Comparative Analysis of the DRNGD and the NNGD Algorithms in Complex Domain

Igor R. Krcmar, Milorad M. Bozic, and Petar S. Maric
Faculty of Electrical Engineering
University of Banja Luka
Banja Luka, BiH
{ikrcmar, mbozic, pmaric}@etfbl.net

Abstract— **Comparative analysis of the data reusing nonlinear gradient descent and the normalized nonlinear gradient descent algorithms is provided. Starting point of the analysis is linearization of the model of data reusing iterations. Further, this allows application of the Z-transform and analysis of the algorithms in the complex domain. Notion of bandwidth of a neural nonlinear adaptive finite impulse response filter is introduced and relationship between pole placement of the algorithm and filter bandwidth is established. Effects of the output neuron nonlinearity and large bandwidth, on the filter performance, are analyzed. Requirement for a large filter bandwidth leads the output neuron to saturation, thus decreasing overall filter performance. Nonlinear system identification experiments performed on the benchmark nonlinear systems support the analysis.**

Keywords-adaptive filter, bandwidth, complex domain, data reusing nonlinear gradient descent algorithm, normalized nonlinear gradient descent algorithm

I. INTRODUCTION

The least mean squares (LMS) algorithm, due to its robustness and simplicity, is the most frequently used algorithm in system identification and time series prediction tasks. However, when dealing with nonlinear processes, the LMS algorithm can show poor performance [1,2]. Its nonlinear counterpart, the nonlinear gradient descent (NGD) algorithm operated on a neural adaptive finite impulse response (FIR) filter, inherits simplicity and robustness of the LMS algorithm, while it copes with nonlinearities in the process [3]. The dynamics of the NGD algorithm, operated on the nonlinear FIR filter, is described by

$$y(k) = \Phi\left(\mathbf{x}^T(k)\mathbf{w}(k)\right) \tag{1}$$

$$e(k) = d(k) - y(k) \tag{2}$$

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta\Phi'(net(k))\mathbf{x}(k)e(k) \tag{3}$$

where $y(k)$ is the output of the filter, $e(k)$ is the instantaneous error at the output, $d(k)$ is some teaching (desired) signal, $\mathbf{x}(k) = [x_1(k), x_2(k), \ldots, x_N(k)]^T$ is the input vector, $\mathbf{w}(k) = [w_1(k), w_2(k), \ldots, w_N(k)]^T$ is the weight vector, $\Phi(\cdot)$ is nonlinear activation function at the output neuron, N is length of the weight vector and the input vector, η is the learning rate parameter, k is the discrete time instant, $(\cdot)'$ denotes the first derivative and $(\cdot)^T$ denotes vector transpose. The aim of a neural adaptive filter, based on the NGD algorithm, is to achieve optimal weight values by iterating (1) – (3), thus minimising the cost function $J(k) = 1/2\,e^2(k)$.

Regardless inherent nonlinearity, the NGD algorithm can exhibit slow convergence, especially when applied in non stationary environment [4]. Further, the algorithm specified by (1) – (3) is an *a priori* algorithm. The fact, that the updated weight vector $\mathbf{w}(k+1)$ is available before the arrival of the next input vector $\mathbf{x}(k+1)$ and carries a new information on the system, provided by the latest measurement of the system output, $y(k)$, can help to improve error estimation. Thus, an *a posterior* output estimate $\bar{y}(k)$ can be calculated as $\bar{y}(k) = \Phi[\mathbf{x}^T(k)\,\mathbf{w}(k+1)]$. The corresponding *a posterior* output error is given by as $\bar{e}(k) = d(k) - \bar{y}(k)$ and inequality $|\bar{e}(k)| \le \gamma|e(k)|$; $0 < \gamma < 1$, should hold [5,2].

