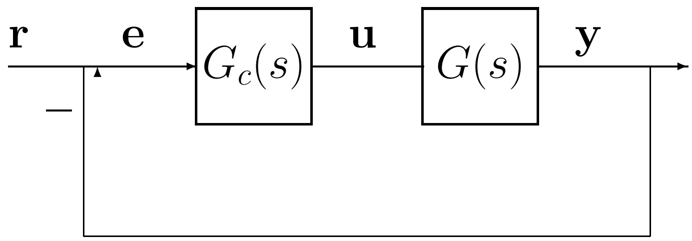
Reference Input - II

- On page 17-5, compensator implemented with a reference command by changing to feedback on $e(t) = r(t) - y(t)$ rather than $-y(t)$



  - So $u = G_c(s)e = G_c(s)(r - y)$, and have $u = -G_c(s)y$ if $r = 0$.
  - Intuitively appealing because it is the **same approach** used for the classical control, but it turns out not to be the best approach.

- Can improve the implementation by using a more general form:

$$\begin{aligned} \dot{x}_c &= A_c x_c + Ly + Gr \\ u &= -Kx_c + \bar{N}r \end{aligned}$$

  - Now explicitly have two inputs to the controller (y and r)
  - $\bar{N}$ performs the same role that we used it for previously.
  - Introduce G as an extra degree of freedom in the problem.

- **First:** if $\bar{N} = 0$ and $G = -L$, then we recover the same implementation used `previously`, since the controller reduces to:

$$\begin{aligned} \dot{x}_c &= A_c x_c + L(y - r) = A_c x_c + B_c(-e) \\ u &= -Kx_c = -C_c x_c \end{aligned}$$

  - So if $G_c(s) = C_c(sI - A_c)^{-1}B_c$, then the controller can be written as $u = G_c(s)e$ (negative signs cancel).

- **Second:** this generalization does not change the closed-loop poles of the system, regardless of the selection of G and $\bar{N}$, since

$$\begin{aligned} \dot{x} &= Ax + Bu \quad , \quad y = Cx \\ \dot{x}_c &= A_c x_c + Ly + Gr \\ u &= -Kx_c + \bar{N}r \end{aligned}$$

$$\Rightarrow \quad \begin{bmatrix} \dot{x} \\ \dot{x}_c \end{bmatrix} = \begin{bmatrix} A & -BK \\ LC & A_c \end{bmatrix} \begin{bmatrix} x \\ x_c \end{bmatrix} + \begin{bmatrix} B\bar{N} \\ G \end{bmatrix} r$$

$$y = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_c \end{bmatrix}$$

  – So the closed-loop poles are the eigenvalues of $\begin{bmatrix} A & -BK \\ LC & A_c \end{bmatrix}$ regardless of the choice of G and $\bar{N}$

  – G and $\bar{N}$ impact the forward path, not the feedback path

- **Third:** given this extra freedom, what is the best way to use it?

  – One good objective is to select G and $\bar{N}$ so that the state estimation error is **independent** of r.

  – With this choice, changes in r do not tend to cause such large transients in $\tilde{x}$

  – Note that for this analysis, take $\tilde{x} = x - x_c$ since $x_c \equiv \hat{x}$

$$\begin{aligned} \dot{\tilde{x}} &= \dot{x} - \dot{x}_c = Ax + Bu - (A_c x_c + Ly + Gr) \\ &= Ax + B(-Kx_c + \bar{N}r) - (\{A - BK - LC\}x_c + LCx + Gr) \end{aligned}$$

$$\begin{aligned}\dot{\tilde{x}} &= Ax + B(\bar{N}r) - (\{A - LC\}x_c + LCx + Gr) \\ &= (A - LC)x + B\bar{N}r - (\{A - LC\}x_c + Gr) \\ &= (A - LC)\tilde{x} + B\bar{N}r - Gr \\ &= (A - LC)\tilde{x} + (B\bar{N} - G)r\end{aligned}$$

