# Multivariable Model State Feedback: Computationally Simple, Easy-to-Tune Alternative to MPC

**Suraj Mhatre and Coleman Brosilow**
Chemical Engineering Dept., A. W. Smith Building, Case Western Reserve University, Cleveland, OH 44106

*Methods for the design and tuning of multivariable model state feedback (MMSF) systems for complete and triangular output decoupling of inherently stable processes are presented. MMSF is a method of implementing multivariable IMC controllers that substantially reduces the complexity of the implementation and compensates for control effort saturation. Complexity is reduced because the controls are formed as a linear combination of past and current model states. Saturation-induced directionality problems are avoided by temporarily increasing filter time constants to bring the control vector into the constraint set. The possibility that needed future controls will be inadequate to compensate for past control actions is avoided by setting minimum values for the controller filter time constants. The feedback structure of MMSF facilitates on-line tuning, one process variable at a time. Also, because MMSF is a form of IMC, it can be tuned off-line to accommodate process uncertainty using $H_\infty$ methods.*

## Introduction

Multivariable model state feedback (MMSF) provides a practical way of implementing multivariable internal model control (IMC) systems. This work is the extension to the multivariable case of the single-variable model state feedback algorithm of Coulibaly et al. (1995). Zheng et al. (1993) suggest a modified IMC structure as an antiwindup design. Their structure includes the MMSF structure as a special case. However, unlike the current work, they give no guidance on how to choose the elements within their general structure.

As with standard IMC control systems, the control structure used for MMSF in this article is limited to the control of inherently stable multivariable systems. Extension to unstable processes requires modification of the IMC structure so that it becomes internally stable even for unstable processes. One way to accomplish such a modification is to force the process model outputs to track the measured process outputs in a manner similar to that proposed by Berber and Brosilow (1999) for single-input, single-output systems. Such a modification still allows use of the model state-feedback methods developed in this article. Another limitation of the current article is that it addresses only single-degree-of-freedom control systems. That is, disturbances are assumed to enter directly into the process output rather than through part of the process. This limitation can be overcome using the approach of Dong and Brosilow (1997) for two-degree-of-freedom control systems.

When implemented within a multivariable cascade (Boyce, 1996) MMSF has the potential to compete favorably with current model predictive control (MPC) technologies such as DMC (Cutler and Ramaker, 1979). The potential advantages of MMSF over MPC methods are (1) the relative ease of design for totally decoupled control; (2) the simplicity of its implementation, (3) its flexibility with regard to specifying control system behavior when controls saturate, or are lost by, equipment malfunction or by being placed into manual operation; and (4) its ability to be tuned either ''on-line,'' one process variable at a time, or ''off-line'' using model uncertainty descriptions and $H_\infty$ methods (Stryczek and Brosilow, 1996; Zhou and Doyle, 1998). The latter property comes directly from the fact that MMSF is a form of IMC.

In MMSF the IMC control efforts are formed as a linear combination of past and present model states. For this reason, the on-line computational expense is potentially orders of magnitude less than that of MPC algorithms that use on-line optimization to compute the control efforts. This property of MMSF should allow a single MMSF algorithm to control all the units of integrated processes such as an ethylene process (Emoto et al., 1995), even when the units have widely varying time constants, thereby achieving substantial gains in

Correspondence concerning this article should be addressed to C. Brosilow.
Present address for S. Mhatre: Aspen Technology, Inc., Houston, TX.

operating efficiency. Flexibility of operation can be maintained while controlling many units because of MMSF's ability to allow both controls and outputs to be conveniently switched between automatic and manual operation.

Off-line computations for MMSF are also quite modest. Unlike multivariable lead–lag implementations of IMC, MMSF does not require the explicit inversion of the multivariable process model, even though MMSF gives exactly the same control efforts as the lead–lag implementation of the IMC controller in the absence of control effort saturation. For processes with more than three controls and process variables, inversion of the process model is a very difficult task, and the resulting controller, if obtainable, can be very complex.

An important feature of MMSF is that its performance is excellent even when the control effort saturates. On the other hand, the performance of lead–lag IMC control systems can be, and often is, extremely poor when controls saturate (Campo and Morari, 1990; Coulibaly et al., 1995; Zheng and Kothare, 1993). There are two reasons for the poor performance of saturating lead–lag IMC controllers: (1) simply chopping the control efforts at their saturation values does not generally yield a valid IMC control vector, and this can lead to very large overshoots or produce needlessly sluggish responses; and (2) in lead–lag IMC controllers, the control effort is based only on the setpoint and the disturbance estimate, and thereby ignores the fact that the applied control effort may not have been the computed control effort. A valid IMC control vector is one that can be obtained from an IMC controller and that lies within the constraint set. As we shall demonstrate, it is a relatively simple task to temporarily adjust the MMSF filter time constants on-line to bring the controls within the constraint set. Further, the adjustment can be carried out so as to give priority to the responses of the most important process variables. The MMSF structure automatically compensates for past control effort saturation, because the model states from which the control effort is computed always reflect the applied control.

While the MMSF structure automatically compensates for past control effort saturation, it does not automatically compensate for future control effort saturation. For control systems with small filter time constants relative to the process time constants, failure to appropriately compensate for the possibility of future control effort saturation can result in limit cycles or highly oscillatory responses. Two ways of compensating for future control effort saturation are (1) use an algorithm that projects controls into the future, assuming that disturbances remain constant such as in MPC (Muske and Rawlings, 1993) or internal model predictive control (Coulibaly et al., 1995); and (2) select the filter time constants large enough so that future control effort saturation does not cause problems. In this article, we use the latter approach and suggest an algorithm for selecting lower bounds on all the filter time constants to avoid saturation-induced oscillations. Experience with several examples indicates that the necessary lower bounds on the filter time constant are not significantly higher than the lower bounds that arise from the imposition of reasonable limits on the controller noise amplification.

In addition to the methods presented herein, the model state-feedback gains used to form the control efforts in MMSF can also be obtained by setting up and solving an infinite horizon, unconstrained, linear quadratic regulator problem. However, we do not recommend this approach, because it loses some of the insights gained by the proposed approach, and does not lend itself well to bringing computed control efforts within the constraint set, as required to achieve good process variable responses.

The last section of this article (the fifth section) discusses how one moves controls between manual to automatic without upsetting the behavior of the entire control system. Included in this discussion is a method for on-line tuning of a MMSF system, one process variable at a time.

Examples are provided throughout the article to illustrate the suggested methodologies. However, we provide no direct comparisons with any of the various MPC algorithms, because valid comparisons are quite difficult to construct. Most MPC algorithms rely on optimization of a scalar objective to get good process variable responses in spite of control effort constraints. For any fixed set of weights in the objective function, actual performance of the MPC algorithm will depend on where the nominal operating point is within the constraint set. Thus, a valid comparison has to compare performance over the entire operating range, and not just for a single-step setpoint change, at say, the midpoint of the operating range. The criterion that one uses for the comparison is also problematic. If the objective function minimization in MPC is carried out exactly (usually it's not), the MMSF can at best achieve the same scalar objective. However, it has been recognized for almost 50 years that minimizing a scalar objective can lead to very oscillatory, and unacceptable, process variable responses unless sufficient weight is placed on penalizing large control efforts. Thus, the actual objective function is usually somewhat arbitrary, and is chosen to achieve ''good,'' rather than the minimum of a realistic cost functional. In multivariable control systems, the situation is further complicated by how much weight to put on whether responses to setpoint changes are decoupled or not. Rather than get into this morass we hope to convince the reader that MMSF competes favorably with MPC, because it achieves smooth, rapid, decoupled, responses to setpoint changes, without overshoots (for perfect models) that appear to be about as fast as possible subject to the control effort constraints. This is accomplished with negligible on-line computational effort beyond that required to simulate the model. Finally, the MMSF algorithm lends itself well to either ad-hoc or $H_\infty$ tuning methods (Stryczek and Brosilow, 1996) for imperfect (that is, uncertain) process models.

## Multivariable Model State Feedback

Multiinput, multioutput (MIMO) processes are usually represented as matrices of transfer functions. Kailath (1980) shows how to express such matrices as a product of two matrices, $N(s)$ and $D^{-1}(s)$. Matrix $D(s)$ has only polynominal elements, while matrix $N(s)$ has elements that are polynomials in $s$ and $e^{-s}$. Since such matrix fraction descriptions play a central role in extending MSF to multivariable systems, they are reviewed below.

### *Matrix fraction descriptions*

The right matrix fraction representation of the multivariable process $G(s)$ is given as

$$G(s) = N_R(s, e^{-s}) D_R^{-1}(s). \quad (1)$$

Both $D_R(s)$ and $N_R(s, e^{-s})$ are matrices with each entry a polynomial in $s$ and $e^{-s}$. The degree of the denominator matrix, which is also the order of the state-space realization, is defined as

$$Degree\left[D_R(s)\right] = Degree\left\{Determinant\left[D_R(s)\right]\right\}. \quad (2)$$

In the same way, one can define the left matrix fraction representation of the process $G(s)$ as

$$G(s) = D_L^{-1}(s) N_L(s, e^{-s}). \quad (3)$$

For the rest of the article, we will consider only right matrix fraction representations of the processes and drop the subscript on $D(s)$ and $N(s, e^{-s})$.

There is no unique matrix fraction for the given process. Given any MFD, a number of other MFDs can be obtained by choosing any nonsingular polynomial matrix $W(s)$, and the new numerator and denominator matrices are given as

$$\overline{N}(s, e^{-s}) = N(s, e^{-s}) W(s) \qquad \overline{D}(s) = D(s) W(s), \quad (4)$$

so that

$$G(s) = \overline{N}(s, e^{-s}) \overline{D}^{-1}(s). \quad (5)$$

For the MSF implementation we choose a $\overline{D}(s)$ that will give us a minimal realization in the time domain.

### *Model state-feedback law*

Model state feedback is one of the ways to implement internal model control. Zheng et al. (1993) suggest a modified IMC structure as an antiwindup design. Their structure includes the MSF structure of Figure 1 as a special case. However, they give no guidance on how to choose the various matrix elements of Figure 1.