If we use this principle, and iterate (1)-(3), L times on the same measurement data, i.e. input vector $\mathbf{x}(k)$ and teaching signal $d(k)$ are kept constant, we get a data-reusing NGD (DRNGD) algorithm [6,7]. The equations, that describe DRNGD algorithm for nonlinear adaptive FIR filter, are

$$y_i(k) = \Phi(\mathbf{x}^T(k)\mathbf{w}_i(k)) \tag{4}$$

$$e_i(k) = d(k) - y_i(k) \tag{5}$$

$$\mathbf{w}_{i+1}(k) = \mathbf{w}_i(k) + \eta\Phi'(\mathbf{x}^T(k)\mathbf{w}_i(k))\mathbf{x}(k)e_i(k) \tag{6}$$

subject to $|e_{i+1}(k)| \le \gamma|e_i(k)|$; $0 < \gamma < 1$, $i = 1, 2, \ldots, L$. From (6) we have $\mathbf{w}(k+1) = \mathbf{w}_{L+1}(k)$ and $\mathbf{w}(k) = \mathbf{w}_1(k)$.

Normalised GD algorithms for neural adaptive filters have improved performance comparing to the LMS and the NGD algorithm. Improved performance means lower sensitivity to certain filter design parameters, faster convergence, and higher accuracy [8,9,3,4]. It is well known fact, from the control system theory, that system should provide fast and accurate response to the input signal [10,11]. System bandwidth is responsible for the system response, i.e. larger bandwidth provides faster response. Unfortunately, large bandwidth can open a door to disturbances acting on the system, it can put system nonlinearities into the game, and reduce stability margin. Also, requirement for a small steady state error might result in slow response of the system [10,11]. The same holds

for neural adaptive filters and algorithms operating on them, since they form a dynamical system.

Till now, relationship between the data reusing LMS (DRLMS) and normalized LMS (NLMS) algorithms was established and performance of the algorithms was studied and compared [3,6]. Performance of the DRLMS algorithm tends to tone of the NLMS algorithm, as number of DR iterations tends to infinity. There are indices that similar relationship holds for the DRNGD and the NNGD algorithms [3,6,8]. In this paper, we provide comparative analysis of the DRNGD and the NNGD [4] algorithms . The paper is organized as follows. The second section brings analysis of the DRNGD algorithm in complex domain. Based on the analysis performance of the DRNGD and the NNGD algorithms are compared. The third section gives experimental verification of the analysis, while Section IV concludes the paper.

II. Analysis of the DRNGD algorithm in the complex domain

Equations (4)-(6) give the state space description of a discrete time (DT) nonlinear time invariant (NLTI) system. The teaching signal $d(k)$, the weight vector $\mathbf{w}(k)$, and the input vector $\mathbf{x}(k)$ define an operating point of the system. Thus, in the vicinity of the operating point nonlinear model can be substituted with the linear one, obtained through the process of linearization. A Taylor series expansion of the filter output (4) yields

$$\begin{aligned} y_i(k) &= y(k) + \sum_{j=1}^{N} \frac{\partial y_i(k)}{\partial w_i^{(j)}(k)} \Delta w_i^{(j)}(k) \\ &+ \frac{1}{2!} \sum_{l=1}^{N} \sum_{j=1}^{N} \frac{\partial y_i(k)}{\partial w_i^{(j)}(k) \partial w_i^{(l)}(k)} \Delta w_i^{(j)}(k) \Delta w_i^{(l)}(k) + \cdots \\ &= y(k) + \sum_{j=1}^{N} \Phi_k' x_j(k) \Delta w_i^{(j)}(k) \\ &+ \frac{1}{2!} \sum_{l=1}^{N} \sum_{j=1}^{N} \Phi_k'' x_l(k) x_j(k) \Delta w_i^{(j)}(k) \Delta w_i^{(l)}(k) + \cdots \end{aligned} \tag{7}$$

where partial derivatives are computed at the operating point, $\Delta w_i^{(j)}(k)$ denotes j^{th} component of the vector $\Delta \mathbf{w}_i(k) = \mathbf{w}_i(k) - \mathbf{w}(k)$, and $\Phi_k = \Phi(\mathbf{x}^T(k)\mathbf{w}(k))$. A truncated Taylor series of (7) gives

$$y_i(k) \approx y(k) + \sum_{j=1}^{N} \Phi_k' x_j(k) \Delta w_i^{(j)}(k) \tag{8}$$