- Thus we can eliminate the effect of r on $\tilde{x}$ by setting $G \equiv B\bar{N}$

- **Fourth:** if this generalization does not change the closed-loop poles of the system, then what does it change?
  - The zeros of the y/r transfer function, which are given by:

`general` $$\det \left[\begin{array}{cc|c} sI - A & BK & -B\bar{N} \\ -LC & sI - A_c & -G \\ \hline C & 0 & 0 \end{array}\right] = 0$$

`previous` $$\det \left[\begin{array}{cc|c} sI - A & BK & 0 \\ -LC & sI - A_c & L \\ \hline C & 0 & 0 \end{array}\right] = 0$$

`new` $$\det \left[\begin{array}{cc|c} sI - A & BK & -B\bar{N} \\ -LC & sI - A_c & -B\bar{N} \\ \hline C & 0 & 0 \end{array}\right] = 0$$

- Hard to see how this helps, but consider the scalar case:

$$
\begin{aligned}
\texttt{new} \qquad \det \left[\begin{array}{cc|c} sI - A & BK & -B\bar{N} \\ -LC & sI - A_c & -B\bar{N} \\ \hline C & 0 & 0 \end{array} \right] &= 0 \\
\Rightarrow \quad C(-BKB\bar{N} + (sI - A_c)B\bar{N}) &= 0 \\
-CB\bar{N}(BK - (sI - [A - BK - LC])) &= 0 \\
CB\bar{N}(sI - [A - LC]) &= 0
\end{aligned}
$$

  - So that the zero of the y/r path is the root of $sI - [A - LC] = 0$ which is the pole of the estimator.
  - With this selection of $G = B\bar{N}$ the estimator dynamics are canceled out of the response of the system to a reference command.
  - No such cancelation occurs with the `previous` implementation.

- **Fifth:** select $\bar{N}$ to ensure that the steady-state error is zero.
  - As before, this can be done by selecting $\bar{N}$ so that the DC gain of the closed-loop y/r transfer function is 1.

$$
\left.\frac{y}{r}\right|_{DC} \triangleq \left[\begin{array}{cc} C & 0 \end{array} \right] \left(- \left[\begin{array}{cc} A & -BK \\ LC & A_c \end{array} \right]^{-1} \right) \left[\begin{array}{c} B \\ B \end{array} \right] \bar{N} = 1
$$

- **The `new` implementation of the controller is**

$$
\begin{aligned}
\dot{x}_c &= A_c x_c + Ly + B\bar{N}r \\
u &= -Kx_c + \bar{N}r
\end{aligned}
$$

  - Which has two separate inputs y and r
  - Selection of $\bar{N}$ ensure that the steady-state performance is good
  - The new implementation gives better transient performance.

Figure 13: Example #1: $G(s) = \frac{8 \cdot 14 \cdot 20}{(s+8)(s+14)(s+20)}$.

• Method #1: `previous` implementation.

• Method #2: `previous`, with the reference input scaled to ensure that the DC gain of $y/r|_{DC} = 1$.

• Method #3: `new` implementation with both $G = B\bar{N}$ and $\bar{N}$ selected.

