

Adaptive Dynamic Neural Network Estimators

D. G. Lainiotis, K. N. Plataniotis
Florida Institute of Technology
Department of Electrical and Computer Engineering
150 W. University Blvd., Melbourne Fl. 32901, USA

Abstract - The problem of state estimation for linear or nonlinear models with unknown parameters is very important in many engineering problems. As such it has been addressed extensively through the use of statistical methodologies i.e. ALF, EKF, etc. In this paper the solution to the problem of adaptive estimation for unknown state variable or chaotic models through the use of adaptive dynamic neural estimators is proposed. The proposed adaptive neural estimators are developed and their advantages are discussed. Extensive computer simulations of the application of the proposed adaptive neural estimator to state estimation as well as chaotic series prediction illustrate the effectiveness of the adaptive neural solution.

I. INTRODUCTION

Estimation theory (filtering) has received considerable attention in the past four decades because of its practical significance in solving engineering and scientific problems. As a result of the combined research efforts of many scientists in the field numerous filtering algorithms, have been developed through the use of statistical methodologies. These can be classified in two major categories, namely linear and nonlinear filtering algorithms, corresponding to linear (or linearized) physical dynamic models, and nonlinear physical models [1]-[8].

The two more serious problems facing the systems engineering community nowadays is model uncertainty and nonlinearity [2],[3],[7]. In real world situations it is not possible to represent, adequately, system characteristics such as nonlinearity, time delay and/or time varying parameters. The need to deal with increasingly complex nonlinear systems, and the tremendous increase in computing power has recently led to a reevaluation of the conventional estimation methods. Nonlinear and adaptive filtering has been established as an important theoretical and applied research area [7]. The need for new estimation methods that can handle more realistic assumptions, and can take advantage of the new hardware capabilities is quite apparent [10].

A great deal of effort has been devoted in the past for the problem of adaptive nonlinear estimation. Numerous filters have been obtained using statistical methodologies, and extending the basic linear theory to nonlinear as well as adaptive systems. Among them the so called Extended Kalman Fil-

ter (EKF) [1], [7], and the Adaptive Lainiotis Filter (ALF) [2]-[9] are the most notable. However, a generally applicable optimal nonlinear filter and its practical implementation are not known.

Recently, the emerging technology of neural networks [9]-[12] has been successfully applied to the solution of the adaptive estimation problem [13]-[21]. Given the difficulties the conventional estimators encounter, neural solutions constitute a unique and novel alternative [21]. The trained multilayer perceptrons with their massive parallelism, capability to approximate arbitrary continuous functions, appear to offer a new promising tool in estimation theory.

In this paper a general framework for neural estimators is proposed. Trained multilayer networks are integrated with traditional adaptive techniques to design a neural adaptive filter capable of providing robust and accurate solutions to a broad class of adaptive and nonlinear problems.

The paper is organized as follows: In section II a short description of the adaptive estimation problem for linear and nonlinear state space models as well as for chaotic signals are given. The structure of the neural network as state estimator is also discussed followed by the proposed framework for the adaptive neural estimators. Section III presents the first part of the experimental results where the effectiveness of the proposed methodology is illustrated via simulations. In this section the adaptive neural estimator is used to predict the values of an partially unknown chaotic system. Section IV demonstrates the robustness of the new method as state estimator using a series of adaptive state space models. Finally, section V contains a summary of this work.

II. ADAPTIVE SYSTEMS

In general, the state space model for an adaptive stochastic nonlinear system in discrete time has the following form

$$x(k+1) = f(k, x(k), u(k), \theta(k), w(k)) \quad (1)$$

$$z(k) = h(k, x(k), \theta(k), v(k)) \quad (2)$$

where

$x(k)$ is the n dimensional state of the system which the initial value $x(0)$ is coming from a known probability density function

$u(k)$ is the system's deterministic input vector

$z(k)$ is the system's output vector (measurement)

$f(), h()$ are in general arbitrary non linear functions

$w(k), v(k)$ are stochastic inputs to the system, plant and measurement noises

$\theta(k)$ is the unknown parameter vector, that summarizes the uncertainty in the above system

From the above state space model it can be seen that in a discrete adaptive system there are two mapping for a neural network to identify. One is the mapping $f()$ which maps the past system state and the inputs (deterministic and stochastic) to the new state $x(k+1)$. The second is mapping $h()$ which transforms the state $x(k)$ into the measurable output $z(k)$.

A different mapping is the network's objective when it is used to predict the next values of a chaotic dynamic time series. It is well known that chaotic signals are usually generated by the state evolution of chaotic nonlinear systems. In a discrete time the state evolution of a chaotic signal is generally described by the following equation

$$x(k+1) = f(k, x(k), \theta(k)) \quad (3)$$

In chaotic systems in discrete space, the signal's states may be attracted to and remain on a compact subset of the state space, known as attractor, for a given set of initial values $x(0)$, and parameters θ . In chaotic signal analysis the network's objective is to identify the mapping $f()$ which maps the current state of the chaotic model to the next one.

