ONLINE CENTERED NLMS ALGORITHM FOR CONCEPT DRIFT COMPENSATION

M. Cejnek[*], *J. Vrba*[†]

Abstract: This paper introduces an online centered normalized least mean squares (OC-NLMS) algorithm for linear adaptive finite impulse response (FIR) filters and neural networks. As an extension of the normalized least mean squares (NLMS), the OC-NLMS algorithm features an approach of online input centering according to the introduced filter memory. This key feature can compensate the effect of concept drift in data streams, because such a centering makes the filter independent from the nonzero mean value of signal. This approach is beneficial for applications of adaptive filtering of data with offsets. Furthermore, it can be useful for real-time applications like data stream processing where it is impossible to normalize the measured data with respect to its unknown statistical attributes. The OC-NLMS approach holds superior performance in comparison to the NLMS for data with large offsets and dynamical ranges, due to its input centering feature that deals with the nonzero mean value of the input data. In this paper, the derivation of this algorithm is presented. Several simulation results with artificial and real data are also presented and analysed to demonstrate the capability of the proposed algorithm in comparison with NLMS.

Key words: *gradient methods, adaptive algorithms, concept drift, streaming data*

Received: July 16, 2020 **DOI:** 10.14311/NNW.2021.31.018
Revised and accepted: October 30, 2021

1. Introduction

The normalized least-mean-squares (NLMS) algorithm [9] is the most common modification of least-mean-squares (LMS) for FIR filters [4]. In field of neural networks, the LMS is the most known and used gradient descent (GD) algorithm, used for various applications in the fields of cybernetics (e.g. signal processing and computational intelligence). The GD algorithm is quite widely applicable in the sense of system identification [3], neural network learning [7], echo cancelling [11], time series prediction [10].

[*]Matous Cejnek – Corresponding author; Dept. of Instrumentation and Control Engineering, Faculty of Mechanical Engineering, Center of Advanced Aerospace Technology, Czech Technical University in Prague, Technicka Street 4, 16607, Prague 6, Czechia, E-mail: matous.cejnek@fs.cvut.cz

[†]Jan Vrba; Department of Computing and Control Engineering, University of Chemistry and Technology in Prague, Czechia, E-mail: jan.vrba@vscht.cz

However, the mentioned algorithms are derived according to the independence assumptions of adaptive filters analysis [12] and that is a problem for most of the real-life applications, especially for neural networks working in real-time applications. According to these assumptions, the signal used by the adaptive filter should be zero mean and stationary. Such conditions cannot be fulfilled in various cases of online signal processing. If an adaptive algorithm is used offline, it is possible to transform the data in a more suitable form (data normalization) to eliminate such issues. For example z-score transformation is a frequently used operation. The transformation stands as follows

$$x_z(k) = \frac{x(k) - \bar{x}}{\sigma_x}, \tag{1}$$

where the $x(k) \in \mathbb{R}$ is a variable (input or target of filter), the $\bar{x} \in \mathbb{R}$ is mean value of $x(k)$, $\sigma_x \in \mathbb{R}$ is standard deviation of $x(k)$ and k is the discrete time index. This transformation is used during offline preprocessing because it may decrease the condition number of used data. Hence, this transformation leads to better performance of adaptive algorithms. However, in a lot of applications, it is computationally demanding to use adaptive algorithms online. In this case, it is not possible to transform the data according to statistical attributes that are not known in a given time frame.

Methods how to deal with this issue have been developed in the past for analog adaptive filters [6, 8], based on the necessity for DC offset compensation. However, DC offsets in given applications are considered much smaller than the information in a signal. Thus, the offsets do not represent a significant challenge. However, this is not the case of digital signal processing of data streams, where the offsets can be much bigger than the information in data. These issues have not been so readily studied, hence algorithms as such that of the proposed are necessary.

This proposed method is an extension of the previously presented approach [2]. The previous method was designed for use only in cases where the filter output was formed only from the history of the filter target. The proposed approach is inspired by data centering. Because the statistical attributes of future data are not known during real-time measuring of the data stream, the proposed approach can use only the actual data or data from memory. This proposed modification of learning is based on NLMS (instead of LMS) because of the NLMS ability to scale the learning rate to preserve the convergence. In this paper, the proposed algorithm is called OC-NLMS (online centered NLMS).

