Chemical Engineering Science, Vol. 47, No. 4, pp. 737–753, 1992.
Printed in Great Britain.

0009–2509/92 $5.00 + 0.00

A NONLINEAR DMC ALGORITHM AND ITS APPLICATION TO A SEMIBATCH POLYMERIZATION REACTOR

T. PETERSON, E. HERNÁNDEZ, Y. ARKUN† and F. J. SCHORK
School of Chemical Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0100, U.S.A.

(*Received for publication* 6 *August* 1991)

Abstract—In this paper a new nonlinear model-predictive control method is developed and applied to a semibatch polymerization reactor. The proposed control algorithm uses an explicit nonlinear process model and the basic elements of the classical dynamic matrix control (DMC). Update of the DMC model by a disturbance vector which accounts for the effect of nonlinearities in the prediction horizon is the key feature of the method.

1. INTRODUCTION

Model-predictive control (MPC) is now widely recognized as a powerful methodology to address industrially important control problems. This is supported by many reported industrial applications and academic studies (McAvoy *et al.*, 1989). The general strategy of MPC algorithms is to utilize a model to predict the output into the future and minimize the difference between this predicted output and the desired one by computing the appropriate control actions. In particular, one MPC technique, dynamic matrix control (DMC), has been studied very extensively (Prett and García, 1988).

Most of the MPC techniques, including DMC, are based on linear models and are thus not very well-suited for the control of nonlinear systems. Because of this, there have been numerous efforts to extend MPC techniques for the control of nonlinear systems. These include the papers by Brengel and Seider (1989), Eaton *et al.* (1988), Li and Biegler (1989) and Li *et al.* (1990).

In this paper we present a novel MPC algorithm which uses the main structure of DMC, and yet explicitly accounts for nonlinearities in the process model. We implement this controller on a semibatch polymerization reactor and illustrate the utility of the proposed nonlinear MPC technique by simulation results. The paper is structured as follows. Section 2 gives an overview of the linear DMC algorithm. This paves the way to the nonlinear extension of DMC which is discussed in Section 3. Finally, simulation results for the polymerization reactor are presented in Section 4.

2. LINEAR DMC

The extended DMC algorithm here developed relies heavily on the standard, linear DMC control law. Because of this, we will provide a brief explanation of this algorithm (for more details see García and Morshedi, 1986). The major elements of DMC are: (a) the model, (b) estimation of the disturbance and projection into the future, and (c) computation of the control inputs.

2.1. *Modeling*

Consider the single-input/single-output case (SISO), without any loss of generality. The model used is a discrete "step-response model" of the form

$$y(k) = \sum_{i=1}^{N} a_i \Delta u(k-i) + a_N u(k-N-1) + d(k) \tag{1}$$

where

- k is the sampling time
- u is the input
- a_i are the step-response coefficients
- d is the unmodeled or disturbance effects on the output
- N is the number of step-response coefficients needed to adequately describe the process dynamics
- $\Delta u(k)$ is the change in the input defined as $u(k) - u(k-1)$.

2.2. *Estimation of the disturbance effects*

The current value of the disturbance effects can be estimated by subtracting the effects of past inputs on output from the current measurement of the output, i.e.

$$d(k) = y^{\text{meas}}(k) - \sum_{i=1}^{N} a_i \Delta u(k-i) - a_N u(k-N-1). \tag{2}$$

2.3. *Prediction into the future*

In the sequel capital and lowercase boldface letters will denote matrices and vectors, respectively. The

†Author to whom correspondence should be addressed.

linear estimates of the future output are given by

$$\underbrace{\begin{bmatrix} y(k+1) \\ y(k+2) \\ y(k+3) \\ \\ y(k+M) \\ \\ y(k+P) \end{bmatrix}}_{\mathbf{y}^{lin}} = \underbrace{\begin{bmatrix} a_{ss}u(k-N) \\ a_{ss}u(k-N+1) \\ a_{ss}u(k-N+2) \\ \\ a_{ss}u(k-N+M-1) \\ \\ a_{ss}u(k-N+P-1) \end{bmatrix} + \begin{bmatrix} a_2 & a_3 & \cdots & \cdots & a_N \\ a_3 & a_4 & \cdots & a_N & 0 \\ a_4 & a_5 & \cdots & 0 & 0 \\ \vdots & & & \vdots & \\ a_{M+1} & \cdots & \cdots & 0 & 0 \\ \vdots & & & & \\ a_{p+1} & \cdots & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u(k-1) \\ \Delta u(k-2) \\ \Delta u(k-3) \\ \\ \Delta u(k-N+1) \end{bmatrix}}_{\mathbf{y}^{past}}$$

$$+ \underbrace{\begin{bmatrix} a_1 & & & 0 \\ a_2 & a_1 & & \\ a_3 & a_2 & a_1 & \\ & & & \\ a_M & a_{M+1} & \cdots & a_1 \\ \vdots & & & \\ a_P & a_{P-1} & \cdots & a_{P-M+1} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \Delta u(k+2) \\ \vdots \\ \Delta u(k+M-1) \end{bmatrix}}_{\Delta \mathbf{u}} + \underbrace{\begin{bmatrix} d(k+1) \\ d(k+2) \\ d(k+3) \\ \vdots \\ d(k+M) \\ \vdots \\ d(k+P) \end{bmatrix}}_{\mathbf{d}} \tag{3}$$

or

$$\mathbf{y}^{lin} = \mathbf{y}^{past} + \mathbf{A}\Delta\mathbf{u} + \mathbf{d}. \tag{4}$$

As shown in Fig. 1, P denotes the length of the prediction horizon. M is the move horizon or the number of future moves $\Delta u(k), \ldots, \Delta u(k+M-1)$ calculated by the DMC algorithm discussed below. $\mathbf{A}$ is the dynamic matrix composed of the step-response coefficients. The effects of the known past inputs on the future outputs is defined by the vector $\mathbf{y}^{past}$.

Often the future behavior of the disturbance is not known, and it is customary to assume that the future values will be equal to the currently estimated value, i.e. $d(k+i) = d(k)$ for $i = 1, \ldots, P$ and $d(k)$ is calculated from eq. (2).

With these definitions the future output can now be predicted from eq. (4) for any given vector of future control moves $\Delta\mathbf{u}$.

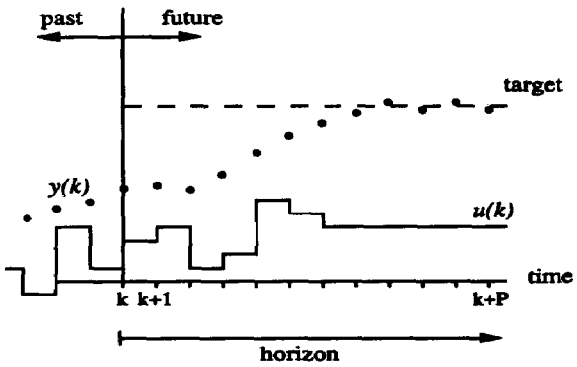


Fig. 1. The moving horizon of DMC.

2.4. *Calculation of the control inputs*

The following optimization problem is used to calculate the control inputs:

$$\min_{\Delta\mathbf{u}} \sum_{i=1}^{P} \gamma^2(i)[y^{sp}(k+i) - y^{lin}(k+i)]^2 + \sum_{j=1}^{M} \lambda^2(j)[\Delta u(k+M-j)]^2. \tag{5}$$

y^{sp} is the set point, γ and λ are the time-varying weights on the output error and on the change in the input, respectively. The solution to the above problem is a least-squares solution in the form of the following linear DMC control law:

$$\Delta\mathbf{u} = (\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}\mathbf{A} + \boldsymbol{\Lambda}^T\boldsymbol{\Lambda})^{-1}\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}(\mathbf{y}^{sp} - \mathbf{y}^{past} - \mathbf{d}) \tag{6}$$

where $\boldsymbol{\Gamma}$ and $\boldsymbol{\Lambda}$ are diagonal matrices containing the weights $\gamma(i)$ and $\lambda(i)$, respectively. Usually only the first calculated move $\Delta u(k)$ is implemented and the calculations are repeated at the next sampling time to account for changing disturbances and to incorporate feedback. P, M, $\boldsymbol{\Gamma}$ and $\boldsymbol{\Lambda}$ are used as tuning parameters by the designer.

3. NONLINEAR DMC

The nonlinear DMC algorithm proposed here stems from reinterpreting the disturbance vector, $\mathbf{d}$, appearing in the prediction equations (3). If the linear DMC is implemented on a linear plant, the vector $\mathbf{d}$ includes the effects of external disturbances, assuming a perfect linear model. However, if the same control law is applied to a nonlinear plant (see Fig. 2), the vector $\mathbf{d}$ will contain contributions from nonlinearities defined as $\mathbf{d}^{nl}$ as well as external disturbances defined

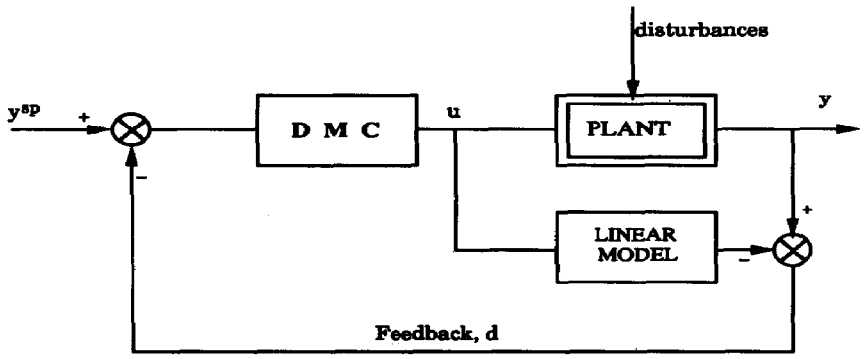


Fig. 2. Linear DMC.

as $\mathbf{d}^{ext}$, i.e.

$$\begin{bmatrix} d(k+1) \\ \vdots \\ d(k+P) \end{bmatrix} = \begin{bmatrix} d^{ext}(k+1) \\ \vdots \\ d^{ext}(k+P) \end{bmatrix} + \begin{bmatrix} d^{nl}(k+1) \\ \vdots \\ d^{nl}(k+P) \end{bmatrix}. \quad (7)$$

In linear DMC the vector $\mathbf{d}$ and therefore the future disturbance vector due to nonlinearities, $\mathbf{d}^{nl}$, is assumed constant. In the nonlinear DMC formulation, the vector $\mathbf{d}^{nl}$ will be estimated using a nonlinear dynamic model and will no longer be assumed to be constant. The development proceeds as follows.

First, in accordance with eqs (4) and (7) we define an *extended linear* DMC model over the future prediction horizon of length P:

$$\mathbf{y}^{el} = \mathbf{y}^{past} + \mathbf{A}\Delta\mathbf{u} + \mathbf{d}^{ext} + \mathbf{d}^{nl}$$

where $\mathbf{d}^{nl}$ is varying from one sampling time to another, and $\mathbf{d}^{ext}$ is assumed to remain constant over the prediction horizon. The optimal DMC inputs for the extended linear model become

$$\Delta\mathbf{u} = (\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}\mathbf{A} + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}(\mathbf{y}^{sp} - \mathbf{y}^{past} - \mathbf{d}^{ext} - \mathbf{d}^{nl}). \quad (8)$$

The objective is to compute a time-varying disturbance vector, $\mathbf{d}^{nl}$, that captures the future disturbance due to nonlinearities and renders the above $\Delta\mathbf{u}$ optimal for the *nonlinear* model. This is accomplished by requiring the output of the extended linear model ($\mathbf{y}^{el}$) match the output from the nonlinear model ($\mathbf{y}^{nl}$) at all the future sampling times:

$$\begin{bmatrix} y^{nl}(k+1) \\ \vdots \\ y^{nl}(k+P) \end{bmatrix} = \begin{bmatrix} y^{el}(k+1) \\ \vdots \\ y^{el}(k+P) \end{bmatrix} = \mathbf{y}^{past} + \mathbf{A}\Delta\mathbf{u} + \mathbf{d}^{ext} + \begin{bmatrix} d^{nl}(k+1) \\ \vdots \\ d^{nl}(k+P) \end{bmatrix}. \quad (9)$$

This is a set of P simultaneous nonlinear equations in P unknowns, $d^{nl}(k+1), \ldots, d^{nl}(k+P)$. The quantities $\mathbf{y}^{past}$ and $\mathbf{d}^{ext}$ can be separately calculated as before (see below), and $\Delta\mathbf{u}$ is obtained from eq. (8). Note also that the terms containing nonlinearities with respect to $\mathbf{d}^{nl}$ appear in the $\mathbf{y}^{nl}$ vector. This is because the output from the nonlinear model will have nonlinear dependence on the input, which in turn depends on $\mathbf{d}^{nl}$ through eq. (8).

