Chemical Engineering Communications
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/gcec20

QUADRATIC PROGRAMMING SOLUTION OF DYNAMIC MATRIX CONTROL (QDMC)

CARLOS E. GARCIA [a] & A.M. MORSHEDI [a]

[a] Shell Development Company, P.O. Box 1380, Houston, Texas, 77001
Version of record first published: 20 Jan 2011.

To cite this article: CARLOS E. GARCIA & A.M. MORSHEDI (1986): QUADRATIC PROGRAMMING SOLUTION OF DYNAMIC MATRIX CONTROL (QDMC), Chemical Engineering Communications, 46:1-3, 73-87

To link to this article: http://dx.doi.org/10.1080/00986448608911397

Chem. Eng. Commun. Vol. 46 pp. 073–087
0098-6445/86/4603-0073$02.00/0

QUADRATIC PROGRAMMING SOLUTION OF DYNAMIC MATRIX CONTROL (QDMC)

CARLOS E. GARCIA and A.M. MORSHEDI

Shell Development Company
P.O. Box 1380
Houston, Texas 77001

(Received June 18, 1985; in final form December 27, 1985)

QDMC is an improved version of Shell's Dynamic Matrix Control (DMC) multivariable algorithm which provides a direct and efficient method for handling process constraints. The algorithm utilizes a quadratic program to compute moves on process manipulated variables which keep controlled variables close to their targets while preventing violations of process constraints. Several on-line applications have demonstrated its excellent constraint handling properties, transparent tuning and robustness, while requiring minimal on-line computational load.
KEYWORDS Constraints Dynamic Matrix Control (DMC) Model-Predictive Control Multivariable Control Quadratic Programming (QP)

INTRODUCTION

Most process control applications consist of not only keeping controlled variables at their setpoints but also keeping the process from violating operating constraints. For more than a decade we at Shell have been implementing a multivariable computer control algorithm called Dynamic Matrix Control (DMC) with great success. The method calculates moves on manipulated variables which minimize future projections of controlled variable errors and constraint violations in the least-squares sense (Cutler and Ramaker, 1979, Prett and Gillette, 1979, Cutler, 1983).

Throughout the years, however, we have been increasingly encountering applications which demand tight constraint control. In addition, there has been an increasing need for improved on-line tuning capabilities for DMC.

An extended method for the solution of the DMC problem is introduced here. The method denoted as QDMC (Quadratic/Dynamic Matrix Control) consists of the on-line solution of a quadratic program (QP) which minimizes the sum of squared deviations of controlled variable projections from their setpoints subject to maintaining projections of constrained variables within bounds (Morshedi, *et al.* 1982). In contrast with DMC where constraints are enforced via least squares, the use of a QP provides rigorous handling of constraint violations by formulating them as linear inequalities, therefore allowing tighter constraint control.

The discussion that follows is divided into three parts. For completeness, an overview of DMC is presented first. Then the fundamentals of QDMC are introduced, concluding with a discussion of its implementation on a pyrolysis furnace temperature control problem.

DMC: AN OVERVIEW

DMC is a model predictive controller which exhibits excellent properties due to its particular structure. The technique was developed in Shell as part of its process computer control activities and has been analyzed in the control literature (Garcia and Morari, 1982). The method is rigorously derived for linear systems (as is any other conventional controller) and therefore, any analysis on DMC and its features must be done in a linear theory framework.

Linear Input–Output Model

Without loss of generality, let us consider a linear dynamic system with one output O and an input I. In computer applications only the behavior of the system at the sampling intervals is of interest. Therefore, a discrete representation of the dynamics is used here. One such representation is given by;

$$O(k+1) = \sum_{i=1}^{M} a_i \, \Delta I(k-i+1) + O_0 + d(k+1) \tag{1}$$

where k denotes discrete time; O_0 is the output initial condition; $\Delta I(k)$ is a change in input (or manipulated variable) at different time intervals k; $O(k)$ is the value of the controlled variable at time k; $d(k)$ accounts for un-modelled factors that affect $O(k)$; a_i are the unit step response coefficients of the system; and M is the number of time intervals required for the system to reach steady-state. Therefore, $a_i = a_M$ for $i \geq M$.

Note that the term $d(k+1)$ has been added to the input–output description to take into account unmodelled effects on the measured output, which consist of unmeasured disturbances and/or modelling errors. Inclusion of this factor is crucial to the derivation of DMC as we now show.

Controller Design

The objective of any controller is to find the moves of the manipulated variables, $\Delta I(k)$, which would make the output $O(k)$ best match a target value O_s in the face of disturbances. Assuming the present time interval to be $\bar{k}$, in DMC a projection of the output $O(k)$ over P future time intervals ($\bar{k}+1$ to $\bar{k}+P$) is matched to the setpoint O_s by prescribing a sequence of future moves.

