MPC Advanced Topics

MPC Using Orthogonal Basis Functions

- The fundamental technique in discrete-time MPC is the generation of an optimal future control trajectory
  - Such trajectories are assumed to have finite length (N_c)
  - Standard MPC algorithms assume control increments $\triangle u[k]$ beyond N_c are equal to zero
  - Clearly, this is not always a correct assumption, though far-future control signals can be expected to be small in magnitude
  - But use of long control horizons increase the computational complexity of the algorithm
- This chapter presents an alternative MPC formulation which addresses both of these concerns
- The next chapter will present another approach (dual-mode) based on stabilizing state-feedback control

Orthonormal Basis Functions

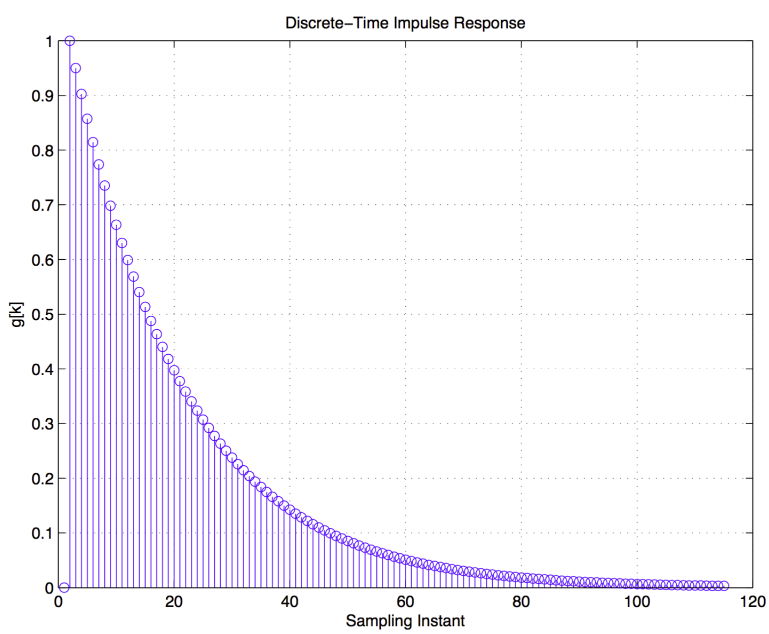
- The general idea of decomposing a signal in terms of basis functions is widely applied in systems theory

- For example, we normally represent a stable discrete-time system $G(z)$ in the form of its Laurent expansion

$$G(z) = \sum_{k=1}^{\infty} g_k z^{-k}$$

- Here, the functions $\{z^{-k}\}$ form an *orthonormal basis set* for the space of functions represented by $G(z)$
  - An orthonormal basis means that inner products among different basis functions within the set equals zero; with the same basis function equals one
  - The coefficients g_k in the above case are known as the *Markov parameters* of the system and comprise its discrete-time pulse response

- One immediate problem that arises with the Laurent expansion is the requirement for a large number of parameter to describe a slowly decaying time response, as shown below



- This difficulty suggests we seek an alternative basis set that hopefully requires fewer parameters in the description
- In particular, we consider expansions of the general form

$$g(z) = \sum_{k=1}^{\infty} c_k f_k(z)$$

where the functions $f_k(z)$ are general orthonormal basis functions and the c_k are the corresponding expansion coefficients

- Well-known basis functions include single-pole Laguerre and two-parameter Kautz functions; we shall focus on the Laguerre functions here

Laguerre Functions

- Discrete-time Laguerre functions are generated from the discretization of continuous-time Laguerre functions and are given by

$$\begin{aligned}
\Gamma_1(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}} \\
\Gamma_2(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}} \cdot \frac{z^{-1}-a}{1-az^{-1}} \\
&\vdots \\
\Gamma_N(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}} \cdot \left(\frac{z^{-1}-a}{1-az^{-1}}\right)^{N-1}
\end{aligned}$$

- In this formulation, a represents a pole of the discrete-time Laguerre functions, so must satisfy $0 \le a < 1$ for stability
  - The free parameter a is an available degree of freedom for model design

- Orthonormality of the Laguerre functions can be expressed in terms of the the orthonormal equations

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} \mathit{\Gamma}_m\left(e^{jm}\right)\mathit{\Gamma}_m\left(e^{jw}\right)^* dw = 1$$

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} \mathit{\Gamma}_m\left(e^{jm}\right)\mathit{\Gamma}_n\left(e^{jw}\right)^* dw = 0 \quad m \neq n$$

- For compatibility with our state-space MPC formulation, it will be convenient to generate a state-space realization of the Laguerre functions
- We first note that

$$\mathit{\Gamma}_k(z) = \mathit{\Gamma}_{k-1}(z)\frac{z^{-1}-a}{1-az^{-1}}$$

where $\Gamma_1 = \dfrac{\sqrt{1-a^2}}{1-az^{-1}}$.