Any standard nonlinear equation solution method can be used to obtain the vector $\mathbf{d}^{nl}$ satisfying eq. (9). One way is by using a successive substitution or a fixed-point algorithm:

$$\mathbf{d}^{nl}_{l+1} = \mathbf{d}^{nl}_l + \beta(\mathbf{y}^{nl}_l - \mathbf{y}^{el}_l) \quad (10)$$

where l is the iteration number; β is a relaxation factor in (0, 1) used to enlarge the region of convergence.

A flow chart for the nonlinear DMC algorithm showing the fixed-point iterations is given in Fig. 3. Only the first control action $\Delta u(k)$ is implemented after the iterations (10) converge, and the iterative DMC calculations are repeated at the next sampling time. When it converges, the algorithm gives all the advantages of classical DMC. A control block diagram is shown in Fig. 4. The internal loop wrapped around the DMC block represents the fixed-point iterations. At each sampling time the switch on the control input closes with the linear model during iterations. Upon convergence it opens and closes with the plant, and the cycle is repeated at the next sampling time.

3.1. *Convergence*

Convergence of the fixed-point iterations (10) can be investigated by using contraction mapping ideas. Before stating the main result, it is necessary to introduce the pertinent notation and mathematical tools.

The nonlinear time-invariant system to be treated can be described by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$$
$$\mathbf{y} = \mathbf{g}(\mathbf{x}). \quad (11)$$

For the sake of simplicity, only SISO systems will be considered although the theorems can be extended to MIMO systems. The solution of the system (11) will be denoted as $\mathbf{X}(t, \mathbf{x}_0, \mathbf{u}, t_0)$ also called the state transition operator. For piecewise constant inputs $[\mathbf{u}(t) = \mathbf{u}_k; t_k \leqslant t < t_k + T]$ the solution operator can

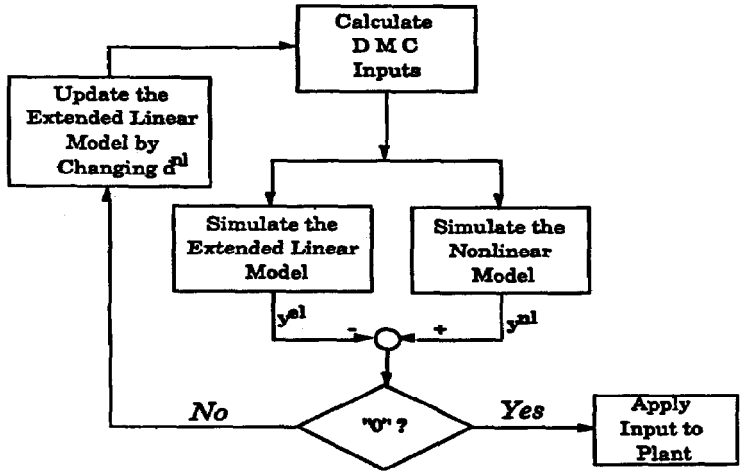


Fig. 3. The flow chart of the nonlinear DMC algorithm.

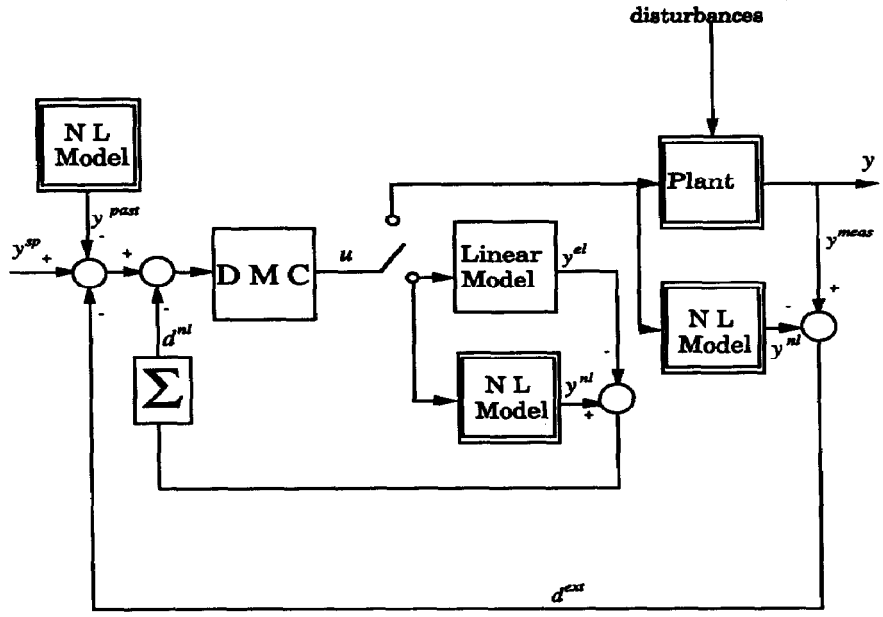


Fig. 4. Block diagram of the nonlinear DMC algorithm.

be simplified to

$$\mathbf{x}_{k+1} = \mathbf{X}(t_k + T, \mathbf{x}_k, \mathbf{u}_k, t_k) \tag{12}$$

or simply

$$\mathbf{x}_{k+1} = \mathbf{X}(t_{k+1}, \mathbf{x}_k, \mathbf{u}_k) \tag{13}$$

where

$\mathbf{x}_k$ is state vector at time t_k

$\mathbf{u}_k$ is input applied to the system from time t_k to time $t_k + T$.

The derivative of $\mathbf{X}(t_{k+1}, \mathbf{x}_k, \mathbf{u}_k)$ with respect to the state vector, $\mathbf{x}_k$, and the input, $\mathbf{u}_k$, are defined as follows (Economou, 1985):

$$\Phi_k = \left.\frac{\partial \mathbf{X}(t, \mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{x}_k}\right|_{t=t_{k+1}}$$
$$\Gamma_k = \left.\frac{\partial \mathbf{X}(t, \mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k}\right|_{t=t_{k+1}} \tag{14}$$

and are obtained from the solution of two sets of linear time-varying ODEs (Economou, 1985):

$$\dot{\Phi} = \nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x}, \mathbf{u})|_{\mathbf{x}=\mathbf{X}(t, x_k, u_k)}\Phi \qquad t \geqslant t_k \tag{15}$$
$$\Phi(t_k) = \mathbf{I};\ \Phi_k = \Phi(t_{k+1})$$

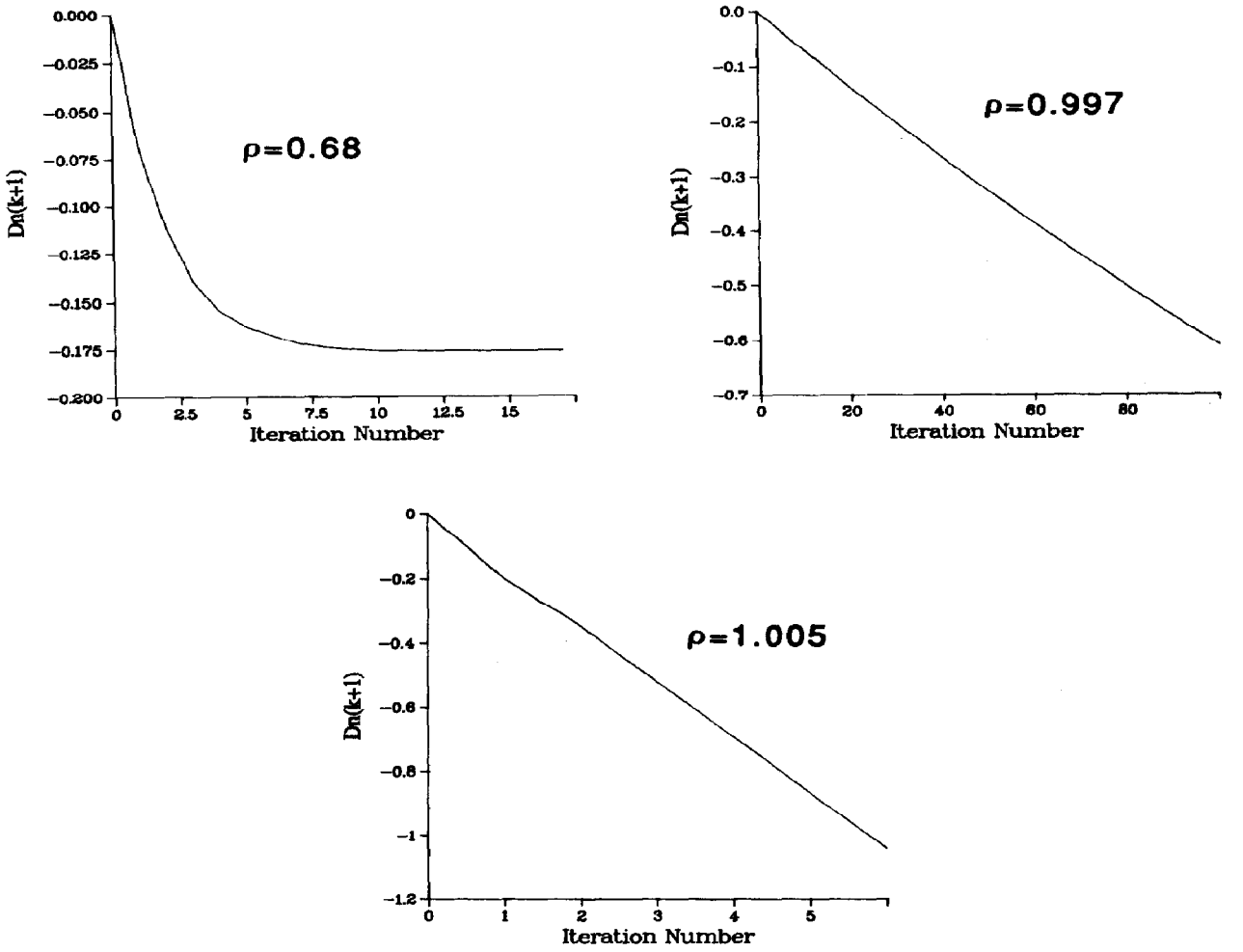


Fig. 5. Correlation between $\rho[T'(x)]$ and convergence.

and

$$\dot{\Gamma} = \nabla_{\mathbf{x}} \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x} = \mathbf{X}(t, \mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{u} = \mathbf{u}_k}} \Gamma + \nabla_{\mathbf{u}} \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x} = \mathbf{X}(t, \mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{u} = \mathbf{u}_k}} \quad t \geqslant t_k \tag{16}$$

$$\Gamma(t_k) = 0; \quad \Gamma_k = \Gamma(t_{k+1}).$$

A tool that will be needed throughout this development is the concept of a contradiction mapping and the contraction mapping theorem which are now presented.

Definition 1 (contraction mapping): *An operator* $\mathbf{F}$ *in a Banach space* X *with norm* $\|\cdot\|$ *is called a contraction mapping on the closed set* $S \subset X$ *if there exists a real number,* α, $0 < \alpha < 1$, *such that*

$$\|\mathbf{F}(\mathbf{x}_1) - \mathbf{F}(\mathbf{x}_2)\| \leqslant \alpha \|\mathbf{x}_1 - \mathbf{x}_2\| \ \forall \mathbf{x}_1, \mathbf{x}_2 \in S. \tag{17}$$

This definition is not very useful from a computational point of view since it says nothing about how to find the contraction constant, α. The following lemma is useful in this context.

Lemma 1: *Let the operator* $\mathbf{F}$ *be differentiable on a closed, convex set* S. *Then,* $\mathbf{F}$ *is a contraction of* S *if and only if:* $\|\nabla_{\mathbf{x}} \mathbf{F}(\mathbf{x})\| < 1$ *for all* $\mathbf{x}$ *in* S.

This result follows from the mean value theorem. A complete proof may be found, for example, in Economou (1985).

