[redacted]

# A RECURSIVE HYBRID ALGORITHM FOR NON-LINEAR SYSTEM IDENTIFICATION USING RADIAL BASIS FUNCTION NETWORKS

S. Chen†, S.A. Billings‡ and P.M. Grant†

†Department of Electrical Engineering
University of Edinburgh
Mayfield Road
Edinburgh EH9 3JL
Scotland

‡Department of Control Engineering
University of Sheffield
Mappin street
Sheffield S1 3JD
England

January 1991

Research Report No. 422

## Abstract

Recursive identification of non-linear systems is investigated using radial basis function networks. A novel approach is adopted which employes a hybrid clustering and least squares algorithm. The recursive clustering algorithm adjusts the centres of the radial basis function network while the recursive least squares algorithm estimates the connection weights of the network. This hybrid algorithm significantly enhances the real-time or adaptive capability of radial basis function models. The application to simulated and real data are included to demonstrated the effectiveness of this hybrid approach.

## 1. Introduction

Modelling non-linear systems using radial basis function (RBF) networks has certain attractive advantages. The general approximation capabilities of the RBF network provides the theoretical foundation for representing complex processes. Furthermore, the response of the RBF network is linear with respect to the connection weights of the network. Provided that the other parameters, the RBF centres, can be chosen appropriately, the linear least squares method can therefore be employed to estimate these weights.

For off-line system identification, blocks of data are usually available, and an orthogonal least squares algorithm [1,2] can be used to fit RBF models. This algorithm not only provides the least squares estimate for the RBF weights but also selects appropriate centres automatically from the data set. Moreover, the information regarding how many centres are required to fit the data adequately is revealed during the identification procedure.

For on-line or adaptive applications of RBF models, however, some kind of recursive identification algorithm is required. A simple solution is to fix the RBF centres first and to update only the RBF weights in real-time using the recursive least squares or least mean squares algorithm. This can only work well if changes in the underlying system are small. It is advantageous to update RBF centres and weights simultaneously because this will significantly improve both the modelling capability and the tracking property. Moody and Darken [3] suggested using an $n$-means clustering technique to adjust centres in real-time and derived a hybrid clustering and least mean squares algorithm. In the present study we adapt this idea to non-linear system identification using RBF models. In order to improve the convergence properties further, we propose a hybrid clustering and Givens least squares algorithm. The Givens least squares algorithm [4] has superior numerical properties and has a further advantage that it can be implemented using systolic arrays [5].

The paper is organized as follows. Section 2 introduces a brief summary of RBF networks applied to model single-input single-output non-linear systems. The derivation of the hybrid clustering and Givens least squares algorithm is given in Section 3, and application to a simulated time series process, a liquid level system and a heat exchanger are included in Section 4. The extension to multi-input multi-output systems is straightforward and this is discussed in Section 5. Finally some concluding remarks are given in Section 6.

## 2. Modelling non-linear systems using RBF networks

Many single-input single-output non-linear systems can be described in terms of some non-linear functional expansion of lagged inputs and outputs as follows:

$$y(t)=f_s(y(t-1),\ldots,y(t-n_y),u(t-1),\ldots,u(t-n_u))+e(t) \qquad (1)$$

where $y(t)$, $u(t)$ and $e(t)$ are the system output, input and white noise respectively; $n_y$ and $n_u$ are the lags of the output and input respectively; and $f_s(.)$ is some non-linear function. Most of the discussion in the current study is based on this system representation.

The assumption for the system representation (1) is that the noise source is white and enters the system additively. In general, however, the noise source may be correlated and can enter the system in a more complicated manner. These possibilities can be accomodated in the following more general system representation:

$$y(t)=f_s(y(t-1),\ldots,y(t-n_y),u(t-1),\ldots,u(t-n_u),e(t-1),\ldots,e(t-n_e))+e(t) \qquad (2)$$

where $n_e$ is the lag of the noise, and $e(t)$ is white. The system (2) is known as the NARMAX model [6,7]. The identification procedure developed in the present study can be extended to this general system.

The RBF network depicted in Fig.1 is a two-layer processing structure. The first layer consists of an array of computing units. Each unit contains a parameter vector called a centre, and this calculates the Euclidean distance between the centre and the network input vector. The unit then passes the result through a non-linear function. The second layer is essentially a linear combiner. The overall response of such a network is a mapping $f_r:\mathbf{R}^m \to \mathbf{R}$, that is

$$f_r(\mathbf{v})=\sum_{i=1}^{n}\theta_i\phi(||\mathbf{v}-\mathbf{c}_i||) \qquad (3)$$

where $\mathbf{v}\in\mathbf{R}^m$ is the network input vector; $\phi(.)$ is a function from $\mathbf{R}^+$ to $\mathbf{R}$; $||.||$ denotes the Euclidean norm; $\mathbf{c}_i \in \mathbf{R}^m$, $1 \le i \le n$, are the RBF centres; $\theta_i$, $1 \le i \le n$, are the connection weights; and $n$ is the number of computing units in the first layer.

The aim in the present study is to use the RBF network response $f_r(.)$ to capture or to approximate the underlying dynamics $f_s(.)$ in (1). Define $m = n_y + n_u$ and let

$$\mathbf{v}(t) = [y(t-1) \cdots y(t-n_y) u(t-1) \cdots u(t-n_u)]^T . \tag{4}$$

The idea then becomes one that uses

$$\hat{y}(t) = f_r(\mathbf{v}(t)) \tag{5}$$

as the one-step-ahead predictor for $y(t)$. In the present study, $\phi(.)$ is chosen as the so called thin-plate-spline function:

$$\phi(v) = v^2 \log(v) . \tag{6}$$

This choice of $\phi(.)$ provides good modelling capability and is discussed, for example, by Powell [8]. Other choices of $\phi(.)$ can also be employed.

Whether a non-linear model is adequate can be tested using the following two model validity approaches. Define the one-step-ahead prediction error or residual

$$\epsilon(t) = y(t) - \hat{y}(t) . \tag{7}$$

The first model validation method computes the following correlation functions [9]:

$$\left.\begin{array}{rl} \Psi_{\epsilon\epsilon}(k) & k \neq 0 \\ \Psi_{u\epsilon}(k) & \text{for all } k \\ \Psi_{\epsilon(\epsilon u)}(k) & k \ge 0 \\ \Psi_{u^{2'}\epsilon}(k) & \text{for all } k \\ \Psi_{u^{2'}\epsilon^2}(k) & \text{for all } k \end{array}\right\} \tag{8}$$

where $\epsilon u(t) = \epsilon(t+1) u(t+1)$, $u^{2'}(t) = u^2(t) - \overline{u^2(t)}$ and $\overline{u^2(t)}$ is the time average or mean value of $u^2(t)$. In general, if the correlation functions (8) are within the 95% confidence bands, $\pm 1.96/N^{1/2}$, the model is regarded as adequate, where $N$ is the number of data samples. The alterr.ative approach is the chi-squared statistical test [10,11]. Define an $\eta$-dimensional vector valued function

$$\Omega(t) = [\omega(t) \omega(t-1) \cdots \omega(t-\eta+1)]^T \tag{9}$$

where $\omega(t)$ is some function of the past inputs, outputs and prediction errors. The chi-squared statistic is defined as

$$\zeta = N \mu^T (\Gamma^T \Gamma)^{-1} \mu \tag{10}$$

with

$$\mu = N^{-1}\sum_{t=1}^{N}\Omega(t)\epsilon(t)/\sigma_\epsilon \tag{11}$$

and

$$\Gamma^T\Gamma = N^{-1}\sum_{t=1}^{N}\Omega(t)\Omega^T(t) \tag{12}$$

where $\sigma_\epsilon^2$ is the variance of $\epsilon(t)$. If the values of $\zeta$ for several different choices of $\omega(t)$ are within 95% acceptance region, the model is regarded as adequate.

## 3. Hybrid clustering and Givens least squares algorithm

For on-line identification applications using the RBF network, some recursive rules are essential to update the centres and weights. The centres should suitably sample the network input domain and should be able to track the changing patterns of data. Moody and Darken [3] suggested the $n$-means clustering procedure as a good updating rule for the RBF centres. The $n$-means clustering technique is well documented in many pattern classification text books (e.g. [12]). Because the response of the network is linear with respect to its weights, it is natural to consider the recursive least squares method for adjusting the weights. These observations suggest that the recursive identification algorithm for RBF models should have a hybrid structure consisting of:

○ Recursive $n$-means clustering sub-algorithm for adjusting the RBF centres.

○ Recursive least squares sub-algorithm for updating the RBF weights.

Details of these two sub-algorithms are now given.

Given initial centres $\mathbf{c}_i(0)$, $1 \le i \le n$, and an initial learning rate for the centres $\alpha_c(0)$, at each sample $t$ the recursive $n$-means clustering algorithm consists of the following computational steps:

□ Compute distances and find a minimum distance

$$a_i(t) = ||\mathbf{v}(t) - \mathbf{c}_i(t-1)||, \quad 1 \le i \le n,$$
$$k = arg\,[min\,\{a_i(t), \quad 1 \le i \le n\}].$$

□ Update centres and re-compute $k$th distance

$$\mathbf{c}_i(t) = \mathbf{c}_i(t-1), \quad 1 \le i \le n \text{ and } i \ne k,$$
$$\mathbf{c}_k(t) = \mathbf{c}_k(t-1) + \alpha_c(t)(\mathbf{v}(t) - \mathbf{c}_k(t-1)),$$
$$a_k(t) = ||\mathbf{v}(t) - \mathbf{c}_k(t)||.$$

The initial centres are often chosen randomly. The learning rate should be $\alpha_c(t) < 1$, and should slowly decrease to zero. In the present application $\alpha_c(t)$ is computed according to

$$\alpha_c(t)=\alpha_c(t-1)/(1+int[t/n])^{1/2} \tag{13}$$

where *int*[.] denotes the integer part of the argument. Other computing rules can also be applied to $\alpha_c(t)$.

The convergence properties of the $n$-means clustering procedure were studied by MacQueen [13]. The $n$-means clustering is based on a linear learning rule, thus guaranteeing rapid convergence. It is also an unsupervised procedure using only the network input data. No desired response is required and the procedure will not be affected by the learning rule used for the weights. Notice the similarities between the $n$-means clustering and Kohonen self-organising algorithm [14].

