SPREADSHEET PROCEDURE FOR SIMULATING SETPOINT TRACKING IN SISO BY DYNAMIC MATRIX CONTROL

Larry K. Jang and Roger C. Lo
California State University • Long Beach, CA 90840-5103

The objective of this work is to present a spreadsheet tool that illustrates an ideal case of dynamic matrix control (DMC) calculations. The ideal case presented in this work is a hypothetical single-input-single-output DMC control system for setpoint tracking, in the absence of disturbance and mismatch between the measured and predicted process variable, at various move suppression coefficients. Students in an undergraduate process control class who followed the procedure to build the spreadsheet would gain a good basic understanding of DMC.

A BRIEF OUTLINE OF DYNAMIC MATRIX CONTROL METHOD

Dynamic matrix: prediction of future process response based on present and future control moves

Industry has widely accepted control strategies based on model predictive control (MPC), primarily in multiple-input-multiple-output processes. The first known industrial application of linear MPC was known as dynamic matrix control (DMC). Initially, DMC was introduced for the single-input-single-output (SISO) cases.[1] The development of DMC and its recent status has been reported in the literature.[2]

Moving horizon control is a key feature of dynamic matrix control. This type of controller employs a model internal to the controller architecture to predict the process variable for n_p steps into the future based on control moves for n_p steps in the past. The future process response predicted from the past control moves is called "free response."[3] Then, the algorithm calculates the predicted error (setpoint–free response, with model mismatch and disturbance taken into consideration) for future n_p steps, followed by an optimization scheme to calculate the present and future control moves needed to minimize or suppress the net error (predicted error–future

Larry K. Jang *is professor of chemical engineering at California State University, Long Beach. He earned his B.S. and M.S. from National Taiwan University and Ph.D. from University of Southern California, all in chemical engineering. His area of research in recent years is remote automatic control using LabVIEW technology.*

Roger C. Lo *is an assistant professor in the Department of Chemical Engineering at California State University, Long Beach. He received his Ph.D. in chemical engineering from Texas A&M University. Roger teaches undergraduate and graduate required courses (fluids, engineering mathematics, and transport phenomena) and also numerical analysis using Excel and MATLAB for chemical engineering calculations. Roger's research interest focuses on microfluidics and its applications at the interface of biology, chemistry, and engineering, such as microreactors, miniaturized high-throughput chemical/biological assays, and portable instruments for environmental analysis and monitoring.*

process response due to the present and future control moves). Once the computation is done, only the first calculated control move is implemented and the algorithm moves one time step ahead. The free response is predicted and new control moves are calculated again for this new time step. The procedure of predicting the future free response, calculating the present and future control moves to suppress error, and implementing the first calculated control move is carried out as time moves on.

The first crucial step in implementing DMC in a SISO case is to obtain information about how the process responds to a step change in the controller output. This procedure is very much like the traditional approach of finding the dynamic model of a process by making a step change in the controller output (in the manual mode) and observing the response of the process variable, followed by fitting the response curve to the model chosen (such as the first-order-plus-dead-time (FOPDT) model). With the model parameters (such as process gain, first-order time constant, and dead time), an engineer may use well-established tuning rules, such as Ziegler-Nichols method, Cohen-Coon method, and Internal Model Control (IMC) method, to find appropriate tuning parameters for a proportional-integral-derivative (PID) feedback controller. However, instead of fitting response data to a particular model, the DMC method calculates the coefficients of the system's step response model (SRM).[1]

Assuming that the system is initially at steady state and the initial value of the process variable is 0, the coefficient can be defined as follows:

$$\text{Coefficient of SRM} = a_i = \frac{y(t_i)}{\Delta u(t_0)} \tag{1}$$

where $\Delta u(t_0)$ is the magnitude of step change made (and held constant) in the controller output at $t = t_0$ and $y(t_i)$ is the response of the process variable at $t = t_i$. We may consider the t_i's as discrete times of choice and/or sampling times. In DMC applications, at least 10 discrete time steps are needed between t_0 and t_{ss} (time to reach the next steady state). The number of time steps taken to reach steady state is called *model horizon* (denoted as m in this work). If the controller output changes from the previous value by the magnitudes of $\Delta u(t_0)$, $\Delta u(t_1)$,..., $\Delta u(t_{nc-1})$, at present and future times ($t = t_0, t_1,.., t_{nc-1}$) and held constant thereafter, we may predict the response of the process at any future time t_i ($i = 1, 2,..., n_p$) by the principle of superposition. The effect of a control move made at t_0 would take t_i time steps to reach t_i. Likewise, the effect of a control move made at t_1 would take t_{i-1} steps to reach t_i, and so forth. Since the effect is additive, we have Eq. (2).

$$y(t_i) = a_i \Delta u(t_0) + a_{i-1} \Delta u(t_1) + \ldots (\text{for } n_c \text{ terms}), \tag{2}$$

Assuming that one is making 12 steps of prediction ahead (*i.e.*, the *prediction horizon* $n_p = 12$) based on the subsequent control moves for 5 steps (at t_0, t_1, t_2, t_3, and t_4) and held constant thereafter (*i.e.*, the control horizon $n_c = 5$), Eq. (2) may be rewritten in the matrix form [Eqs. (3) and (4)]:

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ y(t_3) \\ \cdot \\ \cdot \\ y(t_{12}) \end{bmatrix} = \begin{bmatrix} a_1 & 0 & 0 & 0 & 0 \\ a_2 & a_1 & 0 & 0 & 0 \\ a_3 & a_2 & a_1 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{11} & a_{10} & a_9 & a_8 & a_7 \\ a_{12} & a_{11} & a_{10} & a_9 & a_8 \end{bmatrix} \begin{bmatrix} \Delta u(t_0) \\ \Delta u(t_1) \\ \Delta u(t_2) \\ \Delta u(t_3) \\ \Delta u(t_4) \end{bmatrix} \tag{3}$$

$$[\boldsymbol{y}]_{12\times1} = [\boldsymbol{A}]_{12\times5}\,[\boldsymbol{\Delta u}\ (\boldsymbol{present\ and\ future})]_{5\times1} \tag{4}$$

The first term on the right-hand side of Eqs. (3) and (4) ($[A]_{12\times5}$) is called the dynamic matrix. Note that the present and future control moves $\Delta u(t_0)$, ..., $\Delta u(t_4)$ are not known *a priori*. We need to use the procedure below to calculate them.

FREE RESPONSE: PREDICTION FOR FUTURE PROCESS RESPONSE BASED ON PAST CONTROL MOVES

Free response is defined as the predicted process response for $t > t_0$ (at t_1, t_2,..., t_{np}) based on the past control moves ($\Delta u(t_{-np})$, $\Delta u(t_{-np+1})$,...., $\Delta u(t_{-1})$) *if no further controller outputs are changed* at t_0, t_1, t_2,.... The effect of the control move made at t_{-1} would take 2 steps to reach t_1, 3 steps to reach t_2, ..., and so forth. Likewise, the effect of the control move made at t_{-2} would take 3 steps to reach t_1, 4 steps to reach t_2,..., and so forth. Therefore, for a system in which $n_p = 12$, we would obtain the prediction vector $[y_p]_{12\times1}$ for the free response (due to past control moves only) as

$$\begin{bmatrix} y_p(t_1) \\ y_p(t_2) \\ \cdot \\ \cdot \\ y_p(t_{12}) \end{bmatrix} = \begin{bmatrix} y(t_{-12}) \\ y(t_{-12}) \\ \cdot \\ \cdot \\ y(t_{-12}) \end{bmatrix} + \begin{bmatrix} a_{13} & \cdot & \cdot & a_3 & a_2 \\ a_{14} & \cdot & \cdot & a_4 & a_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{24} & \cdot & \cdot & a_{14} & a_{13} \end{bmatrix} \begin{bmatrix} \Delta u(t_{-12}) \\ \Delta u(t_{-11}) \\ \cdot \\ \cdot \\ \Delta u(t_{-1}) \end{bmatrix} \tag{5}$$

or

$$\left[y_p(\text{future})\right]_{12\times1} = \left[y\left(t_{-n_p}\right)\right]_{12\times1} + \left[A_p\right]_{12\times12} \left[\Delta u(\text{past})\right]_{12\times1} \tag{6}$$

where

$$[\boldsymbol{A_p}]_{12\times12} = \begin{bmatrix} a_{13} & \cdot & \cdot & a_3 & a_2 \\ a_{14} & \cdot & \cdot & a_4 & a_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{24} & \cdot & \cdot & a_{14} & a_{13} \end{bmatrix}$$

where $y(t_{-np})$ is the value of the process variable at the beginning of prediction n_p steps in the past. For a system at initial steady state, the process variables at the beginning of prediction during moving horizon control $\{y(t_{-np}), y(t_{-np+1}),..., y(t_{-1})\}$ are all zero for $t < t_0$.

If the setpoint profile y_{sp} for $t > t_0$ is known, we may define the free-response error vector as

$$\begin{bmatrix} E(t_1) \\ E(t_2) \\ . \\ . \\ E(t_{12}) \end{bmatrix} = \begin{bmatrix} y_{sp}(t_1) \\ y_{sp}(t_2) \\ . \\ . \\ y_{sp}(t_{12}) \end{bmatrix} - \begin{bmatrix} y_p(t_1) \\ y_p(t_2) \\ . \\ . \\ y_p(t_{12}) \end{bmatrix} \quad (7)$$

or

$$[E]_{12\times1} = [y_{sp}]_{12\times1} - [y_p]_{12\times1} \quad (8)$$

To reduce or suppress the free-response error, DMC must instruct the final control element to make control moves for n_c steps at the present and future time steps ($t_0, t_1, t_2, ..., t_{nc-1}$).

