Determining the Model Order of Nonlinear Input/Output Systems

Carl Rhodes
Chemical Engineering, 210-41, California Institute of Technology, Pasadena, CA 91125

Manfred Morari
Institut für Automatik, ETH-Z/ETL, CH-8092 Zürich, Switzerland

A method for determining the proper regression vector for recreating the dynamics of nonlinear systems is presented. The false nearest neighbors (FNN) algorithm, originally developed to study chaotic time series, is used to determine the proper regression vector for input/output system identification and inferential prediction using only time-series data. The FNN algorithm for solving these problems is presented, and the problem of analyzing noise corrupted time series is discussed. The application of the algorithm to a number of examples including an electrical-leg stimulation experiment, an industrial pulp-digester model, a polymerization model, and a distillation-column simulation is presented and the results are analyzed.

Introduction

Recently, there have been significant advances in the control of nonlinear systems. For these advances to be applied to systems in the chemical process industry, it is necessary to develop nonlinear models that can be easily utilized by these nonlinear control schemes. One method commonly used for developing nonlinear models involves physical modeling of the system to be controlled from first principles of chemistry and physics. While these models can be extremely accurate for simple systems and systems where the effects of unmodeled dynamics are small (such as mechanical systems), it is often the case that a small, accurate model based on physical modeling alone cannot be developed for large-scale chemical processes.

The reason that models based on physical intuition may not be well-suited for nonlinear control is twofold. The first reason that these models fail is that first-principle models can be very large. Complete physical models consisting of hundreds of states are not uncommon for many chemical processes such as distillation. While this may not be problematic for simulation studies, current methods of nonlinear control are not equipped to deal with such large descriptions. The second reason that physical models may not be sufficient for control purposes is that there may be effects that are either difficult or impossible to account for in the physical modeling process. These effects are quite common in chemical processes and may be difficult to quantify in systems exhibiting strongly nonlinear behavior.

For these reasons, in some situations it is necessary to use nonlinear "black box" identification methods to develop an accurate model of system dynamics. Recently, there has been a great deal of work examining the black-box identification of nonlinear systems using input/output data. A good overview of the work in this field can be found in (Sjöberg et al., 1995). While most of the recent work has focused on utilizing neural networks, other recently published works have examined nonlinear ARMAX model structures (Chen and Billings, 1989), radial basis functions, wavelets, hinging hyperplanes (Beiman, 1993), and MARS (Friedman, 1991).

The common focus of all these works is determining and approximating the functional relationship between a regression vector and an output vector. In other words, a set of observed regressors $\psi(t)$ for $t = 1, \ldots, N$ and observed outputs $y(t)$ related to that regression vector are given. All these methods attempt to find a functional relationship

$$y(t) = G[\psi(t)] \tag{1}$$

that minimizes some error function E, which is often of the form

Correspondence concerning this article should be addressed to M. Morari.
Present address of C. Rhodes: Institut für Automatik, ETH-Z/ETL, CH-8092 Zürich Switzerland

$$E = \sum_{i=1}^{N} \mathrm{Err}[y(i) - G[\psi(i)]], \tag{2}$$

where Err is typically some norm operator. Commonly, the regression vector for single-input/single-output systems consists of delayed versions of the input and output:

$$\psi(t) = [y(t-\tau), y(t-2\tau), \ldots, y(t-l\tau), u(t-\tau), \ldots, u(t-m\tau)]. \tag{3}$$

While many works focus on ways of parameterizing and determining the optimal function G, there is relatively little work for nonlinear systems on determining the proper form of the regressor ψ. Specifically, in nearly all of these studies the number of delayed terms in the regression vector (l and m) is assumed to be known. The function G is then estimated using the observed regressors and outputs in the time series. If the differences between the values of the approximated function $G[\psi(t)]$ and the actual output $y(t)$ are large, there could be two sources of the error. The first is an estimated function G, which does not do a good job of representing the relationship between the regressor and the output. The second possibility is that the regressor $\psi(t)$ simply does not contain enough information to accurately predict the future output. If the prediction error is large, it is impossible to tell from the output of these commonly utilized algorithms whether the source of the error is an incorrect functional approximation or a regression vector that does not contain enough terms (the MARS modeling scheme is the one exception; the algorithm "trims" away terms in the regressor that do not reduce the prediction error).

In the identification of linear systems, determining the exact source of the error is not a problem. Due to the linearity of the system, choosing the regression vector is the only critical step since the functional relationship is defined by the linearity of the system. Since computing the unknown parameters involves relatively little computation, determining the proper regression vector involves computing the optimal linear model for various regression vectors and using some function (such as AIC or the F-test) to determine when the error no longer significantly decreases with respect to the complexity resulting from additional terms (Söderström and Stoica, 1989). For nonlinear systems, this method is infeasible since nonlinear optimizations typically require large amounts of computational effort and it is difficult to guarantee that nonlinear modeling schemes will find an optimal solution.

For nonlinear systems, it would be preferable to break the identification process into two distinct parts. First, the proper regression vector should be determined. The proper regression vector should contain only those terms needed to accurately predict the output. Once the proper regression vector is determined, the functional relationship between the regressor and the output can be estimated.

While there is extensive work in determining the proper regression vector for linear systems (AIC, F-test, etc.), there is relatively little work in the field of nonlinear systems. A method based on geometric ideas for determining the proper dimension for the regression vector is presented here. These ideas build upon similar methods developed for the analysis of self-driven chaotic time series (Abarbanel et al., 1993; Abarbanel, 1996). Another approach to this problem is the statistical approach. Using this method, the optimal prediction error is estimated by embedding the data and analyzing the variance of the outputs for regression vectors that are close in the regressor space (Poncet and Moschytz, 1994). By analyzing the average variance over the regression space, an estimate of the error of the optimal model is found for each regression vector. When the error fails to decrease significantly as a function of increasing model order, the proper regression vector is found. A third method is quite similar to the AIC, and is used to identify the proper dimension of hidden Markov models (HMM) (Finesso, 1990).

Embedding Theorems for Nonlinear Systems

First-principle models of physical systems often take the following form

$$\frac{dx}{dt} = f(x, u) \tag{4}$$

$$y = h(x), \tag{5}$$

where $x \in \Re^n$ is the state vector of the system; u is a controlled input; and y is a measured output. While models arrived at by first principles often do a good job of representing the major features of the dynamics of physical systems, the dynamics of the mathematical model and the actual physical system can be quite different. This can be the result of unmodeled dynamics and/or errors in the parameters contained in functions f and h. Since it may not be possible to measure all the states due to either economic or physical limitations, determining the unknown parameters and proper form of the unmodeled dynamics of the physical model (Eq. 4) can be difficult.

For these reasons, it would be desirable to be able to compute a model of the system dynamics directly from input/output data. In mathematical terms, a model of the form

$$y(t) = G[y(t-\tau), y(t-2\tau), \ldots, y(t-l\tau), u(t-\tau), \ldots, u(t-m\tau)] \tag{6}$$

should be found where τ is the sampling time of the system. A few interesting questions arise naturally from this discussion. Assume the system to be modeled is known exactly (say it is Eq. 4) and the controlled input (u) only changes at the sampling times of the system:

1. Does a representation in the form of Eq. 6 exist?
2. How many delayed terms (l and m) are needed to represent the input/output dynamics of Eq. 4 if the size of the state space is known ($x \in \Re^n$)?
3. Can the function G be determined directly from the functions f and h?

If the functions f and h are linear, then the answers to these questions are known. An equivalent representation of the input/output dynamics does exist, and a number of terms $l = m = n$ are sufficient to represent the input/output dynamics of the system described by Eq. 4. In addition, for linear systems the function G can be determined exactly from the functions f, h, and the time delay τ using the z-transformation (Franklin et al., 1986).

When the functions f and h are nonlinear, the answers to these questions are not as simple. The well-known results for linear systems are based strongly on the linearity of the system. Specifically, the term G can be calculated because the dynamics are invariant with regard to the location of the trajectory in the state space of the system. For nonlinear systems, computing the function G from f and h is impossible (using current methods) except in trivial cases. For this reason, different methods must be utilized.