- Since we will be working with the Laguerre functions in the time domain, we will require the inverse Z-transforms of the $\mathit{\Gamma}_i(z)$
- Defining the vector sequences $\boldsymbol{l}_i[k]$ as the inverse Z-transform of $\mathit{\Gamma}_i(z)$, we assemble the column vectors $\boldsymbol{l}_i[k]$ into a matrix,

$$L[k] = \left[\begin{array}{cccc} \boldsymbol{l}_1[k] & \boldsymbol{l}_2[k] & \ldots & \boldsymbol{l}_N[k] \end{array}\right]$$

- It can be shown that the set of discrete-time Laguerre functions satisfy the following difference equation

$$L[k+1] = A_l L[k]$$

where the matrix A_l is dimension $N \times N$ and is a function of parameters a and $\beta = \left(1-a^2\right)$

– Initial conditions are given by

$$L[0]^T = \sqrt{\beta}\left[\begin{array}{cccccc} 1 & -a & a^2 & -a^3 & \ldots & (-1)^{N-1}a^{N-1} \end{array}\right]$$

Example 9.1

- For the case where $N = 5$ we have

$$A_l = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ \beta & a & 0 & 0 & 0 \\ -a\beta & \beta & a & 0 & 0 \\ a^2\beta & -a\beta & \beta & a & 0 \\ -a^3\beta & a^2\beta & -a\beta & \beta & a \end{bmatrix}; \quad L[0] = \sqrt{\beta}\begin{bmatrix} 1 \\ -a \\ a^2 \\ -a^3 \\ a^4 \end{bmatrix}$$

- A special case occurs when $a = 0$:

$$A_l = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 \\ 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 & 1 & 0 \end{bmatrix}$$

and

$$L[0]^T = \left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & \ldots & 0 \end{array}\right]$$

- In this case, the Laguerre functions become a set of pulses, thus collapsing to the disrete-time pulse response representation introduced in the standard formulation

Example 9.2

- Generate the first three Laguerre functions for the cases where $a = 0.5$ and $a = 0.9$.

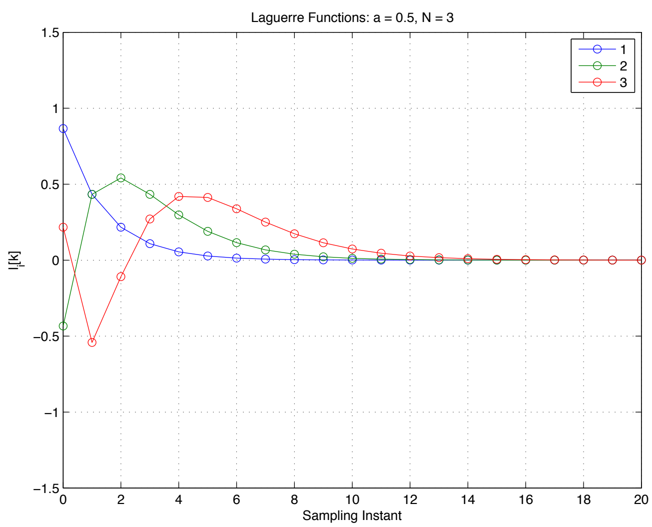
Solution

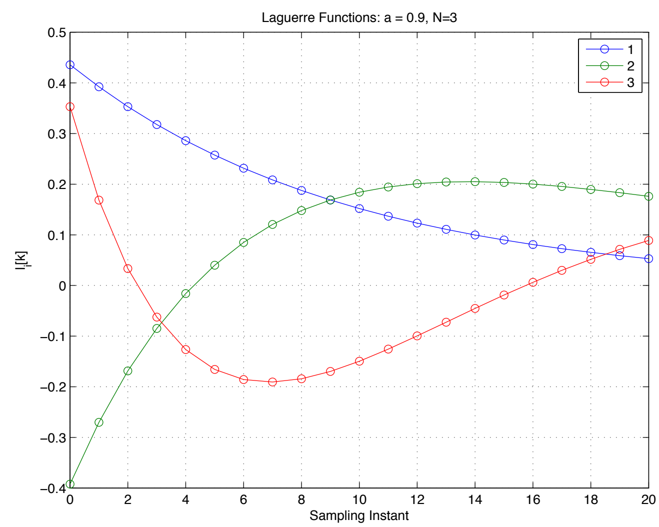
- Expanding the state equations for the first three Laguerre functions, we get

