Recurrent Neural Identification and Control of a Continuous Bioprocess via First and Second Order Learning

Ieroham Baruch, Carlos-Roman Mariaca-Gaspar

Abstract:

This paper applies a new Kalman Filter Recurrent Neural Network (KFRNN) topology and a recursive Levenberg-Marquardt (L-M) learning algorithm capable to estimate parameters and states of highly nonlinear unknown plant in noisy environment. The proposed KFRNN identifier, learned by the Backpropagation and L-M learning algorithm, was incorporated in a direct and indirect adaptive neural control schemes. The proposed control schemes were applied for real-time recurrent neural identification and control of a continuous stirred tank bioreactor model, where fast convergence, noise filtering and low mean squared error of reference tracking were achieved.

Keywords: *backpropagation learning, continuous stirred tank bioreactor, direct adaptive neural control, indirect adaptive sliding mode control, Kalman filter recurrent neural network identifier, Levenberg-Marquardt learning.*

1. Introduction

The universal approximation abilities of the artificial neural networks to approximate complex non-linear relationships without prior knowledge of the model structure, makes them a very attractive alternative to the classical modeling and control techniques [1], [2], [3]. This property has been proved by the universal approximation theorem [3]. Among several possible network architectures the ones most widely used are the Feedforward (FFNN) and the Recurrent Neural Networks (RNN). In a feedforward neural network the signals are transmitted only in one direction, starting from the input layer, subsequently through the hidden layers to the output layer, which requires applying a tap delayed global feedbacks and a tap delayed inputs to achieve a nonlinear autoregressive moving average neural dynamic plant model. A recurrent neural network has local feedback connections to some of the previous layers. Such a structure is suitable alternative to the first one when the task is to model dynamic systems, and the universal approximation theorem has been proved for the recurrent neural networks too. The preferences given to recurrent neural network identification with respect to the classical methods of process identification are clearly demonstrated in the solution of the "bias-variance dilemma" [3]. Furthermore, the derivation of an analytical plant model, the parameterization of that model and the Least Square solution for the unknown parameters have the following disadvantages: (a) the analytical model did not include all factors having influence to the process behavior; (b) the analytical model is derived taking into account some simplifying suppositions which not ever match; (c) the analytical model did not described all plant nonlinearities, time lags and time delays belonging to the process in hand; (d) the analytical model did not include all process and measurement noises which are sensor and actuator dependent. In (Sage, [4]) the method of invariant imbedding has been described. This method seemed to be a universal tool for simultaneous state and parameter estimation of nonlinear plants but it suffer for the same drawbacks because a complete nonlinear plant model description is needed. Furthermore, the managing of noisy input/output plant data is required to augment the filtering capabilities of the identification RNNs, [5]. Driven by these limitations, a new Kalman Filter Recurrent Neural Network (KFRNN) topology and the recursive Backpropagation (BP) learning algorithm in vector-matrix form has been derived [6] and its convergence has been studied [6], [7]. But the recursive BP algorithm, applied for KFRNN learning, is a gradient descent first order learning algorithm which does not allow to augment the precision and accelerate the learning [5], [7]. Therefore, the aim of this paper was to use a second order learning algorithm for the KFRNN, as the Levenberg-Marquardt (L-M) algorithm is, [8]. The KFRNN with L-M learning was applied for Continuous Stirred Tank Reactor (CSTR) model identification [9], [10]. The application of KFRNNs together with the recursive L-M could prevent all the problems caused by the use of the FFNN, thus improving the learning and the precision of the plant state and parameter estimation in presence of noise. Here, the parameters and states, obtained from the KFRNN identifier will be used in order to design a Direct and Indirect Adaptive Neural Control (DANC and IANC) of CSTR bioprocess plant model.

2. Kalman Filter RNN

This section is dedicated to the KFRNN topology, the recursive Backpropagation and the recursive Levenberg-Marquardt algorithms for the KFRNN learning. The KFRNN is applied as a state and parameter estimator of nonlinear plants.

2.1. Topology of the KFRNN

Let us consider the linearized plant model (1), (2), represented in a state-space form:

$$X_{d.}(k+1) = A_d(k)\,X_d(k) + B_d(k)\,U(k) + \Theta_1(k) \quad (1)$$

$$Y_d(k) = C_d(k)\,X_d(k) + \Theta_2(k) \quad (2)$$

Where: E [.] means mathematical expectation; the pro-

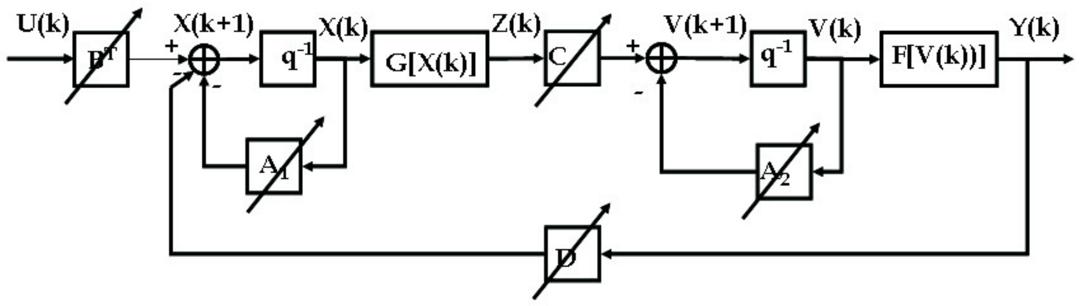

Fig. 1. Block-diagram of the KFRNN topology.

cess and measurement noises Θ_1 (.), Θ_2 (.) are white, with $\Theta_1(k)$, $\Theta_2(s)$ and the initial state $X_d(k_0)$ independent and zero mean for all k, s, with known variances $E\ [X_d(k)\ X_d^T(k)] = P_0$, $E[\Theta_1(k)\ \Theta_1^T(k)] = Q(k)\ \delta\ (k\text{-}\tau)$, $E[\Theta_2(k)\ \Theta_2^T(k)] = R(k)\ \delta\ (k\text{-}\tau)$, where $\delta\ (k\text{-}\tau) = 1$ if $k = \tau$, and 0 otherwise. The optimal Kalman filter theory is completely described in [4], and we would not repeat it here. For us the Kalman Filter (KF) is a full rank optimal state estimator capable to estimate the system states, to filter the process and measurement noises, taking in hand all plant information available like: input/output plant data, all parameters of the plant model (1), (2), and the given up noise and initial state statistics (mean and variance). The basic Kalman filter equations for the estimated state and output variables are given by:

$$X_{d.}(k+1) = \mathrm{A}_e(k)\,\mathrm{X}_e(k) + \mathrm{K}_e(k)\,\mathrm{Y}_d(k) + \mathrm{B}_d(k)\,U(k) \quad (3)$$

$$A_d\,(k) = A_d\,(k) \text{-} K_d\,(k)\,C_d\,(k) \quad (4)$$

$$Y_e\,(k) = C_d\,(k) X_d\,(k) \quad (5)$$

Where: $Xe\ (k)$ is the estimated state vector with dimension N_e; $A_e\ (k)$ is a $(N_e \text{ x } N_e)$ closed-loop KF state matrix; $Y_e\ (k)$ is the estimated plant output vector variable with dimension L; $K_e(k)$ is the optimal Kalman filter gain matrix with dimension $(N_e \text{ x } L)$. This gain matrix is computed applying the optimal Kalman filtering methodology given in [4]. So, the KF performed noise filtration by means of an optimal closed-loop feedback which has the drawback that the feedback amplified the noise components of the error, especially when the feedback gain is high. The second drawback is that the KF design needs complete plant parameter and noise information, which means that if the plant data are incomplete the process noise level is augmented. To overcome this we need to take special measures like to augment the filtering capabilities of the KF. The third drawback is that the KF could not estimate parameters and states in the same time processing noisy measurements with unknown noise statistics, and it will be our task. To resolve this task we need to derive the topology and the BP learning algorithm of a new recurrent KF-like neural network subject of learning and capable to estimate parameters and states in the same time. First of all we could rewrite the equation (3) defining a new extended input vector, containing all available input/output information issued by the plant, and second - we could modify the output equation (5), so to convert it to an output noise filter. After that we obtain:

$$X_{e.}(k+1) = A_d(k) X_e(k) \text{-} K_e(k)\,Y_e(k) + B_2(k)\,U_e(k) \quad (6)$$

$$B_2 = [B_d\,;K_e];\, U_e^T = [U\,;Y_d] \quad (7)$$

$$Z(k) = C_d(k) X_e(k) \quad (8)$$

$$Y_e(k+1) = A_2\,Y_e(k) + Z(k) \quad (9)$$

The obtained new KF RNN topology is given in Fig. 1.

The first layer of the KFRNN represented the plant model (equations (10)-(13)) and the second layer - represented the output noise filtering model (equations (14)-(18)). The KF RNN topology is described by the following equations:

$$X(k+1) = \mathrm{A}_1\mathrm{X}(k) + \mathrm{BU}(k) \text{-} \mathrm{DY}(k) \quad (10)$$

$$B = [B_1\,;B_0];\, UT = [U_1\,;U_2] \quad (11)$$

$$A_1 = \text{block-diag}\,(A_{1,i}),\ |\,A_{1,i}\,| < 1 \quad (12)$$

$$Z_1(k) = G[X(k)] \quad (13)$$

$$C = [C_1\,;C_0];\, Z^T = [Z_1\,;Z_2] \quad (14)$$

$$V_1(k) = CZ(k) \quad (15)$$

$$V(k+1) = V_1(k) + A_2V(k) \quad (16)$$

$$A_1 = \text{block-diag}\,(A_{2,i}),\ |A_{2,i}\,| < 1 \quad (17)$$

$$Y(k) = F[V(k)] \quad (18)$$

Where: X, Y, U are vectors of state, output, and augmented input with dimensions N, L, $(M+1)$, respectively, Z is an $(L+1)$ - dimensional input of the feedforward out-put layer, where Z_1 and U_1 are the $(N\text{x}1)$ output and $(M\text{x}1)$ input of the hidden layer; the constant scalar threshold entries are $Z_2 = \text{-}1$, $U_2 = \text{-}1$, respectively; V is a $(L\text{x}1)$ pre-synaptic activity of the output layer; the super-index T means vector transpose; $A1$, $A2$ are $(N\text{x}N)$ and $(L\text{x}L)$ block-diagonal weight matrices; B and C are $[N_d\ (M+1)]$ and $[Lx(N+1)]$- augmented weight matrices; B_0 and C_0 are $(N\text{x}1)$ and $(L\text{x}1)$ threshold weights of the hidden and output layers; $F[.]$, $G[.]$ are vector-valued tanh(.) or sigmoid(.) -activation functions with corresponding dimensions. Here the input vector U and the input matrix B of the KF RNN are augmented so to fulfill the specifications (7) and the matrix D corresponded to the feedback gain matrix of the KF.

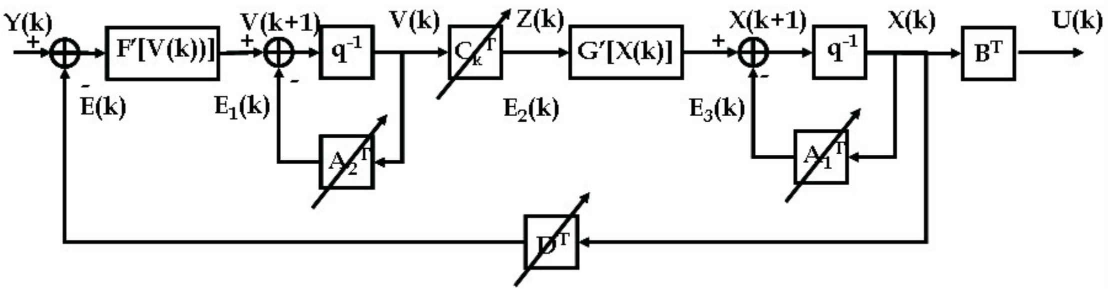

Fig. 2. Block-diagram of the adjoint KFRNN topology.

The dimension of the state vector of the KF RNN is chosen using the simple rule which is: $N=L+M$. From Fig.1 we could see that here we have a two layer Jordan canonical topology with a global feedback which filtered the process noise better then a two layer feedforward topology containing input and output tap delays representing a successive noise sensitive NARMA model, [6].

