MPC Performance Tuning

- MPC algorithms incorporate a number of adjustable design parameters that give enormous flexibility to the configuration of the control system
- By delivering all this flexibility, they can also complicate the task of controller tuning
- Among the degrees of freedom one can deploy in order to achieve key control objectives in MPC are the following:
  - Horizons, N_p, N_c
  - Weights, Q, R
  - Observer dynamics
  - Reference trajectory
  - Disturbance model
- The notion of *tuning* in MPC refers to the systematic adjustment of problem degrees of freedom with a view toward achieving desired performance outcomes
- Before addressing specific tuning guidelines, et's first review some key ideas related to stability and performance for control systems

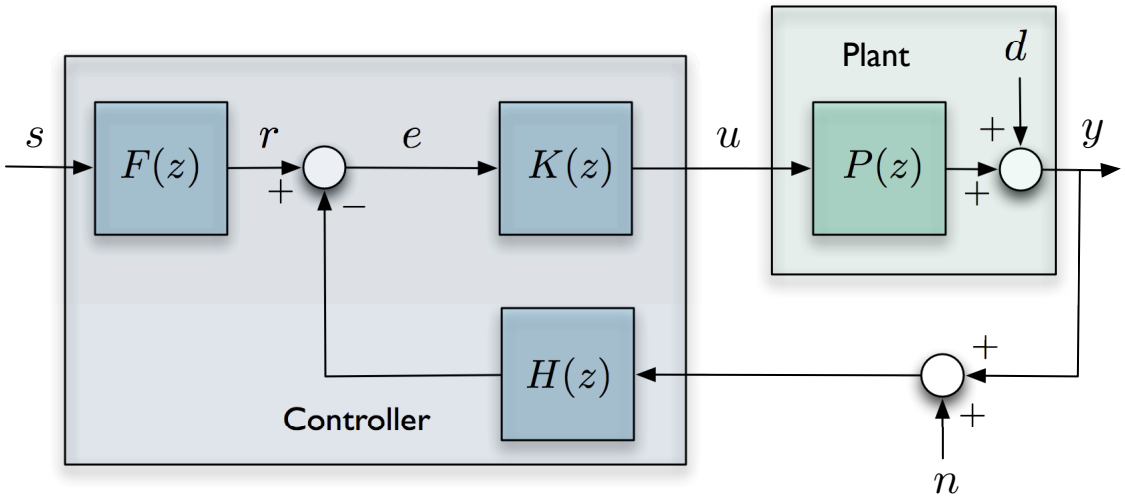
The role of feedback

Feedback is used to overcome the effects of uncertainty

Feedback is dangerous

- It is well-known that introducing feedback brings with it the possibility of destabilizing a stable system
- If the system model and environment are known perfectly, then perfect control can be achieved using an open-loop control strategy
- But feedback can be very effective for reducing effects of disturbances and model uncertainty
  - If our *only* objective was to follow a set-point well, then we don't need feedback - open-loop works fine!
- But we can deduce other information from set-point responses, so it's not pointless to use them

Two-degree-of-freedom system



- In the two-degree-of-freedom system, set-point $\boldsymbol{s}$ is filtered by a pre-filter $F(z)$ before becoming reference $\boldsymbol{r}$
- Controller has two parts: i) forward path controller $K(z)$; and ii) feedback path controller $H(z)$
- Output disturbance $\boldsymbol{d}$ and measurement noise $\boldsymbol{n}$ are also present
- We need to find transfer functions relating input signals $\boldsymbol{s}$, $\boldsymbol{d}$, and $\boldsymbol{n}$, to output signals $\boldsymbol{y}$ and $\boldsymbol{u}$
- For the general (vector-valued) case,

$$\begin{aligned}\boldsymbol{y}(z) &= \boldsymbol{d}(z) + P(z)K(z)\boldsymbol{e}(z)\\ \boldsymbol{e}(z) &= F(z)\boldsymbol{s}(z) - H(z)\left[\boldsymbol{n}(z) + \boldsymbol{y}(z)\right]\end{aligned}$$

- This gives

$$\boldsymbol{y}(z) = \boldsymbol{d}(z) + P(z)K(z)\left\{F(z)\boldsymbol{s}(z) - H(z)\left[\boldsymbol{n}(z) + \boldsymbol{y}(z)\right]\right\}$$

  and

$$\left[I + P(z)K(z)H(z)\right]\boldsymbol{y}(z) =$$

$$\boldsymbol{d}(z) + P(z)K(z)F(z)\boldsymbol{s}(z) - P(z)K(z)H(z)\boldsymbol{n}(z)$$

- We can express this result as

$$\boldsymbol{y}(z) = S(z)\boldsymbol{d}(z) + S(z)P(z)K(z)F(z)\boldsymbol{s}(z) - T(z)\boldsymbol{n}(z)$$

  where we define the *sensitivity function* $S(z)$ and the *complementary sensitivity function* $T(z)$ as

$$\begin{aligned}S(z) &= \left[I + P(z)K(z)H(z)\right]^{-1}\\ T(z) &= \left[I + P(z)K(z)H(z)\right]^{-1}P(z)K(z)H(z)\end{aligned}$$

- In feedback theory, the smaller the sensitivity, the better the feedback, i.e., effect of $\boldsymbol{d}$ is kept small
- Likewise, the smaller the complementary sensitivity, the smaller the effect of measurement noise $\boldsymbol{n}$
- Note also
$$S(z) + T(z) = I$$
which means you can't have $S(z)$ and $T(z)$ small (near zero) simultaneously
- Note that the response of the output to the set-point can be designed independantly via the pre-filter $F(z)$
- The MPC control problem considers tracking a reference subject to constraints
  - Although this looks straighforward mathematically in the algorithm, it's not necessarily related to good feedback properties
  - It is especially challenging to consider the effect of feedback properties under constraints - few analytical tools exist

Special Cases

- Here we consider some special choices of parameters and deduce how the predictive controller will behave

