БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ . BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, 51
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, 51

София . 2001 . Sofia

A Recurrent Neural Network Approach for Identification and Control of Nonlinear Objects

Ieroham Baruch, Jose Martin Flores Albino**, Boyka Nenkova**

** Institute of Information Technologies, 1113 Sofia*
*** CINVESTAV – IPN, Mexico*

1. Introduction

The Neural Network (**NN**) modelling and application to system identification, prediction and control was discussed for many authors [1–5]. Mainly, two types of **NN** models are used: Feedforward (**FFNN**) and Recurrent (**RNN**). The main problem here is the use of different **NN** mathematical descriptions and control schemes, according to the structure of the object model. For example, N a r e n d r a and P a r t h a s a r a t h y [1, 2], applied **FFNN** for system identification and direct model reference adaptive control of various non-linear objects. They considered four object models with a given structure and supposed that the order of the object dynamics is known. Y i p and P a o [3] solved control and prediction problems by means of a flat-type functional **FFNN**, used for direct inverse model learning control. P h a m [4] applied Jordan **RNN** for robot control. S a s t r y [5] introduced two types of neurones – Network Neurones and Memory Neurones to solve identification and adaptive control problems, considering that the object model is also autoregressive one. In [7], some schemes of **NN** and **RNN** applications to control, especially of direct model reference adaptive control, are surveyed. All drawbacks of the described in the literature **NN** models could be summarised as follows: there exists a great variety of **NN** models and a universality is missing [1–5]; all **NN** models are sequential in nature as implemented for systems identification. (The **FFNN** model uses one or two tap-delays in the input, [1, 2] and **RNN** models usually are based on the autoregressive model [5], which is one-layer sequential one); some of the applied **RNN** models are not trainable, others are not trainable in the feedback part [4]. Most of them are dedicated to a **SISO** and not to a **MIMO** applications [3]; in more of the cases, the stability of the **RNN** is not considered, [4], especially during the learning; in the case of **FFNN** application for systems identification, the object is given in one of the four described in [1] object models, the linear part of the object model, especially the system order, has to be known and the **FFNN** approximates only the non-linear part of the object model [1]; all these **NN** models are nonparametric ones [7], and so, not applicable for an indirect adaptive control systems design; all this **NN** models does not perform state and parameter estimation in the same time [7].

The major disadvantage of all this approaches is that the identification **NN** model applied is a nonparametric one, that does not permit them to use the obtained information directly for control systems design objectives. Baruch et all [6], in their previous paper, applied the state-space approach to describe **RNN** in an universal way, defining a Jordan canonical two- or three-layer **RNN** model, named Recurrent Trainable Neural Network (**RTNN**). This **NN** model is a parametric one, permitting to use the obtained during the learning parameters for control systems design. Furthermore, the **RTNN** model is a system state predictor/estimator, which permits to use the obtained system states directly for state-space control. The aim of this paper is to use the **RTNN** as an identification and state estimation tool in an indirect adaptive control system of nonlinear objects. The proposed Indirect Neural Adaptive Control System (**INACS**) is studied by means of various non-linear, discrete-time dynamic objects. Three types of object models are suggested: models, non-linear on their output, state and input. Simulation examples of nonlinear objects, controlled by the proposed **INACS**, are given.

2. Systems identification and state estimation using the RTNN model

The existing problem is to identify the discrete-time nonlinear object and simultaneously to estimate its states by means of a **RTNN**. The stable multivariable object under test is described by the following discrete-time equation:

(1) $$d(k) = F\,[d(k-1),\, d(k-2), \ldots,\, u(k-1)],$$

where $d(k)$ is a $\boldsymbol{l}$-output vector; $u(k-1)$ is a m-input vector and $F(.)$ is a smooth single l-dimensional vector-valued function. The described in [6] discrete-time, two-layer Jordan canonical **RTNN** architecture, improved with an additional stability preserving restriction on the feedback weight matrix of the hidden layer, is given in the form:

(2) $$X(k+1)=JX(k)+BU(k),\ Z(k)=S[X(k)];\quad Y(k)=S[CZ(k)];\quad |J_i| < 1,$$

where the two layer equations of **RTNN** are separated by semicolon; Y, X, U are output, state and input vectors with dimensions $l, n, m,$ respectively; J = block-diag(J_i) is a ($n \times n$)-block-diagonal weight matrix; J_i are blocks of J with (1×1) or (2×2) dimensions, where the last equation of (2) is a stability condition, imposed on all blocks J_i of $J,$ which is in fact restriction, imposed on the discrete-time system eigenvalues of the state matrix $J;$ B and C are ($n\times m$) and ($l\times n$)- weight matrices; $S(x)$ is a vector-valued activation function, given as:

(3) $$S'(\text{inp}) = [s(\text{inp}_1),\, s(\text{inp}_2), \ldots,\, s(\text{inp}_j),\, s(\text{inp}_n)]$$

Here inp is the input variable of the activation function $\boldsymbol{s}(\text{inp}_j)$, which is an element of the vector function $S.$ The activation functions in use are the sigmoid and the saturation function, given by the equations:

(4) $$s(\text{inp}) = 1/[1+\exp(-\text{inp})];\quad \text{inp}=\Sigma\,(w_i x_i + w_{io});\qquad \text{sat}(\text{inp}) = \begin{cases} +1, & \text{inp}\geq +1, \\ \text{inp}, & 0\leq \text{inp}\leq +1, \\ 0, & \text{inp}<0, \end{cases}$$

where w_0, w_{i0} are trainable weights of the **RTNN**; S' signifies a vector transpose of $S.$ The saturation function is used as approximation of the sigmoid function in the forward step of learning to improve the **RTNN** architecture, facilitating its realisation. The given above **RTNN** model could be linearised and its dynamic behaviour could be studied by the first stability law of Liapunov, where the stability condition (2), for the discrete time state-space model, could be used. Than it is easy to analyse the **NN** model controllability, observability and identifiability. From the block structure of **B** and **C**, corresponding to the block structure of **J**, we can conclude if the **RTNN** could be learned or not, [6]. The main advantage of the proposed two layer Jordan Canonical Form (**JCF**) **RTNN** architecture is that the described **JCF RTNN** model is an universal hybrid neural model which contains one output **FF** layer and one recurrent hidden layer with completely decomposed dynamics, as the matrix **J** is block-diagonal. So, it has a minimum number of parameters and it is completely parallel, as the Jordan canonical form is parallel with respect to the regressive model, which is a sequential one. The **RTNN** architecture is described in state-space form (**SISO** or **MIMO**) and could serve as an nonlinear dynamic identificator and one-step ahead state predictor/estimator. The **RTNN** model is non-linear in large and linear in small, so the matrices *J, B, C,* obtained as a result of learning, could be used for analytical design of linear state/output feedback/feedforward control laws. Finally, the **RTNN** could solve the optimal control problem itself by means of a **NN** mapping, [1]. The obtained **RTNN** model is a robust model, as the learning method applied for weights adjustment is a dynamic **BP** method, which is based on the network sensitivity model, [1]. It also permits to perform node pruning and weight fixing during the learning, which permits to obtain the simplest low order dynamic model approximation of the nonlinear object dynamics.

The identification task could be performed, minimising the instantaneous squared error between the object and **RTNN** outputs with respect to the **RTNN** weights. The cost function is given in the form:

(5) $$\xi(k) = (1/2)\Sigma\ [\,di(k) - yi(k)]^2;\ i \in C,$$

where $di(k)$ and $yi(k)$ are the desired response (the object output) and the actual response (the **RTNN** output) of the output $\boldsymbol{i}$ in the discrete-time moment $\boldsymbol{k}$, respectively, and the set *C* includes all *l* output neurones of the **RTNN**.

