Data-Driven Gradient Descent Direct Adaptive Control for Discrete-Time Nonlinear SISO Systems

Igor R. Krcmar, Milorad M. Bozic, and Petar S. Maric

***Abstract*—A novel data-driven gradient descent (GD) adaptive controller, for discrete-time single-input and single output (SISO) systems, is presented. The controller operates as the least mean squares (LMS) algorithm, applied to a nonlinear system with feedback. Normalization of the learning rate parameter provides robustness of the overall system to modeling errors and environment nonstationarity. Convergence analysis reveals that the controller forces tracking error to zero, with bounded control signal and the controller weights. The experiments on the benchmark systems support the analysis.**

***Index Terms*—Data-Driven Controller, Gradient Descent Algorithms, Direct Adaptive Control, Nonlinear SISO Systems, Discrete Time.**

Original Research Paper
DOI: 10.7251/ELS1418039K

I. Introduction

Control means decision making. In order to decide well, one has to have a goal, as well as, information on a controlled process and process environment. Within automatic control setup, the goal specifies desired behavior of the overall system. Information, regarding the process and influence of the environment, is given by the process model and/or process measurements. In many automatic control applications process model is not available. Instead, process measurements contain information on the process behavior. Therefore, many control algorithms were developed to cope with this situation. Usually, they are referred to as data-driven or model-free. This emphasizes situation where control is based, mainly, on the available measurement data. In these situations, concepts like feedback linearization [1], regardless their firm theoretical foundation and good performance, are not applicable. If we look back to the past, probably, the first data-driven control procedure is one proposed by Ziegler and Nichols for tuning proportional-integral-derivative (PID) controllers. However, many data-driven control algorithms were reported in the literature, till now. These are virtual reference feedback tuning (VRFT), iterative feedback tuning (IFT) and data-driven model-free adaptive control (MFAC), to mention a few. The VRFT is reference model adaptive control technique. Within VRFT framework, it is assumed that controller structure is known a-priori and controller parameters are determined by system identification procedure, using virtual reference signal. It is designed for discrete-time single-input and single output (SISO) linear time invariant (LTI) systems. The VRFT procedure may produce controller that results in a non stable closed loop system [2,3]. The IFT, also proposed for an unknown discrete-time SISO LTI system, determines controller parameters through iterative gradient-based local search procedure. Computation of the gradient of criterion function, with respect to the controller parameters, is based on available input/output (I/O) measurement data of the controlled plant [4]. The MFAC approach, in opposition to the VRFT and IFT, was developed for a class of discrete-time nonlinear systems. The main characteristic of the MFAC is utilization of local dynamic linearization data models. The models are computed along the dynamic operation points of the closed-loop system. Computation is performed by the dynamic linearization techniques, with a pseudo-partial derivative concept, based on the real-time I/O measurements of the controlled plant [3]. The above mentioned algorithms, i.e. VRFT, IFT and data-driven MFAC, are direct adaptive control algorithms, as they tune controller without prior plant identification procedure. Further, in applications rich with measurement data, regarding processes with high complexity, neural networks (NNs), with their ability to approximate functions through training procedure [5, 1], might be a good solution. Specially, in nonlinear control tasks, NNs are employed as nonlinear process models and/or nonlinear controllers [6, 7, 1]. However, design of an appropriate NN, meaning choice of an NN structure, input and output vectors, training set and training algorithm, might be a difficult task. Within signal prediction applications, we design a model that should produce, on its output, the desired signal. Usually, the model is parameterized and model parameters are tuned in the real-time based on the incoming process measurements, i.e.

Manuscript received 9 May 2014. Accepted for publication 11 June 2014.

I. R. Krcmar is with the Faculty of Electrical Engineering, University of Banja Luka, Banja Luka, Bosnia and Herzegovina (phone: +387-65-927-212; fax: +387-51-211-408; e-mail: ikrcmar@etfbl.net).

M. M. Bozic is with the Faculty of Electrical Engineering, University of Banja Luka, Banja Luka, Bosnia and Herzegovina (e-mail: mbozic@etfbl.net).