Mean-level control

- Choose $N_c = 1$ and $R = 0$.
- The cost function can be written,

$$J(k) = \sum_{i=1}^{N_p} \| \hat{y}(k+i|k) - r \|_Q^2$$

- Since here we can only untilize one control move, $\triangle u(k)$, then if $N_p \to \infty$, the optimal strategy is to move the control to that level which will give $y = r$ in the steady-state
- In this case, the transient response of the system will be the open-loop response of the plant
- This is a common configuration for feedback systems; it also gives us a compatible form with which to describe the unconstrained model predictive controller formulation developed previously and shown below

Deadbeat control

- Let $N_c = n$, the number of states in the plant
- Assume $R = 0$; assume $r(k+i) = r$, and choose $N_p \geq 2n$
- The cost function is then,

$$J(k) = \sum_{i=n}^{2n} \| \hat{y}(k+i|k) - r \|_Q^2$$

'Perfect' control

- Choose $N_p = 1$ and $N_c = 1$ with $R = 0$
- The cost function is then,

$$J(k) = \| \hat{y}(k+1|k) - r(k+1) \|_Q^2$$

- An optimal strategy in this case is to choose input signals so that the next output matches the input reference as closely as possible. There is no concern about future steps.
- Consider the SISO transfer function case,

$$Y(z) = G(z)U(z) = \frac{B(z)}{A(z)}U(z)$$

- Writing as a difference equation,
  $b_0u(k) = y(k+1) + \ldots + a_n y(k+1-n) - b_1u(k-1) - \ldots - b_nu(k-n)$
- We can choose $u(k)$ such as to make $y(k+1) = r(k+1)$ by setting
  $b_0u(k) = r(k+1) + a_1y(k) + \ldots + a_n y(k+1-n) - b_1u(k-1) - \ldots - b_nu(k-n)$
- If this is done at each step, we will eventually have

$$y(k) = r(k)$$
$$y(k-1) = r(k-1)$$

  so that,
  $b_0u(k) = r(k+1) + a_1r(k) + \ldots + a_n r(k+1-n) - b_1u(k-1) - \ldots - b_nu(k-n)$

  or

$$B(z^{-1})u(k) = A(z^{-1})r(k+1)$$

- So, what we essentially have is a controller that is the *inverse* of the plant.

Example 7.1

- It will be instructive to examine an example to develop insight into the effect various tuning parameters have on stability and performance
- Consider the system described by (Rossiter, Ch. 5.2)

$$Y(z) = \frac{z^{-1} + 0.2z^{-2}}{(1 - 0.9z^{-1})(1 - 0.8z^{-1})} U(z)$$

$$G(z) = \frac{z^{-1} + 0.2z^{-2}}{1 - 1.7z^{-1} + 0.72z^{-2}}$$

$$= \frac{z + 0.2}{z^2 - 1.7z + 0.72}$$

– The state space representation is

$$\boldsymbol{x}_m(k+1) = \begin{bmatrix} 1.7 & -0.72 \\ 1 & 0 \end{bmatrix} \boldsymbol{x_m}(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0.2 \end{bmatrix} \boldsymbol{x_m}(k) + [0]\, u(k)$$

- Writing the augmented state vector,

$$\boldsymbol{x}(k) = \begin{bmatrix} \triangle \boldsymbol{x}_m(k) \\ y(k) \end{bmatrix}$$

– The state equations become

$$\boldsymbol{x}(k+1) = \begin{bmatrix} \triangle \boldsymbol{x}_m(k) \\ y(k) \end{bmatrix} = \begin{bmatrix} A_m & 0_m^t \\ C_m A_m & 1 \end{bmatrix} \begin{bmatrix} \triangle \boldsymbol{x}_m(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} \triangle u(k)$$

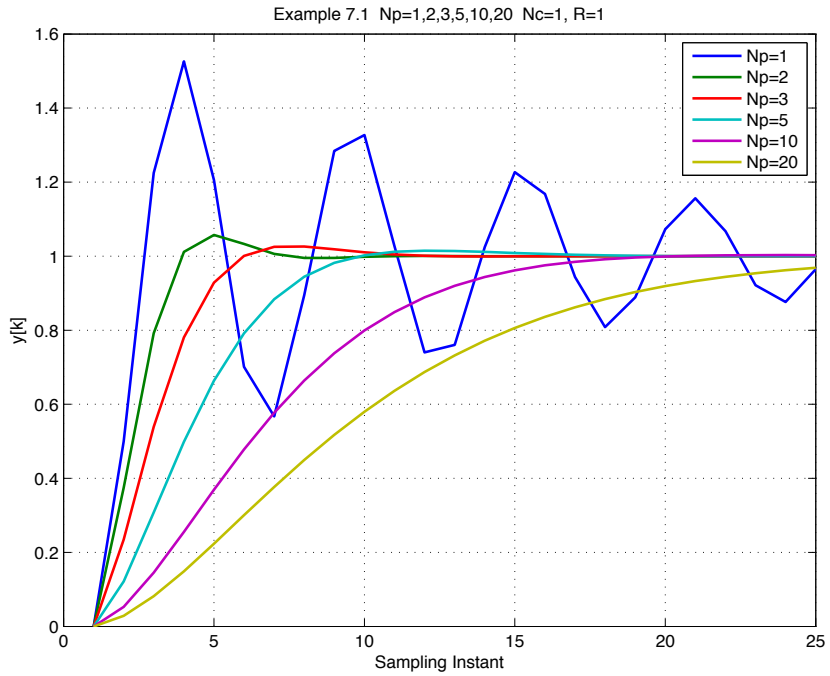
$$y(k) = \begin{bmatrix} 0_m & 1 \end{bmatrix} \begin{bmatrix} \triangle \boldsymbol{x}_m(k) \\ y(k) \end{bmatrix}$$

