STOCHASTIC NEURAL NETWORKS WITH APPLICATIONS
TO NONLINEAR TIME SERIES

by

Tze Leung Lai
Samuel Po-Shing Wong

DECEMBER 2000

Department of Statistics
STANFORD UNIVERSITY
Stanford, California
STOCHASTIC NEURAL NETWORKS WITH APPLICATIONS
TO NONLINEAR TIME SERIES

by

TZE LEUNG LAI
Department of Statistics
Stanford University

SAMUEL PO-SHING WONG
Department of Information and Systems Management
Hong Kong University of Science and Technology

DECEMBER 2000

This research was supported in part by National Science Foundation grant
DMS-00-72523 and National Security Agency grant MDA-904-00-1-0018

Department of Statistics
Sequoia Hall
STANFORD UNIVERSITY
Stanford, California 94305-4065

http://www-stat.stanford.edu
Stochastic Neural Networks with Applications to Nonlinear Time Series

TZE LEUNG LAI AND SAMUEL PO-SHING WONG*

December 19, 2000

Abstract

Neural networks have a burgeoning literature in nonlinear time series. We consider here a variant of the conventional neural network model, called the stochastic neural network, that can be used to approximate complex nonlinear stochastic systems. We show how the EM algorithm can be used to develop efficient estimation schemes that have much lower computational complexity than those for conventional neural networks. This enables us to carry out model selection procedures such as the BIC to choose the number of hidden units and the input variables for each hidden unit. On the other hand, stochastic neural networks are shown to have the universal approximation property of neural networks. Other important properties of the proposed model are also given, and model-based multi-step ahead forecasts are provided. For illustration, we fit stochastic neural network models to several real and simulated time series. Our results show that the fitted models improve postsample forecasts over conventional neural networks and other nonlinear/nonparametric models.

KEY WORDS: EM; Hidden Markov models; Model selection; Neural networks; Nonlinear stochastic systems.

*Tze Leung Lai is Professor, Department of Statistics, Stanford University. Samuel Po-Shing Wong is Assistant Professor, Department of Information and Systems Management, Hong Kong University of Science and Technology. Lai's research was supported by the National Science Foundation, the National Security Agency and the Center for Advanced Study in the Behavioral Sciences. Wong's research was supported by the Research Grants Council in Hong Kong.
1. INTRODUCTION

Linear time series models (such as ARMA, ARMAX and ARIMA) have been well developed and widely used because of their tractability and ease of interpretation. However, they may be overly simplistic and fail to capture many essential features of the underlying process, resulting in unsatisfactory multi-step ahead predictions. Nonlinear time series analysis has gained much attention during the past two decades. A number of nonlinear models have been developed, following Tong’s seminal work on TAR (threshold autoregressive) models. These nonlinear time series models, a comprehensive account of which is given by Tong (1990), assume a priori certain parametric forms whose appropriateness may be difficult to justify in real applications, as pointed out by Chen and Tsay (1993a) who proposed for empirical modeling of time series data functional-coefficient autoregressive (FAR) models of the form

\[ y_t = f_1(Y_t^*)y_{t-1} + \cdots + f_p(Y_t^*)y_{t-p} + \epsilon_t, \tag{1} \]

in which the \( \epsilon_t \) are i.i.d. zero-mean random variables, \( Y_t^* = (y_{t-i_1}, \ldots, y_{t-i_d})^T \) with \( i_1 < \ldots < i_d \) chosen from \( \{1, \ldots, p\} \) and \( f_1, \ldots, f_p \) are unspecified functions to be estimated from the data by local (i.e., nearest-neighbor-type) linear regression methods. Because of the sparsity of the data in high dimensions, these local regression methods typically require \( d \) to be 1 or 2. To deal with nonparametric regression in higher dimensions, Chen and Tsay (1993b) considered additive autoregressive models of the form \( y_t = f_1(y_{t-i_1}) + \cdots + f_d(y_{t-i_d}) + \epsilon_t \), in which the \( f_i \) are unspecified functions to be estimated nonparametrically by GAM (generalized additive model) techniques of Hastie and Tibshirani (1990). Making use of Friedman’s (1991) multivariate adaptive splines (MARS), Lewis and Stevens (1991) proposed another method called ASTAR (adaptive spline threshold autoregression) for threshold autoregressive modeling.

In this paper we develop a considerably more flexible nonlinear time series modeling methodology by making use of single-layer neural networks. We circumvent various computational and statistical difficulties that accompany this greater flexibility by making use of a randomized version of neural networks, called stochastic neural networks (SNN). Section 2 introduces this class of nonlinear time series models and shows how they broaden the scope of the aforementioned models. Section 3 presents some probabilistic properties of SNNs and studies multi-step ahead forecasts in this connection. Section 4 considers estimation in SNN. In particular, it shows that the EM algorithm can be decoupled into separate logistic regressions, one for each hidden unit, and a weighted linear least squares regression at each iteration. Making use of this attractive computational fea-
ture, it is possible to implement model selection procedures to determine the number of hidden units and the set of lagged observations and exogenous variables. The details are given in Section 5. Section 6 illustrates SNN modeling and forecasting with both real and simulated examples. Some concluding remarks are given in Section 7, where other approaches to building SNN models for time series data are also discussed.

2. NEURAL NETWORK MODELS AND STOCHASTIC NEURAL NETWORKS

Let $x_t = (y_{t-1}, \ldots, y_{t-p})^T$. To begin with, consider the general form of Tong’s TAR model

$$y_t = \sum_{j=1}^{J} (\beta_j + b_j^T x_t) I(r_{j-1} \leq y_{t-d} < r_j) + \epsilon_t,$$

where the $\epsilon_t$ are i.i.d. zero-mean random variables, $1 \leq d \leq p$ represents the delay and the $r_j$ represent thresholds at which the level and the vector of autoregressive parameters change into new values from those given by $\beta_j$ and $b_j$ ($r_0 = -\infty < r_1 < \ldots < r_J = \infty$). Here and in the sequel we used $I(A)$ to denote the indicator variable (with $I(A) = 1$ if $A$ occurs and $I(A) = 0$ otherwise). Due to difficulties in determining the thresholds empirically, the TAR model is usually limited to a single threshold $r_1$ (i.e., $J = 2$) and the threshold does not involve interactions among the lagged predictor variables. Moreover, the regression function of $y_t$ on $x_t$ has jump discontinuities. Lewis and Stevens (1991) use Friedman’s MARS algorithm to fit the considerably more flexible ASTAR model

$$y_t = \beta_0 + \sum_{j=1}^{J} \beta_j \prod_{k=1}^{m(j)} \delta_{k(j)}(y_{t-k(j)} - r_{k(j)})_+ + \epsilon_t,$$

in which $k(j)$ are distinct integers from $\{1, \ldots, p\}$, $\delta_{k(j)} = 1$ or $-1$, and $a_+$ denotes $\max(a, 0)$. Unlike TAR, the ASTAR model allows interactions among the lagged predictor variables and threshold values and has continuous regression functions $E(y_t|x_t)$. However, the adaptive choice of $k(j)$, $m(j)$ and the thresholds $r_{k(j)}$ in MARS is too complex for theoretical analysis of the time series model (3) and its computational task becomes prohibitive for fitting long series.

2.1 Neural Networks

One can regard (3) as an approximation, with tensor products of univariate linear splines as basis functions, to the nonlinear time series model

$$y_t = f(y_{t-1}, \ldots, y_{t-p}) + \epsilon_t$$

(4)
in which $f$ is unknown; the MARS algorithm provides a data-dependent construction of the approximation to $f$. An alternative choice of basis functions is provided by single-layer neural networks, leading to time series models of the form

$$y_t = \beta_0 + \sum_{j=1}^{J} \beta_j \psi(\alpha_j + a_j^T x_t) + \epsilon_t,$$  \hspace{1cm} (5)

where $\psi(u) = 1/(1 + e^{-u})$ is the logistic function, with $\psi(\infty) = 0$ and $\psi(-\infty) = 1$. Barron (1993) has proved the following "universal approximation" property of neural networks: Suppose $f : \mathbb{R}^p \to \mathbb{R}$ has Fourier transform $\hat{f}$ such that $\int_{\mathbb{R}^d} |\hat{f}(w)| \|w\|dw = C < \infty$. Let $\mu$ be a probability measure on the ball $B(r) = \{x \in \mathbb{R}^p : \|x\| \leq r\}$. Then given any $J \geq 1$, there exist $\beta_0, \beta_j, \alpha_j$ and $a_j$ ($j = 1, \ldots, J$) such that

$$\int_{B(r)} \left\{ f(x) - \beta_0 - \sum_{j=1}^{J} \beta_j \psi(\alpha_j + a_j^T x) \right\}^2 d\mu(x) \leq (2rC)^2 / J.$$

(6)

The McCulloch-Pitts (1943) model of a neuron is based on a vector $x$ of stimuli or input variables so that the neuron is activated when some linear combination $a^T x$ of the stimuli reaches or exceeds a threshold $-\alpha$, i.e., $\alpha + a^T x \geq 0$. The logistic function $\psi(u/c)$ can be used as a smooth alternative to the indicator function $I(u \geq 0) = \lim_{c \to 0} \psi(u/c)$ for $u \neq 0$, so $\sum_{j=1}^{J} \beta_j \psi(\alpha_j + a_j^T x)$ can be regarded as the total output of a layer of $J$ neurons, assuming that the $j$th neuron has an output $\beta_j$ after activation (defined by the logistic function). In view of (6), we can approximate the general autoregressive model (4) by the neural network (5) if $J$ is sufficiently large. These neural network models have a burgeoning literature in time series modeling and prediction (cf. Weigend, Rumelhart and Huberman (1991), Weigend and Gershenfeld (1993)) and in applications to finance and econometrics (cf. White (1992), Hutchinson, Lo and Poggio (1994)) and to engineering (cf. Sanner and Slotine (1992), Anderson (1994), Gupta and Rao (1994), Polycarpou (1996), Zbikowski and Hunt (1996)).