$$\begin{bmatrix} l_1[k+1] \\ l_2[k+1] \\ l_3[k+1] \end{bmatrix} = \begin{bmatrix} a & 0 & 0 \\ 1-a^2 & a & 0 \\ -a\left(1-a^2\right) & 1-a^2 & a \end{bmatrix} \begin{bmatrix} l_1[k] \\ l_2[k] \\ l_3[k] \end{bmatrix}$$

with the initial condition $l_1[0] = \sqrt{1-a^2}$, $l_2[0] = -a\sqrt{(1-a^2)}$, and $l_3[0] = a^2\sqrt{(1-a^2)}$

- Calculating the Laguerre functions iteratively, we obtain the following:





- It is easy to see that a faster Laguerre pole brings about faster response; e.g., for $a = 0.5$, the Laguerre functions decay to zero in less than 15 samples, whereas for $a = 0,9$, decay takes some 50 samples to occur

Laguerre Functions in MPC

- Laguerre functions have seen widespread use in the area of system modeling, where it is possible to characterize long impulse response functions with comparatively few Laguerre terms
- In model predictive control, Laguerre functions can be used to parameterize the optimal control trajectory in place of the traditional impulse response model

Design Framework

- Traditional MPC regards the control trajectory ($\triangle u[k]$, $\triangle u[k+1]$, $\triangle u[k+2]$, ..., $\triangle u[k+N_c]$) as the impulse response of a stable dynamic system
- In the framework of orthonormal functions, we can instead describe this dynamic sequence as a linear combination of Laguerre functions,

$$\triangle \boldsymbol{u}_k = L\boldsymbol{c}_k$$

  where $\boldsymbol{c}_k$ comprises the N Laguerre coefficients associated with time step k,

$$\boldsymbol{c}_k = \begin{bmatrix} c_1 & c_2 & \dots & c_N \end{bmatrix}^T$$

- Substituting for $\triangle \boldsymbol{u}_k$ in the state equations, we can re-write the output equation as

$$\boldsymbol{Y}_k = F\boldsymbol{X}_k + \Phi L\boldsymbol{c}_k$$

- Now, outputs are a direct function of the parameter vector $\boldsymbol{c}_k$, which can now be optimized directly in the MPC design

Cost Function

- Considering a SISO system, the original cost function is expressed as

$$J_k = (\boldsymbol{Y}_k - \boldsymbol{R}_s)^T (\boldsymbol{Y}_k - \boldsymbol{R}_s) + \triangle \boldsymbol{u}_k^T R \triangle \boldsymbol{u}_k$$

- Making the substitution for $\triangle \boldsymbol{u}_k$ above, and proceeding as in Chapter 4 to satisfy the stationarity condition, we obtain an optimal solution for the Laguerre parameter vector $\boldsymbol{c}_k$,

$$\boldsymbol{c}_k^* = \left[L^T \Phi^T \Phi L + \lambda L^T L\right]^{-1} L^T \Phi^T \left[\boldsymbol{R}_s - F \boldsymbol{X}_k\right]$$

  – Note that orthonormality implies $L^T L = I$; this will only be achieved when the Laguerre functions decay completely