To get some insight into the problem, first the case of autonomous (self-driven) systems will be examined. The following question was first studied in the analysis of chaotic time series. Given the following system

$$\frac{dx}{dt} = f(x) \tag{7}$$

$$y = h(x), \tag{8}$$

where $x \in \Re^n$, it was shown by Takens (1981) that the output at any moment can be generically written as a nonlinear function of time-delayed versions of that same output. In other words, there exists some G such that

$$y(t) = G[y(t-\tau), y(t-2\tau), \ldots, y(t-l\tau)]. \tag{9}$$

Additionally, it was shown by Takens that $l > 2n$ is a sufficient condition for this relationship to exist. The proof is based on the Whitney embedding theorem of differential geometry, which states that any k-dimensional manifold can be embedded in the space $\Re^{2k+1}$ (Guillemin and Pollack, 1974). Takens showed that an embedding exists between the state space of the original system and the time-delayed version of the system. Since an embedding is a nonlinear one-to-one relationship, the original state space dynamics also exist in the delayed coordinates. These ideas were later extended by another set of authors (Sauer et al., 1991) to a slightly more general result and a separate result relevant to inferential prediction that is discussed in the section on FNN for inferential measurement selection.

For input/output systems, a similar result to that of Takens was first suggested by Casdagli (1992). In this article, a brief outline of the methods needed to prove an embedding theorem for nonlinear systems was given. For discrete-time systems, these results were recently formalized by Poncet et al. (1995) using the methods outlined by Casdagli. In this article, it was shown that l, $m = d + 1$ is a sufficient condition to represent the dynamics of a system with d states under standard assumptions, such as state observability. While a formal proof does not yet exist for continuous-time systems, the results should be similar to the discrete-time result.

While these results are interesting from a mathematical perspective, they are little help when it comes to identifying the number of delay terms needed to model a system from input/output data. For reasons of parsimony, it is preferable to determine the smallest possible number of delay terms needed to represent the dynamics. In most identification problems, the dimension of the system generating the time series is unknown. In addition even when the dimension of the state-space system is known *a priori*, there may exist a smaller representation of the dynamics (since the condition is sufficient and not necessary). For these reasons, there is a need to determine the proper "dimensionality" of the system in the time-delay description directly from input/output data.

FNN: Algorithm for Determining the Proper Embedding Dimension

Identification consists of two distinct steps: determining the past terms that are needed for predicting future outputs, and determining the function relating those past terms to the future terms. While many methods for determining the functional relationship have been published recently, many of these methods do not deal with the problem of determining the proper regressor vector. Often it is simply "assumed" that the dynamics can be represented using a particular regression vector, and then the functional relation is calculated.

The false nearest neighbors (FNN) algorithm was originally developed for determining the smallest dimension regression vector needed to recreate the dynamics of autonomous chaotic systems (Kennel et al., 1992). The idea behind the FNN algorithm is geometric in nature. If there is enough information in the regression vector to predict the future output, then any two regression vectors that are close in the regressor space will also have future outputs that are close in some sense. If there is not enough information present in the regressor vector to recreate the dynamics of the system, then there will be some neighborhoods in the regressor space with a wide range of future outputs. These trajectories, which are close in the regression space and with vastly different outputs, can be thought of as *false neighbors*, since they are close in the regression space only because of projection onto a space with a dimension too small to represent the dynamics of the system. For noise-free data, there will no longer be any false neighbors when the dimension of the regression vector is large enough to allow accurate prediction of future outputs.

Original FNN algorithm for autonomous time series

To determine whether neighbors are true or false, a test must be defined to determine whether the neighbors have future outputs that are "far apart." For this purpose, a ratio test determines whether the distance between future outputs is significantly larger than the distance between time-delay regression vectors that are close in the regressor space. If the distance between future outputs is "large" when divided by the distance between two points that are "nearest neighbors" in the regressor space, then the neighbors are considered to be false.

Here is the original FNN algorithm, which was designed for autonomous systems:

1. Form the set of regressors

$$\psi_l(t) = [y(t-\tau), \ldots, y(t-l\tau)] \tag{10}$$

and related outputs $y(t)$ from time-series data.

2. Identify the closest point (in the Euclidean sense) to a given point in the regression space. That is, for a given regressor $\psi_l(k)$ find another regressor $\psi_l(j)$ in the data set such that the following distance d is minimized:

$$d = \| \psi_l(k) - \psi_l(j) \|_2. \tag{11}$$

It should be noted that times k and j themselves are not necessarily close to one another. In fact, if k and j are always close to one another, the sampling time may be too small and there may be problems in accurately estimating the dimension of the regression vector (Fredkin and Rice, 1995).

3. Determine if the following expression is true or false:

$$\frac{|y(k)-y(j)|}{\|\psi_l(k)-\psi_l(j)\|_2} \leq R, \tag{12}$$

where R is a previously chosen threshold value. If expression 12 is true, then the neighbors are recorded as *true* neighbors. If the expression is false, then the neighbors are *false* neighbors.

4. Continue the algorithm for all times k in the data set. Calculate the percentage of points in the data set that have false nearest neighbors.

5. Continue the algorithm for increasing l until the percentage of false nearest neighbors drops to zero (or some acceptably small number). If the percentage of false neighbors is large, then the regressor vector must be extended to include more terms.

FNN and input/output dynamics

For determining the proper regressor for input/output dynamics, the only change to the original FNN algorithm involves the regression vector itself. For input/output dynamics, the regression vector must contain delayed versions of both the input and the output:

$$\psi_{l,m}(k)=[y(k-\tau),\ldots,y(k-l\tau),u(k-\tau),\ldots,u(k-m\tau)]. \tag{13}$$

To determine the proper dimension of the regressor, the percentage of false nearest neighbors must be determined for different numbers of delays in *both* the input and output. Here is an outline of the FNN algorithm for input/output data.

1. Identify the nearest neighbor to a given point in the regressor space. For a given regressor

$$\psi_{l,m}(k)=[y(k-\tau),\ldots,y(k-l\tau),u(k-\tau),\ldots,u(k-m\tau)], \tag{14}$$

find the nearest neighbor $\psi_{l,m}(j)$ such that the distance d is minimized:

$$d=\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2. \tag{15}$$

2. Determine if the following expression is true or false:

$$\frac{|y(k)-y(j)|}{\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2} \leq R. \tag{16}$$

If expression 16 is true, then the neighbors are *true* neighbors. If the expression is false, then the neighbors are *false* neighbors.

3. Continue the algorithm for all times k in the data set. Calculate the percentage of points in the data set that have false nearest neighbors.

4. Continue the algorithm for increasing l and m until the percentage of false nearest neighbors drops to zero (or some acceptably small number). If the percentage of false neighbors is large, then the regressor vector must be extended to include more delayed input and/or output terms.

Commonly for state-space systems of identical size, the regression vector for an input/output system needed to recreate the dynamics will have a dimension two times larger than the regressor needed for an autonomous system. Because there are two parameters present in the regression vector, a two-dimensional search is needed to determine the proper number of delayed terms. This makes the FNN algorithm for input/output time series slightly more difficult to implement than for autonomous systems. In addition, as will be shown in the next section, more data are needed by FNN to analyze input/output dynamics.

Data requirements and the choice of the threshold R

The one parameter that needs to be determined before performing the false nearest neighbors algorithm is the threshold constant R. Choosing a suitable threshold is important for the following reasons, which will be illustrated later in this subsection. If R is too small, then the percentage of false nearest neighbors may not drop to zero in the proper dimension. On the other hand, if R is too large, it is possible that the FNN algorithm will predict that future outputs are predictable with a regressor dimension that is too small.

As will be shown, the optimal choice of R will depend on knowledge of the function G, which relates the regressor to the future outputs. Since the FNN algorithm is intended to determine the proper regression vector independent of G, at first glance the results that follow may appear to be of little use. These results are important in certain limiting cases, however, and the derivation of these results will be helpful for the analysis of noise-corrupted data, which is presented in the following subsection.