Derivation of the DMC equations From (1) the projected output for any future time $\bar{k} + l$, $l > 0$ is:

$$
\begin{aligned}
O(\bar{k} + l) &= \sum_{i=1}^{l} a_i \, \Delta I(\bar{k} + l - i) && \text{(effect of future moves)} \\
&+ O_0 + \sum_{i=l+1}^{M} a_i \, \Delta I(\bar{k} + l - i) && \text{(effect of past moves)} \qquad (2) \\
&+ d(\bar{k} + l) && \text{(predicted disturbance)}
\end{aligned}
$$

For simplicity let us define

$$O^*(\bar{k} + l) = O_0 + \sum_{i=l+1}^{M} a_i \, \Delta I(\bar{k} + l - i) \qquad (3)$$

to be the contribution to $O(\bar{k} + l)$ due to past input moves up to the present time $\bar{k}$. This term can always be computed from the past history of moves.

Using this definition, one can write (2) for times $\bar{k} + 1$ up to $\bar{k} + P$ to produce a set of P equations for the output projections as follows:

$$\begin{bmatrix} O(\bar{k}+1) \\ \vdots \\ O(\bar{k}+P) \end{bmatrix} = \begin{bmatrix} O^*(\bar{k}+1) \\ \vdots \\ O^*(\bar{k}+P) \end{bmatrix} + \underset{=}{A} \begin{bmatrix} \Delta I(\bar{k}) \\ \vdots \\ \Delta I(\bar{k}+N-1) \end{bmatrix} + \begin{bmatrix} d(\bar{k}+1) \\ \vdots \\ d(\bar{k}+P) \end{bmatrix} \qquad (4)$$

where

$$\underset{=}{A} = \begin{bmatrix} a_1 & 0 & & \underset{=}{0} \\ a_2 & a_1 & & \\ \vdots & \vdots & \ddots & \\ a_N & a_{N-1} & & a_1 \\ \vdots & \vdots & & \vdots \\ a_M & a_{M-1} & & a_{M-N+1} \\ \vdots & \vdots & & \vdots \\ a_M & a_M & \cdots & a_M \end{bmatrix}$$

is called the "dynamic matrix" of the system. Note that in the DMC formulation only N moves are computed, i.e.

$$\Delta I(k) = 0 \quad \text{for} \quad k > \bar{k} + N$$

Setting these moves to zero imparts important stability properties to the resulting controller. In particular, as a result of our experience, selecting $P = N + M$ generally yields a stable controller. This is discussed below in the section on tuning parameter selection.

Estimation of unmodelled effects $d(k)$ in DMC The set of equations (4) requires a prediction of the unmodelled effects $d(k)$. Since future values of the "disturbance" $d(k)$ are not available, the best one can do is to use an estimate.

From Eq. (1) for $k = \bar{k} - 1$, and Eq. (3) for $l = 0$, we obtain

$$O(\bar{k}) = O^*(\bar{k}) + d(\bar{k}) \tag{5}$$

Therefore, $d(\bar{k})$ can be estimated using the current feedback measurement $O_m(\bar{k})$ of O together with past input moves information. In the absence of any additional knowledge of $d(k)$ over future intervals (as is true in most cases), the predicted disturbance is assumed to be equal to the present, "measured" $d(\bar{k})$;

$$\begin{aligned} d(\bar{k} + l) &= d(\bar{k}) \\ &= O_m(\bar{k}) - O^*(\bar{k}); \qquad l = 1, \ldots, P \end{aligned}$$

Solution of the DMC equations Given this set of equations the DMC control problem is defined as finding the N future input moves $\Delta I(\bar{k}) \cdots \Delta I(\bar{k} + N - 1)$ so that the sum of squared deviations between the projections $O(\bar{k} + l)$ and the target O_s are minimized. This is equivalent to the least-squares (LS) solution of the DMC equations:

$$\begin{bmatrix} O_s - O^*(\bar{k}+1) - d(\bar{k}) \\ \vdots \\ O_s - O^*(\bar{k}+P) - d(\bar{k}) \end{bmatrix} = \mathbf{e}(\bar{k}+1) = \underline{A}\mathbf{x}(\bar{k}) \tag{6}$$

where $\mathbf{e}(\bar{k} + 1)$ is a P-dimensional vector of projected deviations from the target and

$$\mathbf{x}(\bar{k}) = [\Delta I(\bar{k}) \cdots \Delta I(\bar{k} + N - 1)]^T$$

is the vector of future moves. Such least-squares solution is given by

$$\mathbf{x}(\bar{k}) = (\underline{A}^T\underline{A})^{-1}\underline{A}^T\mathbf{e}(\bar{k} + 1) \tag{7}$$

In DMC only the move computed for the current interval of time $\bar{k}$ is implemented. The computation is repeated at every sampling time k when a new feedback measurement is obtained and used to update $\mathbf{e}(\bar{k} + 1)$. Failure to compute a move at each sampling time could impair the disturbance handling features of the algorithm.

