Control of Nonlinear Systems Using Polynomial ARMA Models

Evelio Hernández and Yaman Arkun
School of Chemical Engineering, Georgia Institute of Technology, Atlanta, GA 30332

Most of the advanced nonlinear control algorithms require a model of the system to be controlled. Unfortunately, most of the processes in the chemical industry are nonlinear, and fundamental models describing them are lacking. Thus there is a need for the identification and control of nonlinear systems through available input-output data. In this article, we briefly introduce the input-output model used (polynomial ARMA models), and analyze its stability and invertibility. This paves the way to the development of a nonlinear-model-predictive controller. Implementation issues such as modeling of disturbance, state and parameter estimation are discussed. The theory presented is illustrated through examples.

Introduction

Most of the processes in the chemical industry are nonlinear in nature, yet not many of these processes are adequately described by models derived from first principles. Thus, it becomes imperative to model them from available input-output data. The overall goal of this article is to establish a methodology that provides the user with theory and practical procedures for the control of nonlinear processes based on models identified from input-output data.

The first step toward a nonlinear control scheme is to obtain an estimate of the nonlinear system's dynamics. This is achieved by identifying the system's input-output map with a nonlinear model structure. The particular structure selected in this work is the polynomial ARMA structure which is presented in the next section. The discussion places emphasis on the analysis of the polynomial ARMA models instead of the identification procedure involved. The main contributions of this article are the use of the identified polynomial ARMA models obtained for nonlinear control purposes. It shows how to construct the nonlinear inverse of the polynomial ARMA models and uses this inverse within a control strategy. The limitations of the inverse model approach are observed, and connections are established with the feedback linearization of discrete-time systems.

Once the limitations of the inverse model approach are understood, the control scheme is improved by relaxing the inverse of the polynomial ARMA models. The controller design is obtained by relaxing the inverse from a performance point of view. Instead of requiring the plant's output to match a particular desired value at the next sampling instant, a more relaxed performance is specified. This gives rise to extended horizon controllers and to the more general model-predictive control family of controllers. In addition, controller implementation is detailed including discussions on disturbance estimation, model predictions into the future, state estimation, and parameter estimation. It also presents illustrative examples of the above-mentioned theory.

Most of the material discussed in this article is presented from a single-input-single-output (SISO) point of view for simplicity in exposition rather than limitations in the theory. We will inject comments throughout the article to address the generalizations of the presented results to the multivariable case.

Nonlinear Model Structure

In the case of discrete time linear systems, the general relationship between the input and output of a SISO system can be modeled as:

$$y(k)=\frac{A(q)}{B(q)}u(k) \tag{1}$$

where $A(\cdot)$ and $B(\cdot)$ are polynomial functions of the shift operator, q; $u(k)$ and $y(k)$ are the input and output signals, respectively. Unfortunately, a similar model structure covering any general nonlinear system would have to be more of an abstract idea than a practical tool. It is for this reason that

Correspondence concerning this article should be addressed to Y. Arkun.

there are numerous model structures proposed for the identification of nonlinear systems. Among these we can mention Volterra series models, block-oriented models such as the Hammerstein and Wiener models, bilinear models, state affine models, neural networks, and so on. For a recent review on nonlinear model structures, see Haber and Unbehauen (1990). The model selected for this study has a polynomial ARMA structure as discussed, for example, in Korenberg et al. (1988).

Assume that $u(k-\varrho)$ is the first input to have an effect on the current output, $y(k)$. Then, we could use a polynomial model to express $y(k)$ as a function of previous outputs, and inputs prior to and including time $k-\varrho$:

$$y(k)=\theta_0+\sum_{i=1}^{n_y}\theta_{1,i}y(k-i)+\sum_{i=\varrho}^{n_u}\theta_{2,i}u(k-i)+\sum_{i=1}^{n_y}\sum_{j=1}^{i}\theta_{3,(i,j)}\,y(k-i)\,y(k-j)+\sum_{i=\varrho}^{n_u}\sum_{j=\varrho}^{i}\theta_{4,(i,j)}\times u(k-i)u(k-j)+\sum_{i=1}^{n_y}\sum_{j=\varrho}^{n_u}\theta_{5,(i,j)}\,y(k-i)\,u(k-j)+\ldots \tag{2}$$

or simply

$$y(k)=p_n[y(k-1),\ \ldots,\ y(k-n_y),\ u(k-\varrho),\ \ldots,\ u(k-n_u)] \tag{3}$$

where n is the order of the polynomial in Eq. 2. If the system to be modeled is of a multivariable nature, then Eq. 2 can be extended in a straightforward fashion by including on the righthand side the appropriate monomials in the additional input and output signals.

This type of model has been used by several authors for the modeling of simulated and actual nonlinear systems (for example, Korenberg et al., 1988). The structure has also been justified from a theoretical point of view by using the Stone-Weierstrass theorem. In particular, Díaz and Desrochers (1988) have formulated the following result: "Every input-output map in which the (discrete) output is a continuous function of the (discrete, bounded) input can be approximated arbitrarily well, over a finite period of time, by a system satisfying a regression-type polynomial Eq. 3, except in the neighborhood of a finite number of points."

In essence the theorem of Díaz and Desrochers states that with reasonable assumptions on the inputs and the time of validity of the approximation, any discrete input-output map can be approximated with a polynomial ARMA model that has the structure of Eq. 3.

Theoretical justifications similar to the the one given above are available for many other types of nonlinear model structures. For example, in the case of neutral networks, results by Cybenko (1989) assert that any continuous function restricted to a bounded space can be approximated with a neural network model if sigmoids are used for the activation function. Furthermore, there are numerous results available for both the continuous and discrete Volterra series (for example, Boyd and Chua, 1985). Thus, although theoretical justifications are important, they should not be the only consideration when selecting a particular model structure. For this reason, it is necessary to justify the use of polynomial models from a more practical point of view.

The main advantage of polynomial models is that their one-step-ahead prediction can be formulated as a linear regression. This greatly simplifies the estimation of the parameters from input-output data. To see this, define the following regressors:

$$\begin{aligned} v_1(k)&=y(k-1)\\ v_2(k)&=y(k-2)\\ &\vdots\\ v_{n_y+1}&=u(k-1)\\ &\vdots\\ v_{n_u+n_y}+1(k)&=u(k-n_u)\\ v_{n_u+n_y}+2(k)&=y(k-1)y(k-1)\\ &\vdots \end{aligned} \tag{4}$$

Then, define the signal vector:

$$\underline{v}^T(k)=[1\ v_1(k)\ v_2(k)\ \ldots\ v_{N_T}(k)]$$

and the parameter vector:

$$\underline{\theta}^T=[\theta_0\ \theta_{1,1}\ \theta_{1,2}\ \ldots\ \theta_{2,1}\ \ldots]$$

Then the model in Eq. 3 can be written as:

$$\hat{y}(k)=\underline{v}^T(k)\underline{\theta}+\epsilon(k) \tag{5}$$

which is a linear regression. Other models, such as neural networks, cannot be casted in the same way and thus nonlinear optimizations have to be performed to identify the parameters.

Moreover, polynomial models are superior to Volterra series models in the sense that the number of parameters needed to approximate a system is generally much less with polynomial models. This is due to the fact that the model in Eq. 3 includes previous outputs as well as previous inputs, while Volterra models only include previous inputs. Futhermore, although the Hammerstein structure is amenable to controller design and handling process noise, we will use polynomials for the added generality in modeling systems. For example, a Hammerstein model is incapable of modeling systems that exhibit output multiplicities as displayed by various chemical systems.

The added generality offered by polynomial models is not without a price since the number of possible terms (and thus parameters) in Eq. 3 is much larger than those in other simpler models such as the Hammerstein. This disadvantage, however, can be overcome by using statistical-model-building algorithms to select the important terms in the model. The backbone of such an algorithm is composed of the following steps (for a more detailed algorithm and its implementation on chemical engineering examples, see Kortmann and Unbehauen, 1988; Hernández, 1992):

1. Evaluate all the possible $v_i(k)$ of Eq. 4 to be included in the model.
2. Select the term that is most correlated with the output.
3. Check if the model has improved. If not, then go to step 6.

4. Check if any of the terms included so far in the model should be deleted. Delete any such term.

5. Select the term of those remaining that can best "explain" the unaccounted structure remaining in the residuals. Go to step 3.

6. End.

A shortcoming commonly associated with polynomial models is their tendency toward exhibiting unbounded behavior. Various researchers have cited this disadvantage as the reasoning for their choice of other model structures such as neural networks or radial basis function models (Chen et al., 1990, 1991; Pottmann and Seborg, 1991). Both of these model structures generally guarantee boundedness of the outputs by their use of sigmoid and Gaussian functions. Because these models maintain their output predictions bounded, they are credited with better performance. Unfortunately, however, these other models are, in general, highly nonlinear in the parameters to be identified, and thus the parameter identification problem becomes quite a challenge.

The nonlinear parameter estimation problem can be avoided while guaranteeing bounded behavior by using polynomial models with simple modifications. In particular, consider the model constructed from the composition of an invertible, bounded nonlinear function and a polynomial function of previous inputs and outputs:

$$y(k+1)=\sigma\{p_n\,[y(k),\ \ldots\ y(k-n_y),\ u(k-\varrho+1),\ \ldots\ u(k-n_u)]\} \quad (6)$$

where, for example, $\sigma(x)$ can be given by:

$$\sigma(x)=\frac{1}{1+\exp(-x)} \quad (7)$$

which is the sigmoid function commonly used in neural network models. This sigmoid function is continuous, invertible and bounded between 0 and 1.

The identification problem can still be posed as one linear in the parameters by applying the inverse of the sigmoid operator to both sides of Eq. 6:

$$y^{ir}(k)=\ln\left[\frac{y(k+1)}{1-y(k+1)}\right]=p_n[y(k),\ \ldots,\ y(k-n_y),\ u(k-\varrho+1),\ \ldots,\ u(k-n_u)] \quad (8)$$

And so the problem of modeling $y^{ir}(k)$ as a polynomial function of previous inputs and outputs is again formulated as a linear regression. This time assume that the residual between $y^{ir}(k)$ and $p_n(\cdot)$ is a zero mean, Gaussian process. The results and discussions to be presented in the rest of the article are applicable to both polynomial models and models of the type shown in Eq. 6, but will be presented for polynomial models for the sake of simplicity.