- Thus the optimal input sequence is recovered as

$$\triangle \boldsymbol{u}_k^* = L \boldsymbol{c}_k^*$$

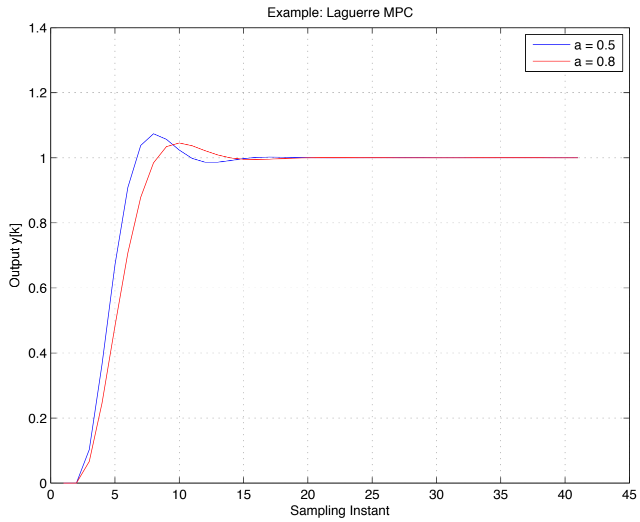
Example 9.3

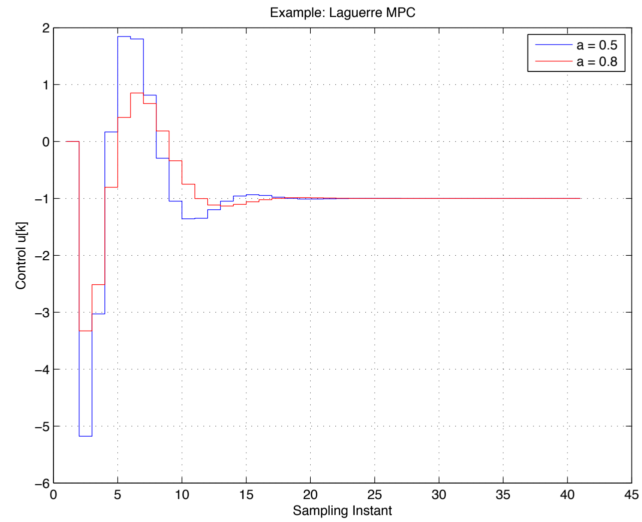
- Recall the undamped oscillator system from Example 5.1,

$$\begin{bmatrix} x_1[k+1] \\ x_2[k+1] \end{bmatrix} = \begin{bmatrix} 0.9801 & 0.0993 \\ -0.3973 & 0.9801 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix} + \begin{bmatrix} 0.0993 \\ -0.0199 \end{bmatrix} u[k]$$

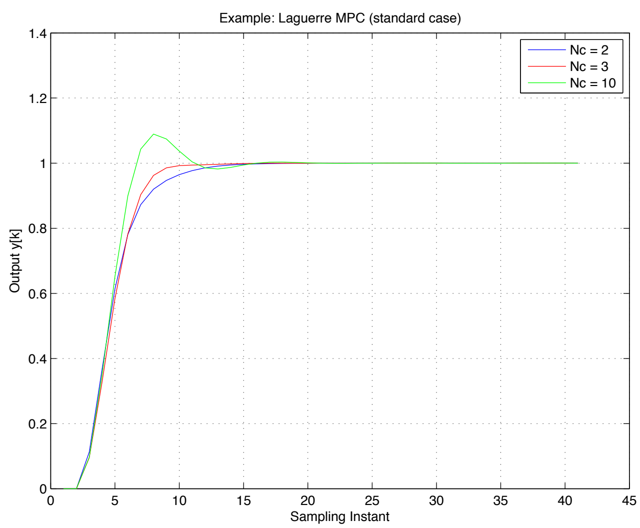
$$y[k] = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix}$$

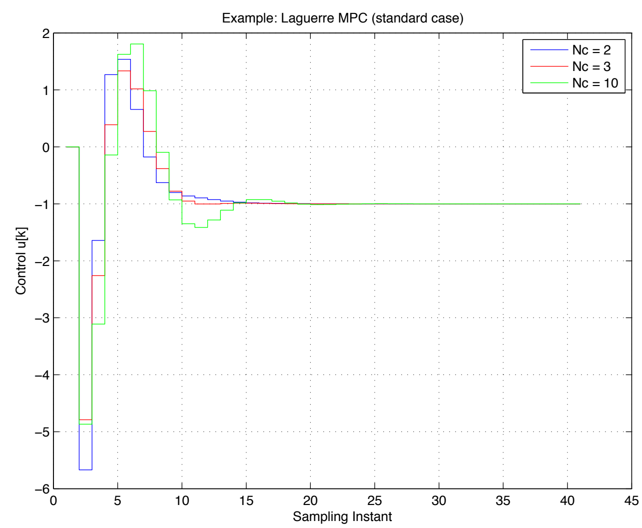
- We now implement MPC using Laguerre functions as presented above, setting the number of Laguerre function, $N = 3$, the Laguerre pole parameter $a = 0.5$, and the prediction horizon $N_p = 10$
- In addition, the control weighting is set to $\lambda = 0.01$
- The plots below show the output for Laguerre MPC when the parameter $a = 0.5$ and $a = 0.8$



