ELEC4410

Control System Design

Lecture 4: Affine Parameterisation. PID Revisited, Time Delays and Undesirable Closed Loop Poles

School of Electrical Engineering and Computer Science

The University of Newcastle

Outline

- Revisit PID design using the affine parametrisation.

Outline

- Revisit PID design using the affine parametrisation.
- Control of time delayed plants using the affine parametrisation. (Also showing the connections with the Smith controller.)

Outline

- Revisit PID design using the affine parametrisation.
- Control of time delayed plants using the affine parametrisation. (Also showing the connections with the Smith controller.)
- Use of interpolation constraints to remove undesirable open loop poles.

Outline

- Revisit PID design using the affine parametrisation.
- Control of time delayed plants using the affine parametrisation. (Also showing the connections with the Smith controller.)
- Use of interpolation constraints to remove undesirable open loop poles.

Reference: Control System Design, Goodwin, Graebe & Salgado.

PID Synthesis using the Affine Parametrisation

- We illustrate the ideas by choosing a simple First Order Model:

$$G_o(s) = \frac{K_o}{\nu_o s + 1}$$

PID Synthesis using the Affine Parametrisation

- We illustrate the ideas by choosing a simple First Order Model:

$$G_o(s) = \frac{K_o}{\nu_o s + 1}$$

- We employ the affine synthesis methodology. Since there are no unstable zeros, the model is exactly invertible.

PID Synthesis using the Affine Parametrisation

- We illustrate the ideas by choosing a simple First Order Model:

$$G_o(s) = \frac{K_o}{v_o s + 1}$$

- We employ the affine synthesis methodology. Since there are no unstable zeros, the model is exactly invertible.
- We then choose

$$G_o^i(s) = (G_o(s))^{-1} = \frac{v_o s + 1}{K_o}$$

PID Synthesis using the Affine Parametrisation

- In order for $Q(s)$ to be biproper, $F_Q(s)$ must have relative degree 1, such as

$$F_Q(s) = \frac{1}{\alpha s + 1}$$

PID Synthesis using the Affine Parametrisation

- In order for $Q(s)$ to be biproper, $F_Q(s)$ must have relative degree 1, such as

$$F_Q(s) = \frac{1}{\alpha s + 1}$$

- This implies that our final choice for Q(s) is of the form:

$$Q(s) = F_Q(s)G_o^i(s) = \frac{v_o s + 1}{K_o(\alpha s + 1)}$$

and the controller becomes

$$C(s) = \frac{Q(s)}{1 - Q(s)G_o(s)} = \frac{v_o s + 1}{K_o \alpha s} = \frac{v_o}{K_o \alpha} + \frac{1}{K_o \alpha s}$$

which is a PI controller.

PID Synthesis using the Affine Parametrisation

- With the PI controller parameters found above the nominal complementary sensitivity becomes

$$T_o(s) = Q(s)G_o(s) = F_Q(s) = \frac{1}{\alpha s + 1}$$

where α becomes a tuning parameter.

PID Synthesis using the Affine Parametrisation

- With the PI controller parameters found above the nominal complementary sensitivity becomes

$$T_o(s) = Q(s)G_o(s) = F_Q(s) = \frac{1}{\alpha s + 1}$$

  where α becomes a tuning parameter.

- Choosing α smaller makes the loop faster, whereas a larger value for α slows the loop down.

PID Synthesis using the Affine Parametrisation

- With the PI controller parameters found above the nominal complementary sensitivity becomes

$$T_o(s) = Q(s)G_o(s) = F_Q(s) = \frac{1}{\alpha s + 1}$$

  where α becomes a tuning parameter.

- Choosing α smaller makes the loop faster, whereas a larger value for α slows the loop down.
- We thus see a direct connection between the design variable α and the final closed loop performance.

PID Synthesis using the Affine Parametrisation

- With the PI controller parameters found above the nominal complementary sensitivity becomes

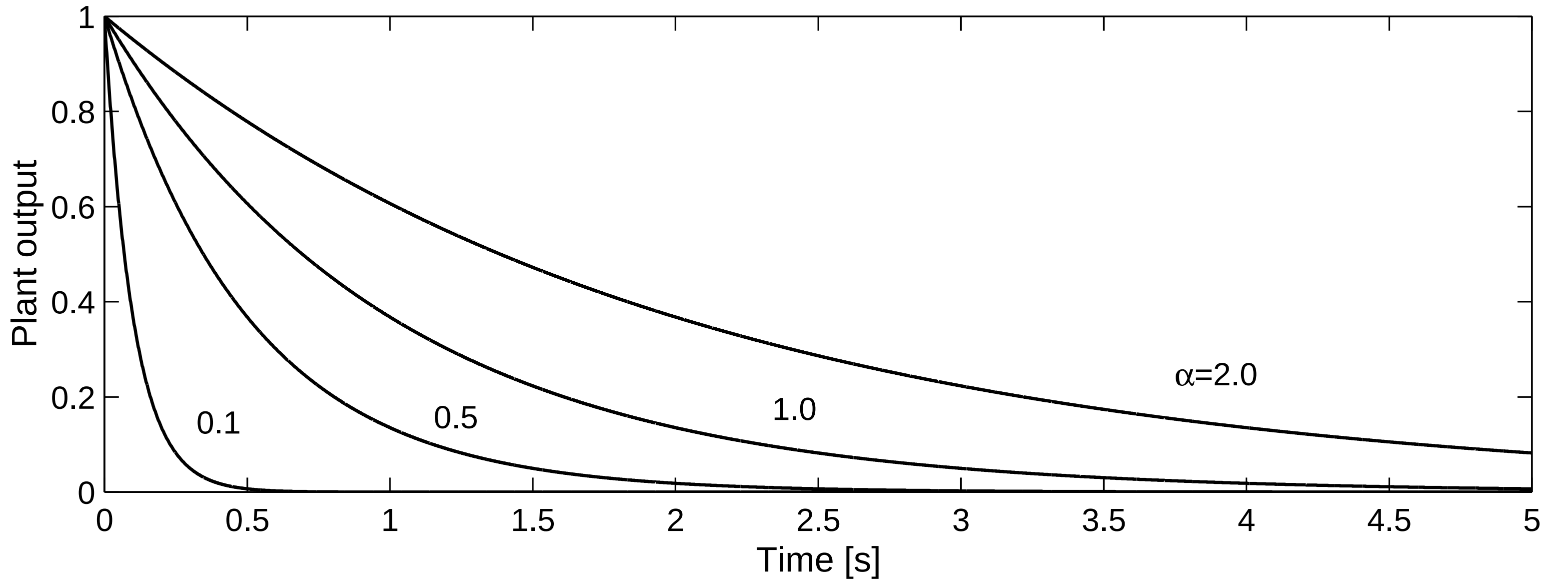
$$T_o(s) = Q(s)G_o(s) = F_Q(s) = \frac{1}{\alpha s + 1}$$

where α becomes a tuning parameter.

