to visualize curves by listening to them.…"

Raman entered the doctoral program in applied mathematics at Cornell in 1989 and received his Ph.D. in 1994. His dissertation, supervised by David Gries, won the Distinguished Doctoral Dissertation Award of the Association for Computing Machinery that year. After taking his degree, he held a research position at the Digital Equipment Corporation and is now with Adobe Systems.

Raman's primary motive for developing ASTER was to facilitate his own reading, particularly for his studies at Cornell. Various textbooks and course notes were available as TEX or LATEX documents, and Raman needed a tool to read them with. He decided to build his own, starting with a simpler program called TEXTALK, then going on to ASTER. It is interesting to note that TEX also began as a project meant purely to satisfy the author's own needs. Knuth was frustrated with the cumbersome process of typesetting for his magnum opus, the multivolume *Art of Computer Programming*, and so he took a decade off to do something about it.

Parsing Mathematics

ASTER has three main components. A recognizer parses LATEX notation and creates an internal representation that is easier for the program to manipulate. An audio formatting language, called AFL, renders the parsed text using both speech and nonspeech sounds. The third component is a facility for audio browsing, or actively traversing the structure of a document.

The recognizer extracts structure and ideally even meaning from the TEX-encoded text. When given a mathematical expression, it parses the entire input before the audio rendering begins. Looking ahead in the text is something that even simple speech-synthesis systems may have to do—for example, a question mark at the end of a sentence can alter the intonation of the beginning—but those systems never rearrange the words spoken. AsTeR must carry out some deeper transformations.

A simple example is the audio formatting of the expression $\log_{10} x$, which a listener might prefer to hear spoken as "the logarithm of x to the base 10." In creating this rendering, AsTeR cannot simply process the symbols in their original sequence. Integrals present a similar challenge, because the listener needs to know the variable of integration as soon as possible. Thus

$$\int_0^\infty e^{-x} dx$$

might be read as "the integral with respect to x, from zero to infinity, of e to the minus x, dx." The LaTeX encoding is $$\int_0^\infty {e^{-x}}\dx$$, which requires AsTeR to search out the \dx at the end of the expression before it can render the \int at the beginning. To generate renderings for texts like these, AsTeR must break the expression down into its component pieces and then reassemble them in a different order.

Mathematical expressions have a treelike structure. The equation $y = x + 2$ can be understood as a tree that has the = operator as its root, with two branches. One of the branches consists of the symbol y, whereas the other branch is a subtree with the operator + as its root and with further branches x and 2. The tree can be represented in prefix notation as (= y (+ x 2)). AsTeR employs a similar notation internally.

Parsing is a straightforward matter for mathematical expressions written in a programming language such as Fortran or Pascal. But real mathematical writing, including its TeX representation, is highly ambiguous. An expression such as $f(x + y)$ might mean "the product of f and $x + y$," or it might mean "the function f applied to $x + y$." Similarly, cos x sin y could be parsed as either (× (cos x) (sin y)) or as (cos (× x (sin y))). Superscripts are another construct that can have multiple meanings: x^{-1} means $1/x$, but $\sin^{-1}$ refers to the inverse sine function; A^T is probably the transpose of a matrix and D^2 may be a second derivative. AsTeR's recognizer is able to resolve many of these ambiguities; those that remain are left until the rendering phase, when the user can specify how a particular notation is to be interpreted.

Speaking Mathematics

When the recognizer has done its work, the second component of AsTeR takes over to render the parsed expression in sound. It does so by applying rules written in AFL, the audio formatting language. The rules determine not only what words are spoken but also *how* they are spoken, controlling the pitch and speed of the voice and a variety of other qualities such as breathiness and smoothness. The rules also invoke nonspeech audio cues.

AsTeR's standard rule for rendering fractions reads a simple expression such as a/b as "a over b," but a more complicated instance such as $(x + y)/(x - y)$ is given as "the fraction with numerator $x + y$ and denominator $x - y$." A few special cases are recognized, so that 1/2 can be rendered as "one-half" rather than "one over two." All of the AFL rules are subject to modification.

The rendering of superscripts and subscripts is an area where changes in voice quality provide an intuitive vocal analogue of the visual rendering. Superscripts are read at a higher pitch and subscripts at a lower pitch. Such voice cues can help to resolve ambiguities in an audio rendering. For instance, $x^n + 1$ is readily distinguished from x^{n+1}, even without an explicit verbal marker of where the exponent ends.

An even more direct mapping from visual space to auditory space helps the listener to discern the structure of tables and matrices. With stereophonic output, AsTeR can vary the relative loudness of the left and right sound channels while reading the rows of a matrix, so that the voice seems to be moving through the structure.

Nonspeech sounds provide a concise and unobtrusive way of conveying certain other textual features. In a bulleted list, a brief tone can announce each new item, rather than repeating the word "bullet." Sounds played continuously in the background while speech continues can serve to emphasize or highlight a passage of text, providing an audio equivalent of italic type and boldface.