The model state-feedback control systems of Figure 1 uses two elements, $K_{sp}$ and $K(s)D(s)^{-1}$, to form the IMC controller. Elements $K(s)$ and $K_{sp}$ are computed so that in the absence of saturation, this structure has exactly the same performance as a lead–lag IMC controller. Therefore,

$$Q_{IMC}(s) = G_-^{-1}(s) F(s) = \left(I + K(s) D^{-1}(s)\right)^{-1} K_{sp}, \quad (6)$$

where $G_-(s)$ is an invertible part of the process and $F(s)$ is the IMC filter. Solving Eq. 6 for $K(s)$, we get the feedback

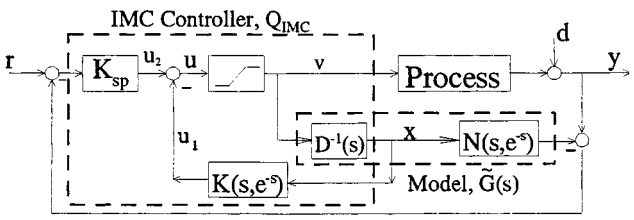


**Figure 1. Model state feedback implementation of the IMC control.**

matrix to be

$$K(s) = K_{sp} F^{-1}(s) G_-(s) D(s) - D(s)$$
$$= K_{sp} F^{-1}(s) G_+^{-1}(s) N(s) - D(s), \quad (7a)$$

where $G_+(s)$ is the portion of the model that is not to be inverted by the IMC controller; that is, $G(s) = G_+(s) G_-(s)$.

We choose $K_{sp}$ as a matrix of constants such that the order of polynomials in the matrix $K(s)$ is at least one less than the order of the polynomials in $D(s)$. It is relatively easy to show that $K_{sp}$ is the same as $Q_{IMC}(\infty)$, that is,

$$K_{sp} = \left\{ \lim_{s \to \infty} \left[ F^{-1} G_-(s) \right] \right\}^{-1}. \quad (7b)$$

Notice that there is no need to form the IMC controller, or to invert $G_-(s)$ in order to form $K(s)$. As we shall see, we usually choose $G_+(s)$ to be either diagonal or triangular, and so it is relatively easy to invert.

### *Minimum phase processes*

Minimum phase processes are invertible processes, and hence we can design the controller by inverting the entire process and adding the filter to make the controller realizable and avoid excessive noise amplification. Consider the $2 \times 2$ process given by

$$G(s) = \begin{pmatrix} \dfrac{k_{11}}{\tau_{11}s+1} & \dfrac{k_{12}}{\tau_{12}s+1} \\ \dfrac{k_{21}}{\tau_{21}s+1} & \dfrac{k_{22}}{\tau_{22}s+1} \end{pmatrix} \quad (8)$$

We propose the following criteria for choosing the particular matrix fraction:

1. The realization must be stable.
2. The implementation should be as simple as possible.
3. The number of states should be kept to a minimum.
4. The implementation must generalize to any order system.

Based on these criteria we choose the following polynomial matrix for $D(s)$ when $\tau_{i,1} \neq \tau_{i,2}$, $i = 1, 2$

$$\left[\text{when } \tau_{i,1} = \tau_{i,2}, i = 1, 2, \text{then } D(s) \text{ is chosen as } \begin{pmatrix} (\tau_{11}s+1) & 0 \\ 0 & (\tau_{22}s+1) \end{pmatrix}\right]$$

$$D(s) = \begin{pmatrix} (\tau_{11}s+1)(\tau_{21}s+1) & 0 \\ 0 & (\tau_{12}s+1)(\tau_{22}s+1) \end{pmatrix}. \quad (9)$$

This choice of $D(s)$ is certainly stable and easy to implement as two decoupled second-order differential equations. From $N(s) = G(s)D(s)$ we have

$$N(s) = \begin{pmatrix} k_{11}(\tau_{21}s+1) & k_{12}(\tau_{22}s+1) \\ k_{21}(\tau_{11}s+1) & k_{22}(\tau_{12}s+1) \end{pmatrix}. \quad (10)$$

Next we define the states as

$$\begin{pmatrix} x_1(s) \\ x_2(s) \end{pmatrix} = \begin{pmatrix} \dfrac{1}{(\tau_{11}s+1)(\tau_{21}s+1)} & 0 \\ 0 & \dfrac{1}{(\tau_{12}s+1)(\tau_{22}s+1)} \end{pmatrix} \begin{pmatrix} u_1(s) \\ u_2(s) \end{pmatrix}. \tag{11}$$

For the preceding definition of $D(s)$ the state realization is given as

$$\begin{pmatrix} \ddot{x}_1 \\ \dot{x}_1 \\ \ddot{x}_2 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} \dfrac{-(\tau_{11}+\tau_{21})}{\tau_{11}\tau_{21}} & \dfrac{-1}{\tau_{11}\tau_{21}} & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & \dfrac{-(\tau_{12}+\tau_{22})}{\tau_{12}\tau_{22}} & \dfrac{-1}{\tau_{12}\tau_{22}} \\ 0 & 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} \dot{x}_1 \\ x_1 \\ \dot{x}_2 \\ x_2 \end{pmatrix} + \begin{pmatrix} \dfrac{1}{\tau_{11}\tau_{21}} & 0 \\ 0 & 0 \\ 0 & \dfrac{1}{\tau_{12}\tau_{22}} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}. \tag{12}$$

The next step is to find feedback matrix $K(s)$ and $K_{sp}$ from Eq. 7:

$$K(s) = \begin{pmatrix} K_{sp_{11}} & K_{sp_{12}} \\ K_{sp_{21}} & K_{sp_{22}} \end{pmatrix} \begin{pmatrix} (\epsilon_1 s+1) & 0 \\ 0 & (\epsilon_2 s+1) \end{pmatrix} \times \begin{pmatrix} k_{11}(\tau_{21}s+1) & k_{12}(\tau_{22}s+1) \\ k_{21}(\tau_{11}s+1) & k_{22}(\tau_{12}s+1) \end{pmatrix} - \begin{pmatrix} (\tau_{11}s+1)(\tau_{21}s+1) & 0 \\ 0 & (\tau_{12}s+1)(\tau_{22}s+1) \end{pmatrix}. \tag{13}$$

Choose matrix $K_{sp}$ such that the coefficient of the $s^2$ term in each polynomial becomes zero. The matrix $K_{sp}$ that satisfies this condition is

$$K_{sp} = H^{-1}E^{-1},$$

where

$$H \equiv \begin{pmatrix} \dfrac{k_{11}}{\tau_{11}} & \dfrac{k_{12}}{\tau_{12}} \\ \dfrac{k_{21}}{\tau_{21}} & \dfrac{k_{22}}{\tau_{22}} \end{pmatrix} \quad \text{and} \quad E \equiv \begin{pmatrix} \epsilon_1 & 0 \\ 0 & \epsilon_2 \end{pmatrix}. \tag{14}$$

Substituting $K_{sp}$ from Eq. 14 into Eq. 13 gives a $K(s)$ where each element is a first-order polynomial. The controls, $u(s)$, are then a linear combination of states, as given by

$$u(s) = -K(s)x(s) + K_{sp}(r - \tilde{d}). \tag{15}$$

Finally, the model output is also a linear combination of states, as given by $N(s)x(s)$.

For a minimum phase $n \times n$ matrix of transfer functions whose denominators are polynomials we suggest forming $D(s)$ as a diagonal matrix of polynomials such that each diagonal element, $d_{ii}$, is the product of the denominators in column $i$. Thus the degree of $d_{ii}$ is the sum of the degrees of all the denominator terms in the $i$th column of $G(s)$. Again, the matrix $N(s)$ is formed by postmultiplying the process matrix by $D(s)$.

*Example 1.* This is a $2 \times 2$ minimum phase process:

$$G(s) = \begin{pmatrix} \dfrac{4}{10s+1} & \dfrac{-2}{20s+1} \\ \dfrac{3}{8s+1} & \dfrac{-1}{12s+1} \end{pmatrix}. \tag{16}$$

We choose $D(s)$ to be

$$D(s) = \begin{pmatrix} (10s+1)(8s+1) & 0 \\ 0 & (20s+1)(12s+1) \end{pmatrix} \tag{17}$$

$$N(s) = \begin{pmatrix} 4(8s+1) & -2(12s+1) \\ 3(10s+1) & -1(20s+1) \end{pmatrix}. \tag{18}$$

The $K_{sp}$ matrix and the feedback matrix $K(s)$ are given by

$$K_{sp} = \begin{pmatrix} -20\epsilon_1^{-1} & 24\epsilon_2^{-1} \\ -90\epsilon_1^{-1} & 96\epsilon_2^{-1} \end{pmatrix} \tag{19}$$

$$K(s) = \begin{pmatrix} (640\epsilon_1^{-1}+720\epsilon_2^{-1}-26)s + (-80\epsilon_1^{-1}+72\epsilon_2^{-1}-1) & (480\epsilon_1^{-1}-480\epsilon_2^{-1}+16)s + (40\epsilon_1^{-1}-24\epsilon_2^{-1}) \\ (-2{,}880\epsilon_1^{-1}+2{,}880\epsilon_2^{-1}-72)s + (-360\epsilon_1^{-1}+288\epsilon_2^{-1}) & (2{,}160\epsilon_1^{-1}-1{,}920\epsilon_2^{-1}+52)s + (180\epsilon_1^{-1}-96\epsilon_2^{-1}-1) \end{pmatrix}. \tag{20}$$

### *Processes with dead times*

In this section we consider processes with delays, but without any right half-plane transmission zeros. Consider the following $2 \times 2$ process with delays.

$$G(s) = \begin{pmatrix} \dfrac{k_{11}}{\tau_{11}s+1}e^{-\theta_{11}s} & \dfrac{k_{12}}{\tau_{12}s+1}e^{-\theta_{12}s} \\ \dfrac{k_{21}}{\tau_{21}s+1}e^{-\theta_{21}s} & \dfrac{k_{22}}{\tau_{22}s+1}e^{-\theta_{22}s} \end{pmatrix}. \tag{21}$$

We choose $D(s)$ exactly the same way as we did for a minimum phase process:

$$D(s)=\begin{pmatrix}(\tau_{11}s+1)(\tau_{21}s+1) & 0\\ 0 & (\tau_{12}s+1)(\tau_{22}s+1)\end{pmatrix}. \quad (22)$$