- Choosing α smaller makes the loop faster, whereas a larger value for α slows the loop down.
- We thus see a direct connection between the design variable α and the final closed loop performance.
- This is one of the principal advantages of the affine parametrisation methodology.

PID Synthesis using the Affine Parametrisation

Effect of α on output disturbance rejection



PID Synthesis using the Affine Parametrisation

- PI and PID controllers are traditionally tuned in terms of their parameters.

PID Synthesis using the Affine Parametrisation

- PI and PID controllers are traditionally tuned in terms of their parameters.
- However, systematic design, trade-off decisions and deciding whether a PI(D) is sufficient or not, is significantly easier in the model-based affine structure as the PI(D) gains are explicit functions of the desired closed loop response.

PID Synthesis using the Affine Parametrisation

- PI and PID controllers are traditionally tuned in terms of their parameters.
- However, systematic design, trade-off decisions and deciding whether a PI(D) is sufficient or not, is significantly easier in the model-based affine structure as the PI(D) gains are explicit functions of the desired closed loop response.
- Inserting a first order model into the affine structure automatically generates a PI controller.

PID Synthesis using the Affine Parametrisation

- PI and PID controllers are traditionally tuned in terms of their parameters.
- However, systematic design, trade-off decisions and deciding whether a PI(D) is sufficient or not, is significantly easier in the model-based affine structure as the PI(D) gains are explicit functions of the desired closed loop response.
- Inserting a first order model into the affine structure automatically generates a PI controller.
- Inserting a second order model into the Q-structure automatically generates a PID controller.

PID Synthesis using the Affine Parametrisation

- Whether a PI(D) is sufficient for a particular process is directly related to whether or not a first (second) order model can approximate the process well up to the frequencies where performance is limited by other factors such as delays, actuator saturations, sensor noise or fundamentally unknown dynamics.

PID Synthesis using the Affine Parametrisation

- Whether a PI(D) is sufficient for a particular process is directly related to whether or not a first (second) order model can approximate the process well up to the frequencies where performance is limited by other factors such as delays, actuator saturations, sensor noise or fundamentally unknown dynamics.
- The first and second order models are easily obtained from step response models.

PID Synthesis using the Affine Parametrisation

- Whether a PI(D) is sufficient for a particular process is directly related to whether or not a first (second) order model can approximate the process well up to the frequencies where performance is limited by other factors such as delays, actuator saturations, sensor noise or fundamentally unknown dynamics.
- The first and second order models are easily obtained from step response models.
- Using this method, the control engineer works directly in terms of observable process properties (rise time, gain, etc.) and closed loop parameters providing an insightful basis for making trade-off decisions. The PI(D) parameters follow automatically.

PID Synthesis using the Affine Parametrisation

- Whether a PI(D) is sufficient for a particular process is directly related to whether or not a first (second) order model can approximate the process well up to the frequencies where performance is limited by other factors such as delays, actuator saturations, sensor noise or fundamentally unknown dynamics.
- The first and second order models are easily obtained from step response models.
- Using this method, the control engineer works directly in terms of observable process properties (rise time, gain, etc.) and closed loop parameters providing an insightful basis for making trade-off decisions. The PI(D) parameters follow automatically.
- The approach does not preempt the design choice of cancelling or shifting the open-loop poles - both are possible and associated with different trade-offs.

Affine Parametrisation for Systems having Time Delays

- The first method we consider is to replace the time delay with it's Pade approximation given by:

$$e^{-s\tau_o} \approx \frac{2 - s\tau_o}{2 + s\tau_o}$$

Affine Parametrisation for Systems having Time Delays

- The first method we consider is to replace the time delay with it's Pade approximation given by:

$$e^{-s\tau_o} \approx \frac{2 - s\tau_o}{2 + s\tau_o}$$

- The usual technique for affine parametrisation can then be applied to design $Q(s)$.

Affine Parametrisation for Systems having Time Delays

- The first method we consider is to replace the time delay with it's Pade approximation given by:

$$e^{-s\tau_o} \approx \frac{2 - s\tau_o}{2 + s\tau_o}$$

- The usual technique for affine parametrisation can then be applied to design $Q(s)$.
- The Pade approximation usually works well if the time delay is smaller than the dominant time constant of the plant.

Affine Parametrisation for Systems having Time Delays

- The first method we consider is to replace the time delay with it's Pade approximation given by:

$$e^{-s\tau_o} \approx \frac{2 - s\tau_o}{2 + s\tau_o}$$

- The usual technique for affine parametrisation can then be applied to design $Q(s)$.
- The Pade approximation usually works well if the time delay is smaller than the dominant time constant of the plant.
- For the case where the time delay is comparable or larger than the dominant time constant the Smith controller should be considered.

Affine Parametrisation for Systems having Time Delays

- We consider here a special class of linear systems, namely those that be written as

$$G_o(s) = e^{-s\tau_o}\overline{G_o}(s)$$

where $\overline{G_o}(s)$ is a stable rational transfer function.

Affine Parametrisation for Systems having Time Delays

- We consider here a special class of linear systems, namely those that be written as

$$G_o(s) = e^{-s\tau_o}\overline{G_o}(s)$$

where $\overline{G_o}(s)$ is a stable rational transfer function.

- A classical method for dealing with pure time delays as in the above model, was to use a dead-time compensator.