Now, linear description of the filter output is as follows

$$\Delta y_i(k) = \Phi_k' \mathbf{x}^T(k) \Delta \mathbf{w}_i(k) \tag{9}$$

where $\Delta y_i(k) = y_i(k) - y(k)$. If we subtract $d(k)$ from both sides of (8) we have

$$\begin{aligned} e_i(k) &\approx e(k) - \sum_{j=1}^{N} \Phi_k' x_j(k) \Delta w_i^{(j)}(k) \\ &= e(k) - \Phi_k' \mathbf{x}^T(k) \Delta \mathbf{w}_i(k) \end{aligned} \tag{10}$$

Further, subtraction of $\mathbf{w}(k)$ from both sides of (6) gives

$$\Delta \mathbf{w}_{i+1}(k) = \Delta \mathbf{w}_i(k) + \eta \Phi'(\mathbf{x}^T(k)\mathbf{w}_i(k)) \mathbf{x}(k) e_i(k) \tag{11}$$

If we introduce $e_i(k)$, given by (10), in (11), it becomes

$$\begin{aligned} \Delta \mathbf{w}_{i+1}(k) &= \Delta \mathbf{w}_i(k) - \eta \Phi'(\mathbf{x}^T(k)\mathbf{w}_i(k)) \Phi_k' \mathbf{x}(k) \mathbf{x}^T(k) \Delta \mathbf{w}_i(k) \\ &+ \eta \Phi'(\mathbf{x}^T(k)\mathbf{w}_i(k)) \mathbf{x}(k) e(k) \end{aligned} \tag{12}$$

Under the assumption of a slow weight update, i.e. $\Phi_k' = \Phi(\mathbf{x}^T(k)\mathbf{w}_1(k)) \approx \Phi(\mathbf{x}^T(k)\mathbf{w}_2(k)) \approx \ldots \approx \Phi(\mathbf{x}^T(k)\mathbf{w}_L(k))$, (9), (10), and (12) describe LTI system. Therefore application of the Z-transform [12] on (9) and (12) yields

$$\Delta Y(z) = \Phi_k' \mathbf{x}^T(k) \Delta \mathbf{W}(z) \tag{13}$$

$$z \Delta \mathbf{W}(z) = \left[\mathbf{I} - \eta \left(\Phi_k' \right)^2 \mathbf{x}(k) \mathbf{x}^T(k) \right] \Delta \mathbf{W}(z) + \eta \left(\Phi_k' \right) \mathbf{x}(k) E(z) \tag{14}$$

where $\mathbf{I}$ denotes identity matrix, z denotes complex variable, $\Delta \mathbf{W}(z) = Z[\Delta \mathbf{w}_i(k)]$, $E(z) = Z[e(k)]$, and $\Delta Y(z) = Z[\Delta y_i(k)]$. Also, initial value $\Delta w_0^{(j)}(k) = 0$, $j = 1,2, \ldots, N$. From (14) we have

$$\Delta \mathbf{W}(z) = \left[(z-1)\mathbf{I} - \eta \left(\Phi_k' \right)^2 \mathbf{x}(k) \mathbf{x}^T(k) \right]^{-1} \eta \left(\Phi_k' \right) \mathbf{x}(k) E(z) \tag{15}$$

