Chapter 15

SISO Controller Parameterizations

This chapter treats a novel way of expressing a control transfer function.

We will see that this novel parameterization leads to deep insights into control system design and reinforces, from an alternative perspective, many of the ideas that we have previously studied.

The key feature of the new parameterization is that it renders the closed loop sensitivity functions linear (*or more correctly*, *affine*) in a design variable. We thus call this the *affine parameterization*.

The main ideas presented include

- motivation for the affine parameterization and the idea of open loop inversion
- affine parameterization and Internal Model Control
- affine parameterization and performance specifications
- PID synthesis using the affine parameterization
- control of time delayed plants and affine parameterization. Connections with the Smith controller
- interpolation to remove undesirable open loop poles.

Open Loop Inversion Revisited

Recall that control implicitly and explicitly depends on plant model inversion. This is best seen in the case of open loop control.

In open loop control the input, $U(s)$, is generated from the reference signal $R(s)$, by a transfer function $Q(s)$, i.e. $U(s) = Q(s)R(s)$. This leads to an input-output transfer function of the following form:

$$T_o(s) = G_o(s)Q(s)$$

This simply formula highlights the fundamental importance of inversion, as $T_0(j\omega)$ will be 1 only at those frequencies where $Q(j\omega)$ inverts the model. Note that this is consistent with the prototype solution to the control problem described earlier.

A key point is that $T_0(s) = G_0(s)Q(s)$ is affine in $Q(s)$. On the other hand, with a conventional feedback controller, $C(s)$, the closed loop transfer function has the form

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

The above expression is nonlinear in $C(s)$.

Comparing the two previous equations, we see that the former affine relationship holds if we simply parameterize $C(s)$ in the following fashion:

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

This is the essence of the idea presented here.

Affine Parameterization. The Stable Case

We can invert the relationship given on the previous slide to express $C(s)$ in terms of $Q(s)$ and $G_0(s)$:

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

We will then work with $Q(s)$ as the design variable rather than the original $C(s)$.

Note that the relationship between $C(s)$ and $Q(s)$ is one-to-one and thus there is no loss of generality in working with $Q(s)$.

Stability

Actually a very hard question is the following: Given a stable transfer function $G_0(s)$, describe all controllers, $C(s)$ that stabilize this nominal plant.

However, it turns out that, in the $Q(s)$ form, this question has a very simple answer, namely all that is required is that $Q(s)$ be stable.

This result is formalized in the lemma stated on the next slide.

Lemma 15.1: (*Affine parameterization for stable systems*). Consider a plant having a stable nominal model $G_0(s)$ controlled in a one d.o.f. feedback architecture with a proper controller. Then the nominal loop is internally stable if and only if Q(s) is any stable proper transfer function when the controller transfer function $C(s)$ is parameterized as

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

Proof:

We note that the four sensitivity functions can be written as

$$T_o(s) = Q(s)G_o(s)$$
$$S_o(s) = 1 - Q(s)G_o(s)$$
$$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$$
$$S_{uo}(s) = Q(s)$$

We are for the moment only considering the case when $G_0(s)$ is stable. Then, we see that al of the above transfer functions are stable if and only if $Q(s)$ is stable.

This particular form of the controller, i.e.

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

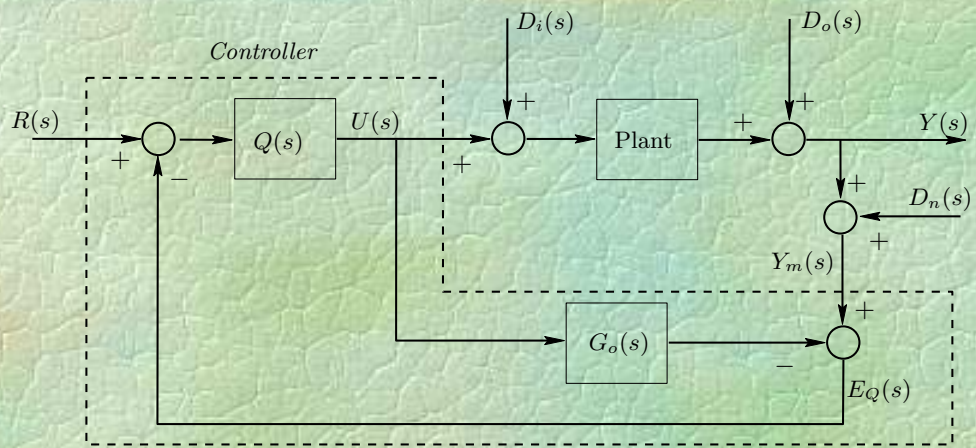
Can be drawn schematically as on the next slide.