Calculation of present and future control moves

If the controller makes control moves of $\Delta u(t_0)$, $\Delta u(t_1)$, $\Delta u(t_2)$, $\Delta u(t_3)$, and $\Delta u(t_4)$ for n_c steps (= 5 in this work), we may anticipate that such present and future control moves would cause the process to respond by the magnitudes given by Eqs. (3) and (4). Therefore, the free response error may be reduced due to the present and future control moves, resulting in an effective or net error expressed as a net error vector $[NE]_{12\times1}$:

Net Error = (Free response error due to past control moves $\Delta u(t < t_0)$) –

(Process response due to present and future control moves $\Delta u(t \geq t_0)$

$$\begin{bmatrix} NE(t_1) \\ NE(t_2) \\ . \\ . \\ NE(t_{12}) \end{bmatrix} = \begin{bmatrix} E(t_1) \\ E(t_2) \\ . \\ . \\ E(t_{12}) \end{bmatrix} - \begin{bmatrix} a_1 & 0 & . & . & 0 \\ a_2 & a_1 & 0 & . & 0 \\ a_3 & a_2 & a_1 & 0 & 0 \\ . & . & . & . & . \\ a_{11} & a_{10} & a_9 & a_8 & a_7 \\ a_{12} & a_{11} & a_{10} & a_9 & a_8 \end{bmatrix} \begin{bmatrix} \Delta u(t_0) \\ \Delta u(t_1) \\ \Delta u(t_2) \\ \Delta u(t_3) \\ \Delta u(t_4) \end{bmatrix} \quad (9)$$

$$[NE]_{12\times1} = [E]_{12\times1} - [A]_{12\times5}\,[\Delta u \text{ (present and future)}]_{5\times1} \quad (10)$$

It can be said that the present and future control moves would serve to reduce or suppress the free response error. In Eqs. (9) and (10), it is assumed that disturbance does not exist and that there is no mismatch between the measured and predicted process variable. It is also assumed that the controller does not reach saturation and there is no constraint on the control moves. To compute the present and future control moves, an objective function Φ is defined as

$$\Phi = \sum_{i=1}^{n_p} \left\{ \left[NE(t_i)\right]^2 + \lambda\left[\Delta u(t_i)\right]^2 \right\} \quad (11)$$

where λ is defined as the move suppression coefficient. Optimization of DMC gives the required present and future control moves that will serve to minimize Φ for a given λ[1,3]

Δu *(present and future)*$]_{5\times1}$ =

$$\begin{bmatrix} \Delta u(t_0) \\ \Delta u(t_1) \\ \Delta u(t_2) \\ \Delta u(t_3) \\ \Delta u(t_4) \end{bmatrix} = \left([\boldsymbol{A}^T]_{5\times12} [\boldsymbol{A}]_{12\times5} + \lambda [\boldsymbol{I}]_{5\times5} \right)^{-1} [\boldsymbol{A}^T]_{5\times12} [E]_{12\times1} \quad (12)$$

where $[A^T]$ is the transpose of the dynamic matrix [A] and [I] is a 5×5 identity matrix for $n_c = 5$. Apparently, the greater the move suppression coefficient λ, the smaller the $\Delta u(t_i)$ values or a more conservative control action; and vice versa the control action becomes more aggressive.

Once the calculation is done, only the first control move $\Delta u(t_0)$ is implemented and the horizon moves one time step further. By treating t_1 as the present time, t_0 is then one time step in the past, and so forth. In the next step of calculation at t_1, the previously calculated $\Delta u(t_0)$ would be considered as the control move one step in the past. So, its value (and only this Δu) should be entered at the bottom of the [Δu(past)] vector for the past control moves [second term on the right-hand side of Eqs. (5) and (6)]. Then [y_p], [E], [NE], and the new [Δu(present and future)] are recalculated at t_1. The whole procedure may be repeated for n_p steps.

If there is no mismatch between the measured and predicted process response, then the actual process response for future steps at t_1, t_2,... should be the sum of the free response and

Moving horizon control is a key feature of dynamic matrix control. This type of controller employs a model internal to the controller architecture to predict the process variable for n_p steps into the future based on control moves for n_p steps in the past.

the process response due to present and future moves [Eq. (13)]

$$y\left[(\text{actual response})\right]_{12\times1} = \left[y_p\right]_{12\times1} + [A]_{12\times5}\,[\Delta u(\text{present and future})]_{5\times1} \tag{13}$$

However, since only the first control move $\Delta u(t_0)$ is implemented at t_0, we may predict the actual process response of $y(t_1)$ only [Eq. (14)]

$$y(t_1) = y_p(t_1)\left(\text{predicted at } t_0 \text{ based on } y(t_{12}) \text{ and } \left[\Delta u(\text{past})\right]\right) + a_1 \Delta u(t_0) \tag{14}$$

As the control horizon moves forward, the whole calculation repeats.

As the control horizon moves forward, the whole calculation repeats. In a system where $a_1 = 0$ (due to dead time), the actual process response $y(t_1)$ is the same as the free response $y_p(t_1)$ predicted at t_0 due to past control moves.