2.2. BP Learning of the KFRNN

So the KF RNN topology corresponded functionally to the KF definition (6)-(9) and ought to be learnt applying the BP learning algorithm derived using the adjoint KF RNN (see Fig. 2) based on KF RNN topology applying the diagrammatic method, [11].

The BP learning algorithm, expressed in vector-matrix form is as follows:

$$W(k+1) = W(k) + W(k) + W(k-1); \; |W_{ij}| < W_0 \tag{19}$$

$$E(k) = Y_d(k) - Y(k); E_1(k) = F'[Y(k)]\,E(k) \tag{20}$$

$$F'[Y(k)] = [1 - Y^2(k)] \tag{21}$$

$$C(k) = E_1(k)\,Z^T(k) \tag{22}$$

$$\Delta A_2(k) = E_1(k)\,V^T(k) \tag{23}$$

$$Vec(\Delta A_2(k)) = E_1(k) \cdot X(k) \tag{24}$$

$$E_3(k) = G'[Z(k)]\,E_2(k); E_2(k) = CT(k)\,E_1(k) \tag{25}$$

$$G'[Z(k)] = [1 - Z^2(k)] \tag{26}$$

$$B(k) = E_3(k)\,U^T(k) \tag{27}$$

$$D(k) = E_3(k)\,Y^T(k) \tag{28}$$

$$A_1(k) = E_3(k) \cdot X^T(k) \tag{29}$$

$$Vec(A_1(k)) = E_3(k) \cdot X(k) \tag{30}$$

Where: $F'[.]$, $G'[.]$ are derivatives of the tanh(.) activation functions; W is a general weight, denoting each weight matrix (C, A_1, A_2, B, D) in the KF RNN model, to be updated; $W(C, A_1, A_2, B, D)$, is the weight correction of W; Y_d is an L-dimensional output of the approximated plant taken as a reference for KF RNN learning; η, α are learning rate parameters; ΔC is an weight correction of C; ΔB is an weight correction of B; ΔD is an weight correction of D, ΔA_1 is the weight correction of A_1, ΔA_2 is the weight correction of A_2; the diagonals of the matrices A_1, A_2 are denoted by $Vec\ (A_1(k))$, $Vec\ (A_2(k))$, respectively, where (24), (30) represented their learning as an element-by-element vector products; E, E_1, E_2, E_3, are error vectors (see Fig. 2), predicted by the adjoint KF RNN model. So, the KF RNN is capable to issue parameter and state estimations for control purposes, thanks to the optimization capabilities of the BP learning algorithm, applying the "correction for error" delta rule of learning (see Haykin, [3]). The stability of the KF RNN model is assured by the activation functions $[-1, 1]$ bounds and by the local stability weight bound conditions given by (12), (17). The stability of the KF RNN movement around the optimal weight point has been proved by one theorem and the rate of convergence lemma, (see the Ph.D. thesis of Mariaca [7]). It is stated below.

Theorem of stability of the BP KF RNN used as a plant identifier [7]: Let the KF RNN topology is given by equations (10)-(18) (see Fig.1) and the nonlinear plant model, is as follows:

$$X_{d.}(k+1) = G[X_d(k), U(k)] \tag{31}$$

$$Y_d(k) = F[X_d(k)] \tag{32}$$

Where: $\{Y_d(.), X_d(.), U(.)\}$are output, state and input variables with dimensions L, N_d, M, respectively; $G(.)$, $F(.)$ are vector valued nonlinear functions with respective dimensions.

Under the assumption of KF RNN identifiability made, the application of the BP learning algorithm for C, A_1, A_2, B, D, in general vector-matrix form, described by equation (19)-(30), and the learning rates $\eta(k)$, $\alpha(k)$ (here they are considered as time-dependent and normalized with respect to the error) are derived using the following Lyapunov function:

$$L(k) = L_1(k) + L_2(k) \tag{33}$$

Where: $L_1(k)$ and $L_2(k)$ are given by:

$$L_2(k) = \mathrm{tr}\,(\tilde{W}_{A1(k)}\tilde{W}^T_{A1(k)}) + \mathrm{tr}\,(\tilde{W}_{A2(k)}\tilde{W}^T_{A2(k)}) + \mathrm{tr}\,(\tilde{W}_{B(k)}\tilde{W}^T_{B(k)}) + \mathrm{tr}\,(\tilde{W}_{C(k)}\tilde{W}^T_{C(k)}) + \mathrm{tr}\,(\tilde{W}_{D(k)}\tilde{W}^T_{D(k)})$$

Where:

$$\tilde{W}_{A1(k)} = \hat{A}_{1(k)} - A_1^*,\ \tilde{W}_{A2(k)} = \hat{A}_{2(k)} - A_2^*,$$
$$\tilde{W}_{B(k)} = \hat{B}_{(k)} - B^*,\ \tilde{W}_{C(k)} = \hat{C}_{(k)} - C^*,$$
$$\tilde{W}_{D(k)} = \hat{D}_{(k)} - D^*$$

are vectors of the weight estimation error and ($A_1^*, A_2^*, B^*, C^*, D^*$) and ($\hat{A}_{1(k)}, \hat{A}_{2(k)}, \hat{B}_{(k)}, \hat{C}_{(k)}, \hat{D}_{(k)}$) denoted the ideal neural weight and the estimated neural weight at the k-th step, respectively, for each case.
Then the identification error is bounded, i.e.:

$$L(k+1) = L_1(k+1) + L_2(k+1) < 0 \tag{34}$$

$$\Delta L(k+1) = L(k+1) - L(k) \tag{35}$$

Where the condition for $L_1(k+1) < 0$ is fulfilled when the maximum learning rate is chosen in the limits, given below:

$$\frac{\left(1-\frac{1}{\sqrt{2}}\right)}{\psi\max} < \eta\max < \frac{\left(1+\frac{1}{\sqrt{2}}\right)}{\psi\max}$$

For $L_2(k+1) < 0$ fulfillment we have the condition:

$$\Delta L_2(k+1) < -\eta\max|e(k+1)|^2 - \\ -\alpha\max|e(k)|^2 + d(k+1) \tag{36}$$

Note that $\eta\max$ changes adaptively during the learning process of the network, where:

$$\eta_{max} = \max_{i=1}^{5}(\{\eta_i\}$$

Here all: the unmodeled dynamics, the approximation errors and the perturbations, are represented by the d-term, and the complete proof of that theorem and the convergence lemma for (36) are given in the Appendix A and can be seen also with more details in [7].

2.3 . Recursive Levenberg-Marquardt Learning of the KFRNN

The general recursive L-M algorithm of learning, [5], [7], [8] is given by the following equations:

$$W(k+1) = W(k) + P(k)\nabla Y[W(k)]E[W(k)] \tag{37}$$

$$Y[W(k)] = g[W(k), U(k)] \tag{38}$$

$$E^2[W(k)] = \{Y_p(k) - g[W(k), U(k)]\}^2 \tag{39}$$

$$DY[W(k)] = \frac{\partial}{\partial W} g[W, U(k)]\Big|_{W=W(k)} \tag{40}$$

Where: W is a general weight matrix (A_1, A_2, B, C, and D) under modification; P is the covariance matrix of the estimated weights updated; $DY[.]$ is an Nw-dimensional gradient vector; Y is the KFRNN output vector which depends of the updated weights and the input; E is an error vector; Y_p is the plant output vector, which is in fact the target vector. Using the same KFRNN adjoint block diagram (see Fig.2), it was possible to obtain the values of the gradients $DY[.]$ for each updated weight, propagating the value $D(k) = I$ through it. Following the block diagram of Fig. 2, equation (37) was applied for each element of the weight matrices (A_1, A_2, B, C, D) in order to be updated. The corresponding gradient components (40) are obtained as follows:

$$DY[C_{ij}(k)] = D_{1,i}(k)Z_j(k) \tag{41}$$

$$DY[A_{2ij}(k)] = D_{1,i}(k)V_j(k) \tag{42}$$

$$D_{1,i}(k) = F_i'[Y(k)] \tag{43}$$

$$DY[A_{1ij}(k)] = D_{2,i}(k)X_j(k) \tag{44}$$

$$DY[B_{ij}(k)] = D_{2,i}(k)U_j(k) \tag{45}$$

$$DY[D_{ij}(k)] = D_{2,i}(k)Y_j(k) \tag{46}$$

$$D_{2,i}(k) = G_i'[Z_i(k)]C_iD_{1,i}(k) \tag{47}$$

Therefore, the Jacobean matrix could be formed as:

$$DY[W(k)] = [DY(C_{ij}(k)), DY(A_{2ij}(k)), DY(B_{ij}(k)), \\ DY(A_{1ij}(k)), DY(D_{ij}(k))] \tag{48}$$

The $P(k)$ matrix was computed recursively by the equation:

$$P(k) = \alpha^{-1}(k)\{P(k-1) - P(k-1). \\ \Omega[W(k)]S^{-1}[W(k)]\Omega^T[W(k)]P(k-1)\} \tag{49}$$

Where the $S(.)$, and $\Omega(.)$ matrices were given as follows:

$$S[W(k)] = \alpha(k)\Lambda(k) + \Omega^T[W(k)]P(k-1)\Omega[W(k)] \tag{50}$$

$$\Omega^T[W(k)] = \begin{bmatrix} & \nabla Y^T[W(k)] & \\ 0 \ \cdots & 1 & \cdots \ 0 \end{bmatrix};$$

$$\Lambda(k)^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \rho \end{bmatrix}; 10^{-4} \le \rho \le 10^{-6};$$

$$0{,}97 \le \alpha(k) \le 1;\ 10^3 \le P(0) \le 10^6 \tag{51}$$

The matrix $\Omega(.)$ had dimension (Nwx2), whereas the second row had only one unity element (the others were zero). The position of that element was computed by:

$$i = k \bmod (nw) + 1; k > nw \tag{52}$$

After this, the given up topology and learning were applied for the CSTR system identification.

3. Recurrent Trainable NN

This section is dedicated to the topology, the BP and the L-M algorithms of RTNN learning. The RTNN could be obtained from the KFRNN removing the output local and global feedbacks. The RTNN was used as a feedback/feedforward controller.

3.1. Topology of the RTNN

The RTNN model and its learning algorithm of dynamic BP-type, together with the explanatory figures and stability proofs, are described in [6], [7], so only a short description will be given here. The RTNN topology, derived in vector-matrix form, was given by the following equations:

$$X(k+1) = AX(k) + BU(k) \tag{53}$$

$$B = [B_1; B_0]; U^T = [U_1^T; U_2^T] \tag{54}$$

$$A = \text{block-diag}\,(Ai),\ |Ai| < 1 \quad (55)$$

$$Z_1(k) = G[X(k)] \quad (56)$$

$$C = [C_1\,; C_0]; Z^T = [Z_1{}^T; Z_2{}^T] \quad (57)$$

$$V(k) = CZ(k) \quad (58)$$

$$Y(k) = F[V(k)] \quad (59)$$

Where: X, Y, U are vectors of state, output, and augmented input with dimensions $N, L, (M+1)$, respectively, Z is an $(L+1)$ dimensional input of the feedforward output layer, where Z_1 and U_1 are the $(N\text{x}1)$ output and $(M\text{x}1)$ input of the hidden layer; the constant scalar threshold entries are $Z_2 = -1, U_2 = -1$, respectively; V is a $(L\text{x}1)$ pre-synaptic activity of the output layer; the super-index T means vector transpose; A is $(N\text{x}N)$ block-diagonal weight matrix; B and C are $[N\text{x}(M+1)]$ and $[L\text{x}(N+1)]$-augmented weight matrices; B_0 and C_0 are $(N\text{x}1)$ and $(L\text{x}1)$ threshold weights of the hidden and output layers; $F[.], G[.]$ are vector-valued tanh(.) or sigmoid(.) -activation functions with corresponding dimensions. Equation (55) represents the local stability condition imposed on all blocks of A. The dimension of the state vector X of the RTNN is chosen using the simple rule of thumb which is: $N=L+M$.