The reader is invited to show that this figure is equivalent to

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

$\nabla\nabla\nabla$

Figure15.1: *Youla's parameterization of all stabilizing controllers for stable plants*



Modelling Errors

This description for the controller can also be used to give expressions for the achieved (*or true*) sensitivities when there exists model errors. Specifically, we have

$$S(s) = (1 - Q(s)G_o(s))S_\Delta(s) = S_o(s)S_\Delta(s)$$

$$T(s) = Q(s)G_o(s)(1 + G_\Delta(s))S_\Delta(s)$$

$$S_i(s) = (1 - Q(s)G_o(s))(1 + G_\Delta(s))S_\Delta(s)G_o(s)$$

$$S_u(s) = Q(s)S_\Delta(s)$$

$$S_\Delta(s) = \frac{1}{1 + Q(s)G_\epsilon(s)}$$

where $G_\epsilon(s)$ and $G_\Delta(s)$ are the additive and multiplicative modeling errors, respectively.

Nominal Design

Returning to the nominal design case (*no modelling errors*) we recall that

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

All of these equations are linear (*strictly*, *affine*) in $Q(s)$. This makes design particularly straightforward.

Prototype Control Solution

Specifically, if we look at $T_0(s)$ we recall that

$$T_o(s) = Q(s)G_o(s)$$
$$S_o(s) = 1 - Q(s)G_o(s)$$
$$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$$
$$S_{uo}(s) = Q(s)$$

We also recall that a reasonable design goal is to have $T_0(s)$ near 1 since this implies that the system output exactly follows the reference signal. Thus a prototype controller would seem to be to simply choose

$$Q(s)=[G_0(s)]^{-1}$$

Unfortunately, $[G_0(s)]^{-1}$ is likely to be improper in practice.

Design Considerations

Hence we introduce a small filter $F_Q(s)$ to keep $Q(s)$ proper.

It thus seems that a reasonable choice for $Q(s)$ might be

$$Q(s) = F_Q(s)[G_o(s)]^{-1}$$

where $[G_0(s)]^{-1}$ is the exact inverse of $G_0(s)$. Not unexpectedly, we see that inversion plays a central role in this prototype solution.

Although the design proposed above is a useful starting point it will usually have to be further refined to accommodate more detailed design considerations. In particular, we will investigate the following issues:

1. *non-minimum phase zeros*
2. *model relative degree*
3. *disturbance rejection*
4. *control effort*
5. *robustness*

1. Non Minimum Phase Zeros

Recall that, provided $G_0(s)$ is stable, then $Q(s)$ only needs to be stable to ensure closed loop stability. However, this implies that, if $G_0(s)$ contains NMP zeros, then they cannot be included in $[G_0(s)]^{-1}$. One might therefore think of replacing the previous equation by

$$Q(s) = F_Q(s)G_o^i(s)$$

where $G_0^i(s)$ is a stable approximation to $[G_0(s)]^{-1}$.

For example, if one factors $G_0(s)$ as:

$$G_o(s) = \frac{B_{os}(s)B_{ou}(s)}{A_o(s)}$$

where $B_{0s}(s)$ and $B_{0u}(s)$ are the stable and unstable factors in the numerator, respectivcely, with $B_{0u}(0) = 1$, then a suitable choice for $G_0^i(s)$ would be

$$G_o^i(s) = \frac{A_o(s)}{B_{os}(s)}$$

2. Model Relative Degree

To have a proper controller it is necessary that Q(s) be proper. Thus it is necessary that the shaping filter, $F_Q(s)$, have relative degree at least equal to the relative degree of $[G_0^i(s)]^{-1}$. Conceptually, this can be achieved by including factors of the form $(\tau s+1)^{n_d}$ $(\tau \in \mathbb{R}^+)$ in the denominator.

