NEURAL NETWORK BASED HAMMERSTEIN SYSTEM IDENTIFICATION USING PARTICLE SWARM SUBSPACE ALGORITHM

S.Z. Rizvi, H.N. Al-Duwaish
Department of Electrical Engineering,
King Fahd Univ. of Petroleum & Minerals, Dhahran, Saudi Arabia
srizvi@kfupm.edu.sa, hduwaish@kfupm.edu.sa

Keywords: Particle Swarm Optimization, Neural Network Training, Subspace Identification, Static Nonlinearity, Dynamic Linearity, Radial Basis Function (RBF) Neural Networks.

Abstract: This paper presents a new method for modeling of Hammerstein systems. The developed identification method uses state-space model in cascade with radial basis function (RBF) neural network. A recursive algorithm is developed for estimating neural network synaptic weights and parameters of the state-space model. No assumption on the structure of nonlinearity is made. The proposed algorithm works under the weak assumption of richness of inputs. The problem of modeling is solved as an optimization problem and Particle Swarm Optimization (PSO) is used for neural network training. Performance of the algorithm is evaluated in the presence of noisy data and Monte-Carlo simulations are performed to ensure reliability and repeatability of the identification technique.

1 INTRODUCTION

The Hammerstein Model belongs to the family of block oriented models and is made up of a memoryless nonlinear part followed by a linear dynamic part as shown in Figure 1. It has been known to effectively represent and approximate several nonlinear dynamic industrial processes, for example pH neutralization process (Fruzzetti, K.P., Palazoglu A., McDonald, K.A., 1997), heat exchanger (Eskinat, E., Johnson, S.H., Luyben, W.L., 1991), nonlinear filters (Haddad, A.H., Thomas, J.B., 1968), and water heater (Abonyi, I., Nagy, L., Szeifert, E., 2000).

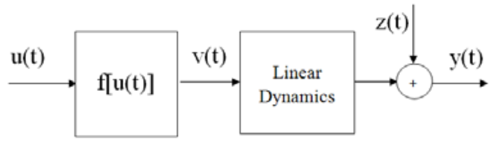

Figure 1: Block diagram of a Hammerstein model.

A lot of research has been carried out on identification of Hammerstein models. Hammerstein Systems can be modeled by employing either nonparametric or parametric models. Nonparametric models represent the system in terms of curves resulting from expansion of series such as the Volterra series or kernel regression. Parametric representations are more compact having fewer parameters. Notable parametric identification techniques can be found in (Narendra, K.S., Gallman, P., 1966), (Billings, S., 1980), (Al-Duwaish, H., 2001), (Vörös, J., 2002), (Wenxiao, Z., 2007) and in references therein. Nonparametric identification techniques can be found in several papers including, but not limited to those of (Greblicki, W., 1989), (Al-Duwaish, H., Nazmulkarim, M., Chandrasekar, V., 1997), (Zhao, W., Chen, H., 2006).

Recently, subspace identification has emerged as a well known method for identification of linear systems. It is computationally less complicated as compared to conventional prediction error methods (PEM), does not require initial estimate of a canonical model like PEM and, is easily extendable to systems having multiple inputs and outputs (Katayama, T., 2005). However, its use is restricted mostly to linear systems. To make use of this, attempts have been made to extend subspace linear identification to nonlinear systems such as Wiener and Hammerstein systems including use of static nonlinearity in the feedback path (Luo, D., Leonessa, A., 2002), assuming known nonlinearity structures (Verhaegen, M., Westwick, D., 1996), and using least squares support vector machines (Goethals, I., Pelckmans, K., Suykens, J.A.K., Moor, B.D., 2005).

In this paper, a new subspace based method is proposed for Hammerstein model identification, which uses radial basis function (RBF) network in cascade with a state-space model. A recursive algorithm is developed for parameter estimation of the two subsystems.

The paper is arranged as follows. Section 2 looks at the model structure proposed in this work. Section 3 takes a detailed look at the proposed identification scheme. Section 4 describes the proposed algorithm in detail and section 5 includes numerical examples, their results, and analysis.