The recursive least squares algorithm is based on a recursive solution of the normal equation. Define the hidden layer output vector at $t$ as

$$\Phi(t)=[\phi_1(t)\cdots\phi_n(t)]^T=[\phi(a_1(t))\cdots\phi(a_n(t))]^T \tag{14}$$

and the connection weight vector at $t$ as

$$\Theta(t)=[\theta_1(t)\cdots\theta_n(t)]^T. \tag{15}$$

The weighted normal equation can then be written as

$$(\mathbf{X}_t^T\mathbf{W}_t\mathbf{X}_t)\Theta(t)=\mathbf{X}_t^T\mathbf{W}_t\mathbf{y}_t \tag{16}$$

where

$$\mathbf{X}_t=\begin{bmatrix}\Phi^T(1)\\ \cdot\\ \cdot\\ \cdot\\ \Phi^T(t)\end{bmatrix}, \tag{17}$$

$$\mathbf{y}_t=[y(1)\cdots y(t)]^T, \tag{18}$$

and $\mathbf{W}_t$ is a $t\times t$ diagonal matrix defined recursively by

$$\mathbf{W}_t=\begin{bmatrix}\lambda(t)\mathbf{W}_{t-1} & 0\\ 0 & 1\end{bmatrix},\quad \mathbf{W}_1=1. \tag{19}$$

$\lambda(t)$ is the usual forgetting factor at $t$. The recursive least squares algorithm solves (16) to give

$$\Theta(t)=(\mathbf{X}_t^T\mathbf{W}_t\mathbf{X}_t)^{-1}\mathbf{X}_t^T\mathbf{W}_t\mathbf{y}_t. \tag{20}$$

It is well-known that if the number of parameters $n$ is large, the least squares problem may become ill-conditioned and the use of Givens transformations [4] to solve the recursive least squares problem has numerical advantages over the algorithms based directly on the normal equation. $\mathbf{W}_t^{1/2}\mathbf{X}_t$ can be decomposed into

$$\mathbf{W}_t^{1/2}\mathbf{X}_t = \mathbf{Q}(t)\mathbf{S}(t) \tag{21}$$

where

$$\mathbf{S}(t) = \begin{bmatrix} 1 & s_{12}(t) & s_{13}(t) & \cdot & \cdot & \cdot & s_{1n}(t) \\ 0 & 1 & s_{23}(t) & \cdot & \cdot & \cdot & s_{2n}(t) \\ 0 & 0 & \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & \cdot & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot & \cdot & \cdot \\ \cdot & & & & \cdot & 1 & s_{n-1n}(t) \\ 0 & \cdot & \cdot & \cdot & 0 & 0 & 1 \end{bmatrix} \tag{22}$$

and $\mathbf{Q}(t)$ is a $t \times n$ matrix with orthogonal columns that satisfy

$$\mathbf{Q}^T(t)\mathbf{Q}(t) = \mathbf{D}(t) = diag\{d_1(t), \cdots, d_n(t)\}. \tag{23}$$

$\Theta(t)$ can be obtained by solving the triangular system

$$\mathbf{S}(t)\Theta(t) = \mathbf{z}(t) \tag{24}$$

where $\mathbf{z}(t)$ is an $n$-dimensional vector given by

$$\mathbf{z}(t) = \mathbf{D}^{-1}(t)\mathbf{Q}^T(t)\mathbf{W}_t^{1/2}\mathbf{y}_t. \tag{25}$$

Givens least squares algorithm can be employed to derive (24) and thus to solve for $\Theta(t)$. The algorithm is initialized by setting

$$\left.\begin{aligned} &\mathbf{z}(0) = 0 \\ &\sigma_\epsilon(0) = 0 \\ &\mathbf{S}(0) = \mathbf{I} \\ &\mathbf{D}(0) = \mathbf{I}/\rho \end{aligned}\right\} \tag{26}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix and $\rho$ is a large positive scalar. The forgetting factor $\lambda(t)$ is usually computed according to the rule [15]

$$\lambda(t) = \lambda_0\lambda(t-1) + 1 - \lambda_0. \tag{27}$$

$\lambda_0$ and $\lambda(0)$ are chosen to be less than but close to 1. At each sample $t$, the computational procedure is as follows:

□ Perform Givens transformations

$$\begin{bmatrix} \mathbf{D}^{1/2}(t-1) & [\ \mathbf{S}(t-1) & \mathbf{z}(t-1)\ ] \\ & [\ 0\cdots0 & \sigma_\epsilon(t-1)\ ] \\ \delta^{1/2} & [\ \Phi^T(t) & y(t)\ ] \end{bmatrix} \to \begin{bmatrix} \mathbf{D}^{1/2}(t) & [\ \mathbf{S}(t) & \mathbf{z}(t)\ ] \\ & [\ 0\cdots0 & \sigma_\epsilon(t)\ ] \\ & [\ 0\cdots0 & 0\ ] \end{bmatrix} \tag{28}$$

where $\delta$ is initialized to $1/\lambda(t)$.

□ Solve the triangular system (24) for $\Theta(t)$.

Explicit formulations for the Givens transformations will now be given. First let $l=1+n$ and introduce an $l \times l$ diagonal matrix $\tilde{\mathbf{D}}(t)$ as

$$\tilde{\mathbf{D}}(t)=\begin{bmatrix}\mathbf{D}(t) & 0\\ 0 & \sigma_{\epsilon}^{2}(t)\end{bmatrix}=diag\{\tilde{d}_1(t),\cdots,\tilde{d}_l(t)\}. \tag{29}$$

Next define an $l \times l$ upper triangular matrix $\tilde{\mathbf{S}}(t)$ as

$$\tilde{\mathbf{S}}(t)=\begin{bmatrix}\mathbf{S}(t) & \mathbf{z}(t)\\ 0\cdots0 & 1\end{bmatrix}=\begin{bmatrix}1 & \tilde{s}_{12}(t) & \tilde{s}_{13}(t) & \cdot & \cdot & \cdot & \tilde{s}_{1l}(t)\\ 0 & 1 & \tilde{s}_{23}(t) & \cdot & \cdot & \cdot & \tilde{s}_{2l}(t)\\ 0 & 0 & & \cdot & \cdot & & \cdot\\ \cdot & & & \cdot & \cdot & \cdot & \cdot\\ \cdot & & & & \cdot & \cdot & \cdot \\ \cdot & & & & \cdot & 1 & \tilde{s}_{l-1l}(t)\\ 0 & \cdot & & \cdot & \cdot\, 0 & 0 & 1\end{bmatrix} \tag{30}$$

and denote

$$\delta^{1/2}[\Phi^{T}(t)\ y(t)]=(\delta^{(0)})^{1/2}[x_1^{(0)}(t)\cdots x_l^{(0)}(t)]. \tag{31}$$

The Givens transformations (28) can then be rewritten more concisely as

$$\begin{bmatrix}\tilde{\mathbf{D}}^{1/2}(t-1) & [\ \ \tilde{\mathbf{S}}(t-1)\ \ ]\\ (\delta^{(0)})^{1/2} & [\ x_1^{(0)}(t)\cdots x_l^{(0)}(t)\ ]\end{bmatrix}\rightarrow\begin{bmatrix}\tilde{\mathbf{D}}^{1/2}(t) & [\ \tilde{\mathbf{S}}(t)\ ]\\ & [\ 0\cdots0\ ]\end{bmatrix}. \tag{32}$$

Assume that after $i-1$ Givens transformations have been performed,

$$(\delta^{(0)})^{1/2}(x_1^{(0)}(t),\cdots,x_l^{(0)}(t)) \tag{33}$$

is transferred to

$$0,\cdots,0,(\delta^{(i-1)})^{1/2}x_i^{(i-1)}(t),\cdots,(\delta^{(i-1)})^{1/2}x_l^{(i-1)}(t). \tag{34}$$

Then the $i$th Givens transformation transfers

$$\left.\begin{matrix}0,\cdots,0, & \tilde{d}_i^{1/2}(t-1), & \tilde{d}_i^{1/2}(t-1)\tilde{s}_{ii+1}(t-1), & \cdots, & \tilde{d}_i^{1/2}(t-1)\tilde{s}_{il}(t-1)\\ 0,\cdots,0, & (\delta^{(i-1)})^{1/2}x_i^{(i-1)}(t), & (\delta^{(i-1)})^{1/2}x_{i+1}^{(i-1)}(t), & \cdots, & (\delta^{(i-1)})^{1/2}x_l^{(i-1)}(t)\end{matrix}\right\} \tag{35}$$

into

$$\left.\begin{matrix}0,\cdots,0, & \tilde{d}_i^{1/2}(t), & \tilde{d}_i^{1/2}(t)\tilde{s}_{ii+1}(t), & \cdots, & \tilde{d}_i^{1/2}(t)\tilde{s}_{il}(t)\\ 0,\cdots,0, & 0, & (\delta^{(i)})^{1/2}x_{i+1}^{(i)}(t), & \cdots, & (\delta^{(i)})^{1/2}x_l^{(i)}(t)\end{matrix}\right\} \tag{36}$$

where

$$\left.\begin{aligned}
&\bar{d}_i(t)=\bar{d}_i(t-1)+\delta^{(i-1)}(x_i^{(i-1)}(t))^2\\
&c=\bar{d}_i(t-1)/\bar{d}_i(t)\\
&b=\delta^{(i-1)}x_i^{(i-1)}(t)/\bar{d}_i(t)\\
&\delta^{(i)}=c\,\delta^{(i-1)}
\end{aligned}\right\}\qquad(37)$$

and

$$\left.\begin{aligned}
&x_k^{(i)}(t)=x_k^{(i-1)}(t)-x_i^{(i-1)}(t)\bar{s}_{ik}(t-1)\\
&\bar{s}_{ik}(t)=c\bar{s}_{ik}(t-1)+bx_k^{(i-1)}(t)
\end{aligned}\right\}\ k=i+1,\ldots,l.\qquad(38)$$

For time-varying or non-stationary systems, in order to provide a continual tracking capability, the time-decreasing learning rate $\alpha_c(t)$ in (13) can be replaced by a constant learning rate $\alpha_c$ and a constant forgetting factor $0<\lambda<1$ can be employed instead of $\lambda(t)$ given in (27). For certain applications, it is vital to reduce computational load as much as possible, and the least squares sub-algorithm within the hybrid structure may be replaced by the least mean squares sub-algorithm at the cost of convergence speed. An application of the hybrid clustering and least mean squares algorithm to adaptive channel equalisation based on RBF equalisers is given in [16].

## 4. Application examples

The hybrid clustering and Givens least squares algorithm derived in the previous section was used to identify three systems.

*Example 1.* This is a simulated time-series process. 1500 samples of data were generated by

$$\begin{aligned}
y(t)=&(0.8-0.5exp(-y^2(t-1)))y(t-1)-(0.3+0.9exp(-y^2(t-1)))y(t-2)\\
&+0.1sin(3.1415926y(t-1))+e(t)
\end{aligned}$$

where the noise $e(t)$ was a Gaussian white sequence with mean zero and variance 0.04. The structure of the RBF model was defined by $m=n_y=2$ and $n=30$. The parameters in the hybrid algorithm were chosen to be:

$$\rho=1000,\ \ \lambda_0=0.99,\ \ \lambda(0)=0.95\ \text{ and }\ \alpha_c(0)=0.9.$$

Initial centres were randomly selected from the region $[-2,\ 2]\times[-2,\ 2]$.