3.2. BP Learning of the RTNN

The same general BP learning rule (19) was used here. Following the same procedure as for the KFRNN, it was possible to derive the following updates for the RTNN weight matrices:

$$E(k) = Y_d(k) - Y(k); E_1(k) = F'[Y(k)]\,E(k) \quad (60)$$

$$\Delta C(k) = E_1(k)\,Z^T(k) \quad (61)$$

$$E_3(k) = G'[Z(k)]\,E_2(k); E_2(k) = C^T(k)\,E_1(k) \quad (62)$$

$$\Delta B(k) = E_3(k)\,U^T(k) \quad (63)$$

$$\Delta A(k) = E_3(k)\,X^T(k) \quad (64)$$

$$Vec(\Delta A(k)) = E_3(k) \cdot X(k) \quad (65)$$

Where $\Delta A,\ \Delta B,\ \Delta C$ are weight corrections of the of the learned matrices A, B, and C, respectively; E, E_1, E_2, and E_3 are error vectors; X is a state vector; $F'(.)$ and $G'(.)$ are diagonal Jacobean matrices, whose elements are derivatives of the tanh activation functions (see equations (21) and (26)). Equation (64) represents the learning of the full feedback weight matrix of the hidden layer. Equation (65) gives the learning solution when this matrix is diagonal vA, which is the present case. The initial values of the weight matrices during the learning are chosen as arbitrary numbers inside a small range. The stability of the RTNN model used as a direct controller is assured by the activation functions $[-1, 1]$ bounds and by the local stability weight bound condition given by (55). The stability of the RTNN movement around the optimal weight point has been proved by one theorem (see the Ph.D. thesis of Mariaca [7] for more details).

Theorem of stability of the BP RTNN used as a direct system controller [7]: Let the RTNN with Jordan Canonical Structure is given by equations (53)-(59) and the nonlinear plant model is given by (31), (32). Under the assumption of RTNN identifiability made, the application of the BP learning algorithm for $A(.)$, $B(.)$, $C(.)$, in general matricial form, described by equation (19), (63)-(65) without momentum term, and the learning rate $\eta(k)$ (here it is considered as time-dependent and normalized with respect to the error) are derived using the following Lyapunov function:

$$L(k) = L_1(k) + L_2(k) \quad (66)$$

Where: $L_1(k)$ and $L_2(k)$ are given by:

$$L_1(k) = \tfrac{1}{2}\, e^2(k)$$
$$L_2(k) = \text{tr}\,(\tilde{W}_{A(k)}\tilde{W}^T_{A(k)}) + \text{tr}\,(\tilde{W}_{B(k)}\tilde{W}^T_{B(k)}) + \text{tr}\,(\tilde{W}_{C(k)}\tilde{W}^T_{C(k)})$$

Where:

$$\tilde{W}_{A(k)} = \hat{A}_{(k)} - A^*,\ \tilde{W}_{B(k)} = \hat{B}_{(k)} - B^*,\ \tilde{W}_{C(k)} = \hat{C}_{(k)} - C^*$$

are vectors of the estimation error and (A^*, B^*, C^*) and $(\hat{A}_{(k)}, \hat{B}_{(k)}, \hat{C}_{(k)})$ denoted the ideal neural weight and the estimate of the neural weight at the k-th step, respectively, for each case.

Let us define: $\Psi_{max} = \max_k \|\Psi(k)\|$, and $\vartheta_{max} = \max_k \|\vartheta(k)\|$, where $\Psi(k) = \partial o(k)/\partial W(k)$, and $\vartheta(k) = \partial y(k)/\partial u(k)$, where W is a vector composed by all weights of the RTNN, used as a system controller, and $\|\cdot\|$ is an Euclidean norm in $\Re^n$.

Then the identification error is bounded, i.e.:

$$L(k+1) = L_1(k+1) + L_2(k+1) < 0 \quad (67)$$

$$\Delta L(k+1) = L(k+1) - L(k) \quad (68)$$

Where the condition for $L_1(k+1) < 0$ fulfillment is that the maximum rate of learning is inside the limits:

$$0 < \eta_{max} < \frac{2}{\vartheta^2_{max}\Psi^2_{max}}$$

and for $L_2(k+1) < 0$, we have:

$$\Delta L_2(k+1) < -\eta_{max}\,|e(k+1)|^2 + \beta(k+1) \quad (69)$$

Note that η_{max} changes adaptively during the learning process of the network, where:

$$\eta_{max} = \max_{i=1}^{3}(\{\eta_i\}$$

Here all: the unmodelled dynamics, the approximation errors and the perturbations, are represented by the β-term, and the complete proof of that theorem and the rate of convergence lemma, are given in [7].

3.3. Recursive Levenberg-Marquardt Learning of the RTNN

The general recursive L-M algorithm of learning [5],

[7], [8] is given by equations (37)-(40), where W is the general weight matrix (A, B, C) under modification; Y is the RTNN output vector; E is an error vector; Y_p is the plant output vector. Using the RTNN adjoint block diagram [5], it was possible to obtain the values of $DY[.]$ for each updated weight propagating $D=I$. Applying equation (40) for each element of the weight matrices (A, B, C), the corresponding gradient components are obtained as:

$$DY[C_{ij}(k)] = D_{1,i}(k)Z_j(k) \qquad (70)$$

$$D_{1,i}(k) = F_i'[Y_i(k)] \qquad (71)$$

$$DY[A_{ij}(k)] = D_{2,i}(k)X_j(k) \qquad (72)$$

$$DY[B_{ij}(k)] = D_{2,i}(k)U_j(k) \qquad (73)$$

$$D_{2,i}(k) = G_i'[Z_i(k)]C_i D_{1,i}(k) \qquad (74)$$

Therefore the Jacobean matrix could be formed as:

$$DY[W(k)] = [DY(C_{ij}(k)), DY(A_{ij}(k)), DY(B_{ij}(k))] \qquad (75)$$

The $P(k)$ matrix was computed recursively by equations (49)-(52). Next, the given up RTNN topology and learning were applied for CSTR system control.

4. Adaptive Control Systems Design

This section is dedicated to the design of direct and indirect (sliding mode) adaptive control system using the KF RNN as a nonlinear plant identifier. The RTNN was used as a feedback/feedforward controller in the case of direct adaptive neural control.

4.1. Direct Adaptive Neural Control Scheme

This section described the direct adaptive control using KFRNN as plant identifier and RTNN as a plant controller (feedback / feedforward). The block-diagram of the control system is given in Fig. 3. The following study described the linearized model of that closed-loop control system.

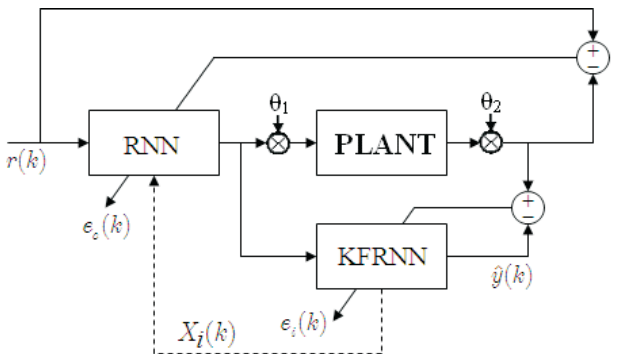

Fig. 3. Block-diagram of the closed-loop neural control system.

Let us present the following z-transfer function representations of the plant, the state estimation part of the KFRNN, and the feedback and feedforward parts of the RTNN controller:

$$W_p(z) = C_p(zI - A_p)^{-1}B_p \qquad (76)$$

$$P_i(z) = (zI - A_i)^{-1}B_i \qquad (77)$$

$$Q_1(z) = C_c(zI - A_c)^{-1}B_{1c} \qquad (78)$$

$$Q_2(z) = C_c(zI - A_c)^{-1}B_{2c} \qquad (79)$$

The control systems z-transfer functions (76)-(79) are connected by the following equation, which is derived from Fig. 3, and is given in z-operational form:

$$Y_p(z) = W_p(z)[I + Q_1(z)P_i(z)]^{-1}Q_2(z)R(z) + \theta(z) \qquad (80)$$

$$\theta(z) = W_p(z)\theta_1(z) + \theta_2(z) \qquad (81)$$

Where: $\theta(z)$ represented a generalized noise term. The RTNN and the KFRNN topologies were controllable and observable, and the BP algorithm of learning was convergent, [5], [7], so the identification and control errors tended to zero:

$$E_i(k) = Y_p(k) - Y(k) \to 0; k \to \infty \qquad (82)$$

$$E_c(k) = R(k) - Y_p(k) \to 0; k \to \infty \qquad (83)$$

This means that each transfer function given by equations (76)-(79) was stable with minimum phase. The closed-loop system was stable and the feedback dynamical part of the RTNN controller compensated the plant dynamics. The feedforward dynamical part of the RTNN controller was an inverse dynamics of the closed-loop system one, which assured a precise reference tracking in spite of the presence of process and measurement noises.

4.2. Indirect Adaptive Control Scheme (Sliding Mode Control)

The indirect adaptive control using the RTNN as plant identifier has been described in, [5]. Later the proposed indirect control has been derived as a Sliding Mode Control (SMC) and applied for control of unknown hydrocarbon biodegradation processes, [6], using the KF RNN identifier with BP learning. Here we applied the KF RNN identifier with L-M learning. The block diagram of the indirect adaptive control scheme is shown in Fig. 4. It contains identification and state estimation KF RNN and a sliding mode controller.

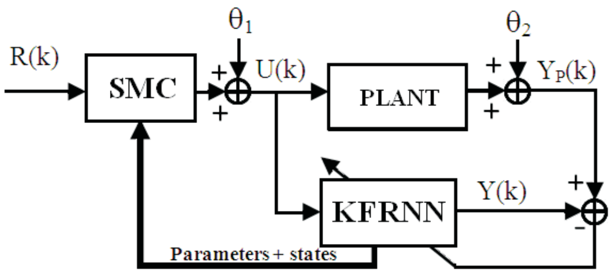

Fig. 4. Block diagram of the closed-loop system containing KF RNN identifier and a SMC.

The stable nonlinear plant is identified by a KF RNN model with topology, given by equations (10)-(18) learned by the stable BP-learning algorithm, given by equations (19)-(30), or using the second order LM-learning algorithm, given by equations (37)-(52). The simplification and linearization of the neural identifier equations (10)-(18), omitting the $DY(.)$ term, leads to the next local linear plant model, extracted from the complete KF RNN model:

$$X(k+1) = A_1 X(k) + BU(k) \tag{84}$$

$$Z(k) = HX(k);\ H = CG'(Z) \tag{85}$$

Where $G'(.)$ is the derivative of the activation function and $L = M$, is supposed.

In [12], the sliding surface is defined with respect to the state variables, and the SMC objective is to move the states form an arbitrary space position to the sliding surface in finite time.

In [13], the sliding surface is also defined with respect to the states but the states of the SISO systems are obtained from the plant outputs by differentiation. In [14], the sliding surface definition and the control objectives are the same. The equivalent control systems design is done with respect to the plant output, but the reachability of the stable output control depended on the plant structure.

In [6], the sliding surface is derived directly with respect to the plant outputs which facilitated the equivalent SMC systems design. Let us define the following sliding surface equation as an output tracking error function:

$$S(k+1) = E(k+1) + \sum_{i=1}^{P} \gamma_i E(k-i+1); |\gamma| < 1 \tag{86}$$

Where: $S(.)$ is the Sliding Surface Error Function (SSEF) defined with respect to the plant output; $E(.)$ is the systems output tracking error; γ_i are parameters of the desired stable SSEF; p is the order of the SSEF. The tracking error in two consecutive moments of time is defined as:

$$S(k+1) = R(k) - Z(k);$$
$$E(k+1) = R(k+1) - Z(k+1) \tag{87}$$

Where $R(k)$, $Z(k)$ are L-dimensional reference and output vectors of the local linear plant model. The objective of the sliding mode control systems design is to find a control action which maintains the systems error on the sliding surface which assure that the output tracking error reaches zero in P steps, where $P < N$. So, the control objective is fulfilled if:

$$S(k+1) = 0 \tag{88}$$

Now, let us to iterate (85) and to substitute (84) in it so to obtain the input/output local plant model, which yields:

$$Z(k+1) = FX(k+1) = F[AX(k) + BU(k)] \tag{89}$$

From (86)-(87), and (89) it is easy to obtain:

$$R(k+1) - Z(k+1) + \sum_{i=1}^{P} \gamma_i E(k-i+1) = 0 \tag{90}$$

The substitution of (89) in (90) gives:

$$R(k+1) - FAX(k) - FBU(k) + \sum_{i=1}^{P} \gamma_i E(k-i+1) = 0 \tag{91}$$

As the local approximation plant model (84), (85), is controllable, observable and stable (see [6], [7]), the matrix A_1 is diagonal, and $L = M$, then the matrix product (HB), representing the plant model static gain, is nonsingular, and the plant states $X(k)$ are smooth non-increasing functions. Now, from (91) it is easy to obtain the equivalent control capable to lead the system to the sliding surface which yields:

$$U_{eq}(k) = (FB)^{-1}\left[-FAX(k) + R(k+1) + \sum_{i=1}^{P} \gamma_i E(k-i+1)\right] \tag{92}$$

Following [12], the SMC avoiding chattering is taken using a saturation function instead of sign one. So the SMC takes the form:

$$U^*(k) = \begin{cases} U_{eq}(k) = & \text{if } \| U_{eq}(k) \| < U_0 \\ -U_0 U_{eq}(k) / \| U_{eq}(k) \| & \text{if } \| U_{eq}(k) \| \geq U_0 \end{cases} \tag{93}$$

The SMC substituted the multi-input multi-output coupled high order dynamics of the linearized plant with desired decoupled low order one.