Throughout this paper, the following convention is used for notations. Lower case variables represent scalars. Lower case bold variables represent vectors. Upper case bold letters denote matrices. The only exception to this convention is the choice of variable for the cost function, where a more conventional J is used to define the cost function.

2 PROPOSED MODEL STRUCTURE

The proposed model structure in this work uses state-space model to estimate the linear dynamic part. The memoryless nonlinear part is modeled using an RBF network. An RBF network is an effective type of neural network that has proved useful in applications like function approximation and pattern recognition. A typical three layer RBF network is shown in Figure 2. The input layer connects the network to its environment. The second layer, known as the hidden layer, performs a fixed nonlinear transformation using basis functions. The output layer linearly weighs the response of the network to the output (Haykin, S., 1999). The external inputs to the system $u(t)$ are fed to the RBF network, which generates the outputs $v(t)$. Considering an RBF network having q number of neurons in the hidden layer, the basis vector is

$$\phi(t) = [\phi\|u(t) - c_1\| \cdots \phi\|u(t) - c_q\|],$$

where c_i is the chosen center for the i^{th} neuron, $\|.\|$ denotes norm that is usually Euclidean, and ϕ_i is

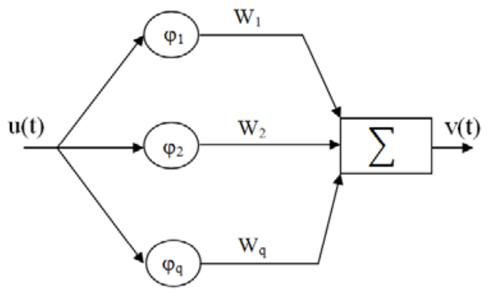

Figure 2: An RBF neural network with q neurons in the hidden layer.

the nonlinear radial basis function for the i^{th} neuron, given by

$$\phi_i(t) = exp\left(-\frac{\|u(t) - c_i\|^2}{2\sigma^2}\right),$$

where σ is the spread of the Gaussian function $\phi_i(t)$. If the set of output layer weights of the RBF network is given by

$$\mathbf{w} = [w_1 \quad w_2 \cdots w_q],$$

the RBF output $v(t)$ is given by

$$v(t) = \mathbf{w}\phi^T(t). \tag{1}$$

Considering a system with a single input and output, the output of the RBF network $v(t)$ in turn acts as input to the state-space model translating it into final output $y(t)$. The equation for $y(t)$ is given by discrete time state-space equation

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}v(t) + \mathbf{w}(t), & (2)\\ y(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}v(t) + z(t), & (3) \end{aligned}$$

where $v(t)$ and $y(t)$ are input and output of the state-space system at discrete time instant (t), $z(t)$ and $\mathbf{w}(t)$ are the measurement and process noise.

3 PROPOSED IDENTIFICATION SCHEME

The problem of Hammerstein modeling is therefore formulated as follows. Given a set of m measurements of noisy inputs $u(t)$ and outputs $y(t)$, the problem is reduced to finding the weights of the RBF network and the matrices of the state-space model.

For the estimation of state-space matrices, N4SID numerical algorithm for subspace identification

(Overschee, P.V., Moor, B.D., 1994) is used. The algorithm determines the order n of the system, the system matrices $\mathbf{A} \varepsilon \Re^{n\times n}$, $\mathbf{B} \varepsilon \Re^{n\times p}$, $\mathbf{C} \varepsilon \Re^{r\times n}$, $\mathbf{D} \varepsilon \Re^{r\times p}$, covariance matrices $\mathbf{Q} \varepsilon \Re^{n\times n}$, $\mathbf{R} \varepsilon \Re^{1\times 1}$, $\mathbf{S} \varepsilon \Re^{n\times 1}$, and the Kalman gain matrix $\mathbf{K}$, where p denotes the number of inputs and r denotes the number of outputs of the system, without any prior knowledge of the structure of the system, given that a large number of measurements of inputs and outputs generated by the unknown system of equations (2) and (3) is provided. In N4SID, Kalman filter states are first estimated directly from input and output data, then the system matrices are obtained (Overschee, P.V., Moor, B.D., 1994).