Various filters and algorithms have been developed in the past extending the original linear theory for certain nonlinearities and for allowing adaptation in model parameters. However, these algorithms require sufficient a priori information of the system, are quite complex, and in most of the cases suboptimal as well [1],[4],[7]-[8].

Recently neural networks have been used to estimate states of dynamic systems. Recurrent neural networks seem to be an answer to these filtering problems where the applicability of other statistical estimators like the Kalman filter is limited.

In this work recurrent multilayer perceptrons trained via the backpropagation rule, have been used as neural state estimators. The standard architecture of the multilayer perceptron consists of an input layer, one or more hidden layers and an output layer. The input layer consists of simple distributing nodes, while the other two or more layers have sigmoidal non linear nodes. When the network is used to provide estimates of the systems's state a buffered delayed version of the output

signals (measurements) are fed to the input nodes, since the actual states are used as desired vectors in the output nodes. In an infinite dimension environment an alternative technique that uses residuals (pseudo-innovation process) between actual and predictive measurements as additional input to the neural network has also successfully used. In this context the neural structure can be viewed as an input recurrent dynamic network that allows information to flow from the output nodes to the input nodes capturing the transient characteristics of the dynamic physical system. During training the weights of the network are calculated iteratively using the steepest descent based backpropagation rule in order to minimize a quadratic function of the error. After training the network is ready to be used as estimator. There is a large amount of experience to indicate that the neural estimator performs exceptionally well in many practical applications.

The trained multilayer perceptron as state estimator enjoys certain advantages over the conventional filters derived through the statistical methodology [18],[20].

- It can handle any assumption or uncertainty concerning the statistics of the actual data generation model. It does not depend on any assumption about the stochastic input. To the contrary the Gaussian nature of the noises in the model, and the independence between the state and the noises, are fundamental assumptions behind any feasible filter derived using statistical methodologies.
- Due to its massive parallel structure and high speed the neural estimator can take full advantage of the new hardware capabilities. That makes the neural and not the statistical estimator the preferable choice for real time signal processing, and automatic control applications.

The main disadvantage of the neural estimation methodology described above is that the designer must have complete knowledge of the system's dynamics during the training phase [18]-[21]. The training procedure is performed in a supervised manner, whereby a desired output signal is used to compute the error [9]-[12]. If state estimation in a state space model is the objective, full knowledge of the model dynamics is required.

However, the ideal model to be used in training is rarely known. Almost all real plants can be characterized as a system with partially known dynamics since nobody can fully realize an actual system with a mathematical model [4]. In the context of adaptive filtering where models with structure or parameter uncertainty are used, the designer never knows the exact dynamics of the physical model. In a similar situation when nonlinear non stationary signals are used the desired outputs may vary in time.

Due to the above difficulties the designer is forced to use an evaluation criterion describing the overall performance of the network in an approximate correct training set. This method requires large training periods with slow training and significant performance and/or robustness degradation.

A different approach is proposed here. Instead of using one network to map approximate dynamics of an adaptive system, several networks are trained independently using different variations of the actual adaptive model. Each one of the networks trained with data pairs obtained using different parameter vectors $\theta(k)$ in the nonlinear system that generates the data. Due to this training methodology each network converges to a different solution. When the training is over, a bank of different neural estimators is available to be applied to the solution of the nonlinear filtering problems. In the mean time another set of multilayer perceptrons are used to provide one step ahead predictions of the systems measurements. In the actual operation phase a nonlinear selection mechanism is used. The adaptive algorithms compares residuals between true and predicted measurements to identify at every time instant which neural estimator provides the best estimate.

The adaptive neural methodology described above uses the Lainiotis partitioning theorem [3]-[5], and it is an extension of the Adaptive Lainiotis Filter (ALF) to neural networks. In this case however, instead of a bank of statistical filters operating around trajectory points, trained multilayer perceptrons are used [21]. The adaptive neural estimation scheme integrates the robustness of the neural estimator with the effectiveness and attractiveness of the partitioning theory. In the next section simulation studies were carried out to test the effectiveness of the new algorithm.

III. CHAOTIC TIME SERIES PREDICTION

The classic time series prediction problem is the one step ahead prediction of the logistic function (Feigenbaum map) [13] in its discrete form:

$$x(k+1) = ax(k)(1-x(k)) \quad (4)$$

The behavior of the time series generated by the above equation depends critically upon the value of the parameter a . If $a < 1$ the map has a single fixed point at the origin, and from a random initial value in the closed interval $[0, 1]$ the time series collapses to a constant value. For $a > 3$ the map generates periodic attractor. Beyond the value $a = 3.6$ the map becomes chaotic. The time series passes every test for randomness, having the spectrum of a white noise process. It is well known in the neural network literature that the future value of the time series can be predicted perfectly once the actual data generation model is learned by a neural network. In the first simu-

lation experiment the objective is the prediction of the map's future values when the critical parameter a is unknown. The experimental set-up is summarized below:

In order to estimate the state of the above model the following estimators had been used in this first experiment:

SIMULATION III. 1

A. adaptive neural estimator:

- a bank of three recurrent multilayer perceptrons, each one trained with data obtained using a different value of the parameter a in equation (4)
- the candidate networks are trained as follows:
- the network one uses data obtained with $a = 3.6$
- the network two uses data obtained with $a = 3.8$
- the network three uses data obtained with $a = 4.0$

For each one of the three perceptrons in the adaptive estimator bank the following set-up is used.