5. Description of the CSTR Bioprocess Plant

The CSTR model given in [9], [10] was chosen as an example of RNN applications in system identification and control of biotechnological plants. Numerical values for the parameters and nominal operating conditions of this model are given in Table 1.

Table 1. Parameters and operating conditions of the CSTR.

Parameters	Parameters
Q = (L/min)	$\rho{\cdot}\rho_c$ = 1000 (g/L)
C_{Af} = 1.0 (mol/L)	$C_p C_{pc}$ = 1 (cal/gK)
$T_f = T_{fC}$ = 350 (K)	Q_{e0} = 103.41 (L/min)
V = (L)	hA = 7x10^5 (cal/min K)
E/R = 9.95x10^3 (K)	T_0 = 440.2 (K)
$-\Delta H$ = 2x10^5 (cal/mol)	k_0 = 7.2x10^{10} (l/min)

The CSTR is described by the following continuous time nonlinear system of ordinary differential equations:

$$\frac{dC_A(t)}{dt} = \frac{Q}{V}(C_{Af} - C_A(t)) - k_0 C_A(t) \exp\left(-\frac{E}{RT(t)}\right) \tag{94}$$

$$\frac{dT(t)}{dt} = \frac{Q}{V}(T_f - T(t)) + \frac{(-\Delta H) C_A(t)}{\rho C_p} \exp\left(-\frac{E}{RT(t)}\right)$$
$$\frac{\rho_c C_{pc} C_A(t)}{\rho C_p V} Q_c(t)\left[1 - \exp\left(\frac{-hA}{Q_c(t)\rho_c C_{pc}}\right)(T_{ef} - T(t))\right. \tag{95}$$

In this model it is enough to know that within the CSTR, two chemicals are mixed and that they react in order o produce a product compound A at a concentration $C_A(t)$, and that the temperature of the mixture is $T(t)$. The reaction is exothermic and it produces heat which slows down the reaction. By introducing a coolant flow-rate $Q_c(t)$, the temperature can be varied and hence the product concentration can be controlled. Here C_{Af} is the inlet feed concentration; Q is the process flow-rate; T_f and T_{ef} are the inlet feed and coolant temperatures, respectively; all of which are assumed constant at nominal values. Likewise, k_0, E/R, V, ΔH, ρ, C_{pc}, C_p, and ρ_c are thermodynamic and chemical constants related to this particular problem. The quantities Q_{c0}, T_0, and C_{A0}, shown in Table 1, are steady values for a steady operating point in the CSTR. The objective was to control the product compound A by manipulating $Q_c(t)$. The operating values were taken from [9] and [10], where the performance of a NN control system is reported.

6. Simulation Results

Some simulation results of the CSTR biotechnological plant neural identification and control are summarized in this part.

6.1. Simulation Results of Bioprocess Plant Neural Identification

Results of detailed comparative graphical simulation of CSTR KFRNN plant identification by means of the BP and the L-M learning are given in Fig. 5 and Fig. 6. A 10% white noise with different variance (SEED parameter) for each run was added to the plant inputs and outputs and the behavior of the plant identification was studied accumulating some statistics of the final MSE% (ξ_{av}) for KFRNN BP and L-M learning. The results for 20 runs are given in Tables 3 and 4.

The mean average cost for all runs (ε) of KFRNN plant identification, the standard deviation (σ) with respect to the mean value, and the deviation (Δ) are presented in Table 2 for the BP and L-M algorithms. They were computed by the formulas:

$$\varepsilon = \frac{1}{n}\sum_{k=1}^{n}\xi_{av_k},\ \sigma = \sqrt{\frac{1}{n}\sum\Delta_i^2}\ ,\ \Delta = \xi_{av} - \varepsilon \qquad (96)$$

The numerical results given in Tables 2, 3, and 4 are illustrated by the bar-graphics in Figures 7a)and b).

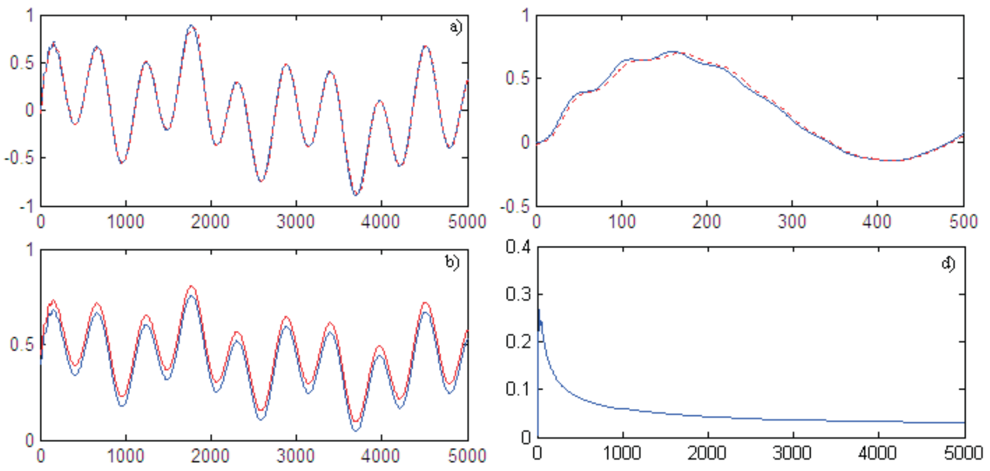

Fig. 5. Graphical results of identification using BP KFRNN learning. a) Comparison of the plant output (continuous line) and KFRNN output (pointed line); b) state variables; c) comparison of the plant output (continuous line) and KFRNN output (pointed line) in the first instants; d) MSE% of identification.

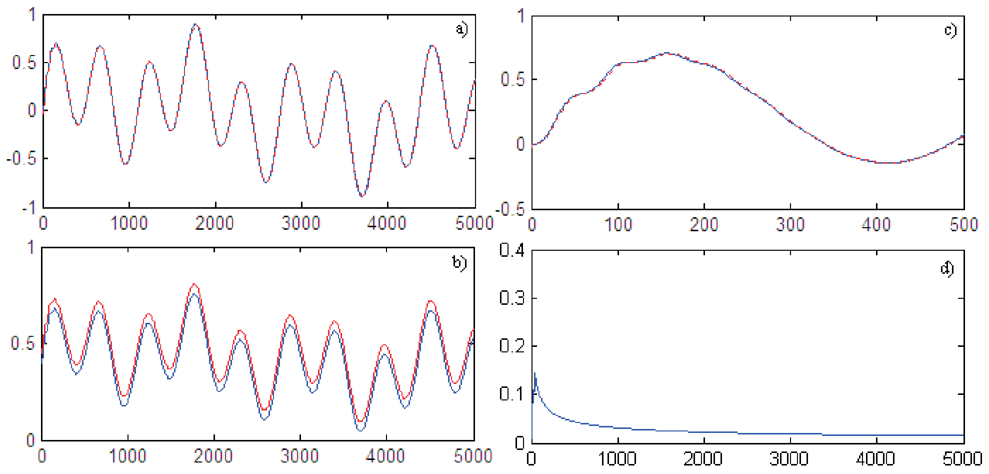

Fig. 6. Graphical results of identification using L-M KFRNN learning. a) Comparison of the plant output (continuous line) and KFRNN output (pointed line); b) state variables; c) comparison of the plant output and KFRNN output in the first instants; d) MSE% of identification.

Table 2. Standard deviations and mean average values of identification validation using the BP and L-M algorithms of KF RNN learning.

BP algorithm	L-M algorithm
$\varepsilon = 0.9457$	$\varepsilon = 0.8264$
$\sigma = 0.0416$	$\sigma = 0.0188$

Table 3. MSE% of 20 runs of the identification program using the KFRNN BP algorithm.

No	1	2	3	4	5
MSE%	0.9559	0.9654	0.8821	0.9614	0.8798
No	6	7	8	9	10
MSE%	0.9444	0.9591	0.9700	0.9685	1.0034
No	11	12	13	14	15
MSE%	0.8523	0.8105	0.9863	0.9038	1.0122
No	16	17	18	19	20
MSE%	0.9688	0.8630	0.8624	0.8521	0.8898

Table 4. MSE% of 20 runs of the identification program using the KFRNN L-M algorithm.

No	1	2	3	4	5
MSE%	0.8123	0.8001	0.8553	0.8360	0.8149
No	6	7	8	9	10
MSE%	0.8072	0.8072	0.8285	0.8236	0.8037
No	11	12	13	14	15
MSE%	0.8659	0.8105	0.8269	0.8218	0.8118
No	16	17	18	19	20
MSE%	0.8628	0.8226	0.8514	0.8288	0.8280

The comparative results showed inferior MSE%, ε, and σ for the L-M algorithm with respect to the BP one.

6.2. Simulation Results of Bioprocess Plant Adaptive Neural Control

The graphical simulation results of DANC using the L-M algorithm of learning are presented in Fig.8 where the final MSE% was 0.854% for the L-M algorithm of learning. Similar results are given on Fig.9 for the indirect SMC control. The final value of the MSE% obtained for the indirect SMC using the L-M algorithm of learning for the KFRNN identifier is of 0.434%.

The graphical results and the obtained final MSE% showed that the indirect SMC control is about twice times more precise that the DANC due to the utilization of the estimated states and parameters in that case, and also due to the SMC algorithm of control which substitute the plant dynamics by a decoupled lower order one.

7. Conclusions

This paper proposed a new KFRNN model for system identification and state estimation of nonlinear plants. The KFRNN is learnt by the first order BP and by the second order L-M recursive learning algorithms. The validating results of system identification reported here gave priority of the L-M algorithm of learning over the BP one which is paid by augmented complexity. The estimated states and parameters of the plant, obtained by this Kalman filter recurrent neural network model are used for direct and indirect adaptive trajectory tracking control system design. The applicability of the proposed neural

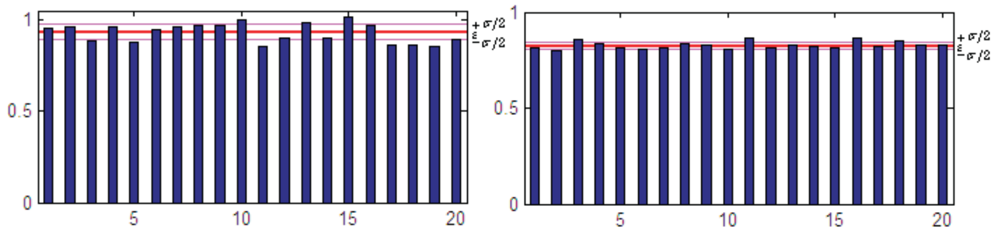

Fig. 7. Comparison between the final MSE% for 20 runs of the identification program: a) using BP algorithm of learning, b) using L-M algorithm of learning.