Affine Parametrisation for Systems having Time Delays

- We consider here a special class of linear systems, namely those that be written as

$$G_o(s) = e^{-s\tau_o}\overline{G_o}(s)$$

where $\overline{G_o}(s)$ is a stable rational transfer function.

- A classical method for dealing with pure time delays as in the above model, was to use a dead-time compensator.
- This idea was introduced by Otto Smith in the 1950's.

Affine Parametrisation for Systems having Time Delays

- Smith's controller is based upon two key ideas:

Affine Parametrisation for Systems having Time Delays

- Smith's controller is based upon two key ideas:
  - Affine synthesis and,

Affine Parametrisation for Systems having Time Delays

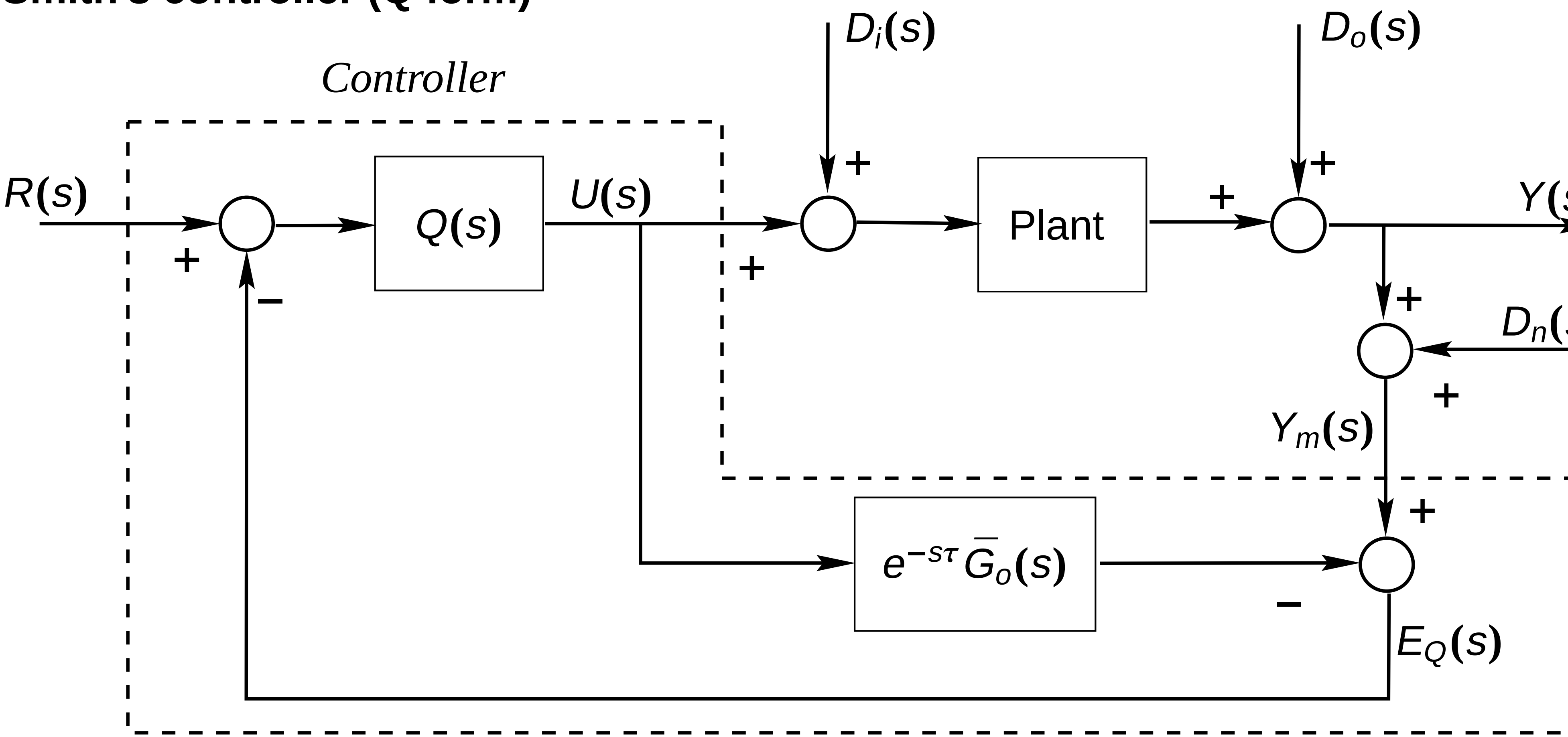
- Smith's controller is based upon two key ideas:
  - Affine synthesis and,
  - the recognition that delay characteristics cannot be inverted.

Affine Parametrisation for Systems having Time Delays

- Smith's controller is based upon two key ideas:
  - Affine synthesis and,
  - the recognition that delay characteristics cannot be inverted.
- The structure of the traditional Smith controller can be obtained from the scheme shown on the next slide, which is a particular case of the general scheme for affine parameterisation given earlier.