Other important issues of identification of polynomial models are discussed elsewhere (Hernández, 1992). The goals of this article are directed toward the control of nonlinear systems using polynomial models. Thus, in the sequel we will assume that the polynomial models are already identified.

State space realization

To satisfy the model analysis objectives, it is necessary to construct a state space realization of the input-output model in Eq. 3. This can be done by using the delay coordinate method. For this let:

$$x_1^a(k)=y(k) \qquad x_1^b(k)=u(k-1)$$
$$\vdots \qquad\qquad \vdots$$
$$x_{n_y+1}^a(k)=y(k-n_y) \qquad x_{n_u}^b(k)=u(k-n_u)$$

or, in the multivariable case, let:

$$x_i^a(k)=\begin{bmatrix} y_1(k-i+1) \\ \vdots \\ y_p(k-i+1)\end{bmatrix};\quad x_j^b(k)=\begin{bmatrix} u_1(k-j) \\ \vdots \\ u_q(k-j)\end{bmatrix}$$
$$i=1,\ \ldots,\ n_y+1; \qquad j=1,\ \ldots,\ n_u$$

where p and q are the number of outputs and inputs, respectively. Using the above definition, the state of the system could be constructed in a direct fashion as:

$$\underline{x}^a(k)=[x_1^a(k),\ \ldots,\ x_{n_y+1}^a(k)]^T$$
$$\underline{x}^b(k)=[x_1^b(k),\ \ldots,\ x_{n_u}^b(k)]^T$$
$$\underline{x}(k)=\begin{bmatrix}\underline{x}^a(k) \\ \underline{x}^b(k)\end{bmatrix} \quad (9)$$

Note that we have clearly divided the state of the system into two components: the output, $\underline{x}^a$, and the input, $\underline{x}^b$, substates. We call the dynamics of $\underline{x}^a$ the "output subsystem." Similarly, the dynamics of $\underline{x}^b$ are denoted as the "input subsystem." Given the definition of the state variables in Eq. 9, we can write the realization of Eq. 3 as (only the SISO realization is given):

$$\begin{bmatrix}\underline{x}^a(k+1) \\ \underline{x}^b(k+1)\end{bmatrix}=\begin{bmatrix} 0 & \cdots & 0 & 0 & & & & \\ 1 & & & 0 & & & & \\ & \ddots & & \vdots & & & & \\ & & 1 & 0 & & & & \\ & & & & 0 & \cdots & 0 & 0 \\ & & & & 1 & & & 0 \\ & & & & & \ddots & & \vdots \\ & & & & & & 1 & 0\end{bmatrix}\begin{bmatrix}\underline{x}^a(k) \\ \underline{x}^b(k)\end{bmatrix}+\begin{bmatrix} p_n\,[x(k),\ u(k)] \\ 0 \\ \vdots \\ 0 \\ u(k) \\ 0 \\ \vdots \\ 0\end{bmatrix} \quad (10)$$

$$y(k)=[1\ 0\ \cdots\ 0\ |0\ \cdots\ 0]\begin{bmatrix}\underline{x}^a(k) \\ \underline{x}^b(k)\end{bmatrix}$$

or

$$\underline{x}(k+1)=F[\underline{x}(k),\ u(k)]$$

$$y(k)=h[\underline{x}(k)]$$

Note that in Eq. 10 we have implicitly defined the state transition map, $F(\underline{x},\ u)$ and the output map $h(\underline{x})$. Also observe that in defining the above system we have stretched the notation and allowed for:

$$y(k+1)=p_n[y(k),\ \ldots,\ y(k-n_y),$$
$$u(k-\varrho+1),\ \ldots,\ u(k-n_u)]=p_n[\underline{x}(k),\ u(k)]$$

Using this state-space representation of the input-output model, we are able to investigate control-oriented properties of the model and consequently various nonlinear control algorithms. For example, one of the most basic, yet important, properties is the stability of the equilibria. Conditions for the global stability of the equilibrium point of a general polynomial model are not available. Nevertheless, conditions for local stability can be easily found, computed, and do yield valuable insight.

A point $(\underline{x}^*,\ u^*)$ is an equilibrium point of Eq. 10 if $\underline{x}^*=F(\underline{x}^*,\ (u^*)$. An equilibrium point is called an attractor of Eq. 10 if the derivative of $\underline{x}(k+1)$ with respect to $\underline{x}(k)$ evaluated at the point $(\underline{x}^*,\ u^*)$ has eigenvalues inside the unit circle (Devaney, 1989). If an equilibrium point is an attractor, then there is a neighborhood of $(\underline{x}^*,\ u^*)$ in which all points tend to $(\underline{x}^*,\ u^*)$ under the forward iteration of the system of Eq. 10 (Devaney, 1989). It can also be shown that if an equilibrium point is an attractor, then it is also stable in the sense of Lyapunov (Kalman and Bertram, 1960). From this, it is clear that the local stability of the system about the equilibrium point $(\underline{x}^*,\ u^*)$ is determined by the eigenvalues of J, the Jacobian of $\underline{x}(k+1)$ with respect to $\underline{x}(k)$. That is:

$$J=\begin{bmatrix} \alpha_1 & \cdots & \alpha_{n_y} & \alpha_{n_y+1} & \beta_1 & \cdots & \beta_{n_u-1} & \beta_{n_u} \\ 1 & & & 0 & 0 & & \cdots & 0 \\ & \ddots & & \vdots & \vdots & & & \vdots \\ & & 1 & 0 & 0 & & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \cdots & & 0 & 1 & & & 0 \\ \vdots & & & \vdots & & \ddots & & \vdots \\ 0 & \cdots & & 0 & & & 1 & 0 \end{bmatrix} \tag{11}$$

where

$$\alpha_i=\left.\frac{\partial p_n[\underline{x}(k),\ u(k)]}{\partial x_i^a(k)}\right|_{\substack{\underline{x}=\underline{x}^* \\ u=u^*}};\quad \beta_j=\left.\frac{\partial p_n[\underline{x}(k),\ u(k)]}{\partial x_j^b(k)}\right|_{\substack{\underline{x}=\underline{x}^* \\ u=u^*}}$$

The discussion provided above leads to the following result:

Theorem 2.1. An equilibrium point $(\underline{x}^*,\ u^*)$ of the dynamical system of Eq. 10 is locally stable in the sense of Lyapunov if the eigenvalues of J_Y are contained inside the unit circle, where J_Y is given by:

$$J_Y=\begin{bmatrix} \alpha_1 & & \alpha_{n_y} & \alpha_{n_y+1} \\ 1 & & & 0 \\ & \ddots & & \vdots \\ & & 1 & 0 \end{bmatrix} \tag{12}$$

Proof. This theorem immediately follows from the fact that according to Eq. 11, the eigenvalues of J are: 0 (with multiplicity n_u) and the eigenvalues of J_Y, given by the derivative of $\underline{x}^a(k+1)$ with respect to $\underline{x}^a(k)$. Similar stability conditions could be derived for multivariable systems once the state equations are properly constructed. □

Invertibility and Connections to Feedback Linearization

We now consider the question of invertibility of a nonlinear dynamical system described by Eq. 10. That is, when can we find an input so that when applied to the system, it produces a particular desired output? This question has clear importance in control since if the answer was positive, then the control problem is solved. An input with this property, however, can rarely be found due to lack of perfect knowledge of the system or certain inherent system limitations. Nevertheless, it is a good starting point for the discussion on the design of control schemes.

As mentioned above, the problem of invertibility is to find an input signal, $u(k)$, so that the earliest affected output, $y(k+\varrho)$ for some $\varrho>0$, remains at a desired value, $r^*(k)$, for all $k=1,\ 2,\ \ldots$. Note that this is the best achievable performance. Interestingly, the calculations of the inverse controller are equivalent to those of the input-output linearization of discrete systems. To parallel these results, we adopt the notation of Monaco and Normand-Cyrot (1988). The derivation here presented is solely for SISO systems, the derivation for MIMO systems follows the same ideas, but it is slightly more complex. We will denote by ' o ' the usual composition of functions, and let:

$$H_0=h(\underline{x});\quad H_j=H_{j-1}\circ F(\underline{x},\ u)\quad j\geq 1 \tag{13}$$

where, recall that $h(\underline{x})$ and $F(\underline{x},\ u)$ have been defined in Eq. 10. The inverse controller is thus given implicitly by:

$$H_\varrho[\underline{x},\ u(k)]=y(k+\varrho)=r^* \tag{14}$$

The Implicit Function Theorem guarantees that Eq. 14 can be solved for $u(k)$ locally if:

$$\left.\frac{\partial H_\varrho[\underline{x}(k),\ u(k)]}{\partial u(k)}=\frac{\partial p_n[\underline{x}(k),\ u(k)]}{\partial u(k)}\right|_{\substack{\underline{x}=\underline{x}^* \\ u=u^*}}\neq 0 \tag{15}$$