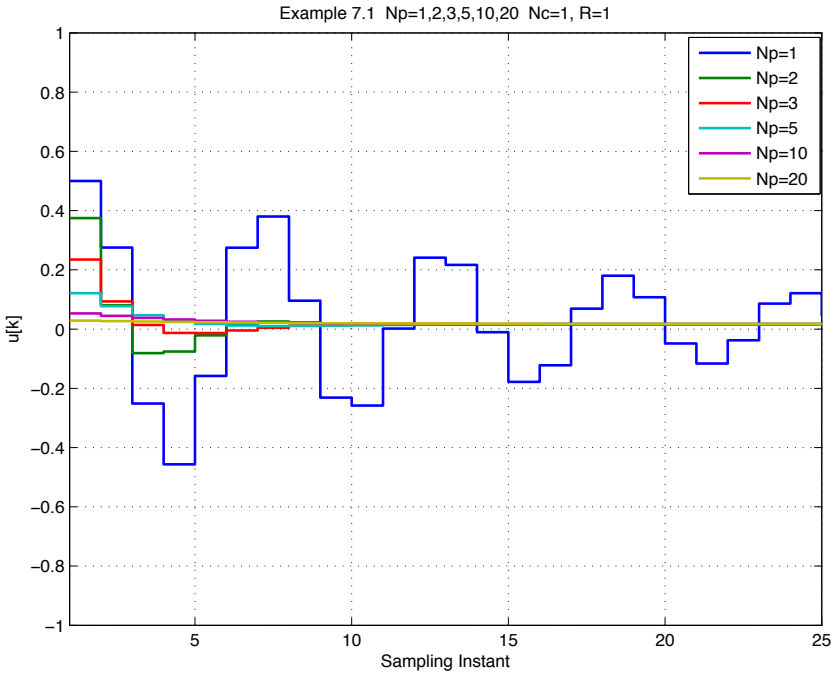
which for our example this gives

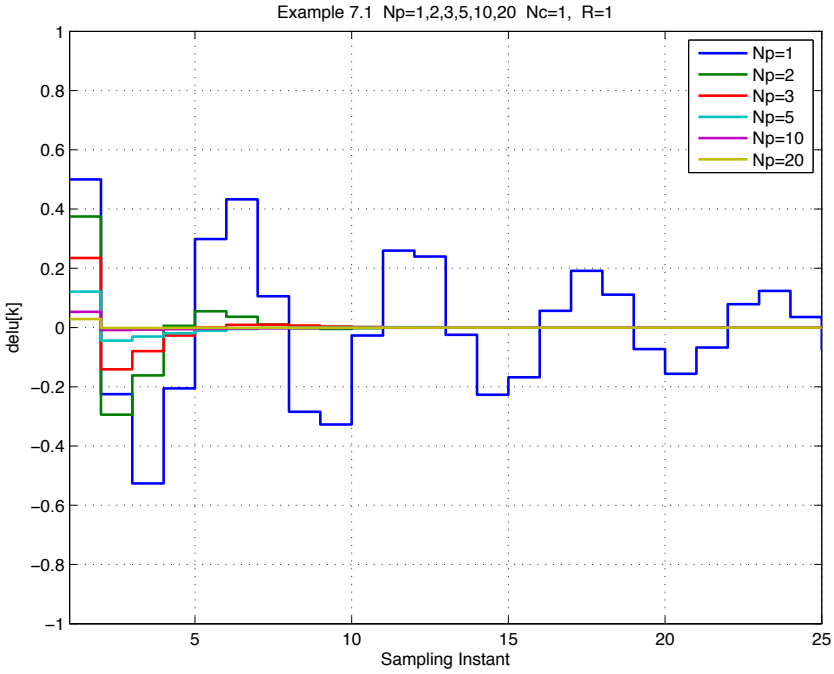
$$A = \begin{bmatrix} 1.7 & -0.72 & 0 \\ 1 & 0 & 0 \\ 1.9 & -0.72 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

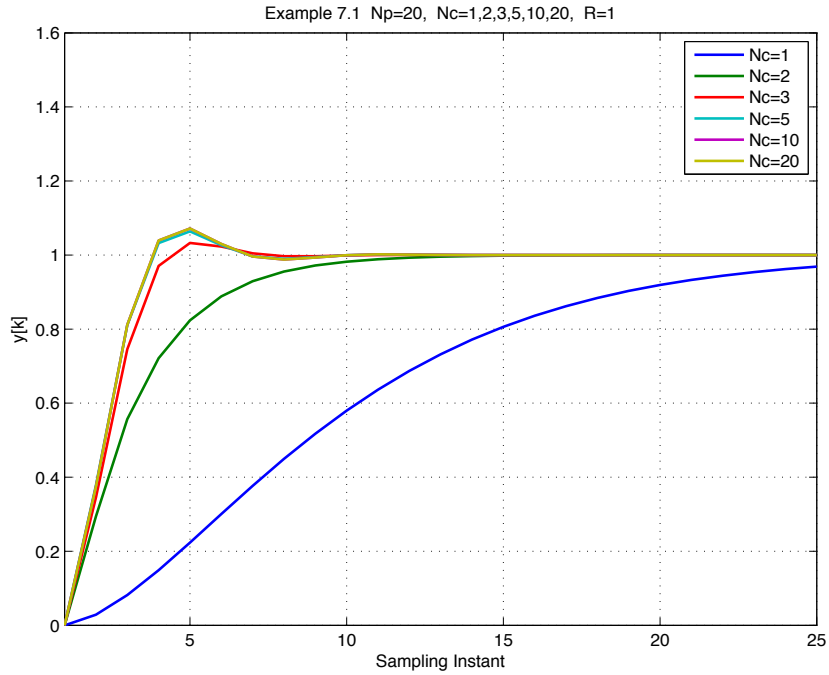
- Note that the eigenvalues of A are $\lambda(A) = \left\{ 1.0 \;\; 0.9 \;\; 0.8 \right\}$.
- Using this example, we can systematically vary the parameters and examine the effect on output response and control input.
- First we'll examine the case where $N_c = 1$ and vary the prediction horizon. The following plots depict the output, control input and control input increment as functions of varying the prediction horizon.