Affine Parametrisation for Systems having Time Delays

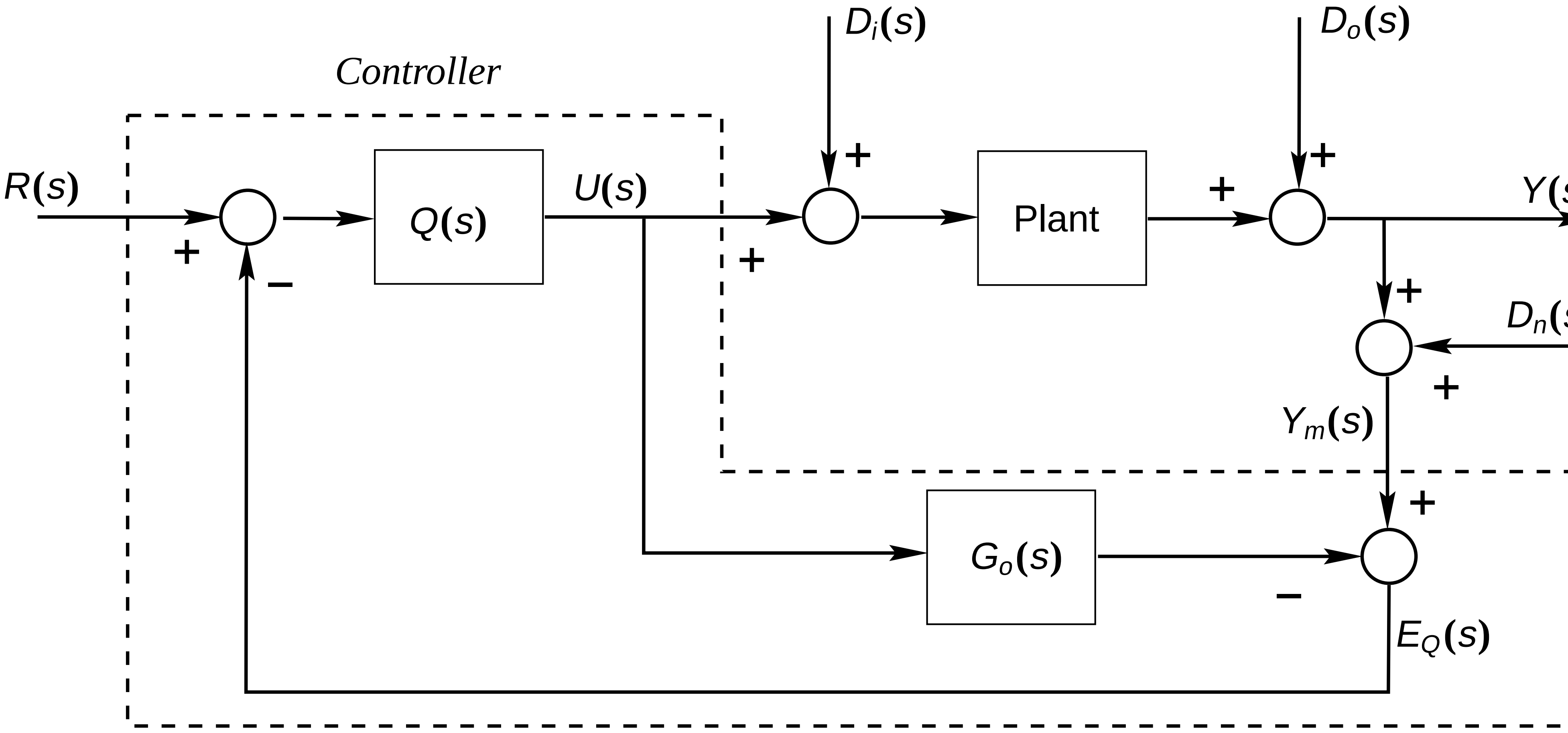
Smith's controller (Q form)



Using earlier results we know the above configuration describes all stabilising controllers. All we need do is choose $Q(s)$ to be a stable proper transfer function.

Affine Parametrisation for Systems having Time Delays

Youla's parametrisation of all stabilising controllers for stable plants



Affine Parametrisation for Systems having Time Delays

- Design of Q(s) for systems with a time delay

Affine Parametrisation for Systems having Time Delays

- Design of Q(s) for systems with a time delay
  - Using the structure shown above, the nominal complementary sensitivity is

$$T_o(s) = e^{-s\tau}\overline{G_o}(s)Q(s)$$

Affine Parametrisation for Systems having Time Delays

- Design of Q(s) for systems with a time delay
  - Using the structure shown above, the nominal complementary sensitivity is

$$T_o(s) = e^{-s\tau}\overline{G_o}(s)Q(s)$$

  - This suggests that $Q(s)$ can be designed considering only the rational part of the model, $\overline{G_o}(s)$.

Affine Parametrisation for Systems having Time Delays

- To carry out the design, the procedures and criteria discussed previously can be used.

Affine Parametrisation for Systems having Time Delays

- To carry out the design, the procedures and criteria discussed previously can be used.
- In particular, we need an approximate (stable, causal and proper) inverse for $G_o(s) = e^{-s\tau}\overline{G_o}(s)$.

Affine Parametrisation for Systems having Time Delays

- To carry out the design, the procedures and criteria discussed previously can be used.
- In particular, we need an approximate (stable, causal and proper) inverse for $G_o(s) = e^{-s\tau}\overline{G_o}(s)$.
- Since the delay has no causal inverse, we seek an approximate inverse for $\overline{G_o}(s)$.

Affine Parametrisation for Systems having Time Delays

- To carry out the design, the procedures and criteria discussed previously can be used.
- In particular, we need an approximate (stable, causal and proper) inverse for $G_o(s) = e^{-s\tau}\overline{G_o}(s)$.
- Since the delay has no causal inverse, we seek an approximate inverse for $\overline{G_o}(s)$.
- This can be achieved directly. Alternatively, one can use the idea of feedback to generate a stable inverse. Thus we might conceive of evaluating $Q(s)$ by

$$Q(s) = \frac{C(s)}{1 + C(s)\overline{G_o}(s)}$$

Affine Parametrisation for Systems having Time Delays

- Note that the form of $Q(s)$ suggested in the previous slide is simply a mechanism for obtaining an approximate inverse for $\overline{G_o}(s)$.

Affine Parametrisation for Systems having Time Delays

- Note that the form of $Q(s)$ suggested in the previous slide is simply a mechanism for obtaining an approximate inverse for $\overline{G_o}(s)$.
- In particular, if $C(s)$ has high gain, then

$$Q(s) = \frac{C(s)}{1 + C(s)\overline{G_o}(s)} \Rightarrow \left(\overline{G_o}(s)\right)^{-1}$$

Affine Parametrisation for Systems having Time Delays

- Note that the form of $Q(s)$ suggested in the previous slide is simply a mechanism for obtaining an approximate inverse for $\overline{G_o}(s)$.
- In particular, if $C(s)$ has high gain, then

$$Q(s) = \frac{C(s)}{1 + C(s)\overline{G_o}(s)} \Rightarrow \left(\overline{G_o}(s)\right)^{-1}$$