Formulation for multivariable systems It should be pointed out that the DMC equations for a multivariable system can be derived similarily as for the single-input single-output (SISO) case. For an r-output, s-input system, a linear dynamic representation is given by

$$\mathbf{O}(k+1) = \sum_{i=1}^{M} \underline{a}_i\, \Delta\mathbf{I}(k - i + 1) + \mathbf{O}_0 + \mathbf{d}(k + 1)$$

where $\mathbf{O}(k)$ is an r-dimensioned vector of outputs, $\underline{a}_i$ is an $r \times s$ matrix of unit step response coefficients for the i^{th} time interval, $\mathbf{\Delta I}(k)$ is the s-dimensioned vector of moves for all manipulated variables at a given time interval, $\mathbf{O}_0$ is the initial condition vector, and $\mathbf{d}(k)$ is a vector of un-modelled factors. For $r = s = 1$ this equation reduces to model (1).

We can define a multivariable system dynamic matrix $\underline{\underline{A}}$ composed of blocks of dimension $P \times N$ of step response coefficient matrices as in (4) relating the i^{th} output to the j^{th} input as follows:

$$\underline{\underline{A}} = \begin{bmatrix} \underline{\underline{A}}_{11} & \underline{\underline{A}}_{12} & \cdots & \underline{\underline{A}}_{1s} \\ \underline{\underline{A}}_{21} & \underline{\underline{A}}_{22} & \cdots & \underline{\underline{A}}_{2s} \\ \vdots & \vdots & & \vdots \\ \underline{\underline{A}}_{r1} & \underline{\underline{A}}_{r2} & \cdots & \underline{\underline{A}}_{rs} \end{bmatrix}$$

where elements from matrices $\underline{\underline{a}}_l$ have been regrouped accordingly. That is, matrix $\underline{\underline{A}}_{ij}$ contains all the ij coefficients in matrices $\underline{\underline{a}}_l$, $l = 1$ to M arranged as in (4).

The corresponding vector of moves is

$$\mathbf{x}(\bar{k}) = [\mathbf{x}_1(\bar{k})^T \mathbf{x}_2(\bar{k})^T \cdots \mathbf{x}_s(\bar{k})^T]^T$$

and the output projection vector becomes:

$$\mathbf{e}(\bar{k}+1) = [\mathbf{e}_1(\bar{k}+1)^T \mathbf{e}_2(\bar{k}+1)^T \cdots \mathbf{e}_r(\bar{k}+1)^T]^T$$

Therefore Eq. (7) is equally valid for multivariable systems.

Tuning Parameters in DMC

Number of moves (N) *vs. horizon* (P) It should be clear from the DMC formulation that as the number of manipulated variable moves (N) increases, DMC has more freedom in matching the output projections to the setpoint. That is, DMC produces tighter control although at the expense of larger moves or, as is well known, for processes with non-minimum phase characteristics the resulting controller could even be unstable (Garcia and Morari, 1982). From our experience, stability (in the case of a perfect model) is ensured in DMC by selecting P such that the steady-state effect of the most future move shows in the projections (see Eq. (4)); that is, $P = N + M$. Therefore, DMC is capable of handling non-minimum phase dynamic characteristics such as inverse response and dead-time. For a more detailed discussion on this subject, the reader is referred to Morshedi *et al.* (1982) and Cutler (1983). Also, a rigorous proof of stability for DMC when $P \gg N$ is given by Garcia and Morari (1982).

Move suppression (λ) In DMC it is usually necessary to restrict or suppress the amplitude of the input moves. Thus, DMC equations are generally formulated as

$$\begin{bmatrix} \mathbf{e}(\bar{k}+1) \\ \hdashline \mathbf{0} \end{bmatrix} = \begin{bmatrix} \underline{\underline{A}} \\ \hdashline \underline{\underline{\Lambda}} \end{bmatrix} \mathbf{x}(\bar{k}) \tag{8}$$

where (for multi-variable systems);

$$\underline{\underline{\Lambda}} = \operatorname{diag}(\underbrace{\lambda_1 \lambda_1 \cdots \lambda_1}_{|\leftarrow N \rightarrow|} \lambda_2 \lambda_2 \cdots \lambda_2 \cdots \lambda_s \lambda_s \cdots \lambda_s)$$

and $\lambda_i > 0$ is the i^{th} input move suppression coefficient. As in any other control formulation, an increase in input penalties is equivalent to reducing the controller gain and therefore, improves the stability properties of DMC, particularly in the face of model inaccuracies (Morari, 1983).