The neural network model can be easily extended to incorporate exogenous variables. Specifically, augment the vector $x_t$ into

$$x_t = (y_{t-1}, \ldots, y_{t-p}, u_{t,1}, \ldots, u_{t,q})^T.$$

(7)

With $u_{t,i} = u_{t-i}$ in (8), the model (7) reduces to the ARX($p, q$) model when $J = 0$. The parameters in (7) can be estimated by the method of least squares. The regression model is linear in the parameters $\beta_j$, but has $J(p + q + 1)$ nonlinear parameters $\alpha_j, a_j^T$. When $J(p + q + 1) \leq 10$, one can use Gauss-Newton or variable metric algorithms to compute the least squares estimate of the
vector of network parameters, which we shall denote by \( \theta \). For larger values of \( J(p + q + 1) \), these algorithms are computationally demanding. Rumelhart et al. (1986) proposed a much simpler procedure, called “backpropagation”, which is a recursive algorithm of the form

\[
\hat{\theta}_t = \hat{\theta}_{t-1} + \eta \{ y_t - f(x_t, \hat{\theta}_{t-1}) \} \nabla f(x_t, \hat{\theta}_{t-1}),
\]

where the positive constant \( \eta \) is called the “learning rate” and

\[
f(x, \theta) = \beta_0 + \sum_{j=1}^{J} \beta_j \psi(\alpha_j + a_j^T x).
\]

Here and in the sequel we use \( \nabla f \) to denote the vector of partial derivatives with respect to \( \theta \). Replacing \( \eta \) by \( \eta_t \) which approaches 0 as \( t \to \infty \), White (1989) made use of the asymptotic theory of stochastic approximation to prove convergence properties of the modified backpropagation algorithm under certain conditions. In practice, when the sample size is not very large and the number of parameters is not small relative to the sample size, it is more reliable to adopt a constant (instead of diminishing) learning rate and to use the batch-mode of backpropagation, which is an iterative (instead of recursive) scheme corresponding to steepest descent defined by

\[
\hat{\theta}^{(k)} = \hat{\theta}^{(k-1)} + \eta \sum_{t=1}^{n} \{ y_t - f(x_t, \hat{\theta}^{(k-1)}) \} \nabla f(x_t, \hat{\theta}^{(k-1)}).
\]

To avoid overfitting, a popular variant of the backpropagation algorithm is to include a penalty term \( \lambda C(\theta) \) in the steepest descent procedure to minimize \( \sum_{t=1}^{n} (y_t - f(x_t, \theta))^2 + \lambda C(\theta) \), where \( C(\theta) \) is the sum of squares (or sum of the absolute values) of the components of \( \theta \). The tuning parameters \( \lambda \) and \( \eta \) are critical to the performance of the algorithm, but in the absence of a comprehensive methodology for their determination, they are usually chosen in practice by extensive experimentation and subjective tinkering.

2.2 Stochastic Neural Networks

Fitting the TAR model (2) to time series data can be regarded as a “divide-and-conquer” procedure that divides the \( x \)-space into certain subregions and approximates the conditional expectation \( E(y_t | x_t) \) by a linear function in each subregion. The subregions are chosen to be of the form \( \{ r_{j-1} \leq y_{t-d} < r_j \} \) with the thresholds \( r_j \) to be estimated from the data. For parsimony of approximation, we shall augment the set of basis functions in the neural network model (5) to provide a more general model of the form

\[
y_t = \beta_0 + b_0^T x_t + \sum_{j=1}^{J} (\beta_j + b_j^T x_t) \psi(\alpha_j + a_j^T x_t) + \epsilon_t.
\]
Note that (8) reduces to the AR(p) model when J = 0 and that the TAR model with the indicator function replaced by the logistic function can also be expressed as a special case of (8). Fitting the neural network model (8) allows general hyperplanes (instead of those parallel to a fixed coordinate axis in TAR) as the thresholds. It is able to carry out the computational task that accompanies this much greater flexibility by using a smooth sigmoidal function to replace the indicator \( I(\alpha_j + a_j^T x_t \geq 0) \), so that gradient-type algorithms can be used to estimate \( \alpha_j \) and \( a_j \) by the method of least squares. Moreover, using \( \psi \) instead of the indicator function allows a somewhat softer split of the data, allowing the data to lie simultaneously in different regions but with different weights. In the context of regression analysis, Jordan and Jacobs (1994) proposed “soft” splits, via separating hyperplanes, of the data by using randomized weights generated from a multinomial distribution. This is called “hierarchical mixture of experts” (HME) in machine learning. Motivated by the advantages of soft splitting demonstrated in their work, we propose to replace \( \psi(\alpha_j + a_j^T x_t) \) in (8) by similar randomization via a Bernoulli random variable with mean \( \psi(\alpha_j + a_j^T x_t) \). In other words, we consider a “stochastic” neuron that is activated with probability \( \psi(\alpha_j + a_j^T x) \), instead of the McCulloch-Pitts neuron that is activated if and only if \( \alpha_j + a_j^T x \geq 0 \). This leads to the stochastic neural network (SNN) model

\[
y_t = \beta_0 + b_0^T x_t + \sum_{j=1}^{J} (\beta_j + b_j^T x_t) I_{ij} + \epsilon_t , \tag{9a}
\]

in which the \( \epsilon_t \) are i.i.d. zero-mean random variables and the \( I_{ij} \) are independent Bernoulli random variables that are independent of \( \{\epsilon_t\} \), such that

\[
P(I_{ij} = 1|x_t) = 1 - P(I_{ij} = 0|x_t) = \psi(\alpha_j + a_j^T x_t) . \tag{9b}
\]

Such SNN models have been introduced in artificial intelligence and neural computation; see Peretto (1984), Hertz, Krogh and Palmer (1991), Anderson and Titterington (1998) and Stroeve, Kappen and Gielen (1999).

Note that (9a) and (9b) completely specify the dynamics of \( y_t \) if the common distribution of the \( \epsilon_t \) is specified. As in HME, we assume the \( \epsilon_t \) to be normal with mean 0 and variance \( \sigma^2 \). The parameter vector of the model is

\[
\theta = (\beta_0, b_0^T, \ldots, \beta_J, b_J^T; \alpha_1, a_1^T, \ldots, \alpha_J, a_J^T; \sigma^2) , \tag{10}
\]

which can be estimated from the data by maximum likelihood. Since the \( I_{ij} \) are latent (unobservable) variables, we use the EM algorithm to compute the maximum likelihood estimate. Assuming
the \( \epsilon_t \) to be normal yields closed-form least squares estimates of the linear parameters \( \beta_j, b_j \) in the M-step of the algorithm. The M-step for the nonlinear parameters \( \alpha_j, a_j \) can be carried out via separate logistic regressions, one for each \( j \), as will be shown in Section 4. Since SNN involves independent Bernoulli random variables while HME involves multinomial vectors, it is computationally simpler than HME. The next two sections will show that the Gaussian assumption on the \( \epsilon_t \) is relatively innocuous when (9) is used as an approximation to the true underlying model, whose actual dynamics is unknown and may be very complex.

3. PROBABILISTIC PROPERTIES AND FORECASTING

To begin with, consider the case \( q = 0 \) in (7) so that \( x_t = (y_{t-1}, \ldots, y_{t-p})^T \). Then the SNN defined by (9a) and (9b) is a functional-coefficient autoregressive (FAR) model given by the stochastic difference equation

\[
x_{t+1} = B(I_t)x_t + (\beta_0 + \beta^T I_t + \epsilon_t, 0, \ldots, 0)^T,
\]

where \( \beta = (\beta_1, \ldots, \beta_j)^T, I_t = (I_{t1}, \ldots, I_{tJ})^T \) with \( E(I_{tj}|x_t) = \psi(\alpha_j + a_j^T x_t) \), and \( B(I_t) \) is the \( p \times p \) companion matrix whose first row is \( b_0^T + \sum_{j=1}^J I_{tj} b_j^T \) and whose \( i \)th row has all entries 0 except the \((i-1)\)th entry which takes the value 1 \((i = 2, \ldots, p)\). Therefore \( x_t \) is a Markov chain whose recurrence, stability and ergodicity properties can be analyzed by making use of products of random matrices (cf. Chen and Guo (1991)) and Lyapunov-type and Lipschitz-mixing methods (cf. Duflo (1997)). The details are given elsewhere. A simple, but stringent, sufficient condition for the geometric ergodicity of \( y_t \) can be derived from Theorem 1.2 of Chen and Tsay (1993a) for FAR models: \( y_t \) is geometrically ergodic if the common support of the \( \epsilon_t \) contains every open interval of the real line and if all the roots of the characteristic equation \( \lambda^p - c_1 \lambda^{p-1} - \cdots - c_p = 0 \) are inside the unit circle, where \( c_k \) is the maximum absolute value of the \( k \)th component of the vector \( b_0 + \sum_{j=1}^J I_{tj} b_j \) taken over all choices of \((I_1, \ldots, I_j)\) with \( I_j = 0 \) or 1.