The evolution of the mean square error (variance of the residuals) obtained using the hybrid algorithm is plotted in Fig.2. During the recursive identification procedure, the mean square error was reduced from an initial 12dB to the noise floor, approaching -14dB. The distribution of the observations and the final RBF centres are depicted in Fig.3.

Several chi-squared tests for the final RBF network were calculated and they were all within the 95% confidence band. Two typical chi-squared tests and the autocorrelations of $\epsilon(t)$ are shown in Figs. 4 and 5 respectively. The model validity tests confirm that this RBF network is an adequate model for the time series.

It can easily be verified that without the noise $e(t)$ this simulated system generates a stable limit cycle as illustrated in Fig.6. The identified RBF network was used to produce iteratively the network output

$$\hat{y}_d(t)=f_r(\mathbf{v}_d(t)),$$

where $\mathbf{v}_d(t)=[\hat{y}_d(t-1)\hat{y}_d(t-2)]^T$. The iterative network outputs produce a similar limit cycle as can be seen from Fig.7. These two limit cycles have approximately a period of 5 in the sense that every five samples complete approximately a circle ($2\pi$ phase angle) in the state space. However, the amplitudes of response appear to change randomly. 100 samples of the autonomous system outputs and the iterative network outputs are shown in Fig.8. Even though the RBF network was identified using the noisy system observations, the iterative network outputs closely matches the response of the autonomous system. This demonstrates that the identified RBF model does capture the underlying dynamics of the system.

*Example 2.* The process considered is a liquid level system. The system consists of a DC water pump feeding a conical flask which in turn feeds a square tank. The system input is the voltage to the pump motor and the system output is the water level in the conical flask. 1000 samples of data generated in an experiment are shown in Fig.9. The RBF model had a structure of $m=n_y+n_u=3+5$ and $n=40$. The parameters for the identification algorithm were chosen to be:

$$\rho=1000,\ \lambda_0=0.99,\ \lambda(0)=0.95 \text{ and } \alpha_c(0)=0.6.$$

Random initial centres were used.

The evolution of the mean square error is depicted in Fig.10. The mean square error was reduced from the initial value of 2dB to the final value of -26dB. The correlation tests for the identified model are shown in Fig.11. It is observed that at three points the values of $\Phi_{u\epsilon}(k)$ are slightly outside the 95% confidence bands. Several chi-squared tests were also computed. Again only at a few isolated points are statistics slightly outside the confidence band, and this was judged a good result, considering that the data was from a real system and the identification algorithm was a recursive one.

*Example 3.* The data was generated from a heat exchanger and contains 996 samples shown in Fig.12. A description of this process and the experiment design was given by Billings and Fadzil [17]. The dimension of the RBF centres was chosen to be $m=n_y+n_u=6+6$ and the number of centres was $n=90$. The parameters for the recursive algorithm were:

$$\rho=1000,\ \lambda_0=0.99,\ \lambda(0)=0.95 \text{ and } \alpha_c(0)=0.5.$$

Initial centres were set randomly.

During the identification procedure, the mean square error was reduced from the initial 16dB to the final -13.5dB, and the evolution of the mean square error is plotted in Fig.13. The correlation tests and several chi-squared tests were computed, and they were all within the 95% confidence bands. Six chi-squared tests are shown in Fig.14.

## 5. Extensions of the recursive hybrid algorithm

The hybrid algorithm of Section 3 can easily be extended to the multi-input multi-output system. Consider

$$\mathbf{y}(t)=\mathbf{f}_s(\mathbf{y}(t-1),\ldots,\mathbf{y}(t-n_y),\mathbf{u}(t-1),\ldots,\mathbf{u}(t-n_u))+\mathbf{e}(t) \qquad (39)$$

and assume that the dimension of $\mathbf{y}(t)$ is $p$. A $p$-output RBF network is required to model the above system, and the netwok input vector is given by

$$\mathbf{v}(t)=[\mathbf{y}^T(t-1)\cdots\mathbf{y}^T(t-n_y)\mathbf{u}^T(t-1)\cdots\mathbf{u}^T(t-n_u)]^T. \qquad (40)$$

The hidden layer of the RBF network remains unchanged, and the output layer of the network contains $p$ linear combiners. Each of these linear combiners is defined as

$$\hat{y}_i(t)=f_{ri}(\mathbf{v}(t))=\sum_{j=1}^{n}\theta_{ij}\phi(||\mathbf{v}(t)-\mathbf{c}_j||),\ \ 1\le i\le p. \qquad (41)$$

$p$ independent least squares estimators can be employed to identify the connection weights of these linear combiners. The above discussion can obviously be applied to the multi-input multi-output NARMAX mdoel

$$\mathbf{y}(t)=\mathbf{f}_s(\mathbf{y}(t-1),\ldots,\mathbf{y}(t-n_y),\mathbf{u}(t-1),\ldots,\mathbf{u}(t-n_u),\mathbf{e}(t-1),\ldots,\mathbf{e}(t-n_e))+\mathbf{e}(t). \qquad (42)$$

## 6. Conclusions

A hybrid clustering and Givens least squares algorithm has been developed for the recursive identification of non-linear systems using a radial basis function network. This hybrid algorithm combines the supervised least squares method with an unsupervised clustering technique. The centres of the radial basis function network are adjusted using the *n*-means clustering technique and the connection weights of the network are updated using the least squares principle. These two learning rules are implemented recursively and are thus appropriate for real-time or adaptive applications. Furthermore, they are linear learning rules, thus guaranteeing rapid convergence. Using three examples, a simulated non-linear time-series and two real processes, it has been shown that this hybrid approach offers a powerful on-line identification algorithm for radial basis function models.

## 7. References

[1] S. Chen, S.A. Billings, C.F.N. Cowan and P.M. Grant, "Non-linear systems identification using radial basis functions", *Int. J. Systems Sci.*, Vol.21, No.1, 1990, pp.2513-2539.

[2] S. Chen, S.A. Billings, C.F.N. Cowan and P.M. Grant, "Practical identification of NARMAX models using radial basis functions", *Int. J. Control*, Vol.52, No.6, 1990, pp.1327-1350.

[3] J. Moody and C. Darken, "Fast-learning in networks of locally-tuned processing units", *Neural Computation*, Vol.1, No.2, 1989, pp.281-294.

[4] W.E. Gentleman, "Least squares computations by Givens transformations without square roots", *J. Inst. Maths. Applics.*, Vol.12, 1973, pp.329-336.

[5] W.E. Gentleman and H.T. Kung, "Matrix triangularization by systolic arrays", *Real-Time Signal Processing IV, Proc. SPIE*, Vol.298, pp.19-26, 1981.

[6] I.J. Leontaritis and S.A. Billings, "Input-output parametric models for non-linear systems, Part I: deterministic non-linear systems; Part II: stochastic non-linear systems", *Int. J. Control*, Vol.41, No.2, 1985, pp.303-344.

[7] S. Chen and S.A. Billings, "Representation of non-linear systems: the NARMAX model", *Int. J. Control*, Vol.49, No.3, 1989, pp.1013-1032.

[8] M.J.D. Powell, "Radial basis function approximations to polynomials", *Proc. 12th Biennial Numerical Analysis Conference*, Dundee, pp.223-241, 1987.

[9] S.A. Billings and W.S.F. Voon, "Correlation based model validity tests for non-linear models", *Int. J. Control*, Vol.44, No.1, 1986, pp.235-244.

[10] T. Bohlin, "Maximum-power validation of models without higher-order fitting", *Automatica*, Vol.4, 1978, pp.137-146.

[11] I.J. Leontaritis and S.A. Billings, "Model selection and validation methods for non-linear systems", *Int. J. Control,* Vol.45, No.1, 1987, pp.311-341.

[12] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis,* John Wiley and Sons, New York, 1973.

[13] J. MacQueen, "Some methods for classification and analysis of multivariate observations", *Proc. 5th Berkeley Symp. on Math. Stat. and Prob., Prat I,* L.M. LeCam and J. Neyman, eds., University of California Press, pp.281-297, 1967.

[14] T. Kohonen, *Self-Organization and Associative Memory,* Springer-Verlag, Berlin, 1987.

[15] L. Ljung and T. Söderström, Theory and Practice of Recursive Identification, MIT Press, Cambridge, 1983.

[16] S. Chen, C.F.N. Cowan, B. Mulgrew and P.M. Grant, "Adaptive radial basis function equalizer using a hybrid clustering and LMS algorithm", Submitted to *IEEE Trans. Communications,* 1991.

[17] S.A. Billings and M.B. Fadzil, "The practical identification of systems with nonlinearities", *Proc. 7th IFAC Symp. on Identification and System Parameter Estimation,* York, U.K., pp.155-160, 1985.

**List of Figures**

Fig.1. Radial Basis Function Network.

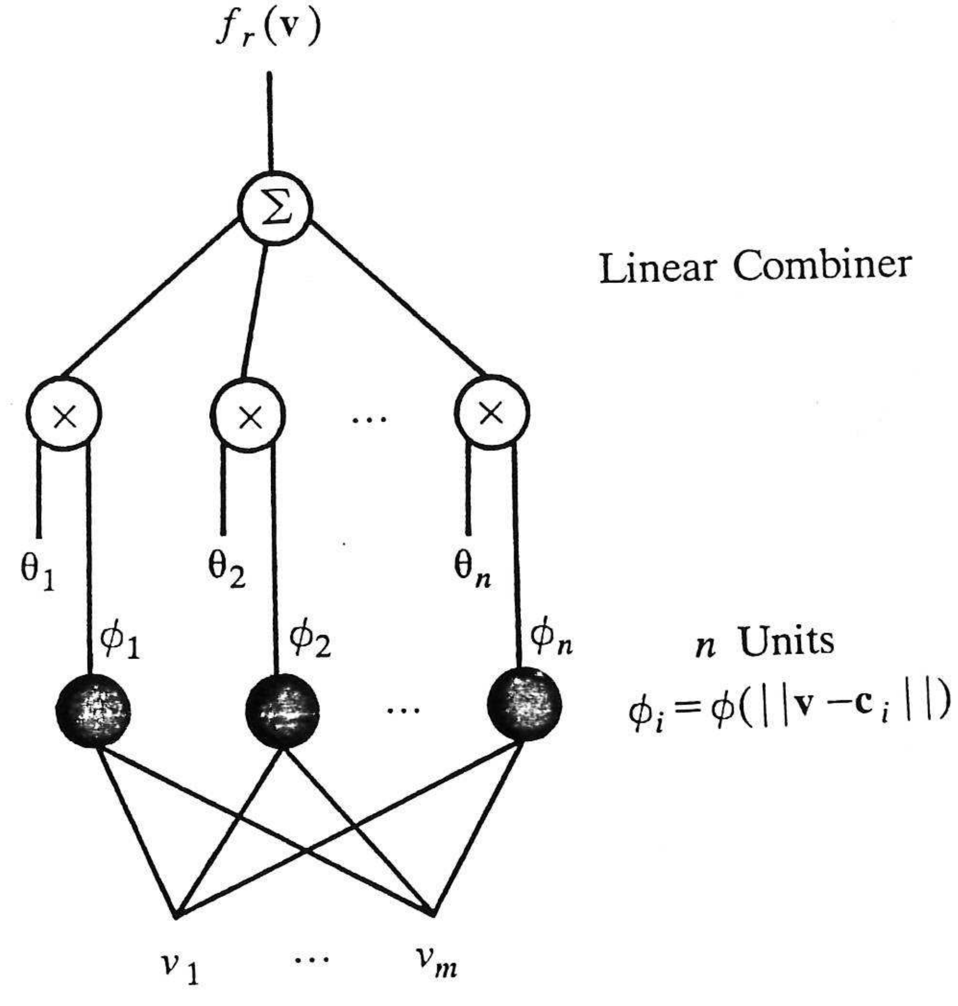


Fig.2. Evolution of Mean Square Error (Example 1).