P. S. Maric is with the Faculty of Electrical Engineering, University of Banja Luka, Banja Luka, Bosnia and Herzegovina (e-mail: pmaric@etfbl.net).

adaptively, in order to minimize a certain criterion. The criterion is a function of the error signal, the signal that represents the difference between some desired signal and the model output [8]. Similar setup one can find in the automatic control reference tracking tasks. In the reference tracking tasks, the controller is tuned, so the overall system output tracks the reference signal, as close as possible. Thus, adaptive signal prediction and reference tracking control are similar problems, as they can share the same criterion to be minimized through the parameters correction procedure. For the given parameterization of the process model, different procedures can be applied to achieve optimal value of the model parameters, regarding the imposed criterion. The gradient descent (GD) approach performs search for the optimal value of the parameters, in the parameter space, as it corrects the parameter values in direction opposite to the gradient of the criterion. A common choice for the criterion is a function of the squared error. Models, applied in the signal prediction tasks, with the feedback from the output to the input, introduce nonlinearity into the signal prediction algorithm. Algorithms, based on such models, are referred to as recurrent [8]. The least mean squares (LMS) algorithm, as a member of the GD class, is a simple, yet most frequently used algorithm for signal prediction and system identification. However, it might have a poor performance, when applied to a nonlinear signal and in varying environment [10]. To that cause, a class of normalized algorithms has been developed [12, 8, 9, 10]. They successfully cope with process nonlinearity and nonstationarity, thus provides faster convergence and higher accuracy. The proposed data-driven GD adaptive controller relies on the ideas established within signal prediction framework. It is direct adaptive controller, designed for discrete-time SISO nonlinear system to perform reference tracking. The controller is parameterized, thus provides linear combination of the reference signal and the output signal discrete-time samples. The parameters are tuned, in the real-time, through the GD procedure, to minimize a certain criterion. The criterion is square of the instantaneous output error, therefore the algorithm operates as the LMS algorithm applied to a nonlinear system with feedback. In order to cope with modeling errors and environment nonstationarity, the learning rate is modified as in normalized algorithms [9, 10, 12]. The proposed algorithm has ability of a constant disturbance rejection. Convergence analysis of the proposed algorithm can be performed, following the approach given in [11, 10]. The rest of the paper is organized as follows. In the Section 2, problem formulation and the controller design procedure are given. The Section 3 gives convergence analysis, as well as, analysis of the disturbance rejection ability of the proposed algorithm. The Section 4 contains experimental verification of the algorithm, while the Section 5 concludes the paper.

II. Problem formulation and controller design procedure

It is assumed that the controlled plant is discrete-time nonlinear SISO system, described by the equation

$$\begin{aligned} y(k+1) &= f(y(k), y(k-1), \ldots, y(k-n_y); \\ & u(k), u(k-1), \ldots, u(k-n_u)) = f(k), \end{aligned} \tag{1}$$

where $y(k) \in R$ and $u(k) \in R$ are the system output and input, respectively, n_y and n_u are the unknown orders, and k denotes discrete-time instant. Further, $f(\cdot) \in R$ is some nonlinear function with continuous partial derivatives with respect to its arguments. The controller parameterization is given by the equation

$$\begin{aligned} u(k) &= w_1(k) r(k) + w_2(k) r(k-1) + \ldots \\ &+ w_{n_c}(k) r(k-n_c+1) + w_{n_c+1}(k) y(k) \\ &+ w_{n_c+2}(k) y(k-1) + \ldots w_{2n_c}(k) y(k-n_c+1), \end{aligned} \tag{2}$$

where $r(k) \in R$ denotes the reference signal, $w_i(k); i = 1, 2, \ldots, 2n_c$ are controller parameters, in what follows called weights, and n_c is the controller order. If we introduce the following vectors $\mathbf{w}(k) = [w_1(k), w_2(k), \ldots, w_{2n_c}(k)]^T$, $\boldsymbol{\varphi}_r(k) = [r(k), \ldots, r(k-n_c+1)]^T$, $\boldsymbol{\varphi}_y(k) = [y(k), \ldots, y(k-n_c+1)]^T$, and $\boldsymbol{\varphi}(k) = [\boldsymbol{\varphi}_r^T(k), \boldsymbol{\varphi}_y^T(k)]^T$, where $(\cdot)^T$ denotes vector transpose, the equation (2) can be written in compact form as

$$u(k) = \boldsymbol{\varphi}^T(k) \mathbf{w}(k). \tag{3}$$

The cost function, as the system output should track the reference signal, is defined by

$$J(k) = (1/2) e^2(k), \tag{4}$$

where $e(k)$ denotes instantaneous error at the system output, and it is given as follows

$$e(k) = r(k) - y(k). \tag{5}$$

The weight vector is updated in the real-time according to the equation

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \Delta \mathbf{w}(k). \tag{6}$$