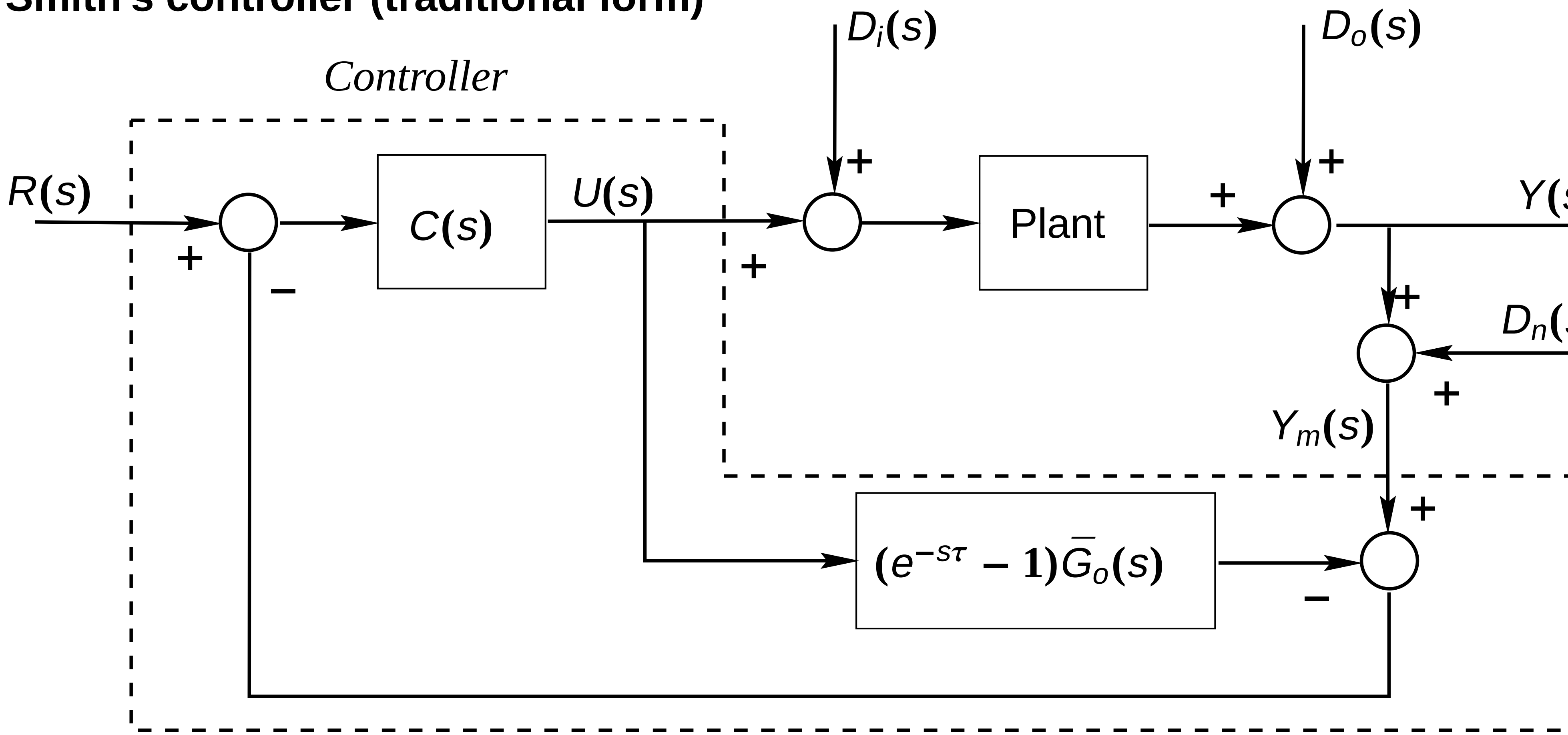
- If we use the above idea to choose $Q(s)$; i.e. put

$$Q(s) = \frac{C(s)}{1 + C(s)\overline{G_o}(s)}$$

then we can redraw the controller as on the next slide.

Affine Parametrisation for Systems having Time Delays

Smith's controller (traditional form)



In this form, we see that the design of $C(s)$ can essentially be based on the nondelayed model. This is precisely the form of the traditional Smith controller.

Further Considerations

- So far we have assumed that the nominal open loop plant model was stable.

Further Considerations

- So far we have assumed that the nominal open loop plant model was stable.
- This meant that all we needed to do was to choose $Q(s)$ to ensure closed loop stability.

Further Considerations

- So far we have assumed that the nominal open loop plant model was stable.
- This meant that all we needed to do was to choose $Q(s)$ to ensure closed loop stability.
- We next examine a methodology, using affine parameterisation, to remove undesirable closed loop poles.

Undesirable Closed Loop Poles

- The idea of the $Q(s)$ parametrisation remains valid since

$$Q(s) = \frac{C(s)}{1 + G_o(s)C(s)}.$$

can always be solved for $Q(s)$ in terms of any $C(s)$.

Undesirable Closed Loop Poles

- The idea of the $Q(s)$ parametrisation remains valid since

$$Q(s) = \frac{C(s)}{1 + G_o(s)C(s)}.$$

can always be solved for $Q(s)$ in terms of any $C(s)$.

- We also recall the following expressions for the sensitivity functions

$$\begin{aligned} T_o(s) &= Q(s)G_o(s) \\ S_o(s) &= 1 - Q(s)G_o(s) \\ S_{io}(s) &= (1 - Q(s)G_o(s))\, G_o(s) \\ S_{uo}(s) &= Q(s) \end{aligned}$$

Undesirable Closed Loop Poles

- Up to this point is has been implicitly assumed that all open loop plant poles were stable and hence could be tolerated in the closed loop input sensitivity function $S_{io}(s)$.

Undesirable Closed Loop Poles

- Up to this point is has been implicitly assumed that all open loop plant poles were stable and hence could be tolerated in the closed loop input sensitivity function $S_{io}(s)$.
- In practice we need to draw a distinction between stable poles and desirable poles.

Undesirable Closed Loop Poles

- Up to this point is has been implicitly assumed that all open loop plant poles were stable and hence could be tolerated in the closed loop input sensitivity function $S_{io}(s)$.
- In practice we need to draw a distinction between stable poles and desirable poles.
- For example, a lightly damped resonant pair might well be stable but is probably undesirable.