After application of matrix inversion lemma [10], (15) becomes

$$\Delta \mathbf{W}(z) = \frac{\eta \Phi_k' \mathbf{x}(k) E(z)}{z - 1 + \eta \left(\Phi_k' \right)^2 \left\| \mathbf{x}(k) \right\|_2^2} \tag{16}$$

where $\|\cdot\|_2$ denotes second vector norm. Combining (13) and (16) we have

$$\begin{aligned} \Delta Y(z) &= \Phi_k' \mathbf{x}^T(k) \frac{\eta \Phi_k' \mathbf{x}(k) E(z)}{z - 1 + \eta \left(\Phi_k' \right)^2 \left\| \mathbf{x}(k) \right\|_2^2} \\ &= \frac{\eta \left(\Phi_k' \right)^2 \left\| \mathbf{x}(k) \right\|_2^2}{z - 1 + \eta \left(\Phi_k' \right)^2 \left\| \mathbf{x}(k) \right\|_2^2} E(z) \end{aligned} \tag{17}$$

Now, under assumption of stability of the DRNGD algorithm, we can formulate the following propositions.

Proposition 1. In the limit, as number of data-reusing iterations, L, tends to infinity, the NGD algorithm yields the normalised NGD (NNGD) algorithm.

To prove the proposition note that $e(k)$ is constant during data reusing iterations, thus $E(z) = e(k)z/(z-1)$. Further, stability assumption, together with (16), and the final value theorem of the Z-transform yields

$$\begin{aligned} \Delta \mathbf{w}(k) &= \mathbf{w}(k+1) - \mathbf{w}(k) = \lim_{L \to \infty} \mathbf{w}_{L+1}(k) - \mathbf{w}(k) \\ &= \lim_{z \to 1} (1 - z^{-1}) \Delta \mathbf{W}(z) = \frac{\Phi_k' \mathbf{x}(k) e(k)}{\left(\Phi_k' \right)^2 \left\| \mathbf{x}(k) \right\|_2^2} \end{aligned} \tag{18}$$

The last expression on the right hand side of (18) gives the weight correction of the NNGD algorithm, which completes the proof.

Proposition 2. The NNGD algorithm tends to provide maximum bandwidth for the neural adaptive FIR filter.

If we introduce optimal learning rate $\eta_{OPT} = 1/[(\Phi_k')^2 \| \mathbf{x}(k)\|_2^2]$ [4] in (17), we have

$$\Delta Y(z) = \frac{E(z)}{z} \tag{19}$$

The optimal learning rate places pole of the discrete transfer function, given by (17), to zero. Therefore the NNGD algorithm provides maximum bandwidth of the neural adaptive filter. Now, we shall give several comments on the obtained results.

Comment 1. From (16) and (17) it is clear that the DR NGD algorithm behaves as the first order DT dynamical system. Therefore, the dynamics of the DR iterations is described by the pole $\alpha_{DR} = 1 - \eta(\Phi_k')^2 \| \mathbf{x}(k)\|_2^2$. Further, the DR iterations, under the assumption of slow adaptation, i.e. small value of η, can be considered as a fixed point iteration (FPI) [3]. Thus, effective value of the pole of the DRNGD algorithm is $\alpha_{DRNGD} = (\alpha_{DR})^L = [1 - \eta(\Phi_k')^2 \| \mathbf{x}(k)\|_2^2]^L$ [3]. In order to provide stability of the DR iterations $|\alpha_{DR}| < 1$, and consequently $\lim_{L->\infty} (\alpha_{DR})^L = 0$. So, if number of DR iterations tends to infinity effective value of the pole of the DRNGD algorithm tends to zero. In this way the DRNGD algorithm tends to provide maximum bandwidth of the neural adaptive FIR filter.

Comment 2. Introduction of the optimal learning rate in the DRNGD algorithm yields the fact that the algorithm reaches its final value in a single iteration. Therefore, $\Delta y_i(k)$ becomes $\Delta y(k) = y(k) - y(k\text{-}1)$ and from (19) we have $\Delta y(k) = e(k\text{-}1) = d(k\text{-}1) - y(k\text{-}1)$, and consequently $y(k) = d(k\text{-}1)$. The NNGD algorithm is optimal at each time instant, thus it is optimal on the whole trajectory.