For Hammerstein identification problem, it is desired that the error between the output of the actual system, and that of the estimated model be minimized. Therefore, in a way this becomes an optimization problem where a cost index is to be minimized. For the system described in equations (1)-(3), the cost index is given by

$$J = \sum_{t=1}^{m} e^2(t) = \sum_{t=1}^{m} (y(t) - \hat{y}(t))^2, \qquad (4)$$

where $y(t)$ and $\hat{y}(t)$ are the outputs of the actual and estimated systems at time instant (t). The weights of the RBF network are therefore updated so as to minimize this cost index. For this purpose, particle swarm optimization (PSO) used.

PSO is a heuristic optimization algorithm which works on the principle of swarm intelligence (Kennedy, J., Eberhart, R., 2001). It imitates animals living in a swarm collaboratively working to find their food or habitat. In PSO, the search is directed, as every particle position is updated in the direction of the optimal solution. It is robust and fast and can solve most complex and nonlinear problems. It generates globally optimum solutions and exhibits stable convergence characteristics. In this work, PSO is used to train the RBF network. Each particle of the swarm represents a candidate value for the weight of the output layer of RBF network. The fitness of the particles is the reciprocal of the cost index given in equation (4). Hence, the smaller the sum of output errors, the more fit are the particles. Based on this principle, PSO updates the position of all the particles moving towards an optimal solution for the weights of RBF neural network.

The i^{th} particle of the swarm is given by a k-dimension vector $\tilde{\mathbf{x}}_i = [\tilde{x}_{i1} \cdots \tilde{x}_{ik}]$, where k denotes the number of optimized parameters. Similar vectors $\tilde{\mathbf{p}}_i$ and $\tilde{\mathbf{v}}_i$ denote the best position and velocity of the i^{th} particle respectively. The velocity of the i^{th} particle is updated as

$$\begin{aligned}\tilde{\mathbf{v}}_i(t+1) = \chi[w\tilde{\mathbf{v}}_i(t) + c_1 r_1(t)\{\tilde{\mathbf{p}}_i(t) - \tilde{\mathbf{x}}_i(t)\} \\ + c_2 r_2(t)\{\tilde{\mathbf{p}}_g(t) - \tilde{\mathbf{x}}_i(t)\}],\end{aligned} \qquad (5)$$

and the particle position is updated as

$$\tilde{\mathbf{x}}_i(t+1) = \tilde{\mathbf{x}}_i(t) + \tilde{\mathbf{v}}_i(t+1). \qquad (6)$$

In the above equations, $\tilde{\mathbf{p}}_g$ denotes global best positions, while c_1 and c_2 are the *cognitive* and *social* parameters respectively, and are both positive constants. Parameter w is the *inertia weight* and χ is called the *constriction factor* (Eberhart, R., Shi, Y., 1998). The value of cognitive parameter c_1 signifies a particle's attraction to a local best position based on its past experiences. The value of social parameter c_2 determines the swarm's attraction towards a global best position.

4 TRAINING ALGORITHM

Given a set of m observations of input and output, $\mathbf{u}\varepsilon\Re^{1\times m}$ and $\mathbf{y}\varepsilon\Re^{1\times m}$, a hybrid PSO/Subspace identification algorithm is proposed below based on minimization of output error given in equation (4).

1. Estimate state-space matrices $\mathbf{A}_0$, $\mathbf{B}_0$, $\mathbf{C}_0$ and $\mathbf{D}_0$ (initial estimate) from original non linear data using N4SID.
2. Iteration = $k = 1$.
3. Initialize PSO with random population of possible RBF network weights.
4. $\mathbf{w}_k = min_{\mathbf{w}\varepsilon\Re^q}\ J\ (\mathbf{A}_{k-1}, \mathbf{B}_{k-1}, \mathbf{C}_{k-1}, \mathbf{D}_{k-1}, \mathbf{w})$.
5. Estimate set of RBF neural network outputs $\mathbf{v}_k\varepsilon\Re^{1\times m}$

$$\begin{aligned}\mathbf{v}_k &= \mathbf{w}_k \Phi^T \\ &= \left[w_{1k} \cdots w_{qk}\right] \begin{bmatrix} \phi(1)_1 \cdots \phi(1)_q \\ \vdots \\ \phi(m)_1 \cdots \phi(m)_q \end{bmatrix}^T \\ &= \left[w_{1k} \cdots w_{qk}\right] \begin{bmatrix} \phi(1) \\ \vdots \\ \phi(m) \end{bmatrix}^T .\end{aligned}$$