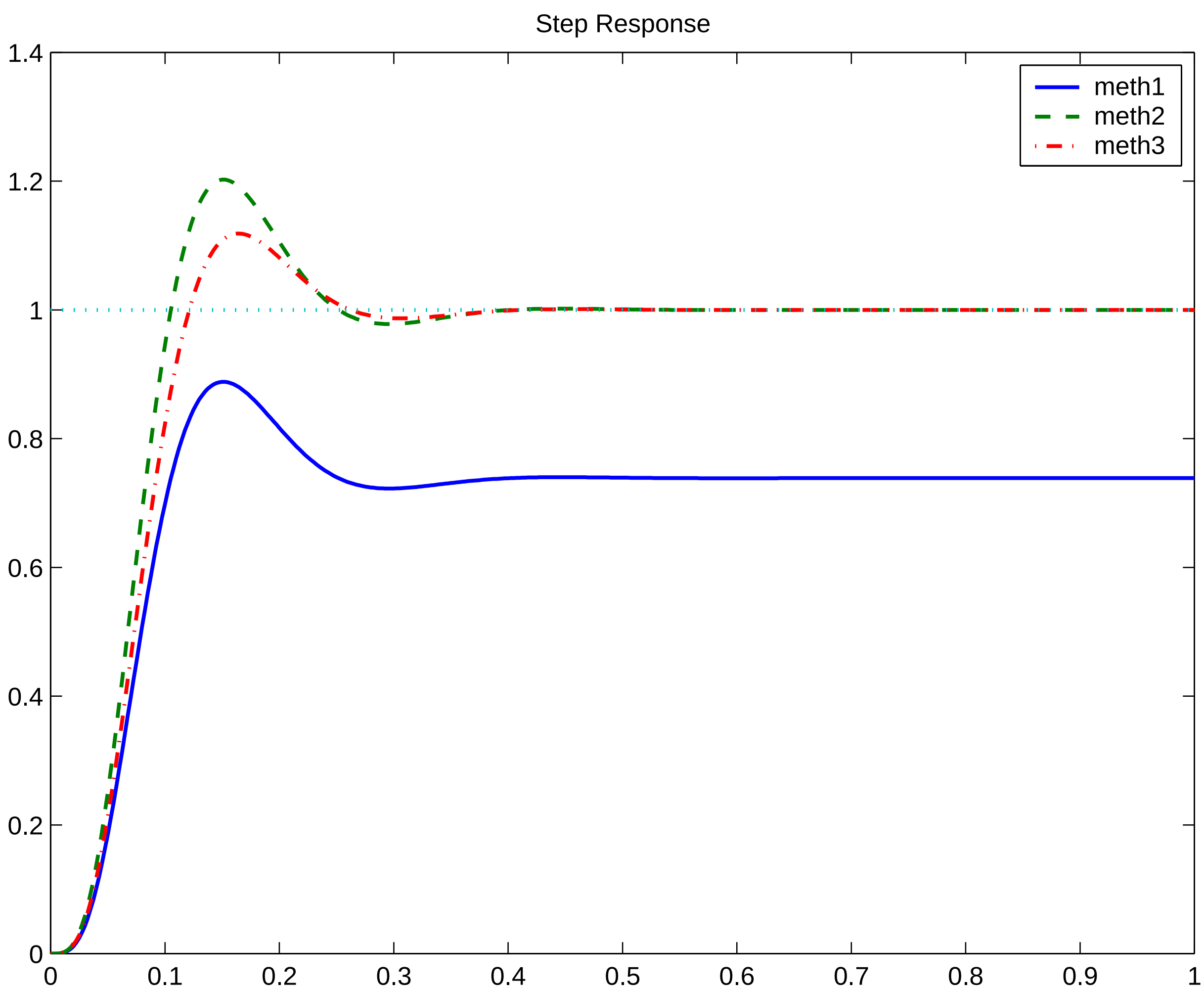


Figure 14: Example #2: $G(s) = \frac{0.94}{s^2 - 0.0297}$.

• Method #1: `previous` implementation.

• Method #2: `previous`, with the reference input scaled to ensure that the DC gain of $y/r|_{DC} = 1$.

• Method #3: `new` implementation with both $G = B\bar{N}$ and $\bar{N}$ selected.