Postmultiplying the process matrix, $G(s)$, by $D(s)$ gives

$$N(s)=\begin{pmatrix}k_{11}(\tau_{21}s+1)e^{-\theta_{11}s} & k_{12}(\tau_{22}s+1)e^{-\theta_{12}s}\\ k_{21}(\tau_{11}s+1)e^{-\theta_{21}s} & k_{22}(\tau_{12}s+1)e^{-0_{22}s}\end{pmatrix}. \quad (23)$$

We define the states as

$$\begin{pmatrix}x_1(s)\\ x_2(s)\end{pmatrix}=\begin{pmatrix}\dfrac{1}{(\tau_{11}s+1)(\tau_{21}s+1)} & 0\\ 0 & \dfrac{1}{(\tau_{12}s+1)(\tau_{22}s+1)}\end{pmatrix}\times\begin{pmatrix}u_1(s)\\ u_2(s)\end{pmatrix}. \quad (24)$$

For this the controller form realization, which is the same as for the minimum phase process, is given as

$$\begin{pmatrix}\ddot{x}_1\\ \dot{x}_1\\ \ddot{x}_2\\ \dot{x}_2\end{pmatrix}=\begin{pmatrix}\dfrac{-(\tau_{11}+\tau_{21})}{\tau_{11}\tau_{21}} & \dfrac{-1}{\tau_{11}\tau_{21}} & 0 & 0\\ 1 & 0 & 0 & 0\\ 0 & 0 & \dfrac{-(\tau_{12}+\tau_{22})}{\tau_{12}\tau_{22}} & \dfrac{-1}{\tau_{12}\tau_{22}}\\ 0 & 0 & 1 & 0\end{pmatrix}\times\begin{pmatrix}\dot{x}_1\\ x_1\\ \dot{x}_2\\ x_2\end{pmatrix}+\begin{pmatrix}\dfrac{1}{\tau_{11}\tau_{21}} & 0\\ 0 & 0\\ 0 & \dfrac{1}{\tau_{12}\tau_{22}}\\ 0 & 0\end{pmatrix}\begin{pmatrix}u_1\\ u_2\end{pmatrix}. \quad (25)$$

To design the IMC controller for this process, first decompose the process into a part to be inverted and a noninverted part. Since there are no right half-plane transmission zeros, we elect to invert as much of the process model as possible. There are two choices depending on whether the smallest delays are on diagonal or not.

*Case 1.* The smallest delays are on the diagonal or can be arranged on the diagonal by row or column swapping, that is,

$$\theta_{11}\le\theta_{12} \quad \text{and} \quad \theta_{22}\le\theta_{21}$$

or

$$\theta_{12}\le\theta_{11} \quad \text{and} \quad \theta_{21}\le\theta_{22}.$$

In this case, the smallest delays in response can be obtained by diagonal decoupling. Consider $\theta_{11}\le\theta_{12}$ and $\theta_{22}\le\theta_{21}$. We can decompose process $G(s)$ as

$$G(s)=\begin{pmatrix}e^{-\theta_{11}s} & 0\\ 0 & e^{-\theta_{22}s}\end{pmatrix}\times\begin{pmatrix}\dfrac{k_{11}}{\tau_{11}s+1} & \dfrac{k_{12}}{\tau_{12}s+1}e^{-(\theta_{12}-\theta_{11})s}\\ \dfrac{k_{21}}{\tau_{21}s+1}e^{-(\theta_{21}-\theta_{22})s} & \dfrac{k_{22}}{\tau_{22}s+1}\end{pmatrix}. \quad (26)$$

Then we have $K(s)$, according to Eq. 7a, as

$$K(s)=\begin{pmatrix}K_{sp_{11}} & 0\\ 0 & K_{sp_{22}}\end{pmatrix}\begin{pmatrix}(\epsilon_1 s+1) & 0\\ 0 & (\epsilon_2 s+1)\end{pmatrix}\begin{pmatrix}e^{\theta_{11}s} & 0\\ 0 & e^{\theta_{22}s}\end{pmatrix}\begin{pmatrix}k_{11}(\tau_{21}s+1)e^{-\theta_{11}s} & k_{12}(\tau_{22}s+1)e^{-\theta_{12}s}\\ k_{21}(\tau_{11}s+1)e^{-\theta_{21}s} & k_{22}(\tau_{12}s+1)e^{-\theta_{22}s}\end{pmatrix}-\begin{pmatrix}(\tau_{11}s+1)(\tau_{21}s+1) & 0\\ 0 & (\tau_{12}s+1)(\tau_{22}s+1)\end{pmatrix}. \quad (27)$$

Let $T_1=(\theta_{12}-\theta_{11})$ and $T_2=(\theta_{21}-\theta_{22})$. Multiplying terms in Eq. 27 gives

$$K(s)=\begin{pmatrix}K_{sp_{11}} & 0\\ 0 & K_{sp_{22}}\end{pmatrix}\times\begin{pmatrix}k_{11}(\tau_{21}s+1)(\epsilon_1 s+1) & k_{12}(\tau_{22}s+1)(\epsilon_1 s+1)e^{-T_1 s}\\ k_{21}(\tau_{11}s+1)(\epsilon_2 s+1)e^{-T_2 s} & k_{22}(\tau_{12}s+1)(\epsilon_2 s+1)\end{pmatrix}-\begin{pmatrix}(\tau_{11}s+1)(\tau_{21}s+1) & 0\\ 0 & (\tau_{12}s+1)(\tau_{22}s+1)\end{pmatrix}. \quad (28)$$

Notice that diagonal elements of matrix $K(s)$ are second-order polynomials in $s$. The off-diagonal elements are also second-order polynomials, though with delay elements. Here, unlike the minimum phase process design, we can only reduce the diagonal elements to first order by making the coefficients of $s^2$ zero. However, the off-diagonal terms cause no difficulties because we need only the past values of the second derivative [that is, $\ddot{x}(t-T_1)$ and $\ddot{x}(t-T_2)$], and these are already available. The $K_{sp}$ matrix for the earlier example is

$$K_{sp}=HE^{-1},$$

where

$$H=\begin{pmatrix}\dfrac{\tau_{11}}{k_{11}} & 0\\ 0 & \dfrac{\tau_{22}}{k_{22}}\end{pmatrix} \quad \text{and} \quad E=\begin{pmatrix}\epsilon_1 & 0\\ 0 & \epsilon_2\end{pmatrix}. \quad (29)$$

To extend this procedure to any order process with delays, simple form $D(s)$ as a diagonal matrix of polynomials such that each diagonal element, $d_{ii}$, is the product of the denominators in column i of $G(s)$. The matrix $N(s)$ is formed by postmultiplying the process matrix by $D(s)$.

*Case 2.* The smallest delays are not on the diagonal. In this case, diagonal decoupling does not give the fastest initial response, and the general form of the IMC controller depends on the kind of decoupling desired. Diagonal decoupling can be obtained by delaying one or more controls so that the resulting process matrix can again be rearranged to have the smallest dead times on the diagonal. Alternatively, one can use triangular decoupling to get the fastest possible initial response of the outputs within some rank ordering. We defer a complete discussion of triangular decoupling to a future article. The following example illustrates diagonal decoupling.

*Example 2.* This is Shell Oil's heavy oil fractionator control problem (Prett and Morari, 1987). It has smallest delays on the diagonal, and hence diagonal decoupling gives the smallest possible delays in the output response:

$$G(s)=\begin{pmatrix} \dfrac{4.05}{50s+1}e^{-27s} & \dfrac{1.77}{60s+1}e^{-28s} \\ \dfrac{5.39}{50s+1}e^{-18s} & \dfrac{5.72}{60s+1}e^{-14s} \end{pmatrix}. \qquad (30)$$

The invertible part of the process is given by

$$G-(s)=\begin{pmatrix} \dfrac{4.05}{50s+1} & \dfrac{1.77}{60s+1}e^{-s} \\ \dfrac{5.39}{50s+1}e^{-4s} & \dfrac{5.72}{60s+1} \end{pmatrix}. \qquad (31)$$

We choose $D(s)$ and $N(s)$ as

$$D(s)=\begin{pmatrix} (50s+1)^2 & 0 \\ 0 & (60s+1)^2 \end{pmatrix}$$

$$N(s)=\begin{pmatrix} 4.05(50s+1)e^{-27s} & 1.77(60s+1)e^{-28s} \\ 5.39(50s+1)e^{-18s} & 5.72(60s+1)e^{-14s} \end{pmatrix}. \qquad (32)$$

$$K_{sp}=\begin{pmatrix} 12.34\epsilon_1^{-1} & 0 \\ 0 & 10.49\epsilon_2^{-1} \end{pmatrix} \qquad (33)$$

Notice the presence of delays in off-diagonal terms that make it possible to feed back the past derivatives of the states.

### *Processes with right half-plane transmission zeros*

For the processes with RHP transmission zeros, MSF implementations of diagonal decoupling controllers is not possible using a constant matrix for $K_{sp}$. The problem arises from the fact that the IMC controller for diagonal decoupling control also has one or more implicit right half-plane transmission zeros, so the inverse of the IMC controller, which is required to compute the feedback matrix $K(s)$ (cf. Eq. 7a), is unstable. To see this, consider the process model that is factored as $G(s)=G_+(s)\ G_-(s)$. $G_+(s)$ is a diagonal matrix containing the smallest dead times in each row of $G(s)$, while $G_-(s)$ contains all right half-plane transmission zeros of $G(s)$. The diagonal decoupling IMC controller is adj$(G_-(s))$/det* $(G_-(s))$, where det* $(G_-(s))$ is the determinant of $G_-(s)$, with the right half-plane transmission zeros replaced by their mirror image about the imaginary axis. The inverse of adj$(G_-(s))$ is $(G_-(s))$/det$(G_-(s))$, which is unstable because of the right half-plane transmission zeros in the determinant of $G_-(s)$.

In the following section we discuss two methods of triangularly decoupling processes with RHP zeros.

## Triangular Decoupling

*Procedure 1.* This procedure is based on the triangular decoupling procedure described by Morari and Zafiriou (1989). They have suggested that when the process $G(s)$ has a single zero, $1/\tau$, the noninvertible part $G_+(s)$ can be written in the form

$$G+(s)=\begin{pmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ 0 & \vdots & \ddots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & 0 \\ \dfrac{\beta_1 s}{\tau s+1} & \dfrac{\beta_2 s}{\tau s+1} & \cdots & \cdots & \dfrac{\beta_{n-1}s}{\tau s+1} & \dfrac{-\tau s+1}{\tau s+1} \end{pmatrix}. \qquad (35)$$

Notice that the noninvertible part $G(s)$ carries the right half-plane zero in the *nn*th element.