Selective weighting of controlled variables (γ) It is possible in DMC to give tighter control to particular controlled variable(s) by increasing the relative weight of the corresponding least-squares residual. This is achieved by pre-multiplying the DMC equations with the matrix of weights $\gamma_i > 0$;

$$\underset{=}{\Gamma} = \operatorname{diag}(\underbrace{\gamma_1\gamma_1 \cdots \gamma_1}_{|\leftarrow P \rightarrow|}\gamma_2\gamma_2 \cdots \gamma_2 \cdots \gamma_r\gamma_r \cdots \gamma_r).$$

Including this weighting matrix, the solution of the DMC equations becomes,

$$\mathbf{x}(\bar{k}) = (\underset{=}{A}^T\underset{=}{\Gamma}^T\underset{=}{\Gamma}\underset{=}{A} + \underset{=}{\Lambda}^T\underset{=}{\Lambda})^{-1}\underset{=}{A}^T\underset{=}{\Gamma}^T\underset{=}{\Gamma}\mathbf{e}(\bar{k}+1) \tag{9}$$

DMC also provides the user with additional tools for control such as feed-forward compensation of measurable disturbances, steady-state control of manipulated variables in case $s > r$, and minimization of constraint violations in the least squares sense. The new QDMC method introduced in the following improves the constraint handling capabilities of DMC thus making the algorithm a very powerful tool for solving complex multivariable constrained control problems.

QDMC: QP Solution of the DMC Equations

In on-line applications, the moves computed in (9) may not be implementable due to process operating limit violations. Three types of process constraints are usually encountered:

Manipulated variable constraints: i.e., valve saturation.

Controlled variable constraints: overshoots in the controlled variables past allowable limits must be avoided.

Associated variables: key process variables which are *not* directly controlled but that must be kept within bounds.

The controller must be able to predict future violations and prescribe moves that would keep these variables within bounds.

Constraints on projections of these variables can be expressed mathematically as a system of linear inequalities:

$$\underset{=}{C}\mathbf{x}(\bar{k}) \geq \mathbf{c}(\bar{k}+1). \tag{10}$$

The matrix $\underset{=}{C}$ contains dynamic information on the constraints and the vector $\mathbf{c}(\bar{k}+1)$ contains the projected deviations of the constrained variables from their limits. Also, in practice, limits on individual moves are usually needed:

$$\mathbf{x}_{\min} \leq \mathbf{x}(\bar{k}) \leq \mathbf{x}_{\max} \tag{11}$$

One can express the least-squares solution of the DMC equations as the following quadratic minimization problem:

$$\min_{\mathbf{x}(\bar{k})} S = \tfrac{1}{2}[\underline{A}\mathbf{x}(\bar{k}) - \mathbf{e}(\bar{k}+1)]^T \underline{\Gamma}^T \underline{\Gamma}[\underline{A}\mathbf{x}(\bar{k}) - \mathbf{e}(\bar{k}+1)] + \tfrac{1}{2}\mathbf{x}(\bar{k})^T \underline{\Lambda}^T \underline{\Lambda}\mathbf{x}(\bar{k}) \quad (12)$$

yielding (9) as a solution. Subjecting this problem to the linear inequality constraints (10) and (11), the following quadratic program (QP) results:

$$\begin{aligned} \min_{\mathbf{x}(\bar{k})} F &= \tfrac{1}{2}\mathbf{x}(\bar{k})^T \underline{H}\mathbf{x}(\bar{k}) - \mathbf{g}(\bar{k}+1)^T \mathbf{x}(\bar{k}) \\ \text{s.t. } & \underline{C}\mathbf{x}(\bar{k}) \geq \mathbf{c}(\bar{k}+1) \\ & \mathbf{x}_{\min} \leq \mathbf{x}(\bar{k}) \leq \mathbf{x}_{\max} \end{aligned} \quad (13)$$

where:

$$\underline{H} = \underline{A}^T \underline{\Gamma}^T \underline{\Gamma} \underline{A} + \underline{\Lambda}^T \underline{\Lambda} \quad \text{(the QP Hessian matrix)}$$

and,

$$\mathbf{g}(\bar{k}+1) = \underline{A}^T \underline{\Gamma}^T \underline{\Gamma}\mathbf{e}(\bar{k}+1) \quad \text{(the QP gradient vector).}$$

Solution of (13) by a QP algorithm at each sampling interval $\bar{k}$ produces an optimal set of moves $\mathbf{x}(\bar{k})$ which satisfies the constraints. Any commercially available QP algorithm could be used for solving (13). Since in QDMC $\underline{H}$ is likely to be fixed at all sampling intervals, a parametric QP algorithm is recommended to reduce on-line computation time (Bazaraa and Shetty, 1979, Fletcher, 1980).