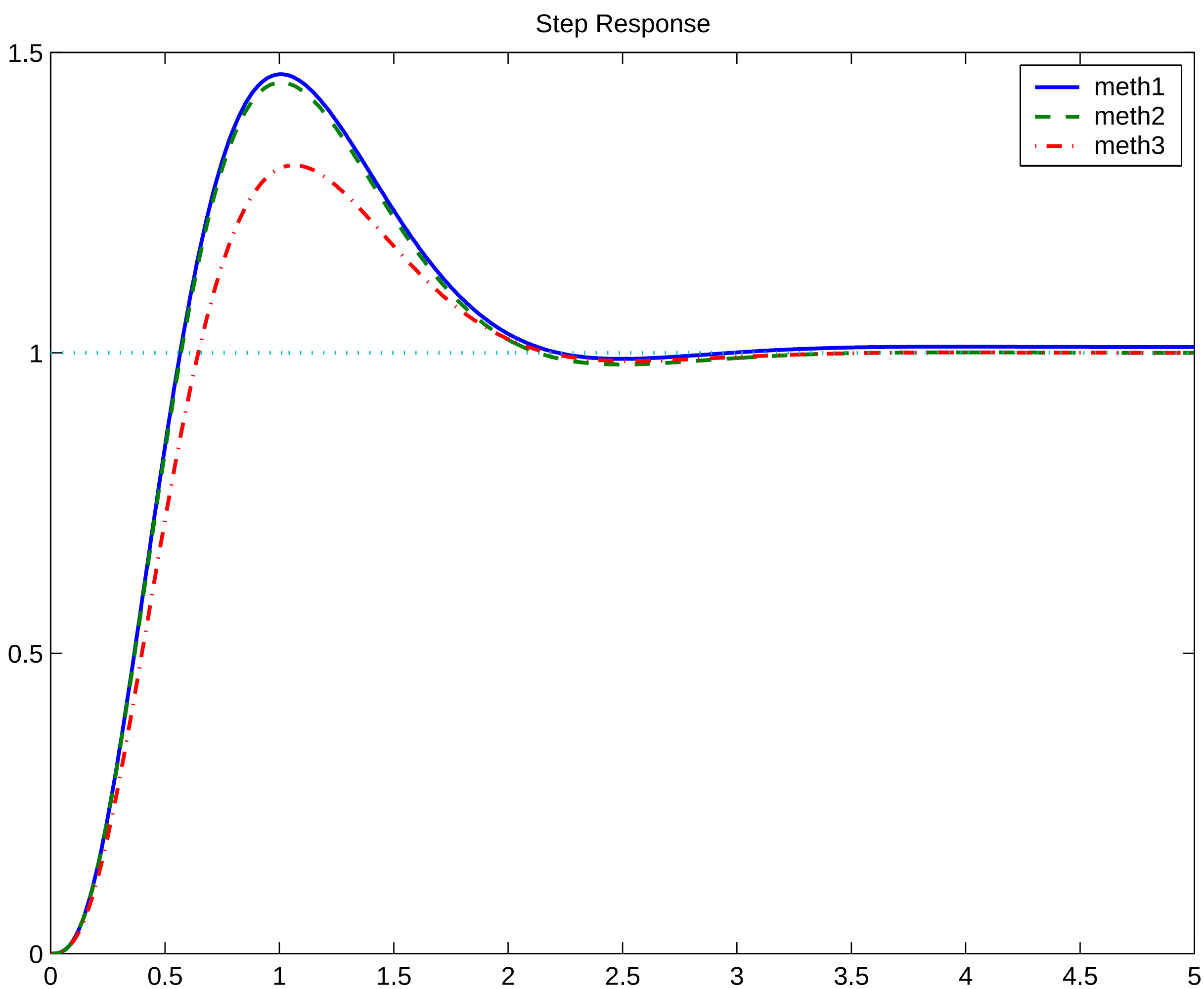


Figure 15: Example #3: $G(s) = \frac{8 \cdot 14 \cdot 20}{(s-8)(s-14)(s-20)}$.

• Method #1: `previous` implementation.

• Method #2: `previous`, with the reference input scaled to ensure that the DC gain of $y/r|_{DC} = 1$.

• Method #3: `new` implementation with both $G = B\bar{N}$ and $\bar{N}$ selected.