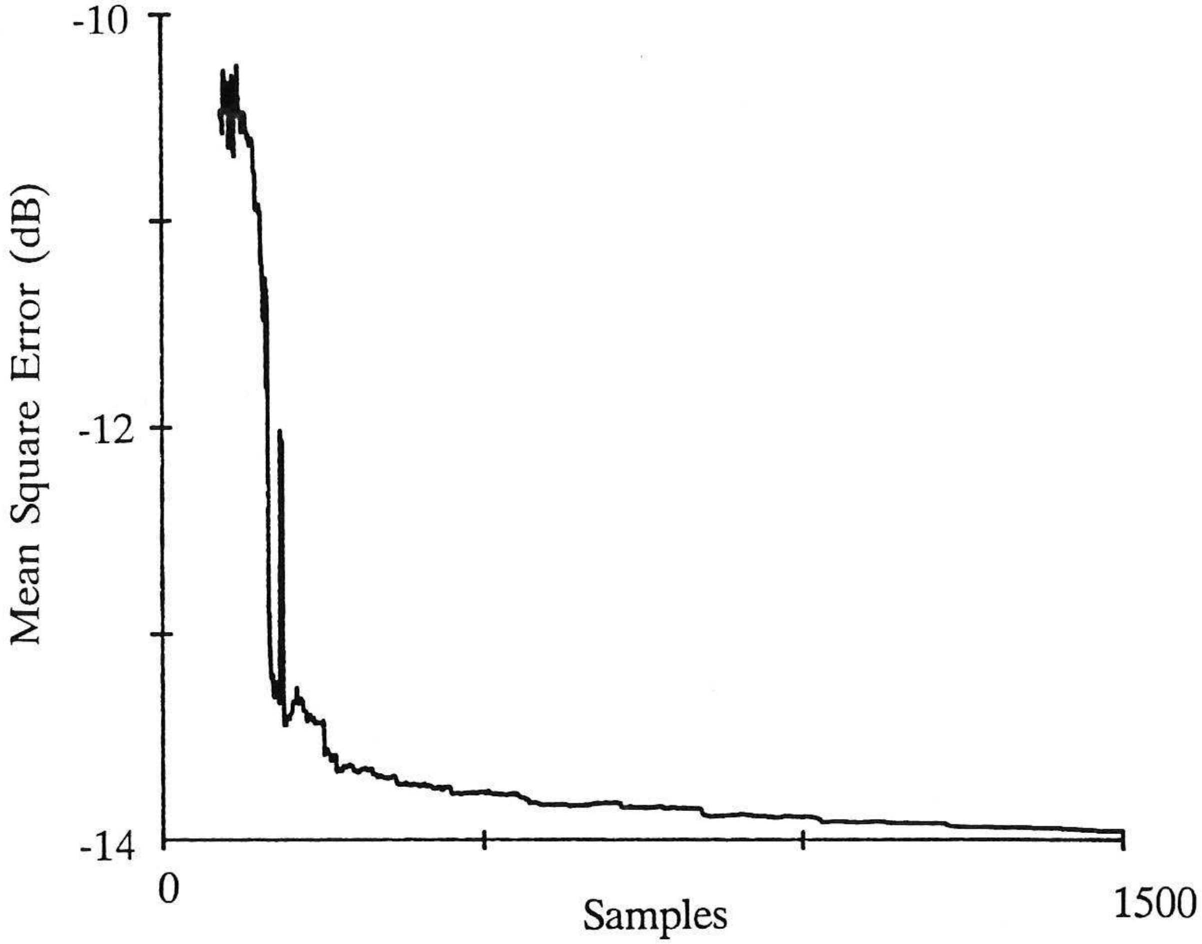


Fig.3. Distribution of Observations and RBF Centres. 1500 observation samples, □: position of RBF centre.

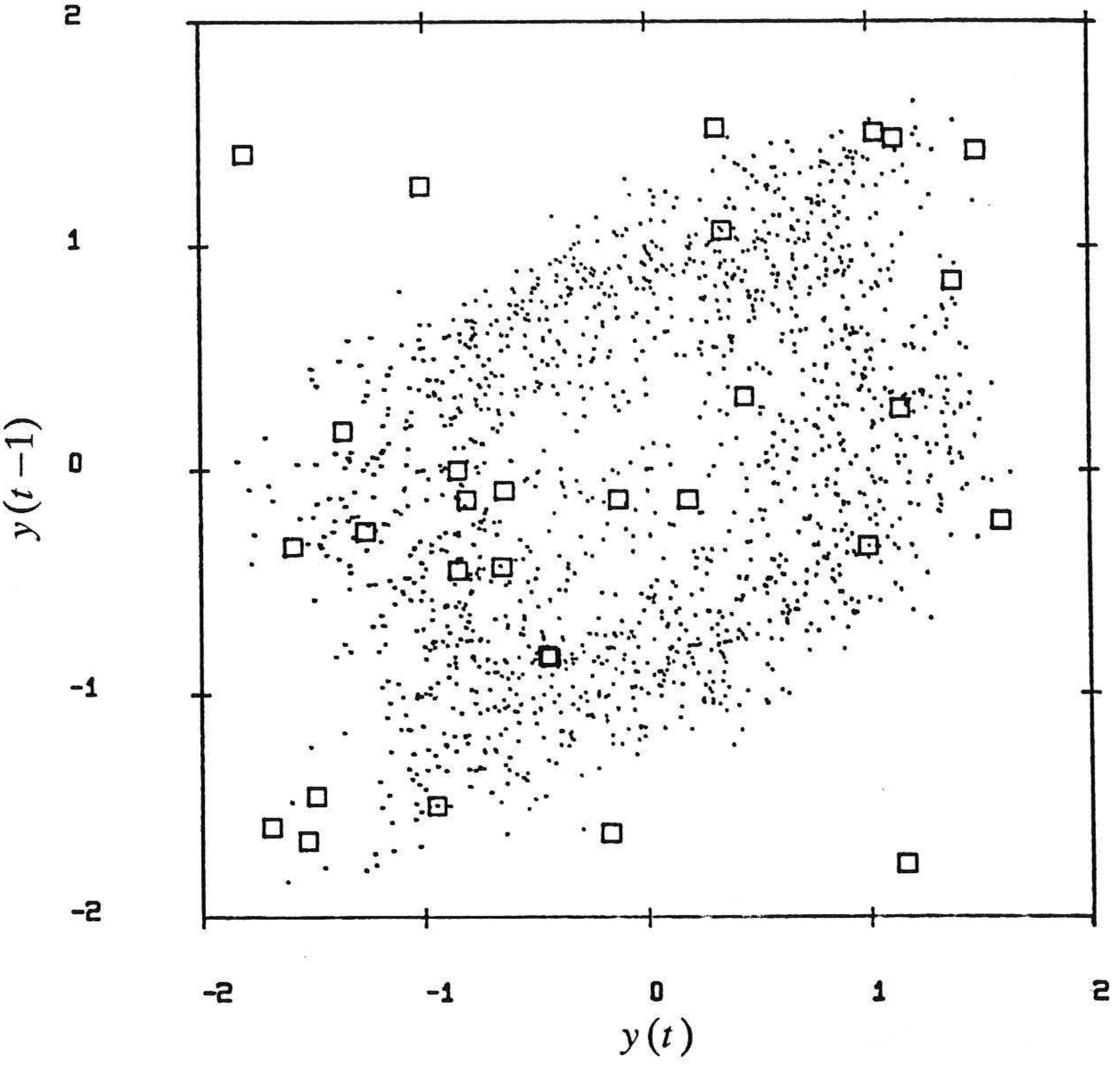


Fig.4. Chi-Squared Tests (Example 1). (a) $\omega(t)=\epsilon^2(t-1)y(t-1)$, (b) $\omega(t)=\epsilon(t-1)y^2(t-1)$, —#— 95% confidence band.