Let the solution of Eq. 14 be given by:

$$u(k)=x_1^b(k+1)=g[\underline{x}(k),\ r^*] \tag{16}$$

To facilitate the connections between the inverse controller and the feedback linearization of discrete systems, it is useful to define the following coordinate transformation:

$$v_i^a(k) = H_{\varrho-i}[\underline{x}(k), u(k)] = y(k+\varrho-i);\ i=1, \ldots, n_y$$

$$v_i^b(k) = x_i^b(k); \qquad i=1, \ldots, n_u \quad (17)$$

Using the feedback law of Eq. 16 and the coordinate transformation defined in Eq. 17, the dynamical system of Eq. 10 is transformed to the following closed-loop system:

$$\begin{bmatrix} \underline{v}^a(k+1) \\ \underline{v}^b(k+1) \end{bmatrix} = \begin{bmatrix} 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & & & 0 & 0 & & \cdots & 0 \\ & \ddots & & \vdots & \vdots & & & \vdots \\ & & 1 & 0 & 0 & & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \cdots & & 0 & 1 & & & 0 \\ \vdots & & & \vdots & & \ddots & & \vdots \\ 0 & \cdots & & 0 & & & 1 & 0 \end{bmatrix} \begin{bmatrix} \underline{v}^a(k) \\ \underline{v}^b(k) \end{bmatrix} + \begin{bmatrix} r^* \\ 0 \\ \vdots \\ 0 \\ g[\underline{v}(k), r^*] \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (18)$$

$$y(k+\varrho-1) = [1\ 0\ \cdots\ 0 | 0\ \cdots\ 0] \begin{bmatrix} \underline{v}^a(k) \\ \underline{v}^b(k) \end{bmatrix}$$

Some similarities between the inverse problem and input-output linearization are apparent from Eq. 18. Note that the dynamics between the reference input, r^*, and the output $y(k)$ are linear; in fact they are given by: $y(z) = z^{-\varrho} r^*(z)$, z being the $\mathcal{Z}$ transform variable. From this we can deduce that the inverse problem is a special case of the input-output linearization of nonlinear discrete systems. Note that in input-output linearization, the output at time $k+\varrho$ is not necessarily set to r^* but to:

$$y(k) = \underline{c}^a \underline{v}^a(k) + r(k)$$

where $\underline{c}^a$ is a vector selected by the designer:

$$\underline{c}^a = [c_1^a\ c_2^a\ \ldots\ c_{n_y+1}^a]$$

which yields the following closed-loop transfer function:

$$y(z) = \frac{z^{-\varrho}}{(1 - c_1^a z^{-1} - \ldots - c_{n_y+1}^a z^{-(n_y+1)})} r(z) \quad (19)$$

Thus the location of the poles of the "output subsystem" are given by the selected value of $\underline{c}^a$. In the case of the inverse controller, all the poles of the "output subsystem" are located at zero.

Further similarities between the two problems are revealed if we study the stability of the closed loop system. From earlier discussions we can derive stability conditions of the closed-loop system in Eq. 18. Again, the local stability of this (closed-loop) system is determined by the eigenvalues of the Jacobian of $\underline{v}(k+1)$ with respect to $\underline{v}(k)$. In order to calculate this Jacobian we would either have to obtain $g[\underline{v}(k), r^*]$ analytically or differentiate equation (Eq. 14) implicitly. The latter choice is usually selected for its simplicity, since there may be not be an analytical solution of $g[\underline{v}(k), r^*]$. The Jacobian of this system becomes:

$$J' = \begin{bmatrix} 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & & & 0 & 0 & & \cdots & 0 \\ & \ddots & & \vdots & \vdots & & & \vdots \\ & & 1 & 0 & 0 & & \cdots & 0 \\ \alpha_1' & \cdots & \alpha_{n_y}' & \alpha_{n_y+1}' & \beta_1' & \cdots & \beta_{n_u-1}' & \beta_{n_u}' \\ 0 & \cdots & & 0 & 1 & & & 0 \\ \vdots & & & \vdots & & \ddots & & \vdots \\ 0 & \cdots & & 0 & & & 1 & 0 \end{bmatrix} \quad (20)$$

where

$$\alpha_i' = \left.\frac{\partial g(\underline{v}(k), r^*)}{\partial v_i^a(k)}\right|_{\substack{\underline{v}=\underline{v}^* \\ \underline{u}=\underline{u}^*}}; \quad \beta_j' = \left.\frac{\partial g(\underline{v}(k), r^*)}{\partial v_j^b(k)}\right|_{\substack{\underline{v}=\underline{v}^* \\ \underline{u}=\underline{u}^*}}$$

We can thus state the following theorem.

Theorem 3.1. An equilibrium point $(\underline{v}^*, u^*)$ of the dynamical system of Eq. 18 is locally stable in the sense of Lyapunov if the eigenvalues of J_U (the Jacobian of the "input subsystem") are contained inside the unit circle, where, J_U is given by:

$$J_U = \begin{bmatrix} \beta_1' & \cdots & \beta_{n_u-1}' & \beta_{n_u}' \\ 1 & & & 0 \\ & \ddots & & \vdots \\ & & 1 & 0 \end{bmatrix} \quad (21)$$

Proof. This theorem immediately follows from the fact that according to Eq. 20, the eigenvalues of J' are: 0 (with multiplicity n_y+1) and the eigenvalues of J_U, given by the derivative of $\underline{v}^b(k+1)$ with respect to $\underline{v}^b(k)$. □

Observe that the local stability of the system depends on the dynamics of the system remaining once the output is fixed to r^*. In the case where $\underline{v}^a$ is set to zero, the dynamics remaining are equivalent to the "zero dynamics" of discrete systems. This follows from the intuitive idea of the "dynamics remaining once the output is set to zero" as well as from the rigorous definition provided by Monaco and Normand-Cyrot (1988, Definition 2.4). This definition also states that the system of Eq. 10 is locally "minimum phase" if the zero dynamics are locally stable, or equivalently if the eigenvalues of J_U are inside the unit circle. Thus we confirm the familiar concept that the inverse controller stabilizes the closed-loop system if the open-loop system is minimum phase.

Remark. It is important to point out that Definition 2.4 of the Monaco and Normand-Cyrot article is based on the assumption that for any $\underline{v}$ in an open subset containing the equilibrium point, there exists an input $\tilde{u}(\underline{v})$ such that the

output obtained $\varrho+1$ steps into the future is equal to $r^*=0$. This assumption, of course, is satisfied from the very construction of the inverse controller. If the inverse controller does not exist, then this entire discussion of zero dynamics becomes irrelevant.

Note that the inverse controller obtained is of an output deadbeat nature, and thus it is likely not to be a very good controller from a practical viewpoint. Usually, deadbeat controllers yield input signals that are too large, and in some cases, as illustrated by the theory, the closed-loop system might become unstable. Nevertheless, the study of the inverse controller has given us insight on the performance limitations of certain systems. In particular, in the next section we will study other controller design methods that originate from the inverse controller, yet display better behavior.

Controller Design Method

In this section we will formulate the controller design method by "relaxing" the inverse controller from an optimal point of view. In this fashion we will obtain controllers that are more practical and applicable to a wider range of systems. We will assume nominal conditions in the design of the controllers presented here. This will isolate the basic ideas behind the controller design from other implementation issues such as filtering of measurement noise and so on. These will be addressed in the later section. To relax the inverse from an optimal point of view, it is advantageous to look at the inverse as the function which solves the following optimization problem:

$$\min_{u(k)}[y(k+\varrho)-r^*]^2 \tag{22}$$

That is, the difference between the output and the setpoint at the next sampling time is minimized by choosing the appropriate input. Consider now a different optimization problem, one of minimizing the difference between the output and the setpoint at sometime into the future, say $k+P$. That is:

$$\min_{u(k)}\ [y(k+P)-r^*]^2 \tag{23}$$

The input in this case is selected so that $u(k)=u(k+1)=\ldots=u(k+P-1)=\xi$.

In actual implementation of this algorithm, the input $u(k)$ is applied and the entire problem is solved once more at the next sampling time. This type of control algorithm gives an added flexibility. The tuning parameter in this case is P. If P equals ϱ, the algorithm reduces to the inverse controller. As P increases, the required performance is more relaxed, and so the algorithm produces better behaved input signals in general. Hernández and Arkun (1992) have provided conditions for the local stability of the closed-loop system when this type of algorithm is used. The idea of the "P-Inverse or Extended Horizon controller" described above can be generalized to include a model predictive type of controller. In this case the objective function is modified to consider more than one output into the future and changes in the input as well:

$$\min_{\Delta u}\sum_{i=1}^{P}\gamma_i[y(k+i)-r^*(k+i)]^2+\sum_{j=0}^{M-1}\lambda_j\Delta u^2(k+j) \tag{24}$$

where γ_i and λ_j are weights on the outputs and inputs, respectively; $y(k+i)$ are the output predictions as given by the input-output model, and $\Delta u(k+j)$ is the change in the inputs at time $k+j$. As usual, only the first input is implemented and the entire calculation is repeated at the next sampling time. The optimization problem formulated in Eq. 24 requires the solution of a nonlinear program. The simulations shown later are obtained by solving Eq. 24 using a Levenberg Marquadt algorithm. This is a gradient-based algorithm and thus produces a local minima. In the particular case of SISO systems using $M=1$, a *global* solution to Eq. 24 has been pointed out in Hernández, 1992.

Unfortunately, there are no general stability results when the control law resulting from Eq. 24 is used. Only special cases of the nonlinear model predictive control law have been studied. One such case is the "P-inverse" controller described above which is obtained by letting $M=1$, $\lambda_1=0$, $\gamma_i=0$, for $i=1, \ldots, P-1$ and $\gamma_P=1$.

The controllers presented in this section have been derived from intuitive notions of relaxing the inverse. We have purposely avoided discussion of very important implementation issues so that the reasoning behind the different controller designs could be presented as clearly as possible. In the next section we will deal with such issues as: plant-model mismatch, disturbance estimation, and filtering of measurement noise.

Controller Implementation

The controllers presented so far have been derived under the assumption of ideal conditions: perfect model and no measurement noise. This section is dedicated to the study of issues arising in the implementation of the model-based controllers derived earlier.

Model predictions into the future

The control schemes presented in previous sections are model-based and thus require predictions into the future. The system of Eq. 10 will form the backbone of the predictions into the future. To this basic structure we will add the capability of estimating the effect of disturbances on future outputs. Initially, we will assume linear disturbance models to compare and contrast the nonlinear MPC formulations with the linear ones in the literature. Afterwards, nonlinear disturbance prediction models are studied. For the linear disturbance models we will use "type 1" and "type 2" models which have been used in the linear MPC design (for example, Morari and Lee, 1991; Ricker, 1991).