6. Estimate state space matrices $\mathbf{A}_k$, $\mathbf{B}_k$, $\mathbf{C}_k$ and $\mathbf{D}_k$ from $[\mathbf{v}_k, \mathbf{y}]$. This estimate of state-space model would be an improvement on the previous estimate.
7. Regenerate $\hat{\mathbf{y}}_k\varepsilon\Re^{1\times m}$.
8. If minimum goal is not achieved, iteration = $k+1$. Repeat steps 3 to 7.

5 SIMULATION RESULTS

5.1 Example 1

The first example considers the following Hammerstein type nonlinear process whose static nonlinearity is given by

$$v(t) = sign\,(u(t))\,\sqrt{|u(t)|}. \tag{7}$$

The dynamic linear part is given by a third order discrete time state-space system

$$\mathbf{A} = \begin{bmatrix} 1.80 & 1 & 0 \\ -1.07 & 0 & 1 \\ 0.21 & 0 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 4.80 \\ 1.93 \\ 1.21 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

The eigen values of the linear subsystem lie at $\lambda_1 = 0.7$, $\lambda_2 = 0.6$, and $\lambda_3 = 0.5$. Desired outputs are generated by exciting the process model with a rich set of uniformly distributed random numbers in the interval $[-1.75, 1.75]$. An RBF network of 10 neurons is initialized with random synaptic weights and centers uniformly distributed in the input interval. PSO social and cognitive parameters c_1 and c_2 are kept almost equal to each other with c_1 slightly larger than c_2 and $c_1 + c_2 \geq 4$ as proposed in (Carlisle, A., Dozier, G., 2001). This allow trusting past experiences as well as ample exploration of the swarm for a global best solution. Constriction factor is kept close to 1 to enable slow convergence with better exploration. Number of particles amount to 10 for the synaptic weights of 10 neurons. A swarm population size of 50 is selected and the optimization process is run for 100 iterations.

The algorithm shows promising results and mean squared output error between normalized outputs of actual and estimated systems converges to a final value of 4×10^{-4} in 24 iterations of the algorithm. Figure 3 shows the nonlinearity estimate. Convergence of mean squared error is shown in Figure 5. An easy way to evaluate the estimate of linear dynamic part lies in comparing the eigen values of the estimated system with true ones. The eigen values of the estimated system lie at $\hat{\lambda}_1 = 0.72, \hat{\lambda}_2 = 0.53$ and $\hat{\lambda}_3 = 0.53$. Figure 4 shows the step response of the dynamic linear part.

To evaluate the performance of the proposed algorithm in noisy environment, zero mean Gaussian additive noise is included at the output of the system such that the signal to noise ratio (SNR) is 10dB. The algorithm performs well in estimating the system despite low output SNR. The final mean squared error

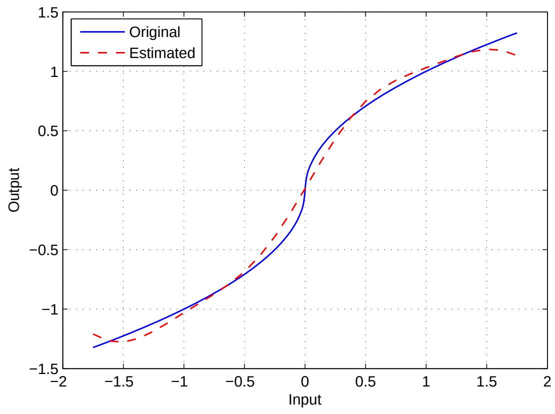

Figure 3: Estimate of square root nonlinearity of example 1.

converges to 1.4×10^{-3} in 30 iterations. Nonlinearity estimate is shown in Figure 6. The eigen values of the estimated system lie at $\hat{\lambda}_1^{10dB} = 0.73$, $\hat{\lambda}_2^{10dB} = 0.73$ and $\hat{\lambda}_1^{10dB} = 0.58$.