Undesirable Closed Loop Poles

- Up to this point is has been implicitly assumed that all open loop plant poles were stable and hence could be tolerated in the closed loop input sensitivity function $S_{io}(s)$.
- In practice we need to draw a distinction between stable poles and desirable poles.
- For example, a lightly damped resonant pair might well be stable but is probably undesirable.
- Say the open loop plant contains some undesirable (including unstable) poles. The only way to remove poles from the complementary sensitivity is to choose $Q(s)$ to contain these poles as zeros.

Undesirable Closed Loop Poles

- This results in cancellation of these poles from the product $Q(s)G_o(s)$ and hence from $S_o(s)$ and $T_o(s)$.

Undesirable Closed Loop Poles

- This results in cancellation of these poles from the product $Q(s)G_o(s)$ and hence from $S_o(s)$ and $T_o(s)$.
- However, the cancelled poles may still appear as poles of the nominal input sensitivity $S_{io}(s)$, depending on the zeros of $1 - Q(s)G_o(s)$, i.e. the zeros of $S_o(s)$.

Undesirable Closed Loop Poles

- This results in cancellation of these poles from the product $Q(s)G_o(s)$ and hence from $S_o(s)$ and $T_o(s)$.
- However, the cancelled poles may still appear as poles of the nominal input sensitivity $S_{io}(s)$, depending on the zeros of $1 - Q(s)G_o(s)$, i.e. the zeros of $S_o(s)$.
- To eliminate these poles from $S_{io}(s)$ we need to also ensure that the offending poles are also zeros of $(1 - Q(s)G_o(s))$.

Undesirable Closed Loop Poles

- This results in cancellation of these poles from the product $Q(s)G_o(s)$ and hence from $S_o(s)$ and $T_o(s)$.
- However, the cancelled poles may still appear as poles of the nominal input sensitivity $S_{io}(s)$, depending on the zeros of $1 - Q(s)G_o(s)$, i.e. the zeros of $S_o(s)$.
- To eliminate these poles from $S_{io}(s)$ we need to also ensure that the offending poles are also zeros of $(1 - Q(s)G_o(s))$.
- The above statements represent a set of additional constraints on $Q(s)$ to ensure closed loop stability.

Undesirable Closed Loop Poles

- This results in cancellation of these poles from the product $Q(s)G_o(s)$ and hence from $S_o(s)$ and $T_o(s)$.
- However, the cancelled poles may still appear as poles of the nominal input sensitivity $S_{io}(s)$, depending on the zeros of $1 - Q(s)G_o(s)$, i.e. the zeros of $S_o(s)$.
- To eliminate these poles from $S_{io}(s)$ we need to also ensure that the offending poles are also zeros of $(1 - Q(s)G_o(s))$.
- The above statements represent a set of additional constraints on $Q(s)$ to ensure closed loop stability.
- The result is summarised in the following Lemma:

Undesirable Closed Loop Poles

Lemma. (*Interpolation constraints to avoid undesirable poles*). Consider a nominal feedback control loop with one d.o.f. and assume $G_o(s)$ contains undesirable (including unstable) open loop poles. We then have

1. Each of the sensitivity functions $T_o(s)$, $S_o(s)$, $S_{io}(s)$ and $S_{uo}(s)$ will have no undesirable poles if and only if: when the controller $C(s)$ is expressed as:
$$Q(s) = \frac{C(s)}{1 + G_o(s)C(s)}.$$
Then $Q(s)$ must satisfy the following (so called) Interpolation constraints:
   (a) $Q(s)$ is proper, stable and has only desirable poles.
   (b) Any undesirable poles of $G_o(s)$ are zeros of $Q(s)$ with, at least, the same multiplicity as $G_o(s)$.
   (c) Any undesirable poles of $G_o(s)$ are zeros of $1 - Q(s)G_o(s)$, with at least the same multiplicity as $G_o(s)$.

Undesirable Closed Loop Poles

Lemma. (cont.)

2. When conditions (b) and (c) are satisfied, then all resultant unstable pole-zero cancellations in $C(s)$ should be performed analytically prior to implementation.

Undesirable Closed Loop Poles

PID Design Revisited

- We return to the design of a PI controller for a first order plant.

Undesirable Closed Loop Poles

PID Design Revisited

- We return to the design of a PI controller for a first order plant.
- We found that design based on cancelling the open loop poles in $G_o(s)$ gave excellent output disturbance rejection.

Undesirable Closed Loop Poles

PID Design Revisited

- We return to the design of a PI controller for a first order plant.
- We found that design based on cancelling the open loop poles in $G_o(s)$ gave excellent output disturbance rejection.
- In chemical processes, however, disturbances are frequently better modelled as occurring at the input to the system.

Undesirable Closed Loop Poles

PID Design Revisited

- We return to the design of a PI controller for a first order plant.
- We found that design based on cancelling the open loop poles in $G_o(s)$ gave excellent output disturbance rejection.
- In chemical processes, however, disturbances are frequently better modelled as occurring at the input to the system.
- We then recall that, the input disturbance response $Y_d(s)$ is given by

$$Y_d(s) = S_{io}(s)D_i(s)$$

$$S_{io}(s) = S_o(s)G_o(s)$$

Undesirable Closed Loop Poles

PID Design Revisited

- Hence, when any plant pole is cancelled in the controller, it remains controllable from the input disturbance, and is still observable at the output.

Undesirable Closed Loop Poles

PID Design Revisited

- Hence, when any plant pole is cancelled in the controller, it remains controllable from the input disturbance, and is still observable at the output.
- Thus the transient component in the input disturbance response will have a mode associated with that pole.