In practice, the underlying dynamics may be much more complex and need not even be Markovian. The actual stochastic process requires specification of the conditional densities \( g_t(y_t|y_{t-1}, \ldots, y_1) \) for all \( t \geq 1 \), assuming the conditional distributions to be absolutely continuous with respect to Lebesgue measure. These unknown densities clearly cannot be estimated from finite amount of data. The \( p \)th order Markov model assumes that \( g_t(y_t|y_{t-1}, \ldots, y_1) = g(y_t|y_{t-1}, \ldots, y_{t-p}) \) for all \( t > p \), so we need only specify a function \( g \) of \( p + 1 \) variables. Although in principle \( g \) can be
estimated nonparametrically when the sample size is very large, in practice one encounters computational and sparse-data difficulties for $p$ that is not small relative to the sample size; typically $p$ is unknown and has to be determined from the data by some model selection procedure. The SNN approximates $g_t(y_t|y_{t-1},\ldots,y_1)$ by a flexible but tractable $p$th order Markov model that uses (9a) and (9b) to determine the conditional distribution of $y_t$ given $x_t = (y_{t-1},\ldots,y_{t-p})^T$. The hyperplane thresholds $\alpha_j + a_j^T x_t$ used to divide the $x_t$-space are soft in view of the Bernoulli randomization (9b). Because all $I_{tj}$ ($1 \leq t \leq n, 1 \leq j \leq J$) are unobservable, the time series data $y_t, 1 \leq t \leq n$, do not provide much information for estimating nonparametrically the common distribution of the $\epsilon_t$ in (9a) even if the $\alpha_j$ and $a_j$ in (9b) are known a priori, so the Gaussian assumption for $\epsilon_t$ is not overly restrictive.

When exogenous variables are present (i.e., $q \geq 1$), the underlying system is even more complex. Instead of the $p$th order Markov model to approximate the system, we now have the controlled Markov model $g_t(y_t|y_{t-1},\ldots,y_1, U_t) = g(y_t|y_{t-1},\ldots,y_{t-p}, u_{t,1},\ldots,u_{t,q})$, where $U_t$ denotes the vector of all exogenous variables up to time $t$ (measurable with respect to the $\sigma$-field $\mathcal{F}_{t-1}$ generated by $y_1,\ldots,y_{t-1}$ and external input variables up to time $t$) and $(u_{t,1},\ldots,u_{t,q})$ is a $q$-dimensional subvector of $U_t$. Defining $x_t$ by (8), the SNN model chooses a special form of $g$ given by (9a) and (9b).

For this controlled Markov model, the conditional expectation of $y_t$ given the past observations is

$$E(y_t|x_t) = \beta_0 + b_0^T x_t + \sum_{j=1}^J (\beta_j + b_j^T x_t)\psi(\alpha_j + a_j^T x_t),$$

(12)

since $E(I_{tj}|x_t) = \psi(\alpha_j + a_j^T x_t)$ by (9b). Hence the regression functions $E(y_t|x_t)$ for the SNN are the same as those for the corresponding neural network (8), and the universal approximation property (6) for neural networks extends to SNNs.

The conditional expectation (12) is the minimum variance one-step ahead predictor of $y_t$ given observations up to time $t-1$. For $m$-step ahead predictors with $m \geq 2$, there are no closed-form expressions like (12), and numerical integration or Monte Carlo methods are needed for their computation. First consider the case $m = 2$ and $q = 0$ (i.e., there are no exogenous variables). Let $b_j = (b_{j1},\ldots,b_{jp})^T$ and $a_j = (a_{j1},\ldots,a_{jp})^T$. Then

$$E(y_t|y_{t-2},\ldots,y_{t-p-1}) = \beta_0 + b_{01} E(y_{t-1}|y_{t-2},\ldots,y_{t-p-1}) + \sum_{k=2}^p b_{0k} y_{t-k}$$

$$+ \sum_{j=1}^J \left\{ (\beta_j + \sum_{k=2}^p b_{jk} y_{t-k} + b_{j1} y_{t-1})\psi(\alpha_j + \sum_{k=2}^p a_{jk} y_{t-k} + a_{j1} y_{t-1}) | y_{t-2},\ldots,y_{t-p-1} \right\}.$$
Although \(E(y_{t-1}|y_{t-2}, \ldots, y_{t-p-1})\) is easily obtained from (12), the above expression also involves \(E[\Psi(y_{t-1}; y_{t-2}, \ldots, y_{t-p})|y_{t-2}, \ldots, y_{t-p-1}]\), where \(\Psi\) is a nonlinear function. We therefore need to evaluate first the conditional distribution of \(y_{t-1}\) given \(y_{t-2}, \ldots, y_{t-p+1}\) and then the conditional expectation by integration. More generally, we need to evaluate expectations with respect to the conditional distribution of \(y_{t-m+i}\) given \(y_{t-m}, \ldots, y_{t-p-m+1}\) for \(i = 1, \ldots, m - 1\). This can best be done by Monte Carlo simulation, generating \(y_{t-m+1}\) and then \(y_{t-m+2}, \ldots\), etc., from (9a) and (9b).

When exogenous variables are present, the probability law of future exogenous variables can also be incorporated to generate them. The desired \(m\)-step ahead predictor can then be computed as an average of the simulated values. Tresp and Hofmann (1998) discuss how to compute not only the predictor but also its standard deviation ("error bar") from Monte Carlo simulations.

4. ESTIMATION USING EM

In the next section we consider the problem of model selection in which the number \(J\) of hidden units and the parameters for each hidden unit (setting the other parameters to be 0) are determined from the data. This section studies the problem of estimating the chosen parameters for the hidden units. Accordingly for the \(j\)th hidden unit, let \(A_j\) denote the subvector of \((\alpha_j, a_{j1}, \ldots, a_{jp})^T\) and \(B_j\) the subvector of \((\beta_j, b_{j1}, \ldots, b_{jp})^T\) whose parameters are to be estimated. Let \(v_{jt}\) be the subvector of \((1, x_t^T)^T\) for which \(A_j^T v_{jt} = \alpha_j + a_j^T x_t\), and \(w_{jt}\) be the subvector of \((1, x_t^T)^T\) for which \(B_j^T w_{jt} = \beta_j + b_j^T x_t\). Thus (9a) can be rewritten as \(y_t = B_0^T w_{0t} + \sum_{j=1}^J I_{tj} B_j^T w_{jt} + \varepsilon_t\). Instead of (10), we now let \(\theta = (B_0^T, \ldots, B_j^T, A_1^T, \ldots, A_j^T, \sigma^2)\).

4.1 Maximum Likelihood via EM

Although the likelihood function can be written down explicitly (see (15) below), it is more convenient to compute the MLE via the EM algorithm, with the complete data consisting of \(y_t, x_t\) and the unobservable \(I_{jt}\). The complete-data log-likelihood is given by

\[
\ell_c(\theta) = \sum_{j=1}^J h_n(A_j) - \sum_{t=1}^n \left( y_t - B_0^T w_{0t} - \sum_{j=1}^J I_{tj} B_j^T w_{jt} \right)^2 / (2\sigma^2) - (n/2) \log(2\pi\sigma^2),
\]

where \(h_n(A_j) = \sum_{t=1}^n \{ I_{tj} \log \psi(A_j^T v_{jt}) + (1 - I_{tj}) \log(1 - \psi(A_j^T v_{jt})) \} \). For the E-step of the EM algorithm, note that

\[
P\{(I_{t1}, \ldots, I_{tJ}) = 1|y_t, x_t\} = g_\theta(y_t, x_t, 1) / f_\theta(y_t, x_t),
\]

\[
E(I_{tj}|y_t, x_t) = \sum_{1,I_{tj}=1} g_\theta(y_t, x_t, 1) / f_\theta(y_t, x_t),
\]

\[
E(I_{tj}|y_t, x_t) = \sum_{1,I_{tj}=1} g_\theta(y_t, x_t, 1) / f_\theta(y_t, x_t),
\]
where \( I = (I_1, \ldots, I_J) \) with \( I_j = 0 \) or \( 1 \), \( f_\theta(y, x) = \Sigma_I g_\theta(y, x, I) \) and

\[
g_\theta(y_t, x_t, I) = \sigma^{-1} \phi \left( \left( y_t - B_0^T w_{0t} - \sum_{j=1}^{J} I_j B_j^T w_{jt} \right) / \sigma \right) \prod_{j=1}^{J} \left\{ \psi(A_j^T v_j) \right\}^{I_j} \left( 1 - \psi(A_j^T v_j) \right)^{1-I_j},
\]

in which \( \phi \) is the standard normal density function. Since \( f_\theta \) is the conditional density function of \( y_t \) given \( \{y_{t-1}, y_{t-2}, \ldots\} \cup \{u_{s,i} \colon s \leq t, 1 \leq i \leq q\} \), the observed-data log-likelihood is

\[
\ell_o(\theta) = \sum_{t=1}^{n} \log f_\theta(y_t, x_t).
\]  

(15)

Although (15) is not used for direct computation of the MLE, it will be used for statistical inference and for model selection. From (14a) it follows that

\[
\sum_{t=1}^{n} E \left\{ \left( y_t - B_0^T w_{0t} - \sum_{j=1}^{J} I_j B_j^T w_{jt} \right)^2 \mid y_t, x_t \right\} = \sum_{t=1}^{n} \sum_{I} \left( y_t - B_0^T w_{0t} - \sum_{j=1}^{J} I_j B_j^T w_{jt} \right)^2 g_\theta(y_t, x_t, I) / f_\theta(y_t, x_t).
\]  

(16)

The E-step of the EM algorithm can be carried out by applying (14) and (16) to take conditional expectation of (13) given the observed data, with the unknown \( \theta \) replaced by its estimate \( \theta^{(k)} \) at the \( k \)th iteration. The M-step of the algorithm updates \( \theta^{(k)} \) by maximizing \( E_{\theta^{(k)}} \{ \ell_c(\theta) | y_1, x_1, \ldots, y_n, x_n \} \). This consists of determining \( (B_0^T, \ldots, B_J^T) \) by weighted least squares (in view of (16)) and \( \sigma^2 \) by the residual sum by squares in the right hand side of (16) divided by \( n \) (in view of (13)), and of separate iteratively reweighted least squares algorithms to maximize \( E_{\theta^{(k)}} \{ h_n(A_j) | y_1, x_1, \ldots, y_n, x_n \} \) (which amounts to logistic regression with \( E(I_{tj} | y_t, x_t) \) as the dependent variable), one for each \( j \). Therefore the M-step is decoupled into \( J \) separate logistic regressions and a weighted linear least squares regression. This decoupling makes the EM algorithm much more tractable than direct maximization of the nonlinear function (15) over the full vector \( \theta \). Convergence of the EM algorithm to the MLE follows from standard theory (cf. Wu (1983), Redner and Walker (1984)). Similar decoupling has been noted in the ME (mixture of experts) context by Jacobs et al. (1991), where "the weights in one expert are decoupled from the weights in other experts."