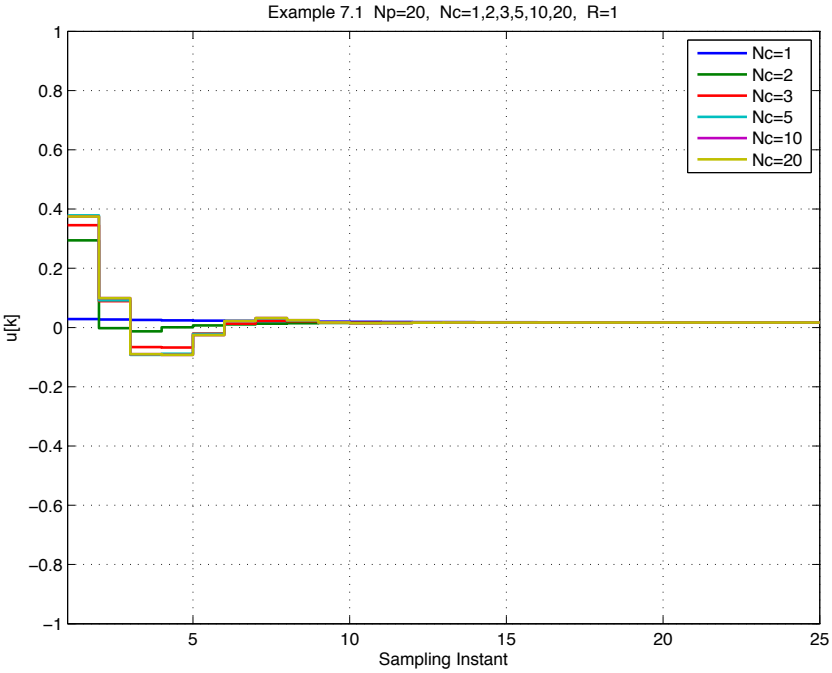


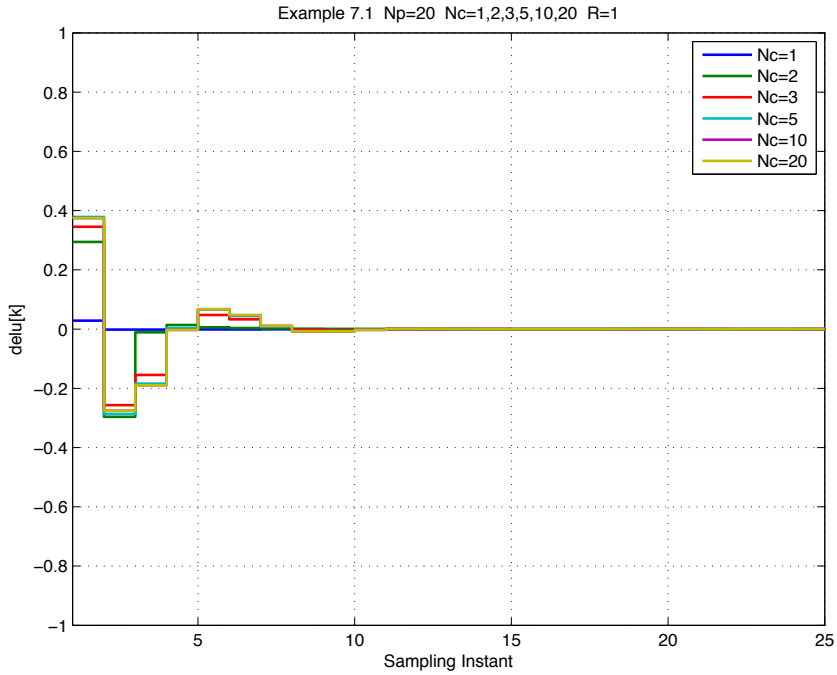




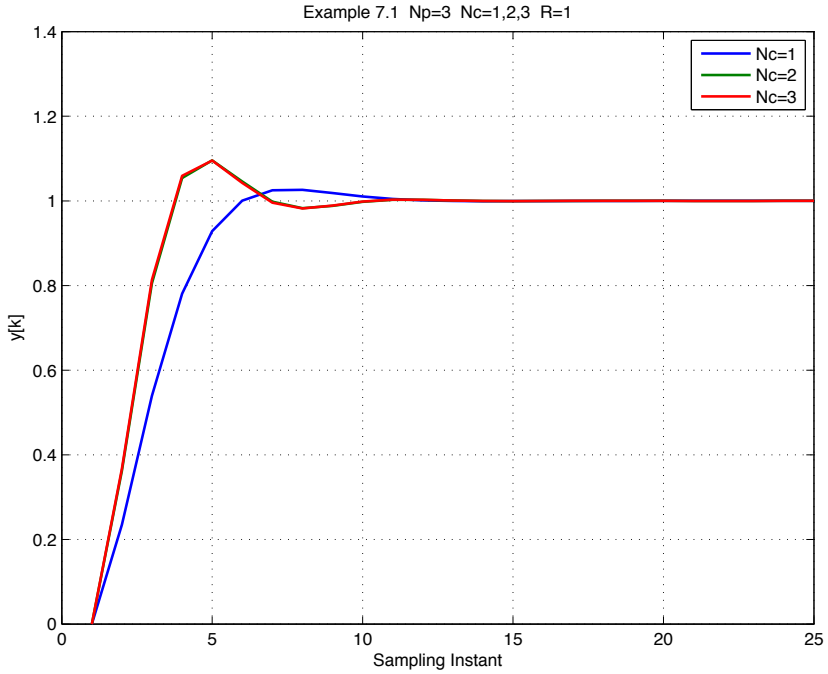
- Next, we'll fix the prediction horizon at $N_p = 20$ and vary the control horizon. Resulting plots follow.