The first type of the disturbance model to be considered is the so-called "type 1" disturbance where it is assumed that independent, random steps affect the plant output. This can be formulated by augmenting the state of Eq. 10 with a state variable, ζ, equal to a random step, that is:

$$\begin{bmatrix}\underline{x}(k+1)\\ \zeta(k+1)\end{bmatrix}=\begin{bmatrix}F[\underline{x}(k),\,u(k)]\\ \Phi_d\zeta(k)+G\omega(k)\end{bmatrix}$$
$$y(k)=h[x(k)]+\zeta(k)+\nu(k) \tag{25}$$

where $\Phi_d=G=1$. Note that ζ is generated from integrating the white noise signal, $\omega(k)$, and $\nu(k)$ is the measurement white

noise. Furthermore, note that the state variable $\underline{x}(k)$ is assumed to be completely deterministic, free of any process noise.

The classical estimation problem is to capture the effects of $\omega(k)$ on future outputs while simultaneously filtering the measurement noise $\nu(k)$. This can be accomplished by using an estimator as:

$$\begin{bmatrix} \underline{\hat{x}}(k+1|k) \\ \hat{\underline{\zeta}}(k+1|k) \end{bmatrix} = \begin{bmatrix} F[\underline{\hat{x}}(k|k), u(k)] \\ \Phi_d \hat{\underline{\zeta}}(k|k) \end{bmatrix}$$

$$\begin{bmatrix} \underline{\hat{x}}(k|k) \\ \hat{\underline{\zeta}}(k|k) \end{bmatrix} = \begin{bmatrix} \underline{\hat{x}}(k|k-1) \\ \hat{\underline{\zeta}}(k|k-1) + L(k)[\tilde{y}(k) - \hat{y}(k|k-1)] \end{bmatrix}$$

$$\hat{y}(k|k) = h[\underline{\hat{x}}(k|k)] + \hat{\underline{\zeta}}(k|k) \tag{26}$$

where $\underline{\hat{x}}(k+1|k)$ and $\hat{\underline{\zeta}}(k+1|k)$ are the state and disturbance estimates at time $k+1$ given plant knowledge up to time k; $\tilde{y}(k)$ is the plant output at time k. The estimator gain, $L(k)$, will be selected optimally in the next subsection.

Future predictions of the output can be obtained by the forward iteration of Eq. 26 while using the standard assumption that the expected value of $\tilde{y}(k) - \hat{y}(k|k-1)$ into the future is zero. This type of disturbance description is able to capture any constant bias between the plant and the model. Therefore, if the combined effects of external disturbance and plant-model mismatch reach a constant value, the type 1 assumption along with a controller with integral action will yield offset free performance.

There may be situations in which the combined effects of disturbances and plant-model mismatch do not behave as "type 1." Instead, they can be described better as slow drifts or ramps. For these cases, we can obtain improved predictions and performance by assuming that the disturbance is given by a doubly integrated white noise process. This is the "type 2" disturbance assumption of Morari and Lee (1991) and Ricker (1991). To account for such a disturbance model, the state is augmented with two disturbance state variables as:

$$\begin{bmatrix} \underline{x}(k+1) \\ \underline{\zeta}(k+1) \end{bmatrix} = \begin{bmatrix} F[\underline{x}(k), u(k)] \\ \Phi_d \underline{\zeta}(k) + G\omega(k) \end{bmatrix}$$

$$y(k) = h[x(k)] + \zeta_1(k) + \nu(k) \tag{27}$$

where

$$\underline{\zeta}(k) = \begin{bmatrix} \zeta_1(k) \\ \zeta_2(k) \end{bmatrix}; \quad \Phi_d = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad G = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{28}$$

Note that the transfer function between ζ_1 and ω is given by:

$$\frac{\zeta_1(z)}{\omega(z)} = \frac{1}{(1-z^{-1})^2} \tag{29}$$

Thus the disturbance is modeled as a doubly integrated white noise. Other linear disturbance models can be easily obtained by modifying the matrices Φ_d and G. The estimator structure for this case is identical to Eq. 26.

Nonlinear disturbance models are more difficult to obtain yet it is important to consider them since the input–output model is nonlinear and it is only logical to assume that the effects of disturbance could also be nonlinear. A general representation of polynomial models with nonlinear dependence on disturbance can be given as:

$$y(k+1) = p_n\,[y(k), \ldots, y(k-n_y), u(k), \ldots, u(k-n_u),$$

$$\zeta(k), \ldots \zeta(k-n_d)] \tag{30}$$

where now the output is a polynomial function of not only previous inputs and outputs but also previous disturbances. A state space representation of the model in Eq. 30 could be obtained by augmenting the state of the system of Eq. 10 with a vector containing the previous disturbances, much in the same way the inputs and the outputs are treated. This state space representation would contain an accurate description of the effects of the inputs and disturbances on future outputs.

Models which include the effect of disturbances on future outputs can be obtained also from statistical modeling. An algorithm to construct such a model contains the following essential steps:

1. Construct a model with terms containing only previous inputs and previous outputs (such as Eq. 3).
2. Approximate the disturbances at any given time as the difference between the true plant output and the output estimated from the model obtained in step 1.
3. Now that an estimate of the disturbance is available, continue building the model obtained in step 1 by considering terms containing previous disturbances.

Our experience with this type of approach is that the algorithm fails to add any terms containing previous disturbances in step 3 because they are deemed statistically unnecessary (as determined by various information criteria, such as the Akaike Information Criteria, AIC, and others). This is not to say that if a model such as Eq. 30 is available and reliable, we construct a state space nonlinear disturbance model with which to predict the effects of inputs and disturbances on the future outputs.

However, an alternative and simpler approach to obtaining nonlinear disturbance models is to assume that the disturbance enters linearly at the first state variable in Eq. 10 (the prediction of the output one time into the future) so that:

$$x_1(k+1) = p_n[x(k), u(k)] + \omega(k) \tag{31}$$

which leads to the following model:

$$\underline{x}(k+1) = F[x(k), u(k)] + G\omega(k)$$

$$y(k) = h[\underline{x}(k)] \tag{32}$$

with $G = [1\ 0\ \ldots\ 0]^T$. This disturbance model has a seemingly unjustifiable simple structure. However, recall that the state of the plant is given as the deterministic output. That is, the output of the plant had there been no noise. The disturbance model in Eq. 32 is thus based on the assumption that if the plant were deterministic, then a polynomial model of the previous (deterministic) inputs and outputs can approximate arbitrarily well any input-output map. This assumption was justified by Díaz and Desrochers (1988), as mentioned earlier. Here the approximation error is given by the process noise, $\omega(k)$. No process noise is assumed to enter any other variable

of the "output subsystem" because these other state variables are simply shifts in time of the first state variable.

It is crucial to recognize that even though the disturbance enters linearly at the first state of Eq. 32, the input-output map will be nonlinear in the previous disturbances. To show this, consider the following example:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} x_1(k)[x_2(k)]^2 + u(k) + \omega(k) \\ x_1(k) \end{bmatrix}$$

$$y(k) = x_1(k) \tag{33}$$

Note that for the sake of clarity, measurement noise has been purposely avoided. If the dependency of the states is removed and the input-output map is calculated, we obtain:

$$y(k+1) = y(k)y^2(k-1) + \omega(k) + u(k) \tag{34}$$

It is interesting to point out that even though the input-output map *seems* linear in the disturbance signal $\omega(\cdot)$, it is really not. To see this clearly, we can calculate the total derivative of $y(k+1)$ with respect to $\omega(k-1)$; if this relationship were linear, then the derivative would be a constant. However, note that:

$$\frac{dy(k+1)}{d\omega(k-1)} = \frac{\partial y(k+1)}{\partial \underline{x}(k+1)} \left\{ \frac{\partial \underline{x}(k+1)}{\partial \underline{x}(k)} \frac{d\underline{x}(k)}{d\omega(k-1)} + \frac{\partial \underline{x}(k+1)}{\partial \omega(k-1)} \right\} + \frac{\partial y(k+1)}{\partial \omega(k-1)} \tag{35}$$

which, from Eq. 33 is equal to:

$$\frac{dy(k+1)}{d\omega(k-1)} = \{x_2(k)\}^2 \tag{36}$$

Thus, even though the disturbance enters linearly in the state, previous disturbances have a nonlinear effect on the outputs due to the nonlinear nature of the input-output map. Contrast the disturbance model in Eq. 32 with that in Eq. 25 or 27 where the output is truly a linear function of all previous disturbances [to check, calculate $y(k+1)/d\omega\,(k-i)$ for all i].