4.2 Asymptotic Theory

Since the observed-data log-likelihood (15) is a smooth function of \( \theta \), the usual asymptotic theory for the MLE holds and can be applied to compute standard errors and construct confidence intervals. In particular, \( \hat{\theta} \) is consistent and asymptotically efficient, and \( (-\ell_o(\hat{\theta}))^{1/2} (\hat{\theta} - \theta) \) has a
limiting standard normal distribution under a stationary SNN model, where \( \hat{\ell}_o \) denotes the matrix of second partial derivatives of (15); see the Appendix for a precise statement of the result and its proof.

As pointed out in Section 3, the true underlying model may be much more complex and the SNN is a tractable approximation to it. In this case, under certain regularity conditions, \( n^{-1} \ell_o(\theta) \) converges to some smooth function \( \lambda(\theta) \) and \( \hat{\theta} \) converges to the maximizer \( \theta^* \) of \( \lambda(\theta) \) as \( n \to \infty \), with probability 1 under the true underlying model. Moreover, \( \hat{\theta} - \theta^* \) still has an asymptotically normal distribution with mean vector 0. Thus the SNN with parameter \( \theta^* \) is the best approximating SNN (in the sense of minimizing the Kullback-Leibler divergence) to the true underlying model and \( \hat{\theta} \) is a consistent estimate of \( \theta^* \). These results, therefore, are extensions of those of White (1981) concerning least squares estimates, based on i.i.d. vectors of observed responses and covariates, of the parameters of misspecified nonlinear regression models. The details are given in the Appendix.

### 4.3 Generalized Residuals and Discussion

The linear parameters \( \mathbf{B}_j \) of the SNN are estimated by linear least squares in the EM algorithm. The soft threshold for the \( j \)th hidden unit is determined by the nonlinear parameter vector \( \mathbf{A}_j \) and is estimated via logistic regression in the EM algorithm. The normality assumption on the \( \epsilon_t \) is only used explicitly in the conditional distribution of the unobservable Bernoulli variables \( I_{t1}, \ldots, I_{tJ} \) given the observed data for each E-step. If we regard the estimated SNN as an approximation to the unknown underlying model (which may be much more complicated than SNNs), the iterative determination of \( \mathbf{A}_j \) and \( \mathbf{B}_j \) in the EM algorithm can be interpreted as successive approximations, with \( \mathbf{B}_j \) given by least squares after each choice of the hyperplane thresholds \( \mathbf{A}_j^T \mathbf{v}_{jt} \). In practice, these soft thresholds need only be determined roughly and the iteratively reweighted least squares algorithm to determine \( \mathbf{A}_j \) within each M-step can be terminated after a few iterations.

Although the SNN defined by (9a) and (9b) has the same regression function as that of the neural network model (8), the residuals \( y_t = \{ \hat{\beta}_0 + \hat{\mathbf{b}}_0^T \mathbf{x}_t + \sum_{j=1}^J (\hat{\beta}_j + \hat{\mathbf{b}}_j^T \mathbf{x}_j) \psi(\hat{\alpha}_j + \hat{\mathbf{a}}_j^T \mathbf{x}_t) \} \) for (8), providing estimates of the unobservable \( \epsilon_t \), are not applicable to the SNN which involves additional unobservable Bernoulli random variables \( I_{tj} \) with means \( \psi(\alpha_j + a_j^T \mathbf{x}_t) \). For the SNN, we consider the following generalized residuals in the sense of Cox and Snell (1968):

\[
\begin{align*}
    r_t &= y_t - \left\{ \hat{\beta}_0 + \hat{\mathbf{b}}_0^T \mathbf{x}_t + \sum_{j=1}^J (\hat{\beta}_j + \hat{\mathbf{b}}_j^T \mathbf{x}_j) E_{\hat{\theta}}(I_{tj}|y_t, \mathbf{x}_t) \right\} \\
    &= \sum_{I} \left\{ y_t - \hat{\beta}_0 - \hat{\mathbf{b}}_0^T \mathbf{x}_t - \sum_{j=1}^J (\hat{\beta}_j + \hat{\mathbf{b}}_j^T \mathbf{x}_j I_j) \right\} g_{\hat{\theta}}(y_t, \mathbf{x}_t, I)/f_{\hat{\theta}}(y_t, \mathbf{x}_t),
\end{align*}
\]
which is a weighted sum of linear regression residuals \( y_t - \hat{\beta}_0 - \hat{\beta}_0^T x_t - \sum_{j=1}^J (\hat{\beta}_j + \hat{\beta}_j^T x_j I_j) \) over the \(2^J\) choices of \( I \) with \( I_j = 0 \) or 1. Plots of these generalized residuals versus time and additional covariates (such as higher-order lagged variables) can be used for diagnostics of the SNN model.

5. MODEL SELECTION

The SNN model (9a) and (9b) has \((J + 1)(p + q + 1)\) linear parameters \(\beta_0, b_0^T, \ldots, \beta_J, b_J^T\), a variance parameter \(\sigma^2\), and \(J(p + q + 1)\) nonlinear parameters \(\alpha_j, a_j^T(j = 1, \ldots, J)\). In view of this potentially large number of parameters that may lead to overfitting, it is desirable to choose \(J\) and the relevant parameters suitably (setting the other parameters to be 0). Because of the relative ease in computing the MLE \(\hat{\theta}\) via the EM algorithm and the log-likelihoods \(\ell_\theta(\hat{\theta})\) via (15), we can apply standard model selection criteria such as the BIC to do this.

We describe here an efficient computational scheme to perform model selection for SNNs. The scheme consists of forward selection of variables for each candidate number \(J\) of hidden units, followed by choice of \(J\) using the BIC and then backward elimination of parameters for the chosen \(J\). Recall that \(\text{BIC} = -2\ell_\theta(\hat{\theta}) + \kappa \log n\), where \(\kappa\) denotes the number of components of the parameter vector \(\theta\). Let \(x_{it}(i = 1, \ldots, p + q)\) denote the \(i\)th component of the vector \(x_t\), or equivalently, the \(i\)th input variable at time \(t\). The forward selection procedure chooses for each \(J \in \{1, 2, \ldots, J_{\max}\}\), where \(J_{\max}\) is some prescribed upper bound for \(J\), the variables to be included in the SNN model with \(J\) hidden units. It enters one variable at a time, choosing from the set of variables not already entered the one that gives the largest observed log-likelihood (15). It stops entering variables as soon as the BIC does not decrease when an additional variable is included. The forward selection procedure thereby associates with each \(J\) a set of selected variables and the corresponding BIC value, denoted by \(\text{BIC}(J)\). It then chooses the smallest \(J^*\) that attains \(\min_{1 \leq J \leq J_{\max}} \text{BIC}(J)\).

The backward elimination procedure eliminates parameters from the SNN model with \(J^*\) hidden units. Starting with the set of estimated parameters at the conclusion of the forward selection procedure, it eliminates the parameters sequentially as follows. Note that the asymptotic covariance matrix of the estimate of the parameter vector \(\theta\) of an SNN model can be estimated by \((-\hat{\ell}_\theta(\hat{\theta}))^{-1}\), from which we obtain an estimated standard error (s.e.) of each parameter estimate. At the beginning of each elimination step, we let \(\hat{\theta}\) be the vector of parameters not yet eliminated and choose the parameter \(\theta_i\) with the smallest \(|\hat{\theta}_i|\). If the BIC can be reduced by removing \(\theta_i\), eliminate \(\theta_i\) from \(\theta\) and go to the next step. On the other hand, if the BIC shows no decrease by
removing $\theta_t$, the elimination procedure stops with the estimated parameter vector $\hat{\theta}$ for the SNN with $J^*$ hidden units.

6. NUMERICAL EXAMPLES

6.1 Piecewise Linear Autoregression

Consider the piecewise linear autoregressive model

$$y_t = \begin{cases} 
1 + 0.7y_{t-1} + 0.05y_{t-2} + \eta_t & \text{if } y_{t-2} < (y_{t-1} + y_{t-3})/2, \\
0.8y_{t-1} + \eta_t & \text{otherwise}, 
\end{cases}$$

where the $\eta_t$ are i.i.d. standard normal random variables. The minimum variance one-step ahead predictor of $y_t$ is given by $E[y_t|y_{t-1}, y_{t-2}, y_{t-3}]$, which is equal to the piecewise linear function

$$f(y_{t-1}, y_{t-2}, y_{t-3}) = (1 + 0.7y_{t-1} + 0.05y_{t-2})I(2y_{t-2} < y_{t-1} + y_{t-3}) + 0.8y_{t-1}I(2y_{t-2} \geq y_{t-1} + y_{t-3}).$$

Letting $J_{\text{max}} = 2$, we applied the estimation and model selection procedures described in Sections 4 and 5 to fit an SNN model to $n = 100$ successive observations generated from this time series model, with $x_t = (y_{t-1}, y_{t-2}, y_{t-3})$ in (9a) and (9b). The initial observation $y_1$ is generated from an approximating stationary distribution that is obtained by first generating 1000 additional observations $y_0, y_{-1}, \ldots, y_{-99}$. For comparison, we also used the neural network training algorithm of Venables and Ripley (1994) to fit to the same data neural networks NN1 and NN2, with 1 and 2 hidden units respectively. In addition, we used Friedman's (1991) MARS algorithm to estimate the regression (18) nonparametrically (i.e., without assuming the actual functional form). This amounts to fitting the ASTAR model (3) of Lewis and Stevens (1991) to the same data. Another regression method we used to estimate (18) nonparametrically is projection pursuit regression (PPR), introduced by Friedman and Steutzle (1981), which consists of finding one-dimensional projections $a_j^T x_t$ and using the supersmoother or other smoothing methods to estimate the functions $h_j$ in fitting the regression model $y_t = \sum_{j=1}^J h_j(a_j^T x_t) + \epsilon_t$. A comparative study of these methods is given in Table 1.