TABLE 1
Coefficient of step response model (SRM) used in this work

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
0.00	0.948	1.297	1.425	1.473	1.490	1.496	1.499	1.499	1.500
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}	a_{30}
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500

SAMPLE CALCULATION BY SPREADSHEET PROCEDURE

Based on the MPC theory and DMC strategy outlined above, the authors developed a spreadsheet procedure using data for a hypothetical first-order-plus-dead-time model with process gain $K_p = 1.5$, time constant $\tau_p = 1$, and deadtime $\theta_p = 1$. The SRM coefficients for this process model are shown in Table 1 and Figure 1. The size of time step $\Delta t = 1$ is chosen in this work. This process stabilizes in 10 time steps or model horizon m = 10. The established DMC guidelines recommend that the pre-

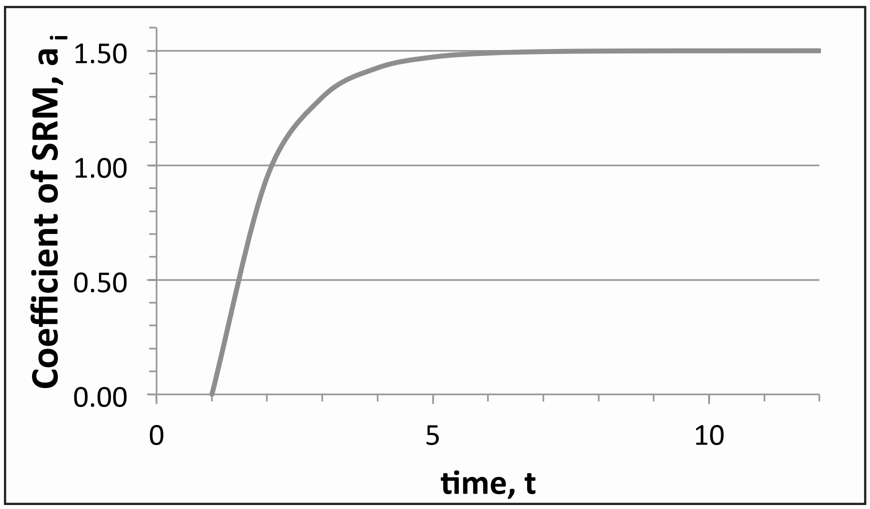

***Figure 1.** Plot of SRM Coefficient a_i versus time.*

TABLE 2
Dynamic matrix $[A]_{12x5}$ for prediction horizon $n_p = 12$ and control horizon $n_c = 5$ using the coefficients a_i (i = 1, 2, .., 12) from the SRM.

0.000	0.000	0.000	0.000	0.000
0.948	0.000	0.000	0.000	0.000
1.297	0.948	0.000	0.000	0.000
1.425	1.297	0.948	0.000	0.000
1.473	1.425	1.297	0.948	0.000
1.490	1.473	1.425	1.297	0.948
1.496	1.490	1.473	1.425	1.297
1.499	1.496	1.490	1.473	1.425
1.499	1.499	1.496	1.490	1.473
1.500	1.499	1.499	1.496	1.490
1.500	1.500	1.499	1.499	1.496
1.500	1.500	1.500	1.499	1.499

TABLE 3
Matrix $([A^T][A]+\lambda[I])^{-1}[A^T]$ for move suppression coefficient $\lambda = 3.0$

0.000	0.160	0.127	0.071	0.033	0.012	0.004	0.002	0.001	0.000	0.000	0.000
0.000	-0.091	0.087	0.085	0.050	0.020	0.009	0.005	0.003	0.003	0.003	0.003
0.000	-0.044	-0.127	0.065	0.070	0.031	0.017	0.012	0.010	0.010	0.009	0.009
0.000	-0.018	-0.059	-0.140	0.046	0.033	0.028	0.026	0.026	0.026	0.025	0.025
0.000	-0.007	-0.026	-0.073	-0.172	-0.026	0.028	0.048	0.055	0.058	0.059	0.059

diction horizon n_p exceeds the model horizon by 20% or 50% (n_p ~ 12-15) and that the control horizon n_c = 0.5 m ~ 5.[1] Since the purpose of this module is to illustrate the concept of DMC without using Visual Basic (or other sophisticated software), the authors choose n_p = 12 and n_c =5 for illustrating the procedure.

With n_p = 12 and n_c =5, the dynamic matrix $[A]_{12\times5}$ based on the SRM coefficients (Table 1) is given in Table 2. By following the standard matrix operations in Excel for transpose, multiplication, and addition, the matrix $([A^T][A]+\lambda[I])^{-1}[A^T]$ for $\lambda = 3.0$ is calculated and summarized in Table 3. The matrix $[A_p]_{12\times12}$ (for predicting the free response of the process) is given in Table 4. Note that the elements of $[A_p]_{12\times12}$ should be entered according to Eq. (6), where the values of the elements are from Table 1.

To initialize the calculation, it is assumed that the system is at steady state at t_0 and that the process variable and the past control moves are both zero for $t < t_0$ for at least n_p steps in the past. The spreadsheet shown in Table 5 (following page) illustrates the calculation for the time step at t_0. The following procedure was used to create Table 5:

(1) *Column G is created to count the past and present time steps -12, -11,..., -1, 0. Column H is created to count the present and future time steps 0, 1, 2,..., 12;*

(2) *Assign 0.00 values to Q65-Q77 for present and past values of process variable;*

(3) *Highlight cell I65 and perform "=$Q65 " (the beginning process variable (t_{-12}) for prediction of free response). Highlight cell I66, perform "= I65," and copy downward. In this manner, elements in Column I have the $y(t_{-12})$ value;*