Comment 3. Usually, the optimal learning rate, within the NNGD algorithm, is modified in order to compensate for the linearization errors [3, 4]. Then the learning rate becomes $\eta = 1/[(\Phi_k')^2 \| \mathbf{x}(k)\|_2^2 + C]$, where C appears as the algorithms design parameter and, in most of the applications, takes some small, suitable chosen, positive value. Now, (17) becomes

$$\Delta Y(z) = \frac{\alpha}{z - 1 + \alpha} E(z) \tag{20}$$

where $\alpha = (\Phi_k')^2 \| \mathbf{x}(k)\|_2^2 / [(\Phi_k')^2 \| \mathbf{x}(k)\|_2^2 + C]$. Constant C has to be chosen to provide stability of the NNGD algorithm, i.e. $|1 - \alpha| < 1$, however overall behaviour of the algorithm will be suboptimal.

Comment 4. The fact that the NNGD algorithm tends to provide maximum bandwidth of the neural adaptive FIR filter might jeopardize the stability of the algorithm. Large bandwidth has the task to force the *a posterior* output error to zero, as in the dead beat controller [10]. This fact requires large correction of weights, which might lead the output neuron to saturation. In that case $\alpha = 0$ and pole of the NNGD algorithm equals 1. Therefore, the algorithm becomes unstable. These facts indicate the choice of the value of constant C. It should be chosen neither too small, nor too large, because in either of these cases one can expect problems with stability of the NNGD algorithm.

Comment 5. Similar line of reasoning, as given within the above comment, holds for the DRNGD algorithm. However, it is worth noting that relatively large value of α_{DR} may yield relatively small value of the effective pole value $\alpha_{DRNGD} = (\alpha_{DR})^L$. Therefore, the DRNGD algorithm may provide large correction of weights, through the mechanism of DR iterations, without forcing the output neuron to saturation.

Comment 6. In the case of linear activation function, the whole analysis holds. Further, there is no need for linearization procedure and the assumption on slow weight adaptation can be omitted. Also, there is no need for comments regarding the output neuron saturation. Obtained results then relates to the DRLMS algorithm and the NLMS algorithm.

III. EXPERIMENTAL RESULTS

In order to confirm the analysis system identification experiments were carried out on nonlinear systems. The experiments were performed as Monte Carlo simulations with 100 independent runs. In all the experiments the logistic function $\Phi(x) = 1/(1+\exp(-\beta x))$ was nonlinear activation function of the output neuron and the slope of the logistic nonlinearity was set $\beta = 4$, and the number of DR iterations were taking value from the set $L = \{1, 3, 5, 10, 30, 50, 100\}$. In the first experiment the DRNGD and the NNGD algorithms were applied for system identification of the nonlinear system given by

$$y(k) = 1/(1 + \exp(-\beta \mathbf{x}^T(k)\mathbf{w}_0)) \tag{21}$$

where $\boldsymbol{w}_0 = [-0.09\ 0.34\ -0.16\ 0.11\ 0.14]^T$. The estimator was nonlinear neural adaptive FIR filter of the filter order $N = 5$, inputs to the system identification experiment were realizations of the Gaussian random process with zero mean and unite variance, and the initial value of weights were normally distributed random values, with zero mean and unite variance. The learning rate parameter, of the DRNGD algorithm, was $\eta = 0.01$. The constant C of the NNGD algorithm was $C = 0.1$. Performance measure was the square of the second norm of the weight error vector, $\mathbf{v}(k) = \mathbf{w}_0(k) - \mathbf{w}(k)$. Results of the first experiment were summarized on Fig. 1. and Fig. 2. Fig. 1. shows convergence curves for the DRNGD and the NNGD algorithm, while Fig. 2. presents average of the effective pole value for the applied algorithms. From the Fig. 1. it is obvious that increase in the number of DR iterations, L, improves performance of the DRNGD algorithm, and in the limit the DRNGD algorithm approaches performance of the NNGD algorithm. Fig. 2. shows that effective value of pole of the DRNGD algorithm approaches pole value of the NNGD algorithm as number of DR iterations increases. Further, there is obvious correlation between effective pole placement and performance of the DRNGD algorithm.