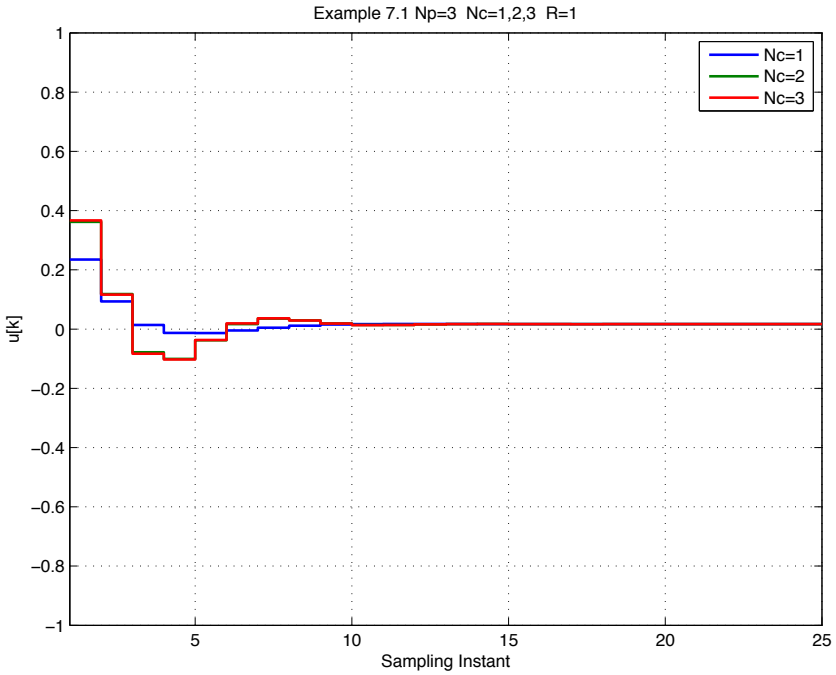


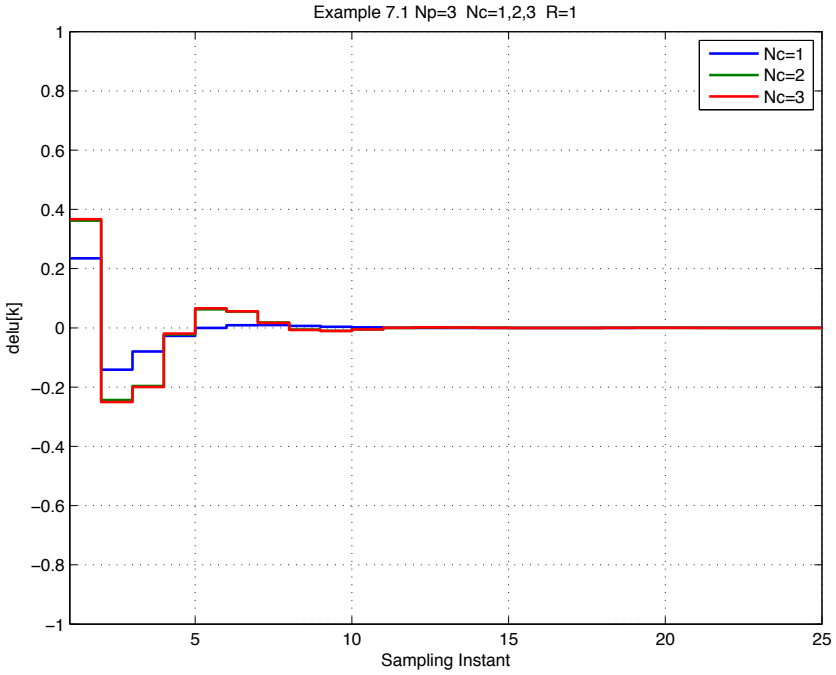




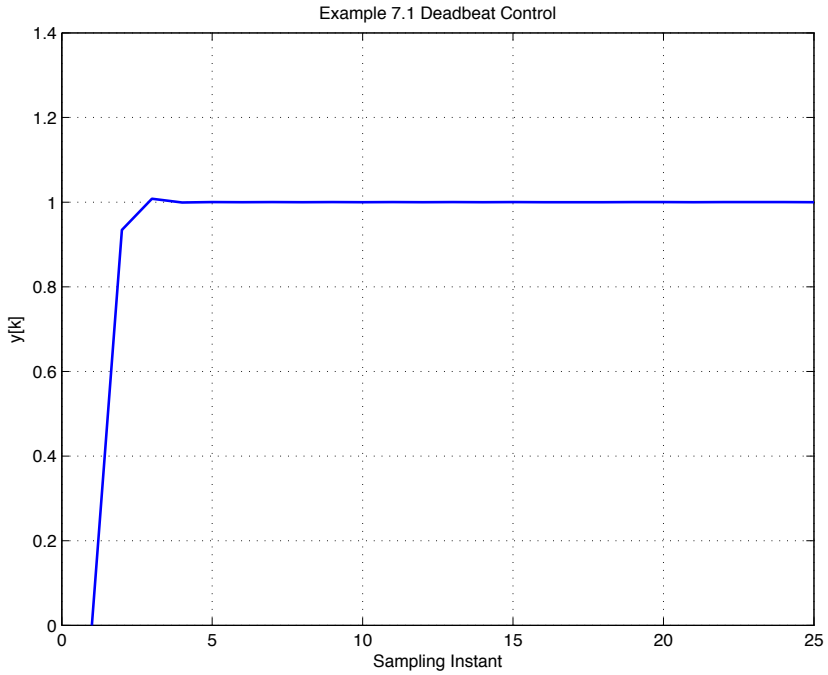
- It appears from these plots that qualitative improvements in system response are achieved for increasing values of the control horizon
- Setting the prediction horizon to a smaller value ($N_p = 3$) gives the following results.