The weights are corrected, through the GD procedure, thus minimizing the cost function (4). Therefore, the weights correction vector is given by

$$\Delta \mathbf{w}(k) = -\eta(k) \nabla_{\mathbf{w}(k)} J(k), \tag{7}$$

where $\eta(k)$ denotes the learning rate and $\nabla_{\mathbf{w}(k)}(\cdot)$ denotes gradient of a scalar function with respect to the weight vector $\mathbf{w}(k)$. In order to compute the weight correction vector (7),

first we have to compute the gradient of the cost function

$$\nabla_{\mathbf{w}(k)} J(k) = \nabla_{\mathbf{w}(k)} \left(\frac{1}{2} e^2(k) \right) = e(k) \nabla_{\mathbf{w}(k)} e(k). \tag{8}$$

To determine the gradient on the right hand side of the equations (8), we have to compute partial derivatives of the error $e(k)$ with respect to the weight $w_i(k)$, where $1 \le i \le 2n_c$

$$\frac{\partial e(k)}{\partial w_i(k)} = -\frac{\partial y(k)}{\partial w_i(k)} = -\sum_{j=1}^{n_c} \frac{\partial f(k)}{\partial y(k-j)} \frac{\partial y(k-j)}{\partial w_i(k)} - \sum_{j=1}^{n_c} \frac{\partial f(k)}{\partial u(k-j)} \frac{\partial u(k-j)}{\partial w_i(k)}. \tag{9}$$

Now we introduce

$$\alpha_j(k) = \frac{\partial f(k)}{\partial y(k-j)}; \beta_j(k) = \frac{\partial f(k)}{\partial u(k-j)}. \tag{10}$$

Combining (9) and (10) we have

$$\frac{\partial e(k)}{\partial w_i(k)} = -\frac{\partial y(k)}{\partial w_i(k)} = -\sum_{j=1}^{n_c} \alpha_j(k) \frac{\partial y(k-j)}{\partial w_i(k)} - \sum_{j=1}^{n_c} \beta_j \frac{\partial u(k-j)}{\partial w_i(k)}. \tag{11}$$

Computation of the partial derivatives, on the right hand side of (11), will be carried out under the assumption of slowly changing weights [13, 8]. Therefore, the following holds

$$\frac{\partial y(k-j)}{\partial w_i(k)} \approx \frac{\partial y(k-j)}{\partial w_i(k-j)}; \frac{\partial u(k-j)}{\partial w_i(k)} \approx \frac{\partial u(k-j)}{\partial w_i(k-j)}. \tag{12}$$

At this point, it is worth noting that the coefficients $\alpha_j(k)$ and $\beta_j(k)$ are time varying and they define the linearized model of the nonlinear system (1), computed along the trajectory that system (1) follows as it advances through time. As we assume the system (1) is unknown, the coefficients $\alpha_j(k)$ and $\beta_j(k)$ can not be determined. Even if we can determine them, from the available measurement data, to represent dominant dynamics of the system (1), the modeling error is still present. To remedy the situation, normalization of the learning rate parameter, as proposed in [8, 9, 10, 11], will be applied. Therefore, we can continue the algorithm derivation by taking into account the assumptions

$$\alpha_1(k) = \alpha, \alpha_j(k) \equiv 0; j = 2, 3, \ldots, n_c, \tag{13}$$

and

$$\beta_1(k) = \beta, \beta_j(k) \equiv 0; j = 2, 3, \ldots, n_c. \tag{14}$$

Now, (11) can be rewritten as follows

$$\frac{\partial e(k)}{\partial w_i(k)} = -\frac{\partial y(k)}{\partial w_i(k)} = -\alpha \frac{\partial y(k-1)}{\partial w_i(k-1)} - \beta \frac{\partial u(k-1)}{\partial w_i(k-1)}. \tag{15}$$