B. recurrent multilayer perceptron

network topology:

- two input nodes: the current and the previous value of the time series are used as input signals
- one output node: the one step ahead value of the time series
- two hidden layers with 4-1 hidden nodes respectively

learning parameters:

- learning rate: 0.05, momentum: 0.1

Training procedure:

- backpropagation training algorithm
- the target vector is the actual one step ahead value of the chaotic signal.
- the network tries to minimize the square error between the current output and the target vector
- training data sets of 100 points are produced by running the system equation (4) using every time a random selected initial condition from the closed interval $[0, 1]$.
- the training procedure is terminated if the training error tolerance is less than 0.01 or if the number of iterations of the training set is more than 2500
- the test data record consists of a sequence of data points produced separately from the training record using a new initial condition. The test data set is generated using the value $a = 4$.

Since there is no theoretical analysis to justify the performance of the neural estimators Monte Carlo techniques are used to verify the results. The figure of merit used to compare performance is the mean square error averaged over 100 Monte Carlo runs. Namely, the following performance index,

is used

$$MSE = \frac{1}{mc} \cdot \sum_{i=1}^{mc} (x(k) - \hat{x}(k/k))^2 \quad (5)$$

The performance of the adaptive estimator is given in Fig. 1-4. For comparison purposes the performance of a neural estimator which had been trained using the parameter value $a=3.6$ is also given. This neural estimator is called 'mismatched'

SIMULATION III.2

The most difficult part in the design of the adaptive estimator is to select values for the parameter a . In the first experiment the actual value of the parameter used to generate the test data was included in the design. In this second experiment the adaptive estimator uses three networks each of them trained using the values $a=3.6, a=3.9, a=4.5$ respectively. It can be noticed that the real value $a=4$ is not included in the set. Moreover the performance of the adaptive estimator is compared with that of a simple "mismatched" neural estimator which was trained using random values from the interval (3.5,4.5). The networks' configurations are the same as above. In Fig. 5-8, it can be seen that the adaptive neural estimator correctly recognizes the candidate model which is closer to the correct one (Fig. 8). The 'mismatched' estimator fails completely (Fig. 6).

In order to verify the results obtained previously similar experiments are performed for the two dimension Henon map [21]. Its equation is given by:

$$x(k+1) = 1 - 1.4x^2(k) + 0.3x(k) \quad (6)$$

The Henon map form is qualitatively similar to that of the Feigenbaum (logistic) map.

SIMULATION III.3

In the third simulation the unknown parameter is the value of the constant term in equation (6). The set -up for the adaptive estimator is now

C. adaptive neural estimator:

- a bank of three recurrent multilayer perceptrons, each one trained with data obtained using a different value of the constant term in equation (6)
- each one of the candidate networks is trained using a different value of the parameter a
- the network one uses data obtained with $a=1.4$
- the network two uses data obtained with $a=0.8$
- the network three uses data obtained with $a=1.0$

The same configuration as before is used for the multilayer perceptrons. The performance of the adaptive estimator is compared with that of a 'mismatched' estimator trained using the parameter value $a=0.6$. The results are depicted in Fig. 9-12.

SIMULATION III.4

In last experiment the robustness of the estimators with respect to the parameter values used in the design is investigated. The adaptive estimator is designed without any network trained with the exact value $a=1$. The following values are used in the training phase, $a=0.6, a=0.9, a=1.2$. The 'mismatched' estimator is a simple recurrent network trained using random values from the interval (0,1) as constant term in equation (6). According to the results depicted in Fig. 13-16 the 'mismatched' estimator cannot tolerate the ignorance about the constant term in the model and fails completely. To the contrary the adaptive estimator correctly identifies that the second net which was trained with value 0.9 is the closest to the actual model. The Adaptive estimator uses it to provide the estimates.

Observations:

- The adaptive neural estimator learned to emulate the logistic map and was able to accurately predict the time series on a novel testing set, obtained independently of the training set using different initial conditions
- The adaptive algorithm successfully detected the actual data generation model, and selected the appropriate trained neural estimator from its bank to provide the necessary estimation. In the more realistic design of experiments III.3, and III.4 the network identifies the model that is closer to the actual model.
- The decision about the parameter value during the training phase is crucial for the performance of the estimator. All the 'mismatched' estimators failed to provide accurate results.

IV. ADAPTIVE STATE ESTIMATION

The neural networks' ability to represent nonlinear dynamic systems has served as initiative for their application to nonlinear filtering problems. In this section we investigate the performance of the adaptive neural estimator in nonlinear state estimation problems. Moreover since the major advantage of the proposed new algorithm is its ability to adapt in changing environments in all the simulation studies it will be assumed that more than one dynamic models generate the data.