Theorem 1 (contraction mapping theorem): *If* $\mathbf{F}$ *is an operator in a Banach space* X *and is a contraction mapping of the closed convex* $S \subset X$, *and* $\mathbf{F}$: $S \to S$, *then:*

(1) $\mathbf{F}$ *has a fixed point* $\mathbf{x}^*$ *in* S;
(2) $\mathbf{x}^*$ *is the unique fixed point of* $\mathbf{F}$ *in* S;
(3) *The sequence of successive approximations defined by*

$$\mathbf{x}_{l+1} = \mathbf{F}(\mathbf{x}_l)$$

converges to $\mathbf{x}^*$ *with*

$$\|\mathbf{x}_l - \mathbf{x}^*\| \leqslant \alpha^l \|\mathbf{x}_0 - \mathbf{x}^*\| \tag{18}$$

where α *is the contraction constant of* $\mathbf{F}$.

Noting that the iterations in eq. (10) describe a sequence of successive approximations, we may postulate the followng theorem.

Theorem 2 [convergence of the iterations (10)]: *Let* $\mathbf{d}_i^{nl}$ *denote the* l*th iterate of the vector* $\mathbf{d}^{nl}$ *at the current sampling time. Then the operator* $\mathbf{F}(\mathbf{d}^{nl})$ *is given by:*

$$\mathbf{d}_{i+1}^{nl} = \mathbf{F}(\mathbf{d}_i^{nl}) = \mathbf{d}_i^{nl} + \beta(\mathbf{y}_i^{ni} - \mathbf{y}_i^{el}). \tag{19}$$

Furthermore, let $\mathbf{F}$: $S \to S$, S *Banach.*
The following results hold:

(i) $\mathbf{F}'(\mathbf{d}^{nl})$ *is a matrix whose* ij*th element is given by*

$$\begin{aligned}[\mathbf{F}'(\mathbf{d}^{nl})]_{i,j} = [&(1-\beta)\mathbf{I} - \beta\mathbf{A}(\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}\mathbf{A} \\ &+ \boldsymbol{\Lambda}^T\boldsymbol{\Lambda})^{-1}\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}]_{i,j} \\ &+ \beta\nabla_{\mathbf{x}}\mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_i}\boldsymbol{\Psi}_{i,j}(T)\end{aligned} \tag{20}$$

where $\mathbf{T}$ *is the sampling time and*

$$\begin{aligned}\dot{\boldsymbol{\Psi}}_{i,j} = &\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1},\,\mathbf{x}_{i-1},\,\mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}\boldsymbol{\Psi}_{i,j} \\ &- \nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1},\,\mathbf{x}_{i-1},\,\mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}[\mathbf{1}_L(\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}\mathbf{A} \\ &+ \boldsymbol{\Lambda}^T\boldsymbol{\Lambda})^{-1}\mathbf{A}^T\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}]_{i,j} \\ &\left.\begin{aligned}\boldsymbol{\Psi}_{i,j}(0) &= \boldsymbol{\Psi}_{i-1,j}(T) \quad \text{for} \quad i = 2, \ldots, P \\ \boldsymbol{\Psi}_{1,j}(0) &= 0\end{aligned}\right\} \\ &j = 1, \ldots, P\end{aligned} \tag{21}$$

$$x_l = \mathbf{x}_{k+l};\ u_l = \mathbf{u}_{k+l}$$

$$\mathbf{1}_L \equiv \underbrace{\begin{bmatrix} 1 & & & 0 \\ 1 & \ddots & & \\ \vdots & \ddots & \ddots & \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}}_{M} \Bigg\} P. \tag{22}$$

(ii) *If* $\|\mathbf{F}'(\mathbf{d}^{nl})\| < 1$ *for all* $\mathbf{d}^{nl}$ *in* S, *then there exists a unique* $\mathbf{d}^{nl}$ *in* S *satisfying*

$$\mathbf{y}^{nl}(\mathbf{d}^{nl}) = \mathbf{y}^{el}(\mathbf{d}^{nl})$$

which can be found via the fixed-point iteration (10).

Proof: First note that the fixed point of $\mathbf{F}$ occurs when $\mathbf{y}^{nl}(\mathbf{d}^{nl}) = \mathbf{y}^{el}(\mathbf{d}^{nl})$, and that all conditions of the contraction mapping theorem are satisfied if $\mathbf{F}$ is a contraction. □

Theorem 2 is general but difficult to apply. One of the limitations of this theorem is finding a closed convex set, S, such that F maps S into itself. Another difficulty is finding a compatible norm such that $\|\mathbf{F}'(\mathbf{d}^{nl})\|$ is less than one for all $\mathbf{d}^{nl}$ in S. If a particular norm fails the test, another norm may satisfy it. Therefore, a search over all compatible norms might have to be performed. For these reasons, it is necessary to find other results that, although possibly weaker, yield a better understanding of the convergence of the iterations.

In what follows, a divergence condition is obtained by using the spectral radius.

Lemma 2 (Rall, 1969): *The spectral radius* ρ *of a linear operator* $\mathbf{L}$ *is less than or equal to any other compatible norm of the operator, i.e.*

$$\rho(\mathbf{L}) = \max_i |\lambda_i(\mathbf{L})| \leqslant \|\mathbf{L}\| \tag{23}$$

where λ_i *denotes the eigenvalues.*

Lemma 3 (Rall, 1969): *For a linear operator, there exists a compatible norm which is arbitrarily close to the spectra radius of the operator.*

Corollary 1 (corollary to theorem 2): *If the spectral radius of* $\mathbf{F}'(\mathbf{d}^{nl})$ *is greater than one for all* $\mathbf{d}^{nl}$ *in* S, *then the iterations will diverge from almost every initial point. That is, the set of initial conditions for which the iterations will converge has measure zero.*

Proof. If the spectral radius of the derivative of the operator at the fixed point is greater than one, then the fixed point is not an attractor. Therefore, the set of initial conditions for which the iterations converge is denumerable, thus having measure zero. □

We have obtained a result that is easier to check since it is not required to search over all compatible norms. At the same time, this result is weaker in that only a condition for divergence (not convergence) of the iterations is obtained. However, even though $\rho[\mathbf{F}'(\mathbf{d}^{nl})] > 1$ for some $\mathbf{d}^{nl}$ is not a necessary condition for the iterations to diverge, it should be taken as a warning of possible convergence problems. This is illustrated in the following example.

Example: Figure 5 shows the result of three different calculations of $\mathbf{d}^{nl}$ performed at different conditions. Only the first element in the $\mathbf{d}^{nl}$ vector is shown. Note that for each iteration, a value of $\rho[\mathbf{F}'(\mathbf{d}^{nl})]$ can be computed. Figure 5(a) shows a calculation where the average of $\rho[\mathbf{F}'(\mathbf{d}^{nl})]$ was 0.68. It can be observed that $\mathbf{d}^{nl}(k+1)$ converged quickly after 12 iterations or so. On the other hand, when the average value of $\rho[\mathbf{F}'(\mathbf{d}^{nl})]$ was 0.997, $\mathbf{d}^{nl}(k+1)$ did not converge even after 100 iterations. Furthermore, when the average of $\rho[\mathbf{F}'(\mathbf{d}^{nl})]$ was 1.005, the iterates of $\mathbf{d}^{nl}(k+1)$ diverged quickly and after six iterations the program was terminated.

Further understanding on the convergence of the iterations can be gained by investigating the effect of an arbitrarily large sampling time and weight on the

input and a small "relaxation factor" β on certain types of plants: those which are globally asymptotically stable. It will also be assumed that Γ is equal to identity and that $\mathbf{A}$ is equal to $\lambda\mathbf{I}$. Before continuing, the definition of the steady-state gain of a globally asymptotically stable nonlinear system is needed.

Lemma 4 (calculation of the steady-state gain of the nonlinear system): *Let the gain of a SISO nonlinear system be defined as the change in the output caused by a change in the input: $\partial y_{k+1}/\partial u_k$, where the subscript k denotes sampling time in this instance. The steady-state gain of a globally, asymptotically stable system* (11) *at the equilibrium point* $(\mathbf{x}_{k+1}, \mathbf{u}_k)$ *is given by:*

$$\left.\frac{\partial \mathbf{y}_{k+1}}{\partial \mathbf{u}_k}\right|_{T\to\infty} = -\frac{\partial \mathbf{g}(\mathbf{x}_{k+1})}{\partial \mathbf{x}_{k+1}}\left[\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}}\right]^{-1} \times \nabla_{\mathbf{u}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}}.$$

Proof: We will first compute $\partial \mathbf{y}_{k+1}/\partial \mathbf{u}_k$ and then take the limit as the sampling time goes to infinity:

$$\frac{\partial \mathbf{y}_{k+1}}{\partial \mathbf{u}_k} = \underbrace{\frac{\partial \mathbf{g}[\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)]}{\partial \mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)}}_{\equiv \mathbf{C}_{k+1}} \underbrace{\frac{\partial \mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)}{\partial \mathbf{u}_k}}_{=\Gamma_k} = \mathbf{C}_{k+1}\Gamma_k. \tag{24}$$

Since the system is globally asymptotically stable and the sampling time is arbitrarily large, when $\mathbf{u}_k$ is applied to the system the state will reach a steady-state solution, $\mathbf{x}_{k+1}$. Therefore, the Jacobian of $\mathbf{f}(\mathbf{x},\mathbf{u})$ with respect to $\mathbf{x}$ evaluated at this equilibrium point will have negative eigenvalues. This means two things, that Γ in eq. (16) will reach a steady state for large sampling times and that the Jacobian of $\mathbf{f}(\mathbf{x},\mathbf{u})$ is invertible. Thus

$$\Gamma_k|_{T\to\infty} = -\left[\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}}\right]^{-1} \times \nabla_{\mathbf{u}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}} \tag{25}$$

and

$$\left.\frac{\partial y_{k+1}}{\partial u_k}\right|_{T\to\infty} = -\frac{\partial \mathbf{g}(\mathbf{x}_{k+1})}{\partial \mathbf{x}_{k+1}} \times \left[\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}}\right]^{-1} \times \nabla_{\mathbf{u}}\mathbf{f}(\mathbf{x},\mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_{k+1},\mathbf{x}_k,\mathbf{u}_k)\\ \mathbf{u}=\mathbf{u}_k}} = \mathbf{C}_{k+1}\Gamma_k|_{T\to\infty} \tag{26}$$

which proves the lemma. □

Theorem 3 (effect of sampling time, relaxation factor and weights of the input on convergence): *Let the nonlinear system to be controlled be described by the dynamic equations shown in eqs* (11) *where* $\mathbf{f}$ *and* $\mathbf{g}$ *are smooth vector fields with bounded derivatives. If the open-loop nonlinear system is globally asymptotically stable for all feasible values of* $\mathbf{u}$, *then the iterations* (10) *will converge if the sampling time and the weight on the input are chosen large enough and the relaxation factor,* β, *is selected small enough.*

Proof: See Appendix B. □

The successive substitution or the fixed-point algorithm can converge slowly. As an alternative the secant method is used to solve the nonlinear equations (9) with superlinear convergence instead of just linear convergence. This is discussed next.

Consider $M = P = 1$ (i.e. one step ahead) and rewrite eq. (9) as

$$\mathbf{h}(\mathbf{d}^{nl}) = \mathbf{y}^{nl}(\mathbf{d}^{nl}) - \mathbf{y}^{el}(\mathbf{d}^{nl}) = 0. \tag{27}$$

Then the secant method tries to solve the nonlinear equation (27) by the following recursion

$$\mathbf{d}^{nl}_{l+1} = \mathbf{d}^{nl}_l - \left[\frac{\mathbf{h}(\mathbf{d}^{nl}_l) - \mathbf{h}(\mathbf{d}^{nl}_{l-1})}{\mathbf{d}^{nl}_l - \mathbf{d}^{nl}_{l-1}}\right]^{-1}\mathbf{h}(\mathbf{d}^{nl}_l). \tag{28}$$

This one-dimensional secant iteration is a specific case of the more general Broyden's method (Dennis and Schnabel, 1983). The Dennis–Moré theorem (Dennis and Moré, 1974) gives the necessary and sufficient conditions for the superlinear convergence of secant iterations. However, compared to the fixed-point algorithm given above, they are impractical to verify and are not elaborated here.

Note that in general $n + 1$ initial guesses for $\mathbf{d}^{nl}$ are needed to start off the n-dimensional secant iterations. These can be obtained from the fixed-point iterations discussed earlier. The nonlinear DMC with both secant and fixed-point iterations have been successfully applied to continuous stirred tank and batch reactors (Peterson, 1990). As expected, the secant method was found much more effective than the fixed-point algorithm. Section 4 presents the results of the secant method applied to a semibatch reactor.