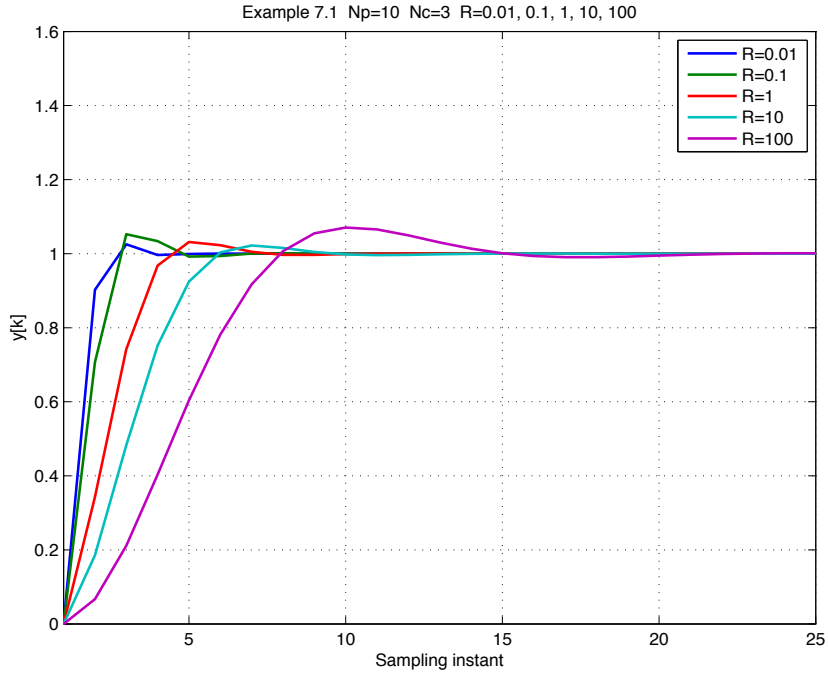


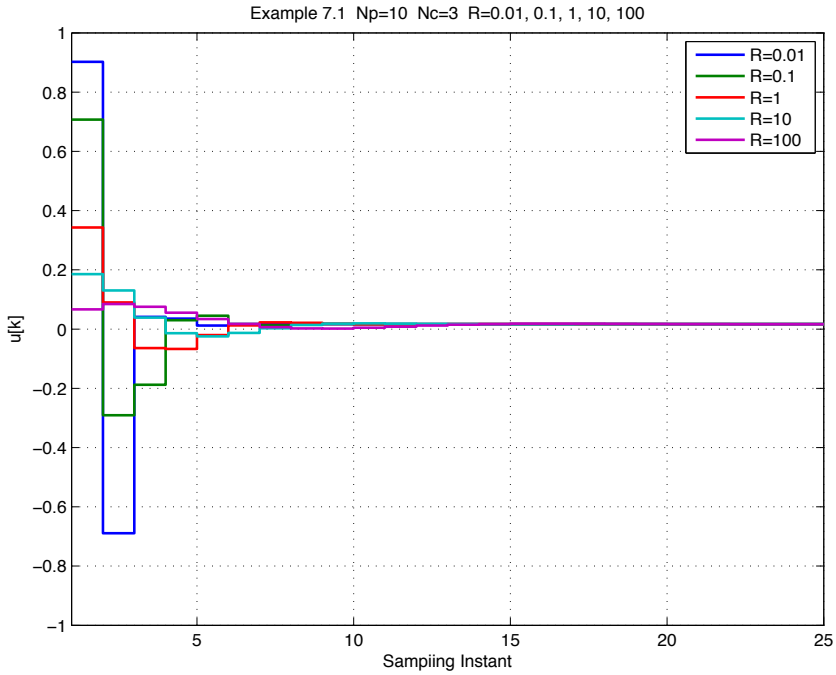


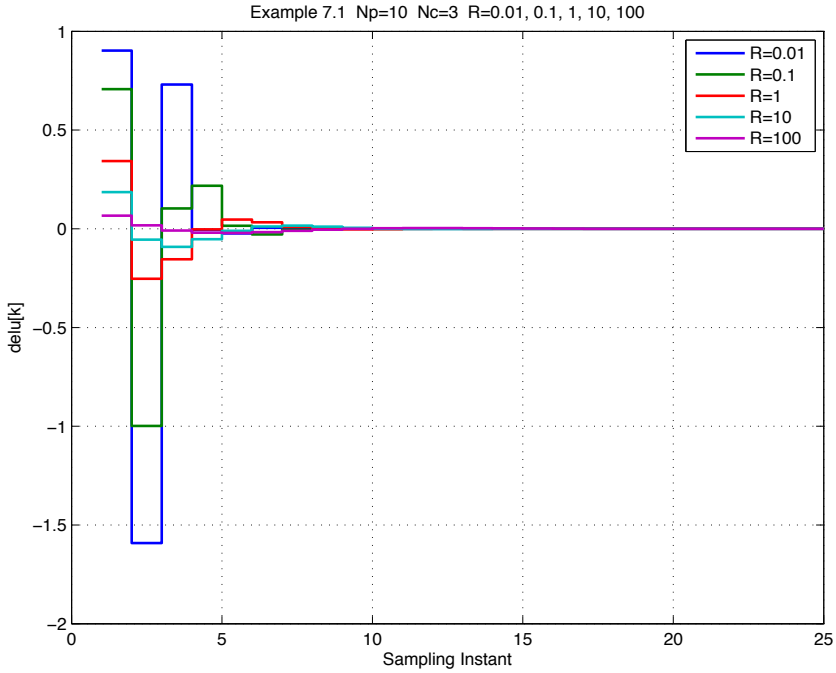
- Clearly, increasing N_c improves the response for small values of N_p
- For comparison, we plot the deadbeat response (achieved by setting $R = 0$)