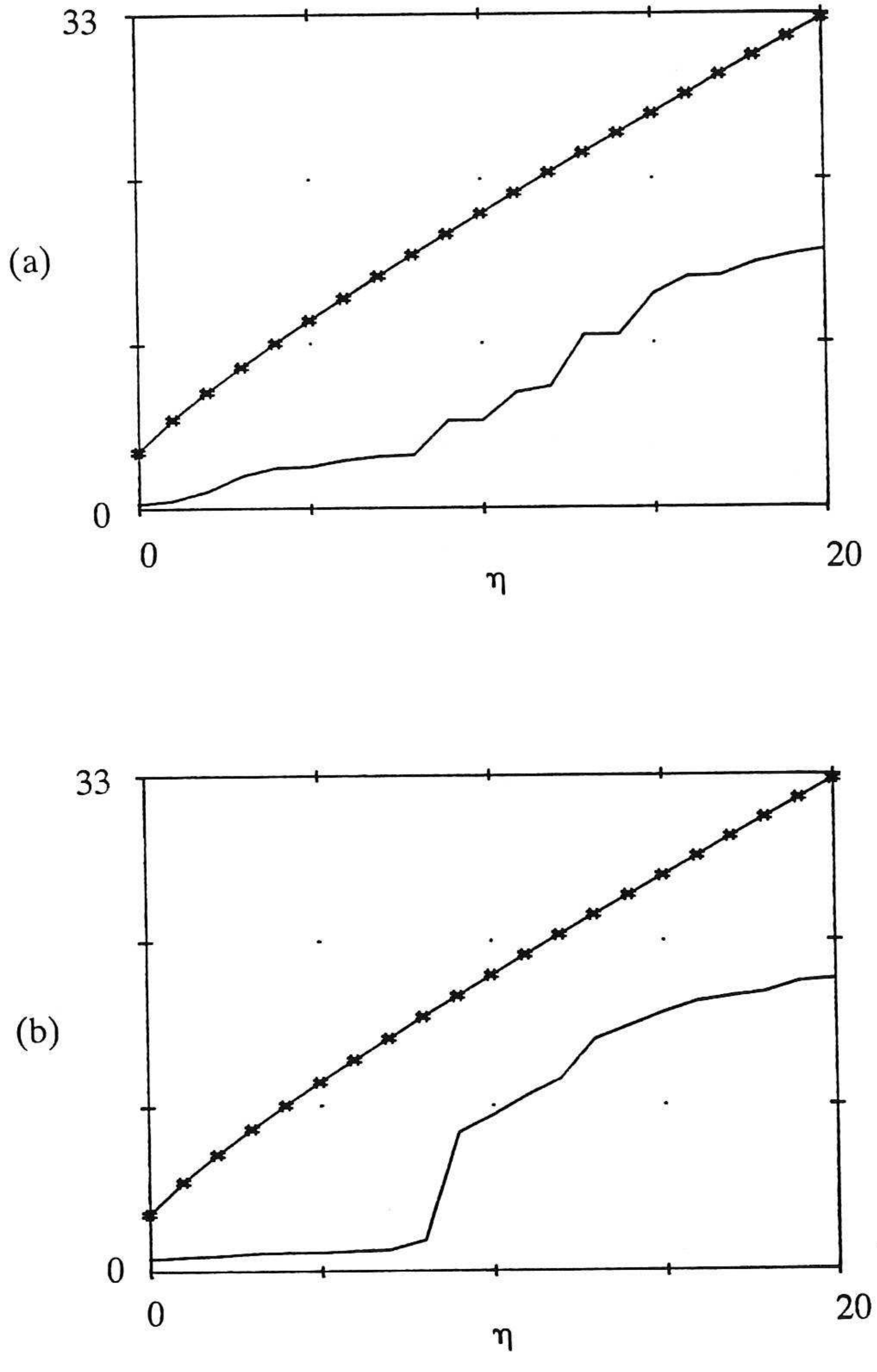


Fig.5. Autocorrelations of Residuals (Example 1). —·— 95% confidence band.

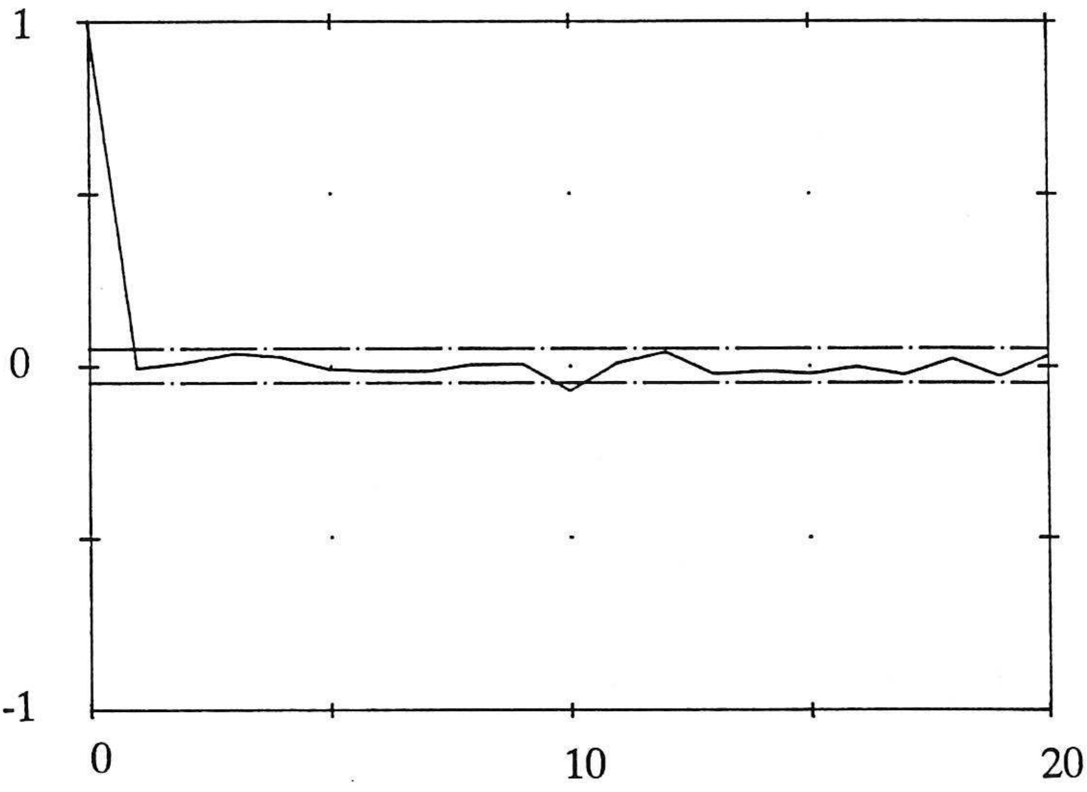


Fig.6. Limit Cycle Generated by Autonomous System Response. 1500 samples, initial condition: $y(0)=0.1$ and $y(-1)=0.01$.

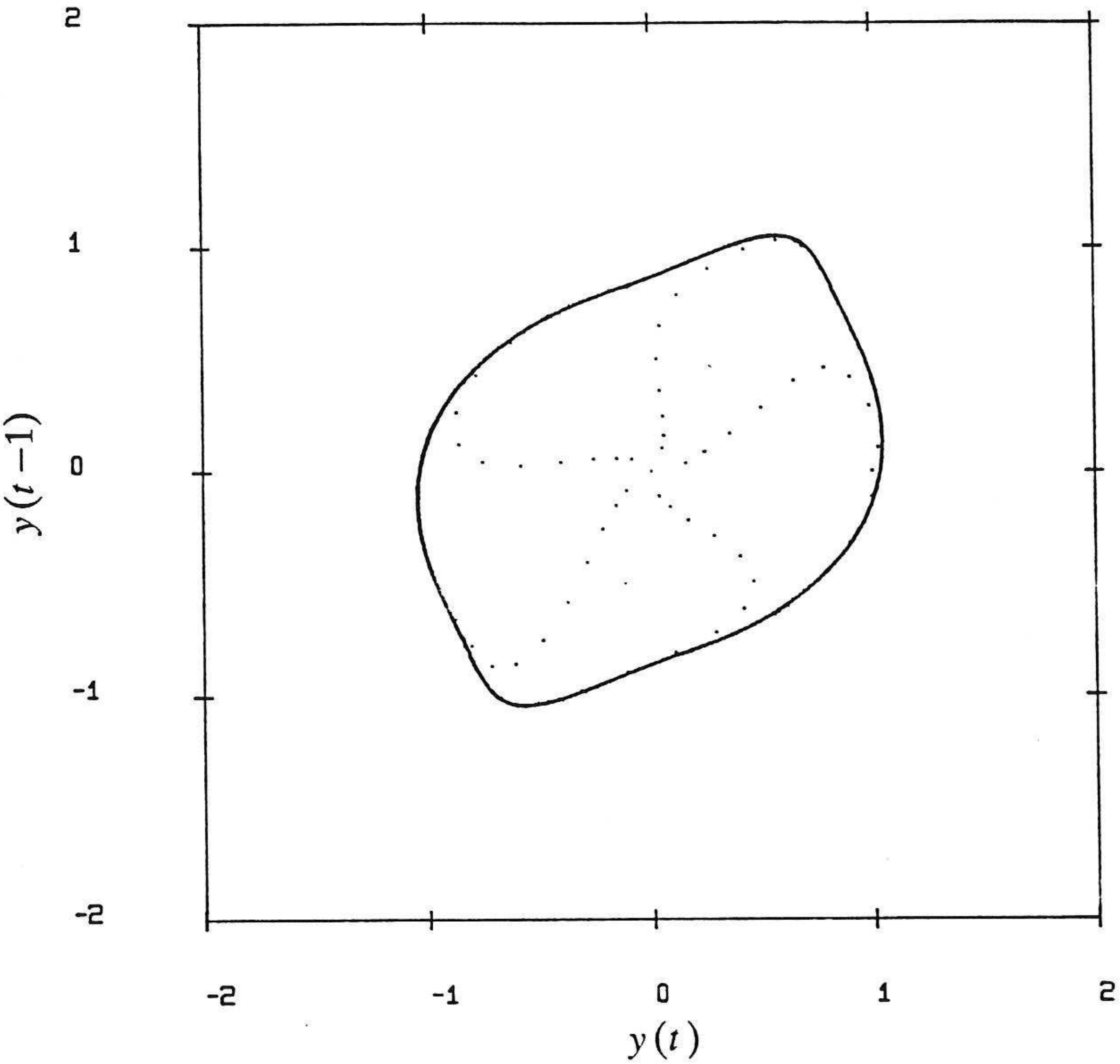


Fig.7. Limit Cycle Generated by Iterative Network Response. 1500 samples, initial condition: $y(0)=0.1$ and $y(-1)=0.01$.