(4) *Assign 0.0 to J65-J76 (for past control moves [Δu(past)];*

(5) *Highlight cell K-65-K76 and perform matrix multiplication $[A_p]$ [Δu (past)] (=MMULT($E$37:$P$48,J65:J76), followed by the combined key stroke,* control+shift+enter, *in Excel, where $[A_p]$ is located in the cells E37-P48 and [Δu(past)] is located in the cells J65-J76. The $ sign is needed to fix the column number and the row number for $[A_p]$ when copying the content of Table 5 down below. However, the cell positions for [Δu(past)] do not need the $ sign in front of the column number or the row number; Excel will automatically reference to the appropriate cell positions when the content of Table 5 is copied downward;*

(6) *Add Column I and K and put the results in L65-L76, that gives (future) free response of process variable due to past control moves;*

(7) *Assign setpoint of 1.00 to M65-M76, assuming that there is a unit step change of 1.000 occurring at t_0^+ and calculate the free response error [E] (Column N) by subtracting Column L from Column M;*

TABLE 4
Prediction matrix $[A_p]_{12x12}$ for future free response of process variable $[y_p]$ ($t > t_0$) due to changes in past controller outputs ($t < t_0$)

1.500	1.500	1.500	1.500	1.499	1.499	1.496	1.490	1.473	1.425	1.297	0.948
1.500	1.500	1.500	1.500	1.500	1.499	1.499	1.496	1.490	1.473	1.425	1.297
1.500	1.500	1.500	1.500	1.500	1.500	1.499	1.499	1.496	1.490	1.473	1.425
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.499	1.499	1.496	1.490	1.473
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.499	1.499	1.496	1.490
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.499	1.499	1.496
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.499	1.499
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.499
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500
1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500	1.500

TABLE 5

Calculation of [Δu (present and future)]$_{5x1}$ to be carried out at $t_0, t_1,..,t_4$ based on free response of process variable y_p (t_1, t_2, ..., t_{12}) and past Δu ($t_{-12}, t_{-11},...t_{-1}$). Note that only the first of the present and future Δu (0.410) is actually carried out at t_0.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
62									y @ t()									
63				t	0		count, now & past	count, now & future	-12	Δu(past)	[A_p] [Δu(past)]	y_p (free response)	y_sp	[E]	Δu(now & future)	y(next)	y(now & past)	
64					`=I65 & copied downwards			0	`=$Q65			`=I65+K65		`=M65-L65	0.410	`=L65+O64*$E$7		
65							-12	1	0.000	0.000	0.000	0.000	1.000	1.000	0.176	0.000	0.000	
66							-11	2	0.000	0.000	0.000	0.000	1.000	1.000	0.064		0.000	
67							-10	3	0.000	0.000	0.000	0.000	1.000	1.000	0.018		0.000	
68							-9	4	0.000	0.000	0.000	0.000	1.000	1.000	0.001		0.000	
69							-8	5	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
70							-7	6	0.000	0.000	0.000	0.000	1.000	1.000	`=MMULT($E$28:$P$32, N65:N76)		0.000	
71							-6	7	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
72							-5	8	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
73							-4	9	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
74							-3	10	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
75							-2	11	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
76							-1	12	0.000	0.000	0.000	0.000	1.000	1.000			0.000	
77							0			0.410							0.000	
78										`=O64	`=MMULT($E$37:$P$48,J65:J76)							

TABLE 6

Calculation of Δu to be carried out at $t_1, t_2,.., t_5$ based on predicted process variable y_p (t_2, t_3, ..., t_{13}) and past Δu ($t_{-11}, t_{-8},...t_0$). Only the first of the present and future Δu (0.176) is actually carried out at t_1. Important cell operations are explained in the text.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
79								`=I63+1	y @ t()									
80				t	1		count, now & past	count, now & future	-11	Δu(past)	[A_p] [Δu(past)]	y_p (free response)	y_sp	[E]	Δu(now & future)	y(next)	y(now & past)	
81					`=I63+1			1	`=$Q82						0.176			
82							-11	2	0.000	0.000	0.389	0.389	1.000	0.611	0.064	0.389	0.000	
83					`=G65+1		-10	3	0.000	0.000	0.532	0.532	1.000	0.468	0.018		0.000	
84							-9	4	0.000	0.000	0.585	0.585	1.000	0.415	0.002		0.000	
85							-8	5	0.000	0.000	0.604	0.604	1.000	0.396	-0.002		0.000	
86					`=I82		-7	6	0.000	0.000	0.611	0.611	1.000	0.389			0.000	
87							-6	7	0.000	0.000	0.614	0.614	1.000	0.386			0.000	
88							-5	8	0.000	0.000	0.615	0.615	1.000	0.385			0.000	
89							-4	9	0.000	0.000	0.615	0.615	1.000	0.385			0.000	
90							-3	10	0.000	0.000	0.615	0.615	1.000	0.385			0.000	
91							-2	11	0.000	0.000	0.615	0.615	1.000	0.385			0.000	
92							-1	12	0.000	0.000	0.615	0.615	1.000	0.385		=Q77	0.000	
93							0	13	0.000	0.410	0.615	0.615	1.000	0.385		and copied	0.000	
94							1			0.176						upwards	0.000	
95										`=O81	`=J77 (and copied upwards)					`=P65		