3. Disturbance Rejection

We recall the following expression for the closed loop sensitivity function in terms of $Q(s)$:

$$T_o(s) = Q(s)G_o(s)$$
$$S_o(s) = 1 - Q(s)G_o(s)$$
$$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$$
$$S_{uo}(s) = Q(s)$$

It would seem that to achieve perfect disturbance rejection at frequency ω_i simply requires that G_0Q be 1 at ω_i. For example, rejection of a d.c. disturbance requires

$$G_0(0)Q(0) = 1.$$

All Stabilizing Controllers to give constant disturbance rejection

Once we have found one value of $Q(s)$ (*say we call it* $Q_a(s)$) that satisfies $G_0(0)Q_a(0) = 1$, then all possible controllers giving constant disturbance rejection can be described as shown on the next slide.

Consider a stable model $G_0(s)$ with input and/or output) disturbance at zero frequency. Then, a one d.o.f. control loop, giving zero steady state tracking error, is stable if and only if the controller $C(s)$ can be expressed in the affine form where $Q(s)$ satisfies

$$Q(s) = s\overline{Q}(s) + [G_o(0)]^{-1}Q_a(s)$$

where $\overline{Q}(s)$ is any stable transfer function, and $Q_a(s)$ is any stable transfer function which satisfies $Q_a(0) = 1$.

Generalization

The above idea can be readily extended to cover rejection of disturbances at any frequency $\omega_{I.}$

4. Control Effort

We see that if we achieve $S_0 = 0$ at a given frequency, i.e. $QG_0 = 1$, then we have infinite gain in the controller C at the same frequency. For example, say the plant is minimum phase, then we could choose $G_0^i(s) = G_0^{-1}(s)$. However, we would then have

$$C(s) = \frac{F_Q(s)G_o^i(s)}{1 - F_Q(s)}$$

By way of illustration, say that we choose

$$F_Q(s) = \frac{1}{(\tau s + 1)^r}$$

then, the high frequency gain of the controller, K_{hfc}, and the high frequency gain of the model, K_{hfg}, are related by

$$K_{hfc} = \frac{1}{\tau^r K_{hfg}}$$

Thus, as we make $F_Q(s)$ faster, i.e. τ becomes smaller, we see that K_{hfc} increases. This, in turn, implies that the control energy will increase. This consequence can be appreciated from the fact that, under the assumption that $G_0(s)$ is minimum phase and stable, we have that

$$S_{uo}(s) = Q(s) = \frac{[G_o(s)]^{-1}}{(\tau s + 1)^r}$$

5. Robustness

Finally, we turn to the issue of robustness in choosing Q(s). We recall from earlier chapters that a fundamental result is that, in order to ensure robustness, the closed loop bandwidth should be such that the frequency response $|T_0(j\omega)|$ rolls off before the effects of modelling errors become significant.

Thus, in the framework of the affine parameterization under discussion here, the robustness requirement can be satisfied if $F_Q(s)$ reduces the gain of $T_0(j\omega)$ at high frequencies. This is usually achieved by including appropriate poles in $F_Q(s)$.

Choice of Q. Summary for the case of Stable Open Loop Poles

We have seen that a prototype choice for $Q(s)$ is simply the inverse of the open loop plant transfer function $G_0(s)$. However, this *ideal* solution needs to be modified in practice to account for the following:

- ***Non-minimum phase zeros.*** Internal stability precludes the cancellation of these zeros. They must therefore appear in $T_0(s)$. This implies that the gain of $Q(s)$ must be reduced at these frequencies for robustness reasons.