1.1 Review of NLMS

Assume the output of the adaptive filter $\tilde{y}(k)$ is given as

$$\tilde{y}(k) = w_1 \cdot x_1(k) + \cdots + w_n \cdot x_n(k) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k), \tag{2}$$

where $(\cdot)^{\mathrm{T}}$ denotes the transposition, $\tilde{y}(k)$ is the filtered signal, $\mathbf{w} = [w_1(k), \ldots, w_n(k)]$ is a vector of adaptive filter parameters (at the beginning the parameters are set to random numbers with normal distribution, zero mean and unit standard deviation) and $\mathbf{x}$ is the input vector (for a filter of size n) as follows

$$\mathbf{x}(k) = [x_1(k), \ldots, x_n(k)]. \tag{3}$$

The NLMS algorithm is a modification of the classical least-mean-squares algorithm (LMS), also known as stochastic gradient descent. The LMS weight adaptation is given as follows

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \Delta\mathbf{w}(k), \tag{4}$$

where $\Delta\mathbf{w}(k)$ is

$$\Delta\mathbf{w}(k) = \frac{1}{2}\mu\frac{\partial e^2(k)}{\partial\mathbf{w}(k)} = \mu \cdot e(k) \cdot \mathbf{x}(k), \tag{5}$$

where $\mu \in \mathbb{R}$ is the learning rate (step size) and $e \in \mathbb{R}$ is error defined as

$$e(k) = y(k) - \tilde{y}(k), \tag{6}$$

where the $\tilde{y}(k)$ is the filtered signal and the $y(k)$ is the original signal (target). According to the general stability criteria of LMS [4]

$$|1 - \mu \cdot ||\mathbf{x}(k)||^2| \leq 1, \tag{7}$$

the NLMS adaptation rule is given as follows

$$\Delta\mathbf{w}(k) = \eta(k) \cdot \mathbf{w}(k) \cdot e(k), \tag{8}$$

where $\eta(k)$ is a learning rate normalized with $||\mathbf{x}(k)||^2$ (input signal power) as follows

$$\eta(k) = \frac{\mu}{\epsilon + ||\mathbf{x}(k)||^2}, \tag{9}$$

where $\epsilon \in \mathbb{R}$ is a small positive constant (regularization term) introduced to preserve stability for inputs close to zero [4]. The NLMS with ϵ is also called ϵ-NLMS. The filter is stable if

$$0 \leq \mu \leq 2 + \frac{2\epsilon}{||\mathbf{x}(k)||^2}, \tag{10}$$

or for the case without regularization term ϵ

$$\mu \in \langle 0, 2 \rangle. \tag{11}$$

It is important to highlight that the NLMS is beneficial over the LMS in the case of suboptimal data. However, the NLMS scaling of the learning rate is used to deal with both the offset and scale of the input data (and not just with the scale). This is the limitation of the NLMS. Moreover, note that the normalized range (11) for learning rate makes the NLMS much easier to apply then the LMS in real-life scenarios.

2. Filter model

2.1 Data transformation approach

Assume the error caused by data offset [6] is given as follows

$$e_o(k) = y(k) - (\mathbf{x}(k) + \mathbf{m}_x)^{\mathrm{T}}\mathbf{w}(k) + m_e, \tag{12}$$

where vector $\mathbf{m}_x$ represents the offset of the input data and the $m_e \in \mathbb{R}$ is offset of the error.

As mentioned before, the NLMS deals with offset as in a similar manner to scale via the change of learning rate. The proposed algorithm extends the NLMS with centering of the input vector. Thus, the proposed algorithm normalizes the learning rate to deal with scale after the data offset described by (12) is removed according to the actual or historical inputs of the filter.

During online signal processing, it is possible to use only historical data to estimate the statistical attributes of data. Data attributes like the mean value are changing during the process of filtering. This causes information loss, where older data are centered on different values than newer data. Thus, making this approach valuable only in the case where information loss is a smaller problem than a bad condition number of the data (ill-conditioned input matrix).

2.2 Proposed algorithm

The proposed OC-NLMS algorithm is based on (2) extended with centering for compensation of the offset described by (12) as follows

$$\tilde{y}(k) = (\mathbf{x}(k) - \bar{\mathbf{x}})^{\mathrm{T}}\mathbf{w}(k) + \bar{y}, \tag{13}$$

where the $\bar{y}$ is the mean value of historic filter target, $\bar{\mathbf{x}}$ is a vector of mean values of input variables. In practice the $\bar{\mathbf{x}}$ is replaced with $\bar{\mathbf{x}}_h$ estimated according to known historic input data $\mathbf{x}$ and the same replacement have to be done with $\bar{y}_h$. Therefore, such replacement yields the following conditions

$$\bar{\mathbf{x}} \approx \bar{\mathbf{x}}_h, \bar{y} \approx \bar{y}_h. \tag{14}$$