When a regressor with the proper embedding dimension is analyzed by the FNN algorithm, the algorithm should predict that the percentage of false neighbors is zero. In order to determine which values of the threshold that this property holds for, assume that the number of delayed terms (l and m of Eq. 3) and the function (G of Eq. 1) relating these delayed terms to the future output are known. For all regression vectors embedded in the proper dimension, if the effects of noise are excluded, two regression vectors that are close in the regression space and their future outputs are related in the following way:

$$y(k)-y(j)=DG(\psi_{l,m}(k))[\psi_{l,m}(k)-\psi_{l,m}(j)]+\mathcal{O}([\psi_{l,m}(k)-\psi_{l,m}(j)])^2, \tag{17}$$

where $DG(\psi)$ is the Jacobian of the function G at ψ. Ignoring higher-order terms, and using the Cauchy–Schwarz inequality

$$|y(k)-y(j)| \leq \|DG(\psi_{l,m}(k))\|_2 \|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2 \tag{18}$$

$$\frac{|y(k)-y(j)|}{\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2} \le \|DG(\psi_{l,m}(k))\|_2 \quad (19)$$

$$\frac{|y(k)-y(j)|}{\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2} \le \max_t \|DG(\psi_{l,m}(t))\|_2$$

$\forall k$ and nearest neighbor j. (20)

For the FNN algorithm to correctly find that there are no false nearest neighbors in the proper embedding dimension, the threshold R must be chosen such that it is larger than $\max_t \|DG(\psi_{l,m}(t))\|_2$.

Another property that is important for the FNN algorithm is that the percentage of false nearest neighbors should be finite if the number of terms in the regression vector is too small. In order to analyze which values of the threshold R this property holds for, assume that there is an infinite amount of data available and that the regression vector being analyzed by the FNN algorithm is too small to correctly predict the output. Since the regression vector is too small, there will be certain regions of the regression space where the output cannot be accurately predicted. This means that two points from the data set with the same location in the regression space may have very different future outputs. In the case of infinite data, where all the regressors in the data set have neighbors that are arbitrarily close, the threshold can be made arbitrarily large (this is formalized in Rhodes and Morari (1997)) and still correctly predict that there are false neighbors in embedding dimensions that are too small.

While the previous theoretical results are only important for specific limiting cases, these results also provide some guidance for the analysis of finite data sets where the function G is unknown. For example, the choice of the threshold may have to be adjusted if the function G is expected to have a large gain. However, for finite data it has been observed that false neighbors tend to have outputs that are "very far" apart so a large threshold should still lead to false neighbors for regressors that don't contain enough delayed terms. While R cannot be arbitrarily large when working with finite amounts of data, in practice fairly large values of R (compared to the gain of G) can be used.

When analyzing any time series with the FNN algorithm, the amount of data needed to carry out an accurate analysis also needs to be considered. For the ratio test in the algorithm to correctly interpret the idea of closeness in the output space, the nearest neighbors must be relatively close in the regression space. If distance between nearest neighbors is large, then the difference between their outputs can be very large, and the ratio test could still call the two data points true neighbors when the outputs are totally uncorrelated. For all neighbors to be close, the time-series data should "fill out" the regression space to be analyzed. To fill an ∞-norm unit ball of n-dimension with data such that all individual data points are within an δ-sized ball of one another, approximately δ^{-n} points are needed. As the dimension of the ball is increased, the amount of data needed to cover the space increases exponentially. This is a well-known problem in nonlinear identification, and is known as the *curse of dimensionality* (Weigend and Gershenfeld, 1994).

The curse of dimensionality has the following implication for the FNN algorithm. In order to accurately assess whether regression vectors of up to dimension n are able to accurately predict future outputs, a time-series of length approximately 10^n is needed. It should be noted that for certain systems, such as autonomous chaotic attractors, the dynamics of the system (and therefore the regression vectors) can lie on a submanifold of the overall regression space and therefore the space to be filled may be of smaller dimension than the entire regression space (see Abarbanel et al., 1993). For some input/output systems that are examined here, it is possible that significant correlations may exist among the terms contained in the regression vector. However, locally the data can lie on a manifold with the same dimension as the dimension of the regression vector. To fill out this manifold, again the amount of data should increase exponentially with the dimension of the regression vector.

Choosing a single threshold that will work well for all data sets is an impossible task. However, the authors have observed that a good choice of the threshold for a wide range of problems is in the range of 10–15. While the scaling of the time series and G for each individual problem may affect the choice of R, it has been observed that qualitatively the results of the FNN algorithm are similar over a wide range of values of the threshold. In cases where the time series is relatively short, the results could be more sensitive to the choice of threshold, and it might be necessary to compare the results from a number of different thresholds to come to a reasonable conclusion on the regressor size.

FNN, time-series, and noise corruption

The original FNN algorithm is not robust to the presence of noise in the time series. The problem can be illustrated in the following way. When noise is present in the time series, two identical regression vectors that would have identical outputs if no noise were present in the time series can have outputs that differ by some finite amount. Therefore even when the noise-corrupted time series is embedded in a regression space with the proper dimension, the original FNN ratio test can fail (Rhodes and Morari, 1997).

Assume (as before), that the proper embedding dimension and the function relating the delayed inputs and outputs to the future output are known. Also assume that the observed inputs ($u^{\text{obs}} = u + \delta^u$) and outputs ($y^{\text{obs}} = y + \delta^y$) contain magnitude-bounded noise ($|\delta^u| \le \delta^u_\infty$, $|\delta| \le \delta^y_\infty$). The observed regression vectors will also contain noise $\psi^{\text{obs}}_{l,n}(t) = [y^{\text{obs}}(t-\tau), \ldots, u^{\text{obs}}(t-\tau), \ldots] = [y(t-\tau)+\delta^y_1, \ldots, u(t-\tau)+\delta^u_1, \ldots,]$. As before,

$$|y^{\text{obs}}(k)-y^{\text{obs}}(j)| \le |G(\psi_{l,m}(k))-G(\psi_{l,m}(j))| + 2\delta^y_\infty \quad (21)$$

$$\le \|DG(\psi_{l,m}(k))\|_2 \|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2 + 2\delta^y_\infty \quad (22)$$

$$\le \|DG(\psi_{l,m}(k))\|_2 \left(\|\psi^{\text{obs}}_{l,m}(k)-\psi^{\text{obs}}_{l,m}(j)\|_2 + 2\sqrt{l\delta^{y2}_\infty + m\delta^{u2}_\infty}\right) + 2\delta^y_\infty \quad (23)$$

$$\le \max_t \|DG(\psi_{l,m}(t))\|_2 \left(\|\psi^{\text{obs}}_{l,m}(k)-\psi^{\text{obs}}_{l,m}(j)\|_2 + 2\sqrt{l\delta^{y2}_\infty + m\delta^{u2}_\infty}\right) + 2\delta^y_\infty. \quad (24)$$

This result can be put in a form similar to the FNN threshold test:

$$\frac{|y^{\text{obs}}(k)-y^{\text{obs}}(j)|}{\|\psi_{l,m}^{\text{obs}}(k)-\psi_{l,m}^{\text{obs}}(j)\|_2} \le \max_t \|DG(\psi_{l,m}(t))\|_2 + \frac{2\sqrt{l\delta_\infty^{y2}+m\delta_\infty^{u2}}\,\max_t\|DG(\psi_{l,m}(t))\|_2+2\delta_\infty^{y}}{\|\psi_{l,m}^{\text{obs}}(k)-\psi_{l,m}^{\text{obs}}(j)\|_2}. \quad (25)$$

There are two distinct terms on the righthand side. The first term is due to the gain of the noise-free system, and the second term is due to noise contamination. Note that the second term is inversely proportional to the separation between the nearest neighbors. In fact, if the original FNN ratio test is used to analyze noise-corrupted time series, then the algorithm will report false neighbors in the proper embedding dimension that result only from noise. Even worse, the problem of false nearest neighbors occurring due to noise is compounded when more data are used for analysis (see Rhodes and Morari, 1997, for more details).

In order to solve this problem, a new threshold test can be used for time series where significant noise corruption is expected. A logical form of this threshold test based on the previous analysis is

$$\frac{|y(k)-y(j)|}{\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2} \le R + \frac{2R\sqrt{l\epsilon^{y2}-m\epsilon^{u2}}+2\epsilon^{y}}{\|\psi_{l,m}(k)-\psi_{l,m}(j)\|_2}, \quad (26)$$

where ϵ^u and ϵ^y are related to the expected magnitude of the noise contained in the input and output, respectively.