In the second experiment, system identification of the nonlinear benchmark system [13], given by

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

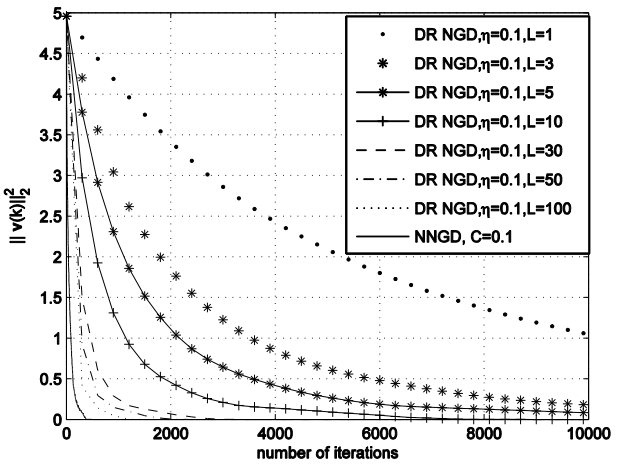

Figure 1. Convergence curves in the first experiment

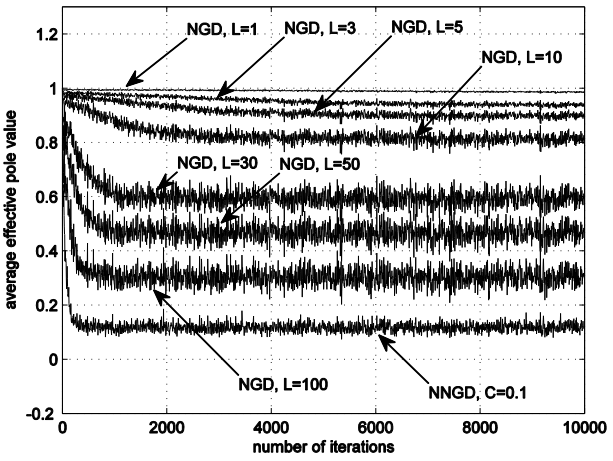

Figure 2. Effective value of the pole of the algorithms in the first experiment

was performed. The applied estimator was nonlinear neural adaptive FIR filter of the order N = 10, the initial value of weights were normally distributed random values, with zero mean and unite variance, the input to the system identification experiment was Gaussian random process with zero mean and unite variance, scaled to fit the range [0.1, 0.8], and the performance measure was the prediction gain, PG = $10\log_{10}(\sigma_y^2/\ \sigma_e^2)$, where σ_y^2 denotes variance of the output prediction and σ_e^2 denotes variance of the output error.

The learning rate parameter, of the DRNGD algorithm, was taking values from the set η = {0.001, 0.01, 0.1, 0.2, 0.3, 0.5}, while the constant C, of the NNGD algorithm, was taking values from the range 0.001 to 100. Fig. 3. shows performance of the NNGD algorithm and an average pole value of the algorithm. Fig. 4. shows performance of the DRNGD algorithm with respect to the values of the learning rate parameter and the number of DR iterations. Further, Fig. 5. brings effective value of the pole of the DRNGD algorithm, while Fig. 6. presents average value of the pole of DR iterations.