The most common used **BP** updating rule, applied for the two-layer **RTNN** canonical model, [6], is the following:

(6) $$W_{ij}(k+1)=W_{ij}(k) + \eta\ \Delta W_{ij}(k),$$

where W_{ij} is a general weight, denoting the $\boldsymbol{ij}$-th weight element of each weight matrix (*C, J, B*) in the **RTNN** model to be updated; ΔW_{ij} , ($\Delta \boldsymbol{C}_{ij}$, ΔJ_{ij}, ΔB_{ij}), is the weight correction of W_{ij}; *h* is the learning rate parameter. If there are some oscillations in the error during the learning, a momentum term can be added in the weight updating rule (6). One of the following two forms can be used:

(7) $$W_{ij}(k+1)=W_{ij}(k) + \eta\,\Delta W_{ij}(k) + \alpha\,\Delta W_{ij}(k-1),$$

$$W_{ij}(k+1)=W_{ij}(k) + \eta(1-\alpha)\ \Delta W_{ij}(k) + \alpha\,\Delta W_{ij}(k-1),$$

where α is a momentum term learning rate parameter. The second equation of (7) establishes some inverse dependence between the learning parameters of the first and second updating terms. The weight corrections of the updated matrices in the discrete-time **RTNN** model, described by equations (1), (2), are performed using the following weight update equations:

– For the output layer

(8) $$\Delta C_{ij}(k) = [T_j(k) - Y_j(k)]\ Y_j(k)\ [1 - Y_j(k)]\ Z_i(k),$$

where ΔC_{ij} is the weight correction of the ij-th elements of the $(l \times n)$ learned matrix C; T_j is a j-th element of the target vector; Y_j is a j-th element of the output vector; Z_i is an i-th element of the output vector of the hidden layer.

– For the hidden layer

(9) $\Delta B_{ij}(k) = RU_i(k)$; $\Delta J_{ij}(k) = R\,X_i(k-1)$; $R = C_i(k)\ [T(k)-Y(k)]\ Z_j(k)[1-Z_j(k)]$,

where ΔB_{ij} is the weight correction of the ij-th elements of the $(m \times n)$ learned matrix B; C_i is a row vector of dimension $(1\times\ l)$, taken from the transposed matrix C'; $[T-Y]$ is a $(l\ \text{x}1)$ output error vector, through which the error is back-propagated to the hidden layer; U_i is an i-th element of the input vector U; X_i is an i-th element of the vector X; ΔJ_{ij} is the weight correction of the ij -th elements of the $(n\times\ n)$ block-diagonal matrix J under learning; R is an auxiliary variable. The applicability of the **RTNN** model for system identification, prediction and control is illustrated by an appropriate example of nonlinear dynamic system.

3. Indirect adaptive neural control system design

In [8], two **FFNNs** are used for self-tuning of **PID** control system. The first **NN** is used for object identification and an input error estimation and the second one is used to tune the **PID** controller parameters. The great complexity of this control scheme is evident. In this paper we propose to use all information that gives the **RTNN** parameter and state estimator for state control systems design.

Let us consider the linearised state-space model of the discrete-time **RTNN**, to be given in the form:

(10) $$X(k+1) = JX(k)+BU(k); \qquad Y(k) = CX(k).$$

The given state space model could be transformed into Luenberger's canonical form [9], as it is:

(11) $V(k+1) = AV(k) + H_1 U_1(k)$; $U_1(k) = H_2U(k)$; $Y(k) = \Delta V(k)$; $V(k) = TX(k)$;

(12) $$A = T^{-1}\ J\ T;\ H = T^{-1}B = H_1 H_2\ ;\ D = C\ T,$$

where T is a nonsingular transformation matrix, [9], and the matrices A, H_1, H_2, D are given in the form:

(13) $$A = \begin{bmatrix} A_{11} A_{11} & \ldots. & A_{1m} \\ A_{11} A_{11} & \ldots. & A_{1m} \\ .. & & . \\ .\ . & & . \\ A_{m1} A_{m2} & \ldots. & A_{mm} \end{bmatrix}\ ;\ A_i = \begin{bmatrix} 0 & l_\sigma \\ \alpha_{ii} & \end{bmatrix};\ A_i = \begin{bmatrix} 0 \\ \alpha_{ij} \end{bmatrix}\quad ;$$