In order to compute partial derivatives $\partial u(k-1)/\partial w_i(k-1)$ we have to refer to the equation (2). Now, we can distinguish two cases. In the first one, the weight $w_i(k-1)$ is connected to the reference signal. Thus, we have

$$\frac{\partial u(k-1)}{\partial w_i(k-1)} = r(k-i) + \sum_{j=1}^{n_c} w_{j+n_c}(k-1) \frac{\partial y(k-j)}{\partial w_i(k-1)} \approx r(k-i) + \sum_{j=1}^{n_c} w_{j+n_c}(k-1) \frac{\partial y(k-j)}{\partial w_i(k-j)}, \tag{16}$$

and in the case the weight $w_i(k-1)$ is connected to the output signal, we have

$$\frac{\partial u(k-1)}{\partial w_i(k-1)} = y(k-i) + \sum_{j=1}^{n_c} w_{j+n_c}(k-1) \frac{\partial y(k-j)}{\partial w_i(k-1)} \approx y(k-i) + \sum_{j=1}^{n_c} w_{j+n_c}(k-1) \frac{\partial y(k-j)}{\partial w_i(k-j)}. \tag{17}$$

Let us introduce the following

$$\pi_i(k) = \frac{\partial y(k)}{\partial w_i(k)}; i = 1, 2, \ldots, 2n_c, \tag{18}$$

and

$$\boldsymbol{\pi}(k) = [\pi_1(k), \pi_2(k), \ldots, \pi_{n_c}(k)]^T, \tag{19}$$

where $\boldsymbol{\pi}(k)$ denotes gradient, with respect to the weight vector, at the system output. Now, the weight update equation becomes

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta(k) e(k) \boldsymbol{\pi}(k), \tag{20}$$

with

$$\boldsymbol{\pi}(k) = \alpha \boldsymbol{\pi}(k-1) + \beta \left(\boldsymbol{\varphi}(k-1) + \sum_{j=1}^{n_c} w_{j+n_c} \boldsymbol{\pi}(k-j) \right). \tag{21}$$

To achieve normalization of the learning rate parameter, first we shall expand the error term (5) into a Taylor series [9]

$$e(k+1) = e(k) + \sum_{i=1}^{2n_c} \frac{\partial e(k)}{\partial w_i(k)} \Delta w_i(k) + \frac{1}{2!} \times \times \sum_{i=1}^{2n_c} \sum_{j=1}^{2n_c} \frac{\partial^2 e(k)}{\partial w_i(k) \partial w_j(k)} \Delta w_i(k) \Delta w_j(k) + \ldots. \tag{22}$$

From (5) and (18) we have

$$\frac{\partial e(k)}{\partial w_i(k)} = -\frac{\partial y(k)}{\partial w_i(k)} = \pi_i(k), \tag{23}$$

while from the weight update we have

$$\Delta w_i(k) = \eta(k) e(k) \pi_i(k). \tag{24}$$

If, for the time being, we neglect higher order terms in the Taylor series expansion (22), as well as, influence of the modeling errors, which comes through the local

gradients $\pi_i(k)$, then the error term becomes

$$e(k+1) = e(k)\left[1-\eta(k)\left\|\boldsymbol{\pi}(k)\right\|_2^2\right], \tag{25}$$

where $\|\cdot\|_2$ denotes the Euclidean norm. The error term in (25) equals zero for

$$\eta(k) = \frac{1}{\left\|\boldsymbol{\pi}(k)\right\|_2^2}. \tag{26}$$

The expression (26) has to be corrected to compensate for the higher order terms in (22), as well as, for modeling errors, thus we have

$$\eta(k) = \frac{1}{C+\left\|\boldsymbol{\pi}(k)\right\|_2^2}. \tag{27}$$

Even though the learning rate (27) provides an optimal solution, it might yield a large bandwidth of the overall system, which in turn may result with the system instability. Therefore, the following learning rate is adjusted according to

$$\eta(k) = \frac{\mu}{C+\left\|\boldsymbol{\pi}(k)\right\|_2^2}, \tag{28}$$

where $0<\mu<1$. Thus, equations (2), (20), (21), and (28) define the proposed algorithm. From the derivation of the algorithm, it is clear that the coefficients α and β, given by (13) and (14), the controller order n_c, the constant C, introduced in (28), as well as, the constant μ are the designed parameters of the algorithm.