3.2. *Calculation of* $\mathbf{y}^{past}$, $\mathbf{d}^{ext}$ *and* $\mathbf{A}$

By definition, the vector $\mathbf{y}^{past}$ is the output at each sampling time in the future with all the future input changes set equal to zero. In the linear DMC, $\mathbf{y}^{past}$ is calculated from the first two terms in eq. (3) based on the past inputs only. The nonlinear equivalent of $\mathbf{y}^{past}$ is to integrate the nonlinear state equations $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u})$ forward from an initial condition of the current modeled states, $\mathbf{x}^m(k)$, with a constant input, $\mathbf{u} = \mathbf{u}(k-1)$, over the future horizon P (García, 1984). Next the output is reconstructed from the integrated states using the algebraic output equation $\mathbf{y} = \mathbf{g}(\mathbf{x})$.

The disturbance effect, $\mathbf{d}^{ext}$ is estimated similar to the linear DMC except that instead of the linear model in Fig. 2 a nonlinear model is simulated in parallel to the plant as shown in Fig. 4. The state is integrated from its initial value (one sampling time

ago), $\mathbf{x}(k-1)$, to the present value, $\mathbf{x}^m(k)$, using the past, known value of the control input. The effects of the past inputs on the current output and the current disturbance estimate can then be calculated from

$$\mathbf{y}^m(k) = \mathbf{g}[\mathbf{x}^m(t)] \tag{29}$$

$$\mathbf{d}^{ext}(k) = \mathbf{y}^{meas}(k) - \mathbf{y}^m(k). \tag{30}$$

Like in linear DMC, $\mathbf{d}^{ext}$ is assumed to remain constant over the prediction horizon P. Note that when a perfect nonlinear model is used, $\mathbf{d}^{ext}$ is due to external disturbances only and does not contain model/plant mismatch due to linear approximation. If the nonlinear model is not perfectly known, $\mathbf{d}^{ext}$ will contain modeling errors in addition to external process disturbances.

The dynamic matrix **A** or the step-response coefficients are calculated either from the linear model obtained after linearizing the nonlinear model at a given steady state or by perturbing the nonlinear dynamic model by a unit step. For our batch reactor application, there is no "steady state", and the dynamic matrix **A** is computed by a procedure given in Section 4.

Stability

The stability of the closed-loop system can be characterized in a similar fashion by using contraction mapping ideas. This time we look at the nonlinear operator **N**

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \mathbf{u}_{k+1}\end{bmatrix} = \mathbf{N}\begin{bmatrix}\mathbf{x}_k\\ \mathbf{u}_k\end{bmatrix}$$

where k denotes the sampling time. The closed-loop system is stable if the norm of the gradient of **N** is less than one for all $(\mathbf{x}, \mathbf{u})$ in a closed convex set S, where $\mathbf{N}: S \to S$.

The following theorem gives sufficient conditions for stability. Even though the theorem's applicability may be limited, it does yield valuable understanding of the effect of various parameters on the closed-loop stability of the control algorithm.

Theorem 4 (nominal closed-loop stability of some globally asymptotically stable systems): *Suppose that the system to be controlled is globally asymptotically stable for all feasible inputs. Furthermore, suppose that the following is valid:*

(1) *The steady-state gain of the system does not change sign.*
(2) *The weight on the change of the input is larger than zero.*
(3) *The sampling time is "long enough".*
(4) *The input horizon,* **M**, *is set to one.*
(5) *The set point is constant in the prediction horizon.*

Then the closed-loop system is guaranteed to be nominally stable.

Proof. See Appendix C.

4. APPLICATION TO A SEMIBATCH POLYMERIZATION REACTOR

The nonlinear DMC algorithm described above was applied, in simulation, to the control of a semibatch reactor for the polymerization of methyl methacrylate. This particular system was chosen because of its industrial importance, its strong nonlinearities, and because it illustrates the capabilities of nonlinear DMC in handling control challenges unique to batch operations. In what follows the reaction system will be briefly described, and the application of nonlinear DMC to both single-loop control and multivariable control of this reactor will be demonstrated.

4.1. *Polymerization reactor model*

The system chosen for study is the semibatch, free radical polymerization of methyl methacrylate. The physical system consists of a jacketed reactor with inlet flow of initiator and solvent as shown in Fig. 6. The initiator used is benzoyl peroxide dissolved in ethyl acetate solvent. Methyl methacrylate monomer is charged to the reactor initially and is not added during the reaction. The manipulated variables are initiator flow, u_1, and cooling jacket temperature, u_2. The controlled outputs are the reaction temperature, y_1, and the number average molecular weight, y_2 defined by

$$y_2 = \frac{\sum_{i=2}^{\infty} M_i n_i}{\sum_{i=2}^{\infty} n_i} \tag{31}$$

where M_i is the molecular weight of molecule i and n_i is the number of molecules with weight M_i (Billmeyer, 1984).

The modeling equations are adopted from Houston and Schork (1987) and Tanner *et al.* (1987). They consist of seven nonlinear state equations describing the energy balance, the first two moments of the molecular weight distribution, the mass balances for the monomer, initiator, solvent, and the total volume. Since it is very detailed, the model is not repeated here and can be found in Peterson (1990). Correlations for the gel and glass effects are taken from Schmidt and Ray (1981). These account for the variation in termination and propagation rates with viscosity. The

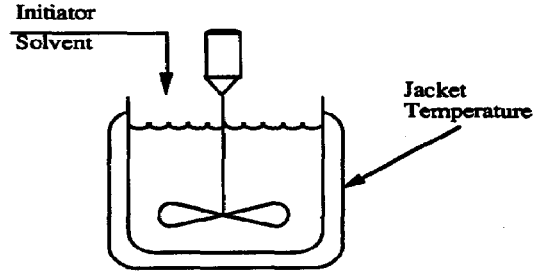


Fig. 6. Semibatch reactor.

gel effect is responsible in part for the autocatalytic nature of the polymerization, and causes control problems due to the strong nonlinearities and open-loop instabilities it introduces.

This model poses some unique problems and challenges for applying nonlinear predictive cotrol. First, the batch nature of the process means there is really no steady state. Control action must be taken at every sample time over a range of reactor conditions to control the outputs. Computational efficiency is a must. Second, the model is much larger and more complicated than those studied before for nonlinear model-predictive control. It has seven states, two inputs (jacket temperature and initiator flow) and two outputs (temperature and number average molecular weight, NAMW). NAMW is a nonlinear function of two state variables. The gel effect correlation also adds complexity to the model since it is not a smooth function.

The main control objective for this reactor is to reach a desired NAMW at the end of the batch in minimum time.

4.2. *DMC models and control algorithms used in simulations*

In all the simulations to follow, the secant method is used in the iterative DMC calculations since it proved to be faster than the successive substitutions. For temperature control the DMC algorithm uses a step-response model. Since the batch reactor does not have a true steady state, the step coefficients are determined as shown in Fig. 7. Keeping a constant input and integrating forward gives the unperturbed output $\mathbf{y}^{\mathbf{past}}$. Stepping the input to a new value and integrating into the future gives the perturbed output, $\mathbf{y}^{\mathbf{per}}$. The step-response coefficients are then calculated from

$$a_i = \frac{[y^{\mathrm{per}}(k+i) - y^{\mathrm{past}}(k+i)]}{\Delta u}. \tag{32}$$

In the case of molecular weight control, open-loop simulation revealed that the initiator had very different effects on the molecular weight depending on the time it was introduced to the batch reactor (Peterson, 1990). This was best observed by adding an initiator *pulse* of 12 min duration (i.e. one sampling time) at different times during the reaction rather than introducing a step input which lasts over the entire time of reaction. Consequently, instead of the step model an impulse model of the following form (Prett and García, 1988) is used in the DMC algorithm for the molecular weight.

$$y(k) = \sum_{i=1}^{N} h_i u(k-i). \tag{33}$$

The impulse coefficients, h_i, are calculated similar to the step-response coefficients by integrating the nonlinear system with a pulse input instead of a step. This similarity allows use of essentially the same secant method for the impulse response model.

For the temperature loop we use $M = P = 1$. For the molecular weight loop, $M = 1$ and at each sampling time the prediction horizon is set equal to the final batch time, t_f, which is defined as the time it takes to reach 85% monomer conversion. Since t_f is not known *a priori*, it is calculated on-line and the prediction horizon (P) is updated accordingly at each sampling time. The nonlinear DMC tries to match the molecular weight with its target set point only at the end of the batch. The batch terminates when 85% conversion is reached.

Finally, it is important to mention that the step-(impulse-) response coefficients were updated from one sampling time to another. After the secant iterations converge and the control $\Delta u(k)$ is implemented, the simulated nonlinear output are used to update either the step coefficients a_1 for the temperature control

$$a_1 = \frac{y^{nl}(k+1) - y^{\mathrm{past}}(k+1)}{\Delta u(k)} \tag{34}$$

or the impulse coefficients h_p for molecular weight targeting

$$h_p = \frac{y^{nl}(k+P) - y^{\mathrm{past}}(k+P)}{u(k)} \tag{35}$$

where $k + P$ denotes the end of the batch. Note that only one coefficient needs to be updated. This is because $M = 1$, and only one future value for each output is predicted [$y^{nl}(k+1)$ for temperature and $y^{nl}(k+P)$ for molecular weight].

4.3. *Single-loop control*

Often the control objective for a batch polymerization reactor is to make the temperature of the

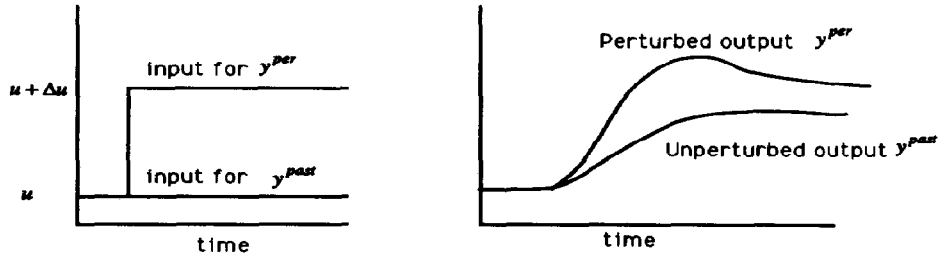


Fig. 7. Estimating step-response coefficients from an unsteady state nonlinear model.

reactor follow some predetermined trajectory. In the following simulations the temperature is regulated around its initial set point and ramped up by adjusting the jacket temperature.

Figure 8 shows the results of the nonlinear DMC with the secant method applied with a sampling time of 12 s, $M = P = 1$, and no plant/mode mismatch for regulation around 340 K. Note that the temperature is perfectly controlled matching the step point at each sampling time. The intersample deviations appear as shaded areas due to small sampling time relative to the plotted time scale. Figures 9 and 10 show the control input (dimensionless jacket temperature) and conversion. Three distinct regions are visible.

In region I, the reaction rate is decreasing slowly as reactants are consumed. The model predicts a reduced

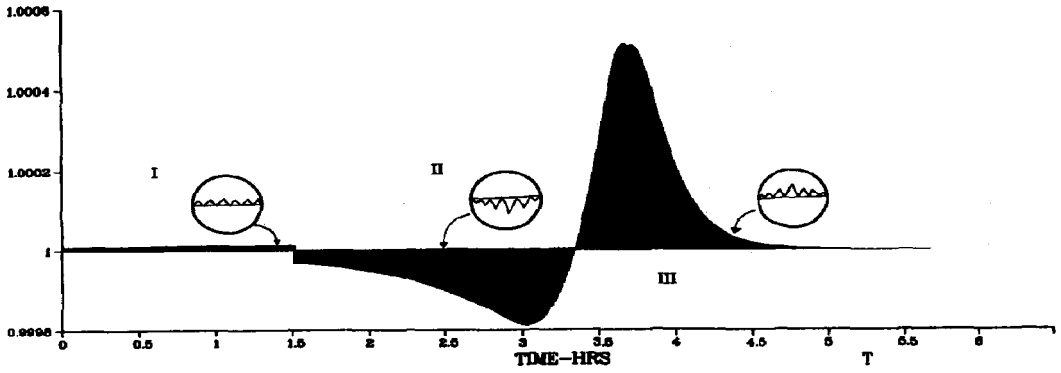


Fig. 8. Dimensionless temperature using secant algorithm. The actual temperature oscillations are at most a few tenths of a degree.