Error $e(k)$ of the proposed filter may thus be expressed as follows

$$e(k) = y(k) - \tilde{y}(k) = y(k) - (\mathbf{x}(k) - \bar{\mathbf{x}})^{\mathrm{T}}\mathbf{w}(k) - \bar{y}. \tag{15}$$

From this, it is possible to derive the OC-NLMS learning rule (in the same way as the derived NLMS learning rule) as follows

$$\Delta\mathbf{w}(k) = \frac{1}{2}\eta(k)\frac{\partial e^2(k)}{\partial \mathbf{w}(k)} = \eta(k) \cdot e(k) \cdot (\mathbf{x}(k) - \bar{\mathbf{x}}), \tag{16}$$

where the normalized learning rate is estimated in every step as follows

$$\eta(k) = \frac{\mu}{\epsilon + ||\mathbf{x}(k) - \bar{\mathbf{x}}||^2}. \tag{17}$$

2.3 Stability and convergence

For better readability, the following substitution is used in this subsection

$$\mathbf{x}_c(k) = (\mathbf{x}(k) - \bar{\mathbf{x}}). \tag{18}$$

To achieve the convergence of this OC-NLMS algorithm, the following condition must be fulfilled

$$|y(k) - \mathbf{x}_c(k)^{\mathrm{T}}\mathbf{w}(k) - \bar{y}| \geq |y(k) - \mathbf{x}_c(k)^{\mathrm{T}}\mathbf{w}(k+1) - \bar{y}|. \tag{19}$$

The new weights $\mathbf{w}(k+1)$ may be obtained from equations (4) and (16). And after simplification, the stability criteria stand as follows

$$|1 - \eta(k) \cdot ||\mathbf{x}_c(k)||^2| \leq 1. \tag{20}$$

Note that OC-NLMS normalized learning rate η is described by (17), thus it is different from (9) used by NLMS. For selection of the default learning rate, the following criteria may be applied (10).

3. Experimental analysis

In this section, experiments and their results to prove the validity of the proposed approach are described. In all experiments, the OC-NLMS was implemented with memory for only the last 100 and 500 samples (according to the experiment). The centering vector $\bar{\mathbf{x}}_h$ is estimated only in accordance with this memory. The regularization term was set to $\epsilon = 0$ because the introduction of the regularization term did not improve the performance for all tested cases.

3.1 Identification of SISO system

The analysis compares performance of the proposed algorithm with the NLMS algorithm for the task of unknown system identification. The unknown system has 10 randomly generated taps. The taps were taken from a distribution with zero mean and unit standard deviation. The adaptive filter has the same order as the unknown system ($n = 10$). The input of the system was both linear and nonlinear signal $u(k)$. The linear signal was formed by an auto-regressive filter, which may be represented as follows

$$u(k) = 0.9u(k-1) + q(k), \tag{21}$$

where $q(k) \in \mathbb{R}$ is normally distributed random variable with mean value $a \in \mathbb{R}$ and standard deviation $b \in \mathbb{R}$. The non-linear signal was formed from the benchmark system [5] with random variable $q(k) \in \mathbb{R}$ as follows

$$u(k) = \frac{u(k-1)}{1 + u(k-1)^2} + q(k)^3. \tag{22}$$

The Gaussian white noise $v(k) \in \mathbb{R}$ was added to output of the unknown system. The mean value of this noise was 0, and the standard deviation $\sigma_v \in \mathbb{R}$ was adjusted to achieve 30dB of SNR (signal to noise ratio) as follows

$$\text{SNR} = 10 \log_{10} \frac{\sigma_y^2}{\sigma_v^2}, \tag{23}$$

where σ_y is standard deviation of the unknown system output.

For every experimental setup (type of signal, offset a, amplitude b) 100 experiments were run and an average was made. The optimal default learning rate μ for every experiment and algorithm was found by testing, where the criteria were

an average of the mean-squared deviation (MSD) after convergence. The MSD is estimated as follows

$$\mathrm{MSD} = \mathrm{E}||\mathbf{w}^o(k) - \mathbf{w}(k)||^2, \tag{24}$$

where $\mathbf{w}^o(k)$ are parameters of the target system. The used experimental setups with simulation results are in Tab. I. Fig. 1 and Fig. 2 show the MSD curves for OC-NLMS and NLMS in case of suboptimal inputs of the unknown SISO system. In Fig. 2 it is possible to observe an increase of the OC-NLMS MSD around the time index value of 2500. This increase is caused by the final error of the adaptive parameters. This error is decreased at the end of the learning because the adaptive parameters cross the correct parameter values (and thus they are temporally lower). This behaviour of OC-NLMS adaptive filter is obvious from Fig. 3.