The results presented above are obtained from a single run of estimation algorithm. To further ensure the reliability and repeatability of the algorithm, Monte-Carlo simulation is carried out and ensemble statistics are tabulated in Table 1. The statistics show encouraging convergence of normalized output squared error and estimation of linear subsystem. Parameters of the nonlinearity cannot be compared because of the nonparametric nature of estimation. At best, the estimates of nonlinearity can be judged from the shapes of estimated nonlinear function as presented in Figures 3 and 6.

5.2 Example 2

The second example considers the following Hammerstein type nonlinear process whose static nonlinearity is given by

$$\begin{aligned} v(t) &= tanh\,[2u(t)] && 1.5 \geq u(t), \\ v(t) &= \frac{exp(u(t)) - 1}{exp(u(t)) + 1} && 4 > u(t) > 1.5. \end{aligned}$$

The dynamic linear part is given by the following second order discrete time state-space system

$$\mathbf{A} = \begin{bmatrix} 1.0 & 1.0 \\ -0.5 & 0.0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

The linear part of the system has eigen values at $\lambda_{1,2} = 0.5 \pm 0.5i$. Desired outputs are generated by

Table 1: Monte-Carlo simulation statistics for example 1

Estimation results without output noise	
Total number of runs	200
Magnitude of actual eigen value λ_1 of linear subsystem	0.7
Mean magnitude of estimated eigen value $\hat{\lambda}_1$ of linear subsystem	0.713
Variance of eigen value estimate $\hat{\lambda}_1$	8×10^{-4}
Magnitude of actual eigen value λ_2 of linear subsystem	0.6
Mean magnitude of estimated eigen value $\hat{\lambda}_2$ of linear subsystem	0.62
Variance of eigen value estimate $\hat{\lambda}_2$	4.5×10^{-3}
Magnitude of actual eigen value λ_3 of linear subsystem	0.5
Mean magnitude of estimated eigen value $\hat{\lambda}_3$ of linear subsystem	0.481
Variance of eigen value estimate $\hat{\lambda}_3$	5.7×10^{-3}
Average number of iterations required for every run	8.26
Average mean squared output error (MSE)	9.8×10^{-4}
Estimation results with output SNR 10dB	
Total number of runs	200
Magnitude of actual eigen value λ_1 of linear subsystem	0.7
Mean magnitude of estimated eigen value $\hat{\lambda}_1$ of linear subsystem	0.75
Variance of eigen value estimate $\hat{\lambda}_1$	1.6×10^{-3}
Magnitude of actual eigen value λ_2 of linear subsystem	0.6
Mean magnitude of estimated eigen value $\hat{\lambda}_2$ of linear subsystem	0.68
Variance of eigen value estimate $\hat{\lambda}_2$	6×10^{-3}
Magnitude of actual eigen value λ_3 of linear subsystem	0.5
Mean magnitude of estimated eigen value $\hat{\lambda}_3$ of linear subsystem	0.4
Variance of eigen value estimate $\hat{\lambda}_3$	6×10^{-2}
Average number of iterations required for every run	8.02
Average mean squared output error (MSE)	6.6×10^{-3}

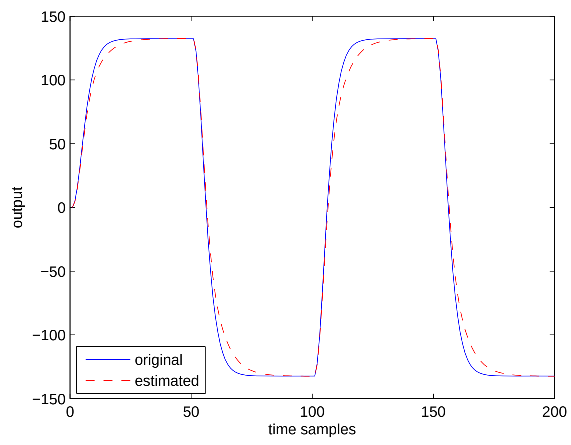

Figure 4: Step response of linear dynamic part of example 1.