The aim of these various devices is to create a true audio notation for mathematics. In written mathematics, succinct notation allows the overall structure of an expression to be taken in at a glance, whereas the same concepts expressed in words would have to be laboriously parsed. AsTeR seeks in a similar way to shift some of the work of listening from the cognitive to the perceptual domain.

Active Listening

Audio formatting makes much information accessible, but not necessarily digestible. Raman writes: "The passive printed document is processed by an active reader, who can view it in many different ways—read only section titles, skip a piece of mathematics, temporarily skip to a different page to read a referenced theorem, reread an interesting passage, and so on.... When it comes to audio, on the other hand, the *document* is the active player and the human is the passive one. The speaker (perhaps on an audio cassette) actively reads in a relentlessly linear fashion, from beginning to end, and the listener simply listens, with little control over the process."

AsTeR gives a measure of control back to the listener. The key to this capability is the treelike internal representation of the text, which describes both the details of mathematical expressions and the larger-scale architecture of the entire document, with its headings and subheadings and other structural markers. Keyboard commands allow for quick navigation through this tree, with much greater flexibility than the fast-forward and rewind controls of a tape player. You can skip from one subhead to the next within an article, or from one term to the next within an equation, then delve into any selected structure in greater detail.

Writing as Programming

ASTER is based on what the computing professions would recognize as a "client-server" model of the publication process. The author or publisher, taking the role of server, encodes the content of a document; the reader, as client, decides how the content will be presented. For this model to work well, the server must choose a style of encoding that captures the underlying meaning of a document, not just its graphic layout.

A markup language like TEX can be used in either way. It has low-level primitives that simply define the appearance of objects. The superscript operator is one of these: It indicates that a character is to be raised above the baseline, without giving a clue to whether the character is an exponent, a limit, or something else. But other commands specify structure more than layout; they include \title and \footnote as well as \frac and \sqrt.

Unfortunately, authors do not always distinguish between content and layout. A particularly troublesome practice is the overloading of operators to give them multiple meanings. On the printed page, notations for a fraction, a Legendre symbol and a logical inference might all look alike, but if they are all built with the \frac control sequence, ASTER and other rendering programs will be unable to distinguish them. The overloading is needless, because TEX makes it easy to define a new control sequence for each concept.

Authors and publishers would be well advised to avoid overloading and ambiguity—and not just for the convenience of certain readers with special needs. Authors cannot know in advance how their works will eventually be put to use, and there may be occasion later to give thanks for extra care taken now. Witness the current scramble to convert documents in dozens of haphazard formats to HTML, the markup language for the World Wide Web. The job would be easier if authors had not written with the thought that the printed page is the final product.

Knuth has argued that the writing of computer programs is first of all a kind of *writing*, and ought to be judged by literary standards. Today the opposite assertion also holds: Writing is a kind of computer programming, in which you prepare not just a printed document but the source code that will generate many renderings of that document. And thus the writer may need to learn some lessons from the software engineer.

Even if ASTER could fluently read all LATEX documents, most of the world's literature would still remain out of reach. A few documents are encoded in formats that lend themselves even better than TEX does to flexible renderings, such as SGML, the Standard Generalized Markup Language. But most documents exist only on paper or in electronic formats that preserve only layout information. Among these formats the most important are Postscript and PDF (Portable Document Format), both of which were invented by Adobe Systems. Adobe has recently announced a plan to address this issue. They have someone on the staff who is certainly well qualified to do so.

Bibliography

Note: The publications of T. V. Raman are available at the World Wide Web site <http://www.research.digital.com/CRL/personal/raman/raman.html>. The demonstration of ASTER is at the URL <http://www.cs.cornell.edu/Info/People/raman/aster/aster-toplevel.html>.

Knuth, Donald E. 1984. *The TEXbook*. Reading, Mass.: Addison-Wesley.

Knuth, Donald E. 1992. *Literate Programming*. Stanford, Calif.: Center for the Study of Language and Information.

Lamport, Leslie. 1986. *LATEX: A Document Preparation System*. Reading, Mass.: Addison-Wesley.

McQuarrie, Liz. 1995. "The Accessibility of PDF and Adobe Acrobat Viewers for the Visually Disabled." Adobe Systems. <http://www.adobe.com/ Acrobat/Access.html>

Raman, T. V. 1994. *Audio System for Technical Readings*. Ph.D. Dissertation. Cornell University. Audio edition distributed by Recordings for the Blind, order number FB190.

Raman, T. V., and David Gries. 1994. "Documents Mean More Than Just Paper!" In *Proceedings of the Second International Workshop on the Principles of Document Processing*.

Raman, T. V. 1995. "ASTER—Towards Modality-Independent Electronic Documents."