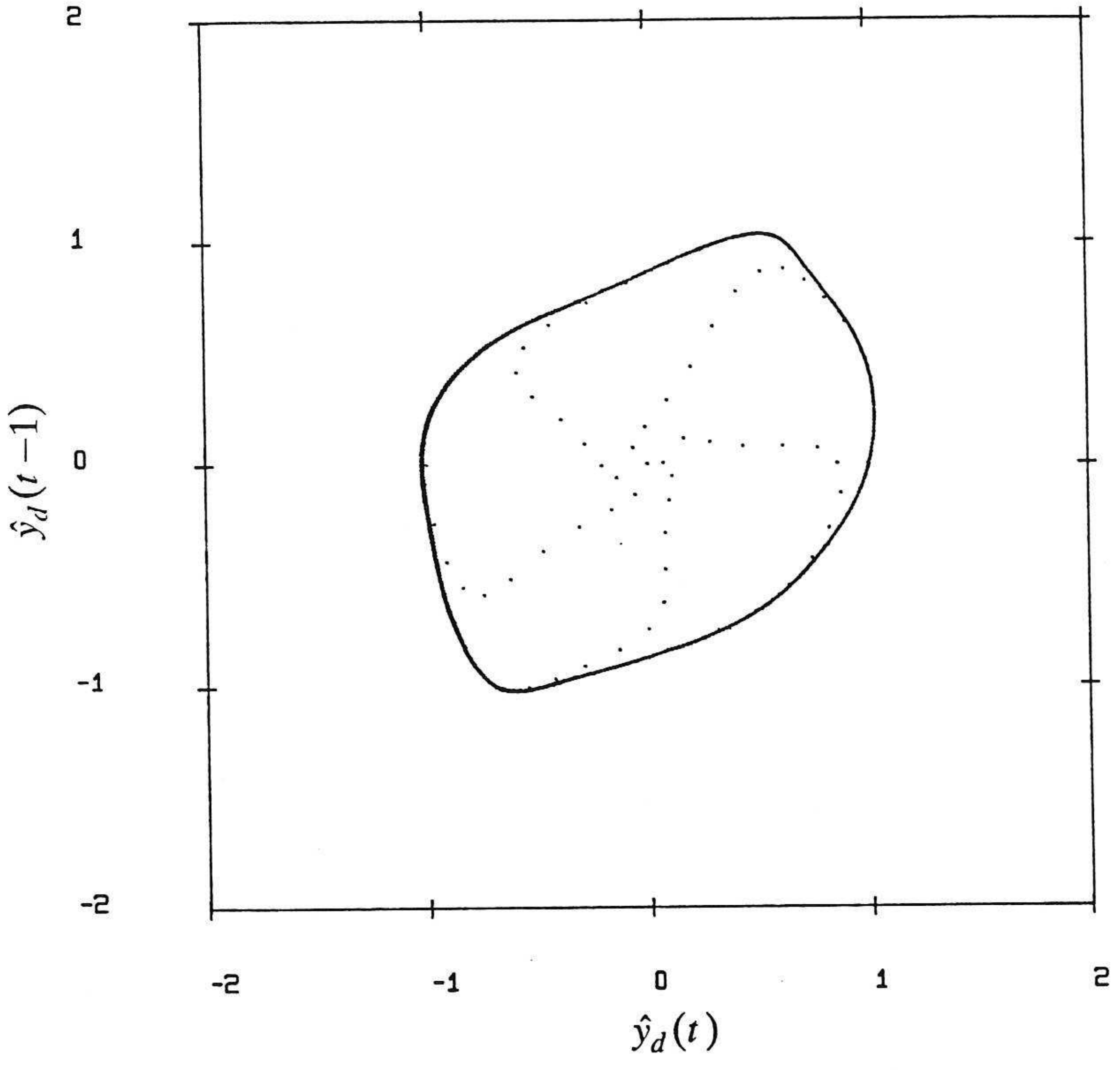


Fig.8. Autonomous System Response (a) and Iterative Network Response (b). Initial condition: $y(0)=0.1$ and $y(-1)=0.01$.

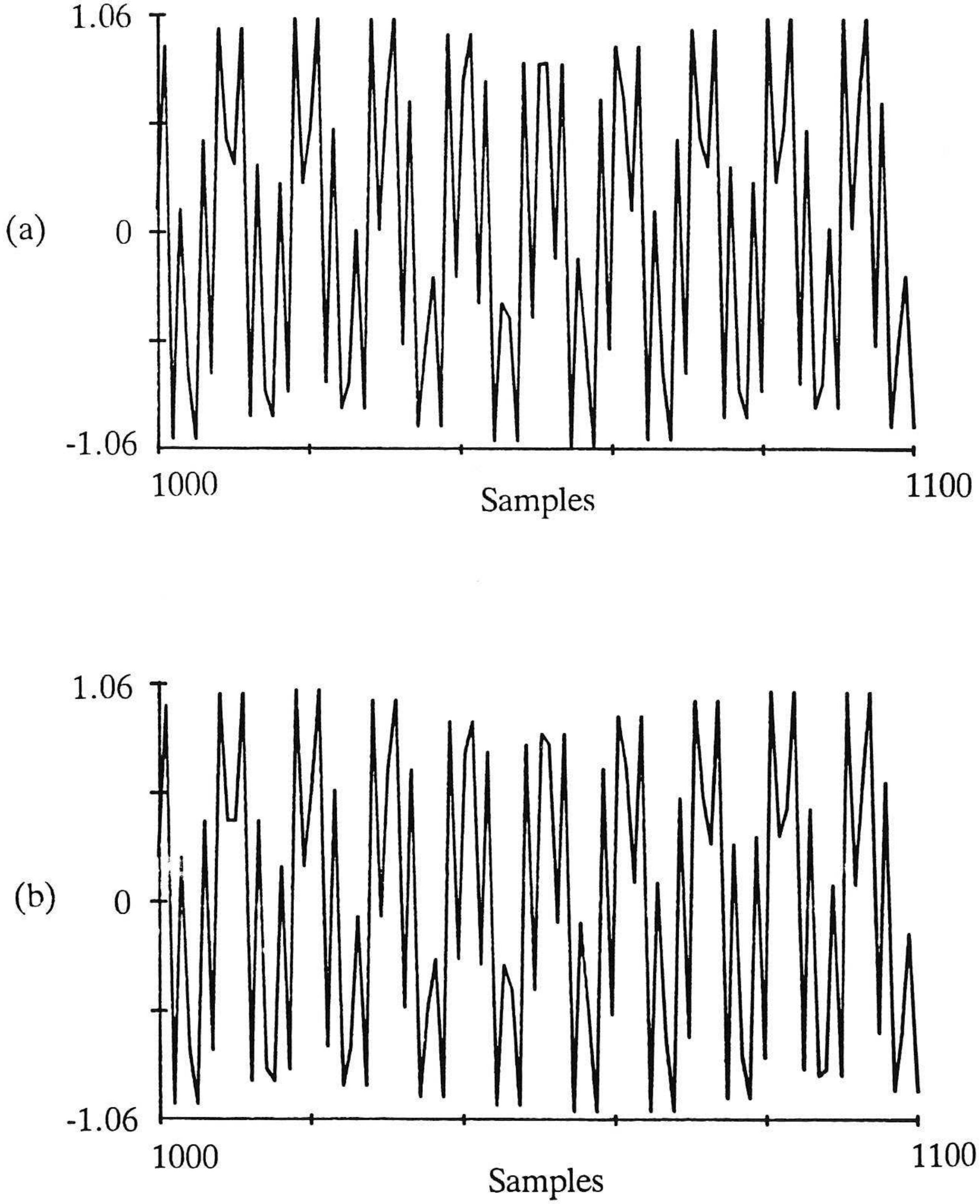


Fig.9. System Outputs and Inputs (Example 2). (a) outputs $y(t)$, (b) inputs $u(t)$.

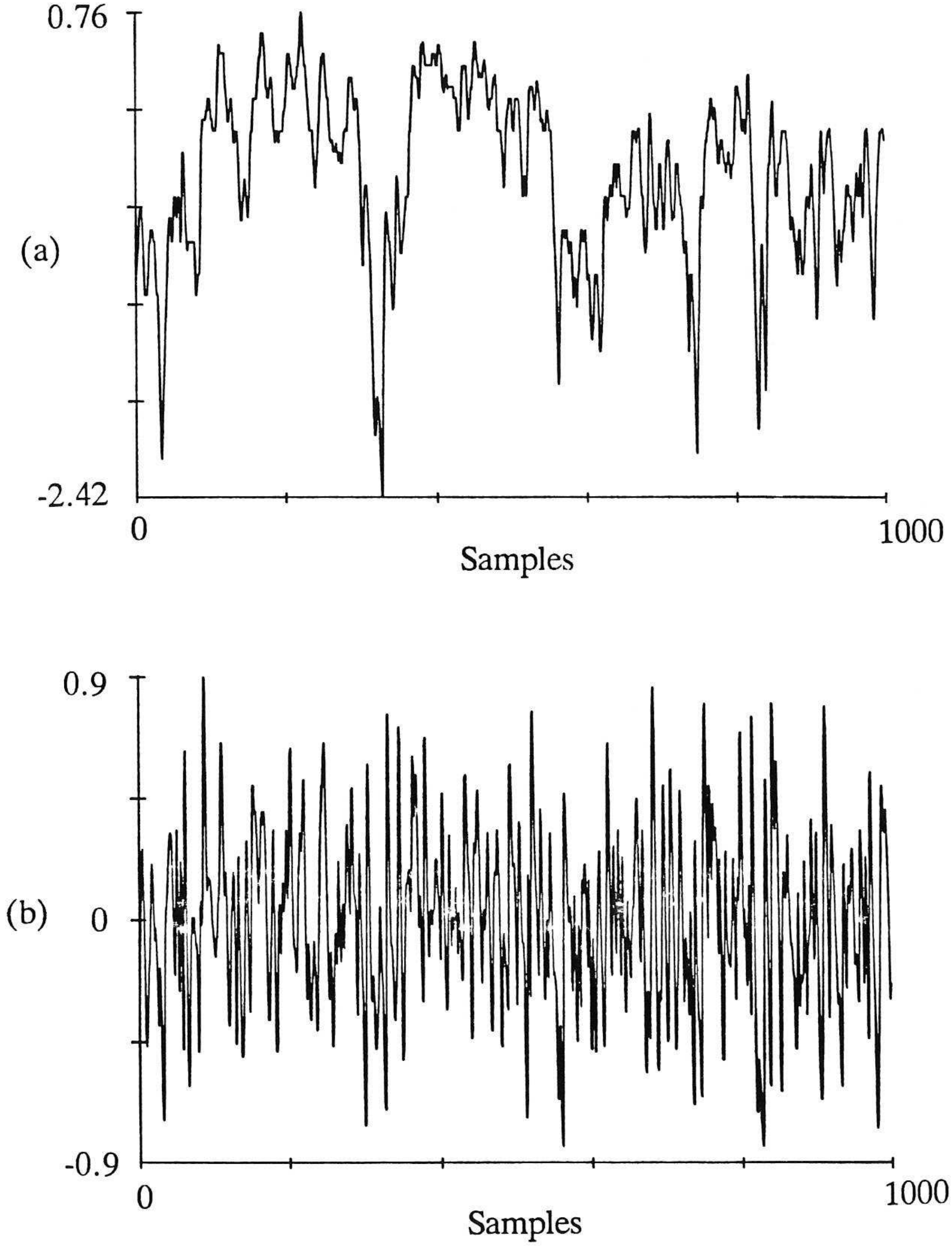


Fig.10. Evolution of Mean Square Error (Example 2).