(8) Highlight cell O64-O68 and perform matrix multiplication according to Eq. (12), where the($[A^T][A]+ \lambda\ [I])^{-1}\ [A^T]$(for λ=3.0) is given in Table 3. Again, in the matrix multiplication "=MMULT($E$28:$P$32,N65:N76)," E28-P32 is the matrix $[A^T]\ [A]+ \lambda\ [\ I])^{-1}\ [A^T]$ and its column number and row number must be fixed by placing the $ sign in front of them. However, the cell positions in the [E] vector do not need the $ sign. The cells O64-O68 are the predicted control moves to be made at t_0, t_1, ..,t_4. But only the first one (O64, with calculated value of 0.410) will be actually implemented;

(9) Highlight cell J77 and assign " = O64." In this way, the present $\Delta u\ (t_0)$ is entered to the bottom of Column J;

(10) Highlight cell P65 to calculate the actual process variable y(next) or $y(t_1)$ according to Eq. (14) to show the effect of the present control move $\Delta u(t_0)$. In the authors' spreadsheet, the coefficient of SRM a_1 is located at cell E7.

To perform the calculation for the moving horizon control, the cells in Table 5 are copied and pasted somewhere else on the spreadsheet for calculations at the next time step at t_1. The pasted cell needs revision before the final outcome (Table 6) is obtained:

(1) Highlight the cell G82 and perform "= G65 +1" and copy downward. The count for now and past will be advanced by 1 from corresponding cells of Table 5. Do the same for H81-H93 and cell I80;

(2) Highlight Q94 and perform "= P65." Doing so will put the y(next) calculated at t_0 as the value of the process valuable y(now) at t_1;

(3) Highlight Q93, perform "= Q77," and copy the formula upward. In this manner, the process vari-

TABLE 7

Calculation of [Δu(present and future)] to be carried out at $t_2, t_3, .., t_6$ based on predicted process variable y_p (t_3, t_4, ..., t_{14}) and past Δu ($t_{-10}, t_{-9}, ...t_1$). This section is the result of copying the cells in Table 6 and pasting it somewhere below.

	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
96						y @ t									
97	t	2.000		count, now & past	count, now & future	-10.000	Δu(past)	$[A_p]$ [Δu(past)]	y_p (free response)	y_{sp}	[E]	Δu(now & future)	y(next)	now & past)	
98					2							0.064			
99				-10	3	0.000	0.000	0.699	0.699	1.000	0.301	0.018	0.699	0.000	
100				-9	4	0.000	0.000	0.813	0.813	1.000	0.187	0.003		0.000	
101				-8	5	0.000	0.000	0.855	0.855	1.000	0.145	-0.002		0.000	
102				-7	6	0.000	0.000	0.870	0.870	1.000	0.130	-0.002		0.000	
103				-6	7	0.000	0.000	0.876	0.876	1.000	0.124			0.000	
104				-5	8	0.000	0.000	0.878	0.878	1.000	0.122			0.000	
105				-4	9	0.000	0.000	0.879	0.879	1.000	0.121			0.000	
106				-3	10	0.000	0.000	0.879	0.879	1.000	0.121			0.000	
107				-2	11	0.000	0.000	0.879	0.879	1.000	0.121			0.000	
108				-1	12	0.000	0.000	0.879	0.879	1.000	0.121			0.000	
109				0	13	0.000	0.410	0.879	0.879	1.000	0.121			0.000	
110				1	14	0.000	0.176	0.879	0.879	1.000	0.121			0.000	
111				2			0.064							0.389	
112															
113						y @ t									
				count,	count,							Δu(now			

TABLE 8

Results for t = 12. Additional column S267-S281 displays the cumulative controller output u.

	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
263				10	23	0.000	0.000	1.000	1.000	1.000	0.000			1.001		
264				11			0.000							1.000		
265																
266						y @ t										
267	t	12.000		count, now & past	count, now & future	0.000	Δu(past)	$[A_p]$ [Δu(past)]	y_p (free response)	y_{sp}	[E]	Δu(now & future)	y(next)	now & past)		u
268					12							0.000			'=S268+J269	
269				0	13	0.000	0.410	1.000	1.000	1.000	0.000	0.000	1.000	0.000	and copied	0.410
270				1	14	0.000	0.176	1.000	1.000	1.000	0.000	0.000		0.000	downwards	0.586
271				2	15	0.000	0.064	1.000	1.000	1.000	0.000	0.000		0.389		0.650
272				3	16	0.000	0.018	1.000	1.000	1.000	0.000	0.000		0.699		0.668
273				4	17	0.000	0.003	1.000	1.000	1.000	0.000			0.874		0.671
274				5	18	0.000	-0.001	1.000	1.000	1.000	0.000			0.955		0.670
275				6	19	0.000	-0.001	1.000	1.000	1.000	0.000			0.988		0.668
276				7	20	0.000	-0.001	1.000	1.000	1.000	0.000			0.999		0.667
277				8	21	0.000	0.000	1.000	1.000	1.000	0.000			1.001		0.667
278				9	22	0.000	0.000	1.000	1.000	1.000	0.000			1.001		0.667
279				10	23	0.000	0.000	1.000	1.000	1.000	0.000			1.001		0.667
280				11	24	0.000	0.000	1.000	1.000	1.000	0.000			1.000		0.667
281				12			0.000							1.000		0.667
282																
283																

able calculated in the time step before will move up one position;

(4) *Highlight J93, perform "= J77" (the bottom of J column in Table 5), and copy the formula upward. In this manner, the Δu(past) from the previous table will move up by one position;*

(5) *The users will then see the revised result of cells O81-O85. Finally, Highlight J94 and perform "= O81" to enter the present control move just calculated.*

Once Table 5 and Table 6 are established, the content of Table 6 can be copied and pasted somewhere down below without any further revisions because the relationships between two successive tables are already established. The result of calculations at t_2 is given in Table 7. If the contents of Table 6 are pasted for subsequent time steps, the final results of calculations for t_{12} are shown in Table 8. In Table 8, the cell S269 is highlighted and "= S268+J269" is performed and copied downward. In this way, the cumulative controller output may be calculated from the control moves since t_0 (Column J).