III. Convergence analysis and disturbance rejection

The goal of the analysis is to examine whether, the overall system can track the reference signal $r(k)=r^*$, so both signals $u(k)$ and $y(k)$ are bounded. Therefore, the following should hold

$$\lim_{k\to\infty} e(k) = 0 \Rightarrow \lim_{k\to\infty} y(k) = r^*, \tag{29}$$

with $|u(k)|<L_u$ and $|y(k)|<L_y$, where $|\cdot|$ denotes an absolute value, while L_u and L_y are some positive constants. In order to (29) holds, the mapping (25) has to be a contraction, i.e.

$$\left|1-\eta(k)\left\|\boldsymbol{\pi}(k)\right\|_2^2\right| < 1 \tag{30}$$

If inequality (30) holds the equation (25) defines fixed point iteration, thus (29) holds. The inequality (30) can be written as

$$-1 < 1-\eta(k)\left\|\boldsymbol{\pi}(k)\right\|_2^2 < 1. \tag{31}$$

Now, we substitute the learning rate $\eta(k)$, defined by the equation (28), into the inequalities (31), which yields the lower bound on the constant C

$$C > -\frac{2-\mu}{2}\left\|\boldsymbol{\pi}(k)\right\|_2^2. \tag{32}$$

Further, if we refer to the equation (3), we can write

$$\Delta u(k) = \Delta\boldsymbol{\varphi}^T(k)\mathbf{w}(k) + \boldsymbol{\varphi}^T(k)\Delta\mathbf{w}(k). \tag{33}$$

Now, from the weight update equation (20), we have

$$\lim_{k\to\infty} e(k) = 0 \Rightarrow \lim_{k\to\infty} \left\|\mathbf{w}(k)\right\| = 0. \tag{34}$$

To investigate behavior of $\Delta\boldsymbol{\varphi}(k)$ we proceed as follows

$$\begin{aligned} \left|y(k+1)-y(k)\right| &= \left|y(k+1)-r^*+r^*-y(k)\right| \\ &< \left|e(k+1)\right|+\left|e(k)\right| = \left|e(k)\right|\left(1+\varepsilon\right), \end{aligned} \tag{35}$$

where $\varepsilon = \left|1-\eta(k)\left\|\boldsymbol{\pi}(k)\right\|_2^2\right| < 1$. Therefore, the following holds

$$\lim_{k\to\infty} e(k) = 0 \Rightarrow \lim_{k\to\infty} \left|y(k+1)-y(k)\right| = 0, \tag{36}$$

and, as $r(k)=r^*$, we have

$$\lim_{k\to\infty} e(k) = 0 \Rightarrow \lim_{k\to\infty} \left\|\Delta\boldsymbol{\varphi}(k)\right\| = 0. \tag{37}$$

From (33), (34), and (37) we conclude

$$\lim_{k\to\infty} e(k) = 0 \Rightarrow \lim_{k\to\infty} \Delta u(k) = 0, \tag{38}$$

which completes the convergence analysis. If the system is under influence of constant disturbances at its output, then equation (1) can be rewritten as

$$y(k+1) = f(k) + d, \tag{39}$$

where d denotes the disturbance. It is obvious that computation of the partial derivatives (15) is not affected by the disturbance, and the algorithm derivation holds in this case. However, the disturbance affects the error signal, thus it will appear in the Taylor series (22). Thus, if adjust the constant C to compensate the disturbance, the analysis holds.

IV. Simulations

In order to investigate performance of the proposed algorithm, the controller was applied, in a reference tracking task, on three benchmark processes. In all experiments the parameters in the equation (15) were set to $\alpha=0$ and $\beta=1$, while the learning rate parameter in (28) was set to $\mu=0.1$. Further, the order of the controller, n_c, was varied from 1 to 15, and values of the constant C were from the set $\{0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9, 2.1, 2.3, 2.5, 3, 4, 5, 10\}$. The averaged squared error was adopted as the performance measure.

A. Experiment 1

In the first experiment, the controller was applied to the process [7] given by the equation

$$y(k+1)=\sin(y(k))+u(k)(5+\cos(y(k)u(k))) \quad (40)$$

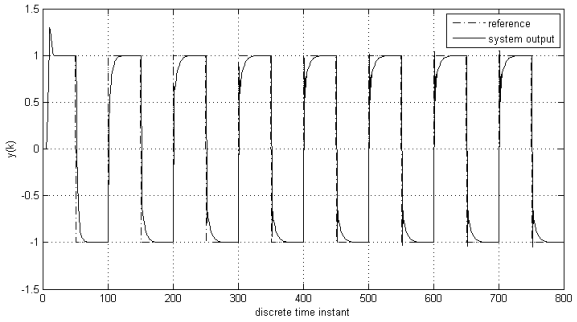

Fig. 1.a Tracking performance of the proposed algorithm in the Experiment 1.