- ***Relative degree.*** Excess poles in the model must necessarily appear as a lower bound for the relative degree of $T_0(s)$, since $Q(s)$ must be proper to ensure that the controller $C(s)$ is proper.
- ***Disturbance trade-offs.*** Whenever we roll T_0 off to satisfy measurement noise rejection, we necessarily increase sensitivity to output disturbances at that frequency. Also, slow open loop poles must either appear as poles of $S_{i0}(s)$ or as zeros of $S_0(s)$, and in either case there is a performance penalty.

- ***Control energy.*** All plants are typically low pass. Hence, any attempt to make $Q(s)$ close to the model inverse necessarily gives a high pass transfer function from $D_0(s)$ to $U(s)$. This will lead to large input signals and may lead to controller saturation.
- ***Robustness.*** Modeling errors usually become significant at high frequencies, and hence to retain robustness it is necessary to attenuate T_0, and hence Q, at these frequencies.

PID Design Revisited

We have previously devoted Chapter 6 to Classical PID design. We will revisit this problem here using the affine parameterization. We will see that this makes certain aspects of the design very straightforward.

PID Synthesis using the Affine Parameterization

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

$$G_0(s) = \frac{K_0}{\nu_0 s + 1}$$

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

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

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:

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

and the controller becomes

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

which is a PI controller.

Note that the above design is very straightforward. Properties of the design are discussed below.

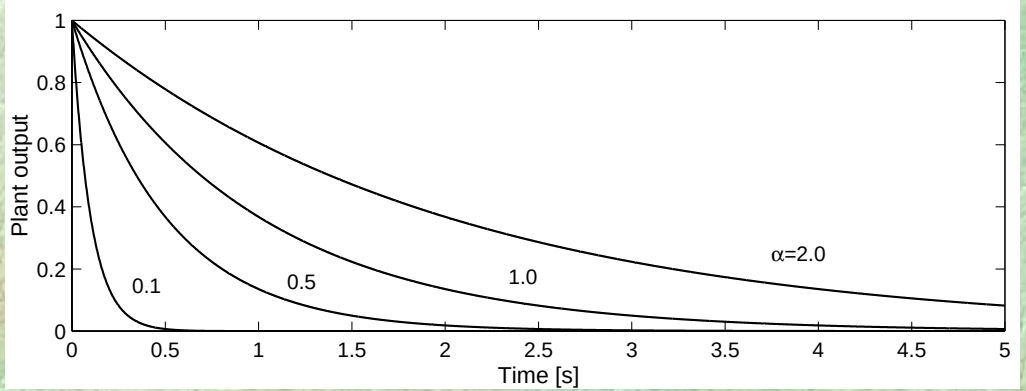
With the PI controller parameters found above the nominal complementary sensitivity becomes

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

where α becomes a tuning parameter. Choosing α smaller makes the loop faster, whereas a larger value for α slows the loop down (*see the next slide*).

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 parameterization methodology.

Figure 15.2: *Effect of α on output disturbance rejection*



PID Design for Second Order Models

The book shows how the above ideas can be readily extended to second order models.

We will not go into details here. Instead, we consider a related topic - namely how to deal with pure time delays.

Affine Parameterization for Systems having Time Delays

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

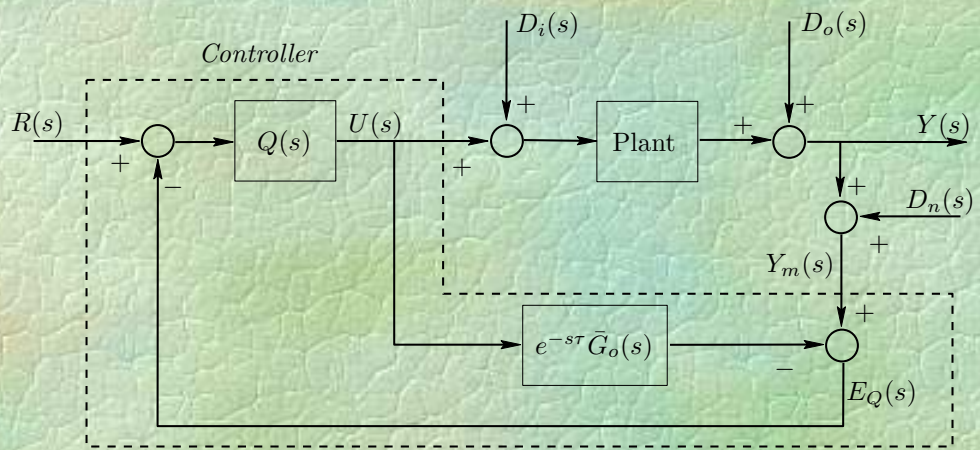
$$G_o(s) = e^{-s\tau}\overline{G}_o(s)$$