(14) $$H_1 = \begin{bmatrix} H_{11} \\ H_{12} \\ .\ . \\ .\ . \\ H_m \end{bmatrix}\ ;\ H_1 = \begin{bmatrix} 0 \\ h_1 \end{bmatrix};\ \mathrm{A}_i = \begin{bmatrix} 1 & b_{12} \ldots. & b_{1m} \\ 0 & 1 \ldots. & b_{2m} \\ .. & & . \\ 0 & 0 \ldots. & 1 \end{bmatrix};$$

The correspondent blocks of the matrices A, H_1 so as the matrix H_2 have following dimensions:

(15) $$\alpha_{ij} = \left|\ \alpha_{\sigma_{i-1}+1}, \ldots, \alpha_{\sigma_i}\ \right|\ ; \qquad \alpha_{ij} = \left|\ \alpha_{\sigma_{i-1}+1}, \ldots, \alpha_j\ \right|\ ;$$
$$h_i\ =\ \left|0\ \ 0\ \ 0\ \ 1, \ldots, 0\right|; i\text{-th position.}$$

The controllability indices σ_i are determined from the number of linearly independent columns of the controllability matrix $\Gamma = | B, JB, J^2B, \ldots, J^{n-1}B |$, associated to the j-th column of the control matrix B (the j-th control input $Uj(k)$). So the numbers of the significant rows of the matrix A are determined as it follows:

(16) $$\gamma = \sum_{j=1}^{i} \sigma_j, j = 1, \ldots, m.$$

Now, it is possible to design a pole-assignment control using the given in [10] methodology. Let us to introduce a state state feedback to the system, given by the equation (11), as it is:

(17) $$U_1(k) = -GV(k),$$

where G is a $(m\times n)$ gain matrix. The closed-loop state equation (11) obtains the form:

(18) $$V(k+1) = (A - H_1 G)V(k) = A_c V(k)$$

where A_c is a $(n\times n)$ state closed loop matrix with the same structure, as the open loop matrix A, [10], which significant rows form the matrices b, a with dimensions $(m\times n)$. Following [10], we can assign a block- diagonal structure of the closed loop state matrix Ac and to choose the significant rows of each diagonal block βii with dimension $(1\times\sigma i)$ to be coefficients of an assigned stable characteristic polynomial of order σi with prescribed dynamics. Then it is possible to write the following expression for the correspondent blocks of the gain matrix and the correspondent control:

(19) $$\beta_i = \alpha_i - G_i; \; G_{ii} = \alpha_{ii} - \beta_{ii}; \; G_{ij} = \alpha_{ij}; \; U(k) = -K X(k); \; K = H_2^{-1} G T,$$

where β_i, α_i, G_i are i-lines of the correspondent matrices; G_{ii}, α_{ii}, β_{ii}; are $(1\times\sigma i)$ diagonal blocks of the correspondent matrices; G_{ij}, α_{ii} are off diagonal blocks of correspondent matrices; K is gain matrix of the original system, T and H_2 are state and control transformation matrices, respectively. In the particular case of dead-beat control, the $(m\times n)$ matrix $\beta = 0$, so the gain matrix $G = \alpha$.

Another control system design approach is the state optimal control system design, minimising a quadratic cost criterion. The relation between this approach and the pole assignment control system design could be find in [11]. There are some optimal control system design method for synthesis of state optimal **P**, **PD**, **PI** and **PID** controllers, given in [12]. The case of **P** control system design has been considered above. Now, the case of **PI** control system design will be considered. Here we shall apply the same pole assignment method, by means of system equation expansion adding another vector- matricial equation to system state equation (17). This equation is the discrete-time integral of system output (18), which could augment the state equation of the system, as it is:

(20) $$X_I(k+1) = T_0 Y(k) + X_I(k) = To\, CX(k) + X_I(k); \; X_a(k+1) = A_1 X_a(k) + B_1 U(k),$$

where $X_I(k)$ is a l-vector of the integral term and T_0 is a period of discretization; $X_a(k)$ is an $[(n+l)\times(n+l)]$ augmented state vector, and the state and control matrices A_1, B_1 of the augmented system have appropriate dimensions. For this augmented system, following the same procedure, it is easy to obtain the corresponding control.