6.2 Markov Chain

Let $\{y_t\}$ be a Markov chain with state space $[0,1]$ and transition probability function

$$p(y|x) = \begin{cases} 
e^{1-y} & \text{if } 0 \leq y \leq x, \\
e^{1-y} - e^{y-x} & \text{if } x \leq y \leq 1. 
\end{cases}$$

(19)
The Markov chain has a nonlinear regression function

\[ E(y_t | y_{t-1}) = y_{t-1} - 1 + \frac{1}{2} \exp(1 - y_{t-1}), \]  

(20)

and the \( \eta_t \overset{\Delta}{=} y_t - E(y_t | y_{t-1}) \) are not i.i.d. and do not have normal marginal distributions, although they form a martingale difference sequence. The chain has a linear stationary density function \( \pi(y) = 2(1 - y), 0 \leq y \leq 1 \). Since \( x_t = y_{t-1} \) is one-dimensional, the nonparametric regression methods MARS and PPR reduce to regression splines and smoothers, and we can take \( a_j = 1 \) in the NN and SNN models (7) and (9). Letting \( J_{\max} = 2 \), we fitted an SNN model to \( n = 300 \) successive observations from the Markov chain (19), with \( y_1 \) generated from the stationary distribution. We also fitted to the same data MARS, PPR, NN1 and NN2 for comparison.

The fitted models in Examples 6.1 and 6.2 can be used to provide one-step ahead forecasts \( \hat{f}_n(x_t) \) of \( y_t \) for \( t \geq n + 1 \). In particular, for the SNN and NN models, \( \hat{f}_n \) is given by (12) with the unknown parameters replaced by their estimates based on \( y_1, \ldots, y_n \). Note that for \( t \geq n + 1 \),

\[ E[y_t - \hat{f}_n(x_t)]^2 = E\eta_t^2 + E[\hat{f}_n(x_t) - f(x_t)]^2. \]

(21)

Accordingly \( y_{n+1}, \ldots, y_{n+k} \) were also generated from each of the models (17) and (19), and Table 1 gives the median and mean, over 50 simulation runs, with the standard error (s.e.) in parentheses, of the average squared error \( k^{-1} \sum_{t=n+1}^{n+k} E[\hat{f}_n(x_t) - f(x_t)]^2 \) for each of the five ways (SNN, NN2, NN1, MARS, PPR) to fit \( \hat{f}_n \). In view of (21), the results in Table 1 show that SNN and NN1 give the best one-step ahead forecasts among its competitors. For SNN, the selected \( J \) is 1 in all 50 simulations of Example 6.1, for which the number of selected parameters ranges from 4 to 12 in the 50 simulations, with an average of 6.7. In Example 6.2, the selected \( J \) is again 1 and all 6 parameters of the full model are selected by the model selection procedure for SNN in each of the 50 simulations.

Insert Table 1 about here

6.3 Nonlinear Transformation of Autoregressive Model

Consider the stationary Markov chain

\[ y_t = x_{t-1}^2 + x_t = 0.99 x_{t-1} + \epsilon_t, \]

(22)

in which the \( \epsilon_t \) are i.i.d. \( N(0, \sigma^2) \) and \( x_0 \) has the stationary distribution that is \( N(0, \sigma^2/[1 - (.99)^2]) \). The time series \( \{y_t\} \) is called the distorted long-memory autoregressive model (Kantz and Schreiber, 1997). In the following we take \( \sigma = 0.5 \) and assume that the system dynamics is unknown to the
observer of \{y_t\}. Figure 1 gives the plot of a simulated \{y_t\} series generated from this model, of which the sudden bursts in the figure are typical features. Note that

\[ E\hat{y}_t^2 = (.99)^2 E\hat{x}_t^8 + \sigma^2 E\hat{x}_t^4 = 29267.29. \quad (23) \]

We first consider the case where the \(x_t\) series is observed with a delay of 8 lags while \(y_t\) is observed without delay, i.e., \((y_t, x_{t-8})\) is observed at time \(t\). To compare the performance of SNN with that of NN in multistep postsample forecasts, we generated 50 sets of 2010 observations from this model. For each set, the first \(n = 2000\) observations were used to fit the model, and the fitted model was then used to compute \(\ell\)-step ahead forecasts \(\hat{y}_{2000+\ell}\) for \(\ell = 1, \ldots, 10\). Specifically we took \(x_t = (y_{t-1}, \ldots, y_{t-7}, x_{t-8})^T\) and \(J_{\text{max}} = 6\) to fit an SNN model. We also used the neural network training algorithm in Venables and Ripley (1994) to fit a neural network (NN) with 6 hidden units. For \(\ell \geq 2\), \(\hat{y}_{2000+\ell}\) was computed by the Monte Carlo method involving 6000 simulations; see the last paragraph of Section 3. For \(\ell = 1\), \(\hat{y}_{2000+\ell}\) was computed via (12). The median of the squared prediction error \((y_{2000+\ell} - \hat{y}_{2000+\ell})^2\) for the 50 simulated data sets based on SNN is compared with that based on NN in the left half of Table 2. Also given in left half of Table 2 is the median, over the 50 simulation runs, of the in-sample average squared error, defined by \(n^{-1} \sum_{t=1}^n (y_t - f(x_t, \hat{\theta}))^2\) in the case of NN (using the notation of Section 2.1) and by (16) divided by \(n\) in the case of SNN. These results show that the fitted SNN model provides better in-sample errors and postsample forecasts than the fitted NN model. The SNN model selection procedure picked 5 hidden units for six data sets and 6 hidden units for forty-four data sets, and the number of parameters selected ranged from 99 to 117, with an average of 114.8.

We next consider the case where the \(x_t\) series cannot be observed and only the \(y_t\) series is observable. We now use \(x_t = (y_{t-1}, \ldots, y_{t-7})^T\) to fit an SNN model with \(J_{\text{max}} = 5\) to a longer series with \(n = 4000\) observations. The results are compared with those of fitting a neural network model with 5 hidden units. Again we generated 50 sets of 4010 observations and computed the postsample forecast error \(y_{4000+\ell} - \hat{y}_{4000+\ell}\) for \(\ell = 1, \ldots, 10\). The SNN model selection procedure picked \(J = 4\) in two data sets and \(J = 5\) in the remaining forty-eight data sets, and the number of parameters selected ranged from 72 to 88, with an average of 87.4. The right half of Table 2 gives the median values, together with the mean values and standard errors (s.e., in parentheses), of \((y_{4000+\ell} - \hat{y}_{4000+\ell})^2\) with \(\ell \geq 1\) and the in-sample average squared error \((\ell = 0)\) for the 50 simulated

\[ 15 \]
data sets. Note that the mean value is much larger than the median value in each case because of large outliers when there are sudden bursts as in Figure 1. Note also that the mean values for SNN are considerably smaller than $E(y_{4000+\ell}^2)$ given by (23), but that there are a few cases in which the mean values for NN are larger than (23), implying that the neural network $\ell$-step ahead forecast may perform worse, for $\ell \geq 6$, than the simple forecast that uses the process mean $E y_t = 0$. Figure 2 summarizes the simulation results in this case ($n = 4000$ and $\{x_t\}$ unobservable) by a boxplot of \( \log\{\text{Error (NN)}/\text{Error (SNN)}\} \), where “Error” refers to $(y_{4000+\ell} - \hat{y}_{4000+\ell})^2$ in the case $\ell \geq 1$ and to the in-sample average squared error in the case $\ell = 0$.