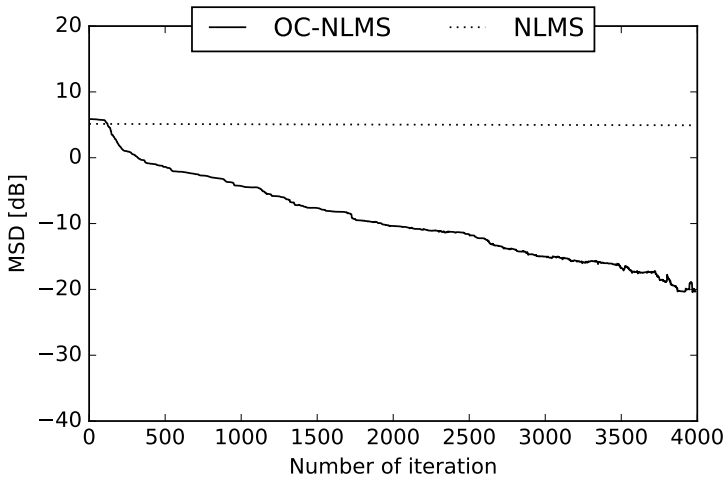

Fig. 1 *Performance comparison between OC-NLMS and NLMS for identification of system with linear input (mean value of 100 and standard deviation of 10).*

System input			μ		average MSD	
Type	a	b	OC-NLMS	NLMS	OC-NLMS	NLMS
linear	0	1	0.1	0.1	7.31E-4	3.37E-4
linear	100	10	0.02	0.9	1.14E-3	8.35E-2
non-linear	0	1	0.02	0.02	1.45E-3	9.02E-4
non-linear	100	10	0.02	0.01	1.23E-3	9.85E+0

Tab. I *Identification experimental setups and results for SISO systems. Inputs of the system are processed via linear (21) and non-linear (22) equations. The "a" is a mean value and the "b" is the standard deviation of the system input.*

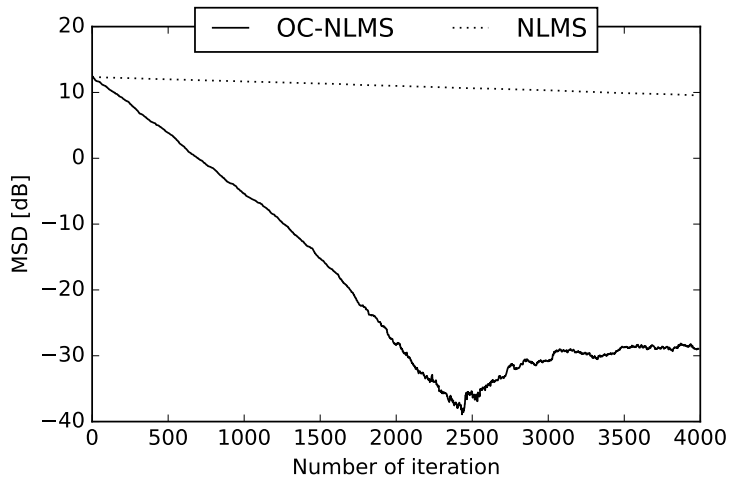

Fig. 2 *Performance comparison between OC-NLMS and NLMS for system with non-linear input (mean value of 100 and standard deviation of 10).*

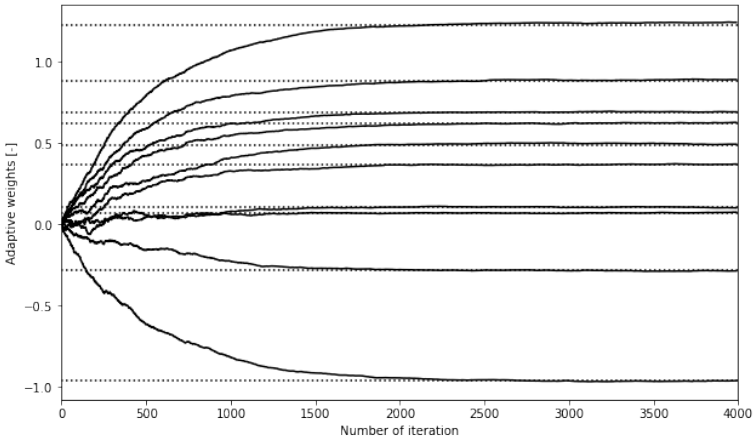

Fig. 3 *Detail on adaptation of the OC-NLMS algoritm individual adaptive weights – example with SISO system identification.*