- Now we'll set $N_p = 10$, $N_c = 3$ and vary the control increment weighting

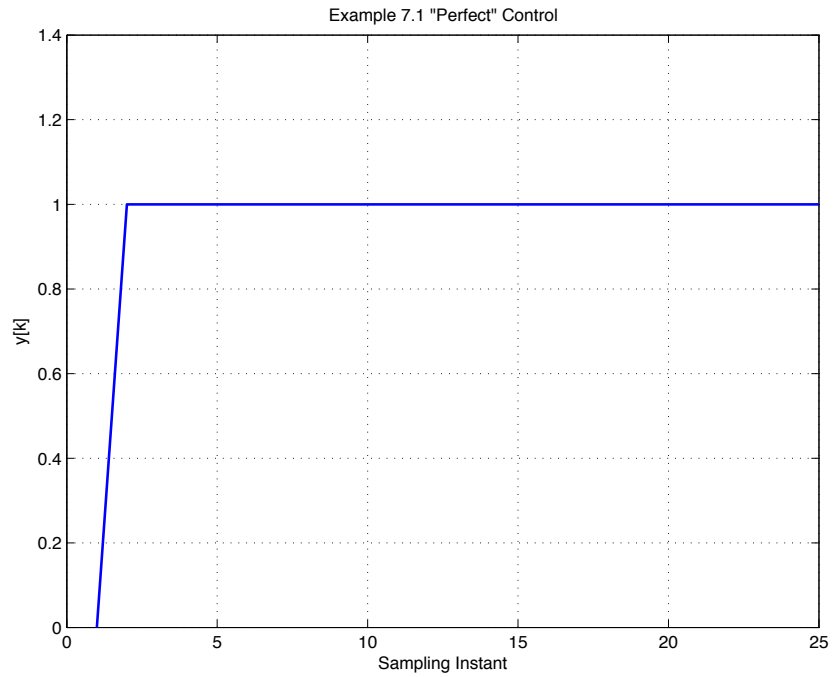


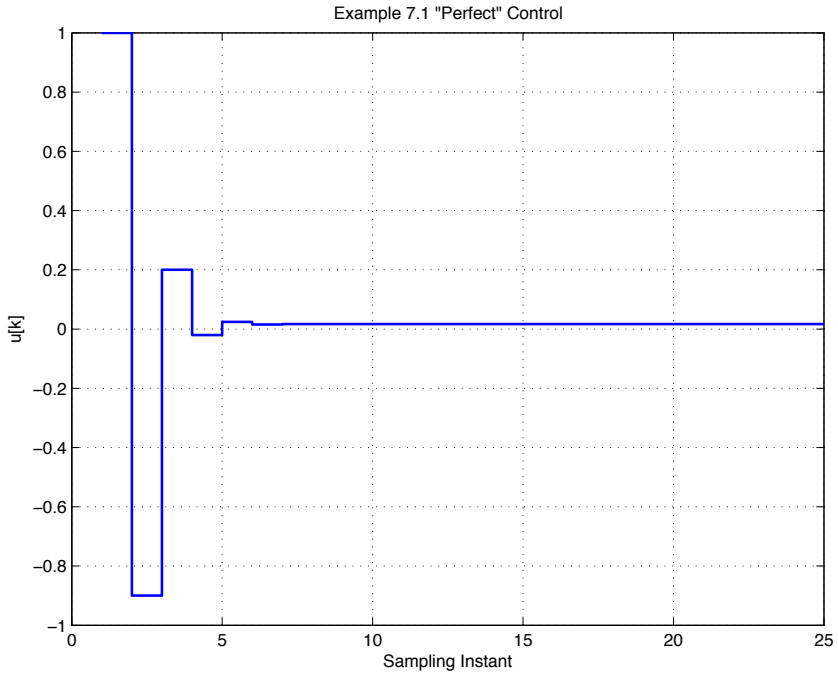


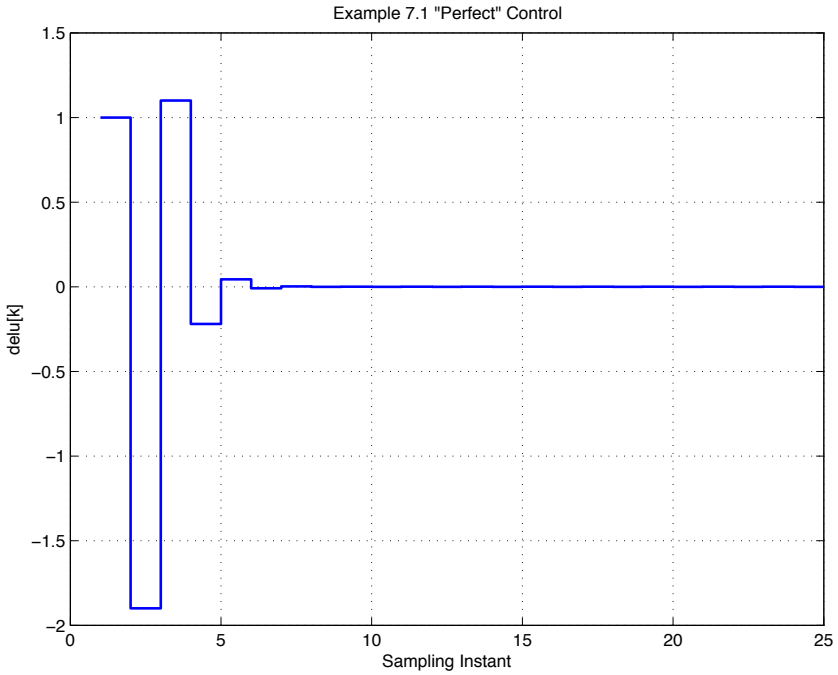


- It is apparent that lower values of control increment weighting allow larger excursions in the control signal and bring about a faster response

- Let's now see if we can force "perfect" control by setting $N_p = N_c = 1$ and $R = 0$. Results appear below.