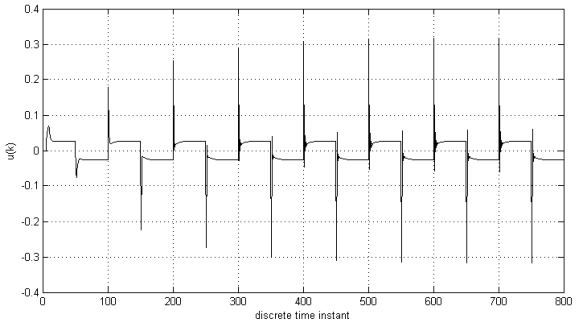

Fig. 1.b Control signal in the Experiment 1.

Figures 1.a to 1.c gives tracking performance, control signal, and error signal of the proposed algorithm, respectively, applied to the system (40), with parameters $C=0.5$ and $n_c=3$. Value of the performance measure was 0.1059.

Figure 1.d gives performance surface in the experiment 1. Minimum of the performance measure was 0.09166, and it was attained for $n_c=1$ and $C=3$.

B. Experiment 2

In the second experiment, the controller was applied to the process [6] given by the equation

$$y(k+1)=\frac{y(k)}{1+y^2(k)}+u^3(k) \quad (41)$$

Figures 2.a to 2.c shows tracking performance, control signal, and error signal of the proposed algorithm, respectively, applied to the system (41), with parameters $C=0.5$ and $n_c=3$. The performance measure was 0.13196.

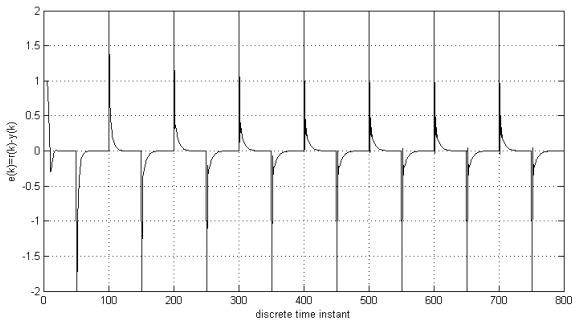

Fig. 1.c The error signal in the Experiment 1.

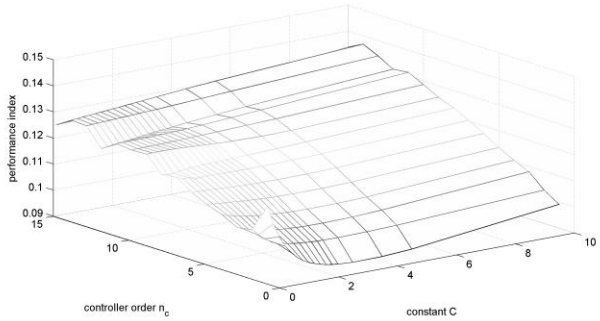

Fig. 1.d Performance index in the Experiment 1.

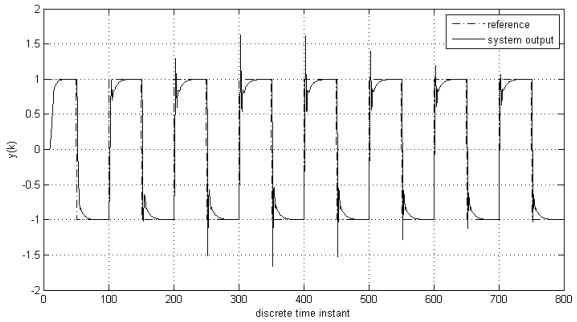

Fig. 2.a Tracking performance of the proposed algorithm in the Experiment 2.