QDMC Constraint Equations

The excellent performance of DMC hinges on its particular formulation as described above. Since constraint handling is nothing other than shifting the control priorities to constrained variables, it is crucial that the dynamic matrix formulation and structure are preserved in formulating the constraint equations. In the following we show how the inequalities in Eq. (10) are formulated for each of the constrained variables.

Manipulated variables The vector $\mathbf{x}(\bar{k})$ contains not only the present moves to be implemented but also predictions of the future moves. This gives an indication of where the manipulated variables will lie in the future.

One can bound the predicted level of the i^{th} input as follows:

$$I_{i\min} \leq I_i(\bar{k}) + \sum_{l=1}^{n} \Delta I_i(\bar{k}+l-1) \leq I_{i\max} \quad (14)$$

where $n = 1, \ldots, N$, that is, the total number of moves; $I_i(\bar{k})$ is the present value of the i^{th} manipulated variable; and $I_{i\min}$, $I_{i\max}$ are the lower and upper limits respectively. In matrix form, these constraints are expressed as:

$$\begin{bmatrix} -\underline{1}_L & & \\ & \ddots & \\ & & -\underline{1}_L \\ \underline{1}_L & & \\ & \ddots & \\ & & \underline{1}_L \end{bmatrix} \mathbf{x}(k) \geq \begin{bmatrix} (I_1(k) - I_{1\max}) & \mathbf{1} \\ \vdots & \\ (I_s(\bar{k}) - I_{s\max}) & \mathbf{1} \\ (I_{1\min} - I_1(\bar{k})) & \mathbf{1} \\ \vdots & \\ (I_{s\min} - I_s(\bar{k})) & \mathbf{1} \end{bmatrix} \quad (15)$$

where $\mathbf{1} = (1 \ 1 \cdots 1)^T$ and $\underline{1}_L$ is an $N \times N$ lower triangular matrix:

$$\underline{1}_L = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & & & & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

While only one equation (14) (for $n = 1$) suffices to prevent violations, constraining all the affected manipulated variable projections as in (15) provides improved performance.

Controlled variables Utilizing the concepts described above, the QP can be made to prescribe moves so that projections of the controlled variable responses lie within bounds. For example, for a single output system, with respective maximum and minimum limits $O_{\max}$, $O_{\min}$, the constraint equations are formulated as:

$$\begin{bmatrix} -\underline{A} \\ \underline{A} \end{bmatrix} \mathbf{x}(\bar{k}) \geq \begin{bmatrix} (O_s - O_{\max})\mathbf{1} - \mathbf{e}(\bar{k}+1) \\ (O_{\min} - O_s)\mathbf{1} + \mathbf{e}(\bar{k}+1) \end{bmatrix} \tag{16}$$

Note that the error projection vector defined in Eq. (6) is employed here. Extension to the multiple-output case is straightforward.

Associated variables As with controlled variables it is possible to have the QP keep projections of associated variables within limits. However, a new projection vector must be created. Analogous to (16) above, constraints on the projections of a single associated variable 'a' are expressed as:

$$\begin{bmatrix} -\underline{B} \\ \underline{B} \end{bmatrix} \mathbf{x}(\bar{k}) \geq \begin{bmatrix} \mathbf{e}_a(\bar{k}+1) - a_{\max}\mathbf{1} \\ a_{\min}\mathbf{1} - \mathbf{e}_a(\bar{k}+1) \end{bmatrix} \tag{17}$$

where:

$$\mathbf{e}_a(\bar{k}+1) = \begin{bmatrix} a^*(\bar{k}+1) + [a_m(\bar{k}) - a^*(\bar{k})] \\ \vdots \\ a^*(\bar{k}+P) + [a_m(\bar{k}) - a^*(\bar{k})] \end{bmatrix}$$

is the associated variable projection vector; $\underline{B}$ is the dynamic matrix for the associated variable; $a^*(k+1)$ is the effect of past inputs on the projection of a; $a_m(k)$ is the measured feedback and $a_{\min}$, $a_{\max}$ the constraint limits. Extension of (17) to handle multiple associated variables is straightforward.

Tuning of QDMC

All tuning parameters given for DMC still apply for the constrained case. However, in QDMC control quality is additionally influenced by the selection of the projection interval to be constrained. In practice, only a subset of all P projections are constrained in Eqs. (16) and (17), starting with the l^{th} projection, where $l > 1$. This subset of projections form a "constraint window" of future intervals of time over which QDMC will prevent constraint violations from occurring.