SIMULATION IV.1:

The objective of this experiment is to illustrate the effectiveness of the recurrent neural network as estimator and assess the attractiveness of the adaptive neural estimator for changing actual models. Two different state space models are used in this numerical example. The first system is a signal observed in noise as follows:

$$x(k+1) = 1.7 \exp(-2x^2(k)) + 0.1w(k) \quad (7)$$

$$z(k) = x^3(k) + 0.1v(k) \quad (8)$$

where $w(k)$, $v(k)$ are independent standard white Gaussian sequences, and the initial state $x(0)$ is assumed Gaussian with zero mean and variance 0.25.

The second system is a linear state space model with state dependent noises. The equations of the this system are:

$$x(k+1) = 0.5x(k) + 0.5 \tanh(x(k) + 0.5w(k)) \quad (9)$$

$$z(k) = x(k) + 0.5x^3(k)v(k) \quad (10)$$

with $w(k)$, $v(k)$ are independent standard white Gaussian sequences, and the initial state $x(0)$ is assumed Gaussian with zero mean and variance 0.25.

It must be emphasized that all the statistical filters have restricted applicability in chaotic nonlinear systems like the one in eq. (7),(8). Moreover there is no feasible statistical filter that can be used to estimate the state of the system in eq. (9),(10).

An adaptive neural estimator is applied to this problem. The adaptive estimator has in its bank two multilayer perceptrons. The Network I is trained using data which are generated running the equations (7),(8). The Network II is trained using the data which have been produced from the system of equations (9),(10). After training the recurrent nets are used as neural filters. To test the efficiency of the proposed algorithm in a time varying environment we assume that the data generation model is not the same during the actual operation phase. Specifically, in the first half of the testing period the measurements are coming from the first model. In the remaining time the second model is used to provide the actual measurements. The objective of the adaptive network is to correct identify the active model, and then use its decision to provide the best estimate of the current state. The detailed configuration of the recurrent nets used by the adaptive estimator is summarized below:

A. input recurrent neural network

network topology:

- two input nodes: the current and the previous measurements are used as input signals
- one output node: the estimates of the system states. The neural network has so many output nodes, as the states of the model
- two hidden layers with 4-4 hidden nodes respectively

learning parameters:

- learning rate: 0.05, momentum: 0.2

Training procedure:

- backpropagation training algorithm
- the target vector during training is a state vector which is generated running the equations of one model
- the network tries to minimize the square error between the current output and the target vector
- each training set consists of 200 input/output pairs ($z(k)$, $x(k)$).
- the training procedure is terminated if the training error tolerance is less than 0.01 or if the number of iterations of the training set is more than 5000
- the test data record consists of a sequence of data points produced separately from the training record. In order to generate the test data both the models are used.

The performance of the adaptive estimator and the mean square averaged over 100 Monte Carlo runs are given in Fig. 17-19. The adaptive algorithm correct identifies the change in the data generation process during the operation phase and uses the appropriate trained neural network in its bank to provide the estimate.

SIMULATION IV.2

In this last simulation test a linear, and a highly nonlinear state space models are used to test the performance of the adaptive algorithm. The first one is described by the following equations:

$$x(k+1) = 0.588x(k) + w(k) \quad (11)$$

$$z(k) = x(k) + v(k) \quad (12)$$

where $w(k)$, $v(k)$ are independent white Gaussian sequences with variance 0.25, and the initial state $x(0)$ is assumed standard white Gaussian.

The second one is the model described by the equations (9)-(10). During the actual operation phase in the first half of the testing period equations (11)-(12) are used to generate the data. From the data point 100 until the end the alternative model is used. The adaptive estimator has two networks in the bank. Network I is trained with data from model (11)-(12),

and network II is trained according to the model (9)-(10). The same configuration as in simulation IV.1 is used during training. The graphs in Fig. 20-22 summarize the results of this last experiment. It is obvious that the adaptive neural estimator can operate in a changing environment. In both experiments the algorithm had successfully identify the model change in a limited number of steps.

V. CONCLUSIONS

The problem of adaptive estimation via neural networks was addressed in this paper. New adaptive neural estimators were proposed and applied in numerous problems. The results can be summarized as follows:

- The neural estimator provides reliable and consistent solutions to the estimation problem, especially in nonlinear or chaotic models.
- In realistic situations where the actual model is not completely known or changes during the implementation phase an adaptive neural estimator must be used. The adaptive neural estimator can efficiently detect the active model and use the recurrent perceptrons in its bank to provide the best estimate.

In conclusion, the neural estimator with its capability to provide accurate solutions to adaptive estimation problem under more realistic assumptions, and its massively parallel structure and high speed, constitutes a new promising tool in estimation theory.