Notice that this threshold test has two distinct limits. When the data are "dense," the second term on the righthand side dominates. In this limit, the new threshold test simply determines whether the two neighbors' outputs are within some specified noise-related distance ($2R\sqrt{l\epsilon^{y2}+m\epsilon^{u2}}+2\epsilon^{y}$) of one another. In the limit where neighbors are relatively far apart (where problems associated with noise corruption are not expected), this threshold test reverts back to the original FNN threshold test.

While this new threshold test does a better job in dealing with noise-corrupted data, there are more parameters to be determined. The main parameter R should be chosen as in the original FNN algorithm, and the terms ϵ^y and ϵ^u should be no larger than the magnitude of the expected noise in the output and input, respectively. Keep in mind that this threshold test accounts for the worst case, and is conservative in its construction. For this reason, determining the exact magnitude of the noise is not always necessary. In fact, it has been observed that by simply including this extra term in the threshold test better results are obtained, even when the value of ϵ^y is significantly smaller than the magnitude of the noise (Rhodes and Morari, 1997). When the magnitude of the noise is known, it has been observed that choosing a value of ϵ equal to around one-tenth of the standard deviation of the noise leads to good results when using the modified FNN algorithm.

Case Studies

Electrical-leg stimulation

The first example is the identification of the dynamics of a human leg stimulated by an electrical signal. Jan Schultheiss was involved in research to help paraplegics walk with the aid of a controlled electrical signal at the Swiss Paraplegic Center at Balgrist Hospital in Zürich, Switzerland (del Re et al., 1994). The data are from an experiment that measures the effect of stimulating the quadriceps muscle with an electrical signal. The input is the pulse width of an electrical stimulation signal to the muscle, and the output is the angle of leg extension measured at the knee. The final goal of the project is to design a controller for helping paraplegics to walk by utilizing controlled electronic stimulation.

The time-series data consists of 654 input/output samples where the pulse width of the electronic input that stimulates the leg is scaled to lie between 0 and 1. The leg angle output is scaled such that the full leg extension (or full quadriceps contraction) is 1, the knee being at a resting position is 0, and a leg that swings past its normal resting position on relaxation of the quadriceps causes a negative output. The sampling time for the system is 100 ms. A plot of the entire time series to be analyzed can be seen in Figure 1.

The original FNN algorithm with a threshold of 10 was applied to the entire time series. Since the time series is relatively short, problems associated with noise corruption are not expected. The results of the FNN algorithm for this data set are given in Table 1. By examining the results, it can be seen that the percentage of false nearest neighbors drops to a low percentage of 1.1 when the regression function takes the form.

$$y(t)=G[y(t-1),\, y(t-2),\, u(t-1),\, u(t-2)]. \quad (27)$$

While it is difficult to determine the definitive "proper" form of the regression vector since there are so few data, it appears that utilizing two delayed versions of both the output and input should lead to accurate prediction. Additionally, a regression vector with two delayed output terms and a single input delay may work reasonably well since the percentage of

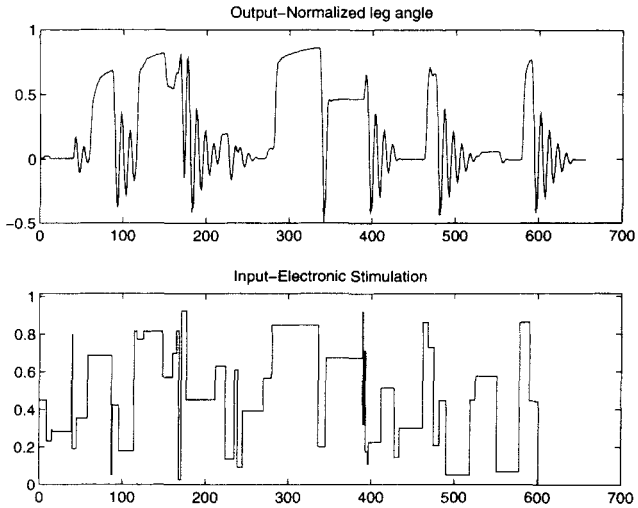

Figure 1. Leg stimulation, experimental time series.

Table 1. FNN Results for Leg-Stimulation Data, $R = 10$

Input Delays m	Output Delays l			
% FNN	0	1	2	3
0	100.0	72.2	13.5	4.2
1	87.4	36.5	5.5	2.6
2	82.2	27.8	1.1	0.3
3	74.5	23.2	0.5	0.3

Table 4. MSPE Error for Leg-Stimulation Simulation

Input Delays m	Output Delays l			
MSPE	0	1	2	3
0	4.7×10^{-2}	4.6×10^{-2}	4.5×10^{-2}	4.3×10^{-2}
1	3.9×10^{-2}	1.4×10^{-2}	5.5×10^{-3}	8.1×10^{-3}
2	2.8×10^{-2}	2.1×10^{-2}	7.5×10^{-3}	2.9×10^{-3}
3	2.8×10^{-2}	2.3×10^{-2}	1.9×10^{-3}	2.9×10^{-3}

FNN is again fairly small. To illustrate the effect of a change in the threshold value R, the results of the FNN algorithm with $R = 15$ are presented in Table 2. While the percentage of false nearest neighbors is smaller than in Table 1 for each entry, qualitatively the results are similar. In this case, the results of the FNN algorithm are relatively unchanged over a wide range of threshold values R.

In order to verify the results of the FNN algorithm, the nonlinear functional fitting algorithm MARS (Friedman, 1991) was used to determine the function G for the same number of delay terms analyzed in Tables 1 and 2. The first 500 points of the time series are used to build a model and the mean squared prediction error (MSPE) is determined by comparing the results of the MARS modeling to the last 154 points of the experimental time series. After the model was found for a given regression vector, two types of error analysis were performed. The first determines the one-step-ahead prediction error, where the actual past outputs and inputs are used to predict the next output of the system. The results of the one-step-ahead analysis are given in Table 3. The second analysis involves a simulated model that utilizes the inputs and only the initial conditions of the experimental time series. Obviously, the second method typically has a larger error, and the results of this method are given in Table 4.

In examining these results, it appears that large values of FNN correspond to larger values of the MSPE for both one-step prediction and simulation. For the one-step-ahead prediction, the MSPE should always decrease when more terms are used. However, for the simulation results the error is not guaranteed to decrease since the error resulting from a single step may compound when previous predicted outputs are used to determine future outputs. For these results, it appears that the MSPE does not significantly decrease when including more than two delayed versions of both the input and output. To give an idea of the accuracy of the MARS modeling scheme for the regression vector given in Eq. 27, the results of the one-step-ahead prediction and simulation are shown in Figures 2 and 3, respectively.

Without the FNN method, it would be necessary to build many models with different numbers of delayed terms. After these models were built, the results would have to be analyzed and compared to determine a suitable number of delayed terms. With the FNN method, a suitable number of delayed terms can be determined in a single step before completing the nonlinear functional modeling. By utilizing the FNN algorithm, time can be saved when performing the entire nonlinear identification process.