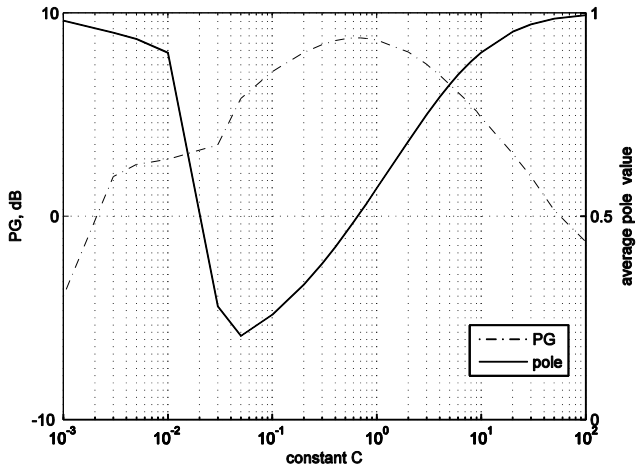

Figure 3. Performance curve and pole value of the NNGD algorithm in the second experiment

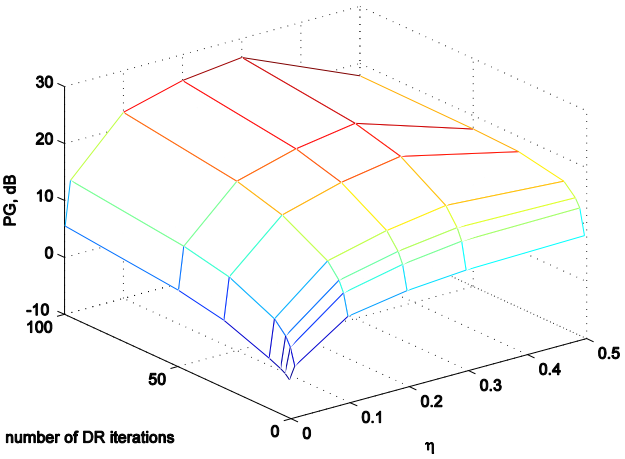

Figure 4. Performance surface of the DRNGD algorithm in the second experiment

From Fig. 3. it is obvious that for small values of the constant C the NNGD algorithm forces the output neuron to saturation, e.g. for C=0.001 average value of the activation of the output neuron was 18.32. Thus, pole of the algorithm becomes close to 1 and the overall behaviour of the estimator is quite poor. Also, performance of the estimator decreases as value of the constant C becomes very large. In this case value of the pole becomes close to 1, but the output neuron does not go to saturation, e.g. for C=100 average value of the activation of the output neuron was 0.93. On the other hand, from Fig. 4. is clear that for certain values of the learning rate and the number of DR iterations the DRNGD algorithm outperforms the NNGD algorithm. For very low value of η and low value of L, the DRNGD algorithm exhibits poor performance, due to the very slow adaptation of weights. In this case, as presented on Fig. 5, the effective pole value is close to 1. In the case of large value of L and relatively large value of η, the effective pole value is close to 0, thus the DRNGD algorithm forces the output neuron to saturation, and overall performance of the

estimator decreases. However, from Fig. 5. and Fig. 6. can be seen that DR mechanism may provide small effective pole value even in case of small value of the learning rate parameter, thus avoiding saturation of the output neuron.