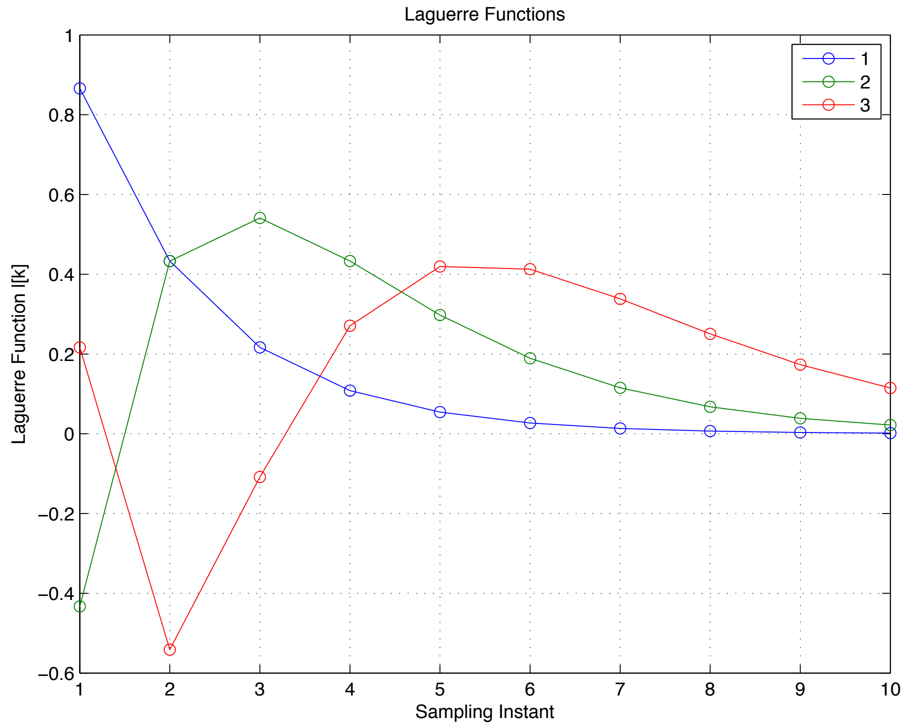


- For comparison, the same system was implemented with standard MPC control for $N_p = 10$ and prediction horizon values $N_c = 2$, 3, and 10.





- Note that the Laguerre MPC algorithm solves for an approximation to the case where $N_c = N_p$; the similarity between the Laguerre and standard case is evident in the example
- Let's take a look at the Laguerre functions for this example,



- Clearly, the functions do not decay completely within the prediction horizon; this could also be gleaned from the plot of Example 9.2
- Incomplete decay can also be detected by testing the Laguerre function matrix for orthonormality:

$$L^T L = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & .9998 & -0.0014 \\ -.0001 & -.0014 & .9914 \end{bmatrix}$$

- The matrix departs from the identity, indicating the function set is not perfectly orthonormal

- Note further that as $N_p \to \infty$, then the Laguerre solution approaches the LQR result (for a sufficient number of Laguerre coefficients)
  - An approximation to the LQR result will still be obtained for finite N_p so long as the Laguerre functions decay completely and sufficient coefficients are taken

MPC Incorporating Direct Feed-Through Terms

- Standard formulations of MPC (like the state-space approach of Chapter 4) assume that the "D" term is equal to zero
- This assumption is a reasonable one for many real-world processes, and simplifies the development of the algorithm
- Nonetheless, processes exist where direct feed-through plays an important role in the dynamic response
  - One example is the control of lithium-ion battery charging/discharging where the cell's ohmic resistance generates a near immediate voltage change with the application of current
- In order to cater for such systems, we need to consider a modification to the standard MPC algorithm
- Ordys and Pike[1] developed an alternative formulation for state-space models which admits a direct feed-through term while retaining embedded integral action

Modified Algorithm Formulation

- We consider a full state-space model of the form

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k + Bu_k \\ y_k &= C\boldsymbol{x}_k + Du_k \end{aligned}$$

[1] A. Ordys and A. Pike, "State-Space Generalized Predictive Control Incorporating Direct Through Terms", *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, FL, 1998.