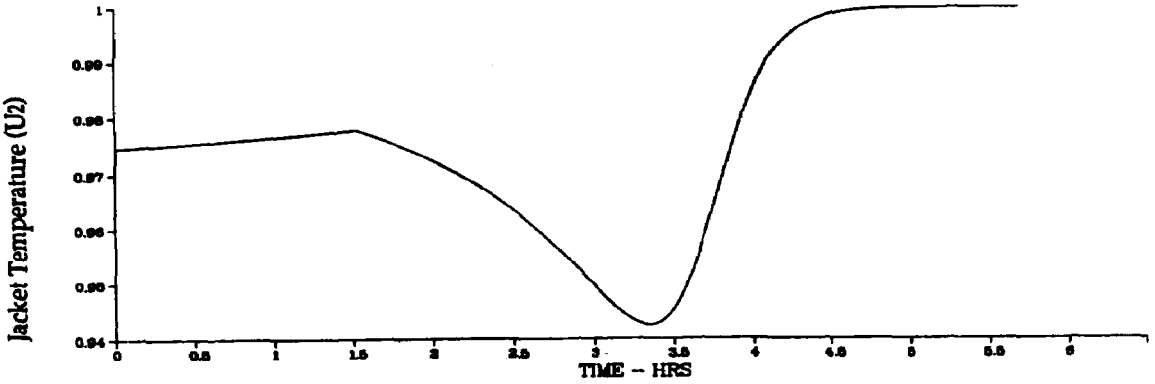


Fig. 9. Dimensionless jacket temperature.

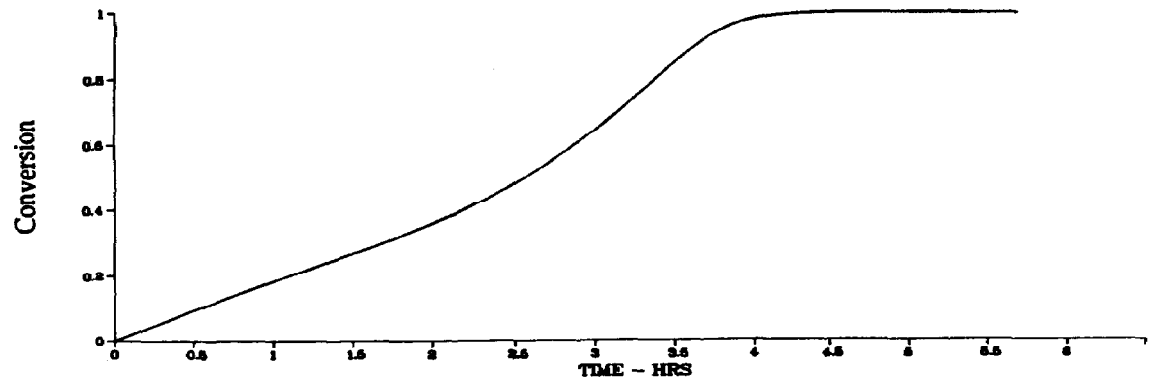


Fig. 10. Monomer conversion.

heat generation rate and increases the jacket temperature (see Fig. 9) which increases the reaction temperature during the first part of the sampling period. The decreasing reaction rate brings the temperature down to the set point at the beginning of the next sampling. Since this is a batch reaction with constantly changing conditions and no real steady state, the intersampling oscillations of the output are unavoidable with a discrete controller.

In region II, the gel effect causes the reaction to accelerate, creating more heat to be removed. Thus, a trend just opposite to that of region I is observed. The intersample deviations are more pronounced in this region due to the gel effect. When the reactor is most exothermic (at 3 h), if the calculated control action is kept constant for more than two sampling times, the reactor experiences a runaway. This makes frequent, precise control moves necessary to operate in this region.

In region III the depletion of reactants overcomes the gel effect and the reaction rate slows again. The intersampling oscillations follow the same trend as in region I. When all of the monomer is reacted, the reactor and jacket are both at the same temperature. Generally the reaction is stopped before reaching complete conversion. In this case we have chosen to stop the reactor arbitrarily at 85% conversion.

The results for regulation along a temperature trajectory are shown in Fig. 11. The temperature is initially regulated at 340 K. At one hour the set point is ramped to reach 343 K at two hours. After two hours the temperature is regulated at 343 K. The trajectory is followed perfectly, reaching the set point at the end of each 12 s sampling time. Trends similar to Fig. 8 appear after 2 h in the simulation. The temperature oscillates below the set point when the gel effect causes the reaction to become highly exothermic, and it oscillates above the set point when the reaction slows down due to monomer depletion.

Figure 12 shows the result of applying PID control to the batch polymerization reactor model with a sampling time of 12 s and the same set-point trajectory. Even though the controller is sluggish early in the polymerization, it eventually turns unstable after 2 h when the reaction enters the extreme exothermic region. This is not surprising since calculations indicate that the system gain increases by approximately two and a half orders of magnitude as the reaction enters the autocatalytic (gel effect) region.

4.4. *Multivariable control*

In this section the nonlinear DMC with the secant algorithm is used to solve control problems (for the semibatch polymerization reactor) associated with a combination of regulation and targeting of terminal (end of batch) conditions.

Attempts to control average molecular weight with jacket temperature (with all initiator added at the beginning of the reaction) were unsuccessful since the reactor temperature is not directly controlled, and eventually turns unstable (Peterson, 1990). Therefore, one has to control molecular weight (y_2) *and* temperature (y_1) by manipulating the jacket temperature (u_2)

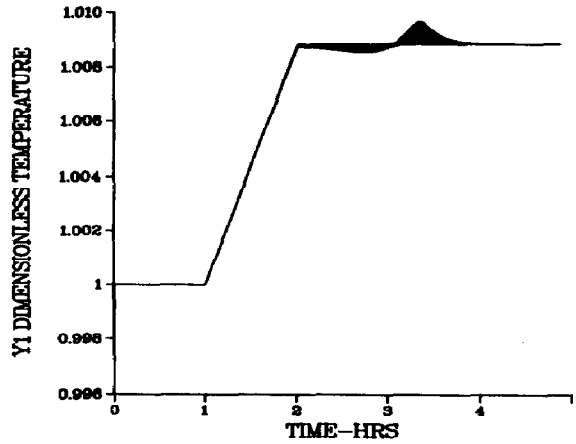
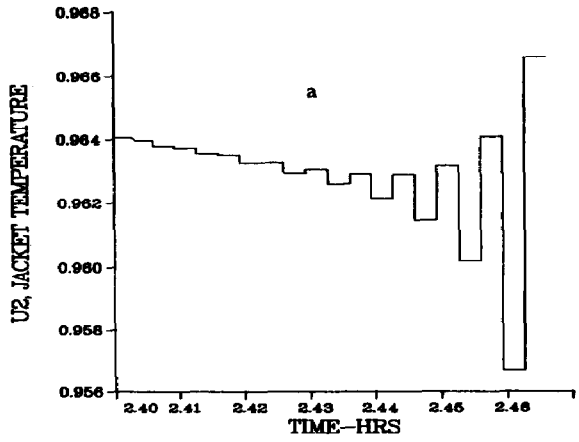


Fig. 11. Temperature control with the secant algorithm.



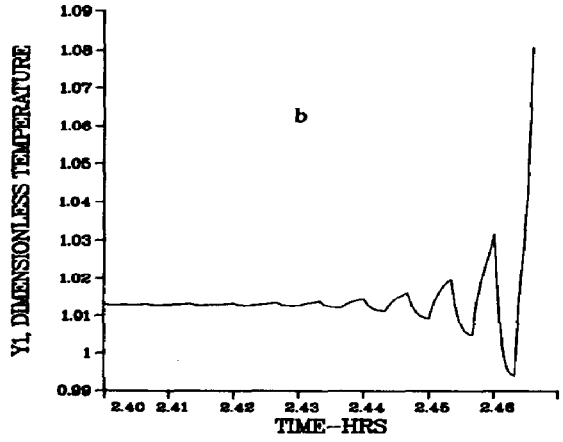


Fig. 12. Instability in the PI controller.

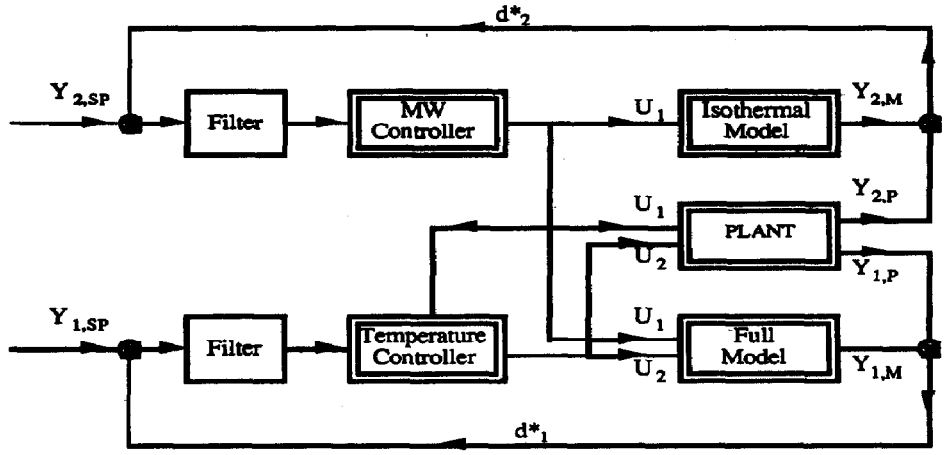


Fig. 13. MIMO block diagram.

and the initiator flow (u_1). Instead of developing a multivariable nonlinear DMC controller for the full 2×2 nonlinear input–output operator, a different approach which exploits the model structure is taken. Specifically, examination of the model reveals that the jacket temperature appears only in the state equation for the reactor temperature. Therefore, if the reactor temperature could be "perfectly" regulated, the molecular weight could be controlled by using the nonlinear *isothermal* mode. Temperature would then be controlled by using the full model with initiator calculated from the isothermal model. The block diagram is shown in Fig. 13.

The molecular weight and temperature controllers are two separate nonlinear DMC controllers. In the temperature loop there is no plant/model mismatch since the model is the full perfect model. However, the molecular weight loop has a mismatch since control is based on the isothermal model, i.e. it is assumed that the temperature loop is perfectly regulating. As we have seen in the single-loop simulations, temperature is regulated tightly; therefore, this mismatch is not expected to be serious. In addition, first-order filters are incorporated into each loop to achieve robustness against any model/plant mismatches.

Finally note that with this control structure different sampling times can be easily incorporated into the different control loops. For example, since molecular weight feedback is often only available from chromatographic techniques with long measurement deadtimes, a sampling time of 12 min is used for the molecular weight controller, while a sampling time of 9 s is used for the temperature controller.

The objective of the molecular weight controller is to calculate the amount of fed initiator such that the average molecular weight is equal to the desired value at the end of the batch. This can be accomplished by regulating the average molecular weight (i.e. keeping the average molecular weight at set point at all times during the batch), or by manipulating the inputs in such a way as to bring the average molecular weight to its target at the end of the reaction (targeting). The first alternative (regulation) will force the average molecular weight to be on target at all times, including the final time. Regulation may be desirable since it will provide a product which not only has the correct average molecular weight, but one that also has a low polydispersity (narrow distribution). Targeting, on the other hand, will produce a product with the desired average molecular weight, but with a larger polydispersity (wider distribution).

When regulation of average molecular weight *via* nonlinear DMC is implemented, the controller first calculates addition of initiator to the reactor. This is because the uncontrolled average molecular weight decreases with decreased monomer concentration early in the polymerization. Later, as the uncontrolled molecular weight tends to increase (due to the gel effect), the controller calculates a negative initiator addition, i.e. removal of initiator from the reactor, a physical impossibility. The unidirectional nature of the manipulation (initiator can be added, but not removed), makes regulation of average molecular weight by initiator feed impossible. Instead, the targeting problem is solved by the nonlinear DMC using the secant method to predict and control the average molecular weight at the end of the batch. The end of the batch is defined for this study, to be the time when the monomer conversion reaches 85%.

Figure 14 shows when each simulated batch reaches 85% conversion. It can be seen that the molecular weight at 85% conversion increases almost linearly with the height of the initiator pulse, indicating that the secant method should yield good results for this problem.