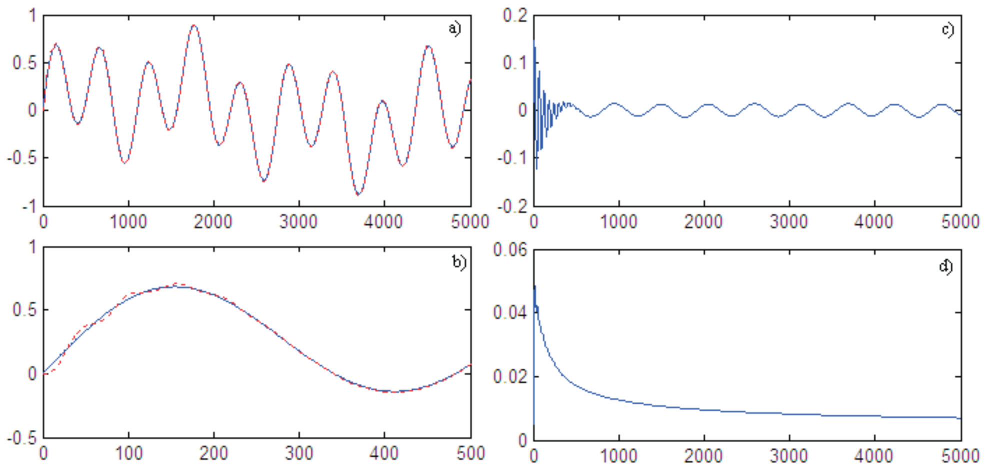

Fig. 8. Detailed graphical simulation results of CSTR plant DANC using L-M learning. a) comparison between the plant output and the reference signal; b) comparison between the plant output and the reference signal in the first instants; c) control signal; d) MSE% of control.

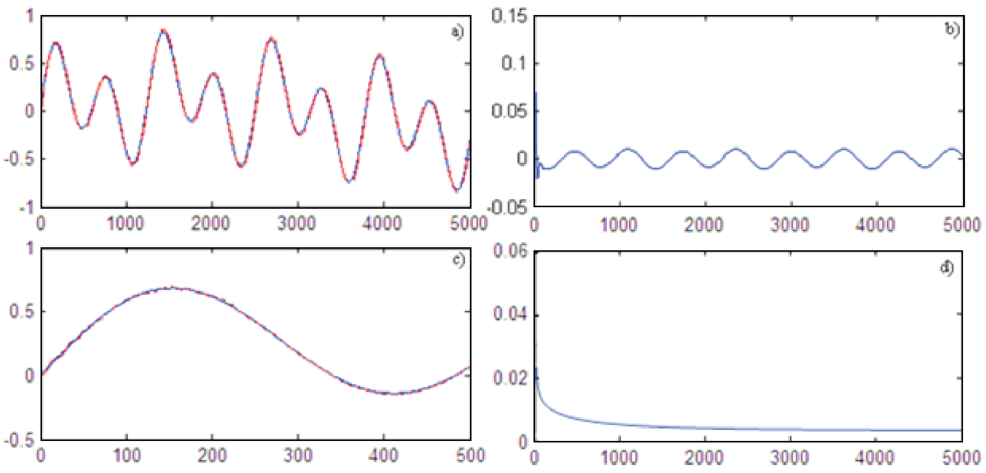

Fig. 9. Detailed graphical simulation results of CSTR plant Sliding Mode Indirect Control using L-M KFRTNN learning. a) comparison between the plant output and the reference signal; b) comparison between the plant output and the reference signal in the first instants; c) control signal; d) MSE% of control.

control system, learnt by the BP and L-M algorithms, was confirmed by simulation results with a CSTR plant. The results showed good convergence of the two algorithms applied. The graphical and numerical validation identification results showed that the L-M algorithm of learning is more precise but more complex then the BP one. The control results of DANC and IANC (SMC) showed a great precision of reference tracking (the final MSE% is 0.854% for the DANC and 0.434 for the indirect SMC). The better results obtained with the indirect SMC are due to the utilization of the estimated states and parameters in that case, and also due to the SMC algorithm of control which substitute the plant dynamics by a decoupled lower order one.

ACKNOWLEDGMENTS
The graduated Ph.D. student Carlos-Roman Mariaca-Gaspar is thankful to CONACYT for the scholarship received during his studies at the Department of Automatic Control, CINVESTAV-IPN, MEXICO.

AUTHORS
Ieroham Baruch*, Carlos-Roman Mariaca-Gaspar - Department of Automatic Control, CINVESTAV-IPN, Av. IPN No 2508, 07360 Mexico City, Mexico. Phone: (+52-55)5747-3800/ext. 42-29. E-mails: {baruch;cmariaca}@ctrl.cinvestav.mx.
* Corresponding author

References
[1] Narendra K.S., Parthasarathy K., "Identification and Control of Dynamical Systems Using Neural Networks", *IEEE Transactions on Neural Networks*, vol. 1, no. 1, 1990, pp. 4-27.
[2] Hunt K.J., Sbarbaro D., Zbikowski R., Gawthrop P.J., "Neural Network for Control Systems (A survey)", Automatica 28 (1992), pp. 1083-1112.
[3] Haykin S., *Neural Networks, a Comprehensive Foundation*, Second Edition, Section 2.13, 84-89; Section 4.13, 208-213. Prentice-Hall, Upper Saddle River, New Jersey 7458, 1999.
[4] Sage A.P., *Optimum Systems Control*, Prentice-Hall Inc., Library of Congress Catalog Number 68-20862, Englewood Cliffs, New Jersey, 1968.
[5] Baruch I. S., Mariaca-Gaspar, C. R., "A Levenberg-Marquardt Learning Applied for Recurrent Neural Identification and Control of a Wastewater Treatment Bioprocess", *International Journal of Intelligent Systems*, Wiley Periodicals, Inc., vol. 24, 2009, pp. 1094-1114.
[6] Baruch I.S., Mariaca-Gaspar C.R., Barrera-Cortes J., "Recurrent Neural Network Identification and Adaptive Neural Control of Hydrocarbon Biodegradation Processes. In: Hu Xiaolin, Balasubramaniam P. (eds.), *Recurrent Neural Networks*, I-Tech Education and Publishing KG, Vienna, Austria, ISBN 978-953-7619-08-4, 2008, Chapter 4, pp. 61-88.
[7] Mariaca Gaspar C.R., *Topologies, Learning and Stability of Hybrid Neural Networks, Applied for Nonlinear Biotechnological Processes*, Ph. D. Thesis (in Spanish), Baruch, I.S., Martinez-Garcia, J.C. (thesis directors), Department of Automatic Control, CINVESTAV-IPN, Mexico City, 3rd July 2009.
[8] Ngia L.S., Sjöberg J., "Efficient Training of Neural Nets for Nonlinear Adaptive Filtering Using a Recursive Levenberg Marquardt Algorithm", *IEEE Trans. on Signal Processing*, vol. 48, 2000, pp. 1915-1927.
[9] Zhang T., Guay M., "Adaptive Nonlinear Control of Continuously Stirred Tank Reactor Systems". In: *Proceedings of the American Control Conference*, Arlington, 25th-27th June, 2001, pp. 1274-1279.
[10] Lightbody G., Irwin G.W., "Nonlinear Control Structures Based on Embedded Neural System Models", *IEEE Trans. on Neural Networks*, no. 8, 1997, pp. 553-557.
[11] Wan E., Beaufays F., "Diagrammatic Method for Deriving and Relating Temporal Neural Network Algorithms", *Neural Computations*, vol. 8, 1996, pp. 182-201.
[12] Young K. D., Utkin V.I., Ozguner U., "A Control Engineer's Guide to Sliding Mode Control", *IEEE Transactions on Control Systems Technology* 7 (3), 1999, pp. 328-342.
[13] Levent A., "Higher Order Sliding Modes, Differentiation and Output Feedback Control", *International Journal of Control, Special Issue Dedicated to Vadim Utkin on the Occasion of his 65th Birthday* (Guest Editor: Fridman L.M.), ISSN 0020-7179, vol. 76, no. 9/10, 15th June-10th

July 2003, pp. 924-941.

[14] Eduards C., Spurgeon S.K., Hebden, R.G., "On the Design of Sliding Mode Output Feedback Controllers", *International Journal of Control, Special Issue Dedicated to Vadim Utkin on the Occasion of his 65th Birthday* (Guest Editor: Fridman L.M.), ISSN 0020-7179, vol.76, no. 9/10, 15th June-10th July 2003, pp. 893-905

Appendix A: Stability proof of the theorem for KF RNN topology and BP learning.

Let the Extended Recurrent Trainable Neural Network with Jordan Canonical Structure given by (1), (2), (3), (4), (5), (6), (7) and the nonlinear plant model as follows:

$$x(k+1) = f[x(k), v(k)] \tag{A.1}$$
$$y(k) = h[x(k)] \tag{A.2}$$

and the plant and activation functions fulfill the following assumptions:

Assumption 1: The plant dynamics is locally Lipchitz, so the nonlinear functions $f(\cdot), h(\cdot)$ are as:

$$f := \{f \mid f = \sigma + \Delta f, \| \Delta f \| \le f_0 + f_1 \| x(k) \| \}$$
$$h := \{h \mid h = \sigma + \Delta h, \| \Delta h \| \le h_0 + h_1 \| x(k) \| \}$$

and Δf, Δh are modeling errors, which reflex the effect of unmodelled dynamics.

Assumption 2: The activation functions have the following Taylor approximation:

$$\sigma(\overline{\theta}) = \sigma(\theta) + \frac{\partial \sigma(\overline{\theta})}{\partial \overline{\theta}}(\overline{\theta} - \theta) + \varsigma$$

with the approximation error bound given by:

$$\|\varsigma\|^2 \le \frac{L}{2}\|\overline{\theta} - \theta\|^2$$

and the output signal error is defined by:

$$e(k) = \hat{y}(k) - y(k)$$
$$e(k+1) = \hat{y}(k+1) - y(k+1) = \Phi[C(k)\hat{x}(k) +$$
$$+ A_2(k)v(k)] - \Phi\left[C^*\hat{x}(k) + A_2^* v(k)\right] - \Delta h[x(k)]$$

Now, let us define the state estimation error, add and subtract the RTNN to the last equation and apply the Assumption 2, then:

$$\Delta(k) = \hat{x}(k) - x(k)$$
$$\Delta(k+1) = \hat{x}(k+1) - x(k+1) = \Gamma[A_1(k)\hat{x}(k) +$$
$$+ B(k)u(k) + D(k)\hat{y}(k)]$$
$$- \Gamma\left[A_1^*\hat{x}(k) + B^*u(k) + D^*\hat{y}(k)\right] - \Delta f[x(k), u(k)]$$

Let us now define the output identification error and put it in terms of the state estimation error as:

$$e(k+1) = \Phi'(k)\delta C(k)\left[\Gamma'(k)\left(\delta A_1(k)\hat{x}(k) + \delta B(k)\right.\right.$$
$$\left.\left. u(k) + \delta D(k)\right) + \Theta_3\right] + \Phi'(k)\delta C^*\left[\Gamma'(k)\left(A_1^*\Delta(k) -\right.\right.$$
$$\left.\left. - B^* O_F + D^*\right) + \Theta_4\right] + \Theta_1 + \Theta_2 - \Delta h\left(x(k), \overline{u}(k)\right)$$

Where: the term $\overline{u}(k) = u(k) + O_F$; $\Theta_{1,2,3,4}$ the are the higher order terms in the Taylor series approximation; $\Delta h(x(k), \overline{u}(k)) = u(k) + O_F$ is the unmodeled dynamics; O_F is an offset.

If Assumptions 1 and 2 fulfill, the learning algorithm for the RTNN is given by (8) and the learning parameters η_k, α_k are normalized and depended on the output error structure.

Then, the approximation error is bounded.

Consider a Lyapunov candidate function as:

$$L(k) = L_1(k) + L_2(k) \tag{A.3}$$

In which $L_1(k)$ and $L_2(k)$ are given by:

$$L_1(k) = \tfrac{1}{2}e^2(k) \tag{A.4}$$

$$L_2(k) = tr\left(\tilde{W}_{A1}(k)\tilde{W}_{A1}^T(k)\right) + tr\left(\tilde{W}_{A2}(k)\tilde{W}_{A2}^T(k)\right) +$$
$$+ tr\left(\tilde{W}_B(k)\tilde{W}_B^T(k)\right) + tr\left(\tilde{W}_C(k)\tilde{W}_C^T(k)\right) +$$
$$+ tr\left(\tilde{W}_D(k)\tilde{W}_D^T(k)\right) \tag{A.5}$$

Where: are vectors of the estimation error and (A_1^*, A_2^*, B^*, C^*, D^*) and ($\widehat{A}_{1(k)}$, $\widehat{A}_{1(k)}$, $\widehat{B}_{(k)}$, $\widehat{C}_{(k)}$, $\widehat{D}_{(k)}$) denoted the ideal neural weight and the estimate of neural weight at the k-th step, respectively, for each case.