3.2 Identification of MISO system

For this experiment, a MISO (multi-input-single-output system) was used. The system may thus be represented via the following relation

$$y(k) = \mathbf{h}^{\mathrm{T}}\mathbf{u}(k) \tag{25}$$

where $\mathbf{h}$ are randomly chosen constants (normal distribution, zero mean, unit standard deviation) and $\mathbf{u}(k)$ is the input vector. The input vector contains 3 previous values from each 3 independent inputs – $u_1(k), u_2(k), u_3(k)$. Thus, the size of the

system is $n = 9$. The independent inputs are formed as follows

$$u_1(k) = 0.9u_1(k-1) + q_1(k), \tag{26}$$

$$u_2(k) = 0.7u_2(k-1) - 0.1u_2(k-2) + q_1(k), \tag{27}$$

$$u_3(k) = \frac{u_3(k-1)}{1 + u3(k-1)^2} + q_3(k), \tag{28}$$

where $q_1(k)$, $q_2(k)$ and $q_3(k)$ are the normally distributed random variables ($\bar{q}_1 = 100$, $\sigma_{q1} = 10$, $\bar{q}_2 = 70$, $\sigma_{q2} = 15$, $\bar{q}_3 = -80$, $\sigma_{q3} = 6$). To the output of the system $y(k)$ was added measurement noise to meet 30dB of SNR (23). The adaptive filters used in this experiment use the same input vector as the system. The founded optimal learning rates are 0.05 for OC-NLMS and 0.9 for NLMS. The results of the experiment are shown in Fig. 4 in form of MSD curves

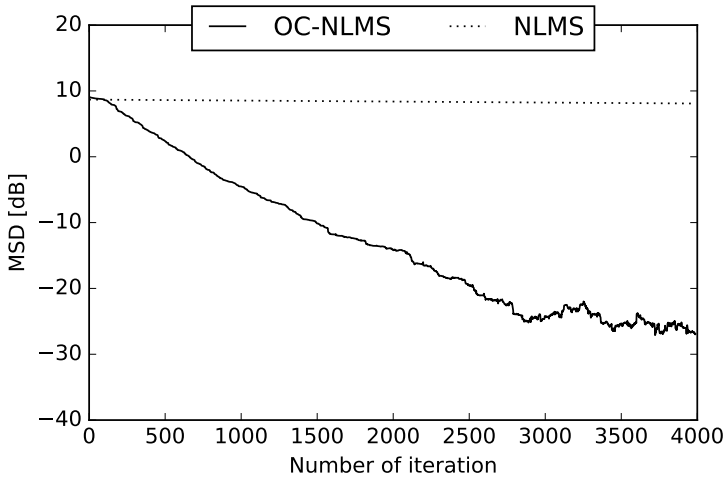

Fig. 4 *Comparison of performance between OC-NLMS and NLMS for MISO system – inputs with different offset.*

3.3 Time series prediction

This experiment uses an open-source data-set [1]. The featured time series are records of the temperature in 36 cities, thus being nonstationary. The temperatures contain an offset from 0, because they are recorded in degrees Fahrenheit. The temperatures are recorded hourly in the years 2015 to 2017. The data set contains corrupted areas in some series in the first half of the data. This problematic part of the data was skipped for all cities to ensure that all series have the same length. The adaptive filters were used in the prediction setup with one sample ahead the prediction horizon (one hour future). The input of the filter consists of 4 days history ($4 * 24$ samples). The best learning rates μ for both tested algorithms (OC-NLMS, NLMS) were found experimentally to achieve the best results for both

measured metrics – mean-absolute-error (MAE):

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |e_i|}{n}, \tag{29}$$

and root-mean-squared error (RMSE):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}. \tag{30}$$