6.4 Sunspot Numbers

Wolf’s sunspot numbers constitute one of the benchmark data sets in nonlinear time series. Weigend, Rumelhart and Huberman (1991) used a feedforward neural network (consisting of 8 hidden units) with weight elimination (NN$_{we}$) to fit the data set that Tong and Lim (1980) had fitted with a TAR model with 12 lags, which Weigend et al. also used for their neural network model. In order to compare with their results, we need to split the data into 3 periods. Period A is from 1700 to 1920; period B is from 1921 to 1955; and period C is from 1956 to 1979. The data in period A are used as a training set from which the model parameters are estimated. The fitted model (on the basis of data in period A) is then used to produce one-step ahead forecasts $\hat{Y}_t$ for both periods B and C. Weigend et al. (1991) used the “average relative variance”

$$\text{arv}(S) = \frac{\sum_{t \in S}(Y_t - \hat{Y}_t)^2}{\#(S)\bar{\sigma}^2}$$

to compare the performance of NN$_{we}$ with that of TAR in modeling sunspot numbers, where $\bar{\sigma}^2 = 1535$ is the variance of the sunspot numbers $Y_t$ and $\#(S)$ denotes the number of observations in $S$. For $t \in A$, $Y_t - \hat{Y}_t$ is the residual in fitting a TAR or NN model to the data in period A. If we fit an SNN model to the data in period A, then $\sum_{t \in A}(Y_t - \hat{Y}_t)^2$ is given by (16) in which the unknown parameters are replaced by their estimates, since generalized residuals have to be used to incorporate the unobservable random variation in the Bernoulli variables $I_{ij}$. Using the SNN modeling procedure in Sections 4 and 5 with $J_{\text{max}} = 2$ and $x_t = (Y_{t-1}, \ldots, Y_{t-8})$ yield arv(A)=0.063, in comparison with the arv(A) values of 0.082 for NN$_{we}$ and 0.097 for TAR; we use 8 lags following the choice of Chen and Tsay (1993). The SNN model selection procedure chooses 1 hidden unit and 16 parameters for these data. The arv(B) values for the three methods are as follows:

Insert Table 2 and Figure 2 about here

16
Thus SNN provides the best one-step ahead forecasts for period B. This is also true for period C, whose arv(C) values are listed below:

\[\text{SNN: 0.27, } \text{NN_{wC}: 0.35, } \text{TAR: 0.28}\]

While Weigend et al. (1991) used the "in-sample" residuals and "out-sample" one-step ahead prediction errors to measure performance, Tong (1990) and Chen and Tsay (1993a) considered multi-step postsample forecasts to assess how well the TAR and FAR models predict sunspot numbers. They both used the data from 1700 to 1979 as the training sample and computed multistep ahead forecasts of the observations in the period 1980-1987. Moreover, instead of \(Y_t\), they used the square root transformed series \(y_t = 2(\sqrt{1+Y_t} - 1)\) to fit either the TAR or FAR model with normal errors \(\epsilon_t\). Using \(J = 1\) hidden unit and \(x_t = (y_{t-1}, \ldots, y_{t-8})\), we applied the estimation and model selection procedures in Sections 4 and 5 to fit an SNN model to the square root transformed series from 1700 to 1979, yielding

\[y_t = 1.75 + 0.96y_{t-1} - 0.13y_{t-3} - 0.11y_{t-5} + I_t(0.42y_{t-1} - 0.71y_{t-2} + 0.43y_{t-3} + 0.29y_{t-5}) + \epsilon_t\]

where the \(\epsilon_t\) are i.i.d. normal with mean zero and variance 2.36, and \(I_t\) are i.i.d. Bernoulli with mean \(\psi(-0.59y_{t-3} + 0.32y_{t-8})\).

By using the procedure in Section 3, \(m\)-step ahead forecasts for the period 1980-1987 were computed from the fitted SNN model. In particular, for \(m \geq 2\), Monte Carlo simulation with 60,000 simulation runs was used. The results are listed in Table 2 which also gives the results reported by Chen and Tsay (1993a) for comparison. As noted by Chen and Tsay, FAR outperforms TAR in 5 out of 8 postsample periods. Table 2 shows that SNN not only outperforms TAR in 5 out of 8 periods but also outperforms FAR in 5 out of 8 periods. Moreover, the largest prediction error is considerably smaller than that of FAR or TAR.

Insert Table 3 about here

6.5 Influenza Mortality Data

The time series of monthly mortality rates (expressed as deaths per 100,000) due to influenza and pneumonia from 1968 to 1978 reported by Shumway and Katzoff (1992) is shown in Figure 1. The series can be downloaded from the website \texttt{www.stat.pitt.edu/~stofer/tsa.html} under the item
flu.dat. There are 132 observations, the first 108 of which are used as training data and are on the left hand side of the vertical line in Figure 3.

Figure 3 shows a strong annual cycle. A preliminary study of the autocorrelation function and the partial autocorrelation function of the data leads us to fit the following seasonal autoregressive model:

\[(1 - \theta_1 B - \theta_2 B^2)(1 - \theta_{12} B^{12})(y_t - \mu) = \epsilon_t,\]  \hspace{1cm} (24)

where the \(y_t\) are the mortality rates, the \(\epsilon_t\) are i.i.d. normal with mean 0 and variance \(\sigma^2\), and \(B\) is the back-shift operator. The maximum likelihood estimates of the parameters are

\[\hat{\theta}_1 = 0.677, \hat{\theta}_2 = -0.372, \hat{\theta}_{12} = 0.683, \hat{\mu} = 0.310, \hat{\sigma}^2 = 0.0055.\]

The estimates can be used to replace the unknown parameters in (24) so that setting \(\epsilon_t = 0\) in (24) gives the one-step ahead forecasts for \(t \geq 15\). Figure 4 plots these AR-based forecasts (solid lines) and the original data (dots). On the left side of the vertical line these represent the fitted values of the AR model, and they represent the postsample one-step ahead forecasts at and on the right side of the vertical line. The figure shows that the AR model captures the general cyclical pattern.

Using the SNN estimation procedure in Section 4 with \(J = 1\) hidden unit and \(x_t = (y_{t-1}, \ldots, y_{t-12})\), we fitted the following SNN model:

\[y_t = 0.555 + 0.33y_{t-1} - 0.6y_{t-2} - 0.587y_{t-3} + 0.417y_{t-12} + I_t(-0.506 + 0.399y_{t-1} + 0.547y_{t-2} + 0.493y_{t-3} - 0.188y_{t-12}) + \epsilon_t,\]

in which the \(\epsilon_t\) are i.i.d. normal with mean 0 and variance 0.00098 and the \(I_t\) are independent Bernoulli with mean \(\psi(-28.01 - 984.53y_{t-1} + 168.71y_{t-2} + 1251.72y_{t-3}).\) Figure 5 plots the SNN based one-step ahead forecasts (solid lines) and the original data (dots). Comparison of Figure 5 with Figure 4 shows that the SNN model gives better in-sample fits and postsample forecasts than the AR model.

We also used NN1 (neural network with 1 hidden unit), MARS and PPR to estimate nonparametrically from the training data the regression function of \(y_t\) on \(x_t\). The cumulative postsample
absolute prediction errors for AR, SNN, NN1, MARS and PPR are plotted for \(109 \leq t \leq 132\) in Figure 6. The figure shows that SNN has the smallest cumulative postsample absolute prediction errors.

Insert Figure 6 about here

7. OTHER APPROACHES AND CONCLUDING REMARKS

Motivated by the HME model of Jordan and Jacobs (1994), we have introduced in this article the SNN time series model that requires much lower computational complexity for parameter estimation than single-layer neural networks but shares the universal approximation property of the latter. We have also developed a BIC-based computational scheme to perform model selection for SNNs. Its superior postsample forecast performance over neural networks shown in Section 6.3 is perhaps due to the much more thorough model selection procedure for SNNs than that constrained by the computational task in fitting neural networks. Moreover, the software available to fit neural networks typically assumes (5) instead of our more general form (8) that involves more parameters, for which an elaborate variable selection scheme is needed to avoid overfitting. Because it provides better bias-variance trade-off than purely nonparametric models like PPR, MARS and FAR, the SNN has better postsample forecast performance as illustrated in Section 6.

For an SNN, each E-step of the estimation procedure in Section 4.1 involves summation over \(I\) in (14) and (16). There are \(2^J\) tuples \((I_1, \ldots, I_J)\) with \(I_j = 0\) or 1. Hence the computational complexity grows exponentially with \(J\). On the other hand, in fitting SNN models to typical time series data, \(J\) can be chosen relatively small and the EM algorithm indeed provides an efficient method to compute the MLE of the parameters of the model. Note in this connection that even for fitting the simpler TAR model (2), the number \(J\) of thresholds is often limited to 2 (with a single threshold \(r_1\)), and there are computational difficulties when \(J\) is large. Although the backpropagation algorithm can be used to train neural network models with hundreds of hidden units, it may produce poor estimates, especially when \(J\) is large; see e.g. Baum and Lang (1991) who describe the famous two-spiral example illustrating the failure of backpropagation. Monte Carlo optimization methods such as simulated annealing are more reliable alternatives to backpropagation. Wong and Liang (1997) recently developed a dynamic importance weighting method for Monte Carlo optimization and showed that it works well for training neural networks in the two-spiral example. Markov Chain Monte Carlo (MCMC) methods have also been used in Bayesian learning of neural networks; see
Neal (1996), de Freitas et al. (2000) and the references therein.

For SNN models with more than 10 hidden units, we can use MCMC methods to train the network model. A variant of the SNN is the belief network or Boltzmann machine, in which the output and input variables are both binary. Even though the binary nature of all the observed and latent variables simplifies the arithmetic, the exponential growth in computational complexity with the number of hidden units makes direct computation "infeasible for networks of significant size", as noted by Neal (1992) who used Gibbs sampling to implement maximum likelihood estimation in such networks.

The use of stochastic units in neural network models also enables one to use the so-called "mean field approximation" in statistical mechanics; see Section 2.4 of Hertz, Krogh and Palmer (1991). Saul, Jaakola and Jordan (1996) have developed a mean field theory for belief networks, providing tractable approximations to the probabilities of state vectors and giving a tractable lower bound on the likelihood of instantiated patterns. This yields a practical alternative to Gibbs sampling, which they find to be "impractically slow". Section V.G of Anderson and Titterington (1999) reviews recent work on mean field approximations and their statistical properties.