REFERENCES

- [1] B.D.O Anderson, J.B. Moore **Optimal Filtering**, Prentice Hall, 1979.
- [2] D.G. Lainiotis, "*Optimal Nonlinear Estimation*", International Journal of Control, Vol. IJC-14, pp. 11137-1148, 1971.
- [3] D.G. Lainiotis, "*Optimal Adaptive Estimation: Structure and Parameter Adaptation*", IEEE Transactions on Automatic Control, Vol. AC-16, pp. 160-170, 1971.
- [4] D.G. Lainiotis, "Partitioning: A Unifying Framework for Adaptive Systems, I-Estimation", Proceedings of IEEE, Vol. 64, pp. 1126-1142, 1976.
- [5] D.G. Lainiotis, "Joint detection, estimation and system identification" Inform. Control J., vol. 19, pp. 75-92, 1972.
- [6] D.G. Lainiotis, J. G. Deshpande "Parameter estimation using splines" J. Information Sciences, vol. 7, 1974.
- [7] D.G. Lainiotis, S.K. Katsikas, "*Linear and Nonlinear Lainiotis filters Survey*", in IFAC Workshop, CAMS-'89, Expert Systems in Marine Automation Proceedings, pp. 280-293, 1989.
- [8] K. Watanabe **Adaptive Estimation and Control: The Partitioning Approach**, Prentice Hall, 1992.
- [9] D.E. Rumelhart, J.L. McClelland, **Parallel Distributed Processing: Explorations into the Microstructure of Cognition, Vol. 1**, M.I.T. Press, 1986.
- [10] B. Kosko **Neural Networks for Signal Processing**, Prentice Hall, 1992.
- [11] L.V. Fausett **Fundamentals of Neural Networks: Architectures, Algorithms, and Applications**, Prentice Hall, 1993.
- [12] D.R. Hush, B.G. Horne "*Progress in Supervised Neural Networks*", IEEE Signal Processing Magazine, pp 8-39, 1993.
- [13] D. Lowe, A.R. Webb "*Time Series Prediction by Adaptive Networks: A Dynamic System Perspective*", IEEE Proceedings Part F, Vol. 138, pp 17-24, 1991.
- [14] Y.H. Pao, G.H. Clark, D.J. Sobajic "*System Identification and Noise Cancellation: A Quantitative Comparative Study of Kalman Filtering and Neural Network Approaches*", Proceedings of ACC-91, Vol. I, pp. 1408-1411, 1991.
- [15] Q. Sun, A.T. Aluani, T.R. Rice, J.E. Gray "*A Neural Network Computation Algorithm for Discrete Time Linear System State Estimation*", Proceedings of IJCNN-92, Vol. I, pp. 443-448, 1992.
- [16] J.P. De Gruyenaecce, H.M. Haffer, "*A Comparison between Kalman Filters and Recurrent Neural Networks*", Proceedings of IJCNN-92, Vol. IV, pp. 247-251, 1992.
- [17] A.J. Kanekar, A. Feliachi, "*State Estimation using Artificial Neural Networks*", Proceedings of IEEE Systems and Engineering Conference, pp. 552-556, 1990.
- [18] D.G. Lainiotis, K.N. Plataniotis, D. Menon, C.J. Charalampous, "*Heave Compensation via Neural Networks*", Intelligent Engineering Systems through Artificial Neural Networks, Vol. 3, pp 143-148, ASME Press, 1993.
- [19] D.G. Lainiotis, K.N. Plataniotis, D. Menon, C.J. Charalampous, "*Adaptive Heave Compensation via Neural networks*", IEEE OCEANS'93 Proceedings, Vol. I, pp 243-248, 1993.
- [20] D.G. Lainiotis, K.N. Plataniotis, D. Menon, C.J. Charalampous, "*Neural Network Application to Ship Position estimation*", IEEE OCEANS'93 Proceedings, Vol. I, pp 243-248, 1993.
- [21] K.N. Plataniotis, **Adaptive State Estimation via Neural Networks**, Ms Thesis, Florida Institute of Technology, 1992.

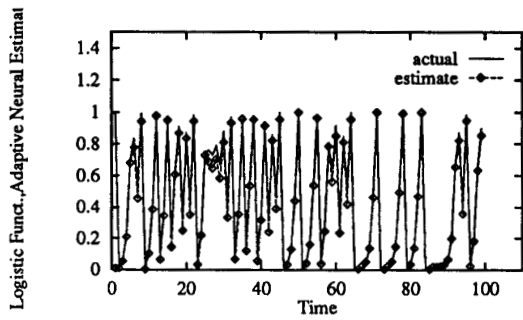


Fig. 1. Simulation III.1, Logistic Map, Adaptive Neural Estimator, One Step Ahead prediction