In the presence of non-minimum phase behavior of controlled and associated variables much improvement in performance is achieved by moving the "constraint window" further down in the horizon. The reason is that any projected violation inside the "constraint window" is handled rigorously by the QP, not unlike a tightly tuned controller. Therefore, if the QP is asked to correct for violations in the earlier projections, severe input moves might be required in the face of non-minimum phase characteristics.

Another solution to this problem consists in having QDMC solve the controlled and associated variable constraint equations in the least-squares sense. This is done by appending the constraint equations to the DMC equations (4) in case that a violation is predicted to occur. Then this augmented system of equations is solved as in (9). Both alternatives are available to the QDMC user.

IMPLEMENTATION

An implementation of the QDMC algorithm on a pyrolysis furnace is described. A diagram of the process is given in Figure 1. The temperature of the process gas stream through zones A, B and C of the furnace firebox is controlled by manipulating the fuel gas pressures to the burners. Cascaded controllers manipulate the fuel gas flows to meet the pressure targets. Feed rate and dilution steam rate are measurable disturbances which are fed forward to the control algorithm.

Unit step responses of temperatures in the three zones with respect to fuel pressures are given in Figures 2 through 4. The time unit is in 0.5 min which means that the process settles in 15 minutes. Additional step response models are needed in QDMC to model the effect of measurable disturbances.

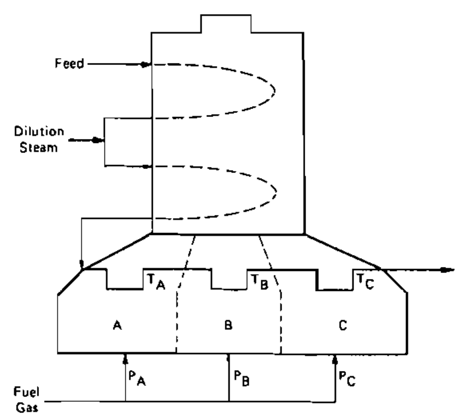

FIGURE 1 Pyrolysis furnace: fuel gas pressures P_A, P_B, P_C are manipulated to control temperatures in zones A, B and C.

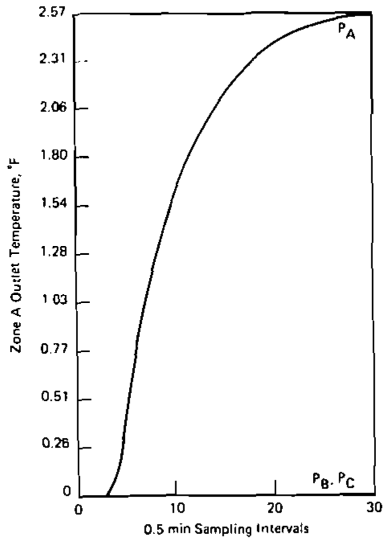

FIGURE 2 Response of zone *A* outlet temperature to unit step changes in zone fuel gas pressure setpoints (1°C = 1.8°F).

Problem Setup and Tuning

Besides imposing high and low limits on fuel gas pressures, bounds are also included for the temperatures. The constraint limits in deviation variables are given in Table I. No associated variables are considered.

Three moves in each manipulated variable are computed and checked for

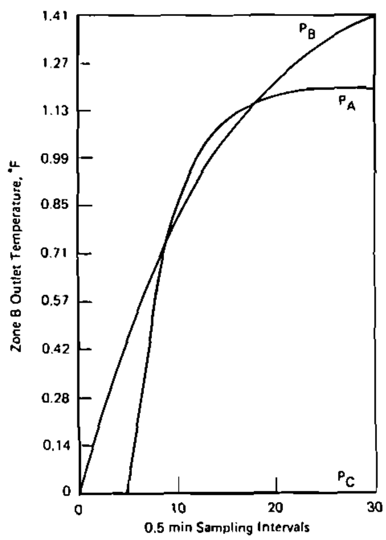

FIGURE 3 Response of zone *B* outlet temperature to unit step changes in zone fuel gas pressure setpoints.

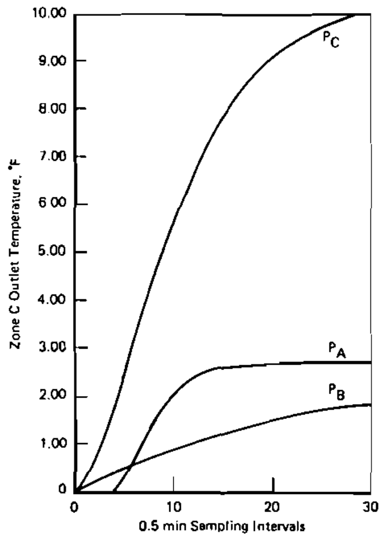

FIGURE 4 Response of zone C outlet temperature to unit step changes in zone fuel gas pressure setpoints.

violations. This yields a move vector of dimension 9. Choosing a horizon of 30, then matrix $\underline{\underline{A}}$ is of dimension 90×9.