In the following results, the MIMO control structure in Fig. 13 is implemented. The secant algorithm is used in both controllers. A first-order filter is used in the molecular weight controller, while the temperature controller is unfiltered. In the molecular weight controller the initiator pulse which causes the molecular weight to reach the filtered set point at the end of

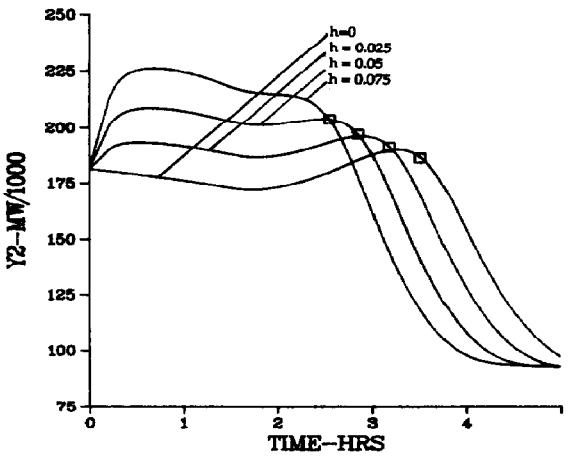


Fig. 14. Response of MW to a 12 min pulse in initiator flow at $t = 0$. The NAMW at 85% conversion for each simulation is indicated by the square marker. The height of the pulse is h.

the batch (85% conversion) is calculated for a 12 min sampling time. In other words $M = 1$ and P is set equal to the time it takes for 85% conversion. The temperature controller predicts and corrects one step ahead with $M = P = 1$.

Figure 15 shows the results of this MIMO control structure with a molecular weight set point of 195,000 and the temperature regulated at 340 K. Figure 15(b) shows the dimensionless reactor temperature as a function of time. In addition to the previously described oscillations due to monomer depletion and autoacceleration in single-loop temperature control (refer to Fig. 8), a new relationship with initiator addition can also be seen. As initiator is added by the molecular weight controller, the reaction rate increases, making it necessary to decrease the jacket temperature in order to keep the reaction temperature constant. Figure 16(a) shows that the slope of the jacket temperature decrease is approximately proportional to the initiator addition. This also increases the intersample oscillations in size [compare region I in Figs 8 and 15(b)]. Figure 16(b) shows that the desired molecular weight at 85% conversion (195,000) was achieved. Similar results are obtained at other temperature and average molecular weight set points.

These simulations show that the nonlinear DMC with the secant algorithm is very effective in calculating inputs for temperature and molecular weight control. In the absence of model/plant mismatch, however, this control algorithm is nothing more than the calculation of an open-loop input profile. The utility of the proposed method can be clearly illustrated when the process is simulated with an inihibitor disturbance to show the feedback properties of the algorithm in the presence of uncertainty.

An inhibitor is added to the monomer to prevent thermal initiation and polymerization during transit and storage. This inhibitor may or may not be removed before polymerization. Either way, the level of inhibition for a given amount of monomer is often unknown. An imprecise estimate of the inhibitor level can distort the expected open-loop molecular weight response. Figure 17 shows the results of the secant algorithm with and without molecular weight feedback. To simulate the effect of residual inhibitor on the polymerization, the initiator efficiency of the *plant* was assumed to be 0.1 for the first two sampling intervals (24 min). After two sampling periods, the inhibitor was assumed to have been consumed in reaction with the initiator, and the initiator efficiency was set to 0.5 for the rest of the polymerization. The initiator efficiency of the *model* remained constant at 0.5. Figure 17 shows three molecular weight profiles. Profile 1 shows control at a temperature of 343 K and an average molecular weight set point of 175,000 with *no inhibitor disturbance*. Profile 2 shows that what happens when there is an inhibitor disturbance and the control inputs are not updated. A different profile results with the final molecular weight about 2,000 units over set point. Profile 3 shows the effect of disturbance feedback. With corrective feedback, the

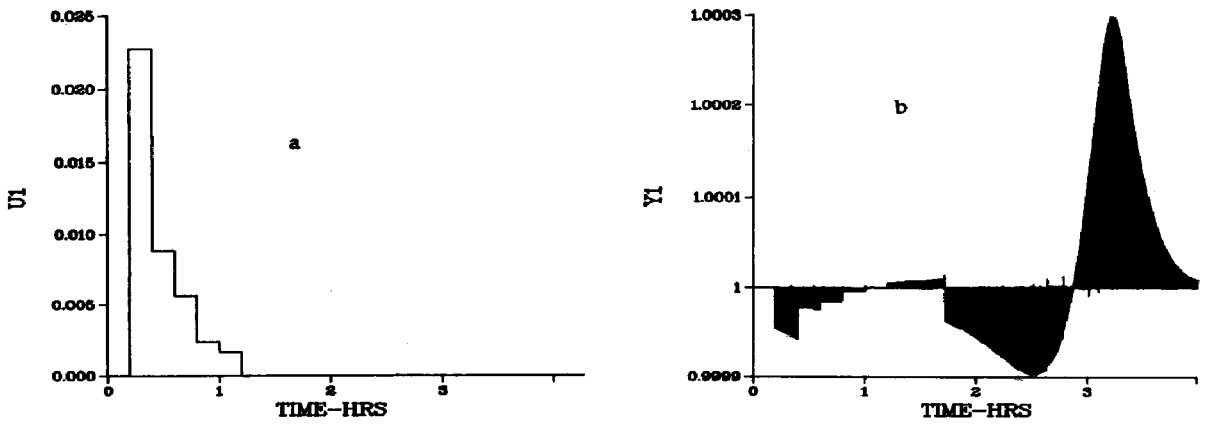


Fig. 15. MIMO simulations: (a) shows the initiator addition; (b) shows the dimensionless temperature.

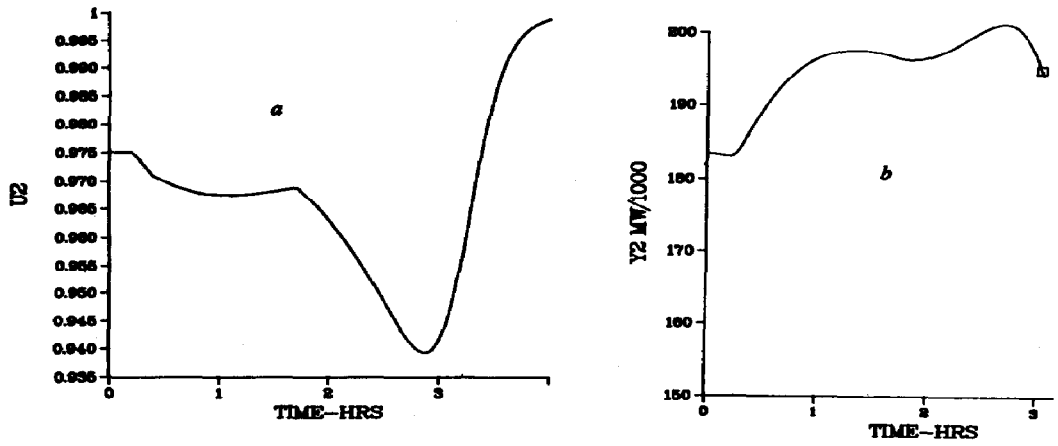


Fig. 16. MIMO simulations: (a) shows the jacket temperature; (b) shows the molecular weight profile.

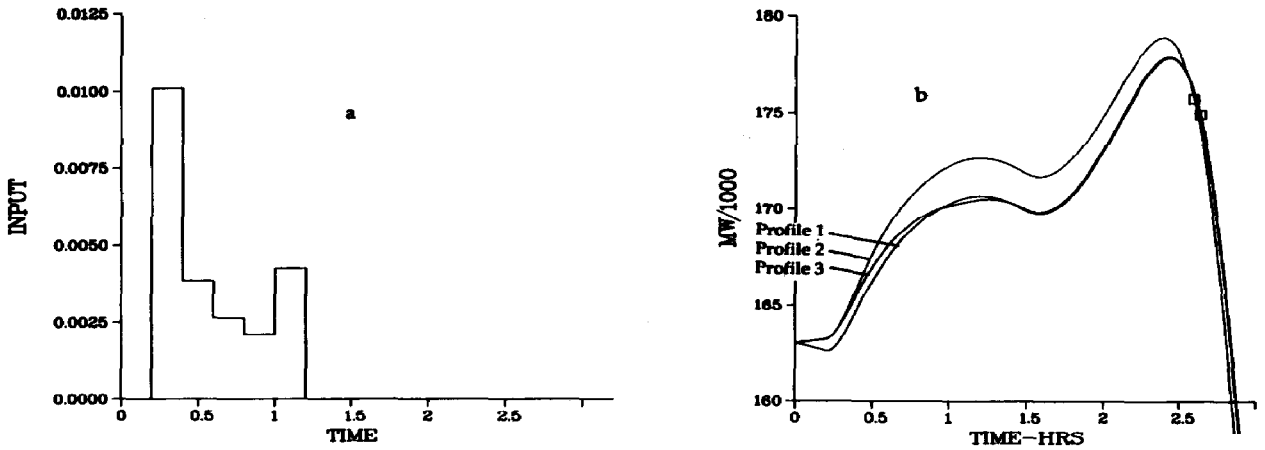


Fig. 17. The result of molecular weight feedback. Profile 1 is for control with a perfect model. Profile 2 indicates the results when the outputs from profile 1 are applied to a model with an inhibitor disturbance. Profile 3 shows feedback correction of the disturbance. (a) shows the initiator associated with profile 1.

molecular weight profile is brought back to follow a path similar to profile 1. The average molecular weight at 85% monomer conversion is at the desired value of 175,000. The resulting new initiator profile is shown in Fig. 17(a).

5. CONCLUSIONS

A new nonlinear predictive-control algorithm using an enhanced version of DMC is given. Its application to control of a semibatch polymerization reactor is demonstrated by simulation. The results show that reactor temperature can be effectively regulated and the number average molecular weight can be forced to meet a target value at the end of the batch despite strong nonlinearities due to the reaction rates and the gel effect. Constraints are not included in the present method. However, they can be addressed if the DMC control calculations are replaced by QDMC (quadratic DMC) calculations.

Acknowledgement—This material is based upon work supported under a National Science Foundation Graduate Fellowship and a Georgia Tech CIMS Fellowship.

REFERENCES

Billmeyer, F. W., 1984, *Textbook of Polymer Science*, 3rd edition. Wiley, New York.

Brengel, D. D. and Seider, W. D., 1989, A multi-step nonlinear predictive controller, in *Proceedings of ACC* 1989, pp. 1100–1105.

Dennis, J. E. and Moré, J. J., 1974, A characterization of superlinear convergence and its application to quasi-Newton methods. *Math. Comput.* **28**, 549–560.

Dennis, J. E. and Schnabel, R. B., 1983, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliff, NJ.

Eaton, J. W., Rawlings, J. B. and Edgar, T. F., 1988, Model predictive control and sensitivity analysis for constrained nonlinear processes, in *Proceedings of* 1988 *IFAC Workshop on Model Based Control* (Edited by T. J. McAvoy, Y. Arkun and E. Zafiriou).

Economou, C., 1985, An operator theory approach to nonlinear controller design. Ph.D. thesis, California Institute of Technology, Pasadena, CA.
García, C. E., 1984, Quadratic dynamic matrix control of nonlinear processes, an application to a batch reaction process. Paper 82f, AIChE Meeting, San Francisco, CA.
García, C. E. and Morshedi, A. M., 1986, Quadratic programming solution of dynamic matrix control (QDMC). *Chem. Engng Commun.* **46**, 472–484.
Houston, W. E. and Schork, F. J., 1987, Adaptive predictive control of a semibatch polymerization reactor. *Polym. Process Engng* **5**, 119–147.
Li, W. C. and Biegler, L. T., 1989, A multi-step, Newton type strategy for constrained nonlinear processes, in *Proceedings of* 1989 *ACC*, pp. 1526–1527.
Li, W. C., Biegler, L. T., Economou, C. G. and Morari, M., 1990, A constrained pseudo-Newton control strategy for nonlinear systems. *Comput. Chem. Engng* (to appear).
McAvoy, T. J., Arkun, Y. and Zafiriou, E. (eds), 1989, *IFAC Workshop Proceedings on Model Based Process Control*. Pergamon Press, Oxford.
Peterson, T. J., 1990, Nonlinear predictive control by an extended DMC. M.S. thesis, Georgia Institute of Technology, Atlanta.
Prett, D. M. and García, C. E., 1988, *Fundamental Process Control*. Butterworths, Boston, MA.
Rall, L. B., 1969, *Computational Solution of Nonlinear Operator Equations*. Wiley, New York.
Schmidt, A. D. and Ray, W. H., 1981, The dynamic behavior of continuous polymerization reactors—I. *Chem. Engng Sci.* **36**, 1401–1410.
Tanner, B. M., Adebekun, A. K. and Schork, F. J., 1987, Feedback control of molecular weight distribution during continuous polymerization. *Polym. Process Engng* **5**, 75–118.