Given the difficulties associated with nonlinear state estimation, we will limit ourselves to constructing estimators with linear correction terms. Following the approach used for the linear disturbance models, the estimator for the model in Eq. 32 is given by:

$$\begin{aligned} \underline{\hat{x}}(k+1\,|\,k) &= F[\underline{\hat{x}}(k\,|\,k),\, u(k)] \\ \underline{\hat{x}}(k\,|\,k) &= \underline{\hat{x}}(k\,|\,k-1) + L(k)\,[\tilde{y}(k) - \hat{y}(k\,|\,k-1)] \\ \hat{y}(k\,|\,k) &= h[\underline{\hat{x}}(k\,|\,k)] \end{aligned} \tag{37}$$

where $L(k)[\tilde{y}(k) - \hat{y}(k\,|\,k-1)]$ is the linear correction due to the plant-model mismatch. Note that the difference between the plant and the model is used to update the state of the system. If there is plant-model mismatch, the future predictions of the output using forward iterations of Eq. 37 would be biased and yield steady-state offset if used for model predictive control. This was also noted by Gangadhar and Zafiriou (1992). The classical solution for this type of problem is to augment the state with a disturbance state variable, whose dynamics are given by (for example, Morari and Stephanopoulos, 1980):

$$\zeta(k+1) = \zeta(k) + \omega(k) \tag{38}$$

This new disturbance term is intended here to act as a bias so that the output of the plant and the model can match at steady state. Thus, adding the integrated disturbance to the output we obtain:

$$\begin{bmatrix} \underline{x}(k+1) \\ \zeta(k+1) \end{bmatrix} = \begin{bmatrix} F[\underline{x}(k),\, u(k)] + G\omega_1(k) \\ \zeta(k) + \omega_2(k) \end{bmatrix}$$

$$y(k) = h[x(k)] + \zeta(k) + \nu(k) \tag{39}$$

The estimator with linear correction terms for this type of formulation is thus given by:

$$\begin{bmatrix} \underline{\hat{x}}(k+1\,|\,k) \\ \hat{\zeta}(k+1\,|\,k) \end{bmatrix} = \begin{bmatrix} F(\underline{\hat{x}}(k\,|\,k),\, u(k)) \\ \hat{\zeta}(k\,|\,k) \end{bmatrix}$$

$$\begin{bmatrix} \underline{\hat{x}}(k\,|\,k) \\ \hat{\zeta}(k\,|\,k) \end{bmatrix} = \begin{bmatrix} \hat{x}(k\,|\,k-1) + L_1(k)[\tilde{y}(k) - \hat{y}(k\,|\,k-1)] \\ \hat{\zeta}(k\,|\,k-1) + L_2(k)[\tilde{y}(k) - \hat{y}(k\,|\,k-1)] \end{bmatrix}$$

$$\hat{y}(k\,|\,k) = h(\underline{\hat{x}}(k\,|\,k)) + \hat{\zeta}(k\,|\,k) \tag{40}$$

This way we can obtain unbiased estimates of the outputs (for disturbances which reach a constant value at steady state), update the nonlinear state and capture nonlinear effects of the disturbances by using simple disturbance models derived from sound engineering assumptions.

State estimation

Our objective now is to select the estimator gains, $L(k)$, appearing in Eqs. 26 and 40. As we shall see, the estimator gains for type 1 and 2 disturbance models can be determined optimally using a Kalman filtering technique. However, for the case of nonlinear disturbance models, the selection of $L(k)$ is much more complicated and some simplifications will be made.

As mentioned above, the first two disturbance models considered have a linear effect on the outputs. Also note that the nonlinear part of the system is completely deterministic. It is not difficult to show (Hernández, 1992) that for such systems the optimal estimator gain can be obtained from the Kalman filter by considering only the stochastic linear part [the dynamics of the disturbance vector, $\zeta(k)$]. If only the steady state Kalman Filter gain is used, then it can be computed from the following equations (Åström and Wittenmark, 1984):

$$L = \Phi_d P_\infty C_d^T (R_2 + C_d P_\infty C_d^T)^{-1} \tag{41}$$

where P_∞ is the solution to:

$$P_\infty = \Phi_d P_\infty \Phi_d^T + G R_1 G^T - \Phi_d P_\infty C_d^T (R_2 + C_d P_\infty C_d^T)^{-1} C_d P_\infty \Phi_d^T \tag{42}$$

R_1 and R_2 are the covariances of ω and ν, respectively; Φ_d and C_d are the transition matrix and the output matrix for the linear, stochastic part.

In the case of "Type 1" disturbances (Eq. 25), $\Phi_d = 1$, $G = 1$ and $C_d = 1$. Solving for P_∞ in Eq. 42, we obtain:

$$P_\infty = \frac{R_1 + R_1\sqrt{1 + 4/\varsigma^2}}{2} \tag{43}$$

where ς is the signal-to-noise ratio, $\sqrt{R_1/R_2}$. Substituting this result into Eq. 17 we obtain the optimal steady-state estimator gain:

$$L = \frac{2}{1 + \sqrt{1 + 4/\varsigma^2}} \tag{44}$$

which is equal to the result obtained by Morari and Lee (1991) and Ricker (1991). Also, as pointed out by those authors, the estimator gain L varies between zero and one as the signal-to-noise ratio, ς, varies from zero to infinity. Thus, in practical applications, it results simpler to tune the filter by varying the gain, L, directly instead of through the signal-to-noise ratio.

The optimal estimator gain for the "type 2" disturbance model assumption can also be calculated from Eqs. 41 and 42. This time, however, the transition and output matrices for the linear, stochastic system is given by:

$$\Phi_d = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \; C_d = [1 \; 0] \text{ and } G = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{45}$$

Since the state to be updated is two-dimensional, the filter gain is also two-dimensional. Similar (although much more tedious) calculations to the one above yield the optimal estimator gain as:

$$l_1 = \frac{\sqrt{2}}{2}\sqrt{\varsigma\sqrt{\varsigma^2 + 16} - \varsigma^2}; \quad l_2 = \frac{(l_1)^2}{l_1 + 2}; \quad L = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix} \tag{46}$$

which is the same result obtained by Ricker (1991). Again, as in the "type 1 disturbance model, it is simpler to tune the estimator by changing the gain l_1 directly instead of through the signal-to-noise ratio. This gain varies between zero and two as the signal-to-noise ratio varies between zero and infinity. Therefore, in practice, the user can pick l_1 between zero and two and calculate l_2 from Eq. 46.

The use of static Kalman filters allowed us to calculate the optimal static observer gains for disturbances of types 1 and 2 (Eqs. 44 and 46, respectively). Unfortunately, the nonlinear estimation theory is not as practical as the linear one. While the optimal estimator for linear systems is linear, optimal estimators for nonlinear systems do not have a particular structure. Even if only the best linear estimator is desired (that is, calculating optimal gains for Eq. 40), its calculation may not be possible because the distribution of the state may have infinite number of moments. In short, optimal estimation of nonlinear systems is a topic of theoretical interest and not of much practical relevance.

One solution to the above problem is to linearize the system at every sampling time and use the optimal filter for the resulting linear system. The filter constructed in this fashion is the so called Extended Kalman Filter (EKF). Even though its construction is not rigorous, it has proven to be successful in a number of practical applications (see Dimitratos et al., 1989). We will delay any more discussion on the subject of nonlinear estimation until the next section, where the problem of simultaneous state and parameter estimation will be discussed.

Parameter estimation

Up to this point, we have assumed that the effect of external disturbances and plant-model mismatch are captured by the disturbance model descriptions. However, the plant-model mismatch encountered could be large in cases where the system has entered regions not observed before by the model. When this occurs, it is advantageous to recur to on-line parameter estimation algorithms for improved predictions. The problem of obtaining on-line parameter estimates can be seriously compounded if the measurements are corrupted by noise. For this reason it is important to filter the output in some fashion prior (or simultaneous to) the parameter update step.

A logical first choice for simultaneous filtering and parameter estimation is to use an EKF approach, where the state of the model in Eq. 10 is augmented by considering the unknown parameters to be state variables. For example, let the vector θ contain the model parameters, then we can formulate the following system by assuming that the parameters dynamics are driven by process noise:

$$\begin{bmatrix} \underline{x}(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} F[\underline{x}(k),\, u(k),\, \theta(k)] + G\omega_1(k) \\ \theta(k) + \omega_2(k) \end{bmatrix} \tag{47}$$

and use an EKF approach to estimate the state and the parameters simultaneously. This type of approach has been used frequently in the process control literature (see de Valliere and Bonvin, 1990).

However, as pointed out by Ljung (1979), the Extended Kalman Filter is not appropriate for the purpose of simultaneous state and parameter estimation. The reason is that it may encounter stability problems since it does not take into account the effects of the parameter vector on the filter gains. Ljung improved the algorithm by directly incorporating this effect, and showed global convergence for the case of linear systems. One of the most important conclusions in Ljung's article is that the modified EKF has equivalent asymptotic behavior to the case where the Kalman gain is parametrized and computed along with the model parameters using a recursive prediction error method (RPEM). That is, construct a parameter vector by augmenting the vector of model parameters, θ, with the estimator gain parameters, L:

$$\vartheta(k) = \begin{bmatrix} \theta(k) \\ L(k) \end{bmatrix} \tag{48}$$

Then, an RPEM is used to obtain the values of ϑ as time progresses. This is a more practical approach since the filter gains are not artificially tuned through assumed noise properties. The filter gains are instead estimated directly through the RPEM. Furthermore, the algorithm has been shown to have improved stability properties over the classical EKF approach (Ljung, 1979).

The equations for the RPEM will be constructed assuming that the system is modeled as in Eq. 37 as opposed to Eq. 40. Note that the difference between the two models is that Eq.

37 does not contain the additional bias used to capture the plant model mismatch. The reason for using the "bias-free" model is that when estimating the model parameters, the bias state (ζ) should be disregarded so that the parameters capture as much of the system's behavior as possible. The appropriate structure of the RPEM for the system described in Eq. 37 is given by:

Time Update

$$\underline{\hat{x}}(k+1|k)=F[\hat{x}(k|k),\ u(k),\ \underline{\theta}(k)] \tag{49}$$

Output Calculation

$$\hat{y}(k+1|k)=h[\hat{x}(k+1|k)] \tag{50}$$

Error Calculation

$$\epsilon(k+1|k)=\bar{y}(k+1)-\hat{y}(k+1|k) \tag{51}$$

Measurement Update

$$\underline{\hat{x}}(k+1|k+1)=\underline{\hat{x}}(k+1|k)+L(k)\epsilon(k+1|k) \tag{52}$$

Parameter Update

$$\underline{\vartheta}(k+1)=\begin{bmatrix}\hat{\theta}(k+1)\\ L(k+1)\end{bmatrix}=\underline{\hat{\vartheta}}(k)+\frac{P_C(k)\psi(k)}{\lambda(k)+\psi^T(k)P_C(k)\psi(k)}\epsilon(k+1|k) \tag{53}$$

$\psi(k+1)$ *Calculation*

$$\psi(k+1)=\left[\frac{\partial\hat{y}(k+1|k+1)}{\partial\underline{\hat{\theta}}(k)}\right]^T=\left[\frac{\partial\hat{y}(k+1|k+1)}{\partial\underline{\hat{x}}(k+1|k+1)}\frac{\partial\underline{\hat{x}}(k+1|k+1)}{\partial\underline{\hat{\theta}}(k)}\right]^T$$

$$=\eta^T(k+1)C_k^T;\ C_k=\frac{\partial h[\underline{\hat{x}}(k|k)]}{\partial\underline{\hat{x}}(k|k)} \tag{54}$$