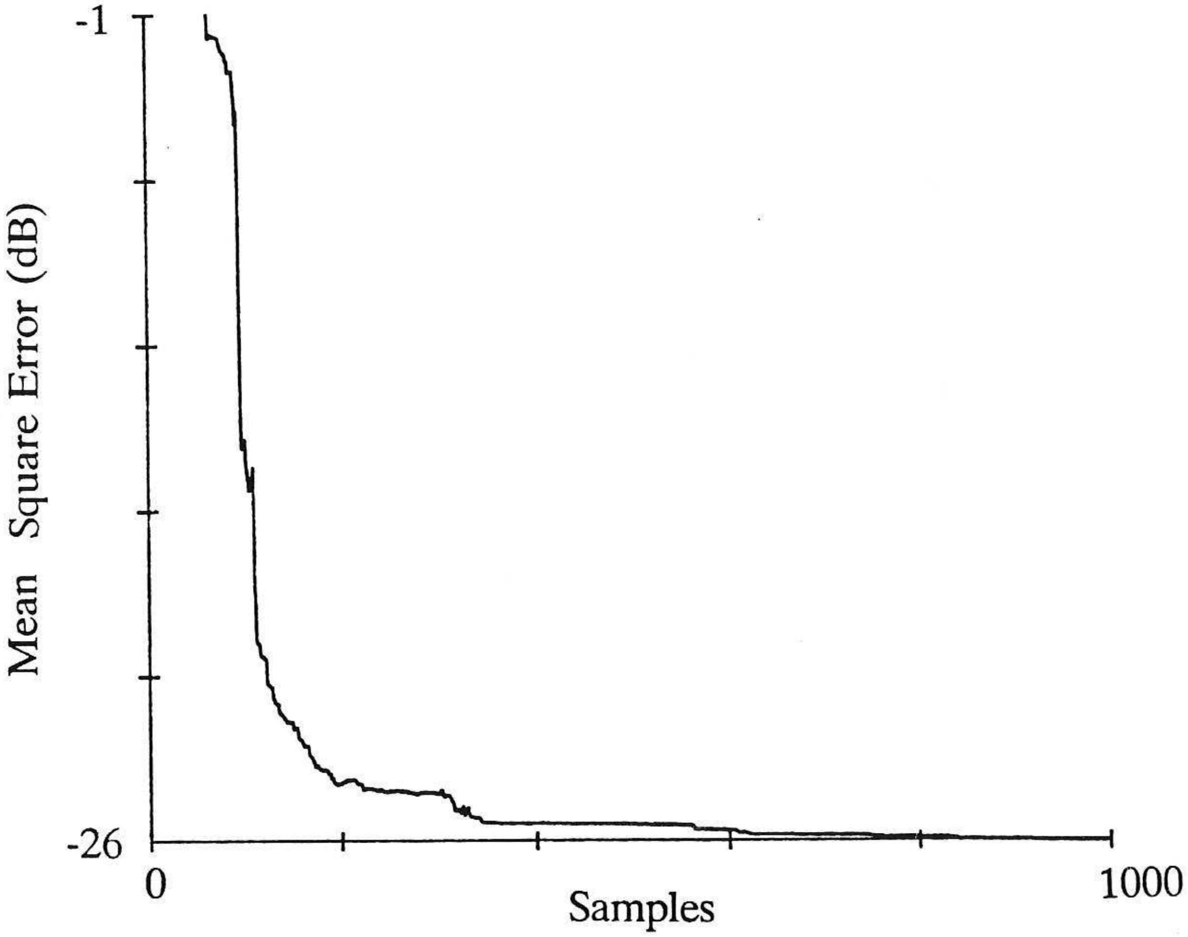


Fig.11. Correlation Tests (Example 2). (a) $\Psi_{\epsilon\epsilon}(k)$, (b) $\Psi_{\epsilon(\epsilon u)}(k)$, (c) $\Psi_{u\epsilon}(k)$, (d) $\Psi_{u^{2'}\epsilon}(k)$, (e) $\Psi_{u^{2'}\epsilon^2}(k)$, —·— 95% confidence band.

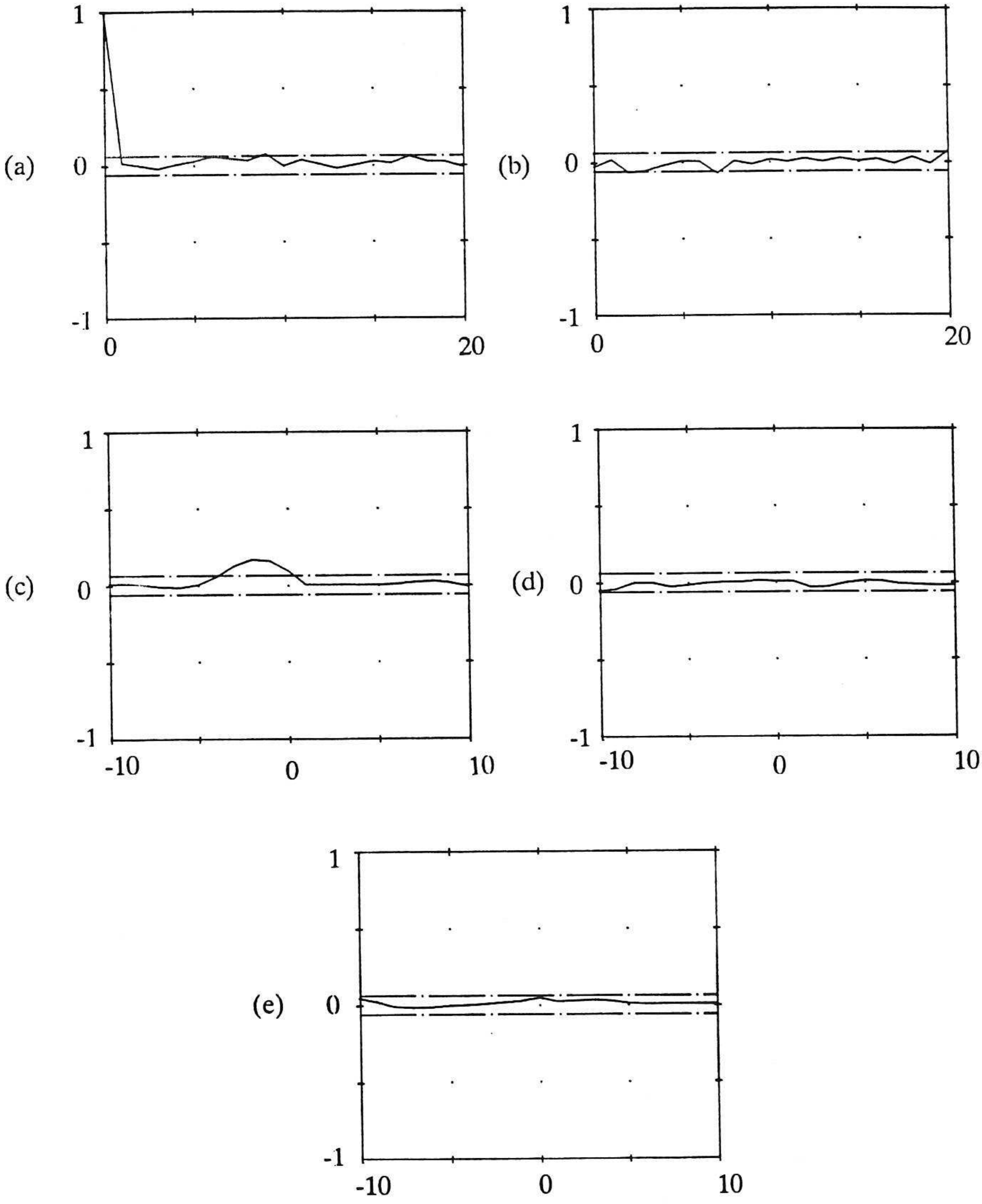
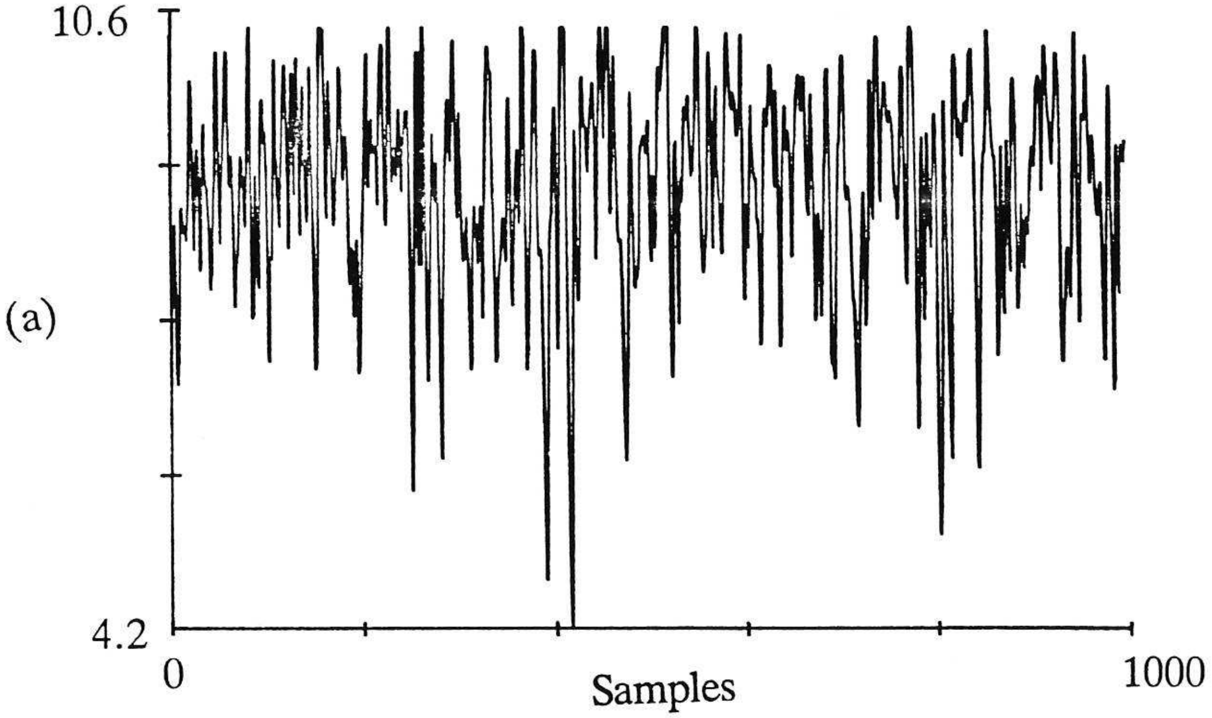


Fig.12. System Outputs and Inputs (Example 3). (a) outputs $y(t)$, (b) inputs $u(t)$.