APPENDIX A: DERIVATION OF EQUATIONS (20) AND (21)

The purpose of this Appendix is to calculate the derivative of the operator $\mathbf{F}$ defined in eq. (19) with respect to $\mathbf{d}^{nl}$. First note that $\mathbf{d}^{nl}$ is a vector whose jth entry, $d^{nl}(k+j)$ $(j = 1, \ldots, P)$, is the estimate of the disturbance caused by nonlinearities at time $k+j$ in the future. Similarly, $\mathbf{F}(\mathbf{d}^{nl})$ is a vector-valued function whose ith entry is $d^{nl}(k+i) + \beta[y^{nl}(k+i) - y^{el}(k+i)]$ $(i = 1, \ldots, P)$. The ijth element of $\mathbf{F}(\mathbf{d}^{nl})$ is given by

$$\left[\frac{\partial \mathbf{F}(\mathbf{d}^{nl})}{\partial \mathbf{d}^{nl}}\right]_{ij} = \left(\frac{\partial \mathbf{d}^{nl}}{\partial \mathbf{d}^{nl}}\right)_{ij} + \beta\left[\left(\frac{\partial \mathbf{y}^{nl}}{\partial \mathbf{d}^{nl}}\right)_{ij} - \left(\frac{\partial \mathbf{y}^{el}}{\partial \mathbf{d}^{nl}}\right)_{ij}\right]; \quad i, j = 1, \ldots P \tag{A1}$$

where

$$\left[\frac{\partial \mathbf{F}(\mathbf{d}^{nl})}{\partial \mathbf{d}^{nl}}\right]_{ij} = \frac{\partial [\mathbf{F}(\mathbf{d}^{nl})]_i}{\partial [\mathbf{d}^{nl}]_j} = \frac{\partial [\mathbf{F}(\mathbf{d}^{nl})]_i}{\partial \mathbf{d}^{nl}(k+j)}. \tag{A2}$$

We now compute each element on the right-hand side of eq. (A1). First

$$\left(\frac{\partial \mathbf{d}^{nl}}{\partial \mathbf{d}^{nl}}\right)_{ij} = \delta_{ij}, \text{where } \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases} \tag{A3}$$

Substituting eq. (9) into eq. (8):

$$\mathbf{y}^{el} = \mathbf{y}^{past} + \mathbf{A}(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}(\mathbf{y}^{sp} - \mathbf{y}^{past}) + [\mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}](\mathbf{d}^{ext} + \mathbf{d}^{nl}) \tag{A4}$$

so that

$$\beta\left(\frac{\partial \mathbf{y}^{el}}{\partial \mathbf{d}^{nl}}\right)_{ij} = \beta[(\mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma})]_{ij} \tag{A5}$$

and finally

$$\beta\left(\frac{\partial \mathbf{y}^{nl}}{\partial \mathbf{d}^{nl}}\right)_{ij} = \beta\nabla_{\mathbf{x}}\mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_i}\frac{\partial \mathbf{x}_i}{\partial [\mathbf{d}^{nl}]_j} \tag{A6}$$

where $\mathbf{x}_i$ is the state vector at time $k+i$. To calculate the partial derivative of $\mathbf{x}_i$ with respect to $d^{nl}(k+j)$, first define

$$\Psi_{ij}(t) = \frac{\partial \mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1})}{\partial [\mathbf{d}^{nl}]_j} \tag{A7}$$

by taking the time derivative of Ψ_{ij}

$$\dot{\Psi}_{ij}(t) = \frac{\mathrm{d}\left(\frac{\partial \mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1})}{\partial [\mathbf{d}^{nl}]_j}\right)}{\mathrm{d}t} = \frac{\partial\left(\frac{\mathrm{d}\mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1})}{\mathrm{d}t}\right)}{\partial [\mathbf{d}^{nl}]_j}. \tag{A8}$$

Since $\mathbf{X}(t, \mathbf{x}, \mathbf{u})$ is the state transition operator, it satisfies its own differential equation. Therefore,

$$\dot{\Psi}_{ij}(t) = \frac{\partial \mathbf{f}[\mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}), \mathbf{u}_{i-1}]}{\partial [\mathbf{d}^{nl}]_j} = \nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x} = \mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u} = \mathbf{u}_{i-1}}}\Psi_{ij} + \nabla_{\mathbf{u}}\mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x} = \mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u} = \mathbf{u}_{i-1}}}\frac{\partial \mathbf{u}_{i-1}}{\partial [\mathbf{d}^{nl}]_j}. \tag{A9}$$

Now note that

$$\frac{\partial \mathbf{x}_i}{\partial [\mathbf{d}^{nl}]_j} = \Psi_{ij}(T). \tag{A10}$$

Next we need to calculate $\partial \mathbf{u}_{i-1}/\partial [\mathbf{d}^{nl}]_j$. Recall that the change in the input is given by

$$\Delta \mathbf{u} = (\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{A}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}(\mathbf{y}^{sp} - \mathbf{y}^{past} - \mathbf{d}^{ext} - \mathbf{d}^{nl}). \tag{A11}$$

Note that $\Delta \mathbf{u}$ is a vector, and at sampling time k it is composed of

$$\Delta \mathbf{u} = \begin{bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k + M - 1) \end{bmatrix}$$

where $\Delta u(k)$ is defined as $u(k) - u(k-1)$. Equation (A9) calls for the derivative of $u(k+i-1)$ with respect to $d^{nl}(k+j)$ for $i, j = 1, \ldots, P$. Therefore, it is required to find the input for sampling times $k, \ldots, k+P-1$ into the future as a function of $\mathbf{d}^{nl}$. Let u_0 denote the input applied to the plant at time $k-1$. Then the input vector, $\mathbf{u}$, is given by

$$\mathbf{u} = \mathbf{1}_L(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{A}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}(\mathbf{y}^{sp} - \mathbf{y}^{past} - \mathbf{d}^{ext} - \mathbf{d}^{nl}) + [u_0, \ldots, u_0]^T \tag{A12}$$

where

$$\mathbf{1}_L = \underbrace{\left.\begin{bmatrix} 1 & & & 0 \\ 1 & \ddots & & \\ \vdots & & \ddots & \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}\right\} P}_{M} \tag{A13}$$

so that

$$\frac{\partial u_{i-1}}{\partial[\mathbf{d}^{nl}]} = -[1_L(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \Lambda^T\Lambda)^{-1}\mathbf{A}^T\Gamma^T\Gamma)_{ij}. \quad \text{(A14)}$$

Substituting eq. (A14) into eq. (A9) we obtain

$$\dot{\Psi}_{ij} = \nabla_x \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}} \Psi_{ij} - \nabla_u \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}} [1_L(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \Lambda^T\Lambda)^{-1}\mathbf{A}^T\Gamma^T\Gamma)_{ij} \quad \text{(A15)}$$

the initial conditions being

$$\left.\begin{aligned} &\Psi_{ij}(0) = \Psi_{i-1,j}(T) \quad i = 2, \ldots, P \\ &\Psi_{1j}(0) = 0 \end{aligned}\right\} \quad j = 1, \ldots, P. \quad \text{(A16)}$$

Combining all the terms of the derivative of F, we finally obtain

$$\{\mathbf{F}'(\mathbf{d}^{nl})\}_{ij} = \{\mathbf{I}(1-\beta) - \beta[\mathbf{A}(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \Lambda^T\Lambda)^{-1}\mathbf{A}^T\Gamma^T\Gamma]\}_{ij} + \beta\nabla_x \mathbf{g}(\mathbf{x}_i)\psi_{ij}(T) \quad \text{(A17)}$$

which is the matrix stated in the theorem.

APPENDIX B: PROOF OF THEOREM 3

First note that the entries of the "dynamic matrix", A, are the step-response coefficients. As the sampling time tends to infinity, these coefficients tend to be all equal to the steady-state gain; thus matrix A can be factorized as:

$$\mathbf{A}|_{T\to\infty} = a1_L \quad \text{(B1)}$$

where 1_L is defined by eq. (22). For globally asymptotically stable systems, as the sampling time tends to infinity, $\mathbf{X}(t + t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1})$ tends to the equilibrium point and thus

$$\nabla_x \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}$$

has negative eigenvalues. This implies that Ψ in eq. (21) reaches a steady-state solution and that

$$\nabla_x \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t+t_{i-1}, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}$$

is invertible. Thus from eq. (21)

$$0 = \nabla_x \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}} \Psi_{i,j} - \nabla_u \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_i, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}} [1_L(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\Gamma^T\Gamma]_{ij} \quad \text{(B2)}$$

$$\Psi_{ij}|_{T\to\infty} = \Bigg[\underbrace{\nabla_x \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_i, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}}_{\nabla_x \mathbf{f}_i}\Bigg]^{-1} \underbrace{\nabla_u \mathbf{f}(\mathbf{x}, \mathbf{u})\Big|_{\substack{\mathbf{x}=\mathbf{X}(t_i, \mathbf{x}_{i-1}, \mathbf{u}_{i-1}) \\ \mathbf{u}=\mathbf{u}_{i-1}}}}_{\nabla_u \mathbf{f}_i} [1_L(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\Gamma^T\Gamma]_{ij}. \quad \text{(B3)}$$

Substituting in eq. (20)

$$[\mathbf{F}'(\mathbf{d}^{nl})]_{ij} = \{\mathbf{I}(1-\beta) - \beta[\mathbf{A}(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \Lambda^T\mathbf{A})^{-1}\mathbf{A}^T\Gamma^T\Gamma)\}_{ij} + \beta\nabla_x \mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_i}[\nabla_x \mathbf{f}_i]^{-1}\nabla_u \mathbf{f}_i[1_L(\mathbf{A}^T\Gamma^T\Gamma\mathbf{A} + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\Gamma^T\Gamma]_{ij}. \quad \text{(B4)}$$

But note that the term $\nabla_x \mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_i}[\nabla_x \mathbf{f}_i]^{-1}\nabla_u \mathbf{f}_i$ was given in lemma 4 as the negative of the steady-state gain at the point $(\mathbf{x}_i, \mathbf{u}_{i-1})$ which is not necessarily the linearization point used to calculate A. This term is denoted as $-a_i'$. Again using the factorization of A given by eq. (B1),

$$\mathbf{F}'(\mathbf{d}^{nl}) = \mathbf{I}(1-\beta) - \beta(\mathbf{I} - \mathbf{D}_\mathbf{A})\left[1_L\left(1_L^T1_L + \frac{\lambda^2}{a^2}\mathbf{I}\right)^{-1}1_L^T\right] \quad \text{(B5)}$$

where

$$\mathbf{D}_\mathbf{A} = \operatorname{diag}\left(\frac{a_i'}{a}\right); \quad i = 1, \ldots, P.$$

Next the maximum singular value, $\sigma^*(\cdot)$, will be chosen as the norm. An upper bound of $\sigma^*[\mathbf{F}'(\mathbf{d}^{nl})]$ can be computed by using the following inequalities

$$\sigma^*(\mathbf{AB}) \leqslant \sigma^*(\mathbf{A})\sigma^*(\mathbf{B})$$
$$\sigma^*(\mathbf{A} + \mathbf{B}) \leqslant \sigma^*(\mathbf{A}) + \sigma^*(\mathbf{B}). \quad \text{(B6)}$$

Therefore,

$$\sigma^*[\mathbf{F}'(\mathbf{d}^{nl})] \leqslant |1-\beta|\sigma^*(\mathbf{I}) + |\beta|\sigma^*(\mathbf{I} - \mathbf{D}_\mathbf{A})\sigma^* \times\left[1_L\left(1_L^T1_L + \frac{\lambda^2}{a^2}\mathbf{I}\right)^{-1}1_L^T\right]. \quad \text{(B7)}$$