$\eta(k+1)$ *Calculation*

$$\eta(k+1)=\frac{d\underline{\hat{x}}(k+1|k+1)}{d\underline{\hat{\vartheta}}(k)}=\frac{\partial\underline{\hat{x}}(k+1|k+1)}{\partial\underline{\hat{x}}(k|k)}\frac{d\underline{\hat{x}}(k|k)}{d\underline{\hat{\vartheta}}(k)}+\frac{\partial\underline{\hat{x}}(k+1|k+1)}{\partial\underline{\hat{\vartheta}}(k)}$$

$$=\frac{\partial[\underline{\hat{x}}(k+1|k)+L(k)\epsilon(k+1|k)]}{\partial\underline{\hat{x}}(k|k)}\frac{d\underline{\hat{x}}(k|k)}{\partial\underline{\vartheta}(k)}+\frac{\partial[\underline{\hat{x}}(k+1|k)+L(k)\epsilon(k+1|k)]}{\partial\underline{\hat{\vartheta}}(k|k)}$$

$$=[I-L(k)C(k)]\frac{\partial F[\underline{\hat{x}}(k|k),\ u(k),\ \underline{\hat{\theta}}(k)]}{\partial\underline{\hat{x}}(k|k)}\eta(k)+\frac{\partial\{F[\underline{\hat{x}}(k|k),\ u(k),\ \underline{\hat{\theta}}(k)]+L(k)\epsilon(k+1|k)\}}{\partial\underline{\hat{\vartheta}}(k)} \tag{55}$$

$P_C(k+1)$ *Calculation*

$$P_C(k+1)=P_C(k)-\frac{P_C(k)\psi(k)\psi^T(k)P_C(k)}{\lambda(k)+\psi^T(k)P_C(k)\psi(k)}+\delta I \tag{56}$$

where $\lambda(k)$ is the familiar "forgetting factor," and $P_C(k)$ is inversely proportional to the covariance of the parameter vector. The term δ is a small and positive parameter selected to maintain P_C positive definite. The initial condition on $\psi(0)$ is a matrix of zeros and $P_C(0)$ is usually selected as a diagonal matrix. The tuning parameters for this algorithm are the initial values of P_C, the forgetting factor and the value of δ.

The recursive prediction error method outlined above will be used both off-line and on-line for the estimation of the parameters. Once the structure of the polynomial ARMA model is selected, the data will be passed through the RPEM method several times until the parameters converge. This approach has been found to be competitive in comparison to other off-line parameter estimation algorithms (Ljung, 1987, Chapter 11). After the model is obtained, the RPEM is also used on-line to update the model parameters in the hope of capturing the variations in the plant. On both of these implementations, a variable forgetting factor is used instead of the fixed one shown above. The forgetting factor is updated as outlined in the algorithm by Fortesque et al. (1981).

Note the fundamental difference between the two approaches used to find the estimator gains. On the one hand, the classical extended Kalman filter approach calls for augmenting the state of the model by adding the model parameters. On the other hand, the recursive prediction error method augments the vector of model parameters by adding the estimator gains. We will use the second approach because of its improved stability properties and practical values.

Examples

The purpose of this section is to illustrate the theory presented with simulations of a chemical system. The system to be studied is a first-order exothermic reaction carried in a continuous stirred tank reactor (CSTR). This system is particularly challenging for our study because, depending on operating conditions, it exhibits unstable behavior. As noted by Sistu and Bequette (1991), if an unstable system is to be controlled with MPC-type controllers, the state of the system must be known or at least well estimated. However, note that the standing assumption throughout the article is that the dynamic equations describing the system are unknown. That is, we only assume knowledge of the systems input-output behavior while the internal dynamics are completely unknown. First-order, exothermic chemical reactions in CSTRs have been extensively studied in the literature, most notably by Uppal et al. (1974). The dynamic equations can be written in dimensionless form as:

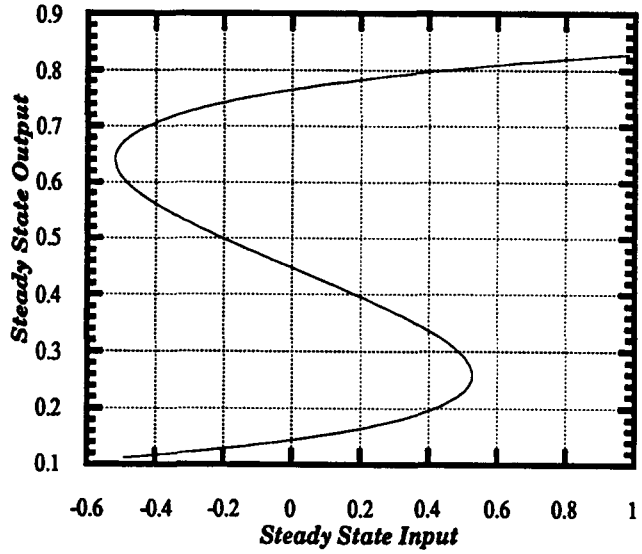

Figure 1. Steady-state output vs. steady-state input for CSTR example.

$$\dot{x}_1 = -x_1 + D_a(1-x_1)\exp\left(\frac{x_2}{1+x_2/\varphi}\right)$$

$$\dot{x}_2 = -x_2 + BD_a(1-x_1)\exp\left(\frac{x_2}{1+x_2/\varphi}\right) + \beta\,(\mu - x_2) \qquad (57)$$

$$y = x_1$$

where x_1 and x_2 are the dimensionless reactant concentration and temperature, respectively. The input, u, is the cooling jacket temperature, the physical parameters are D_a, φ, B and β which correspond to the Damköhler number, the dimensionless activation energy, heat of reaction and heat-transfer coefficients, respectively. The system in Eq. 57 exhibits numerous forms of behavior depending on the values of the physical parameters and the regions of operation. If we choose the following parameters:

$$D_a = 0.072, \quad \varphi = 20.0, \quad B = 8.0, \quad \beta = 0.3 \qquad (58)$$

then the system can exhibit up to three steady states one of which is unstable, as shown in Figure 1. The major challenge ahead is to control the reactor around the unstable region while assuming that the dynamic equations are unknown. To achieve this we begin with the identification step.

Identification

The input-output data needed to identify the polynomial ARMA model was generated by submitting the system to an input generated by the following rules:

1. At the end of a sampling time, the input changes values with probability p_s. The user can control the frequency of change of the input by adjusting this parameter.

2. If the input is to be changed, the new value is to be obtained from an uniform distribution with a prespecified upper and lower bound. This way the user can control the magnitudes of the signals to be applied.

This input signal was selected over the more traditional pseudo-random binary signal (PRBS) for two reasons. First, PRBS signals excite the system using only two input magnitudes, which is insufficient in nonlinear identification. Furthermore, the input sequence generated following the above rules has a symmetric distribution. This is important for several input-dependent analysis available in the literature for the identification of nonlinear systems. For simulation purposes, the output obtained from the system sampled every 0.5 dimensionless time units was corrupted by adding a white noise signal with zero mean and a standard deviation of 0.02. The input was generated using $p_s = 0.5$ and the lower and upper bounds on the input magnitude were selected as -2 and 2. Also, the inputs were scaled by mapping the input range of $(-4, 4)$ to $(0, 1)$. That is, define a new scaled input as:

$$u_s(k) = \frac{u(k)+4}{8}$$

This was done in an effort to maintain the inputs, outputs and the regressor terms in Eq. 4 in the same order of magnitude. The output and input data are shown on Figure 2. An important concern is that the input signal used forces the system from one stable region to another. This is important from an identification point of view so that the data reflects as much of the system's dynamics as possible. However, such operation may not be allowed at an industrial site. Research is underway to reduce the requirements on the input-output data.

The structure used to model this system is that of a polynomial ARMA model followed by a sigmoid function as shown in Eq. 6. This is to illustrate that the ideas presented in this article for polynomial models also apply for this type of model. Furthermore, since the output of the system is physically bounded in [0, 1] the model in Eq. 6 seems to be a more natural choice. The particular structure of the model was determined by using a stepwise model building algorithm as discussed, for example by Kortmann and Unbehauen (1990). The transformation in Eq. 8 was used for this purpose. The following structure was determined:

$$y(k+1) = \sigma\{p_n[y(k), \ldots, y(k-3), u_s(k), \ldots, u_s(k-3)]\}$$

$$\begin{aligned} p_n(\cdot,\cdot) = {} & \theta_0 + \theta_1 y(k) + \theta_2 y^3(k-3) + \theta_3 u_s(k-1)\, u_s(k-3) \\ & + \theta_4 y^2(k-1)\, u_s(k-3) + \theta_5 u_s(k-2) \\ & + \theta_6 y(k-3) u_s^2(k-3) \\ & + \theta_7 u_s(k-3) + \theta_8 y(k) u_s(k-1)\, u_s(k) + \theta_9 u_s(k) \end{aligned} \qquad (59)$$

To finalize the model construction, the parameters of the system were calculated using the RPEM method discussed. The parameters of the RPEM were chosen as follows: $P_C(0) = 0.5I$, $\lambda(0) = 0.95$, $\delta = 0.05$ and:

$$L(0) = [0.8\ 0\ \cdots\ 0]^T$$

The parameters of the model were given by:

$$\begin{aligned} &\theta_0 = -3.075 \quad \theta_1 = 5.719 \quad \theta_2 = -0.575 \quad \theta_3 = 0.191 \\ &\theta_4 = -2.632 \quad \theta_5 = 0.262 \quad \theta_6 = 1.897 \quad \theta_7 = 0.217 \\ &\theta_8 = 0.781 \quad \theta_9 = 0.200 \end{aligned} \qquad (60)$$