To find the unknown $\beta$s, we can use the condition that the individual elements in the feedback matrix have to be polynomials. This procedure is illustrated in following example.

$$K(s)=\begin{pmatrix} \left(2{,}500\epsilon_1^{-1}-50\right)s+\left(50\epsilon_1^{-1}-1\right) & \left(1{,}311s^2+\left(1{,}311\epsilon_1^{-1}+22\right)s+22\epsilon_1^{-1}\right)e^{-s} \\ \left(2{,}827s^2+\left(2{,}827\epsilon_2^{-1}+56.5\right)s+56.5\epsilon_2^{-1}\right)e^{-4s} & \left(3{,}600\epsilon_2^{-1}-60\right)s+\left(60\epsilon_2^{-1}-1\right) \end{pmatrix}. \qquad (34)$$

*Example 3.* The process model is

$$G(s)=\begin{pmatrix} \dfrac{1}{s+1} & \dfrac{2}{3s+1} \\ \dfrac{1}{2s+1} & \dfrac{1}{s+1} \end{pmatrix}. \tag{36}$$

This process has the right half-plane zero at $s=0.39$. Hence, the noninvertible part is chosen as

$$G_+(s)=\begin{pmatrix} 1 & 0 \\ \dfrac{\beta s}{2.56s+1} & \dfrac{-2.56s+1}{2.56s+1} \end{pmatrix}. \tag{37}$$

Now according to Eq. 7a, for the feedback matrix $K(s)$ to be polynomial, $G_+(s)^{-1}N(s)$ has to be a polynomial. Carrying out the required multiplication, we get

$$G_+(s)^{-1}N(s)=\begin{pmatrix} 2s+1 & 2(2s+1) \\ \dfrac{-\beta s(2s+1)+(2.56s+1)(s+1)}{-2.56s+1} & \dfrac{-2\beta s(s+1)+(2.56s+1)(3s+1)}{-2.56s+1} \end{pmatrix}. \tag{38}$$

To make individual elements in the matrix of Eq. 38 polynomials, we have to insist that the numerators of the (2,1) and (2,2) elements each have a zero at $s=0.39$. From this condition we get

$$\beta=\left.\frac{(2.56s+1)(s+1)}{s(2s+1)}\right|_{s=0.39}=4, \tag{39}$$

and the noninvertible part becomes

$$G_+(s)=\begin{pmatrix} 1 & 0 \\ \dfrac{4s}{2.56s+1} & \dfrac{-2.56s+1}{2.56s+1} \end{pmatrix}. \tag{40}$$

The preceding procedure requires that we invert $G_+(s)$. This inverse is relatively easy to obtain, because the matrix partitions into a lower triangular matrix with $(n-1)\times(n-1)$ identity matrix and a nonzero last row (cf. Eq. 35).

*Procedure 2.* This procedure is similar to that given earlier in that it relies on the fact that all the RHP transmission zeros of an $n\times n$ process matrix can be pushed to the least important output, unless the same zeros are present in all the minors of order $(n-1)\times(n-1)$. If the same zero does not appear in all the minors, then some combination of outputs and inputs exist for the $(n-1)\times(n-1)$ system, which will be free of RHP zeros and the RHP zero will appear only in the last remaining output. For further discussion, refer to Rosenbrock (1974). The least important output is usually arranged as the last output.

Let the $n\times n$ process $G(s)$ have the smallest dead times along the diagonal, and have multiple RHP transmission zeros that can be pushed to the last output. The process model can therefore be partitioned as

$$G(s)=G_+(s)G_-(s), \tag{41}$$

where $G_+(s)=$ a diagonal matrix containing the smallest dead times in each row of $G(s)$.

By the preceding factorization, $G_-(s)$ has no dead times along its diagonal, and can be partitioned so that all of its right half-plane zeros appear in the last output. To simplify notation, we let

$$G_-(s)\equiv P(s). \tag{42}$$

We now define the minimum phase outputs of $P(s)$ as $y(s)$ and the nonminimum phase output as $y_n(s)$.

$$y(s)=P_{11}(s)u(s)+P_{1n}(s)u_n(s) \tag{43}$$

$$y_n(s)=P_{n1}(s)u(s)+p_{nn}(s)u_n(s), \tag{44}$$

where
$y(s)$ and $u(s)$ are $n-1$ vectors;
$y_n(s)$ and $u_n(s)$ are scalars;
$P_{11}(s)$ is a $(n-1)\times(n-1)$ submatrix that does not have any RHP transmission zeros;
$P_{1n}$ is a $(n-1)\times 1$ column vector;
$P_{n1}$ is a $1\times(n-1)$ row vector;
$P_{nn}$ is the $nn$th element in $P(s)$.

Let $u(s)$ be the control that decouples the elements of $y(s)$ from each other and from $u_n(s)$. That is,

$$u(s)=P_{11}^{-1}(s)\left(F_{11}(s)(r-\tilde{d})-P_{1n}(s)u_n(s)\right), \tag{45}$$

where $F_{11}(s)$ is a diagonal matrix of filter time constants.

Substituting Eq. 45 into Eqs. 43 and 44 gives

$$y(s)=F_{11}(s)(r-\tilde{d}) \tag{46}$$

$$y_n(s)=P_{n1}(s)P_{11}^{-1}(s)F_{11}(s)(r-\tilde{d}) + \left[p_{nn}(s)-P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)\right]u_n(s). \tag{47}$$

The term $[p_{nn}(s)-P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)]$ in Eq. 47 contains all of the right half-plane zeros of the original system, $G(s)$. The problem of deriving a model state-feedback controller for Eq. 41 can be viewed as (1) evaluating the term, $[p_{nn}(s)-P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)]$ without actually computing $P_{11}^{-1}(s)$; and (2) finding a stable control law for $u_n(s)$ that gives a reasonable response of $y_n(s)$ to its setpoint and disturbances.

We denote our approximation to the term $[p_{nn}(s)-P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)]$ as

$$h_{nn}(s)\approx\left[p_{nn}(s)-P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)\right]. \tag{48}$$

Let
$q_{nn}(s)$ = the stable IMC lead/lag controller that we form by inverting part of $h_{nn}$(s)
$f_{nn}(s)$ = the filter associated with $q_{nn}(s)$.
From this, $u_n(s)$ is chosen as

$$u_n(s) = q_{nn}(s)\left[\left(r_n - \tilde{d}_n\right) - f_{nn}^{-1}(s)P_{n1}(s)P_{11}^{-1}(s)F_{11}(s)\left(r - \tilde{d}\right)\right]. \quad (49)$$

Substituting Eq. 49 into Eq. 47, gives

$$y_n(s) = P_{n1}(s)P_{11}^{-1}(s)\left(1 - h_{nn}(s)q_{nn}(s)f_{nn}^{-1}(s)\right) \times F_{11}(s)\left(r - \tilde{d}\right) + h_{nn}(s)q_{nn}(s)\left(r_n - \tilde{d}_n\right). \quad (50)$$

The "predicted" outputs $y(s)$ and $y_n(s)$ from Eq. 46 and 50 can be rewritten in the form:

$$y(s) \equiv \begin{pmatrix} \underline{y}(s) \\ y_n(s) \end{pmatrix} = \begin{pmatrix} I & 0 \\ P_{n1}P_{11}^{-1}(1 - h_{nn}q_{nn}^*) & h_{nn}q_{nn}^* \end{pmatrix} \begin{pmatrix} F_{11} & 0 \\ 0 & f_{nn} \end{pmatrix} \begin{pmatrix} \underline{r} - \underline{\tilde{d}} \\ r_n - \tilde{d}_n \end{pmatrix} \equiv P_+(s)F(s)\left(r - \tilde{d}\right), \quad (51)$$

where $q_{nn}^*(s) \equiv q_{nn}(s)f_{nn}^{-1}(s)$. That is, $q_{nn}^*(s)$ does not contain the filter, and is, in general, an improper transfer function.

From $P_+(s)$ we can compute the invertible part of $P(s)$, $P_-(s)$, which is

$$P_-(s) = \begin{pmatrix} P_{11} & P_{1n} \\ P_{n1} & \left(\dfrac{P_{nn} - P_{n1}P_{11}^{-1}P_{1n}}{h_{nn}q_{nn}^*} + P_{n1}P_{11}^{-1}P_{1n}\right) \end{pmatrix}. \quad (52a)$$

Applying the approximation given by Eq. 48 yields:

$$P_-(s) \approx \begin{pmatrix} P_{11} & P_{1n} \\ P_{n1} & \left(\dfrac{1}{q_{nn}^*} + p_{nn} - h_{nn}\right) \end{pmatrix}. \quad (52b)$$

We are now in a position to implement the model state feedback using Eq. 52b as $P_-(s)$. In this case, the actual response of the system will only approximate that given by Eq. 51.

*Example 4.* In this example, we elect to evaluate the term $[p_{nn}(s) - P_{n1}(s)P_{11}^{-1}(s)P_{1n}(s)]$ by approximating $P_{11}^{-1}(s)$ as a matrix containing only first-order lead terms of the form:

$$\left[P_{11}^{-1}(s)\right]_{ij} \approx \kappa_{ij}\left(\tau_{ij}s + 1\right), \quad (53)$$

where $\kappa_{ij} = (P_{11}^{-1}(0))_{ij}$, $\tau_{ij}/\kappa_{ij} = ([(P_{11}(s))_{ij}s^{r_j}]_{s=\infty})^{-1}$, and $r_j$ is the order of the $jj$th element of the diagonal IMC filter. Note that from Eq. 6, the $\tau_{ij}$ can also be obtained by postmultiplying $Q_{IMC}(\infty)$ or $K_{sp}$ by a diagonal matrix $(\epsilon_{ii}^{r_i})$.