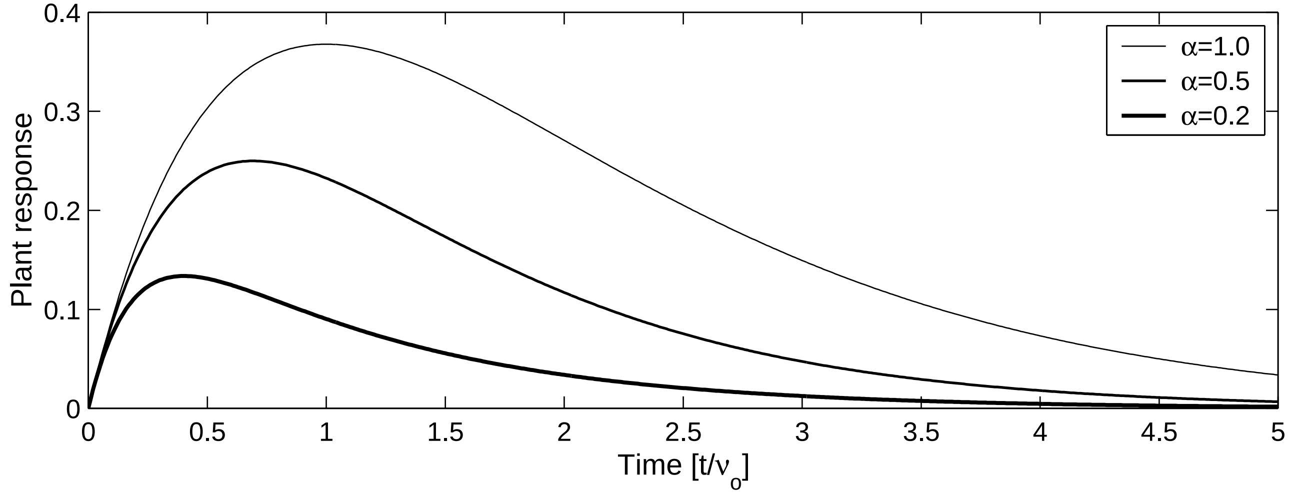
Undesirable Closed Loop Poles

PID Design Revisited

- Hence, when any plant pole is cancelled in the controller, it remains controllable from the input disturbance, and is still observable at the output.
- Thus the transient component in the input disturbance response will have a mode associated with that pole.
- The following slide shows the input disturbance response for the PI controller designed earlier via the affine parametrisation.

Undesirable Closed Loop Poles

Input disturbance rejection with plant pole cancellation, for different values of α.



- Note that changing α changes the magnitude of the response but the slow transient remains since this is dominated by the open loop plant as is evident from:

$$Y_d(s) = S_{io}(s)D_i(s)$$

$$S_{io}(s) = S_o(s)G_o(s)$$

Undesirable Closed Loop Poles

- The origin of this problem is the cancellation of a pole in $G_o(s)$ with a zero in $C(s)$.

Undesirable Closed Loop Poles

- The origin of this problem is the cancellation of a pole in $G_o(s)$ with a zero in $C(s)$.
- As shown earlier, the only way to remove the pole from $S_{io}(s)$ is to choose $F_Q(s)$ in such a way that the offending pole is a zero of $S_o(s) = 1 - Q(s)G_o(s)$, i.e. we require:

$$S_o(-a) = 0 \Longrightarrow T_o(-a) = F_Q(-a) = 1$$

where $a \triangleq \frac{1}{v_o}$

Undesirable Closed Loop Poles

Lemma. Consider the plant model and Youla's parametrisation of all stabilising controllers for stable plants where $Q(s) = (G_o(s))^{-1} F_Q(s)$. Then a PI controller which does not cancel the plant pole, is obtained as

$$C(s) = K_P + \frac{K_I}{s}$$

where

$$K_P = \frac{2\psi_{cl}\omega_{cl}v_o - 1}{K_o}$$

$$K_I = \frac{v_o\omega_{cl}^2}{K_o}$$

and where ψ_{cl} and ω_{cl} are chosen to obtain a closed loop characteristic polynomial given by:

$$A_{cl}(s) = \left(\frac{s}{\omega_{cl}}\right)^2 + 2\psi_{cl}\left(\frac{s}{\omega_{cl}}\right) + 1$$

Undesirable Closed Loop Poles

- The proof of the above result is given in the book Control System Design (Goodwin et. al.).

Undesirable Closed Loop Poles

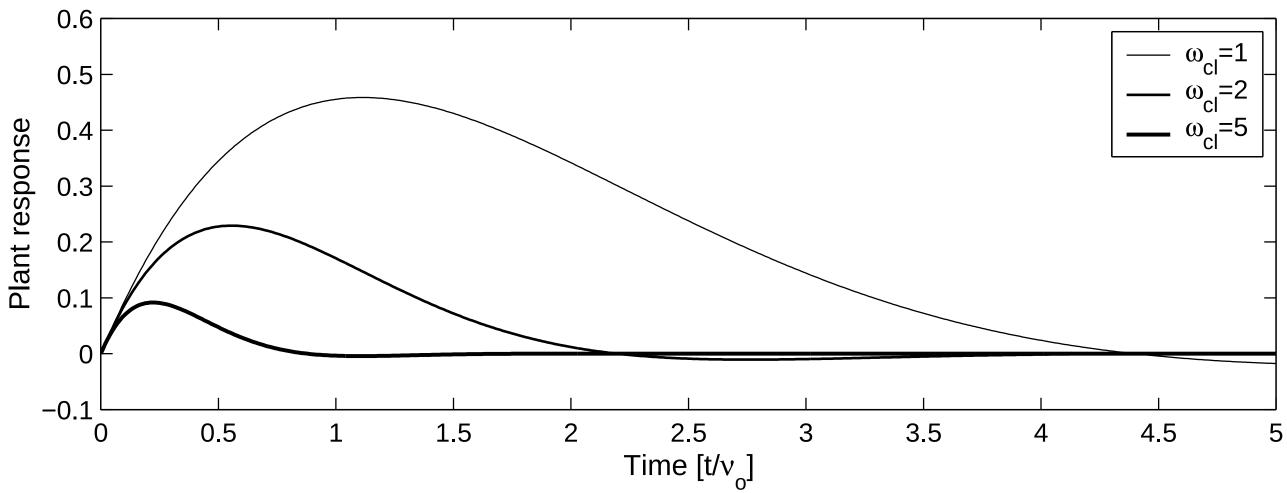
- The proof of the above result is given in the book Control System Design (Goodwin et. al.).
- It suffices to say that the key idea is to ensure that the *slow* open loop pole at $\alpha = \frac{1}{v_o}$ is cancelled in the transfer function $S_o(s) = 1 - G_o(s)Q(s)$; i.e.