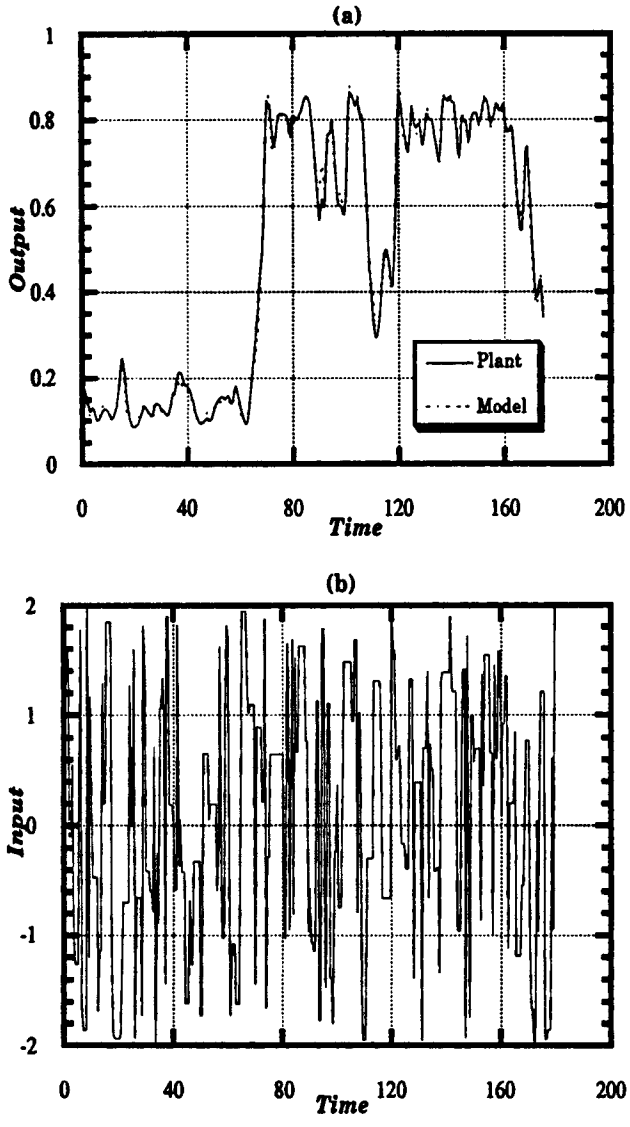

Figure 2. Input-output data for identification of reaction system in CSTR.

The output predicted from the model is also shown in Figure 2.

Analysis

We have described as our major challenge the control of the nonlinear reaction around an unstable equilibrium point. Because of this, let us investigate how well the polynomial ARMA model predicts the desired equilibrium point and if it can predict that it is in fact unstable. We select our desired output to be a first-order trajectory beginning at a value of $y^* = 0.144$ and finishing at the unstable equilibrium point when the input is zero: ($y^* = 0.445$, $u^* = 0$).

To calculate the equilibrium point predicted by the polynomial ARMA model, set $y(k+1) = y(k) = \cdots = y(k-n_y) = y^*$ and $u(k) = u(k-1) = \cdots = u(k-n_u) = u^*$ in Eq. 59. Now set $u^* = 0.0$ ($u_s^* = 0.5$) and solve for the value of y^* that satisfies Eq. 59 given the above restriction. From this calculation, we find that the equilibrium point predicted by

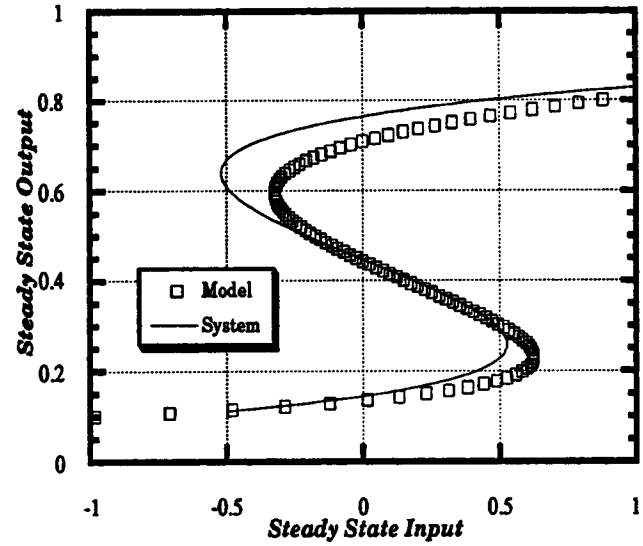

Figure 3. Comparison of steady states of identified model.

the polynomial ARMA model is ($y^* = 0.443$, $u^* = 0.0$) which does not deviate too much from the actual point. The equilibrium curve of the identified model is shown along that of the system in Figure 3.

The stability of the system as predicted from the polynomial ARMA model can be determined by calculating the eigenvalues of the Jacobian of the state transition matrix evaluated at the equilibrium point. For the model's equilibrium point of ($y^* = 0.443$, $u^* = 0.0$), the eigenvalues of the Jacobian were given by: 1.13, -0.227 and 0.229 ± 0.279. Since at least one of the eigenvalues is outside the unit circle, the model correctly predicts that the system is unstable at the equilibrium point under consideration.

The invertibility of the model could also be investigated using theorem 2 to determine the feasibility of using the inverse model for control. Performing these calculations, we find that the spectral radius of the closed loop system is 1.38, thus the inverse system is unstable. It is important to note that the continuous system does not have an unstable inverse, the fact that the polynomial model has an unstable inverse may be due to sampling or simply plant-model mismatch. Nevertheless, these calculations demonstrate that the inverse model controller would be unstable and thus we would be wise to relax the inverse in some fashion.

Control

The control algorithm to be used in this section is the model-predictive controller displayed earlier. Our main focus will be the comparison of the performance obtained by assuming the different disturbance models presented earlier. The simulations here presented will use the same controller: a model-predictive controller with a prediction horizon (P) of 7, a control move horizon (M) of 1, an output weight of 1 (for all times into the future) and an input weight of 0.2 (also for all times). The main difference in the simulations to be presented is the disturbance model assumed.

We begin our simulation studies assuming that the disturbance can be captured by a type 1 disturbance. The simulation

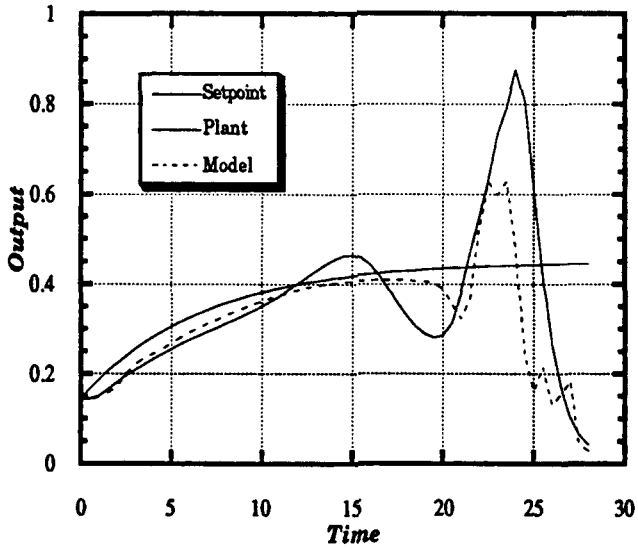

Figure 4. Control using type 1 disturbance assumption.

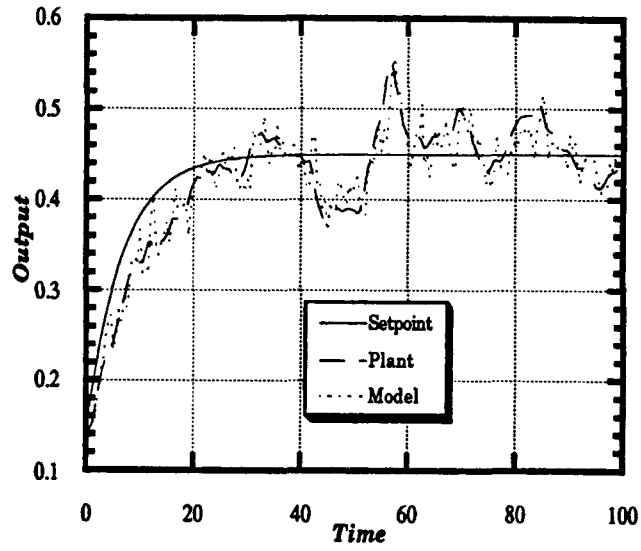

Figure 6. Control using nonlinear disturbance model assumption: measurement noise and process disturbance.

goal was to take the system from a stable equilibrium point ($x_1 = 0.144$, $x_2 = 0.886$, $u = 0.0$) to an unstable one: ($x_1 = 0.445$, $x_2 = 2.75$, $u = 0.0$). First, the output was measured free of noise so that the issue of predicting the disturbance does not get confused with the proper filtering of the relevant signals. As can be seen from Figure 4, the system could not be controlled and becomes unstable. A similar result occurs when a type 2 disturbance model is assumed and thus the results are not shown.

Given that the linear disturbance assumptions yielded unstable behavior, we turn our attention to nonlinear disturbance models given by Eq. 39 to describe the plant-model mismatch more accurately. The goal of the simulation was the same as the one for linear disturbance assumptions. The result of the simulation is shown in Figure 5. Note that this time the controller is able to stabilize the closed-loop system.

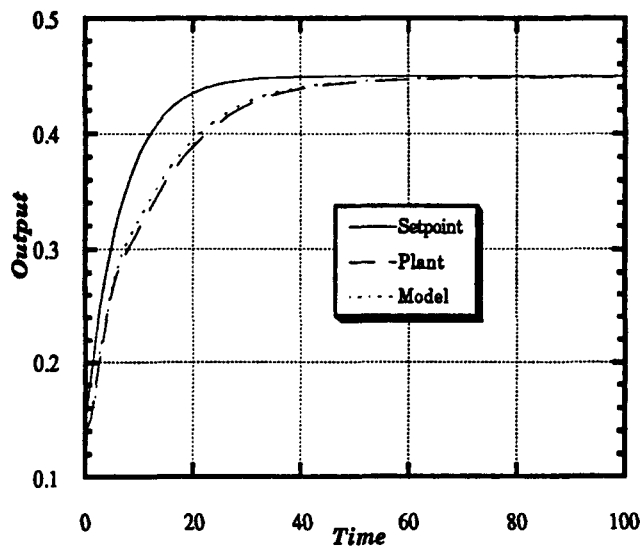

Figure 5. Control using nonlinear disturbance model assumptions: no measurement noise.