Town name	OC-NLMS				NLMS				OC-NLMS advantage	
	MAE		RMSE		MAE		RMSE		MAE [%]	RMSE [%]
	Value	μ	Value	μ	Value	μ	Value	μ		
Vancouver	0.568	0.18	0.814	0.16	0.638	1.26	0.938	1.26	11.03	13.21
Portland	0.619	0.22	0.84	0.22	0.737	1.38	1.034	1.36	15.95	18.74
San Francisco	0.618	0.22	0.842	0.22	0.802	1.44	1.11	1.4	23.01	24.18
Seattle	0.547	0.22	0.735	0.24	0.609	1.4	0.856	1.4	10.25	14.12
Los Angeles	0.698	0.2	1.048	0.2	0.944	1.34	1.448	1.22	26.11	27.65
San Diego	0.714	0.18	1.052	0.18	0.87	1.28	1.398	1.2	17.99	24.71
Las Vegas	0.871	0.16	1.28	0.16	1.119	1.28	1.649	1.14	22.14	22.4
Phoenix	0.725	0.24	1.032	0.22	1.066	1.44	1.465	1.32	31.95	29.58
Albuquerque	1.014	0.16	1.38	0.16	1.256	1.3	1.719	1.24	19.22	19.7
Denver	1.001	0.22	1.38	0.22	1.156	1.36	1.636	1.3	13.42	15.68
San Antonio	0.751	0.22	1.07	0.22	0.868	1.38	1.252	1.36	13.46	14.51
Dallas	0.676	0.24	0.954	0.26	0.751	1.42	1.081	1.38	9.95	11.8
Houston	0.661	0.22	0.942	0.22	0.722	1.36	1.09	1.36	8.44	13.65
Kansas City	0.741	0.24	1.063	0.24	0.773	1.38	1.143	1.34	4.13	6.97
Minneapolis	0.656	0.28	0.905	0.28	0.656	1.42	0.968	1.38	0.05	6.55
Saint Louis	0.744	0.24	1.039	0.24	0.774	1.38	1.134	1.34	3.87	8.35
Chicago	0.629	0.28	0.868	0.3	0.615	1.4	0.898	1.4	**-2.35**	3.34
Nashville	0.802	0.22	1.117	0.22	0.844	1.36	1.231	1.34	5.03	9.3
Indianapolis	0.707	0.24	0.969	0.26	0.727	1.34	1.061	1.34	2.81	8.67
Atlanta	0.698	0.24	0.977	0.24	0.795	1.4	1.167	1.36	12.2	16.28
Detroit	0.653	0.24	0.891	0.26	0.654	1.38	0.949	1.36	0.12	6.12
Jacksonville	0.64	0.18	0.9	0.18	0.697	1.26	1.048	1.3	8.29	14.13
Charlotte	0.713	0.24	0.991	0.26	0.814	1.42	1.189	1.4	12.33	16.61
Miami	0.514	0.14	0.739	0.16	0.539	1.22	0.829	1.26	4.56	10.85
Pittsburgh	0.706	0.24	0.979	0.26	0.706	1.38	1.038	1.36	**-0.04**	5.67
Toronto	0.593	0.24	0.805	0.26	0.601	1.28	0.862	1.3	1.34	6.63
Philadelphia	0.665	0.26	0.938	0.28	0.695	1.4	1.047	1.36	4.19	10.42
New York	0.575	0.26	0.787	0.28	0.612	1.4	0.877	1.38	6.08	10.28
Montreal	0.657	0.24	0.914	0.24	0.668	1.28	0.982	1.3	1.62	6.9
Boston	0.64	0.26	0.875	0.28	0.642	1.42	0.927	1.38	0.31	5.59
Beersheba	0.718	0.22	1.105	0.2	1.294	1.04	2.206	1.0	44.52	49.91
Tel Aviv District	0.642	0.16	0.962	0.16	0.777	1.2	1.209	1.22	17.41	20.47
Eilat	1.154	0.12	1.662	0.1	1.604	1.04	2.518	0.86	28.03	34.0
Haifa	0.861	0.1	1.356	0.08	0.955	1.06	1.715	1.1	9.79	20.94
Nahariyya	0.851	0.08	1.364	0.06	0.901	1.02	1.739	1.08	5.57	21.57
Jerusalem	0.678	0.16	1.007	0.16	0.79	1.16	1.219	1.16	14.26	17.4

Tab. II *Table provides the best values of MAE and RMSE and the corresponding learning rates. The last two columns represent the difference between OC-NLMS and NLMS error in percents. The rare cases where the NLMS outperform the proposed OC-NLMS are highlighted.*

Both error metrics provide different information, thus both of them are presented in the Tab. II. The diversity of the used data is shown in Fig. 5. As it is possible to see in the Tab. II, the proposed OC-NLMS works generally better than NLMS. Both error metrics (MAE and RMSE) are minimized with similar learning rates in a given algorithm, while a different optimal learning rate is required per algorithm. In terms of RMSE, the proposed OC-NLMS features better robustness to suboptimal selection of learning rate than NLMS (Fig. 6).