On the other hand, Gibbs sampling and other MCMC methods are undergoing many new developments. Of particular relevance to SNN modeling of nonlinear time series is sequential importance sampling (SIS), developed by Liu and Chen (1998) and Liu, Chen and Wong (1998) for real-time Monte Carlo computation in dynamic systems. SIS has three ingredients: importance sampling and resampling, rejection sampling, and Markov chain iterations. When Monte Carlo methods such as SIS are applied to SNN modeling, it is more convenient to use a Bayesian approach than the likelihood approach, averaging over the simulated posterior distribution instead of maximizing the simulated likelihood function. The resampling and the rejection control components of SIS reject samples with small weights at an early stage and properly adjust the weights for the remaining samples, thus providing an effective way to construct a good importance sampling distribution. This is particularly attractive for training large networks, for which the parameter space is high-dimensional and the posterior distribution of the network can often be adequately approximated by one that assigns most of its mass to a lower-dimensional subset of the parameter space. Applications of SIS to Bayesian modeling of SNN will be presented elsewhere.
APPENDIX: CONSISTENCY AND ASYMPTOTIC NORMALITY

The two theorems below give sufficient conditions for the consistency and asymptotic normality results mentioned in Section 4.2. It will be assumed throughout the sequel that the parameter space is compact. This compactness assumption is actually made implicitly in the practical implementation of the iterative estimation procedure since overly large values of the estimates and also overly small values of \( \hat{\sigma}^2 \) are typically truncated to avoid numerical difficulties in carrying out the iterative procedure. The first theorem assumes that the true underlying model is indeed an SNN model (9) in which \( \{(x_t, \epsilon_t), t \geq 1\} \) is a stationary ergodic sequence with \( \epsilon_t \) being i.i.d. \( N(0, \sigma^2) \) random variables. In the case where there are exogenous inputs \( u_{t,j} \) in (8), this requires \( \{(u_{t,1}, \ldots, u_{t,q}), t \geq 1\} \) to be stationary and ergodic. When exogenous inputs are absent so that the SNN reduces to a FAR model, this requires the Markov chain (11) to be ergodic. Although the chain has to be initialized at the stationary distribution for it to be a stationary sequence, such initialization is actually not needed for the strong law and central limit theorem used to derive the result. The theorem is proved by applying martingale theory and modifying Hansen’s (1982) arguments to establish consistency and asymptotic normality of GMM estimators based on stationary ergodic observations. To avoid confusion we denote the true parameter value by \( \theta_0 \). We shall use “a.s.” to abbreviate “almost surely” and use \( \Theta \) to denote the parameter space, which is assumed to be compact. Moreover, \( \nabla^2 f \) will be used to denote the Hessian matrix of second partial derivatives \( \partial^2 f / \partial \theta_i \partial \theta_j \).

Theorem 1. For the SNN model defined by (9a) and (9b), suppose that \( \epsilon_t \) is normal and is independent of \( (u_{s,1}, \ldots, u_{s,q}) \) for \( s \leq t \), and that \( \{(x_t, \epsilon_t), t \geq 1\} \) is a stationary ergodic sequence with \( E(||x_t||^2) < \infty \). Then \( \hat{\theta} \to \theta_0 \) a.s. If furthermore \( \theta_0 \) belongs to the interior of \( \Theta \) and \( -E[\nabla^2 \log f_\theta(y_1, x_1)|_{\theta=\theta_0}] \) is positive definite, then \( (-\hat{\epsilon}_o(\hat{\theta}))^{1/2}(\theta - \theta_0) \) has a limiting normal distribution with zero mean and identity covariance matrix, as \( n \to \infty \).

Proof. We first prove \( \hat{\theta} \to \theta_0 \) a.s. Let \( \lambda(\theta) = E\{\log[f_\theta(y_1, x_1)/f_{\theta_0}(y_1, x_1)]\} \) and note that by Jensen’s inequality \( \lambda(\theta) < 0 \) if \( \theta \neq \theta_0 \). Take any \( \delta > 0 \). By the continuity of \( \lambda \) and compactness of \( \Theta \), there exists \( \eta > 0 \) such that \( \lambda(\theta) \leq -\eta \) for all \( \theta \in \Theta \) with \( ||\theta - \theta_0|| \geq \delta \). Moreover, using compactness and the form of \( f_\theta \) together with the fact that the conditional density function of \( y_t \) given \( x_t \) is \( f_{\theta_0}(y, x_t) \), it can be shown that there exists (sufficiently large) \( C > 0 \) such that

\[
E\{\sup_{\theta \in \Theta} \log[f_\theta(y_t, x_t)/f_{\theta_0}(y_t, x_t)]|x_t\} \leq C(1 + ||x_t||^2). \tag{A.1}
\]

Since \( E||x_1||^2 < \infty \), we can choose sufficiently large \( a > 0 \) such that \( E\{(1 + ||x_1||^2)I(||x_1|| > a)\} < \)
Then by (A.1) and stationarity,

$$
E\{\sup_{\theta \in \Theta} \log[f_\theta(y_t, x_t)/f_{\theta_0}(y_t, x_t)] \cdot I(||x_t|| > a) \leq CE\{(1 + ||x_t||^2)I(||x_t|| > a)\} < \eta/3.
$$

Therefore by the ergodic theorem (cf. Durrett (1996)),

$$
\lim_{n \to \infty} n^{-1} \sum_{t=1}^{n} \{\sup_{\theta \in \Theta} \log[f_\theta(y_t, x_t)/f_{\theta_0}(y_t, x_t)] \cdot I(||x_t|| > a) < \eta/3 \text{ a.s.}
$$

(A.2)

Fix any \( \theta \in \Theta \). It can be shown that there exists sufficiently large \( C_\theta \) for which

$$
E\{\sup_{\gamma \in \Theta} |\log f_\gamma(y_t, x_t) - \log f_\theta(y_t, x_t)| \cdot I(||x_t|| \leq a) \leq C_\theta \epsilon
$$

for all \( t \) and sufficiently small \( \epsilon \). Choosing \( 0 < \epsilon(\theta) < C_\theta^{-1} \eta/3 \), it then follows that

$$
E\{\sup_{\gamma \in \Theta} |\log f_\gamma(y_t, x_t)/f_\theta(y_t, x_t)| \cdot I(||x_t|| \leq a) \} < \eta/3
$$

By the Heine-Borel theorem, there exist a finite number, say \( m \), of balls \( B_i = \{ \gamma \in \Theta : ||\gamma - \theta_i|| < \epsilon(\theta_i) \} \) such that \( ||\theta_i - \theta_0|| \geq \delta \) and \( \bigcup_{i=1}^{m} B_i = \Theta \cap \{ \theta : ||\theta - \theta_0|| \geq \delta \} \). Application of the ergodic theorem then yields that for \( i = 1, \ldots, m \),

$$
\lim_{n \to \infty} n^{-1} \sum_{t=1}^{n} \{ \sup_{\gamma \in B_i} |\log f_\gamma(y_t, x_t)/f_{\theta_i}(y_t, x_t)| \} \cdot I(||x_t|| \leq a) < \eta/3 \text{ a.s.}
$$

Since \( \lim_{n \to \infty} n^{-1} \sum_{t=1}^{n} \log f_{\theta_i}(y_t, x_t)/f_{\theta_0}(y_t, x_t) = \lambda(\theta_i) \leq -\eta \), it then follows that

$$
\lim_{n \to \infty} n^{-1} \sum_{t=1}^{n} \{ \sup_{\gamma \in B_i} |\log f_\gamma(y_t, x_t)/f_{\theta_0}(y_t, x_t)| \} \cdot I(||x_t|| \leq a) < -2\eta/3 \text{ a.s.}
$$

(A.3)

Combining (A.3) with (A.2) shows that \( P\{\ell_0(\theta_0) > \sup_{\theta \in \Theta, ||\theta - \theta_0|| > \delta} \ell_0(\theta) \} \) for all large \( n \) = 1.

We next prove the asymptotic normality of \( \hat{\theta} \). Let \( \mathcal{F}_{t-1} \) denote the \( \sigma \)-field generated by \( y_1, \ldots, y_{t-1}, x_1, \ldots, x_t \). Since

$$
E\{\nabla \log f_\theta(y_t, x_t)|_{\theta=\theta_0} | \mathcal{F}_{t-1} \} = 0,
$$

(A.4)

\( \{\sum_{t=1}^{n} \nabla \log f_\theta(y_t, x_t)|_{\theta=\theta_0}, \mathcal{F}_n, n \geq 1 \} \) is a martingale. Note that

$$
E\{(\nabla \log f_\theta(y_t, x_t))(\nabla \log f_\theta(y_t, x_t))^T|_{\theta=\theta_0} | \mathcal{F}_{t-1} \} = -E\{\nabla^2 \log f_\theta(y_t, x_t)|_{\theta=\theta_0} | x_t \},
$$

which is a function of \( x_t \), say \( h(x_t) \). By the ergodic theorem, \( n^{-1} \sum_{t=1}^{n} h(x_t) \) converges a.s. to \( H^{\Delta} = -E[\nabla^2 \log f_\theta(y_1, x_1)|_{\theta=\theta_0}] \), which is assumed to be positive definite. Moreover, since \( \nabla \log f_\theta(y_t, x_t)|_{\theta=\theta_0} \) is stationary and square integrable, the Lindeberg condition holds and therefore
the martingale central limit theorem (cf. Section 7.7a of Durrett (1996)) can be applied to give the limiting standard normal distribution of \((nH)^{-1/2} \sum_{t=1}^{n} \nabla \log f_{\theta}(y_t, x_t)|_{\theta = \theta_0}\). Since \(-\ell_0(\theta_0)/n\) converges a.s. to \(H\) by the ergodic theorem and since \(\hat{\theta} \rightarrow \theta_0\) a.s., the remainder of the proof follows by standard arguments.

A simple refinement of the argument in the preceding paragraph shows that \(\ell_0(\theta)\) obeys the local asymptotic normality (LAN) conditions of LeCam. Therefore LeCam’s local asymptotic minimax theorem and Hajek’s convolution theorem are applicable to show that \(\hat{\theta}\) is asymptotically efficient; see Section 5.6 of LeCam and Yang (1990).