The motivation for the approximation given by Eq. 53 is that (1) an IMC controller formed from Eq. 53 and a diagonal filter of first-order lags captures both the initial and final time-domain behavior of a step input to $P_{11}^{-1}(s)F(s)$; and (2) the resulting control law can be relatively easily implemented by the model state feedback. To illustrate, consider the following $3\times3$ process, which has an RHP zero at $s = 0.39$:

$$G(s) = \begin{pmatrix} \dfrac{1}{s+1} & \dfrac{2}{3s+1} & \dfrac{1}{s+1} \\ \dfrac{3}{4s+1} & \dfrac{-1}{s+1} & \dfrac{-4}{3s+1} \\ \dfrac{1}{2s+1} & \dfrac{1}{s+1} & \dfrac{1}{2s+1} \end{pmatrix} \quad (54)$$

$$G_{11}(s) = \begin{pmatrix} \dfrac{1}{s+1} & \dfrac{2}{3s+1} \\ \dfrac{3}{4s+1} & \dfrac{-1}{s+1} \end{pmatrix} \quad G_{1n}(s) = \begin{pmatrix} \dfrac{1}{s+1} \\ \dfrac{-4}{3s+1} \end{pmatrix} \quad (55)$$

$$G_{n1} = \begin{pmatrix} \dfrac{1}{2s+1} & \dfrac{1}{s+1} \end{pmatrix} \quad g_{nm} = \frac{1}{2s+1}. \quad (56)$$

To estimate $G_{11}^{-1}(s)$ from Eq. 53, we note that

$$G_{11}^{-1}(0) = \frac{1}{7}\begin{pmatrix} 1 & 2 \\ 3 & -1 \end{pmatrix}$$
$$(\tau/\kappa)_{ij} = \begin{pmatrix} 2/3 & 4/9 \\ 1/2 & -2/3 \end{pmatrix}. \quad (57)$$

Therefore

$$G_{11}^{-1}(s) \approx \frac{1}{7}\begin{pmatrix} \left(\dfrac{14}{3}s+1\right) & 2\left(\dfrac{14}{9}s+1\right) \\ 3\left(\dfrac{7}{6}s+1\right) & -\left(\dfrac{14}{3}s+1\right) \end{pmatrix}. \quad (58)$$

Using this approximation for $G_{11}^{-1}(s)$, the $nn$th term in $G(s)$ will be

$$\left(g_{nn}(s) - G_{n1}(s)G_{11}^{-1}(s)G_{1n}(s)\right) \approx \frac{(-2.44s+1)(2.2s+1)}{(3s+1)(2s+1)(s+1)} \equiv h_{nn}(s). \quad (59)$$

Let

$$q_{nn}^*(s) = \frac{(3s+1)(2s+1)(s+1)}{(2.44s+1)(2.2s+1)}. \quad (60)$$

Substituting Eqs. 59 and 60 into Eq. 52b, gives

$$G_-(s) \approx \begin{pmatrix} \dfrac{1}{s+1} & \dfrac{2}{3s+1} & \dfrac{1}{s+1} \\ \dfrac{3}{4s+1} & \dfrac{-1}{s+1} & \dfrac{-4}{3s+1} \\ \dfrac{1}{2s+1} & \dfrac{1}{s+1} & \dfrac{(6.8s+1)}{(3s+1)(s+1)} \end{pmatrix}. \quad (61)$$

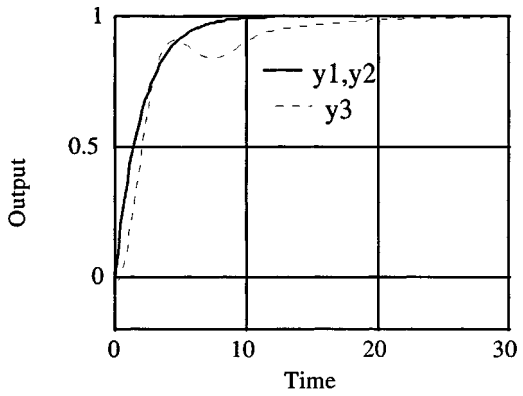


**Figure 2. Response of the system with triangular decoupling.**

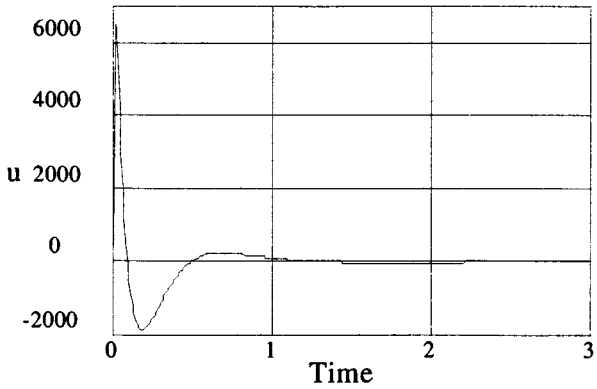


**Figure 3. Heat exchanger control effort for $\epsilon = 0.2c$**

The model state-feedback controller can be calculated using the preceding approximation to $G_{-}(s)$ in Eq. 7a. Figure 2 shows the response of the system for a setpoint change of 1 in all variables, and $\epsilon_1 = \epsilon_2 = \epsilon_3 = 2$. As expected, $y_1$ and $y_2$ are decoupled and follow their filter responses. The RHP zero is pushed to the least important variable, $y_3$.

## Tuning for Stability and Performance in the Presence of Control Effort Limits

In this section we present a method for setting lower bounds on the IMC filter time constants that assure good performance of a model state-feedback control system for perfect models in spite of control effort constraints. Limited numerical results for several systems indicate that these lower bounds are somewhat larger than the bounds obtained by limiting the noise amplification of the channels of the IMC controller to less than 20. However, the computed lower bounds are usually a good deal smaller than the filter time constants required to give good performance for uncertain processes (Stryczek and Brosilow, 1996).

### *The tuning algorithm*

1. *Initialization:* Transform all the lower bounds on control effort to zero, set the upper bound to infinity, and assign arbitrary positive values to all filter time constants.

2. Starting at steady state with all controls at zero, make an arbitrary positive step change in a process variable. Adjust the filter constant associated with that variable so that *all* computed control efforts are equal to or greater than zero for their entire trajectories.

3. Repeat step 2 for all process variables.

### *Discussion*

Transforming all control effort lower bounds to zero is convenient for the developments in the Appendix, but is not actually necessary. What is important is that the tuning start at a steady state. The control effort lower bounds can be temporarily set at the steady-state values of the controls. Setting the upper (lower) bounds to infinity ($-$infinity) is also not necessary provided that the setpoint changes are small enough that the control effort upper (lower) bounds are not encountered. The idea is that we wish to drive the process as hard as possible in the positive (negative) direction so as to find the smallest filter time constant so that none of the computed controls will exceed their lower (upper) bounds. In the Appendix we show that the controls computed by an IMC controller tuned as earlier will never exceed their lower (upper) bounds for arbitrary, and simultaneous, changes in all process variables.

The preceding tuning procedure does not prevent controls from hitting their upper (lower) bounds. The question then is

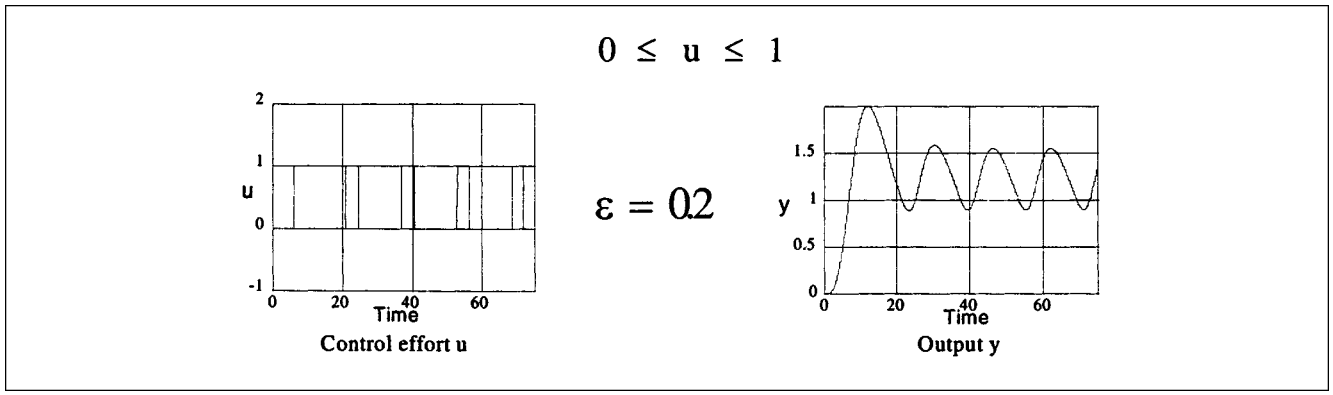


**Figure 4. Instability due to control effort saturation.**

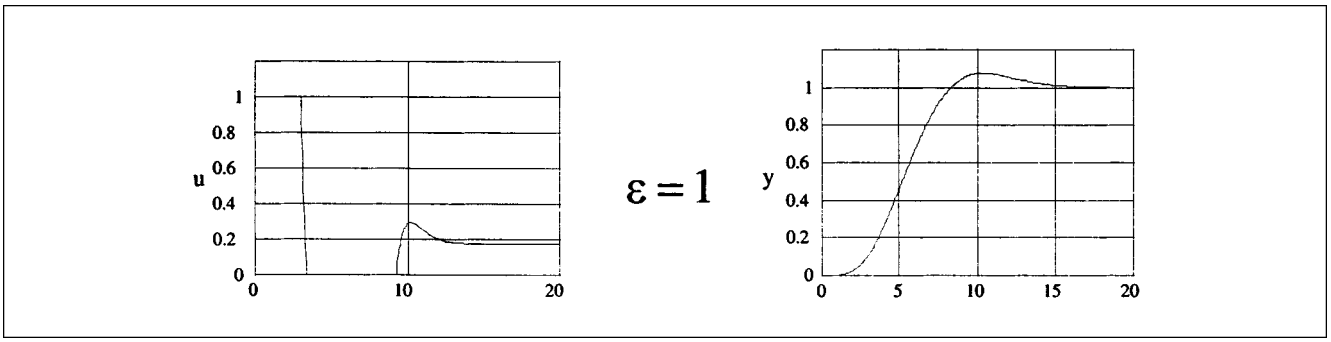


**Figure 5. Overshoot in the output due to control effort saturation.**

how the control system will behave when such bounds are encountered. Qualitatively we argue that encountering upper (lower) bounds simply slows the control system down because of the method we use to bring the controls back into the constraint set. At any instant when constraints are encountered, the filter time constants are increased according to priorities set by the engineer (such as increase the filter time constants in inverse order of the importance of the associate process variable) to bring all the controls into the constraint set. Apply the controls so calculated and reset the filter time constants to their original values before computing the controls to be computed for the next instant. What we attempt to ensure by the preceding tuning algorithm is that there will always exist a set of filter time constants, larger than the nominal (that is, preset) filter time constants, that will bring the controls into the constraint set. Unfortunately, we have, as yet, been unable to prove that we can always find the desired filter time constants. However, numerical experience with a number of different types of SISO and MIMO processes, as given below, indicates that the tuning algorithm does indeed provide good performance in spite of constraints.