where $\overline{G}_0(s)$ 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. Here we give this a modern interpretation via the affine parameterization.

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 in Figure 15.6, which is a particular case of the general scheme given earlier in Figure 15.1.

Figure 15.5: *Smith's controller (Q form)*



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

Design of $Q(s)$ for Delayed Systems

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, $G_0(s)$.

To carry out the design, the procedures and criteria discussed in the previous sections can be used. In particular, we need an approximate (stable, causal and proper) inverse for $G_0(s) = e^{-s\tau}\overline{G}_0(s)$. Since the delay has no causal inverse, we seek an approximate inverse for $\overline{G}_0(s)$. 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

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

Note that the form of $Q(s)$ suggested in the previous slide is simply a mechanism for obtaining an approximate inverse for $\overline{G}_0(s)$. In particular, if $C(s)$ has high gain, then

$$Q(s) = \frac{C(s)}{1 + C(s)\overline{G}_0(s)} \rightarrow \left[\overline{G}_0(s)\right]^{-1}$$

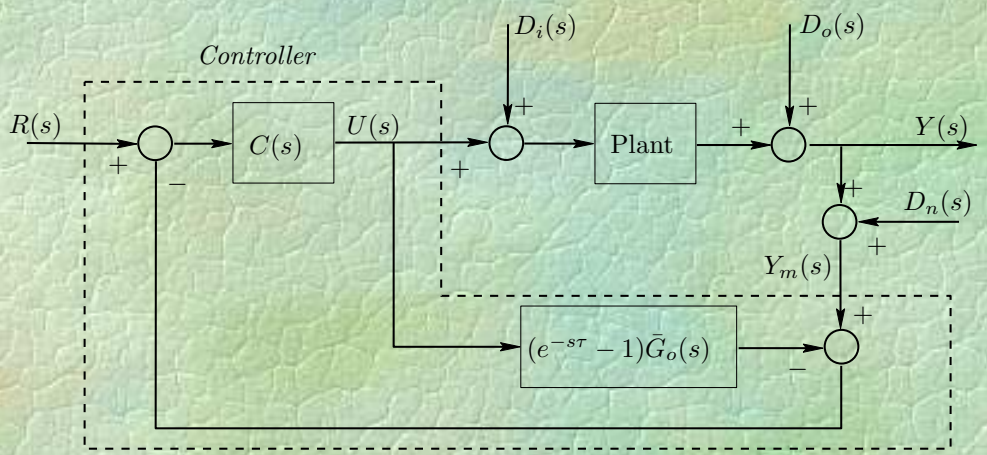
Redrawing the Controller

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.

Figure 15.6: *Smith's controller (traditional form)*



In this form, we see that the design of $C(s)$ can essentially be based on the nondelayed model $\overline{G}_0(s)$. This is precisely the form shown earlier in Figure 7.1 of Section 7.4 of Chapter 7. We ask the reader to review the earlier design described in Chapter 7.

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 consider cases when $G_0(s)$ is not necessarily open loop stable.

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

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

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

We also recall the following expressions for the sensitivity functions

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

In the case when $G_0(s)$ is not open loop stable, then we see that having $Q(s)$ stable is a *necessary* condition for stability but is not sufficient.

Clearly to deal with open loop unstable models we will need to impose additional restrictions of $Q(s)$. This is the topic that we next consider.

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 $S_{i0}(s)$. 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.