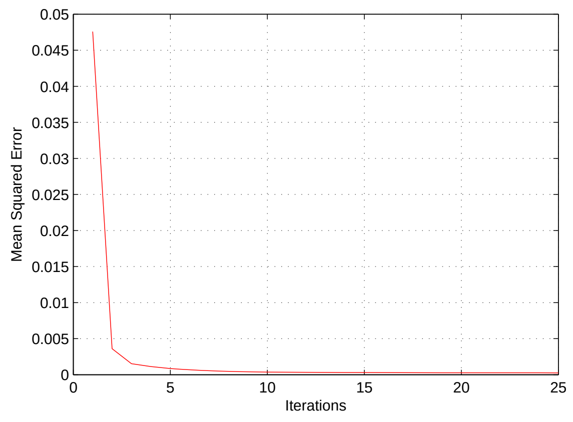

Figure 5: Mean squared error for example 1.

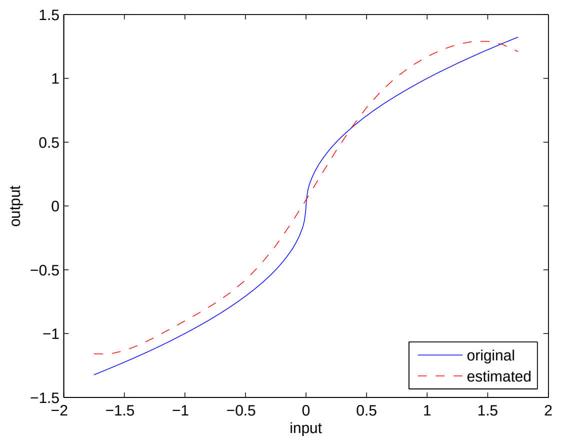

Figure 6: Estimate of square root nonlinearity of example 1 with output SNR 10dB.

exciting the process model with a rich set of uniformly distributed random numbers in the interval $[0,4]$. An RBF network with 25 neurons is initialized with random synaptic weights and uniformly distributed centers chosen within the input interval. PSO social and cognitive parameters and constriction factor are kept similar to example 1. The number of particles is equal to the number of neurons and a population size of 50 gives good results again. The algorithm performs well and estimates the system in 20 iterations. The mean squared output error between normalized values of actual and estimated outputs converges to a final value of 8×10^{-4}. The estimate of nonlinearity shape is shown in Figure 7. Eigen values of the estimated system lie at $\hat{\lambda}_{1,2} = 0.497 \pm 0.5i$. Step response of linear subsystem is shown in Figure 8. The squared error convergence plot is shown in Figure 9.

Estimation is then carried out in noisy environment, with zero mean Gaussian additive noise included at the output of the system such that the signal to noise ratio (SNR) is 10dB. The algorithm performs well in noisy environment. The final mean squared error converges to 1.8×10^{-3} in 30 iterations. Nonlinearity estimate is shown in Figure 10. The eigen values of the estimated system lie at $\hat{\lambda}^{10dB}_{1,2} = 0.493 \pm 0.499i$.

Table 2 shows ensemble statistics of Monte-Carlo simulation for example 2. The statistics show encouraging convergence of normalized output squared error and estimation of linear subsystem. As mentioned before, parameters of the nonlinearity cannot be compared due to the nonparametric nature of estimation. At best, the estimates of nonlinearity can be judged from the shapes of estimated nonlinear function as presented in Figures 7 and 10.

6 CONCLUSION

The PSO/Subspace algorithm is basically a combination of PSO and Subspace N4SID algorithm, and hence its convergence properties are directly related to the convergence properties of PSO and Subspace algorithms. PSO has been usually known to perform better than most evolutionary algorithms (EA)s in finding global optimum provided its parameters are tuned properly according to the application. The subspace algorithm is also known for having no convergence problems. Its numerical robustness is guaranteed because of well understood linear algebra techniques like QR decomposition and singular value decomposition (SVD). As a consequence, it does not experience problems like lack of convergence, slow con-