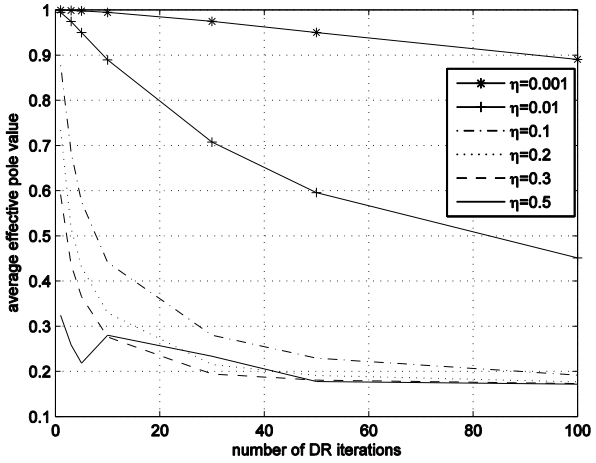

Figure 5. Effective pole value of the DRNGD algorithm in the second experiment

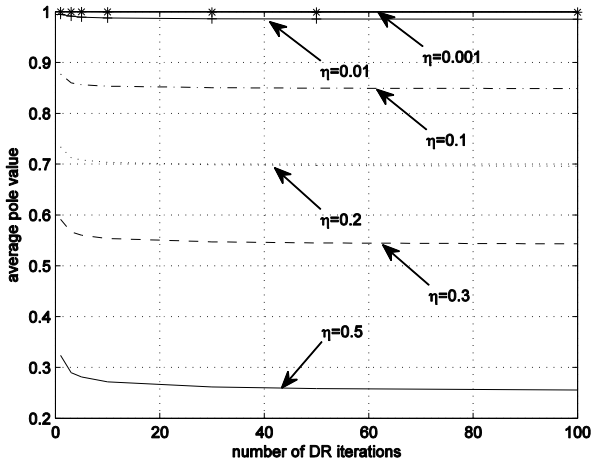

Figure 6. Average pole value of the DR iterations in the second experiment

IV. CONCLUSIONS

The comparative analysis of the DRNGD and the NNGD algorithm in the complex domain has been performed. It has been shown that the DRNGD algorithm, under assumption of slow weight adaptation, i.e. small value of the learning rate parameter, approaches performance of the NNGD algorithm as number of DR iterations tends to infinity. The notion of the neural nonlinear adaptive filter has been introduced and studied. Correlation between the bandwidth of the filter and effective value of the pole of the algorithm has been established. It has been shown that large bandwidth of the filter might force the output neuron of the filter to saturation, thus decreasing performance of the neural nonlinear adaptive filter. An undertaken nonlinear system identification experiments have supported the analysis.

REFERENCES

[1] Simon Haykin, *Adaptive Filter Theory*, 3rd edition, Prentice-Hall, 1996.

[2] J. R. Treichler, C. R. Johnson, and M. G. Larimore, *Theory and design of adaptive filters*, John Wiley and Sons, 1987.

[3] D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures, and Stability*, Wiley Series in Adaptive and Learning Systems for Signal Processing, Communications, and Control, John Wiley & Sons, 2001.

[4] D. P. Mandic,"The NNGD Algorithm for Neural Adaptive Filters," *Electronics Letters*, vol. 36, no. 9, pp. 845-846, 2000.

[5] D. P. Mandic and J. A. Chambers, "Relationships Between the A Priori and A Posteriori Errors in Nonlinear Adaptive Filters," *Neural Computation*, vol. 12, no. 6, pp. 1285-1292, 2000.

[6] J. Benesty and T. Gaensler, "On data-reuse adaptive algorithms," in *Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC2003)*, pp. 31-34, Kyoto, Japan, Sept. 2003.

[7] A. I. Hanna and D. P. Mandic, "A data-reusing nonlinear gradient descent algorithm for a class of complex-valued neural adaptive filters," *Neural Processing Letters*, vol. 17, pp. 85-91, 2003.

[8] S. C. Douglas, "A family of normalized LMS algorithms," *IEEE Signal Processing Letters*, vol. 1, no. 3, pp. 49-51, 1994.

[9] E. Soria-Olivas, J. Calpe-Maavilla, J. F. Guerrero-Martinez, M. Martinez-Sober, and J. Espi-Lopez, " An easy demonstration of the optimum value of the adaptation constant in the LMS algorithm," *IEEE Transaction on Education*, vol. 41, no. 1, pp. 81-83, 1998.

[10] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, *Control System Design*, Prentice Hall, 2000.

[11] A. S. Bazanella, L. Campestrini, and D. Eckhard, *Data-Driven Controller Design, The H2 Approach*, Springer, 2012.

[12] E. Soria, J. Calpe, J. Chambers, M. Martinez, G. Camps, and J. D. Martin Guerrero, " A novel approach to introducing adaptive filters based on the LMS algorithm and its variants," *IEEE Transaction on Education*, vol. 47, no. 1, pp. 127-133, 2004.

[13] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.