Figure 2 shows the process response (y), control moves at every time step (Δu), cumulative controller output (u) and setpoint ($y_{setpoint}$) for time steps up to $n_p = 12$ (Table 8). Once the calculations are done for $\lambda = 3$, the contents of this worksheet may be copied and pasted on several other worksheets. By simply changing the cell that stores the λ value, the results for $\lambda = 1, 5, 8$, and 20 would appear immediately. Figures 3 and 4 summarize the effects of moving suppression coefficient λ on the process variable y as well as cumulative controller output u. It is clear that the greater the λ value, the more conservative the control action and the more sluggish in process response.

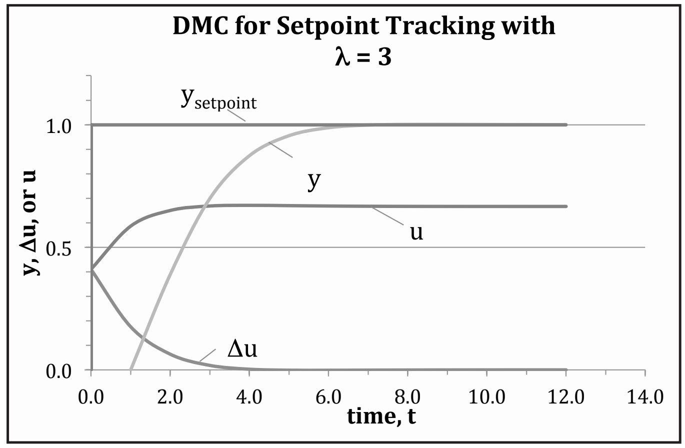

***Figure 2.** Results of calculation for setpoint tracking by DMC with move suppression coefficient λ =3. Since $a_1 = 0$ in this case, $y = y_p$.*

DISCUSSION

This work focuses on setpoint tracking in the absence of disturbance for perfectly linear process models. In reality, many real-world processes exhibit non-linear behaviors, with SRM coefficients (a_i's) varying with steady-state conditions. Also, it is assumed that there is no process or model mismatch between the predicted and measured values of process variable at t_0. If such a mismatch exists, the magnitude $\varepsilon = y_{sp}(t_0) - y_p(t_0)$ must be subtracted from every element of the vector [E] in Eqs. (7) and (8) to make the prediction more accurate.

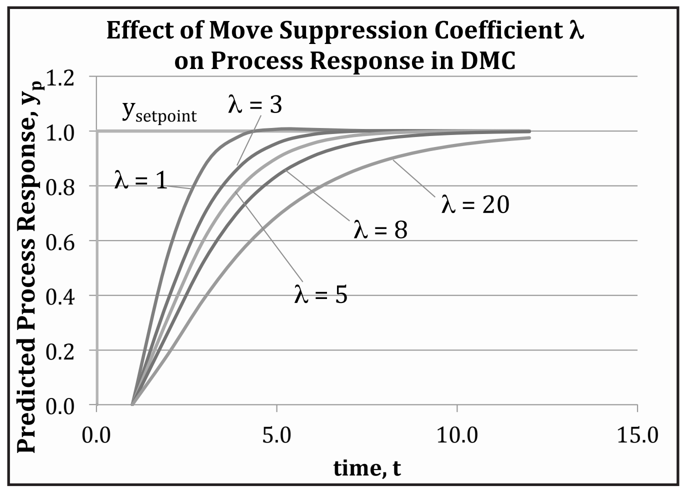

***Figure 3.** Effect of move suppression coefficient λ on the actual process response y or free response y_p, same for this case with $a_1 = 0$.*

With the model having the first-order-plus-dead-time behavior, there are many well-established tuning rules available to tune the feedback controller for this process. However, DMC offers several benefits. DMC is able to predict the future errors and the present and future control moves in order to minimize predicted net errors for future steps. This ability of changing controller outputs based on the predicted future process condition is absent in traditional feedback control. Or, we may say that a DMC system having a very short prediction horizon (n_p) behaves more like a traditional feedback controller. In this work, the authors choose n_p, m, and n_c according to recommended guidelines. Because the control action cannot change after the control horizon ends, a short control horizon n_c results in a large initial controller output, but a few careful changes in control action. The large control action in a short control horizon might overshoot the setpoint. However, as the controller continues to execute, the process variable will eventually settle around setpoint. On the other hand, a long control horizon produces a small initial controller output but more *aggressive changes* in control action. These aggressive changes in control action in a long control horizon can result in oscillation. In general, a control horizon of about one-half model horizon is recommended. Although a long prediction horizon increases the predictive ability of the DMC, the performance of the controller may suffer due to the extra calculations required.