Table 2. FNN results for Leg-Stimulation Data, $R = 15$

Input Delays m	Output Delays l			
% FNN	0	1	2	3
0	100.0	68.3	6.8	3.5
1	87.2	23.8	2.3	1.4
2	80.5	18.5	0.8	0.3
3	70.0	16.5	0.5	0.3

Table 3. MSPE Error for Leg-Stimulation One-Step-Ahead Prediction

Input Delays m	Output Delays l			
MSPE	0	1	2	3
0	4.7×10^{-2}	5.3×10^{-3}	4.1×10^{-4}	1.9×10^{-4}
1	7.7×10^{-2}	4.1×10^{-3}	2.7×10^{-4}	1.4×10^{-4}
2	2.2×10^{-2}	4.7×10^{-3}	1.9×10^{-4}	4.8×10^{-5}
3	1.8×10^{-2}	4.0×10^{-3}	1.6×10^{-4}	6.5×10^{-5}

Continuous polymerization reactor

The following example illustrates identification using data from a model of a continuous polymerization reactor. The model describes the free-radical polymerization of methyl methacrylate with azobisisobutyronitrile as an initiator and toluene as a solvent. For further information on the details of this model and how it is derived, see Doyle et al. (1995). The reaction takes place in a jacketed CSTR, and after some simplifying assumptions are made the first-principles model is

$$\dot{x}_1 = 10(6 - x_1) - 2.4568 x_1 \sqrt{x_2} \tag{28}$$

$$\dot{x}_2 = 80u - 10.1022 x_2 \tag{29}$$

$$\dot{x}_3 = 0.0024121 x_1 \sqrt{x_2} + 0.112191 x_2 - 10 x_3 \tag{30}$$

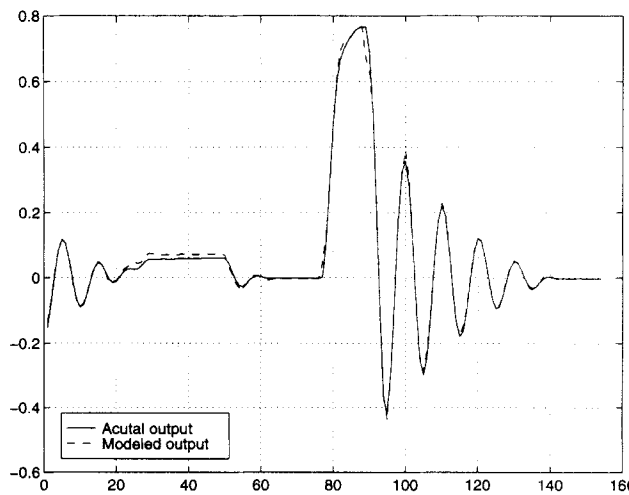

Figure 2. MARS-modeled one-step-ahead prediction for leg stimulation.

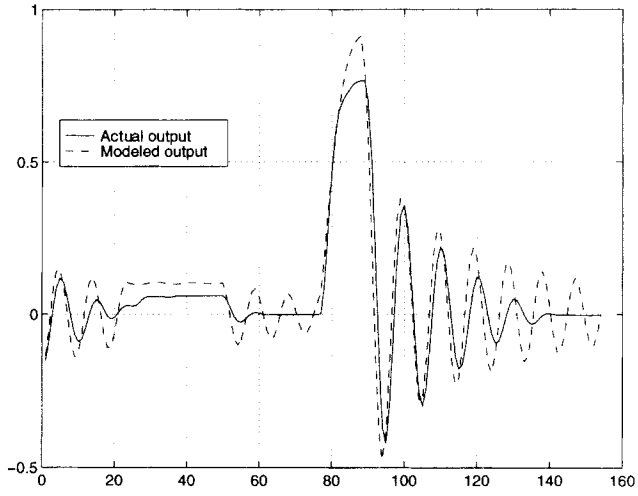

Figure 3. MARS-modeled simulation results for leg stimulation.

$$\dot{x}_4 = 245.978 x_1 \sqrt{x_2} - 10 x_4 \tag{31}$$

$$y = \frac{x_4}{x_3}. \tag{32}$$

The dimensionless state variable x_1 refers to the monomer concentration, x_2 to the initiator concentration, and x_4/x_3 is the number-average molecular weight (and also the output y). The input u is the dimensionless volumetric flow rate of the initiator.

Since a model of the system is known, large amounts of data can be collected for analysis. For this example, a time series of length 50,000 is generated by forcing the system with a uniformly distributed random input over the range 0.007 to 0.015, which is passed through a zero-order hold with a sampling time of 0.2. By driving the system with this input signal, an output that is roughly in the range of 26,000 to 34,000 is produced, which is the desired operating range of the system (Ogunnaike, personal communication, 1995).

The time series, with sampling time 0.2, is then normalized so that both the input and output signal have zero mean and a standard deviation of 1. The original FNN algorithm is then applied to the data, and the results are given in Table 5. By examining the results, it appears that a model of the form

$$y(t) = G[y(t-0.2),\, u(t-0.2),\, u(t-0.4)] \tag{33}$$

should give an accurate estimate of future outputs.

Using MARS a model of the dynamics is built using 2,000 training points. The results of one-step-ahead modeling and the pure simulation are both nearly identical to the actual

Table 5. FNN Results for Noise-Free Polymerization Data

Input Delays m	Output Delays l			
% FNN	0	1	2	3
0	100.0	99.8	94.1	68.1
1	99.7	62.4	0.1	0.0
2	66.6	0.0	0.0	0.0
3	0.1	0.0	0.0	0.0

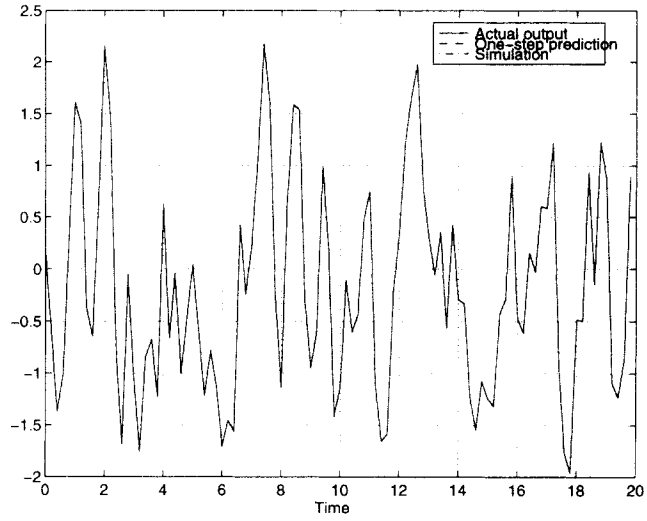

Figure 4. Results of MARS modeling for polymerization example ($l=1$, $m=2$).

output of the system. For prediction of one hundred points not contained in the training set, the MSPE (of the scaled data) is 7.7×10^{-5} for the simulated output and 3.3×10^{-5} for the one-step-ahead predicted output (see Figure 4). For comparative purposes, a MARS model, excluding the term $y(t-0.2)$, was built, and the MSPE error for the one-step-ahead prediction is 1.4×10^{-2}. The difference between the actual and predicted outputs is visible, and is illustrated in Figure 5.

To simulate a more realistic identification problem, normally distributed noise with a standard deviation of 0.1 was added to the output of the time series. The original FNN algorithm was applied again to this noise-corrupted time series, and the results are presented in Table 6. Notice that the percentage of FNN is no longer zero for one output and two input delays. The modified algorithm was then applied to the data with $\epsilon^y = 0.01$ and $\epsilon^u = 0$. The results of the modified algorithm are presented in Table 7. While the term ϵ^y is

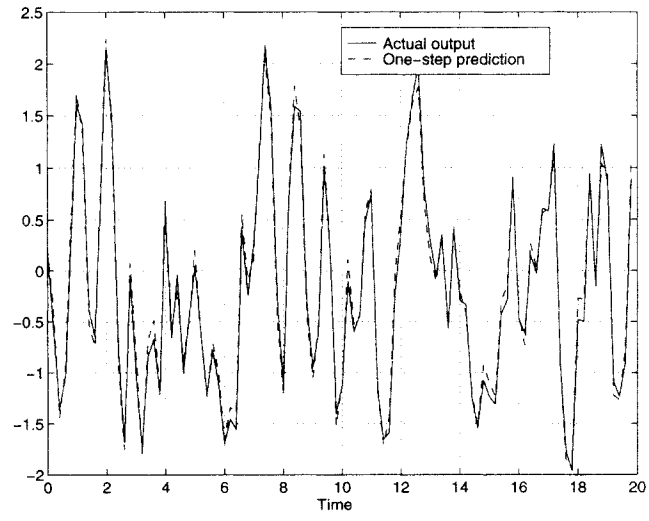

Figure 5. Results of MARS modeling for polymerization example ($l=0$, $m=2$).