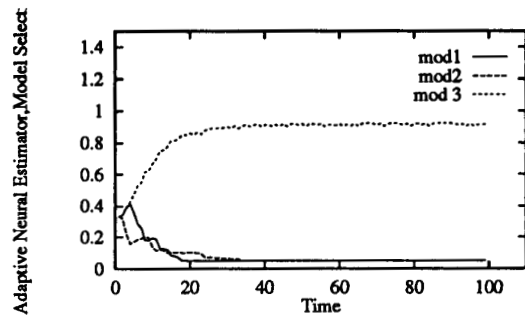


Fig. 4. Simulation III.1, Logistic Map, Adaptive Estimator, Model Selection

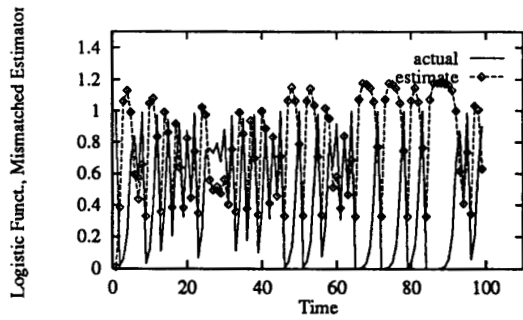


Fig. 2. Simulation III.1, Logistic Map, Mismatched Estimator, One Step Ahead Prediction

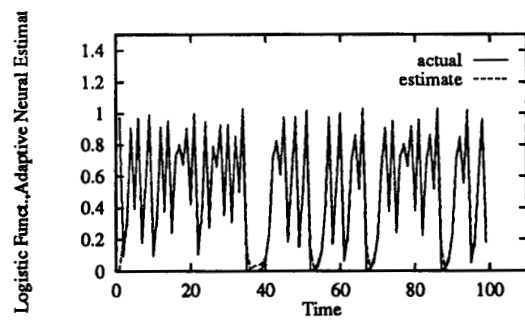


Fig. 5. Simulation III.2, Logistic Map, Adaptive Neural Estimator, One Step Ahead prediction

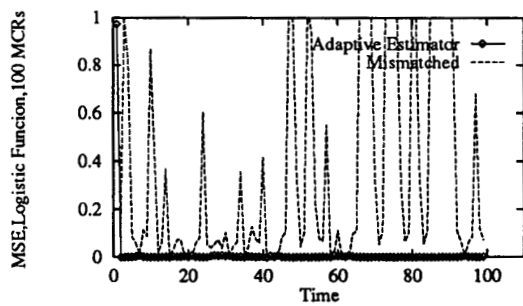


Fig. 3. Simulation III.1, Logistic Map, MSE, Comparative Evaluation, 100 MCRs

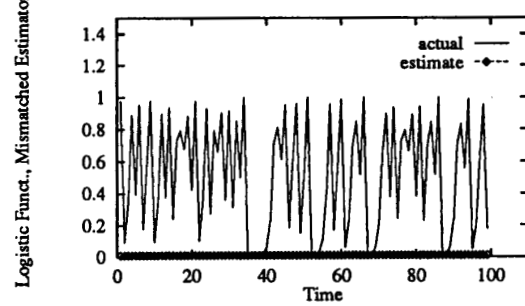


Fig. 6. Simulation III.2, Logistic Map, Mismatched Estimator, One Step Ahead Prediction

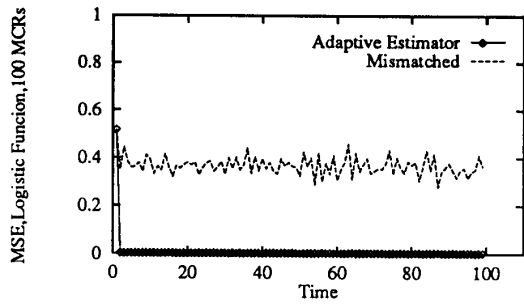


Fig. 7. Simulation III.2, Logistic Map, MSE, Comparative Evaluation, 100 MCRs

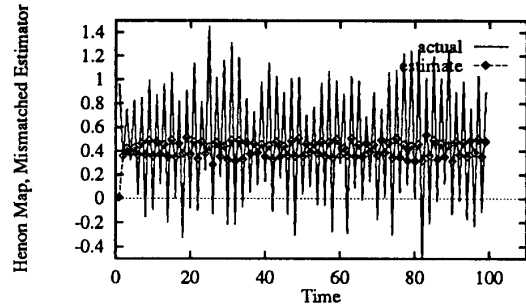


Fig. 10. Simulation III.3, Henon Map, Mismatched Estimator, One Step Ahead Prediction