- Recall for the standard case we assumed $D = 0$ and defined an augmented state-space vector of the form

$$\boldsymbol{X}_k = \left[\begin{array}{c} \triangle \boldsymbol{x}_k \\ \hline y_k \end{array}\right]$$

where $\triangle \boldsymbol{x}_k = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$.

- This gave rise to the augmented state equation

$$\left[\begin{array}{c} \triangle \boldsymbol{x}_{k+1} \\ \hline y_{k+1} \end{array}\right] = \left[\begin{array}{c|c} A & 0 \\ \hline CA & I \end{array}\right] \left[\begin{array}{c} \triangle \boldsymbol{x}_k \\ \hline y_k \end{array}\right] + \left[\begin{array}{c} B \\ \hline CB \end{array}\right] \triangle u_k$$

$$y_k = \left[\begin{array}{c|c} 0 & I \end{array}\right] \left[\begin{array}{c} \triangle \boldsymbol{x}_k \\ \hline y_k \end{array}\right]$$

- In the alternative formulation, we will define a new augmented state vector:

$$\boldsymbol{\chi}_k = \left[\begin{array}{c} \boldsymbol{x}_k \\ \hline u_k \end{array}\right]$$

where the input u_k now appears as a state.

- If we consider the state equations written as follows

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + Bu_k$$
$$u_{k+1} = u_k + \triangle u_{k+1}$$

where $\triangle u_{k+1} = u_{k+1} - u_k$, then we can write the augmented state equation as

$$\boldsymbol{\chi}_{k+1} = \left[\begin{array}{c|c} A & B \\ \hline 0 & I \end{array}\right] \boldsymbol{\chi}_k + \left[\begin{array}{c} 0 \\ I \end{array}\right] \triangle u_{k+1}$$

$$y_k = \left[\begin{array}{cc} C & D \end{array}\right] \boldsymbol{\chi}_k$$

- Importantly, this form resembles the "D=0" form of the standard algorithm and allows us to utilize much of the same mathematical machinery in its solution
- Applying corresponding definitions, we may re-write the state equation as

$$\begin{aligned}\boldsymbol{\chi}_{k+1} &= \tilde{A}\boldsymbol{\chi}_k + \tilde{B}\triangle u_{k+1}\\ y_k &= \tilde{C}\,\boldsymbol{\chi}_k\end{aligned}$$

Modified Algorithm Optimal Solution

- We proceed exactly as in Chapter 4 to generate a recursive form of the state equation, i.e.,

$$\begin{aligned}\boldsymbol{\chi}_{k+1} &= \tilde{A}\boldsymbol{\chi}_k + \tilde{B}\triangle u_{k+1}\\ \boldsymbol{\chi}_{k+2} &= \tilde{A}\boldsymbol{\chi}_{k+1} + \tilde{B}\triangle u_{k+2}\\ &= \tilde{A}\left[\tilde{A}\boldsymbol{\chi}_k + \tilde{B}\triangle u_{k+1}\right] + \tilde{B}\triangle u_{k+2}\\ &= \tilde{A}^2\boldsymbol{\chi}_k + \tilde{A}\tilde{B}\triangle u_{k+1} + \tilde{B}\triangle u_{k+2}\\ \boldsymbol{\chi}_{k+N_p+1} &= \tilde{A}^{N_p+1}\boldsymbol{\chi}_k + \tilde{A}^{N_p}\tilde{B}\triangle u_k + \cdots + \tilde{B}\triangle u_{k+i}\end{aligned}$$

- Assembling into matrix-vector form,

$$\begin{bmatrix}\boldsymbol{\chi}_{k+1}\\ \boldsymbol{\chi}_{k+2}\\ \vdots\\ \boldsymbol{\chi}_{k+N_p+1}\end{bmatrix} = \begin{bmatrix}\tilde{A}\\ \tilde{A}^2\\ \vdots\\ \tilde{A}^{N_p+1}\end{bmatrix}\boldsymbol{\chi}_k + \begin{bmatrix}\tilde{B} & 0 & \cdots & 0\\ \tilde{A}\tilde{B} & \tilde{B} & \cdots & 0\\ \vdots & & \ddots & \vdots\\ \tilde{A}^{N_p}\tilde{B} & \tilde{A}^{N_p-1}\tilde{B} & \cdots & \tilde{A}^{N_p-N_c-1}\tilde{B}\end{bmatrix}\begin{bmatrix}\triangle u_{k+1}\\ \triangle u_{k+2}\\ \vdots\\ \triangle u_{k+N_c+1}\end{bmatrix}$$