This results in cancellation of these poles from the product $Q(s)G_0(s)$ and hence from $S_0(s)$ and $T_0(s)$. However, the cancelled poles may still appear as poles of the nominal input sensivitity $S_{i0}(s)$, depending on the zeros of $1 - Q(s)G_0(s)$, i.e. the zeros of $S_0(s)$. To eliminate these poles from $S_{i0}(s)$ we need to also ensure that the offending poles are also zeros of $[1 - Q(s)G_0(s)]$.

The above represent a set of additional constraints on $Q(s)$ to ensure closed loop stability. The result is summarized in the following lemma:

Lemma 15.4: (*Interpolation constraints to avoid undesirable poles*).

Consider a nominal feedback control loop with one d.o.f. and assume $G_0(s)$ contains undesirable (including unstable) open loop poles. We then have

a) Each of the sensitivity functions $T_0(s)$, $S_0(s)$, $S_{i0}(s)$ and $S_{u0}(s)$ will have *no* undesirable poles if and only if: when the controller $C(s)$ is expressed as

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

Then $Q(s)$ must satisfy the following (*so called*) *Interpolation constraints*:

(*i*) *$Q(s)$ is proper stable and has only desirable poles.*

(*ii*) *Any undesirable poles of $G_0(s)$ are zeros of $Q(s)$ with, at least, the same multiplicity as $G_0(s)$.*

(*iii*) *Any undesirable poles of $G_0(s)$ are zeros of 1 - $Q(s)G_0(s)$, with at least the same multiplicity as $G_0(s)$.*

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

PID Design Revisited

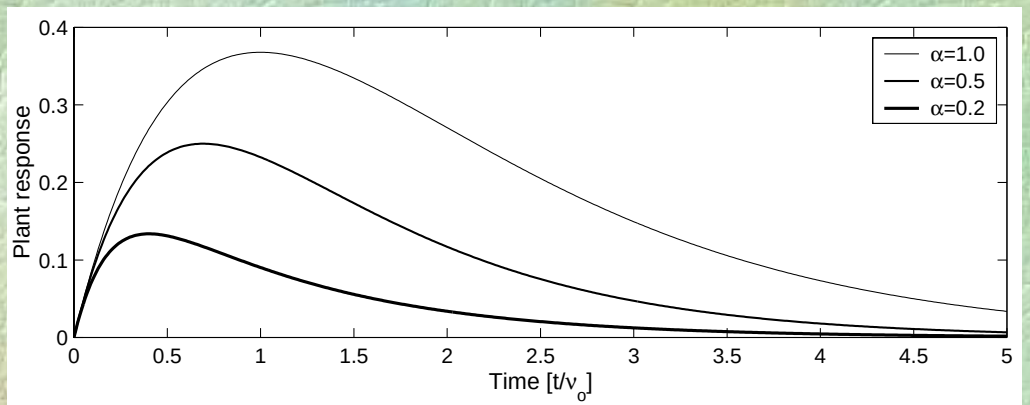
We return to the design of a PI controller for a first order plant. We found that the design of subsection 15.4.2 (based on canceling the open loop poles in $C(s)$) gave excellent output disturbance rejection. In chemical processes, however, disturbances are frequently better modeled as occurring at the input to the system. We then recall that, the input disturbance response $Y_d(s)$ is given by

$$\begin{aligned} Y_d(s) &= S_{io}(s)D_i(s) \\ S_{io}(s) &= S_o(s)G_o(s) \end{aligned}$$

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 parameterization.

Figure 15.7: *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

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

The origin of this problem is the cancellation of a pole in $G_0(s)$ with a zero in $C(s)$. As shown earlier, the only way to remove the pole from $S_{i0}(s)$ is to choose $F_Q(s)$ in such a way that the offending pole is a zero of $S_0(s) = 1 - Q(s)G_0(s)$, i.e. we require

$$S_o(-a) = 0 \Longrightarrow T_o(-a) = F_Q(-a) = 1 \qquad \text{where} \qquad a \triangleq \frac{1}{\nu_o}$$

The key idea is captured in the following lemma:

Lemma 15.5:

Consider the plant model and the control scheme shown in Figure 15.1 where $Q(s) = |G_0(s)]^{-1}F_Q(s)$. Then a PI controller which does not cancel the plant pole, is obtained as

$$C(s) = K_P + \frac{K_I}{s} \qquad \text{with} \qquad K_P = \frac{2\psi_{cl}\omega_{cl}\nu_o - 1}{K_o}; \qquad K_I = \frac{\nu_o}{K_o}\omega_{cl}^2$$

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

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

The proof of the above result is given in the book and will not be repeated here.