Let us consider the equation (A.4). The change of the Lyapunov function in two consecutive samples due to the training process is obtained by:

$$\Delta L_1(k) = L_1(k+1) - L_1(k) = [e(k+1) - e(k)][e(k) +$$
$$+ \tfrac{1}{2}e(k+1) - \tfrac{1}{2}e(k)] \tag{A.6}$$

Then, defining $\Delta e(k)$ as the difference between two consecutive error samples, then the equation (A.6) becomes:

$$\Delta L_1(k) = \Delta e(k)[e(k) + \tfrac{1}{2}\Delta e(k)] \tag{A.7}$$

Where: can be defined as:

$$\Delta e(k) = \left[\frac{\partial e(k)}{\partial W}\right]\Delta \vec{W} \tag{A.8}$$

Putting all weights into one vector as

$$\vec{W} = [[\vec{A}_1]^T\ [\vec{A}_2]^T\ [\vec{B}]^T\ [\vec{C}]^T\ [\vec{D}]^T]^T \tag{A.9}$$

Where:

$\vec{A}_1 = [[\vec{A}_{1n}]^T\ [\vec{A}_{1n}]^T \ldots [\vec{A}_{1n}]]^T$,
$\vec{A}_2 = [[\vec{A}_{21}]^T\ [\vec{A}_{22}]^T \ldots [\vec{A}_{2l}]]^T$,
$\vec{B} = [[\vec{B}_1]^T\ [\vec{B}_2]^T \ldots [\vec{B}_m]]^T$,
$\vec{C} = [[\vec{C}_1]^T\ [\vec{C}_2]^T \ldots [\vec{C}_n]]^T$,
$\vec{D} = [\vec{D}_1]^T$

which represents the weight vectors constructed by their columns. Also let:

$$\eta = \begin{bmatrix} \eta^{A1} & & & & \\ & \eta^{A2} & & & \\ & & \eta^{B} & & \\ & & & \eta^{C} & \\ & & & & \eta^{D} \end{bmatrix}, \quad \alpha = \begin{bmatrix} \alpha^{A1} & & & & \\ & \alpha^{A2} & & & \\ & & \alpha^{B} & & \\ & & & \alpha^{C} & \\ & & & & \alpha^{D} \end{bmatrix} \tag{A.10}$$

Where: (η^{A_1}, η^{A_2}, η^B, η^C, η^D) and (α^{A_1}, α^{A_2}, α^B, α^C, α^D) represented the learning rate matrices, the momentum rate matrices corresponding to the matrix weights (A_1, A_2, B, C, D), respectively, and $\eta^{A_1} = \eta_1 I_{A_1}$, $\eta^{A_2} = \eta_2 I_{A2}$, $\eta^B = \eta_2 I_B$, $\eta^C = \eta_3 I_C$, $\eta^D = \eta_3 I_C$, $\eta^D = \eta_5 I_D$, $\alpha^{A_1} = \alpha_1 I_{A_1}$, $\alpha^{A_2} = \alpha_2 I_{A2}$, $\alpha^B = \alpha_2 I_B$, $\alpha^C = \alpha_3 I_C$, $\alpha^D = \alpha_5 I_D$. Moreover, $\eta_i(i = 1,\ldots,5)$ and $\alpha_i(i = 1,\ldots,5)$ are two positive constants, and I_Z is an identity matrix, where the general symbol Z is substituted by A_1, A_2, B, C, D, respectively. We could define $\underline{\Delta} W$ as:

$$\underline{\Delta} W = \eta\Delta W(k) + \alpha\Delta W(k-1) \tag{A.11}$$

$$\underline{\Delta} W = -\eta\frac{\partial V_1(k)}{\partial W} - \alpha\frac{\partial V_1(k-1)}{\partial W} = -e(k)\eta\frac{\partial e(k)}{\partial W} - e(k-1)\alpha\frac{\partial e(k-1)}{\partial W} = -e(k)\begin{bmatrix} \eta^{A_1} & & & & \\ & \eta^{A_2} & & & \\ & & \eta^B & & \\ & & & \eta^C & \\ & & & & \eta^D \end{bmatrix} \cdot \left[\left[\frac{\partial e(k)}{\partial A_1}\right]^T \left[\frac{\partial e(k)}{\partial A_2}\right]^T \left[\frac{\partial e(k)}{\partial B}\right]^T \left[\frac{\partial e(k)}{\partial C}\right]^T \left[\frac{\partial e(k)}{\partial D}\right]^T\right]^T$$

$$--e(k-1)\begin{bmatrix} \alpha^{A_1} & & & & \\ & \alpha^{A_2} & & & \\ & & \alpha^B & & \\ & & & \alpha^C & \\ & & & & \alpha^D \end{bmatrix} \cdot \left[\left[\frac{\partial e(k-1)}{\partial A_1}\right]^T \left[\frac{\partial e(k-1)}{\partial A_2}\right]^T \left[\frac{\partial e(k-1)}{\partial B}\right]^T \left[\frac{\partial e(k-1)}{\partial C}\right]^T \left[\frac{\partial e(k-1)}{\partial D}\right]^T\right]^T \tag{A.12}$$

$$\Delta e(k) = \left[\frac{\partial e(k)}{\partial W}\right]^T \underline{\Delta} W = -e(k)\cdot\left(\eta_1\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \eta_2\left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \eta_3\left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \eta_4\left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \eta_5\left\|\frac{\partial e(k)}{\partial D}\right\|^2\right)$$
$$-e(k-1)\cdot\left(\alpha_1\left\|\frac{\partial e(k-1)}{\partial A_1}\right\|^2 + \alpha_2\left\|\frac{\partial e(k-1)}{\partial A_2}\right\|^2 + \alpha_3\left\|\frac{\partial e(k-1)}{\partial B}\right\|^2 + \alpha_4\left\|\frac{\partial e(k-1)}{\partial C}\right\|^2 + \alpha_5\left\|\frac{\partial e(k-1)}{\partial D}\right\|^2\right) \tag{A.13}$$

Let:

$$\gamma = \eta_1\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \eta_2\left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \eta_3\left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \eta_4\left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \eta_5\left\|\frac{\partial e(k)}{\partial D}\right\|^2$$
$$\lambda = \alpha_1\left\|\frac{\partial e(k-1)}{\partial A_1}\right\|^2 + \alpha_2\left\|\frac{\partial e(k-1)}{\partial A_2}\right\|^2 + \alpha_3\left\|\frac{\partial e(k-1)}{\partial B}\right\|^2 + \alpha_4\left\|\frac{\partial e(k-1)}{\partial C}\right\|^2 + \alpha_5\left\|\frac{\partial e(k-1)}{\partial D}\right\|^2 \tag{A.14}$$

Then:

$$\Delta e(k+1) = -\gamma e(k+1) - \lambda e(k) \tag{A.15}$$

and

$$\Delta L_1(k+1) = \Delta e(k+1)[e(k+1) + \tfrac{1}{2}\Delta e(k+1)] = -\tfrac{1}{2}e^2(k+1)[2\gamma-\gamma^2] + e(k+1)e(k)[\gamma-1]\lambda + \tfrac{1}{2}\lambda^2 e^2(k) \tag{A.16}$$

Proposing: $\lambda = \gamma-1$, then:

$$\Delta L_1(k+1) = -\tfrac{1}{2}e^2(k+1)[-2\gamma^2+4\gamma-1] - \tfrac{1}{2}\lambda^2[\Delta e(k)]^2 \tag{A.17}$$

According to the Lyapunov stability theory, if convergence must be guaranteed, then $\Delta L(k+1) < 0$, thus $-2\gamma^2+4\gamma-1 > 0$, and:

$$\left(1-\tfrac{1}{\sqrt{2}}\right) < \gamma < \left(1+\tfrac{1}{\sqrt{2}}\right) \tag{A.18}$$

That is:

$$\left(1-\frac{1}{\sqrt{2}}\right) < \eta_1\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \eta_2\left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \eta_3\left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \eta_4\left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \eta_5\left\|\frac{\partial e(k)}{\partial D}\right\|^2 < \left(1+\frac{1}{\sqrt{2}}\right) \tag{A.19}$$

Let: $\eta_{\max} = \max_{i=1}^{3}\{\eta_i\}$. Thus, as long as:

$$\frac{\left(1-\frac{1}{\sqrt{2}}\right)}{\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \left\|\frac{\partial e(k)}{\partial D}\right\|^2} < \eta_{\max} < \frac{\left(1+\frac{1}{\sqrt{2}}\right)}{\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \left\|\frac{\partial e(k)}{\partial D}\right\|^2} \tag{A.20}$$

Note that $\|\cdot\|$ is the Euclidean norm, therefore:

$$\left\|\frac{\partial e(k)}{\partial A_1}\right\|^2 + \left\|\frac{\partial e(k)}{\partial A_2}\right\|^2 + \left\|\frac{\partial e(k)}{\partial B}\right\|^2 + \left\|\frac{\partial e(k)}{\partial C}\right\|^2 + \left\|\frac{\partial e(k)}{\partial D}\right\|^2 = \left\|\frac{\partial e(k)}{\partial W}\right\|^2 \tag{A.21}$$

Now let: $\Psi(k) = \dfrac{\partial e(k)}{\partial W} = -\dfrac{\partial y(k)}{\partial W}$, and $\Psi_{max} = \max_k \|\Psi(k)\|$, then:

$$\frac{\left(1-\frac{1}{\sqrt{2}}\right)}{\Psi_{max}} < \eta_{max} < \frac{\left(1+\frac{1}{\sqrt{2}}\right)}{\Psi_{max}} \tag{A.22}$$

Now, working with equation (A5), we have:

$$L_2(k) = tr\left(\tilde{W}_{A1}(k)\tilde{W}_{A1}^T(k)\right) + tr\left(\tilde{W}_{A2}(k)\tilde{W}_{A2}^T(k)\right) + tr\left(\tilde{W}_B(k)\tilde{W}_B^T(k)\right) + tr\left(\tilde{W}_C(k)\tilde{W}_C^T(k)\right) + tr\left(\tilde{W}_D(k)\tilde{W}_D^T(k)\right)$$

If we consider the change of the Lyapunov function in two consecutive samples due to the training process, we obtain:

$$\begin{aligned}\Delta L_2(k) &= L_2(k+1) - L_2(k)\\ &= tr\left(\tilde{W}_{A1}(k+1)\tilde{W}_{A1}^T(k+1)\right) + tr\left(\tilde{W}_A(k+1)\tilde{W}_A^T(k+1)\right) + tr\left(\tilde{W}_B(k+1)\tilde{W}_B^T(k+1)\right) + tr\left(\tilde{W}_C(k+1)\tilde{W}_C^T(k+1)\right) +\\ &+ tr\left(\tilde{W}_D(k+1)\tilde{W}_D^T(k+1)\right) - tr\left(\tilde{W}_{A1}(k)\tilde{W}_{A1}^T(k)\right) - tr\left(\tilde{W}_A(k)\tilde{W}_A^T(k)\right) - tr\left(\tilde{W}_B(k)\tilde{W}_B^T(k)\right) - tr\left(\tilde{W}_C(k)\tilde{W}_C^T(k)\right) - tr\left(\tilde{W}_D(k)\tilde{W}_D^T(k)\right)\end{aligned} \tag{A.23}$$

Now substituting the following quantities:

$$\tilde{W}_{A_1}(k) = \hat{A}_1(k) - A_1^*, \tilde{W}_{A_2}(k) = \hat{A}_2(k) - A_2^*, \tilde{W}_B(k) = \hat{B}(k) - B^*, \tilde{W}_C(k) = \hat{C}(k) - C^*, \tilde{W}_D(k) = \hat{D}(k) - D^*$$