*Example 5.* Here we consider an SISO system. The process is given as

$$P = \frac{5.64}{(8s+1)(3.1s+1)(1.7s+1)(1.4s+1)}. \qquad (62)$$

For the filter constant of $\epsilon = 0.2$, the control effort is as shown in Figure 3.

If there are constraints on control effort $0 \le u \le 1$, the system is unstable and limits cycles as shown in Figure 4.

For the filter constant of $\epsilon = 1$, the system is stable, but the output shows some overshoot, as can be seen in Figure 5.

Carrying out the tuning procedure described earlier, the filter time constant obtained is 1.3. Figure 6 shows the control effort and the response of the tuned system.

The preceding tuning procedure gives us the fastest response possible without having to worry about stability with the given constraints on manipulated variables. If the initial jump in controller output shown in Figures 5 and 6 causes problems in operator acceptance of the MMSF control system, the jump can be removed by filtering step setpoint changes through a first-order lag. However, such a lag cannot be used when tuning the control system, as described in the fifth section.

*Example 6.* Here we consider the process that has stable zeros very near the imaginary axis:

$$p(s) = \frac{s^2 + 0.001s + 1}{(s+1)^4}. \qquad (63)$$

As an extreme test of the tuning algorithm, we choose the controller, $q(s)$, given by Eq. 64 as the inverse of the process multiplied with the filter. Since the numerator of Eq. 63 has zeros very near the imaginary axis, such a choice will result in a very oscillatory control effort, as can be seen from Figure 7. Since such oscillations are generally unacceptable, we will

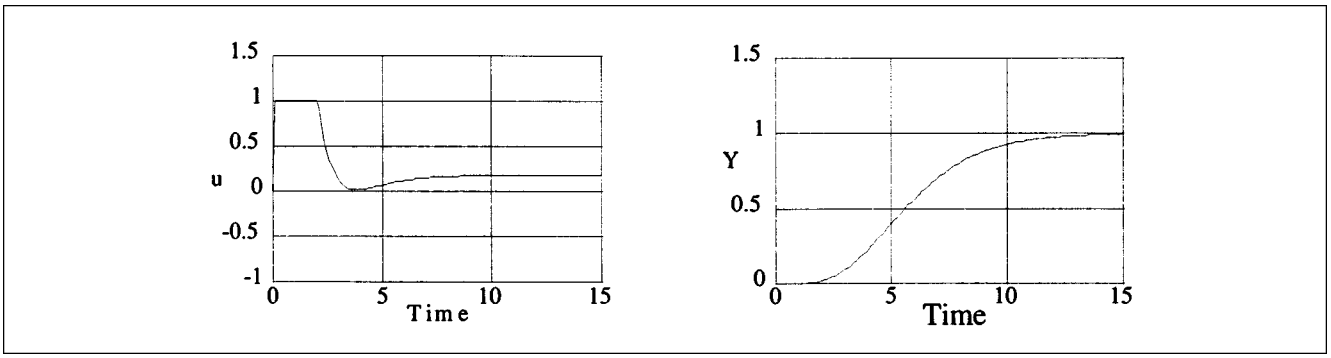


**Figure 6. Response of the tuned system with control effort saturation; $\epsilon = 1.3$.**

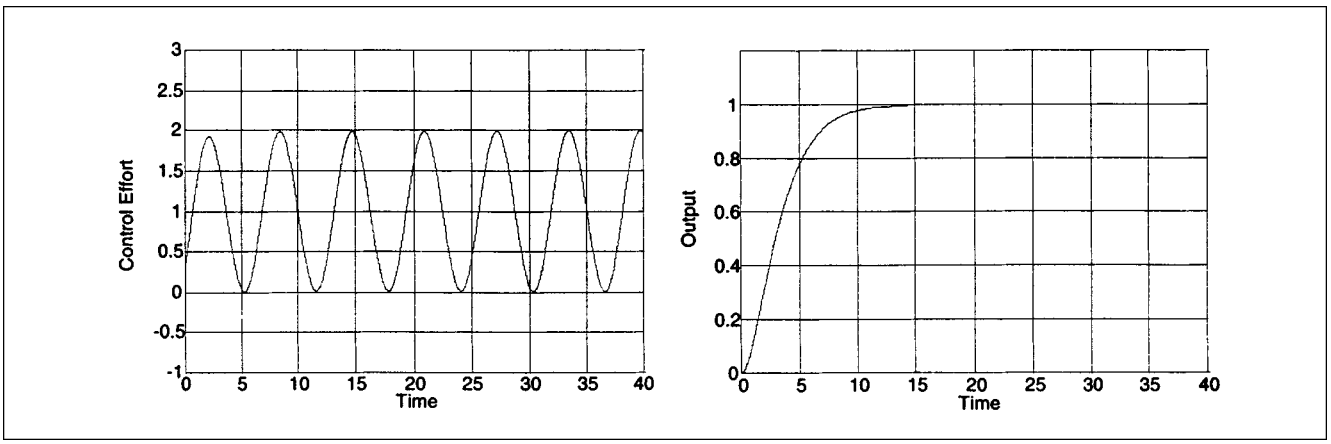


**Figure 7. Response of the tuned system with $\epsilon = 1.75$.**

provide a more reasonable design presently:

$$q(s) = \frac{(s+1)^4}{(s^2+0.001s+1)(\epsilon s+1)^2}. \tag{64}$$

Let the lower bound on the control effort be 0, with no upper bound. Applying the tuning procedure gives $\epsilon = 1.75$. Figure 7 shows the control effort and the output obtained for a perfect model.

While the control effort is quite oscillatory, as might be expected for a controller with a damping ratio of 0.0005, the system is stable, and the control effort does settle in 10,000 time units.

The response of the system with the same filter constant and an upper bound of 1.2 on the control effort is shown in Figure 8. Notice that the control effort never hits the lower bound and the process remains stable.

To avoid oscillating control effort, zeros near the imaginary axis should not be inverted. Instead, we suggest using the damping ratio near 0.5 in the controller. That is, we suggest using a controller of the form:

$$q(s) = \frac{(s+1)^4}{(\epsilon s+1)^4}. \tag{65}$$

A maximum noise amplification factor of 20 for the preceding controller requires a filter time constant of 0.473 or greater. Tuning the controller for saturation yields a filter time constant of 0.5. As can be seen from Figure 9, the control effort from Eq. 65 does not oscillate, and the output response is faster than that in Figure 7, but does not approach one monotonically.

Figure 10 shows the control system response when the upper bound is the same as that for Figure 8 (that is, 1.2).

*Example 7.* This is a $2\times2$ process with constraints on the control efforts:

$$G(s) = \begin{pmatrix} \dfrac{-1}{24.8s^2+11.1s+1} & \dfrac{3}{2s^2+3s+1} \\ \dfrac{1}{12s^2+7s+1} & \dfrac{-1}{2.8s^2+3.1s+1} \end{pmatrix} \tag{65}$$