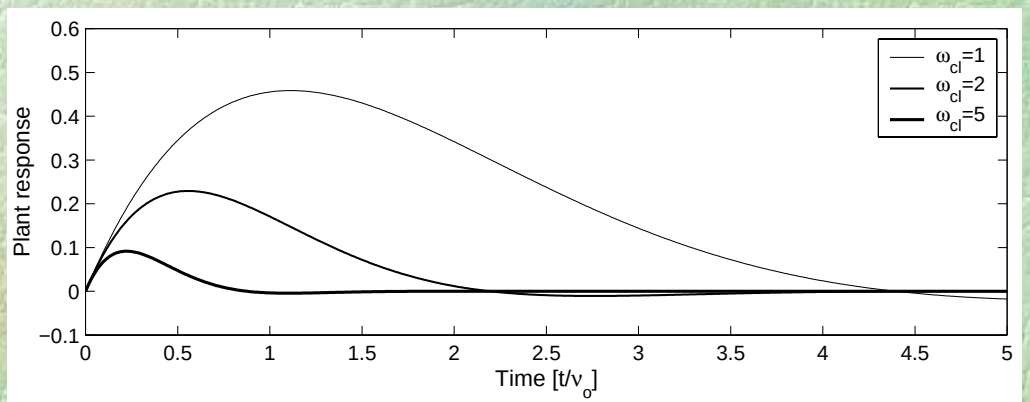
It suffices to say that the key idea is to ensure that the *slow* open loop pole at $\alpha = 1/\nu_0$ is cancelled in the transfer function $S_0(s) = 1 - G_0(s)Q(s)$; i.e.

$$S_o(-a) = 0 \Longrightarrow T_o(-a) = F_Q(-a) = 1 \qquad \text{where} \qquad a \stackrel{\triangle}{=} \frac{1}{\nu_o}$$

We repeat the simulation presented earlier where $\alpha = {}^{1}/_{v_0}$ remained in the input disturbance rejection response.

The new results are presented on the next slide.

Figure 15.8: *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.

(*Compare Figure 15.8 with Figure 15.7*).

Affine Parameterization: The Unstable Open Loop Case

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_{i0}) lay in desirable regions of the complex plane. In this section, we will simplify this procedure by considering a general design problem in which the open loop plant can have one (*or many*) poles in undesirable regions.

We found that extra interpolation constraints on $Q(s)$ were needed to eliminate undesirable poles from the input sensitivity $S_{i0}(s)$. 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 reparameterize $C(s)$ in such a way that the interpolation constraints given in Lemma 15.4 are automatically satisfied? The answer is yes and the solution is described in the following lemma.

Affine parameterization undesirable open loop poles

Lemma 15.6: Consider a one d.o.f. control loop for the plant with nominal model $G_0(s) = \frac{B_0(s)}{A_0(s)}$. We assume that $B_0(s)$ and $A_0(s)$ are coprime polynomials and that $G_0(s)$ may contain undesirable poles (*including unstable poles*).

Then the nominal closed loop will be internally stable and all sensitivity functions will contain only desirable poles if and only if $C(s)$ is parameterized by

$$C(s) = \frac{\dfrac{P(s)}{E(s)} + Q_u(s)\dfrac{A_o(s)}{E(s)}}{\dfrac{L(s)}{E(s)} - Q_u(s)\dfrac{B_o(s)}{E(s)}}$$

where

(*a*) Q_u(s) is a proper stable transfer function having desirable poles.