We could obtain:

$$\begin{aligned}\Delta L_2(k) = tr&\begin{pmatrix}\hat{A}_1(k+1)\hat{A}_1^T(k+1) - \hat{A}_1(k+1)A_1^{*T} - A_1^*\hat{A}_1^T(k+1) + A_1^*A_1^{*T}\\ -\hat{A}_1(k)\hat{A}_1^T(k) + \hat{A}_1(k)A_1^{*T} + A_1^*\hat{A}_1^T(k) - A_1^*A_1^{*T}\end{pmatrix}\\ tr&\begin{pmatrix}\hat{A}_2(k+1)\hat{A}_2^T(k+1) - \hat{A}_2(k+1)A_2^{*T} - A_2^*\hat{A}_2^T(k+1) + A_2^*A_2^{*T}\\ -\hat{A}_2(k)\hat{A}_2^T(k) + \hat{A}_2(k)A_2^{*T} + A_2^*\hat{A}_2^T(k) - A_2^*A_2^{*T}\end{pmatrix}\\ +tr&\begin{pmatrix}\hat{B}(k+1)\hat{B}^T(k+1) - \hat{B}(k+1)B^{*T} - B^*\hat{B}^T(k+1) + B^*B^{*T}\\ -\hat{B}(k)\hat{B}^T(k) + \hat{B}(k)B^{*T} + B^*\hat{B}^T(k) - B^*B^{*T}\end{pmatrix}\\ +tr&\begin{pmatrix}\hat{C}(k+1)\hat{C}^T(k+1) - \hat{C}(k+1)C^{*T} - C^*\hat{C}^T(k+1) + C^*C^{*T}\\ -\hat{C}(k)\hat{C}^T(k) + \hat{C}(k)C^{*T} + C^*\hat{C}^T(k) - C^*C^{*T}\end{pmatrix}\\ +tr&\begin{pmatrix}\hat{D}(k+1)\hat{D}^T(k+1) - \hat{D}(k+1)D^{*T} - D^*\hat{D}^T(k+1) + D^*D^{*T}\\ -\hat{D}(k)\hat{D}^T(k) + \hat{D}(k)D^{*T} + D^*\hat{D}^T(k) - D^*D^{*T}\end{pmatrix}\end{aligned} \tag{A.24}$$

And if the updated learning law is given by (19), then:

$$\Delta L_2(k) = tr\begin{pmatrix}\hat{A}_1(k)\hat{A}_1^T(k) + \eta_{\max}\hat{A}_1(k)\Delta\hat{A}_1^T(k) + \alpha_{\max}\hat{A}_1(k)\Delta\hat{A}_1^T(k-1) + \eta_{\max}\Delta\hat{A}_1(k)\hat{A}_1^T(k)\\ +\eta_{\max}^2\Delta\hat{A}_1(k)\Delta\hat{A}_1^T(k) + \alpha_{\max}\eta_{\max}\Delta\hat{A}_1(k)\Delta\hat{A}_1^T(k-1) + \alpha_{\max}\Delta\hat{A}_1(k-1)\hat{A}_1^T(k)\\ +\alpha_{\max}\eta_{\max}\Delta\hat{A}_1(k-1)\Delta\hat{A}_1^T(k) + \alpha_{\max}^2\Delta\hat{A}_1(k-1)\Delta\hat{A}_1^T(k-1) - \hat{A}_1(k)A_1^{*T}\\ -\eta_{\max}\Delta\hat{A}_1(k)A_1^{*T} - \alpha_{\max}\Delta\hat{A}_1(k-1)A_1^{*T} - A_1^*\hat{A}_1^T(k) - \eta_{\max}A_1^*\Delta\hat{A}_1^T(k)\\ -\alpha_{\max}A_1^*\Delta\hat{A}_1^T(k-1) - \hat{A}_1(k)\hat{A}_1^T(k) + \hat{A}_1(k)A_1^{*T} + A_1^*\hat{A}_1^T(k)\end{pmatrix}$$

$$tr\begin{pmatrix} \hat{A}_2(k)\hat{A}_2^T(k)+\eta_{\max}\hat{A}_2(k)\Delta\hat{A}_2^T(k)+\alpha_{\max}\hat{A}_2(k)\Delta\hat{A}_2^T(k-1)+\eta_{\max}\Delta\hat{A}_2(k)\hat{A}_2^T(k) \\ +\eta_{\max}^2\Delta\hat{A}_2(k)\Delta\hat{A}_2^T(k)+\alpha_{\max}\eta_{\max}\Delta\hat{A}_2(k)\Delta\hat{A}_2^T(k-1)+\alpha_{\max}\Delta\hat{A}_2(k-1)\hat{A}_2^T(k) \\ +\alpha_{\max}\eta_{\max}\Delta\hat{A}_2(k-1)\Delta\hat{A}_2^T(k)+\alpha_{\max}^2\Delta\hat{A}_2(k-1)\Delta\hat{A}_2^T(k-1)-\hat{A}_2(k)A_2^{*T} \\ -\eta_{\max}\Delta\hat{A}_2(k)A_2^{*T}-\alpha_{\max}\Delta\hat{A}_2(k-1)A_2^{*T}-A_2^*\hat{A}_2^T(k)-\eta_{\max}A_2^*\Delta\hat{A}_2^T(k) \\ -\alpha_{\max}A_2^*\Delta\hat{A}_2^T(k-1)-\hat{A}_2(k)\hat{A}_2^T(k)+\hat{A}_2(k)A_2^{*T}+A_2^*\hat{A}_2^T(k) \end{pmatrix}$$

$$+tr\begin{pmatrix} \hat{B}(k)\hat{B}^T(k)+\eta_{\max}\hat{B}(k)\Delta\hat{B}^T(k)+\alpha_{\max}\hat{B}(k)\Delta\hat{B}^T(k-1)+\eta_{\max}\Delta\hat{B}(k)\hat{B}^T(k) \\ +\eta_{\max}^2\Delta\hat{B}(k)\Delta\hat{B}^T(k)+\alpha_{\max}\eta_{\max}\Delta\hat{B}(k)\Delta\hat{B}^T(k-1)+\alpha_{\max}\Delta\hat{B}(k-1)\hat{B}^T(k) \\ +\alpha_{\max}\eta_{\max}\Delta\hat{B}(k-1)\Delta\hat{B}^T(k)+\alpha_{\max}^2\Delta\hat{B}(k-1)\Delta\hat{B}^T(k-1)-\hat{B}(k)B^{*T} \\ -\eta_{\max}\Delta\hat{B}(k)B^{*T}-\alpha_{\max}\Delta\hat{B}(k-1)B^{*T}-B^*\hat{B}^T(k)-\eta_{\max}B^*\Delta\hat{B}^T(k) \\ -\alpha_{\max}B^*\Delta\hat{B}^T(k-1)-\hat{B}(k)\hat{B}^T(k)+\hat{B}(k)\hat{B}^{*T}+\hat{B}^*\hat{B}^T(k) \end{pmatrix}$$

$$+tr\begin{pmatrix} \hat{C}(k)\hat{C}^T(k)+\eta_{\max}\hat{C}(k)\Delta\hat{C}^T(k)+\alpha_{\max}\hat{C}(k)\Delta\hat{C}^T(k-1)+\eta_{\max}\Delta\hat{C}(k)\hat{C}^T(k) \\ +\eta_{\max}^2\Delta\hat{C}(k)\Delta\hat{C}^T(k)+\alpha_{\max}\eta_{\max}\Delta\hat{C}(k)\Delta\hat{C}^T(k-1)+\alpha_{\max}\Delta\hat{C}(k-1)\hat{C}^T(k) \\ +\alpha_{\max}\eta_{\max}\Delta\hat{C}(k-1)\Delta\hat{C}^T(k)+\alpha_{\max}^2\Delta\hat{C}(k-1)\Delta\hat{C}^T(k-1)-\hat{C}(k)C^{*T} \\ -\eta_{\max}\Delta\hat{C}(k)C^{*T}-\alpha_{\max}\Delta\hat{C}(k-1)C^{*T}-C^*\hat{C}^T(k)-\eta_{\max}C^*\Delta\hat{C}^T(k) \\ -\alpha_{\max}C^*\Delta\hat{C}^T(k-1)-\hat{C}(k)\hat{C}^T(k)+\hat{C}(k)\hat{C}^{*T}+\hat{C}^*\hat{C}^T(k) \end{pmatrix}$$

$$+tr\begin{pmatrix} \hat{D}(k)\hat{D}^T(k)+\eta_{\max}\hat{D}(k)\Delta\hat{D}^T(k)+\alpha_{\max}\hat{D}(k)\Delta\hat{D}^T(k-1)+\eta_{\max}\Delta\hat{D}(k)\hat{D}^T(k) \\ +\eta_{\max}^2\Delta\hat{D}(k)\Delta\hat{D}^T(k)+\alpha_{\max}\eta_{\max}\Delta\hat{D}(k)\Delta\hat{D}^T(k-1)+\alpha_{\max}\Delta\hat{D}(k-1)\hat{D}^T(k) \\ +\alpha_{\max}\eta_{\max}\Delta\hat{D}(k-1)\Delta\hat{D}^T(k)+\alpha_{\max}^2\Delta\hat{D}(k-1)\Delta\hat{D}^T(k-1)-\hat{D}(k)D^{*T} \\ -\eta_{\max}\Delta\hat{D}(k)D^{*T}-\alpha_{\max}\Delta\hat{D}(k-1)D^{*T}-D^*\hat{D}^T(k)-\eta_{\max}D^*\Delta\hat{D}^T(k) \\ -\alpha_{\max}D^*\Delta\hat{D}^T(k-1)-\hat{D}(k)\hat{D}^T(k)+\hat{D}(k)D^{*T}(k)+D^*(k)\hat{D}^T(k) \end{pmatrix}$$

Now we could use the followings trace properties:

$tr(AB)=tr(BA); tr(A)+tr(B)+tr(C)=tr(A+B+C); (AB)^T=B^TA^T; tr(AA^T)=tr(A^TA)=\|A\|_2^2;$
$tr(A^T)=tr(A);\ tr(\alpha A)=\alpha tr(A)$

Note that $\|\ \|_2$ is the Euclidean norm, α is a constant and (A_1, A_2, B, C, D) are weight matrices. Then:

$$\begin{aligned}\Delta L_2(k) &= \eta_{\max}^2\left[\left\|\Delta\hat{A}_{1(k)}\right\|^2+\left\|\Delta\hat{A}_{2(k)}\right\|^2+\left\|\Delta\hat{B}_{(k)}\right\|^2+\left\|\Delta\hat{C}_{(k)}\right\|^2+\left\|\Delta\hat{D}_{(k)}\right\|^2\right] \\ &+\alpha_{\max}^2\left[\left\|\Delta\hat{A}_{1(k-1)}\right\|^2+\left\|\Delta\hat{A}_{2(k-1)}\right\|^2+\left\|\Delta\hat{B}_{(k-1)}\right\|^2+\left\|\Delta\hat{C}_{(k-1)}\right\|^2+\left\|\Delta\hat{D}_{(k-1)}\right\|^2\right] \\ &+2\eta_{\max}tr\left(\tilde{A}_{1(k)}\Delta\hat{A}_{1(k)}^T+\tilde{A}_{2(k)}\Delta\hat{A}_{2(k)}^T+\tilde{B}_{(k)}\Delta\hat{B}_{(k)}^T+\tilde{C}_{(k)}\Delta\hat{C}_{(k)}^T+\tilde{D}_{(k)}\Delta\hat{D}_{(k)}^T\right) \\ &+2\alpha_{\max}tr\left(\tilde{A}_{1(k)}\Delta\hat{A}_{1(k-1)}^T+\tilde{A}_{2(k)}\Delta\hat{A}_{2(k-1)}^T+\tilde{B}_{(k)}\Delta\hat{B}_{(k-1)}^T+\tilde{C}_{(k)}\Delta\hat{C}_{(k-1)}^T+\tilde{D}_{(k)}\Delta\hat{D}_{(k-1)}^T\right) \\ &+2\alpha_{\max}\eta_{\max}tr\left(\Delta\hat{A}_{1(k)}\Delta\hat{A}_{1(k-1)}^T+\Delta\hat{A}_{2(k)}\Delta\hat{A}_{2(k-1)}^T+\Delta\hat{B}_{(k)}\Delta\hat{B}_{(k-1)}^T+\Delta\hat{C}_{(k)}\Delta\hat{C}_{(k-1)}^T+\Delta\hat{D}_{(k)}\Delta\hat{D}_{(k-1)}^T\right)\end{aligned}$$