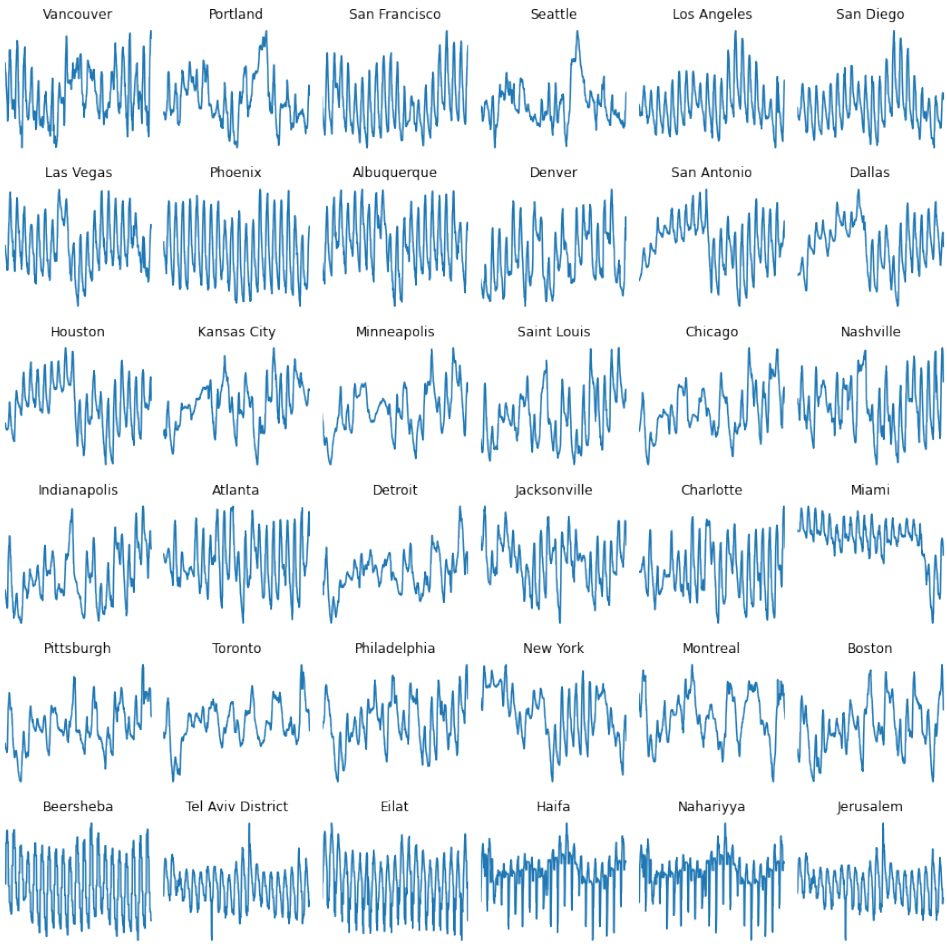

Fig. 5 *Overview of data [1] diversity – last 500 samples (hours) from every series (since 2017-11-09 till 2017-11-30).*

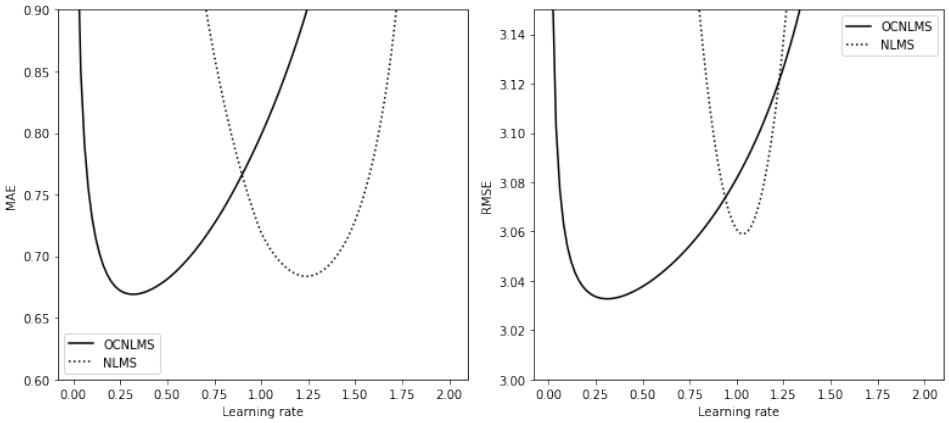

Fig. 6 *Influence of learning rate selection on MAE and RMSE in the prediction of temperature (this plot is for Vancouver temperature time series [1]).*

4. Discussion

4.1 Information loss versus condition number

The experimental results of OC-NLMS algorithm are promising, but it is important to keep in mind that the proposed algorithm causes a small information loss. More complete and diverse tests of general validity should be performed in the future to determine the limitations of the OC-NLMS algorithm. In other words, further research should be conducted to find the border between the importance of full information in the data or better condition number of data.