$$0 \le u_1, u_2 \le 10.$$

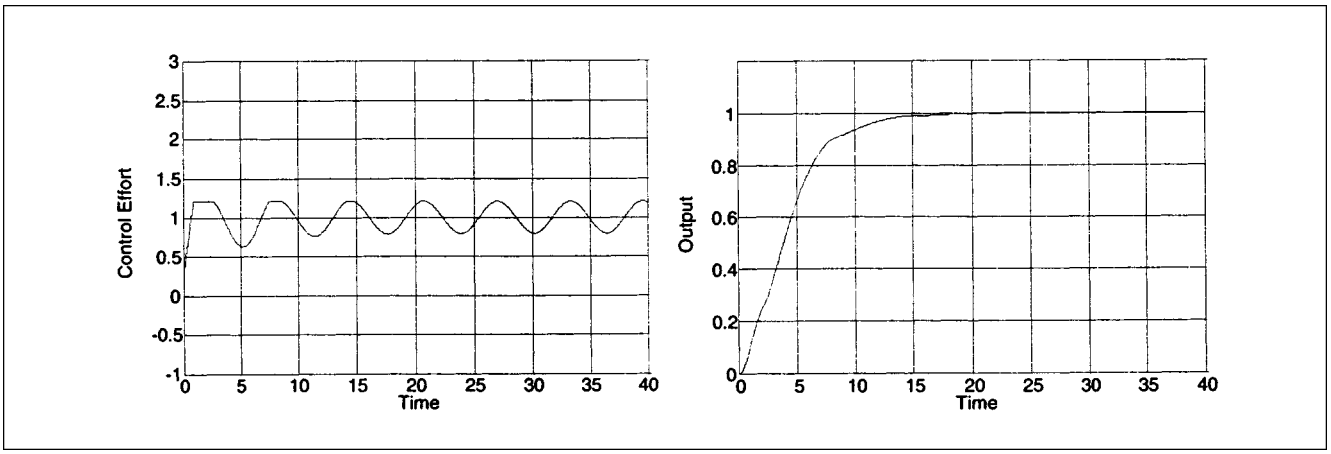


**Figure 8. Response of the tuned system with $\epsilon = 1.75$ and upper bound of 1.2 on the control effort.**

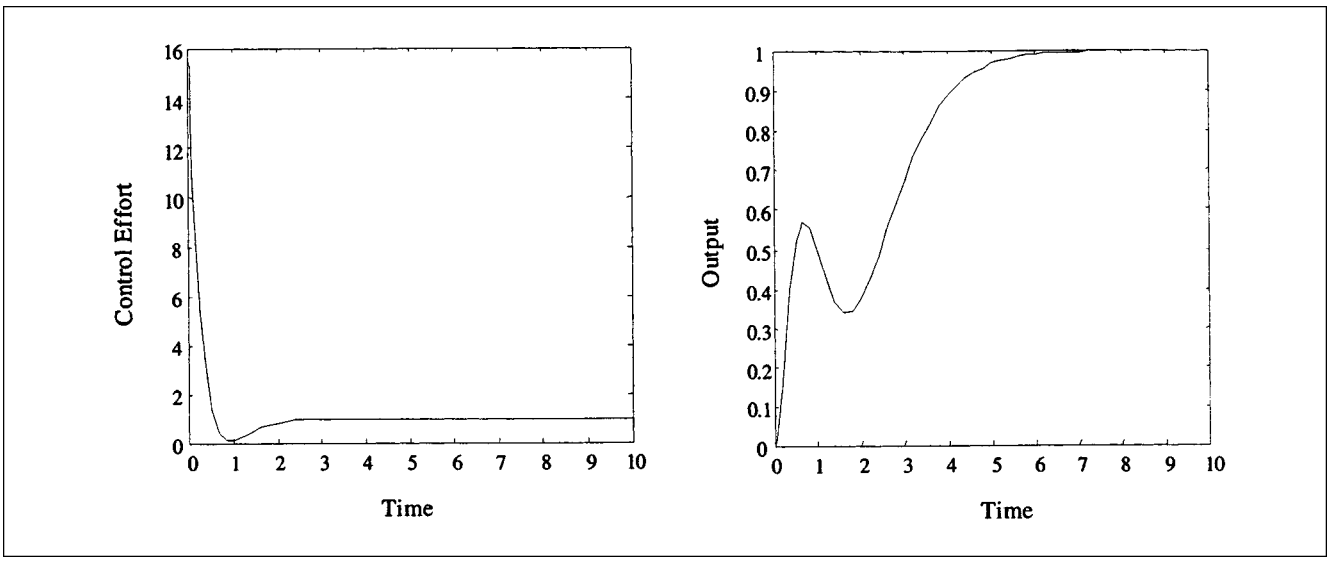


**Figure 9. Response of the tuned system with $\epsilon = 0.14$.**

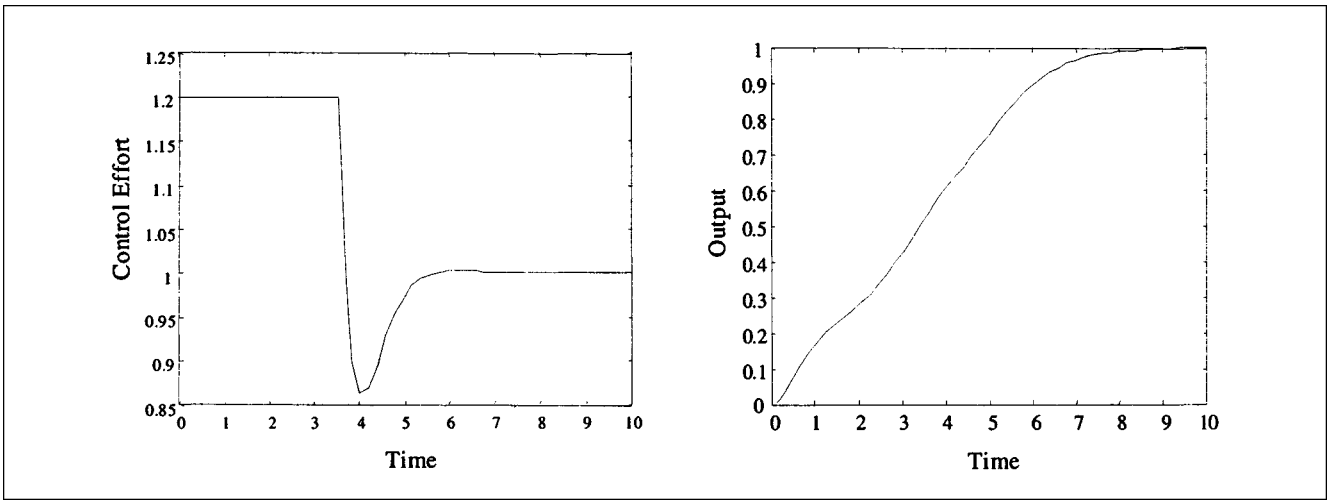


**Figure 10. Response of the tuned system with $\epsilon = 0.5$ and upper bound of 1.2 on the control effort.**

Using the tuning procedure, we first make a step change in setpoint $r_1$ and no change in setpoint $r_2$ (that is, we make a step change in direction [1, 0]). Using an arbitrary value for $\epsilon_2$, $\epsilon_1$ is adjusted so that $u_1$ and $u_2$ are at or above zero for the entire response. For the given example, such a filter constant is $\epsilon_1 = 0.53$. The control effort responses are shown in Figure 11. Similarly, we find $\epsilon_2 = 0.695$, and the control efforts for this tuning are shown in Figure 12. Once the filter constants are tuned, the control efforts for step change in any direction will never hit the constraint at zero. This is illustrated in Figure 13.

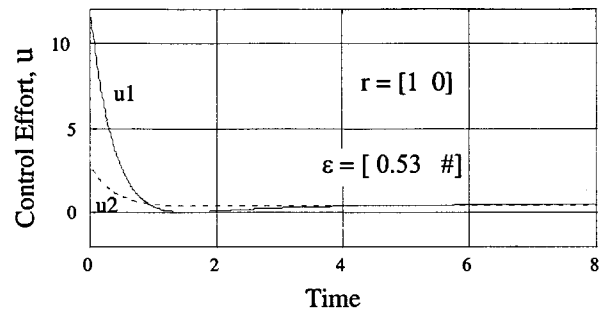


**Figure 11. Tuning filter-constant $\epsilon_1$.**

Figure 14 shows the control effort and output response of the system for the step change of [1, 1], with both the upper and lower bound constraints. Both filter time constants are temporarily increased proportionately when the controls saturate, as in the initial part of the responses shown in Figure 14.

## Accommodating Loss of Controls and Tuning One Process Variable at a Time

From the model state-feedback diagram of Figure 1, the control effort vector can be written as

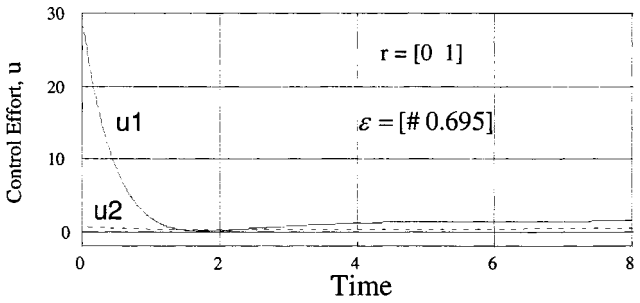


**Figure 12. Tuning for filter-constant $\epsilon_2$.**

$$u(t) = K_{sp}\left[r - \tilde{d}(t)\right] - Kx(t-T) \tag{66}$$

Where $K$ and $K_{sp}$ are constant matrices; $x(t-T)$ is the entire state vector of the model; $T$ is a vector of delays, some of whose elements are zero (cf. Eqs. 28 or 34); $\tilde{d}(t)$ is the current vector of disturbance estimates; and $r$ is the vector of setpoints.

At any instant, the vectors $\tilde{d}(t)$ and $x(t-T)$ are invariant with respect to instantaneous changes in the control vector, $u(t)$. Therefore, we can force the control vector to take on any values we wish at any instant simple by changing the setpoint vector, $r$. To illustrate, assume that control effort one [that is, $u_j(t)$] has failed at some fixed value, $\bar{u}_1$. If the first row of $K_{sp}$ has no zero elements, then we can choose to adjust the setpoint of any output to force $u_1(t)$ to assume a value of $\bar{u}_1$. If the outputs are ordered according to their importance, the first being the most important, then we can elect to allow the $n$th output to deviate from its actual setpoint, and change $r_n$ to force $u_1(t)$ to take on a value of $\bar{u}_1$ at every instant. All of the other controls will, of course, also change so as to maintain outputs one through $n-1$ at their setpoints. Using the same approach, we can force any number of controls to fixed values. Boyce and Brosilow (1996) successfully apply the foregoing approach to some simple multivariable processes.

The ability to place any number of controls into manual operation can be used to tune the control system on-line, one process variable at a time. We note, however, that as with all on-line tuning methods, the method that we are about to describe is only valid at the current process operating point. For tuning over the entire operating range, we recommend ''off-line'' tuning using model uncertainty descriptions and $H_\infty$ methods (Stryczek and Brosilow, 1996; Zhou and Doyle,