- Since the output is calculated from $y_k = C\,\boldsymbol{\chi}_k$, we can assemble the vector of future outputs as

$$\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ \vdots \\ y_{k+N_p+1} \end{bmatrix} = \begin{bmatrix} \tilde{C} \\ \tilde{C}\tilde{A}^1 \\ \vdots \\ \tilde{C}\tilde{A}^{N_p} \end{bmatrix} \tilde{A}\boldsymbol{\chi}_k$$

$$+ \begin{bmatrix} \tilde{C}\tilde{B} & 0 & \cdots & 0 \\ \tilde{C}\tilde{A}\tilde{B} & \tilde{C}\tilde{B} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ \tilde{C}\tilde{A}^{N_p}\tilde{B} & \tilde{C}\tilde{A}^{N_p-1}\tilde{B} & \cdots & \tilde{C}\tilde{A}^{N_p-N_c-1}\tilde{B} \end{bmatrix} \begin{bmatrix} \triangle u_{k+1} \\ \triangle u_{k+2} \\ \vdots \\ \triangle u_{k+N_c+1} \end{bmatrix}$$

- Again, applying corresponding definitions, we write the future output equation as

$$\underset{\rightarrow k}{\boldsymbol{Y}} = \Phi\tilde{A}\boldsymbol{\chi}_k + G\triangle \underset{\rightarrow k}{\boldsymbol{u}}$$

where the notation '$\rightarrow$' denotes a vector of future quantities.

- The vector of optimal future input values may be found by defining the cost function

$$J_k = \left(\boldsymbol{R}_s - \underset{\rightarrow k}{\boldsymbol{Y}}\right)^T \left(\boldsymbol{R}_s - \underset{\rightarrow k}{\boldsymbol{Y}}\right) + \triangle \underset{\rightarrow k}{\boldsymbol{u}}^T R \triangle \underset{\rightarrow k}{\boldsymbol{u}}$$

and satisfying the stationarity condition

$$\frac{\partial J_k}{\partial \triangle \underset{\rightarrow k}{\boldsymbol{u}}} = 0$$

- Solving,

$$\triangle \underset{\rightarrow k}{\boldsymbol{u}}^* = \left(G^T G + \lambda I\right)^{-1} G^T \left(\boldsymbol{R}_s - \Phi\tilde{A}\boldsymbol{\chi}_k\right)$$

where we have defined $R = \lambda I$.