One very important particular case is when we have a **SISO** system and the matrix J is considered to be a diagonal one. Then the correspondent state and control matrices have the form:

(21) $$J = \mathrm{diag}\,(J_i); \quad B' = |b_1, b_2, \ldots, b_m|'; \quad C = | c_1, c_2, \ldots, c_l |.$$

The state and control matrices of the transformed in controllable companion canonical form system for this case could be obtained from (13) for $m=l=1$. The elements of a could be obtained using the well known Leverier-Fadeeva's algorithms, which for diagonal matrix J leads to the following simple recurrent formulae

(22) $$\alpha_{n-k} = \frac{1}{k} \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \alpha_{n-j+1} J_i^{k-j+1} \right); k = 1, 2, \ldots, n.$$

The same expression could be used to compute the coefficients of the closed-loop state matrix β from the desired systems poles, to be assigned. **PI** control system design could be resolved in the same manner augmenting the system with the integral equation, as it was done above. The dead-beat control system design could be done supposing $\beta = 0$.

Another particular case is the **SISO** object set point tracking system control, [13]. In this case of n=1, the system control law has the form:

(23) $$u(k) = (cb)^{-1} \, [-cA \, x(k) + y^{d}(k) + \alpha_1 \, (y^{d}(k+1) - y(k+1))],$$

where $y^{d}(k)$ is the output set-point to track.

Simulation results of an indirect **P**- state and set point tracking indirect adaptive control of nonlinear object, are given in the next part.

4. Simulation results

In this section we show some simulation result of an indirect adaptive control of nonlinear object. The nonlinear **SISO** object is described by the following equations [5]:

(24) $$y(k+1) = [x_1 x_2 x_3 x_5 (x_3 - 1) + x_4] / [1 + x_2^2 + x_3^2];$$
$$x_1 = y(k),\ x_2 = y(k-1),\ x_3 = y(k-2),\ x_4 = u(k),\ x_5 = u(k-1).$$

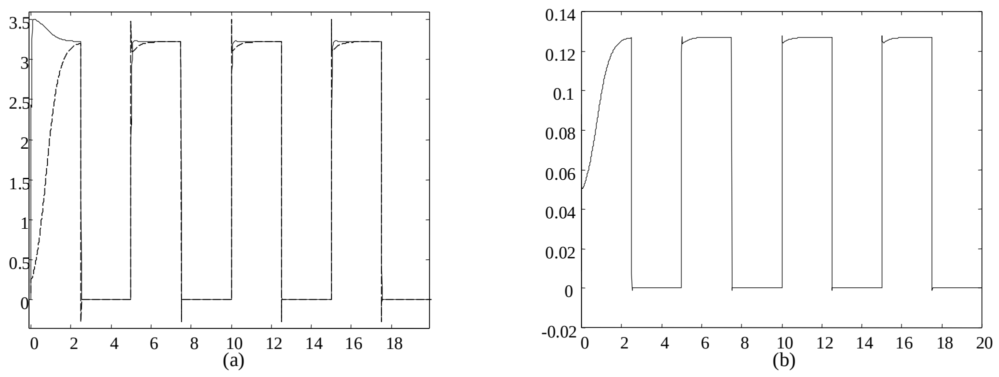

Fig. 1. Continuous line – output of the closed-loop state controlled object; dashed line – RNN output; **RTNN** topology (1, 2, 1), T_0=0.01 s, $h = a = 0.5$, MSE % for t=20 s =1.5% (a); continuous line – control variable of the object (b)

Fig. 1 represents the results of an P state indirect adaptive control of the nonlinear object, described by equations (24), during last 20 iterations of learning. Note that the overshoot in the beginning of the graphics is caused by the poor **RNN** identification, but in few instants of time the **RNN** improved its learning and the control goes better.