So, due to the learning matrix law given by the equations (19)-(29), and collecting the errors as a common factor, using trace properties, we can rewrite $\Delta L_2(k)$ as:

$$\begin{aligned}\Delta L_2(k) &= \eta_{\max}^2\begin{bmatrix}\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{x}_{(k)}^T\right\|^2+\left\|\Phi'[\hat{y}_{(k)}]v^T{}_{(k)}\right\|^2+\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]u^T{}_{(k)}\right\|^2 \\ +\left\|\Phi'[\hat{y}_{(k)}]z^T{}_{(k)}\right\|^2+\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{y}_{(k)}\right\|^2\end{bmatrix}|e_{(k)}|^2 \\ &+\alpha_{\max}^2\begin{bmatrix}\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{x}_{(k-1)}^T\right\|^2+\left\|\Phi'[\hat{y}_{(k-1)}]v^T{}_{(k-1)}\right\|^2 \\ +\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]u^T{}_{(k-1)}\right\|^2+\left\|\Phi'[\hat{y}_{(k-1)}]z^T{}_{(k-1)}\right\|^2 \\ +\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{y}_{(k-1)}\right\|^2\end{bmatrix}|e_{(k-1)}|^2\end{aligned}$$

$$-2\eta_{\max} tr\left\{\begin{pmatrix} \tilde{A}_{1(k)}\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{x}^T_{(k)} + \tilde{A}_{2(k)}\Phi'[\hat{y}_{(k)}]v^T{}_{(k)} \\ +\tilde{B}_{(k)}\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]u^T{}_{(k)} + \tilde{C}_{(k)}\Phi'[\hat{y}_{(k)}]z^T{}_{(k)} \\ +\tilde{D}_{(k)}\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{y}_{(k)} \end{pmatrix} e_{(k)}\right\}$$

$$-2\alpha_{\max} tr\left\{\begin{pmatrix} \tilde{A}_{1(k)}\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{x}^T_{(k-1)} + \tilde{A}_{2(k)}\Phi'[\hat{y}_{(k-1)}]v^T{}_{(k-1)} \\ +\tilde{B}_{(k)}\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]u^T{}_{(k-1)} + \tilde{C}_{(k)}\Phi'[\hat{y}_{(k-1)}]z^T{}_{(k-1)} \\ +\tilde{D}_{(k)}\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{y}_{(k-1)} \end{pmatrix} e_{(k-1)}\right\}$$

$$+2\alpha_{\max}\eta_{\max} tr\left\{\begin{matrix} \Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]e_{(k)}\hat{x}^T_{(k)}\left(\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]e_{(k-1)}\hat{x}^T_{(k-1)}\right)^T \\ +\Phi'[\hat{y}_{(k)}]e_{(k)}v^T{}_{(k)}\left(\Phi'[\hat{y}_{(k-1)}]e_{(k-1)}v^T{}_{(k-1)}\right)^T \\ +\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]e_{(k)}u^T{}_{(k)}\left(\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]e_{(k-1)}u^T{}_{(k-1)}\right)^T \\ +\Phi'[\hat{y}_{(k)}]e_{(k)}z^T{}_{(k)}\left(\Phi'[\hat{y}_{(k-1)}]e_{(k-1)}z^T{}_{(k-1)}\right)^T \\ +\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]e_{(k)}\hat{y}_{(k)}\left(\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]e_{(k-1)}\hat{y}_{(k-1)}\right)^T \end{matrix}\right\}$$

Due to the error definition, collecting and put the following equation as a function of $e(k)$, we get:

$$\begin{pmatrix} \tilde{A}_{1(k)}\Gamma'\left[z_{(k)}\right]C_{(k)}\Phi'\left[\hat{y}_{(k)}\right]\hat{x}^T_{(k)} + \tilde{A}_{2(k)}\Phi'\left[\hat{y}_{(k)}\right]v^T{}_{(k)} + \\ \tilde{B}_{(k)}\Gamma'\left[z_{(k)}\right]C_{(k)}\Phi'\left[\hat{y}_{(k)}\right]u^T{}_{(k)} + \tilde{C}_{(k)}\Phi'\left[\hat{y}_{(k)}\right]z^T{}_{(k)} + \\ \tilde{D}_{(k)}\Gamma'\left[z_{(k)}\right]C_{(k)}\Phi'\left[\hat{y}_{(k)}\right]\hat{y}_{(k)} \end{pmatrix} = e_{(k)} - \underbrace{\begin{pmatrix} \Theta_{1(k)} + \Theta_{2(k)} + \Theta_{3(k)} + \Theta_{4(k)} \\ -\Delta f\left(x_{(k)}, u_{(k)}\right) \end{pmatrix} - O}_{\xi_{(k)}}$$

$$\begin{pmatrix} \tilde{A}_{1(k)}\Gamma'\left[z_{(k-1)}\right]C_{(k-1)}\Phi'\left[\hat{y}_{(k-1)}\right]\hat{x}^T_{(k-1)} \\ +\tilde{A}_{2(k)}\Phi'\left[\hat{y}_{(k-1)}\right]v^T{}_{(k-1)} \\ +\tilde{B}_{(k)}\Gamma'\left[z_{(k-1)}\right]C_{(k-1)}\Phi'\left[\hat{y}_{(k-1)}\right]u^T{}_{(k-1)} \\ +\tilde{C}_{(k)}\Phi'\left[\hat{y}_{(k-1)}\right]z^T{}_{(k-1)} \\ +\tilde{D}_{(k)}\Gamma'\left[z_{(k-1)}\right]C_{(k-1)}\Phi'\left[\hat{y}_{(k-1)}\right]\hat{y}_{(k-1)} \end{pmatrix} = e_{(k-1)} - \underbrace{\begin{pmatrix} \Theta_{1(k-1)} + \Theta_{2(k-1)} + \Theta_{3(k-1)} \\ +\Theta_{4(k-1)} - \Delta f\left(x_{(k-1)}, u_{(k-1)}\right) \end{pmatrix} - O}_{\xi_{(k-1)}}$$

First and second traces gave us four terms as:

$2\eta_{\max}|e(k)|^2 - 2\eta_{\max}\xi(k)e(k)$, and $2\eta_{\max}|e(k\text{-}1)|^2 - 2\eta_{\max}\xi(k\text{-}1)e(k\text{-}1)$

Using the following inequality [6], [7]: $X^TY + \left(X^TY\right)^T \le X^T\Lambda X + Y^T\Lambda^{-1}Y$, which is valid for any $X, Y \in \Re^{n\times m}$, and for any positive definite matrix $0 < \Lambda = \Lambda^T \in \Re^{n\times n}$, we obtained:

$$2\eta_{\max}e(k)\xi(k) = (\eta_{\max}e(k))\xi(k) + \xi(k)(\eta_{\max}e(k)) \le \eta^2_{\max}\|e(k)\|^2_{\Lambda 1} + \|\xi(k)\|^2_{\Lambda_1^{-1}}$$

$$2\alpha_{\max}e(k\text{-}1)\xi(k\text{-}1) = (\alpha_{\max}e(k\text{-}1))\xi(k\text{-}1) + \xi(k\text{-}1)(\alpha_{\max}e(k\text{-}1)) \le \alpha^2_{\max}\|e(k\text{-}1)\|^2_{\Lambda 2} + \|\xi(k\text{-}1)\|^2_{\Lambda_2^{-1}}$$

Analyzing term by term and applying the Rayleigh inequality:

$$\lambda_{\min}(\Lambda)\|x\|^2 \le x^T\Lambda x \le \lambda_{\max}(\Lambda)\|x\|^2$$

we could obtain:

$$\Delta L_2(k) \le \eta_{\max}\left\{\eta_{\max}\begin{bmatrix} \left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{x}^T_{(k)}\right\|^2(\lambda_{\max}(\Lambda_3)+1) \\ +\left\|\Phi'[\hat{y}_{(k)}]v^T{}_{(k)}\right\|^2(\lambda_{\max}(\Lambda_4)+1) \\ +\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]u^T{}_{(k)}\right\|^2(\lambda_{\max}(\Lambda_5)+1) \\ +\left\|\Phi'[\hat{y}_{(k)}]z^T{}_{(k)}\right\|^2(\lambda_{\max}(\Lambda_6)+1) \\ +\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{y}_{(k)}\right\|^2(\lambda_{\max}(\Lambda_7)+1) + \lambda_{\max}(\Lambda_1) \end{bmatrix} - 2\right\}|e_{(k)}|^2$$

$$+\alpha_{\max}\left\{\alpha_{\max}\begin{bmatrix}\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{x}^T_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_3^{-1}\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k-1)}]v^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_4^{-1}\right)+1\right)\\+\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]u^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_5^{-1}\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k-1)}]z^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_6^{-1}\right)+1\right)\\+\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{y}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_7^{-1}\right)+1\right)+\lambda_{\max}\left(\Lambda_2\right)\end{bmatrix}-2\right\}|e_{(k-1)}|^2$$

$$+\left\|\xi_{(k)}\right\|^2_{\Lambda_1^{-1}}+\left\|\xi_{(k-1)}\right\|^2_{\Lambda_2^{-1}}$$

Now, making inner terms equal to one as in the unit circle condition for discrete time, as:

$$\eta_{\max}\begin{bmatrix}\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{x}^T_{(k)}\right\|^2\left(\lambda_{\max}\left(\Lambda_3\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k)}]v^T{}_{(k)}\right\|^2\left(\lambda_{\max}\left(\Lambda_4\right)+1\right)\\+\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]u^T{}_{(k)}\right\|^2\left(\lambda_{\max}\left(\Lambda_5\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k)}]z^T{}_{(k)}\right\|^2\left(\lambda_{\max}\left(\Lambda_6\right)+1\right)\\+\left\|\Gamma'[z_{(k)}]C_{(k)}\Phi'[\hat{y}_{(k)}]\hat{y}_{(k)}\right\|^2\left(\lambda_{\max}\left(\Lambda_7\right)+1\right)+\lambda_{\max}\left(\Lambda_1\right)\end{bmatrix}=1$$

$$\alpha_{\max}\begin{bmatrix}\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{x}^T_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_3^{-1}\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k-1)}]v^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_4^{-1}\right)+1\right)\\+\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]u^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_5^{-1}\right)+1\right)\\+\left\|\Phi'[\hat{y}_{(k-1)}]z^T{}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_6^{-1}\right)+1\right)\\+\left\|\Gamma'[z_{(k-1)}]C_{(k-1)}\Phi'[\hat{y}_{(k-1)}]\hat{y}_{(k-1)}\right\|^2\left(\lambda_{\max}\left(\Lambda_7^{-1}\right)+1\right)+\lambda_{\max}\left(\Lambda_2\right)\end{bmatrix}=1$$

At last we get the final condition:

$$\Delta L_2(k)\le -\eta_{\max}|e(k)|^2-\alpha_{\max}|e(k-1)|^2+d(k) \qquad \text{(A.25)}$$

Where: the unmodeled dynamics and/or perturbations term $d(k)$ is given by:

$$d(k)=\|\xi(k)\|^2_{\Lambda_1^{-1}}+\|\xi(k-1)\|^2_{\Lambda_2^{-1}} \qquad \text{(A.26)}$$

Applying the Lemma of the KF RNN rate of convergence, [6], [7], for the result (A.26) we could conclude that: the d - term must be bounded by the weight matrices and the learning parameter, in order to obtain the final result:
$\Delta L_2(k)\in L_\infty$
As a consequence:
From equations (A.22) and (A.25) we easily could get the equation (20). Therefore the boundedness of the $L(k)$, $k\in Z_0^+$ is guaranteed.
$A_{(k)}\in L_\infty, B_{(k)}\in L_\infty, C_{(k)}\in L_\infty$

Lemma of KF RNN rate of convergence.
Applying the limit's definition, the identification error bound condition is obtained as:

$$\overline{\lim_{t\to\infty}}\frac{1}{k}\sum_{t=1}^{k}\left(|e_t|^2+|e_{t-1}|^2\right)\le d$$

Proof.
Starting from the final result of the Theorem of BP KF RNN stability:
$\Delta L_2(k)\le -\eta_{\max}|e(k)|^2-\alpha_{\max}|e(k-1)|^2+d$
After an analysis of the iterations from $k=0$, we get:

$$L_2(k+1)-L_2(0)\le -\sum_{t=1}^{k}|e_t|^2-\sum_{t=1}^{k}|e_{t-1}|^2+dk \qquad \sum_{t=1}^{k}\left(|e_t|^2+|e_{t-1}|^2\right)\le dk-L_2(k+1)+L_2(0)\le dk+L_2(0)$$

Dividing by k and applying the limit's definition, the identification error bound condition is obtained in the final form:

$$\overline{\lim_{t\to\infty}}\frac{1}{k}\sum_{t=1}^{k}\left(|e_t|^2+|e_{t-1}|^2\right)\le d$$

From here we could see that the term d must be bounded by weight matrices and the learning parameter, in order to obtain:
$\Delta L_2(k)\in L_\infty$