$$S_o(-a) = 0 \Longrightarrow T_o(-a) = F_Q(-a) = 1$$

where $a \triangleq \frac{1}{v_o}$

Undesirable Closed Loop Poles

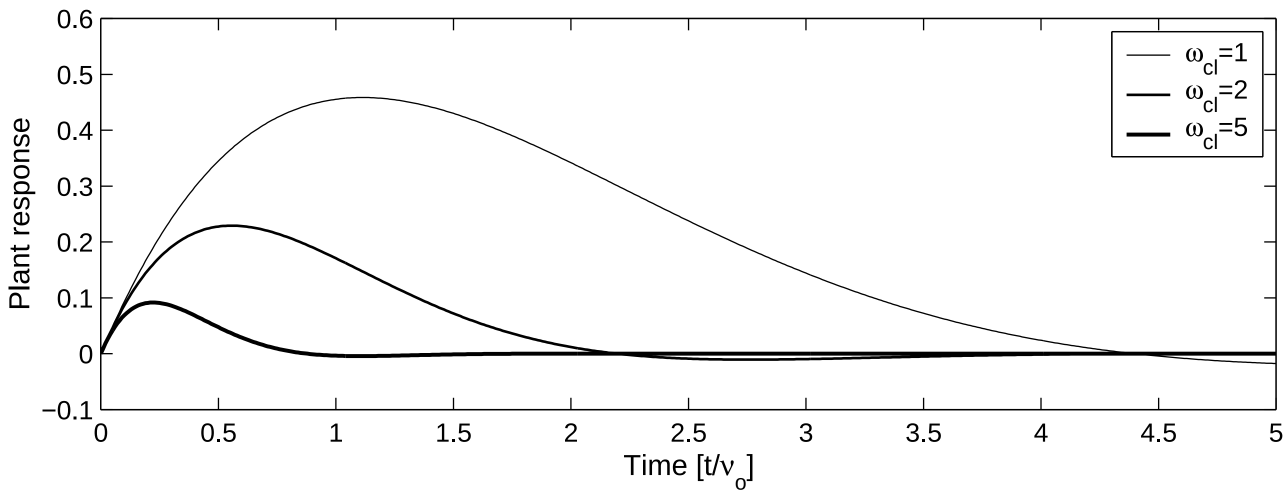
- We repeat the simulation presented earlier where $\alpha = \frac{1}{\nu_o}$ remained in the input disturbance rejection response.
  Input disturbance rejection without plant pole cancellation.



Undesirable Closed Loop Poles

- We repeat the simulation presented earlier where $\alpha = \frac{1}{v_o}$ remained in the input disturbance rejection response.

  Input disturbance rejection without plant pole cancellation.



- We see now that changing the design variable α not only changes the size of the response but it also changes the nature of the transient.

Undesirable Closed Loop Poles

- In the examples given above we went to some trouble to ensure that the poles of all closed loop sensitivity functions (*especially the input disturbance sensitivity,* $S_{io}(s)$) lay in desirable regions of the complex plane.

Undesirable Closed Loop Poles

- In the examples given above we went to some trouble to ensure that the poles of all closed loop sensitivity functions (*especially the input disturbance sensitivity,* $S_{io}(s)$) lay in desirable regions of the complex plane.
- We found that extra interpolation constraints on $Q(s)$ were needed to eliminate undesirable poles from the input sensitivity $S_{io}(s)$.

Undesirable Closed Loop Poles

- In the examples given above we went to some trouble to ensure that the poles of all closed loop sensitivity functions (*especially the input disturbance sensitivity,* $S_{io}(s)$) lay in desirable regions of the complex plane.
- We found that extra interpolation constraints on $Q(s)$ were needed to eliminate undesirable poles from the input sensitivity $S_{io}(s)$.
- In the design examples presented to date we have chosen $Q(s)$ to explicitly account for these interpolation constraints.

Undesirable Closed Loop Poles

- In the examples given above we went to some trouble to ensure that the poles of all closed loop sensitivity functions (*especially the input disturbance sensitivity,* $S_{io}(s)$) lay in desirable regions of the complex plane.
- We found that extra interpolation constraints on $Q(s)$ were needed to eliminate undesirable poles from the input sensitivity $S_{io}(s)$.
- In the design examples presented to date we have chosen $Q(s)$ to explicitly account for these interpolation constraints.
- However, this is a tedious task and one is lead to ask the following question: Can we reparameterise $C(s)$ in such a way that the interpolation constraints given in the Lemma are automatically satisfied?

Summary of results for systems having time-delays

- The key issue is that delays cannot be inverted.

Summary of results for systems having time-delays

- The key issue is that delays cannot be inverted.
- In that sense, delays are related to NMP plant zeros, which cannot be stably inverted either.

Summary of results for systems having time-delays

- The key issue is that delays cannot be inverted.
- In that sense, delays are related to NMP plant zeros, which cannot be stably inverted either.
- A delay of magnitude T, causes similar trade-offs as an unstable zero at $s = T/2$.

Summary of results for systems having time-delays

- The key issue is that delays cannot be inverted.
- In that sense, delays are related to NMP plant zeros, which cannot be stably inverted either.
- A delay of magnitude T, causes similar trade-offs as an unstable zero at $s = T/2$.
- An early controller conceived to deal with the non-invertibility of delays is the famous Smith-predictor.

Summary of results for systems having time-delays

- The key issue is that delays cannot be inverted.
- In that sense, delays are related to NMP plant zeros, which cannot be stably inverted either.
- A delay of magnitude T, causes similar trade-offs as an unstable zero at $s = T/2$.
- An early controller conceived to deal with the non-invertibility of delays is the famous Smith-predictor.
- The trade-offs made in the Smith-predictor can be nicely analysed in the affine structure. Indeed, the structures are very similar. Caution should be exercised, however, not to confuse the generic controller representation of the affine parametrisation with the particular synthesis technique of the Smith-predictor.