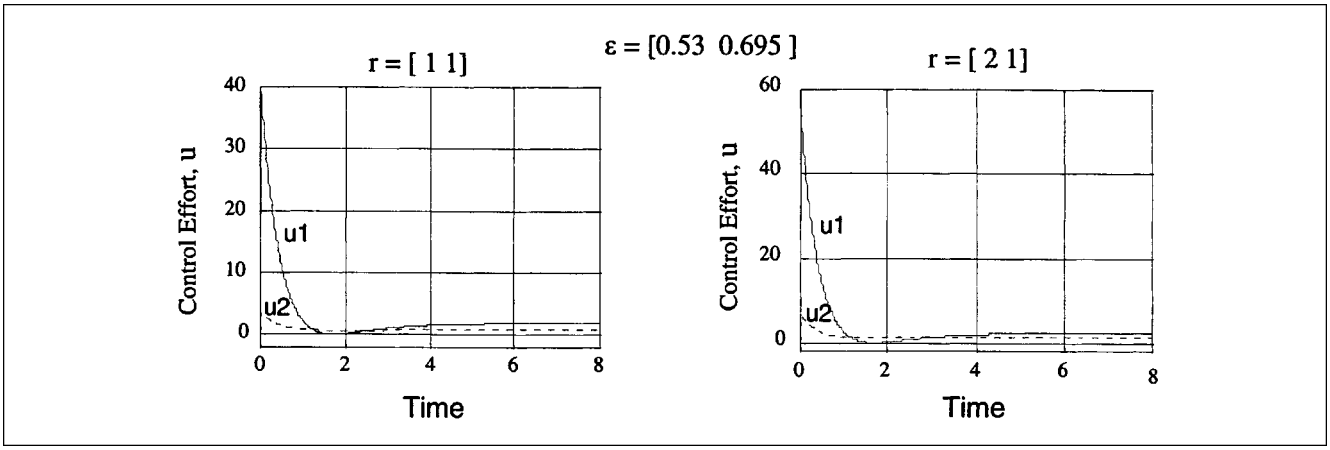


**Figure 13. Control effort for the tuned system for step changes of different directions and magnitudes.**

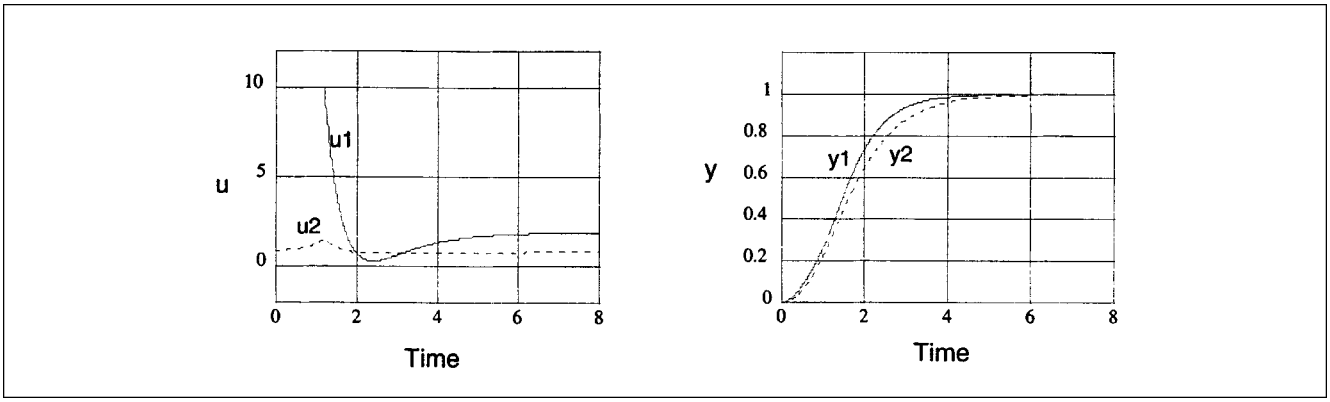


**Figure 14. Control effort and output for the tuned system for step change of [1, 1] and both upper and lower constraints active.**

1998). Let us assume again that the outputs are ordered according to their importance, the first being the most important, and that we have designed our control system for decoupled control. Of course, modeling errors will introduce some coupling. To initialize the procedure, choose nominal values for all filter time constants. Next, choose any control effort for the first output so that the square submatrix of $K_{sp}$ relating the remaining $n-1$ controls to the remaining $n-1$ setpoints is nonsingular. For Example 1, either control effort may be chosen, while for Example 2, only control effort 1 is allowable. Continuously adjust the setpoints $r_2, \ldots, r_n$ so that controls $u_2, \ldots, u_n$ maintain the desired fixed values. Adjust the filter time constant associated with the first output (that is, $\epsilon_1$) to obtain a desirable setpoint response. Next, select a control effort for output 2 so that the resulting $n-2 \times n-2$ submatrix of $K_{sp}$ is nonsingular. Adjust $\epsilon_2$ to obtain a desirable setpoint response for the second output. Increase the value for $\epsilon_2$ so obtained if a setpoint change in output 2 causes too great an affect on output 1. Continue in this manner until all controls are active. Notice that no relative gain-type considerations are required, or relevant. This is because, from the point of view of the algorithm, all controls are always active. We have simply ''fooled'' the algorithm into calculating constant values for some control efforts.

The preceding tuning method is but one of many possible on-line tuning methods. It was presented solely to demonstrate the flexibility of the multivariable model state-feedback algorithm.

## Conclusions

We have presented a model state-feedback algorithm for multivariable processes with control effort constraints where the control objective is either (1) total decoupling for processes with no right half-plane zeros, or (2) total decoupling for all but one process variable when the process has right half-plane zeros that can be reflected into a single output. The algorithm is easy to implement, even for large numbers of controls and outputs, and yields excellent performance, as judged from the behavior of the example processes. It accommodates loss of controls and permits on-line tuning, one process variable at a time.

## Literature Cited


Berber, R., and C. B. Brosilow, ''Algorithm Internal Model Control,'' *Proc. Mediterranean Conf. on Control and Automation*, Haifa, Israel (1999).

Boyce, J., and C. Brosilow, ''Multivariable Cascade Control for Processes With Output Constraints,'' *Proc. Eur. Symp. on Computer Aided Process Engineering*-6 (*ESCAPE*-6), Rhodes, Greece (1996).

Campo, P. J., and M. Morari, ''Robust Control of Processes Subject to Saturation Nonlinearities,'' *Comput. Chem. Eng.*, **14**, (4/5), 343 (1990).

Coulibaly, E., S. Maiti, and C. Brosilow, ''Internal Model Predictive Control (IMPC),'' *Automatica*, **31**, 1471 (1995).

Cutler, C. R., and B. L. Ramker, ''Dynamic Matrix Control—A Computer Control Algorithm,'' AIChE Meeting, Houston, TX (1979).

Dong, J. W., and C. Brosilow, ''The Design, Tuning and Implementation of SISO and MIMO Two Degree of Freedom Control Systems,'' AIChE Meeting, Los Angeles, CA (1997).

Emoto, G., Y. Ota, S. Ebara, H. Matsuo, M. Ogawa, D. B. Raven, D. P. Golden, and J. S. Ayala, ''Improve Unit Profitability by Connecting Several Multivariable Controllers,'' AIChE Meeting, Houston, TX (1995).

Kailath, T., *Linear Systems*, Prentice Hall, Englewood Cliffs, NJ (1980).

Morari, M., and E. Zafiriou, *Robust Process Control*, Prentice Hall, Englewood Cliffs, NJ (1989).

Muske, K., and J. Rawlings, ''Model Predictive Control with Linear Models,'' *AIChE J.*, **39**(2), 262 (1993).

Prett, D. M., and M. Morari, *Shell Process Control Workshop*, Butterworth, Boston (1987).

Rosenbrock, H. H., ''Correction to 'The Zeros of a System,' '' *Int. J. Control*, **20**(3), 525 (1974).

Stryczek, K., and C. Brosilow, ''Mp Tuning of Multivariable Uncertain Processes,'' CPCV, Tahoe City, CA (1996).

Zheng, A. , M. Kothare, and M. Morari, ''Anti-Windup Design for Internal Model Control,'' *CIT-CDS Tech. Memo*, CalTech, Pasadena, CA (1993).

Zhou, K., and J. Doyle *Essentials of Robust Control*, Prentice Hall, Saddle River, NJ (1998).


## Appendix

In this Appendix we show that the controls computed by an IMC controller tuned as suggested in the fourth section will never exceed their lower (upper) bounds for arbitrary, and simultaneous, changes in all process variables. We first show that if the control effort of an SISO IMC control system is greater than zero for some positive setpoint change, then it is greater or equal to zero for any positive setpoint change. We next show that in an MIMO IMC control system we can adjust the IMC filter time constants one at a time to guarantee that no control effort goes below zero for any combination of setpoint changes.

Without loss of generality, we assume that the lower bounds of all control efforts are zero.

Consider a linear SISO process in a state-space representation:

$$\dot{x} = Ax + Bu$$

$$y = Cx.$$

The model state-feedback control effort is given as

$$u = -Kx + K_{sp}r \qquad r \geq 0,$$

where $x$ represents the process states, $u$ is the control effort, $y$ is the system output, and $r$ is the amount of a step setpoint change. Solving for the time domain control, $u(t)$, in the preceding system, gives

$$u(t) = \Big[\Big(-K(BK-A)^{-1}B+1\Big)K_{sp} + K\exp\big(-(BK-A)t\big)\Big((BK-A)^{-1}BK_{sp}\Big)\Big]r. \quad \text{(A1)}$$

Therefore if the minimum of $u$ is greater than or equal to zero for some arbitrary positive $r$, then the minimum is greater than or equal to zero for any positive $r$.

We now show that in decoupled multivariable systems, a setpoint change in process variable $i$ affects the control efforts only through filter constant $i$, and hence tuning of the filter constants can be done independent of one another.

Consider the multivariable process

$$G(s)=\begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \\ g_{n1} & \cdots & \cdots & g_{nn} \end{pmatrix}.$$

And the diagonal filter used is

$$F(s)=\begin{pmatrix} f_1 & 0 & \cdots & 0 \\ 0 & f_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & f_n \end{pmatrix}.$$

Let $l_{ii}$ represent the $ii$th term in the invertible part $G_{-}((s))^{-1}$. Then the controller is given as

$$Q(s)=\begin{pmatrix} l_{11}f_1 & l_{21}f_2 & \cdots & l_{n1}f_n \\ l_{21}f_1 & l_{22}f_2 & \cdots & l_{n2}f_n \\ \vdots & \vdots & \vdots & \vdots \\ l_{n1}f_1 & l_{n2}f_2 & \cdots & l_{nn}f_n \end{pmatrix}.$$

Now a setpoint change in the $i$th output picks up only the $i$th column, which is a function of filter $f_i$ only. Therefore, we can one by one adjust the response of each term to be equal to or greater than zero. That is,

$$L^{-1}\begin{pmatrix} l_{11}f_1 \\ l_{21}f_1 \\ \vdots \\ l_{n1}f_1 \end{pmatrix} r_1 \geq 0,\ L^{-1}\begin{pmatrix} l_{21}f_2 \\ l_{22}f_2 \\ \vdots \\ l_{n2}f_2 \end{pmatrix} r_2 \geq 0, \ldots, L^{-1}\begin{pmatrix} l_{n1}f_n \\ l_{n2}f_n \\ \vdots \\ l_{nn}f_n \end{pmatrix} r_n \geq 0,$$

where $L^{-1}$ indicates inverse Laplace transform. Therefore the control effort for any arbitrary step setpoint change is given by

$$u(t)=L^{-1}\begin{pmatrix} l_{11}f_1 & l_{21}f_2 & \cdots & l_{n1}f_n \\ l_{21}f_1 & l_{22}f_2 & \cdots & l_{n2}f_n \\ \vdots & \vdots & \vdots & \vdots \\ l_{n1}f_1 & l_{n2}f_2 & \cdots & l_{nn}f_n \end{pmatrix}\begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} \geq 0. \quad \text{(A2)}$$

*Manuscript received Mar. 17, 1999, and revision received Feb. 25, 2000.*