For the special case of a **SISO** set-point tracking control system design, we use the technique, describe in [13] where the control is given by the equation (24) and the simulation results are given on Fig.2, and Fig.3, for different set-point signals.

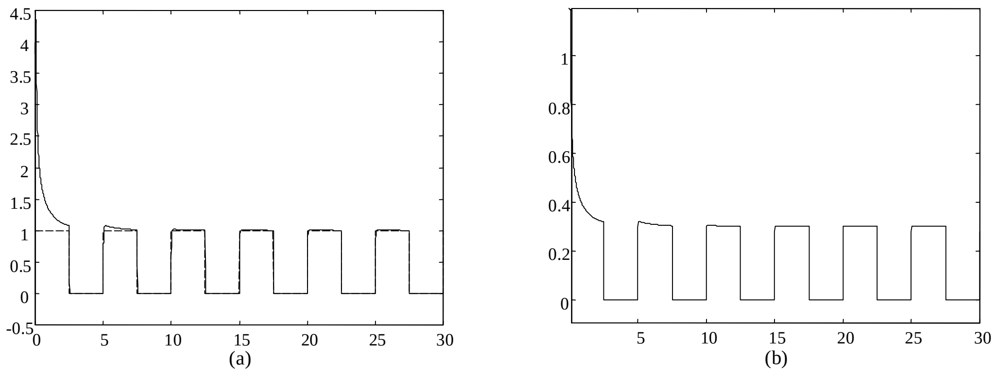

Fig. 2. Continuous line - output of the closed-loop state controlled object; dashed line - **RNN** output,. **RTNN** topology (1, 1, 1), T_0=0.01 s, $h = a =$ 0.5, **MSE %** for t=30 s = 0.017% (a); continuous line – control variable of the object (b)

Fig. 2 represents the results of a set-point tracking indirect adaptive control of the nonlinear object, described by equations (24), during last 20 iterations of learning. Note that the overshoot in the beginning of the graphics is caused by the poor **RNN** identification, but in few instants of time the **RNN** improved its learning and the control goes better.

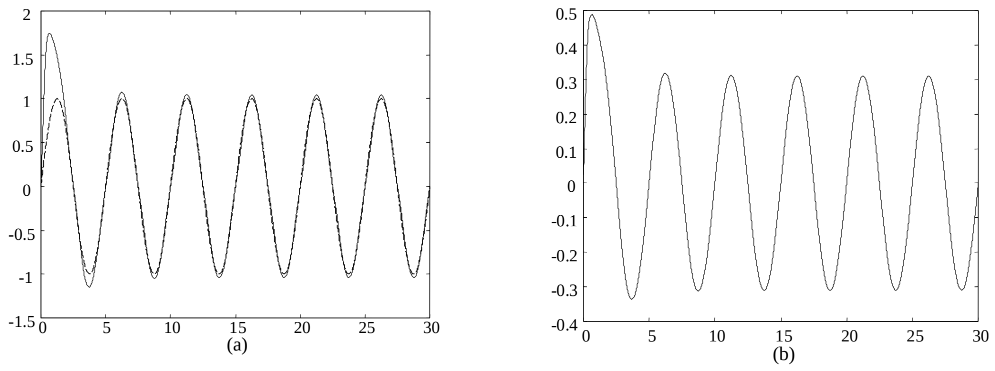

Fig. 3. Continuous line – output of the closed-loop state controlled object; dashed line - **RNN** output,. **RTNN** topology (1, 1, 1), T_0=0.01 s, $h = a =$ 0.5, **MSE %** for t=30 s = 0.00698 % (a); continuous line – control variable of the object. The overshoot in the beginning is caused by the **RNN** poor identification when the system starts to work, but in the next time the system performance goes better (b)

5. Conclusions

A parametric Recurrent Neural Network model and an improved dynamic Back-propagation method of its learning, are applied for nonlinear objects identification and state estimation. The obtained parameters of the RNN model are used for control system design of an indirect adaptive control system. The paper suggests three main types of state-space control with **RNN** state estimation: a proportional; a proportional plus integral and a trajectory- tracking– control. The applicability of the proposed neural indirect adaptive control schemes is confirmed by simulation results.