Table 2: Monte-Carlo simulation statistics for example 2

Estimation results without output noise	
Total number of runs	200
Magnitude of actual eigen values $\lambda_{1,2}$ of linear subsystem	0.7071
Mean magnitude of estimated eigen values $\hat{\lambda}_{1,2}$ of linear subsystem	0.7071
Variance of eigen value estimates	2×10^{-5}
Average number of iterations required for every run	9
Average mean squared output error (MSE)	7×10^{-4}
Estimation results with output SNR 10dB	
Total number of runs	200
Magnitude of actual eigen values $\lambda_{1,2}$ of linear subsystem	0.7071
Mean magnitude of estimated eigen values $\hat{\lambda}_{1,2}$ of linear subsystem	0.7079
Variance of eigen value estimates	6×10^{-5}
Average number of iterations required for every run	21
Average mean squared output error (MSE)	2×10^{-3}

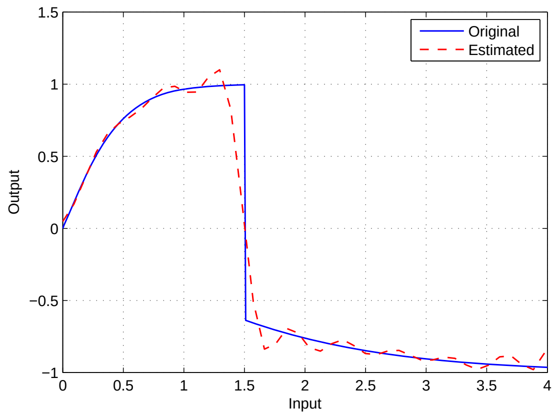

Figure 7: Estimate of tangent-hyperbolic nonlinearity of example 2.

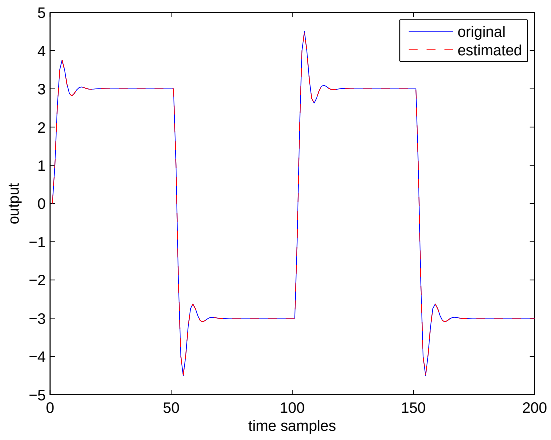

Figure 8: Step response of linear dynamic part of example 2.

vergence or numerical instability (Overschee, P.V., Moor, B.D., 1994). Moreover, in order to assure repeatability and reliability of the proposed algorithm, Monte-Carlo simulations have been carried out. The ensemble statistics presented in Tables 1 and 2 show strong convergence and consistent performance of the proposed algorithm.

The effect of noise is also studied, and the algorithm is seen to converge sufficiently in the presence of noisy data as shown in the simulation results.

ACKNOWLEDGEMENT

The authors would like to acknowledge the support of King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia.

REFERENCES

Abonyi, I., Nagy, L., Szeifert, E. (2000). Hybrid fuzzy convolution modelling and identification of chemical process systems. In *International Journal Systems Science*. volume 31, pages 457-466.

Al-Duwaish, H. (2001). A genetic approach to the identification of linear dynamical systems with static nonlin-

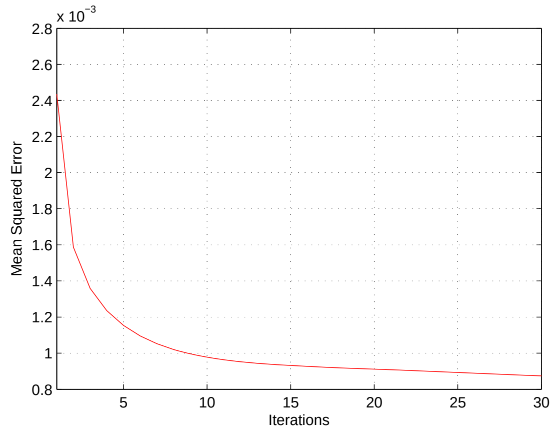

Figure 9: Mean squared error for example 2.