Without assuming the true underlying model to be an SNN, suppose that \(\{(x_t, y_t), t \geq 1\}\) is a stationary ergodic sequence such that

(C1) \(\lambda(\theta) \triangleq E\{\log f_{\theta}(y_1, x_1)\}\) is a continuous function with a unique maximum \(\theta^*\) belonging to the interior of \(\Theta\),

(C2) \(\lim_{\alpha \rightarrow 0} E\{\sup_{\theta \in \Theta} |\log f_{\theta}(y_1, x_1)| I(\|x_1\| > \alpha)\} = 0\),

(C3) \(\lim_{\varepsilon \rightarrow 0} E\{\sup_{\gamma \in \Theta, |\gamma - \theta| < \varepsilon} |\log f_{\gamma}(y_1, x_1) - \log f_{\theta}(y_1, x_1)| I(\|x_1\| \leq \alpha)\} = 0\)

for every \(\theta \in \Theta\) and \(\alpha > 0\).

Then a straightforward modification of the preceding consistency proof shows that \(\hat{\theta} \rightarrow \theta^*\) a.s. In particular, when \(x_t\) under the true underlying model is an ergodic Markov chain initialized at the stationary distribution and \(y_t\) is the component of \(x_{t+1}\) that is not contained in \(x_t\) (as in the stochastic difference equation (11)), \(E \log f(y_1, x_1) - \lambda(\theta)\) is the Kullback-Leibler divergence (or relative entropy) of the approximating SNN model from the true Markovian model with conditional density function \(f\) of \(y_1\) given \(x_1\). On the other hand, we can no longer use the martingale central limit theorem to derive the asymptotic normality of \(\hat{\theta}\) since (A.4) breaks down when the underlying model is not an SNN. To show that

\[
n^{-1/2} \sum_{t=1}^{n} \nabla \log f_{\theta}(y_t, x_t)|_{\theta = \theta^*} \text{ has a limiting normal distribution} \quad (A.5)
\]
as \(n \rightarrow \infty\), we need to impose mixing conditions on the stationary sequence \((y_t, x_t)\), cf. p. 425 of Durrett (1996). Since

\[
0 = n^{-1/2} \nabla \ell_0(\hat{\theta}) = n^{-1/2} \nabla \ell_0(\theta^*) + n^{-1} \nabla^2 \ell_0(\hat{\theta})\{\sqrt{n}(\hat{\theta} - \theta^*)\},
\]

where \(\hat{\theta}\) lies between \(\hat{\theta}\) and \(\theta^*\), we obtain the asymptotic normality of \(\hat{\theta}\) by applying the ergodic theorem to

\[
n^{-1} \nabla^2 \ell_0(\theta^*) = n^{-1} \sum_{t=1}^{n} \nabla^2 \log f_{\theta}(y_t, x_t)|_{\theta = \theta^*}
\]
and appealing to (A.5). This explains the conditions (C4)-(C6) in the following theorem.

**Theorem 2.** Suppose that \{(x_t, y_t), t \geq 1\} is a stationary ergodic sequence (not necessarily an SNN) satisfying conditions (C1)-(C3). Then \( \hat{\theta} \to \theta^* \) a.s., where \( \theta^* \) is given by (C1). Assume furthermore that for some \( \delta > 0 \) and \( \epsilon > 0 \),

(C4) \( E(\|\nabla \log f_\theta(y_1, x_1)|_{\theta = \theta^*}\|^{2+\delta}) < \infty \),

(C5) \( E(\sup_{|\gamma - \theta^*| < \epsilon} \|\nabla^2 \log f_\theta(y_1, x_1)|_{\theta = \gamma}\|) < \infty \),

(C6) \( \{(x_t, y_t), t \geq 1\} \) is strong mixing with mixing rate \( \alpha_n \) such that \( \sum_{n=1}^{\infty} \alpha_{n}^{\delta/2(2+\delta)} < \infty \).

Then \( \sqrt{n}(\hat{\theta} - \theta^*) \) has a limiting normal distribution with mean 0 and covariance matrix

\[
(E\nabla^2 \log f_\theta(y_1, x_1)|_{\theta = \theta^*})^{-1}V(E\nabla^2 \log f_\theta(y_1, x_1)|_{\theta = \theta^*})^{-1},
\]

where \( V = E(\nabla \log f_\theta(y_1, x_1)|_{\theta = \theta^*})\nabla \log f_\theta(y_1, x_1)|_{\theta = \theta^*}^T + 2 \sum_{i=2}^{\infty} (\nabla \log f_\theta(y_1, x_1)*(\nabla \log f_\theta(y_1, x_1))*^T|_{\theta = \theta^*} \).

Conditions (C4) and (C5) ensure that \( \lambda \) is differentiable at \( \theta^* \) and therefore \( \nabla \lambda(\theta^*) = 0 \); moreover, \( E\nabla \log f_\theta(y_1, x_1)|_{\theta = \theta^*} = \nabla \lambda(\theta^*) = 0 \). The strong mixing condition (C6) is satisfied by geometrically ergodic Markov chains since \( \alpha_n \) converges to 0 geometrically fast for such chains.
References


Rates," unpublished manuscript, University of California at Davis.


Table 1. Mean and Median Values of $k^{-1} \sum_{t=n+1}^{n+k} [\hat{f}_n(x_t) - f(x_t)]^2$ for Models (17) and (19)

<table>
<thead>
<tr>
<th>Model (17) $[n = k = 100]$</th>
<th>Model (19) $[n = k = 300]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean(s.e.)</td>
<td>Median</td>
</tr>
<tr>
<td>SNN</td>
<td>0.165(0.010)</td>
</tr>
<tr>
<td>NN1</td>
<td>0.169(0.014)</td>
</tr>
<tr>
<td>NN2</td>
<td>0.205(0.021)</td>
</tr>
<tr>
<td>MARS</td>
<td>0.297(0.019)</td>
</tr>
<tr>
<td>PPR</td>
<td>0.223(0.014)</td>
</tr>
</tbody>
</table>

Table 2. Median and Mean Values of $(y_{n+t} - \hat{y}_{n+t})^2$ for $l = 1, \ldots, 10$, and of In-Sample Average Squared Error ($l = 0$)

<table>
<thead>
<tr>
<th>$x_{t-s}$ observed; n=2000</th>
<th>${x_t}$ unobservable; n=4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNN with $J_{max} = 6$</td>
<td>NN6</td>
</tr>
<tr>
<td>SNN with $J_{max} = 5$</td>
<td>NN5</td>
</tr>
<tr>
<td>Lead l</td>
<td>Median</td>
</tr>
<tr>
<td>0</td>
<td>4.7</td>
</tr>
<tr>
<td>1</td>
<td>6.6</td>
</tr>
<tr>
<td>2</td>
<td>39.1</td>
</tr>
<tr>
<td>3</td>
<td>124.9</td>
</tr>
<tr>
<td>4</td>
<td>289.8</td>
</tr>
<tr>
<td>5</td>
<td>641.3</td>
</tr>
<tr>
<td>6</td>
<td>844.6</td>
</tr>
<tr>
<td>7</td>
<td>933.4</td>
</tr>
<tr>
<td>8</td>
<td>1806.1</td>
</tr>
<tr>
<td>9</td>
<td>2221.9</td>
</tr>
<tr>
<td>10</td>
<td>5349.4</td>
</tr>
</tbody>
</table>

Table 3. Multistep Postsample Forecasts of Annual Sunspot Numbers for 1980-1987

<table>
<thead>
<tr>
<th>Year</th>
<th>Observation</th>
<th>TAR Prediction</th>
<th>Error</th>
<th>FAR Prediction</th>
<th>Error</th>
<th>SNN Prediction</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1980</td>
<td>154.7</td>
<td>160.1</td>
<td>-5.4</td>
<td>168.5</td>
<td>-13.8</td>
<td>157.7</td>
<td>-3.0</td>
</tr>
<tr>
<td>1981</td>
<td>140.5</td>
<td>141.8</td>
<td>-1.3</td>
<td>140.6</td>
<td>-0.01</td>
<td>134.0</td>
<td>6.5</td>
</tr>
<tr>
<td>1982</td>
<td>115.9</td>
<td>96.4</td>
<td>19.5</td>
<td>105.9</td>
<td>10.0</td>
<td>101.5</td>
<td>14.4</td>
</tr>
<tr>
<td>1983</td>
<td>66.6</td>
<td>61.8</td>
<td>4.8</td>
<td>70.9</td>
<td>-3.3</td>
<td>66.2</td>
<td>0.4</td>
</tr>
<tr>
<td>1984</td>
<td>45.9</td>
<td>31.1</td>
<td>14.8</td>
<td>42.1</td>
<td>3.8</td>
<td>37.2</td>
<td>8.7</td>
</tr>
<tr>
<td>1985</td>
<td>17.9</td>
<td>18.1</td>
<td>-0.2</td>
<td>22.5</td>
<td>-4.6</td>
<td>18.8</td>
<td>-0.9</td>
</tr>
<tr>
<td>1986</td>
<td>13.4</td>
<td>18.9</td>
<td>-5.5</td>
<td>12.1</td>
<td>1.3</td>
<td>12.7</td>
<td>0.7</td>
</tr>
<tr>
<td>1987</td>
<td>29.2</td>
<td>29.9</td>
<td>-0.7</td>
<td>7.5</td>
<td>21.7</td>
<td>28.3</td>
<td>0.9</td>
</tr>
</tbody>
</table>
Figure 1: A Simulated Time Series from Distorted Long-Memory Model

Figure 2: Boxplot of Logarithm of Error(NN)/Error(SNN) for $n = 4000$ in Table 2
Figure 3: Mortality Rates over Time (in Months)

Figure 4: Fitted AR Values and Postsample Forecasts
Figure 5: In-sample and Postsample One-step Ahead Forecasts from Fitted SNN Model

Figure 6. Cumulative Absolute Prediction Errors.
SNN: Solid, NN: Dot-Dash, MARS: Short-Dash, PPR: Dot, AR: Long-Dash