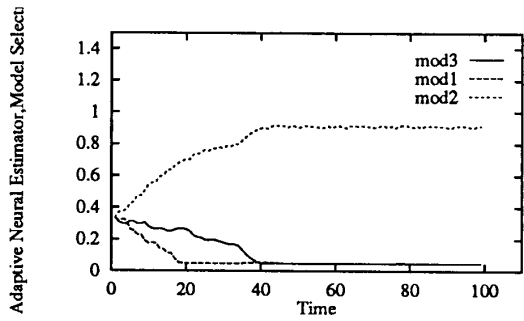


Fig. 8. Simulation III.2, Logistic Map, Adaptive Estimator, Model Selection

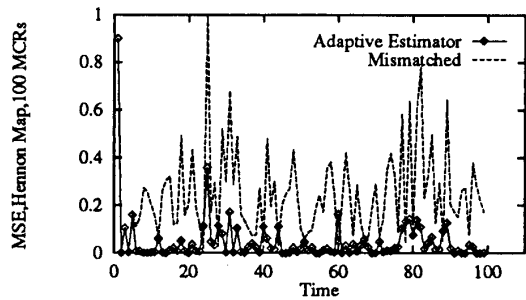


Fig. 11. Simulation III.3, Henon Map, MSE, Comparative Evaluation, 100 MCRs

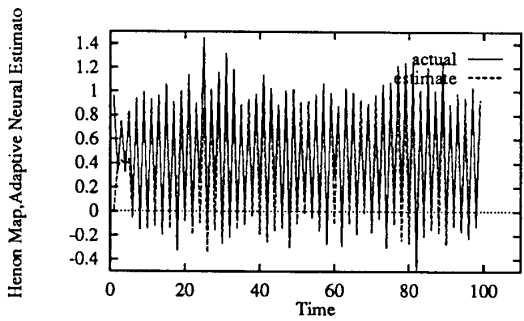


Fig. 9. Simulation III.3, Henon Map, Adaptive Neural Estimator, One Step Ahead prediction

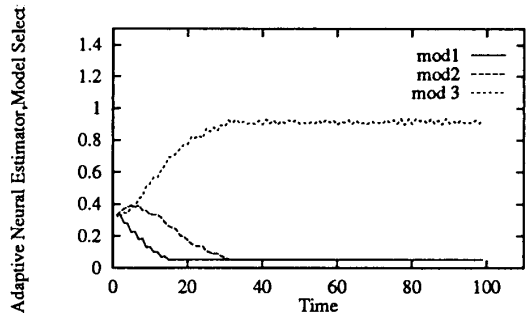


Fig. 12. Simulation III.3, Henon Map, Adaptive Estimator, Model Selection

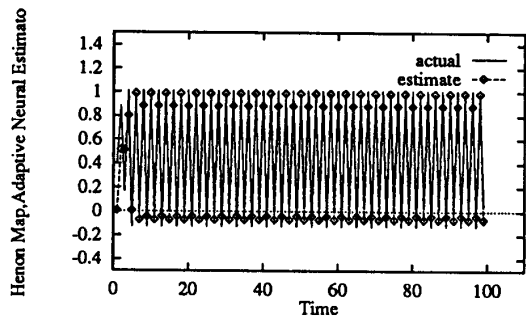


Fig. 13. Simulation III.4, Henon Map, Adaptive Neural Estimator, One Step Ahead prediction

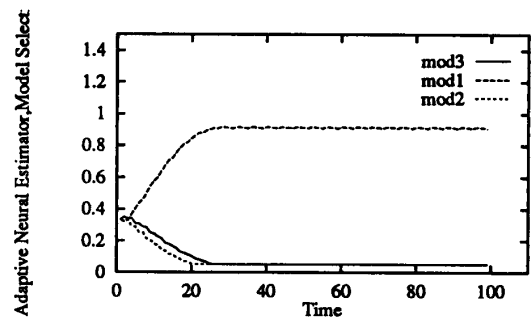


Fig. 16. Simulation III.4, Henon Map, Adaptive Estimator, Model Selection

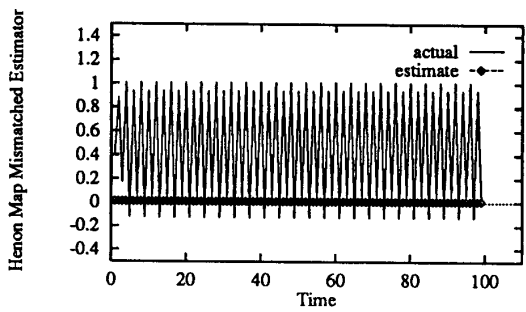


Fig. 14. Simulation III.4, Henon Map, Mismatched Estimator, One Step Ahead Prediction