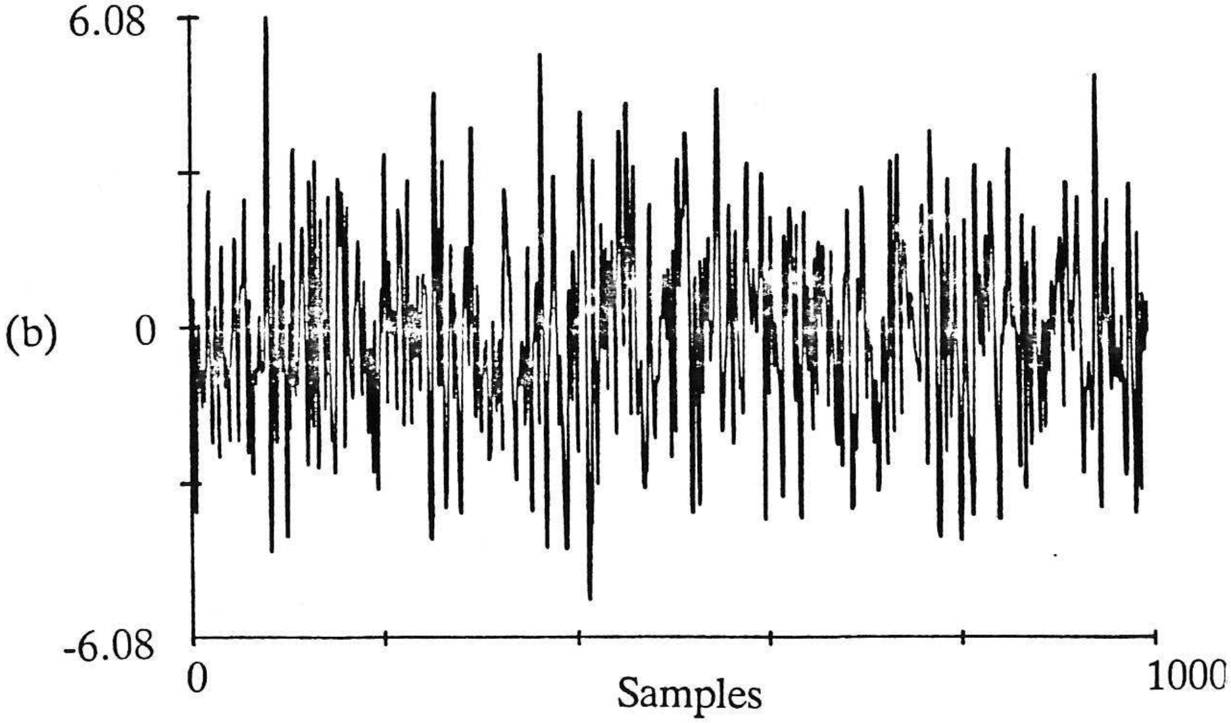


Fig.13. Evolution of Mean Square Error (Example 3).

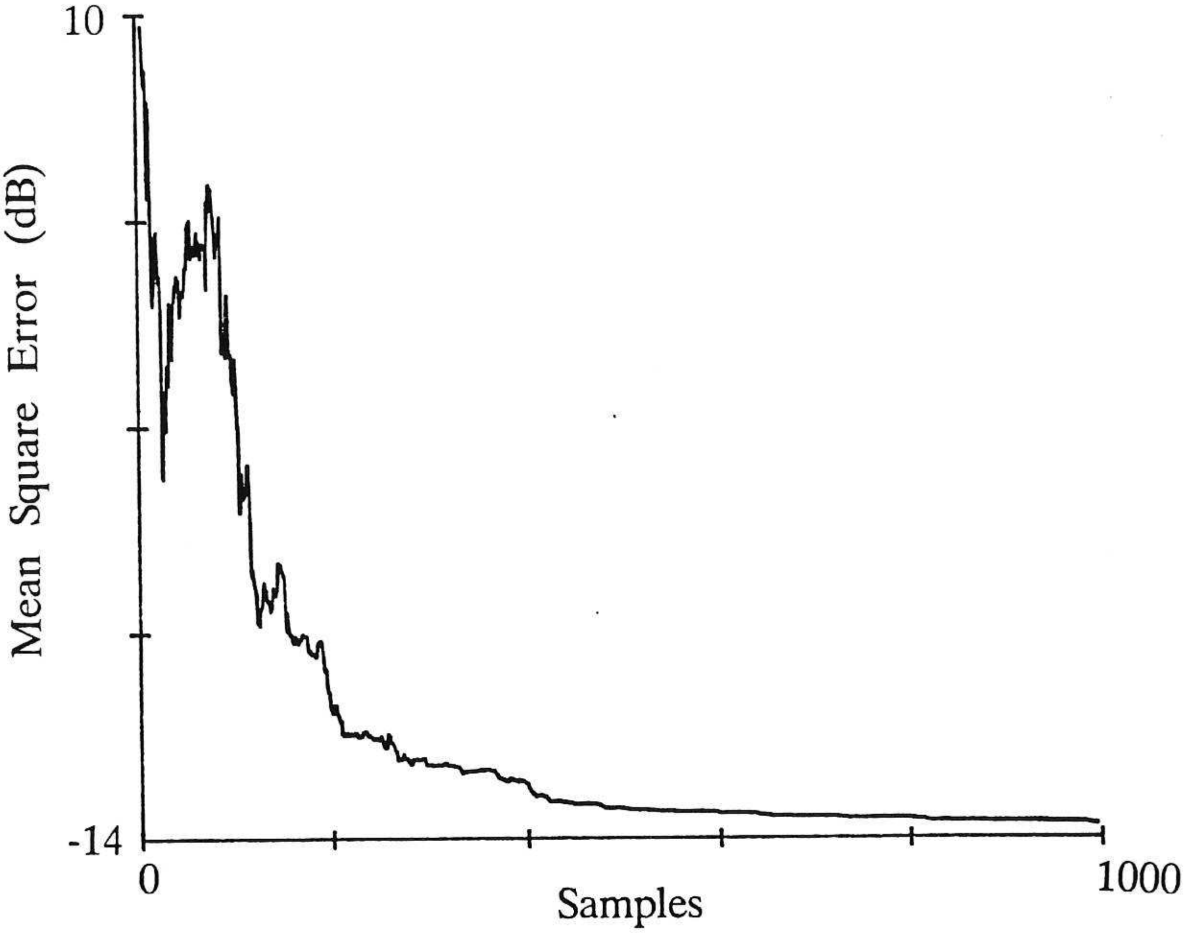
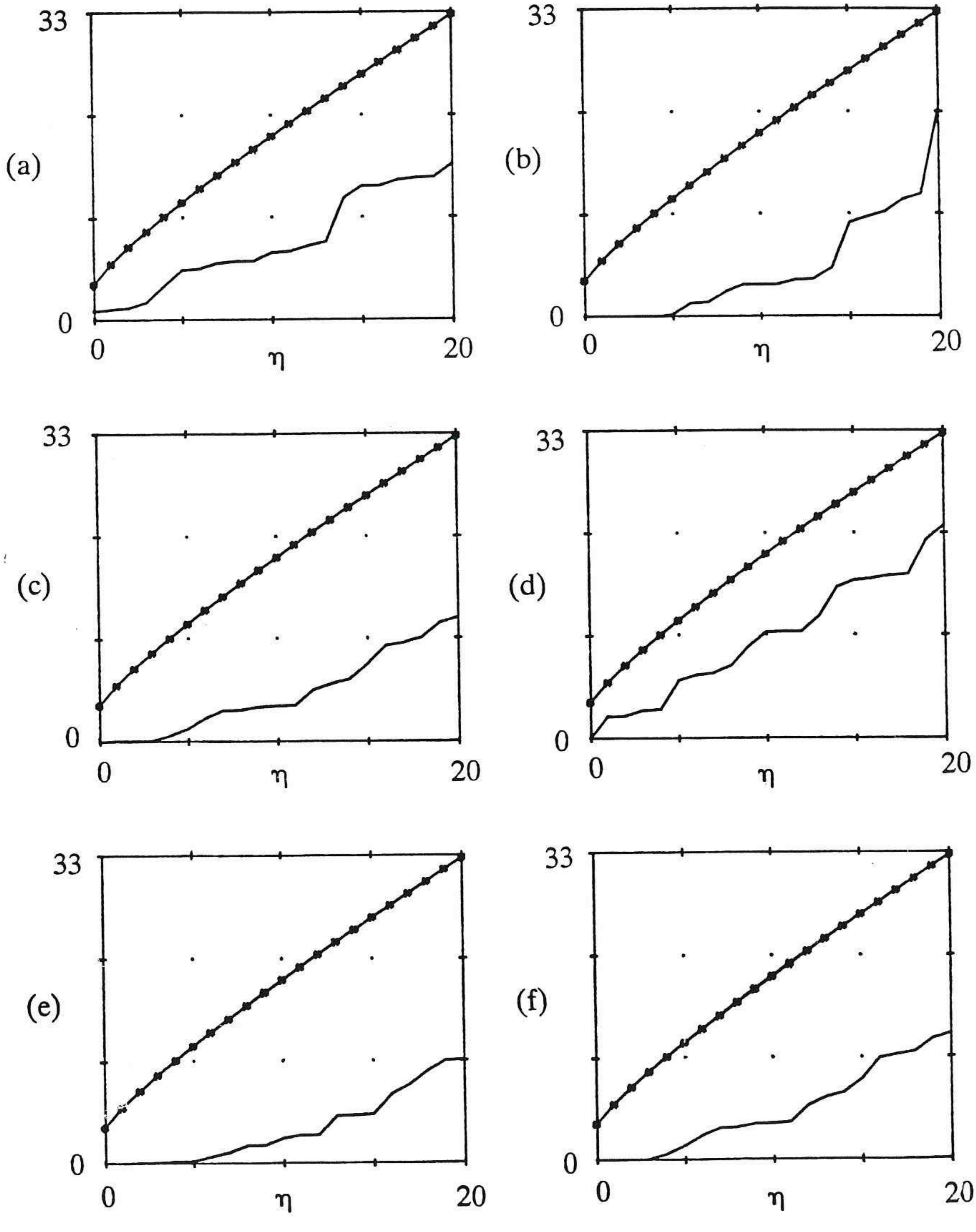


Fig.14. Chi-Squared Tests (Example 3). (a) $\omega(t)=\epsilon(t-1)$, (b) $\omega(t)=u(t-1)$, (c) $\omega(t)=y(t-1)$, (d) $\omega(t)=\epsilon^2(t-1)$, (e) $\omega(t)=u^2(t-1)$, (f) $\omega(t)=y^2(t-1)$, —#— 95% confidence band.


33
0
(a)
0
η
20
33
0
(b)
0
η
20
33
0
(c)
0
η
20
33
0
(d)
0
η
20
33
0
(e)
0
η
20
33
0
(f)
0
η
20