The reason for the improved performance is that the state of the nonlinear realization is estimated when using the nonlinear disturbance models. Note that with linear disturbance models the original state of the system is not estimated; the estimator is only used to obtain the bias between the plant and the model. So, recalling the comments of Sistu and Bequette (1991): when controlling unstable systems using MPC, it is crucial that the state of the process is well estimated. Even though the true state of the process is not estimated (the state of the system of Eq. 57), the realization of the input-output map is able to capture the internal dynamics of the system. That is, for this system it is enough to observe the external information to have a clear picture of the internal dynamics. This is, in fact, the concept of nonlinear observability as presented by Leontaritis and Billings (1985).

The final simulation displays the disturbance rejection and noise filtering properties of the algorithm. This time the goal is, again, to bring the system from the mentioned stable equilibrium point to the unstable one. This time, however, the heat-transfer coefficient is assumed to undergo a pulse: it decreases by 10% at time = 40 and it regains its original value at time = 50. This should occur at a time where the controller has already brought the system into the unstable region. Furthermore, the output measurements were assumed to be corrupted by a white noise signal with variance equal to 0.03 of the average magnitude of the output. The RPEM parameters were the same used on the identification section. The results are shown on Figure 6. Note that the controller is able to stabilize the system in the presence of noise and reject disturbances. The performance, however, is notably worse than with the noise free measurements which indicates the sensitivity of the closed-loop system around the unstable region.

Conclusions

The objective of this article was to develop a methodology for the control of nonlinear systems using input-output in-

formation. To achieve this goal we used polynomial ARMA models for the identification of the nonlinear systems dynamics. The next step in the procedure is to analyze the identified model in order to gather control-relevant information such as stability of the open-loop system and stability of the inverse model controller. During the analysis step we learned about connections between the invertibility question and the feedback linearization of discrete time systems. Furthermore, we gained insight on the problems caused by inverse control and on ways to relax this controller to improve robustness. One such way to relax the inverse controller is through model-predictive control. This controller was developed and we concentrated on implementation issues such as the estimation of the effect of disturbance on future predictions, the selection of the estimator gains and the model parameters. Finally, examples were simulated that displayed the theory.

Acknowledgment

The financial assistance of IBM and International Paper are greatly appreciated.

Notation

B = dimensionless heat of reaction in CSTR example
$\underline{c}^a$ = vector selected by the user in the feedback linearization of discrete time systems
C_d = matrix describing the output of the linear disturbance models
D_a = Damköhler number of CSTR example
$F(\cdot,\cdot)$ = state transition operator for nonlinear system
G = matrix for modeling process noise in disturbance models
$h(\cdot)$ = output read-out map for nonlinear system
J = Jacobian of state equations with respect to complete state
J' = Jacobian of inverse system with respect to complete state
J_U = Jacobian of inverse system with respect to input substate
J_Y = Jacobian of state equations with respect to output substate
k = counter denoting "sampling" or "discretization" time
L = gains of disturbance model estimators
$\mathcal{L}$ = Lipschitz constant
M = control horizon of model-predictive controller
n_u = maximum delay in the input considered in an input/output model
n_y = maximum delay in the output considered in an input/output model
P = prediction horizon of model-predictive controller
p_n = polynomial function of its arguments of order n
P_∞ = steady-state covariance of the state for optimal estimators
q = shift operator: $x(k+1)=q\cdot x(k)$
r = desired plant output
R_1 = covariance of process noise
R_2 = covariance of measurement noise
u = system's input
u_s = scaled system input
$\underline{v}$ = state of the system's inverse
$\underline{x}$ = state of a dynamical system
$\underline{x}^a$ = output part of the state
$\underline{x}^b$ = input part of the state
$y, \tilde{y}, \hat{y}$ = system's output, measured and estimated system output
y^{tr} = transformed output variable in Eq. 8
z = $\mathbb{Z}$ transform variable

Greek letters

β = dimensionless heat-transfer coefficient in CSTR example
γ = weights on error in the output
η = derivative of state with respect to model parameters (RPEM)
λ = forgetting factor in RPEM algorithm
λ_j = weight on change in input at time $k+j-1$ of model-predictive controller
ν = measurement noise signal
Φ_d = matrix describing dynamics of linear disturbance model
φ = dimensionless activation energy of reaction in CSTR example
θ = model parameters
ψ = derivative of estimated output with respect to model parameters (RPEM)
ρ = spectral radius operator
ϱ = delay of an input/output map
ς = signal-to-noise ratio
ζ = disturbance state of disturbance model assumptions
ω = process noise signal

Literature Cited

Åström, K. J., and B. Wittenmark, *Computer Controlled Systems: Theory and Design*, Prentice Hall, Englewood Cliffs, NJ (1984).
Boyd, S., and L. O. Chua, "Fading Memory and the Problem of Approximating Nonlinear Operators with Volterra Series," *IEEE Trans. of Circuits and Systems*, **32**, 1150 (1985).
Chen, S., S. A. Billings, and P. M. Grant, "Nonlinear Identification Using Neural Networks," *Int. J. of Control*, **51**, 1191 (1990).
Chen, S., C. F. N. Cowan, and P. M. Grant, "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," *IEEE Trans. on Neural Networks*, **2**, 302 (1991).
Cybenko, G., "Approximation by Superpositions of Sigmoidal Functions," *Math. of Control, Signals and Systems*, **2**, 303 (1989).
de Valliere, P., and D. Bonvin, "Application of Estimation Techniques to Batch Reactors: iii. Modelling Refinements Which Improve the Quality of State and Parameter Estimation," *Comp. and Chem. Eng.*, **14**, 799 (1990).
Devaney, R., *Introduction to Chaotic Dynamical Systems*. Addison-Wesley, New York (1989).
Díaz, H., and A. Desrochers, "Modelling of Nonlinear Discrete-Time Systems for Input/Output Data," *Automatica*, **24**, 629 (1988).
Dimitratos, J., C. Georgiakis, M. S. El-Aasser, and A. Klein, "Dynamic Modelling and State Estimation for an Emulsion Copolymerization Reactor," *Comp. and Chem. Eng.*, **13**, 21 (1989).
Fortescue, T. R., L. S. Kershenbaum, and B. E. Ydstie," Implementation of Self-Tuning Regulators with Variable Forgetting Factors," *Automatica*, **17**, 831 (1981).
Gangadhar, G., and E. Zafiriou, "Nonlinear Quadratic Dynamic Matrix Control With State Estimation," *Ind. and Eng. Chemistry Res.*, **31**, 1096 (1992).
Haber, R., and H. Unbehauen, "Structure Identification of Nonlinear Dynamical Systems—A Survey on Input/Output Approaches," *Automatica*, **26**, 651 (1990).
Hernández, E., "Control of Nonlinear Systems Using Input-Output Information," PhD Diss., Dept. of Chemical Engineering, Georgia Institute of Technology (1992).
Hernández, E., and Y. Arkun, "A Study of the Control Relevant Properties of Backpropagation Neural Net Models of Nonlinear Dyamical Systems," *Comp. and Chem. Eng.*, **16**, 227 (1992).
Kalman, R. E., and J. E. Bertram, "Control System Analysis and Design via the Second Method of Lyapunov: ii. Discrete Systems," *Trans. of ASME*, 394 (1960).
Korenberg, M., S. A. Billings, Y. P. Liu, and P. J. McIlroy, "Orthogonal Parameter Estimation Algorithm for Nonlinear Stochastic Systems," *Int. J. of Control*, **48**, 193 (1988).
Kortmann, M., and H. Unbehauen, "Structure Detection in the Identification of Nonlinear Systems," *Autom. Prod. Infor. Ind.*, **22**, 5 (1988).
Leontaritis, I. J., and S. A. Billings, "Input-Output Parametric Models for Nonlinear Systems: I. Deterministic Nonlinear Systems," *Int. J. of Control*, **41**, 303 (1985).
Ljung, L., "Asymptotic Behavior of the Extended Kalman Filter as a Parameter Estimator for Linear Systems," *IEEE Trans. on Automatic Control*, **24**, 36 (1979).
Ljung, L., *System Identification-Theory for the User*, Chap. 11, Prentice-Hall, Englewood Cliffs, NJ (1987).

Monaco, S., and D. Normand-Cryot, "Zero Dynamics of Sampled Nonlinear Systems," *System and Control Lett.*, **11**, 229 (1988).

Morari, M., and J. H. Lee, "Model Predictive Control: The Good, the Bad and the Ugly," *Proc. of CPC IV*, 419 (1991).

Morari, M., and G. Stephanopoulos, "Structural Aspects and the Synthesis of Alternate Feasible Control Schemes," *AIChE J.*, **26**, 232 (1980).

Pottmann, M., and D. E. Seborg, "Identification of Nonlinear Processes Using Reciprocal Multiquadric Functions," AIChE Meeting, p. 511 (1991).

Ricker, N. L., "Model Predictive Control: State of the Art," *Proc. of CPC IV*, 271 (1991).

Sistu, P. B., and B. W. Bequette, "Nonlinear Predictive Control of Uncertain Processes: Application to a CSTR," *AIChE J.*, **37**, 1711 (1991).

Uppal, A., W. H. Ray, and A. B. Poore, "On the Dynamic Behavior of Continuous Stirred Tank Reactors," *Chem. Eng. Sci.*, **29**, 967 (1974).

Manuscript received Feb. 28, 1992, and revision received Aug. 5, 1992.