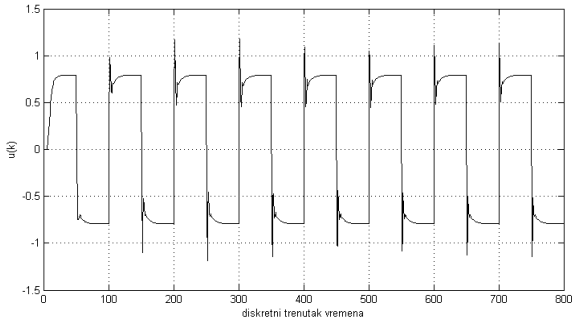

Fig. 2.b Control signal in the Experiment 2.

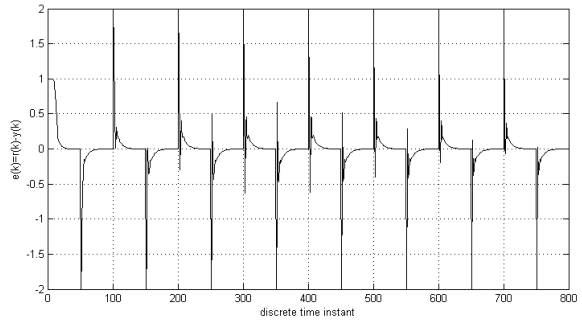

Fig. 2.c The error signal in the Experiment 2.

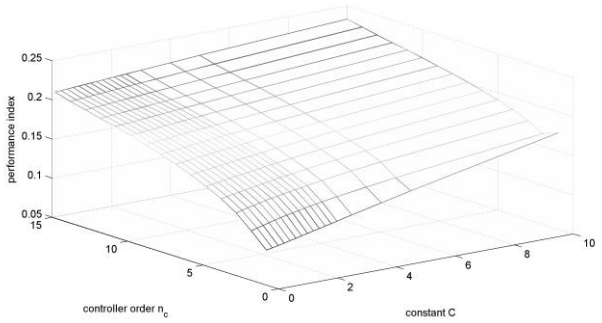

Fig. 2.d Performance index in the Experiment 2.

Figure 2.d gives performance surface in the experiment 2. A minimal value of the performance measure was 0.09296, and it was attained for $n_c = 1$ and $C = 0.1$.

C. Experiment 3

The third experiment was designed to investigate disturbance rejection property of the proposed algorithm. Due to that cause, the controller was applied to the process [6] given by the equation

$$y(k+1) = \frac{y(k)y(k-1)(y(k)+2.5)}{1+y^2(k)+y^2(k-1)} + u(k) \tag{42}$$

Also, at the discrete time instant $k = 370$, the step disturbance of the amplitude $d = 0.2$ was applied to the system output. In this case, a minimal value of the performance measure was 0.02527, and it was attained for $n_c = 1$ and $C = 0.1$. Figures 3.a to 3.c gives tracking performance, control signal, and error signal of the proposed algorithm, respectively, with parameters $C = 0.1$ and $n_c = 1$.

Figure 3.d shows a performance surface in the experiment 3. The minimal value of the performance measure was attained for $n_c = 1$ and $C = 0.1$.

V. Conclusions

The new data-driven GD adaptive controller, for discrete-time nonlinear SISO systems, has been proposed. It was designed for the reference tracking tasks. The controller has simple structure, i.e. it is linear combination of the reference signal and the output signal discrete-time samples. Parameters

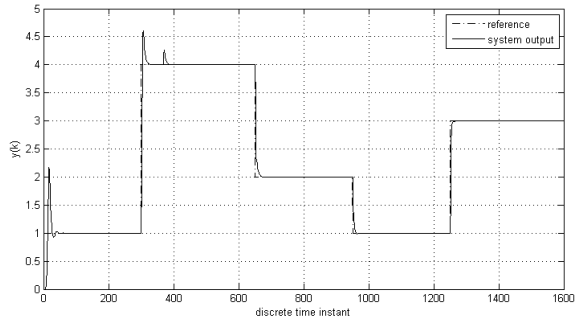

Fig. 3.a Tracking performance of the proposed algorithm in the Experiment 3.

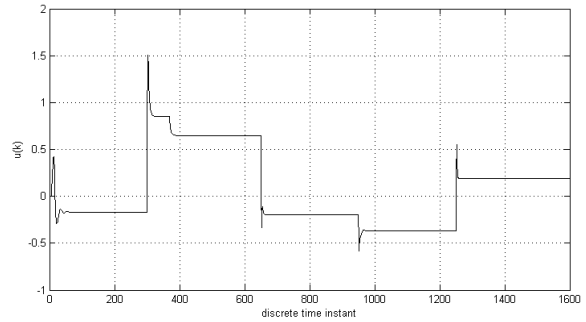

Fig. 3.b Control signal in the Experiment 3.