A total of 10 projections in each controlled variable are checked for high and low limit violations, starting to check from the 5th, 3rd, and 3rd projection of temperatures A, B and C, respectively. Output weighting parameters are selected as $\gamma_i = 1$ and move suppression parameters λ_i as 15, 25 and 30 respectively for the three zones. It must be realized that the particular selection of λ_i is a function of the operator's allowed manipulated variable jaggedness and the actual model mismatch so that for large model errors, λ_i is increased. Therefore, λ_i must be tuned on-line during the operation as is done with any other control algorithm.

On-Line Responses

In order to test the input constraint handling capabilities of QDMC, simultaneous step changes in setpoints of T_B and T_C of magnitudes of +3°F and −3°F were

TABLE I

Constraint limits for the pyrolysis furnace problem in deviation variable units

	High limit	Low limit	Initial condition
T_A (°F)	1.0	−10.0	0.0
T_B (°F)	10.0	−10.0	0.0
T_C (°F)	4.0	−10.0	0.0
P_A (psi)	8.0	−14.0	5.0
P_B (psi)	8.0	−14.0	2.5
P_C (psi)	−19.0	−24.0	−23.0

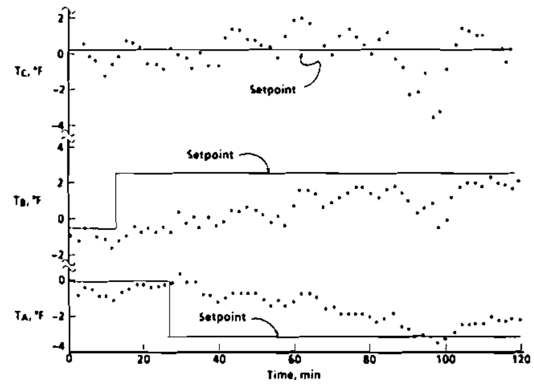

FIGURE 5a Temperature responses to setpoint changes in T_A, T_B; zone B fuel gas pressure setpoint is high limit constrained.

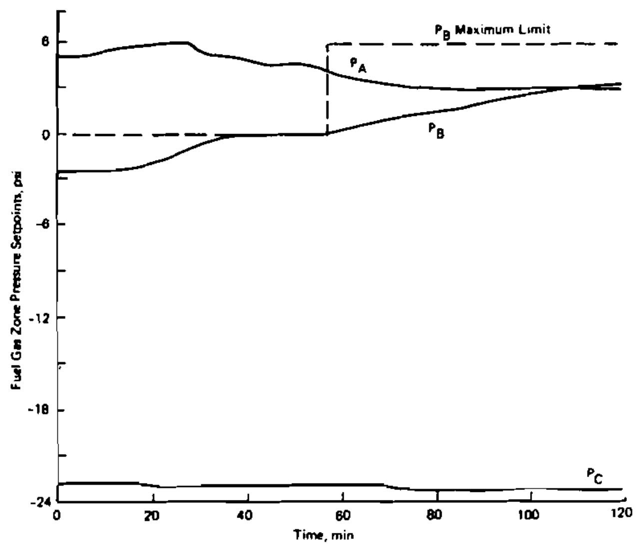

FIGURE 5b Fuel gas pressure setpoints corresponding to responses in Figure 5a (1 psi = 6.893 kPa).

implemented (Figure 5a). Note that QDMC maintains the zone B pressure at its bound until the limit is raised (Figure 5b). Only then were all the temperatures able to go to their respective setpoints. It is important to note that any other integral-action controller would have experienced reset wind-up under such input constraint conditions. Stability in the presence of constraints is a very important property of QDMC.

In another test, the setpoint in T_A was increased by 3°F. As shown in Figure 6 QDMC kept the temperatures from overshooting the bounds. Due to the severity of unmeasurable disturbances, temperatures were often driven out of bounds. The algorithm provided a smooth return to the operating region.

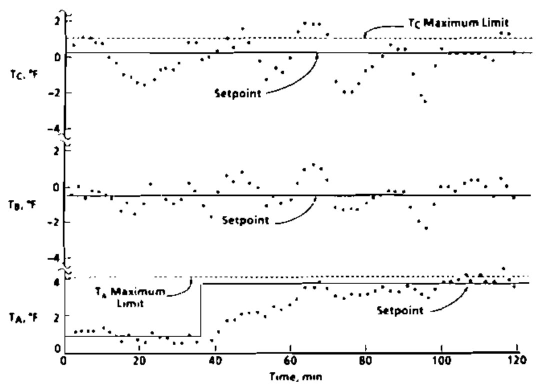

FIGURE 6 Temperature responses to setpoint change in T_A.