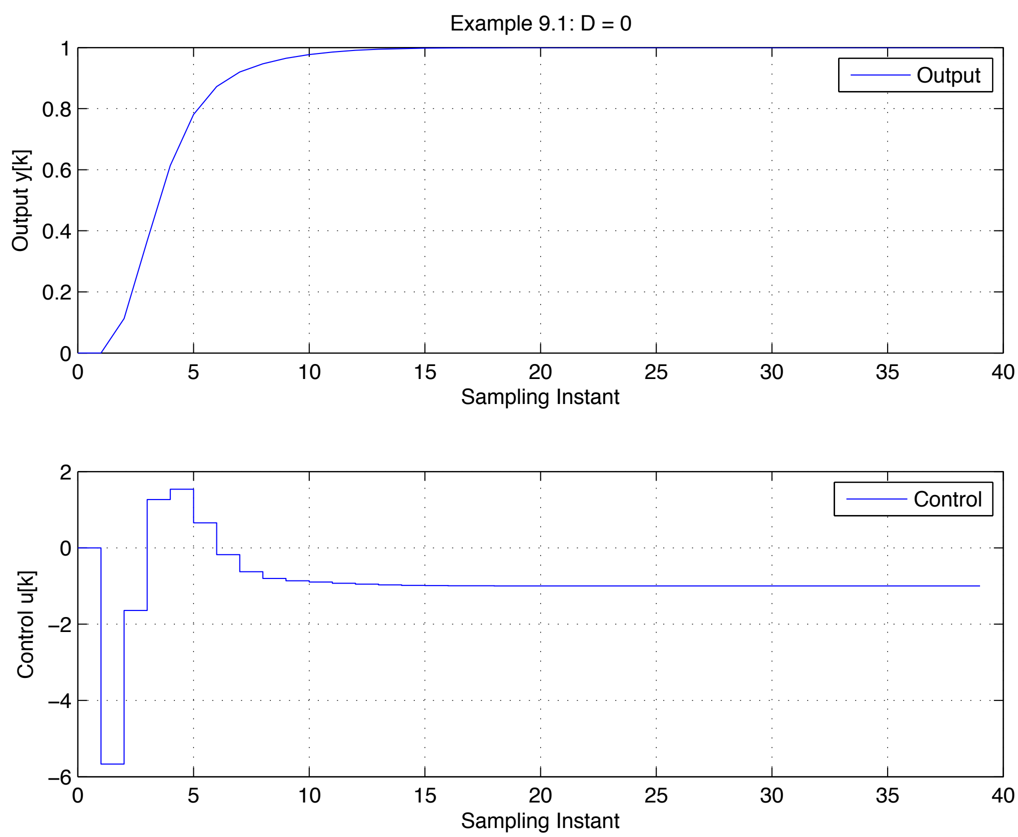
Example 9.4

Let us return to the undamped oscillator system of Example 5.1,

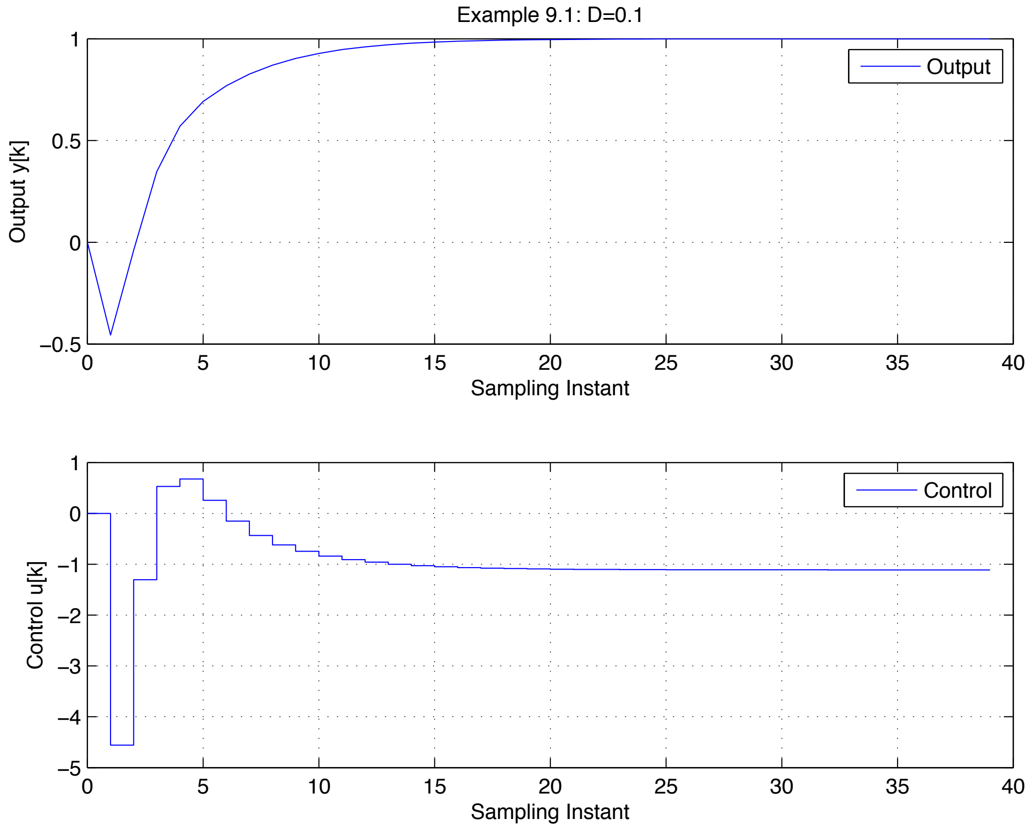
$$\begin{bmatrix} x_1[k+1] \\ x_2[k+1] \end{bmatrix} = \begin{bmatrix} 0.9801 & 0.0993 \\ -0.3973 & 0.9801 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix} + \begin{bmatrix} 0.0993 \\ -0.0199 \end{bmatrix} u[k]$$

$$y[k] = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix}$$

Applying unconstrained MPC with $N_p = 10$, $N_c = 2$, and $\lambda = 0.01$, we obtain the following response to a unit step input.



If we now introduce a non-zero D-matrix, $D = 0.1$, and apply the O&P modified MPC algorithm we obtain the following result.



- Note the significant difference in the output response, which now reflects the non-minimum phase behavior of a zero at 1.6837 which was introduced by the addition of the D-matrix.