Table 6. Original FNN Algorithm Results for Noise-Corrupted Polymerization Data

Input Delays m / % FNN	Output Delays l 0	1	2	3
0	100.0	99.8	94.2	67.5
1	99.7	71.4	3.8	0.1
2	73.4	4.4	0.1	0.0
3	2.4	0.1	0.0	0.0

Table 8. Modified FNN Algorithm Results for Noise-Corrupted Polymerization Data, $\epsilon^y = 0.2$

Input Delays m / % FNN	Output Delays l 0	1	2	3
0	100.0	72.8	58.9	33.1
1	95.4	3.8	0.0	0.0
2	60.5	0.0	0.0	0.0
3	1.4	0.0	0.0	0.0

much smaller than the size of the noise, the results of the algorithm still agree qualitatively with the results of the noise-free analysis. To show the sensitivity of the algorithm to the choice of parameters, the results of the FNN algorithm with $\epsilon^y = 0.02$ are shown in Table 8.

One question that arises naturally from this analysis, is why such a small regression vector is able to represent the dynamics of this system. Since the system has four states, a sufficient (but not necessary) condition for representing the dynamics is a regression vector that includes four delayed versions of the input and output. In this case, the reason is that the system has two states that are nearly unobservable, which leads naturally to a smaller description when analyzing the input/output dynamics. By linearizing the system about the operating point and performing a balanced realization, it is observed that two of the Hankel singular values are more than 2,000 times larger than the remaining singular values. This means that the input/output dynamics of the system can be represented well by a reduced system with only two states (Zhou et al., 1996). While this method is only valid for analysis of linear systems, it is not unexpected that the reduced characteristics of the system should carry over to the nonlinear system, making the smaller dynamical description predicted by the FNN algorithm feasible.

Simulated pulp digester

The third example consists of simulation data that come from a fundamental model of a pulp digester. The data were provided to us in a dimensionless form by Ferhan Kayihan and Marc Gelormino of Weyerhaeuser Corporate Research and Development in order to test our identification methods. The data consist of two inputs and two outputs. The time series also includes the effect of random unknown external disturbances to the simulated system. Since the values of these disturbances were not given to us, these disturbances can be thought of as unmeasured inputs to the system. For the generated data, the inputs and disturbances changed only at the sampling times of the system. The sampling time was chosen by the researchers at Weyerhaeuser to correspond roughly to the dominant time constant of the system. The time series consists of 417 observed sampling periods.

Table 7. Modified FNN Algorithm Results for Noise-Corrupted Polymerization Data, $\epsilon^y = 0.1$

Input Delays m / % FNN	Output Delays l 0	1	2	3
0	100.0	85.9	75.8	48.5
1	97.5	21.7	0.0	0.0
2	67.0	0.1	0.0	0.0
3	1.9	0.0	0.0	0.0

While the FNN algorithm and analysis presented previously only consider single input/single output systems (SISO), it is straightforward to extend the FNN algorithm for systems with multiple inputs and a single output (MISO). For a system with two inputs the regression vector must be extended, such that both inputs are included. The new prediction equation takes the following form

$$y(t) = G[y(t-\tau), \ldots, y(t-l\tau), u_1(t-\tau), \ldots, u_1(t-m_1\tau), u_2(t-\tau), \ldots, u_2(t-m_2\tau)], \quad (34)$$

where l, m_1, and m_2 are the number of delays that need to be determined by the FNN algorithm. By extending the regressor vector ψ of the original FNN algorithm to include the additional input term, the FNN algorithm can be used to search for the smallest number of delays in the output, and both input dimensions needed to recreate the dynamics of the system. Since the output of the system is not affected by stochastic noise, the original FNN algorithm is applied to these data.

For systems with multiple outputs, each individual output can be analyzed separately with the FNN algorithm. It is possible that different outputs of the same system may need different numbers of delayed terms to represent the dynamics. An example of this is presented in some of the original work on the application of FNN to autonomous chaotic systems (Abarbanel, 1996). In this work, it is shown that two different outputs of the same underlying chaotic system require a different number of delayed terms to predict the respective outputs.

For the first output, it was determined [using the average mutual information (Abarbanel et al., 1993)] that there is a time delay of approximately three sampling times before either of the two inputs affect the first output. This issue was examined, since the researchers at Weyerhaeuser informed us that there may be significant time delays associated with this process. Instead of using the standard regression vector, the following regression vector that accounts for the time delay is used:

$$\psi_{l,m_1,m_2}(t) = [y(t-\tau), \ldots, y(t-l\tau), u_1(t-3\tau), \ldots, u_1(t-(2+m_1)\tau), u_2(t-3\tau), \ldots, u_2(t-(2+m_2)\tau)]. \quad (35)$$

While the FNN algorithm could be used to determine that this time delay exists if sufficient data were available, since the amount of data is quite limited, this type of preliminary analysis is necessary for accurate prediction of the proper regression vector.

The results of the FNN analysis for a threshold $R = 10$ are given in Table 9. From the FNN algorithm, it appears that

Table 9. FNN Algorithm Results for Pulp Digester, Output 1

Output Delays l	Input Delays m_1	Input Delays m_2	% FNN
0	0	0	100.00
0	0	1	94.63
0	0	2	47.32
0	1	0	96.34
0	1	1	35.85
0	1	2	3.66
0	2	0	35.37
0	2	1	0
0	2	2	0
1	0	0	93.66
1	0	1	39.02
1	0	2	3.66
1	1	0	36.10
1	1	1	0.98
1	1	2	0
1	2	0	4.39
1	2	1	0
1	2	2	0
2	0	0	50.00
2	0	1	4.39
2	0	2	0
2	1	0	4.39
2	1	1	0
2	1	2	0
2	2	0	0
2	2	1	0
2	2	2	0

Table 10. FNN Algorithm Results for Pulp Digester, Output 2

Output Delays l	Input Delays m_1	Input Delays m_2	% FNN
0	0	0	100.00
0	0	1	95.12
0	0	2	30.73
0	1	0	97.07
0	1	1	32.44
0	1	2	1.46
0	2	0	44.88
0	2	1	1.71
0	2	2	0
1	0	0	95.85
1	0	1	37.07
1	0	2	3.17
1	1	0	48.29
1	1	1	2.44
1	1	2	0
1	2	0	6.34
1	2	1	0
1	2	2	0
2	0	0	48.78
2	0	1	3.17
2	0	2	0
2	1	0	8.29
2	1	1	0
2	1	2	0
2	2	0	0.73
2	2	1	0
2	2	2	0

the proper regression vector has $l = 0$, $m_1 = 2$, and $m_2 = 1$, or a representation of the form

$$y(t) = G[u_1(t-3),\ u_1(t-4),\ u_2(t-3)] \tag{36}$$

should be able to recreate the observed dynamics. To verify the results of the FNN algorithm, a MARS model was built using the first 372 points in the time series. The results of the modeling were then verified using the last 43 points of the time-series data. The results of the modeling using simulation from initial condition can be found in Figure 6. This model appears to be fairly accurate, especially considering that there are unmeasured disturbances to the system.

The results of FNN analysis for output two can be found in Table 10. For this output, there is no significant time delay between the input and output, and a model of the form

$$y(t) = G[y(t-\tau),\ u_1(t-\tau),\ u_1(t-2\tau),\ u_2(t-\tau)] \tag{37}$$

can be used to model the dynamics of the system. The results of MARS modeling for a simulation from initial conditions can be found in Figure 7.

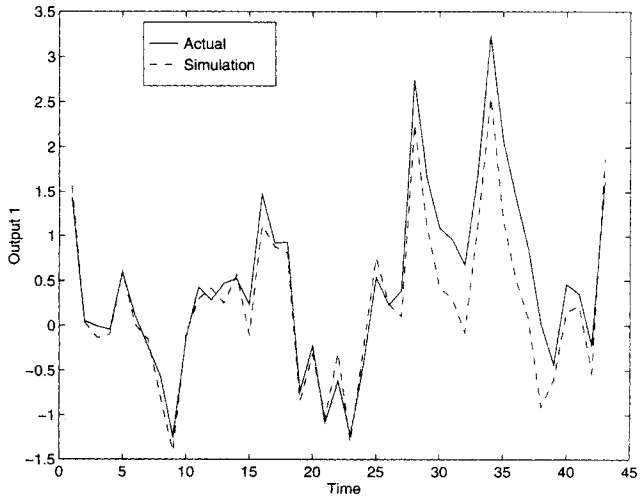

Figure 6. Results of MARS simulation for pulp-digester output 1.