CONCLUSION

QDMC is a robust algorithm for the control of multivariable processing systems in the presence of constraints. Due to its predictive nature, it can handle systems with difficult dynamic characteristics to control, i.e., dead-time and inverse response processes. In addition, it contains very transparent tuning parameters of physical meaning to the user. In particular, certain parameter selections allow for stabilization of the controller in the presence of model mismatch.

Numerous applications of this algorithm within Shell have demonstrated its versatility in handling many types of process control problems encountered in the chemical process industries. It has been used to control batch as well as continuous systems, and processes with as many as 12 manipulated and controlled variables.

Above all, QDMC has proven itself particularly profitable in an on-line optimization environment. Due to constantly changing market conditions and feedstock quality the optimal operating point of a process lies invariably at an intersection of constraints. QDMC provides smooth, violation-free transfer of the operation from one set of constraints to another as dictated by an optimizer. In addition, its robustness characteristics guarantee reliability of the controller over the whole operating region.

NOMENCLATURE

a_i	unit step-response coefficient
$a(k)$	system associated variable at time k
$\underline{\underline{A}}$	dynamic matrix of controlled variable step response coefficients
$\underline{\underline{B}}$	dynamic matrix of associated variable step response coefficients
$\mathbf{c}(\bar{k}+1)$	vector of projected deviations of constrained variables from their bounds

$\underline{C}$	LHS matrix of QP linear inequalities
$d(\bar{k})$	QDMC un-modelled effects term
$\mathbf{e}(\bar{k}+1)$	controlled variable projected setpoint error vector
$\mathbf{e}_a(\bar{k}+1)$	associated variable projected response vector
$\mathbf{e}_i(\bar{k}+1)$	i^{th} controlled variable projected setpoint error vector
$\mathbf{g}(\bar{k}+1)$	QP gradient vector
$\underline{H}$	QP Hessian matrix
$I(\bar{k})$	system manipulated variable at time $\bar{k}$
$\Delta I(\bar{k})$	move of manipulated variable at time $\bar{k}$
k	discrete time
$\bar{k}$	present time
M	number of discrete time intervals required for steady-state
N	number of QDMC input moves
$O(\bar{k})$	system controlled variable at time $\bar{k}$
O_0	controlled variable dynamic model initial condition
O_s	controlled variable setpoint
P	QDMC projection horizon
r	number of manipulated variables
s	number of controlled variables
$\mathbf{x}(\bar{k})$	vector of present and future moves $\Delta I(\bar{k})$
$\mathbf{x}_i(\bar{k})$	i^{th} manipulated variable vector of moves
$\underline{\Gamma}$	matrix of controlled variable weights γ_i
$\underline{\Lambda}$	matrix of move suppression factors λ_i

Superscripts:

*	projection based on moves up to present time $\bar{k}$

Subscripts:

m	feedback measurement
max	maximum bound
min	minimum bound

REFERENCES

1. Bazaraa, M.S., and Shetty, C.M., *Nonlinear Programming Theory and Algorithms, Wiley, N.Y.*, (1979).
2. Cutler, C.R., "Dynamic Matrix Control: An Optimal Multivariable Control Algorithm with Consraints," Ph.D. Thesis, U. of Houston, Houston, TX., (1983).
3. Cutler, C.R., Haydel, J.J., and Morshedi, A.M., "An Industrial Perspective on Advanced Control," *AIChE Annual Meeting, Washington, D.C.*, November 1983.

4. Cutler, C.R., and Ramaker, B.L., "Dynamic Matrix Control—A Computer Control Algorithm," *AIChE 86th National Meeting, Houston, TX,* April 1979.
5. Fletcher, R., *Practical Methods of Optimization,* Vols. 1 and 2, Wiley, Chichester, (1980).
6. Garcia, C.E., and Morari, M., "Internal Model Control. 1. A Unifying Review and Some New Results," *I/EC Process Design and Development,* **21,** 308 (1982).
7. Morari, M., "Robust Stability of Systems with Integral Control," *IEEE Conf. on Decision and Control, San Antonio, TX,* December 1983.
8. Morshedi, A.M., Cutler, C.R., Fitzpatrick, T.J., and Skrovanek, T.A., "A Method of Controlling a Process Combining the Quadratic/Linear Programming Method of Optimization with Dynamic Matrix Control," U.S. Patent Applied for. (Applic. No. K-8838), 1982.
9. Prett, D.M., and Gillette, R.D., "Optimization and Constrained Multivariable Control of a Catalytic Cracking Unit," *AIChE 86th National Meeting, Houston, TX,* April 1979.