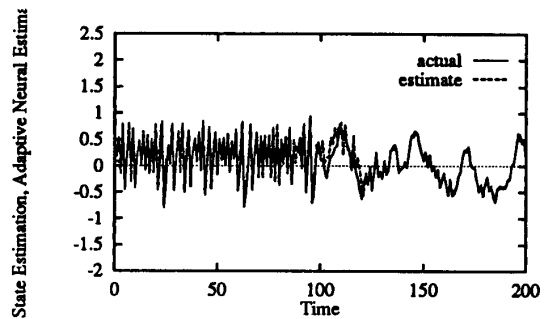


Fig. 17. Simulation IV.1, State Space Model, Adaptive Estimator

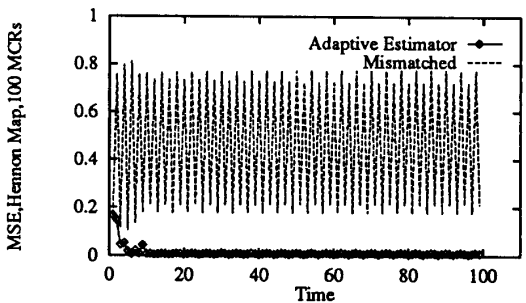


Fig. 15. Simulation III.4, Henon Map, MSE, Comparative Evaluation, 100 MCRs

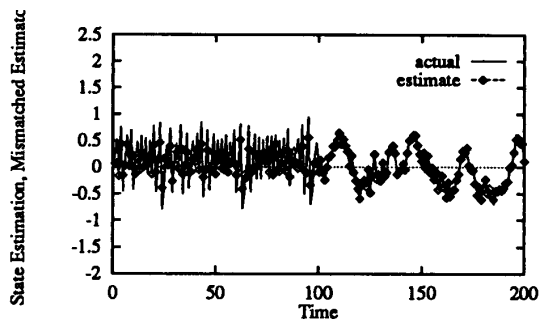


Fig. 18. Simulation IV.1, State Space Model, Mismatched Estimator

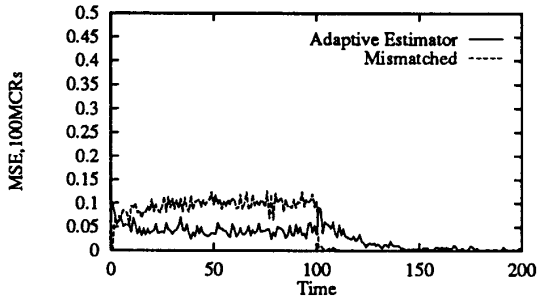


Fig. 19. Simulation IV.1, State Space Model, MSE, Comparative Evaluation, 100 MCRs

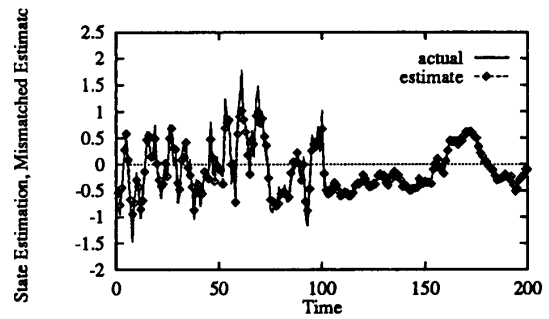


Fig. 22. Simulation IV.2, State Space Model, Mismatched Estimator

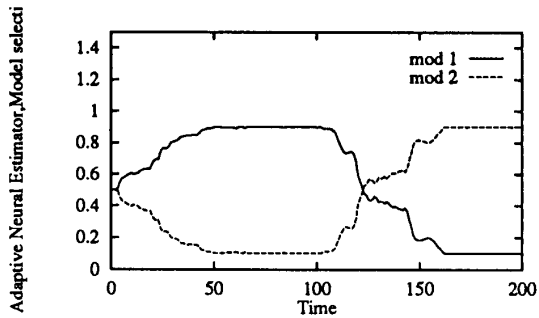


Fig. 20. Simulation IV.1, State Space Model, Adaptive Estimator, Model Selection

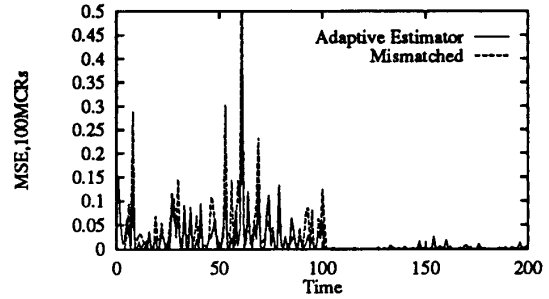


Fig. 23. Simulation IV.2, State Space Model, MSE, Comparative Evaluation, 100 MCRs

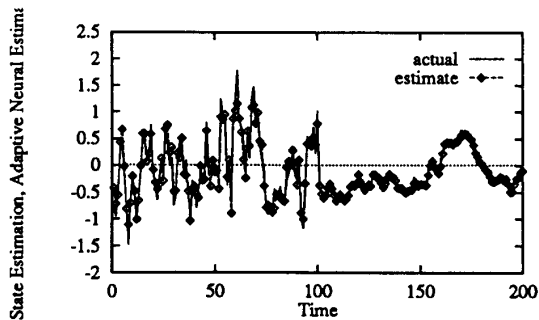


Fig. 21. Simulation IV.2, State Space Model, Adaptive Estimator

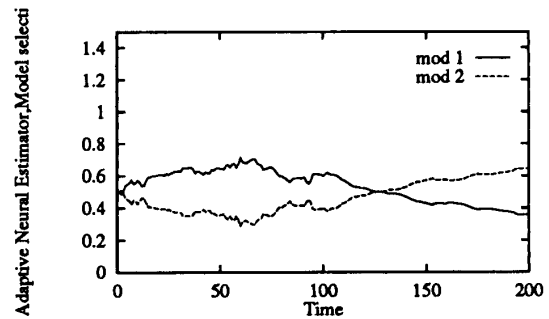


Fig. 24. Simulation IV.2, State Space Model, Adaptive Estimator, Model Selection