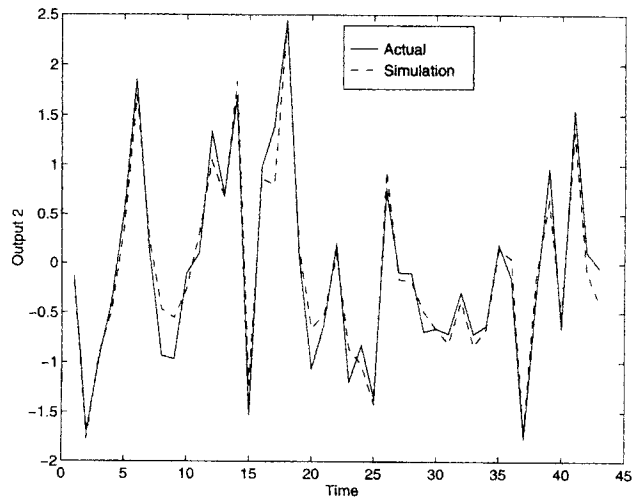

Figure 7. Results of MARS simulation for pulp-digester output 2.

FNN for Inferential Measurement Selection

In many processes in the chemical industry, it is infeasible (for economic or physical reasons) to physically measure the output to be controlled. However, in some of these cases, this output may be able to be determined or "inferred" from other outputs of the system. One example is composition control of a distillation column. The goal of control is often to hold the composition of the product streams constant, but composition analysis introduces a significant measurement delay into the system and the equipment is expensive and difficult to maintain (Mejdell and Skogestad, 1991). Since temperature measurements can be easily made along the column, the output compositions can be estimated using a number of temperature measurements along the column.

Since the temperature can be measured anywhere along the length of the column, the question of where the temperature should be measured for accurate estimation of the composition remains. For linear systems, this problem has been examined using a number of different methods including PLS (Mejdell and Skogestad, 1991), μ-analysis (Lee and Morari, 1991), singular-value analysis (Moore et al., 1987), Brosilow's selection scheme (Joseph and Brosilow, 1978), and measurement selection based on Kalman filters (Harris et al., 1980). While all these methods determine the optimal measurement location in some sense, they are all based on utilization of a linear estimator. To the authors' knowledge, the problem of determining suitable measurement locations for nonlinear inferential prediction has not been studied.

Theory

For systems with multiple outputs where one output is to be predicted as a function of the other outputs, there is an embedding theorem that is similar to the theorem presented in the second section. Consider a system of the form

$$\frac{d\boldsymbol{x}}{dt} = f(\boldsymbol{x}) \tag{38}$$

$$y^p = h(\boldsymbol{x}) \tag{39}$$

$$y_i^s = g_i(\boldsymbol{x}), \tag{40}$$

where $\boldsymbol{x}$ is again a state vector of dimension n; y^p is a scalar *primary* variable (the one to be estimated); and y_i^s are a number of *secondary* variables (those that can be measured). The theorem states that for the preceding system, a function F exists that can predict the primary output from secondary outputs

$$y^p(t) = F[y_{i_1}^s(t), y_{i_2}^s(t), \ldots, y_{i_l}^s(t)] \tag{41}$$

when the number of secondary outputs $l > 2n$ and the secondary outputs are "independent" (Sauer et al., 1991). While the original results are shown for systems with no external forcing, there are no problems in extending this theorem to deal with the preceding system, since the proof relies only on a diffeomorphic relationship that exists between the state space and the space of secondary outputs. This relationship is not affected by the presence of an external input.

Again this result is only a sufficient condition for the relationship to exist, so it may be conservative. In addition, it will be necessary to determine both the number of secondary outputs needed to predict the primary output as well as choosing which of the secondary outputs are best able to predict the primary output from process time-series data. In theory, any combination of independent measurements is able to predict the primary output. However, in practice some sets of measurements may do a better job of inferential prediction. To solve the problem of inferential measurement selection, the FNN algorithm can again be modified.

Applying the FNN algorithm

The FNN algorithm for inferential measurement selection is very similar to the original algorithm. However, in this case the regression vector consists of secondary outputs rather than delayed versions of the input and output. In addition, there is no easy way to determine a method for adding terms to the regressor, as in the input/output case. In fact, to be sure the optimal measurement set is chosen, all combinations of measurements in each dimension must be evaluated.

Here is an outline of the FNN algorithm for inferential measurement selection:

1. Identify the closest point to a given point in the regression space of secondary variables. For the regressor at time k

$$\Omega_{i_1,\ldots,i_l}(k) = [y_{i_1}^s(k), \ldots, y_{i_l}^s(k)] \tag{42}$$

find the regressor $\Omega_{i_1,\ldots,i_l}(j)$ in the data set such that the distance

$$d = \| \Omega_{i_1,\ldots,i_l}(k) - \Omega_{i_1,\ldots,i_l}(j) \|_2 \tag{43}$$

is minimized.

2. Determine if the neighbors are true or false using the following expression

$$\frac{|y^p(k) - y^p(j)|}{\| \Omega_{i_1,\ldots,i_l}(k) - \Omega_{i_1,\ldots,i_l}(j) \|_2} \leq R. \tag{44}$$

A modified version of the threshold test that accounts for noise can be used as well.

3. Repeat the algorithm for all times k in the data set and compute the percentage of false nearest neighbors.

4. Compute the percentage of false neighbors for all combinations of secondary measurements in the dimension l.

5. If the percentage of false nearest neighbors is large for all combinations of measurements in the given regression dimension l, increase the number of secondary measurements in the regressor vector.

For systems that require even a moderately sized regression vector, computing the percentage of false nearest neighbors for all combinations of secondary outputs may be too time-consuming. If the computational time is too large, the heuristic method of computing the number of false nearest neighbors for a given number of measurements, adding the single measurement that reduces the percentage of false neighbors in that dimension, and using this set of measurements to start the search in the next dimension may have to be used.

Table 11. Distillation Column Example

z_f	α	N_{trays}	N_f	D/F	L/F
0.5	1.5	40	21	0.5	2.706
			Antoine Parameters		
Comp. No.		T_b (K)	A	B	C
1		341.9	15.83660	2697.55	−48.78
2		355.4	15.43113	2697.55	−48.78

Example

For this example, inferential measurement selection will be performed for a binary distillation column. The column example here is column A taken from another study on distillation dynamics (Skogestad and Morari, 1988). The Antoine equation is used to determine the temperature of the mixture on each tray and uniformly distributed noise with magnitude 0.1°C was added to the temperatures of each tray. Due to the choice of the Antoine parameters, the mixture has constant relative volatility, and constant molar flows are also assumed. The specific parameters used to generate data for this example are given in Table 11.

The sample data consist of 500 randomly chosen steady states with varying feed composition z_f, distillate composition y_d, and bottoms composition x_b. For each steady state the output composition and temperature along each tray of the column is recorded. The range of variation in these variables is similar to that used in (Mejdell and Skogestad, 1991). The feed composition varies from 0.4 to 0.6, the distillate composition from 0.970 to 0.997, and the bottoms composition from 0.003 to 0.03. For all linear analysis, both the temperature and composition were scaled logarithmically, as is suggested in Mejdell and Skogestad (1991), to improve prediction. For nonlinear analysis, no scaling is performed.

The goal of the analysis is to determine the "optimal" two-tray locations to make temperature measurements for predicting the distillate composition y_d. By examining all combinations of two measurements, the FNN algorithm determines that the smallest number of false neighbors occurs when using trays 10 and 21. For comparison, a linearly based PLS measurement selection method (Mejdell and Skogestad, 1991) suggests using trays 2 and 21.

To compare the measurement selection methods, two different models were built to determine the distillate output from temperature measurements. One model was built using the MARS (nonlinear) modeling scheme, and the second model was built using a linear least-squares estimate. The results of the modeling for the two measurement-prediction sets are given in Table 12. The FNN-based measurement-selection scheme seems to reduce the amount of error for a nonlinear estimation scheme, while the PLS (linearly) -based selection scheme reduces the error of linear estimation. In addition, the modeling error was determined for all combinations of two-tray-based measurements using the MARS scheme, and a minimum error of 1.40×10^{-7} was found for trays 9 and 22. This is quite close to the amount of error found when using the trays suggested by the FNN analysis.