Since $\sigma^*(\mathbf{A}^{-1}) = \sigma_*(\mathbf{A})$, where $\sigma_*(\cdot)$ is the minimum singular value, and for $0 < \beta < 1$

$$\sigma^*[\mathbf{F}'(\mathbf{d}^{nl})] \leqslant 1 - \beta + \beta\sigma^*(\mathbf{I} - \mathbf{D}_\mathbf{A})\frac{\sigma^*(1_L)^2}{\sigma_*\left(1_L^T1_L + \frac{\lambda^2}{a^2}\mathbf{I}\right)}. \quad \text{(B8)}$$

Note that the matrix $[1_L^T1_L + (\lambda/a^2)\mathbf{I}]$ is symmetric. Therefore, its singular values are equal to the absolute values of the eigenvalues. Furthermore, note that

$$\text{eigenvalues}(\mathbf{A} + \xi\mathbf{I}) = \text{eigenvalues}(\mathbf{A}) + \xi \quad \text{(B9)}$$

and since in addition $1_L^T1_L$ is positive semidefinite, the eigenvalues are equal to the singular values

$$\sigma_*\left(1_L^T1_L + \frac{\lambda^2}{a^2}\mathbf{I}\right) = \sigma_*(1_L^T1_L) + \frac{\lambda^2}{a^2}. \quad \text{(B10)}$$

Finally

$$\sigma^*(\mathbf{F}'(\mathbf{d}^{nl})) \leqslant 1 - \beta + \beta\sigma^*(\mathbf{I} - \mathbf{D}_\mathbf{A})\frac{[\sigma^*(1_L)]^2}{\sigma_*(1_L^T1_L) + \frac{\lambda^2}{a^2}}. \quad \text{(B11)}$$

As claimed, β could be made small enough and λ large enough so that the second term in eq. (B11) is negligible and thus $\sigma^*[\mathbf{F}'(\mathbf{d}^{nl})]$ could be made smaller than one for all $\mathbf{d}^{nl}$ in $\mathbf{R}^p$. Therefore F is a contraction in all of $\mathbf{R}^p$ and thus guarantees the convergence of the iterations. □

Remark: Note that by increasing λ the overall performance is sacrificed. By decreasing β the convergence is slower since the maximum singular value of $\mathbf{F}'(\mathbf{d}^{nl})$ reaches $1 - \beta$, which is almost one. Therefore, it is important to find a tradeoff between these two factors and satisfy eq. (B11).

APPENDIX C: PROOF OF THEOREM 4

We begin this development by identifying the operator N for the particular case at hand. The operator mapping $(\mathbf{x}_k, \mathbf{u}_k)$ to $\mathbf{x}_{k+1}$ is simply the state transition operation defined in eq. (12). Furthermore, for a nominal system ($\mathbf{d}^{ext} = 0$), using eq. (8) gives the input for time $k+1$:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{e}_1^T[(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}(\mathbf{y}_{k+1}^{sp} - \mathbf{y}_{k+1}^{past} - \mathbf{d}_{k+1}^{nl})] \tag{C1}$$

where $\mathbf{e}_1^T = [1, 0, \ldots, 0]$. Recall that now k denotes the sampling time. Thus $\mathbf{d}_k^{nl}$ is the vector of disturbance due to nonlinearities for sampling time k *once the iterations have converged*. This is not to be confused with the $\mathbf{d}_i^{nl}$ used before which denoted the ith iterate of the calculation of $\mathbf{d}^{nl}$ at sampling time k. Equations (12) and (C1) partially define the operator N since the right-hand side of eq. (C1) involves quantities which depend explicitly on variables other than $\mathbf{x}_k$ and $\mathbf{u}_k$.

Note that, because the sampling time is large and $M = 1$, the dynamic matrix, A, is given by $a \cdot \mathbf{1} = [a, \ldots, a]'$, where $\mathbf{1} \in \mathbf{R}^P$. Furthermore, note that $\mathbf{y}_{k+1}^{past}$ is the output at sampling times $k+2, \ldots, k+P+1$ assuming that the input is held constant at $\mathbf{u}_k$. Since T is large, $\mathbf{y}_{k+1}^{past}$ is $y^{nl}(k+1)\mathbf{1}$.

Note that the assumptions stated in theorem 4 satisfy those required in theorem 3. Therefore, the iterations (10) are guaranteed to converge. Once the iterations converge, the value of $\mathbf{d}^{nl}$ is that which makes $\mathbf{y}_{k+1}^{nl} = \mathbf{y}_{k+1}^{sl}$ or

$$\mathbf{y}_{k+1}^{sl} = \mathbf{y}_{k+1}^{past} + \mathbf{A}(\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}\mathbf{A} + \mathbf{\Lambda}^T\mathbf{\Lambda})^{-1}\mathbf{A}^T\mathbf{\Gamma}^T\mathbf{\Gamma}(\mathbf{y}_{k+1}^{sp} - \mathbf{y}_{k+1}^{past} - \mathbf{d}_{k+1}^{nl}) + \mathbf{d}_{k+1}^{nl}. \tag{C2}$$

With T large and $M = 1$, eq. (C2) becomes

$$y^{nl}(k+2)\mathbf{1} = y^{nl}(k+1)\mathbf{1} + \frac{a^2\mathbf{1}\mathbf{1}^T}{a^2P + \lambda^2}(y^{sp}\mathbf{1} - y^{nl}(k+1)\mathbf{1} - \mathbf{d}_{k+1}^{nl}) + \mathbf{d}_{k+1}^{nl} \tag{C3}$$

$$\left(\mathbf{I} - \frac{a^2\mathbf{1}\mathbf{1}^T}{a^2P + \lambda^2}\right)\mathbf{d}_{k+1}^{nl} = y^{nl}(k+2)\mathbf{1} - y^{nl}(k+1)\mathbf{1} + \frac{a^2\mathbf{1}\mathbf{1}^T}{a^2P + \lambda^2}[y^{nl}(k+1)\mathbf{1} - y^{sp}\mathbf{1}]. \tag{C4}$$

It can be shown through the Sherman–Morrison–Woodberry formula (refer to Dennis and Schnable, 1983, p. 188) that

$$\left(\mathbf{I} - \frac{a^2\mathbf{1}\mathbf{1}^T}{a^2P + \lambda^2}\right)^{-1} = \left(\mathbf{I} + \frac{a^2}{\lambda^2}\mathbf{1}\mathbf{1}^T\right). \tag{C5}$$

Substituting eq. (C5) into eq. (C4) and noting that $\mathbf{1}^T\mathbf{1} = P$

$$\mathbf{d}_{k+1}^{nl} = \frac{\lambda^2 + a^2P}{\lambda^2}y^{nl}(k+2)\mathbf{1} - \frac{a^2P}{\lambda^2}y^{sp}\mathbf{1} - y^{nl}(k+1)\mathbf{1}. \tag{C6}$$

Finally substituting eq. (C.6) into eq. (C1)

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \frac{aP}{\lambda^2}[y^{sp} - y^{nl}(k+2)]. \tag{C7}$$

where

$$\begin{aligned} y^{nl}(k+2) &= g(\mathbf{x}_{k+2}) \\ \mathbf{x}_{k+2} &= \mathbf{X}(t_{k+2}, \mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \\ \mathbf{x}_{k+1} &= \mathbf{X}(t_{k+1}, \mathbf{x}_k, \mathbf{u}_k). \end{aligned} \tag{C8}$$

Thus eqs (13), and (C7) along with eq. (C8) implicitly define the operator N. In what follows, the Jacobian of N is calculated term by term.

The first terms to be calculated are the derivatives of the state evolution operator with respect to the state and the input. From arguments mentioned before, it is clear that as sampling time tends to infinity both $\mathbf{\Phi}$ and $\mathbf{\Gamma}$ go to steady state and are equal to

$$\begin{aligned} \mathbf{\Gamma}|_{T\to\infty} &= -[\nabla_{\mathbf{x}}\mathbf{f}_t]^{-1}\nabla_{\mathbf{u}}\mathbf{f}_t \\ \mathbf{\Phi}|_{T\to\infty} &= 0. \end{aligned} \tag{C9}$$

Since the vector fields were assumed to have bounded derivatives, $\mathbf{\Gamma}$ is bounded.

The partial derivative of the input with the state is calculated from eq. (C7) as

$$\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{x}_k} = -\underbrace{\frac{aP}{\lambda^2}}_{=\mathbf{C}_{k+2}}\underbrace{\frac{\partial \mathbf{g}(\mathbf{x}_{k+2})}{\partial \mathbf{x}_{k+2}}}_{=\mathbf{\Phi}_{k+1}}\left(\underbrace{\frac{\partial \mathbf{x}_{k+2}}{\partial \mathbf{x}_{k+1}}}_{=\mathbf{\Phi}_k}\underbrace{\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k}}_{=\mathbf{\Gamma}_{k+1}} + \frac{\partial \mathbf{x}_{k+2}}{\partial \mathbf{u}_{k+1}}\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{x}_k}\right) \tag{C10}$$

$$\left(1 + \frac{P}{\lambda^2}a\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1}\right)\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{x}_k} = -\frac{aP}{\lambda^2}\mathbf{C}_{k+2}\mathbf{\Phi}_{k+1}\mathbf{\Phi}_k \tag{C11}$$

and, as sampling time tends to infinity, the $\mathbf{\Phi}$ terms on the right-hand side tend to zero. Furthermore, the coefficient of $\partial \mathbf{u}_{k+1}/\partial \mathbf{x}_k$ is nonzero for cases where the steady-state gain does not change sign (since $a\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1}$ is greater than zero). Thus, $\partial \mathbf{u}_{k+1}/\partial \mathbf{x}_k$ is well-defined and tends to zero as the sampling time tends to infinity.

Finally $\partial \mathbf{u}_{k+1}/\partial \mathbf{u}_k$ is given by

$$\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{u}_k} = 1 - \underbrace{\frac{aP}{\lambda^2}}_{=\mathbf{C}_{k+2}}\underbrace{\frac{\partial \mathbf{g}(\mathbf{x}_{k+2})}{\partial \mathbf{x}_{k+2}}}_{=\mathbf{\Phi}_{k+2}}\left(\underbrace{\frac{\partial \mathbf{x}_{k+2}}{\partial \mathbf{x}_{k+1}}}_{=\mathbf{\Gamma}_k}\underbrace{\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{u}_k}}_{=\mathbf{\Gamma}_{k+1}} + \frac{\partial \mathbf{x}_{k+2}}{\partial \mathbf{u}_{k+1}}\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{u}_k}\right) \tag{C12}$$

$$\left(1 + \frac{P}{\lambda^2}a\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1}\right)\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{u}_k} = 1 - \frac{aP}{\lambda^2}\mathbf{C}_{k+2}\mathbf{\Phi}_{k+1}\mathbf{\Gamma}_k. \tag{C13}$$

Again, it is seen that for large sampling times the right-hand side of eq. (C13) tends to one. Since the coefficient of $\partial \mathbf{u}_{k+1}/\partial \mathbf{u}_k$ in eq. (C13) is nonzero,

$$\frac{\partial \mathbf{u}_{k+1}}{\partial \mathbf{u}_k} = \frac{\lambda^2}{(\lambda^2 + Pa\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1})} < 1 \tag{C14}$$

and the Jacobian of N becomes

$$\mathbf{N}'\begin{pmatrix}\mathbf{x}_k \\ \mathbf{u}_k\end{pmatrix}\Bigg|_{T\to\infty} = \begin{bmatrix} 0 & \mathbf{\Gamma}_k \\ 0 & \dfrac{\lambda^2}{(\lambda^2 + Pa\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1})} \end{bmatrix}. \tag{C15}$$

The singular values of the above matrix are zero and

$$\sigma^* = \sqrt{\mathbf{\Gamma}'\mathbf{\Gamma} + \left(\frac{\lambda^2}{\lambda^2 + P\mathbf{C}_{k+2}\mathbf{\Gamma}_{k+1}}\right)^2} \tag{C16}$$

which can be made less than one by appropriately scaling the input since if a new input is defined as $1/\alpha$ times the old input then

$$\mathbf{\Gamma}^{new} = \alpha\mathbf{\Gamma}^{old} \tag{C17}$$

and then α can be chosen so that eq. (C16) is less than one. This is true for all $(\mathbf{x}_k, \mathbf{u}_k)$ in $\mathbf{R}^{n+1}$ so that N is a contraction and by the contraction mapping theorem the closed-loop system is nominally stable.