- It looks like we've achieved the output objective in a single time step
  - We can verify this result by solving the simultaneous set of difference equations

$$y[k] = 1.7y[k-1] - 0.72y[k-2] + u[k-1] + 0.2u[k-2]$$

$$\begin{aligned} y[1] &= 1.7 \cdot y[0] - 0.72 \cdot 0 + u[0] + 0.2 \cdot 0 = 1 \\ y[2] &= 1.7 \cdot y[1] - 0.72 \cdot y[0] + u[1] + 0.2 \cdot u[0] = 1 \\ y[3] &= 1.7 \cdot y[2] - 0.72 \cdot y[1] + u[2] + 0.2 \cdot u[1] = 1 \\ &\vdots \quad \vdots \quad \vdots \end{aligned}$$

  - Note that this works here since $G(z)$ is proper (but not strictly proper)
- Let's state some general observations:
  - If N_c is small, then increasing N_p results in slower loop dynamics (approaches open-loop)
  - If N_c is large, then increasing N_p improves performance

  - If N_p is small, then increasing N_c can lead to near deadbeat behavior
  - Increasing control weighting R slows down the response
  - Decreasing control weighting R speed up the response

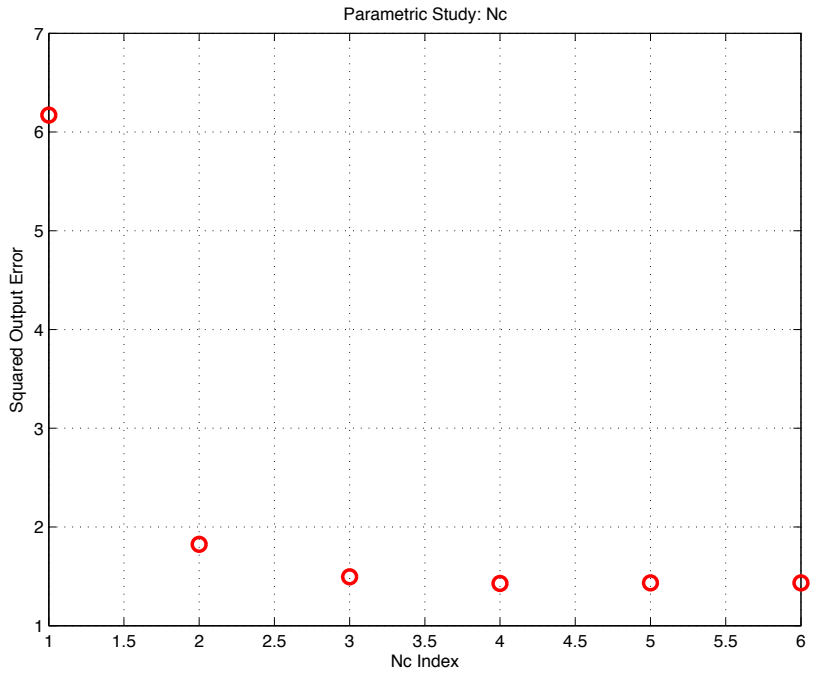
- How do we choose horizons?
- Some guidelines:
  - As N_p is increased, nominal closed-loop performance improves if N_c is large enough
  - As N_c is increased, nominal closed-loop performance improves if N_p is large enough
- For a well-posed optimization, N_c should be large and $N_p - N_c$ should be greater than the system settling time
- How do we choose control weighting?
- Some guidelines:
  - Greater control weighting generally means less active input changes
  - Increasing weights indefinitely reduces control activity to zero - switching off feedback
  - For large prediction horizons, it is possible to out-weigh control with error magnitude measure
- Considerations for open-loop unstable plants
  - Large N_p values in standard MPC design may cause performance degradation for open-loop unstable plants

  - Large N_p values can cause ill-conditioning of the numerical computations for open-loop unstable plants

- In general (but not always) it is difficult to obtain acceptable performance for $N_c = 1$

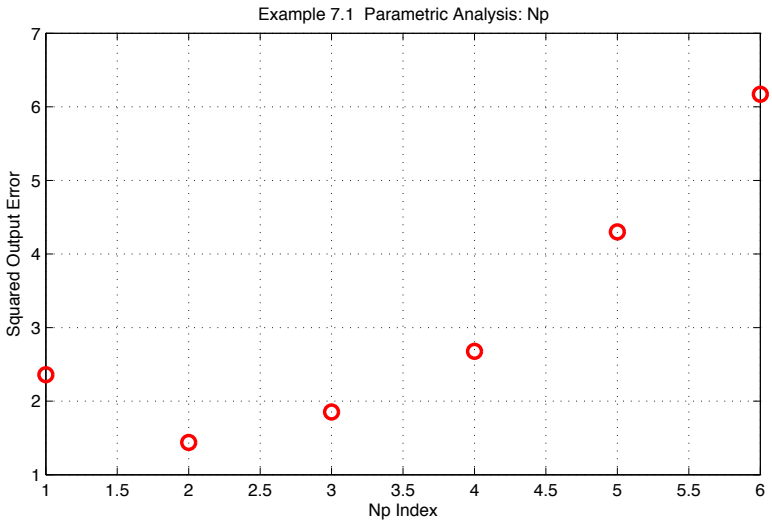
Systematic tuning strategies

- It is often tempting to pursue strategies that seek to 'optimize' output performance in some meaningful way
- One example of this might be to measure the output error for various tuning parameters and compare results
- For the following study, the system of Example 7.1 was used and the sum-of-squares of the output error for $N_p = 20$ was computed and compared for values of $N_c = 1, 2, 3, 5, 10 \text{ and } 20$:



- A clear improvement is seen in moving from $N_c = 1$ to $N_c = 2$. Only marginal improvements are seen with further increases in N_c.

- The next study compares the same error criterion for $N_c = 1$ and different values of N_p.



- This result shows a rather clear minimum for $N_p = 2$

(mostly blank)