Table 12. MSPE for Different Prediction Schemes, Distillation Column

	Prediction Scheme	
Tray No.	Linear	MARS
10,21 (FNN scheme)	9.26×10^{-6}	1.57×10^{-7}
2,21 (PLS scheme)	5.09×10^{-6}	2.79×10^{-6}

Conclusions

By determining the smallest regression vector dimension that allows accurate prediction of the output, the FNN algorithm should reduce the overall computational effort needed to perform nonlinear identification. Instead of repeating a large number of identification experiments with different-sized regression vectors, FNN can determine the proper embedding dimension using relatively little computation time. After the proper dimension is determined, only a single nonlinear function needs to be estimated. For these reasons, an FNN-assisted identification scheme should save time when solving difficult nonlinear identification problems, since the proper number of delayed terms can be determined without fitting any specific models to the system.

To illustrate this fact, the FNN algorithm was applied to a 2,000-point data set from the polymerization example. The FNN algorithm was able to examine 16 different regression vectors (all combinations of 0 to 3 delayed inputs and outputs) in 8 s on a Sun SPARC station 20. For comparison, the same data were modeled using the MARS modeling scheme. With MARS it took 10 s to determine the proper nonlinear model for a single regression vector, and MARS is one of the faster methods of nonlinear modeling. To analyze a data set consisting of 50,000 points, the FNN algorithm took 6.5 min to analyze the same set of 16 regression vectors.

The FNN algorithm appears to be a useful tool for determining the proper embedding dimension for both input/output dynamics and inferential prediction. A number of examples illustrate the success of the FNN in determining the proper regression vector for accurate prediction. While the FNN algorithm is successful in determining the embedding dimension, it does not give any clues about the proper functional relationship between the regression vector and the output. Although this might appear to be a drawback, the main advantage of FNN is that the process of determining the proper regressor is not dependent on a single model structure. This is especially important in nonlinear systems, since a single model structure may not do a good job representing all possible relationships between regressors and outputs. Since FNN is not dependent on any single model structure, it can work in conjunction with any nonlinear functional modeling scheme and be a useful tool in the overall nonlinear identification process.

Acknowledgments

Partial support from the Department of Energy is gratefully acknowledged. The authors would also like to thank Jan Schultheiss for providing the leg stimulation data and Ferhan Kayihan and Marc Gelormino of Weyerhaeuser for providing the data from their fundamental pulp digester model simulation.

Literature Cited

Abarbanel, H. D. I., *Analysis of Observed Chaotic Data*, Institute for Nonlinear Science, Springer-Verlag, New York (1996).

Abarbanel, H. D. I., R. Brown, J. J. Sidorowich, and L. S. Tsimring, "The Analysis of Observed Chaotic Data in Physical Systems," *Rev. Mod. Phys.*, **65**, 1331 (1993).

Beiman, L., "Hinging Hyperplanes for Regression, Classification and Function Approximation," *IEEE Trans. Inform. Theory*, **IT-39**, 999 (1993).

Casdagli, M., "A Dynamical Systems Approach to Modeling Input-Output Systems," *Nonlinear Modeling and Forecasting*, M. Casdagli and S. Eubank, eds., Vol. XII of *A Proceedings Volume in the Santa Fe Institute Studies in the Sciences of Complexity*, Addison-Wesley, Reading, MA, p. 265 (1992).

Chen, S., and S. A. Billings, "Representations of Non-linear Systems: The NARMAX Model," *Int. J. Control*, **49**(3), 1013 (1989).

del Re, L., F. Kraus, J. Schultheiss, and H. Gerber, "Self-Tuning PID Controller for Lower-Limb FES with Nonlinear Compensation," *Proc. Amer. Control Conf.*, Baltimore, MD, Vol. 2, published by American Automatic Control Council, p. 2015 (1994).

Doyle, F. J., B. A. Ogunnaike, and R. K. Pearson, "Nonlinear Model-Based Control Using Second-Order Volterra Models," *Automatica*, **31**, 697 (1995).

Finesso, L., *Consistant Estimation of the Order for Markov and Hidden Markov Chains*, PhD Thesis, Univ. of Maryland, College Park (1990).

Franklin, G. F., J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, Addison-Wesley, Reading, MA (1986).

Fredkin, D. R., and J. A. Rice, "Method of False Nearest Neighbors —a Cautionary Note," *Phys. Rev. E*, **51**, 2950 (1995).

Friedman, J. H., "Multivariate Adaptive Regression Splines," *Ann. Stat.*, **19**(1), 1 (1991).

Guillemin, V., and A. Pollack, *Differential Topology*, Prentice Hall, Englewood Cliffs, NJ (1974).

Harris, T. J., J. F. MacGregor, and J. D. Wright, "Optimal Sensor Location with an Application to a Packed Bed Tubular Reactor," *AIChE J.*, **26**(6), 910 (1980).

Joseph, B., and C. B. Brosilow, "Inferential Control of Processes: I. Steady State Analysis and Design," *AIChE J.*, **24**(3), 485 (1978).

Kennel, M. B., R. Brown, and H. D. I. Abarbanel, "Determining Embedding Dimension for Phase Space Reconstruction Using a Geometrical Reconstruction," *Phys. Rev. A*, **45**, 3403 (1992).

Lee, J. H., and M. Morari, "Robust Measurement Selection," *Automatica*, **27**, 519 (1991).

Mejdell, T., and S. Skogestad, "Estimation of Distillation Compositions from Multiple Temperature Measurements Using Partial-Least-Squares Regression," *Ind. Eng. Chem. Res.*, **30**, 2543 (1991).

Moore, C., J. Hackney, and D. Canter, "Selecting Sensor Location and Type for Multivariable Processes," *Shell Process Control Workshop*, D. M. Prett and M. Morari, eds., Butterworth, Boston, p. 291 (1987).

Poncet, A., and G. S. Moschytz, "Optimal Order for Signal and System Modeling," *Proc. IEEE Int. Symp. on Circuits and Systems*, Vol. 5, London, p. 221 (1994).

Poncet, A., J. L. Poncet, and G. S. Moschytz, "On the Input-Output Approximation of Nonlinear Systems," *Proc. IEEE Int. Symp. on Circuits and Systems*, Vol. 2, Seattle, WA, 1001 (1995).

Rhodes, C., and M. Morari, "False Nearest Neighbors Algorithm and Noise Corrupted Time Series," *Phys. Rev. E*, **55**, 6162 (1997).

Sauer, T., J. A. Yorke, and M. Casdagli, "Embedology," *J. Stat. Phys.*, **65**, 579 (1991).

Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Glorennec, H. Hjalmarsson, and A. Juditsky, "Nonlinear Black-Box Modeling in System Identification: A Unified Overview, *Automatica*, **31**(12), 1691 (1995).

Skogestad, S., and M. Morari, "Understanding the Dynamic Behavior of Distillation Columns," *Ind. Eng. Chem. Res.*, **27**, 1848 (1988).

Söderström, T., and P. Stoica, *System Identification*, Prentice Hall Int. Ser. in Systems and Control Eng., Prentice Hall, Englewood Cliffs, NJ (1989).

Takens, F., "Detecting Strange Attractors in Turbulence," *Dynamical Systems and Turbulence, Warwick 1980*, D. A. Rand and L. S. Young, eds., No. 898 in *Lecture Notes in Mathematics*, Springer-Verlag, New York, p. 366 (1981).

Weigend, A. S., and N. A. Gershenfeld, eds., *Time Series Prediction: Forecasting the Future and Understanding the Past*, Santa Fe Institute Studies in Complexity, Addison-Wesley, Reading, MA (1994).

Zhou, K., J. C. Doyle, and K. Glover, *Robust and Optimal Control*, Prentice Hall, Englewood Cliffs, NJ (1996).

Manuscript received Jan. 13, 1997, and revision received July 30, 1997.