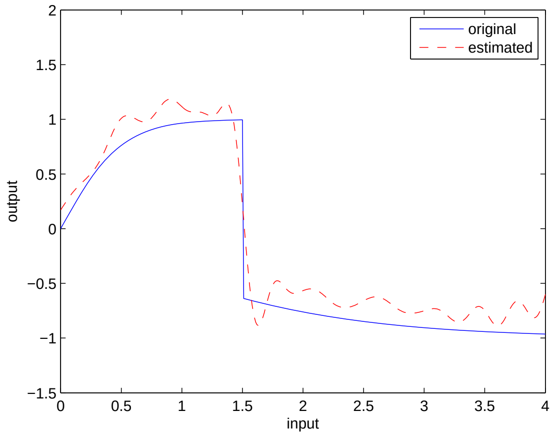

Figure 10: Estimate of tangent-hyperbolic nonlinearity of example 2 with output SNR 10dB.

earities. In *International Journal of Systems Science*. volume 31, pages 307-314.

Al-Duwaish, H., Nazmulkarim, M., Chandrasekar, V. (1997). Hammerstein model identification by multi-layer feedforward neural networks. In *International Journal Systems Science*. volume 28, pages 49-54.

Billings, S. (1980). Identification of nonlinear systems - a survey. In *IEE Proceedings*. volume 127, pages 272-285.

Carlisle, A., Dozier, G. (2001). An off-the-shelf pso. In *In Proceedings of the Particle Swarm Optimization Workshop*. pages 1-6.

Eberhart, R., Shi, Y. (1998). Parameter selection in particle swarm optimisation. In *Evolutionary Programming VII*. pages 591-600.

Eskinat, E., Johnson, S.H., Luyben, W.L. (1991). Use of hammerstein models in identification of nonlinear systems. In *AIChE Journal*. volume 37, pages 255-268.

Fruzzetti, K.P., Palazoglu A., McDonald, K.A. (1997). Nonlinear model predictive control using hammerstein models. In *J. Proc. Control*. vol. 7, page 31-41.

Goethals, I., Pelckmans, K., Suykens, J.A.K., Moor, B.D. (2005). Identification of mimo hammerstein models using least squares support vector machines. In *Automatica*. volume 41, pages 1263-1272.

Greblicki, W. (1989). Non-parametric orthogonal series identification of hammerstein systems. In *International Journal Systems Science*. volume 20, pages 2335-2367.

Haddad, A.H., Thomas, J.B. (1968). On optimal and sub-optimal nonlinear filters for discrete inputs. In *IEEE Transaction on Information Theory*. volume 14, pages 16-21.

Haykin, S. (1999). *Neural Networks - A Comprehensive Foundation*. Prentice-Hall, Second Edition.

Katayama, T. (2005). *Subspace Methods for System Identification*. Springer-Verlag, London.

Kennedy, J., Eberhart, R. (2001). *Swarm Intelligence*. Academic Press.

Luo, D., Leonessa, A. (2002). Identification of mimo hammerstein systems with nonlinear feedback. In *Proceedings of American Control Conference, Galesburg, USA*. pages 3666-3671.

Narendra, K.S., Gallman, P. (1966). An iterative method for the identification of nonlinear systems using hammerstein model. In *IEEE Transaction on Automatic Control*. volume 11, pages 546-550.

Overschee, P.V., Moor, B.D. (1994). N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems. In *Automatica*. volume 30, pages 75-93.

Vörös, J. (2002). Modeling and paramter identification of systems with multisegment piecewise-linear characteristics. In *IEEE Transaction on Automatic Control*. volume 47, pages 184-188.

Verhaegen, M., Westwick, D. (1996). Identifying mimo hammerstein systems in the context of subspace model identification methods. In *International Journal Control*. volume 63, pages 331-349.

Wenxiao, Z. (2007). Identification for hammerstein systems using extended least squares algorithm. In *Proceedings of 26th Chinese Control Conference, China,*. pages 241-245.

Zhao, W., Chen, H. (2006). Recursive identification for hammerstein systems with arx subsystem. In *IEEE Transaction on Automatic Control*. volume 51, pages 1966-1974.