4.2 Time complexity

In Tab. III is the number of iterations needed by individual algorithms (n is size of input vector and the m is the size of memory). As you can see from this comparison, the OC-NLMS has about one-third more multiplications and significantly more additions. However, an important notion is that the memory size m influence only the additions and it is independent from input size n. Thus, the OC-NLMS time complexity is still linear. A further important notion is that if the OC-NLMS

	Multiplications	**Additions**
NLMS	$3n + 2$	$3n - 1$
OC-NLMS	$4n + 3$	$5n + (n + 1)(m - 1)$

Tab. III *Time complexity.*

is implemented in a higher level programming language (e.g., Python, Java, or similar), it is possible to use a numerical library for such computations. In this case, the difference between NLMS and OC-NLMS reduces to a matrix operation which

is much faster than rest of the computations that must be performed iteratively in loops. The implementation of OC-NLMS used in the experimental analysis in this paper is thus only 10 % slower than NLMS due to this reason.

5. Conclusion

This paper proposed the OC-NLMS algorithm as an extension of the normalized least-mean-squares algorithm. This proposed algorithm uses temporary statistical attributes of the measured data to center the input data, what leads to better performance with gradual concept drift like offsets and/or badly scaled data. The simulation results validate this approach for SISO and MISO system identification and for the prediction of nonstationary data.

Acknowledgement

Authors acknowledge support from the ESIF, EU Operational Programme Research, Development and Education, and from the Center of Advanced Aerospace Technology (CZ.02.1.01/0.0/0.0/16_019/0000826), Faculty of Mechanical Engineering, Czech Technical University in Prague. The used dataset [1] is publicly available at Kaggle:
https://www.kaggle.com/selfishgene/historical-hourly-weather-data/.

References

[1] BENIAGUEV D. *Historical hourly weather data 2012-2017*. 2017. Available also from: https://www.kaggle.com/selfishgene/historical-hourly-weather-data.

[2] CEJNEK M., BUKOVSKY I. Online data centering modifications for adaptive filtering with NLMS algorithm. In: *Neural Networks (IJCNN), 2016 International Joint Conference on*, 2016, pp. 1767–1771.

[3] GHAURI S.A., SOHAIL M.F. System identification using LMS, NLMS and RLS. In: *Research and Development (SCOReD), 2013 IEEE Student Conference on*, 2013, pp. 65–69.

[4] MANDIC D.P. A generalized normalized gradient descent algorithm. *IEEE Signal Processing Letters*. 2004, 11(2), pp. 115–118.

[5] NARENDRA K.S., PARTHASARATHY K. Identification and control of dynamical systems using neural networks. *IEEE Transactions on neural networks*. 1990, 1(1), pp. 4–27.

[6] NIN L., PEREZ-MEANA H., SANCHEZ-SINENCIO E., et al. A modular analog NLMS structure for adaptive filtering. *Analog integrated circuits and signal Processing*. 1999, 21(2), pp. 127–142.

[7] RUMELHART D.E., HINTON G.E., WILLIAMS R.J. Learning representations by back-propagating errors. *Cognitive modeling*. 1988, 5(3), pp. 1.

[8] SHOVAL A., JOHNS D.A., SNELGROVE W.M. Comparison of DC offset effects in four LMS adaptive algorithms. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*. 1995, 42(3), pp. 176–185.

[9] WIDROW B., STEARNS S.D. Adaptive signal processing. *Englewood Cliffs, NJ, Prentice-Hall, Inc., 1985, 491 p.* 1985, 1.

[10] WILBUR W.J., KIM W. Stochastic gradient descent and the prediction of MeSH for PubMed records. In: *AMIA Annual Symposium Proceedings*, 2014, pp. 1198.

[11] YADAV J., KUMAR M., SAXENA R., JAISWAL A. Performance Analyis Of Lms Adaptive Fir Filter And Rls Adaptive Fir Filter For Noise Cancellation. *Signal & Image Processing.* 2013, 4(3), pp. 45.

[12] YOUSEF N.R., SAYED A.H. A unified approach to the steady-state and tracking analyses of adaptive filters. *IEEE Transactions on Signal Processing.* 2001, 49(2), pp. 314–324.