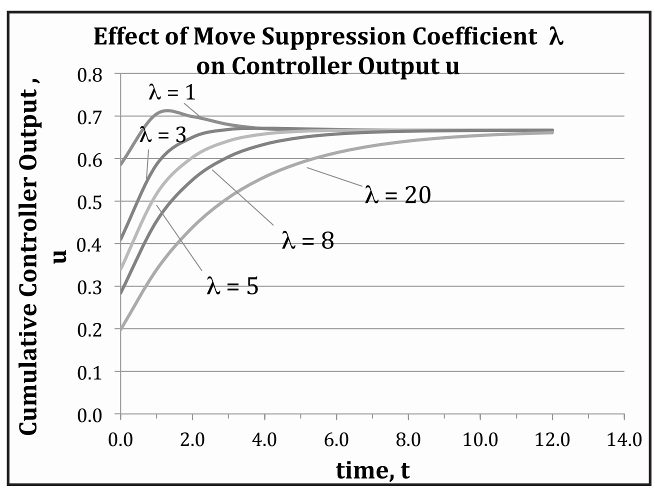

***Figure 4.** Effect of move suppression coefficient λ on the controller output u.*

The spreadsheet procedure developed here allows students to track step-by-step changes of a moving horizon control system with great ease. Interested users may revise this spreadsheet by

(1) using SRM coefficients obtained from real-world

processes or real plant data;

(2) removing or changing the amount of dead time to explore the effects of dead time on y and u. In the absence of dead time, the free response y_p(next) and actual response y(next) will differ by a magnitude of a_1^* Δu(present), where $a_1 \neq 0$;

(3) consulting with publications that use more sophisticated software[4] once they are confident in the basic concept; or

(4) exploring the effects of using different values for various horizons (model horizon, prediction horizon, and control horizon).

The topic of DMC was covered in our department's process control class for the first time in the Fall 2014 semester, as suggested by our department's industrial associates. The topics covered in this class are listed below:

- *Introduction to the course, control practice, elements of typical control systems, design of data acquisition using modern software such as LabVIEW*
- *Case study of process dynamic model: heated stirred tank—from energy balance to closed-loop control system, and case studies including laboratory control experiments*
- *Derivation of dynamic models for industrially important process units*
- *Review of essential mathematical tools (differential equations and Laplace Transform)*
- *Dynamic behavior of first-order systems (first experience of Loop Pro software); Dynamic Matrix*
- *Dynamic behavior of second-order systems*
- *Higher order systems and approximate models for poorly understood systems*
- *Overall transfer function of closed-loop feedback control systems*
- *Stability analysis*
- *Process identification of typical chemical engineering processes*
- *Industrial procedure of controller tuning (process reaction curve method, continuous cyclic response method, frequency response method)*
- *Industrial procedure of controller tuning (Internal Model Control (IMC))*
- *Advanced Control Methods—Cascade, Feedforward/Feedback, Smith Predictor, Gain Scheduling*
- <u>*Introduction to Dynamic Matrix Control*</u>

Besides two hours of lecture each week, this course has a Laboratory/Problem (L/P) component that meets three hours each week. During the L/P hours, students have opportunities running real-world experiments, performing simulation using Loop-Pro software (Workshop 1 through 3 below), and designing spreadsheet procedures (Workshop 4):

- ***Workshop #1**: Level control in gravity drained tanks*
- ***Workshop #2**: Temperature control in shell-n-Tube heat exchangers*
- ***Workshop #3**: Level control in pumped tanks*
- <u>***Workshop #4**: Spreadsheet procedure for Dynamic Matrix Control calculations*</u>

Although the topic of DMC and the associated spreadsheet workshop were covered toward the end of the semester for one lecture and one L/P activity, students had a chance to learn the basics of this advanced concept.

CONCLUSIONS

1. An easy-to-follow spreadsheet procedure is developed for an undergraduate process control class to learn basic aspects of moving horizon control in a dynamic matrix controller (DMC);

2. Based on coefficients of step response model (SRM) obtained from plant data, a dynamic matrix can be constructed for a DMC;

3. The DMC calculates the present and future control moves required to decrease the predicted future errors (and steer the process variable toward setpoint) according to the rules of DMC and a selected value of move suppression coefficient λ; and

4. The greater (smaller) the λ value, the more conservative (aggressive) the control action.

ACKNOWLEDGMENT

This work is partially supported by Faculty Sabbatical Leave Award (Jang) and Research, Scholarship, and Creative Activity Award (Lo) from California State University, Long Beach.

REFERENCES

1. Riggs, J.B., and M.N. Karim, *Chemical and Bio-Process Control*, 3rd Ed. Lubbock, Tex.: Ferret Pub, 2007
2. Lee, J.H., "Model Predictive Control: Review of the Three Decades of Development," *Int. J. Control Autom. Syst.*, **9**(3), 415, (2011)
3. Camacho, E., and C. Bordons, *Model Predictive Control*, 1st Ed. Berlin; New York: Springer, 1999
4. Wang, L., *Model Predictive Control System Design and Implementation Using MATLAB®*, Springer Science & Business Media, 2009 ❒