References

1. N a r e n d r a, K. S., K. P a r t h a s a r a t h y. Identification and control of dynamic systems using neural networks. – In: IEEE Transactions on NNs, 1(1), 1990, 4-27.
2 N a r e n d r a, K. S., K. P a r t h a s a r a t h y. Gradient Methods for the optimisation of dynamical systems containing neural networks. – In: IEEE Transactions on NNs, 2(2), 1991, 252-262.
3. Y i p, P. P. C., Y. H. P a o. A recurrent neural net approach to one-step ahead control problems. – In: IEEE Transactions on SMC, 24(4), 1994, 678-683.
4. P h a m, D.T., S. Y i l d i r i m. Robot control using Jordan neural networks. – In: Proc. of the Internat. Conf. on Recent Advances in Mechatronics, Istanbul, Turkey, Aug. 14-16, 1995. (O. Kaynak, M. Ozkan, N. Bekiroglu, I. Tunay, Eds.). Bogazisi University Printhouse, Istanbul, Turkey, 1995), Vol. II, 888-893.
5. S a s t r y, P. S. , G. S a n t h a r a m, K. P. U n n i k r i s h n a n. Memory networks for identification and control of dynamical systems. – IEEE Transactions on NNs, 5(2), 1994, 306-320.
6. B a r u c h, I., I. S t o y a n o v, E. G o r t c h e v a. Topology and learning of a class RNN. – ELEKTRIK, Vol. 4 Supplement, 1996, 35-42.
7. H u n t, K. J. , D. S b a r b a r o, R. Z b i k o w s k i,P. J. G a w t h r o p. Neural network for control systems – a survey. – Automatica, **28**, 1992, No 6, 1083-1112.
8. O m a t u, S., M.K h a l i l, R.Y u s o f. Neuro-Control and Its Applications. Springer Verlag, London, 1995.
9. B a r u h, I. S. An algorithm and a program for transformation of multivariable automatic control systems into the Luenberger's canonical form. – In: Theory and Application of Cybernetic Systems, vol.1, "Structure and Organization of Control Systems", (Ed. P.Petrov, N.Iliev, K.Tropolov). Sofia, Publ.House of the Bulgarian Academy of Sciences, 1982, 93-100.
10. B a r u c h, I. Use of the Luenberger Canonical Form in The Synthesis of Multivariable Control Systems. – Problems of Engineering Cybernetics, BAS, Sofia, **6**, 1977, 35-39.
11. B a r u c h, I. Synthesis of optimal discrete time control systems with prescribed dynamics. – Problems of Engineering Cybernetics and Robotics, **12**, 1981, 68-73.
12. B a r u h, I. S., J.G. B e n k o f f. Software package for computer-aided design of optimal P, PI, PID Controllers. – Preprints of the 6-th IFAC/IFIP Conf. on "Digital Computer Applications to Process Control", 14-17 Oct., 1980, Dusseldorf, F.R. Germany, .377-381
13. I s i d o r i, A. Nonlinear, Control systems. Third Edition. London, Springer-Verlag, 1995.

Подход рекуррентной нейронной сети для идентификации и управления нелинейных объектов

Йерохам Барух, Хосе Мартин Флорес Албино**, Бойка Ненкова**

**Институт информационных технологий, 1113 София*
***CINVESTAV – IPN, Мексико*

(Р е з ю м е)

Описывается подход для идентификациии нелинейных объектов при помощи нейронной сети. Применяется схема рекуррентной обучаемой нейронной сети (РОНС). Модель нейронной сети параметрический, что позволяет применение параметров, полученных в процессе обучения, при проектировании управляющей системы. РОНС используется как инструмент идентификации и оценки состояния нелинейных дискретных динамических объектов. Предложенный подход иллюстрируется симмуляционными примерами, при чем сравняются выходы при замкнтутом цикле управления состояния с применением РОНС.