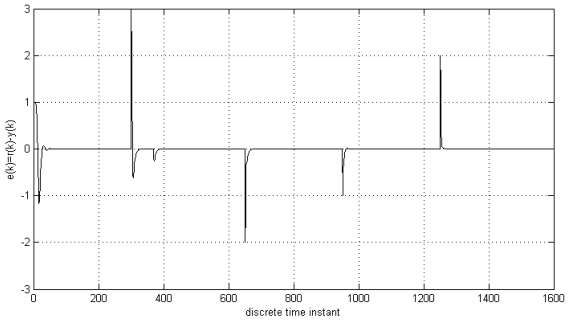

Fig. 3.c The error signal in the Experiment 3.

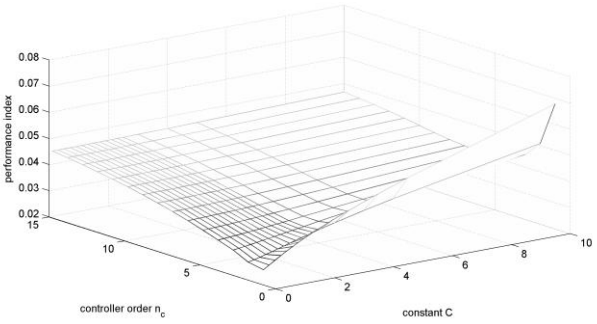

Fig. 3.d Performance index in the Experiment 3.

of the controller are updated in the real-time, as in the case of the LMS algorithm applied to a nonlinear system with feedback. The algorithm is robust to the modeling errors and environment nonstationarity, due to the learning rate parameter normalization. The convergence analysis has been provided. The controller forces tracking error to zero with bounded control signal and controller weights. The experiments, on the

benchmark systems, demonstrate performance of the proposed controller and support the analysis.

REFERENCES

[1] Jeffrey T. Spooner, Manfredi Maggiore, Raul Ordonez, and Kevin M. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximator Techniques*, New York: John Wiley & Sons, Inc., 2002.

[2] M. C. Campi, A. Lecchini, and S. M. Savaresi, "Virtual reference feedback tuning: a direct method for the design of feedback controllers," *Automatica*, vol. 38, pp. 1337-1346, 2002.

[3] Zhongsheng Hou and Shangtai Jin, "Data-Driven Model-Free Adaptive Control for a Class of MIMO Nonlinear Discrete-Time Systems," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 2173-2188, Dec. 2011.

[4] H. Hjalmarsson, S. Gunnarsson, and M. Gevers, "Iterative feedback tuning: Theory and applications," *IEEE Control System Magazine*, vol. 18, no. 4, pp. 26-41, Aug. 1998.

[5] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Englewood Cliffs, NJ: Prentice-Hall, 1999.

[6] K. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, pp. 4-27, Jan. 1990.

[7] Hua Deng, Han-Xiong Li, and Yi-Hu Wu, "Feedback-Linearization-Based Neural Adaptive Control for Unknown Nonaffine Nonlinear Discrete-Time Systems," *IEEE Transactions on Neural Networks*, vol. 19, no. 9, Sep. 2008.

[8] D. Mandic and J. Chambers, *Recurrent Neural Networks for Prediction*, New York: Wiley, 2001.

[9] D. P. Mandic, "NNGD algorithm for neural adaptive filters," *Electronics Letters*, vol. 39, no. 6, pp. 845-846, 2000.

[10] Danilo P. Mandic, "A Generalized Normalized Gradient Descent Algorithm," *IEEE Signal Processing Letters*, vol.11, no. 2, pp. 115-118, Feb. 2004.

[11] D. P. Mandic and I. R. Krcmar, "Stability of NNGD algorithm for nonlinear system identification," *Electronics Letters*, vol. 37, no. 3, pp. 200-202, 2001.

[12] Danilo P. Mandic and Jonathon A. Chambers, "A normalised real time recurrent learning algorithm," *Signal Processing*, vol. 80, pp. 1909-1916, 2000.

[13] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," Neural Computations, vol. 1, no. 3, pp. 270-280, 1989.