(*b*) $P(s)$ and $L(s)$ are polynomials satisfying the following pole assignment equation

$$A_o(s)L(s) + B_o(s)P(s) = E(s)F(s)$$

where $E(s)$ and $F(s)$ are polynomials of suitable degree which have zeros lying in the desirable region of the complex plane, but they are otherwise arbitrary.

This parameterization leads to the following parameterized version of the nominal sensitivities:

$$S_o(s) = \frac{A_o(s)L(s)}{E(s)F(s)} - Q_u(s)\frac{B_o(s)A_o(s)}{E(s)F(s)}$$

$$T_o(s) = \frac{B_o(s)P(s)}{E(s)F(s)} + Q_u(s)\frac{B_o(s)A_o(s)}{E(s)F(s)}$$

$$S_{io}(s) = \frac{B_o(s)L(s)}{E(s)F(s)} - Q_u(s)\frac{(B_o(s))^2}{E(s)F(s)}$$

$$S_{uo}(s) = \frac{A_o(s)P(s)}{E(s)F(s)} + Q_u(s)\frac{(A_o(s))^2}{E(s)F(s)}$$

It is readily verified that these are all stable as required.

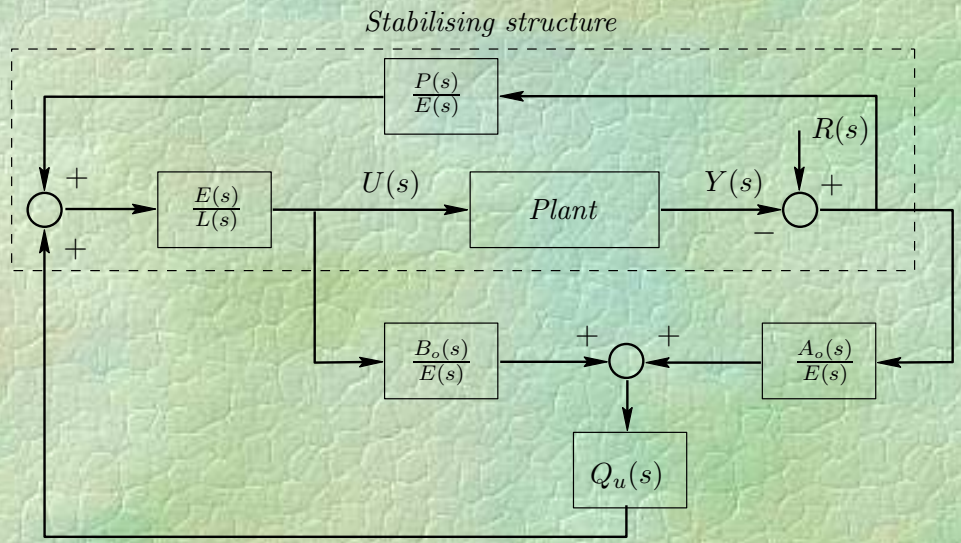
Actually, the result given in Lemma 15.6 simply provides an automatic way of parameterizing $Q(s)$ so that the interpolation constraints are automatically satisfied. Indeed, we find that the original $Q(s)$ is now constrained to the form:

$$Q(s) = \frac{A_o(s)}{F(s)}\left[\frac{P(s)}{E(s)} + Q_u(s)\frac{A_o(s)}{E(s)}\right]$$

where $Q_u(s)$ has desirable poles. It is then verified that this form for $Q(s)$ automatically ensures that the interpolation constraints (*i*) to (*iii*) of Lemma 15.4 hold.

The controller parameterization developed above can also be described in block diagram form. The equation for $C(s)$ directly implies that the controller is as in Figure 15.9 (*next slide*).

Figure 15.9: *Q parameterization for unstable plants*



The above parameterization of all stabilizing controllers for unstable systems, raises the following question:

"If we have an unstable open loop plant, why not simply apply pre-stabilizing feedback and then use the parameterization for the stable system so obtained?"

This is essentially the idea described in the above result as we next show. (*It will turn out that there is a subtle difference*).

We thus to examine the class of all stabilizing controllers for a pre stabilized plant. Thus, consider the configuration shown below.

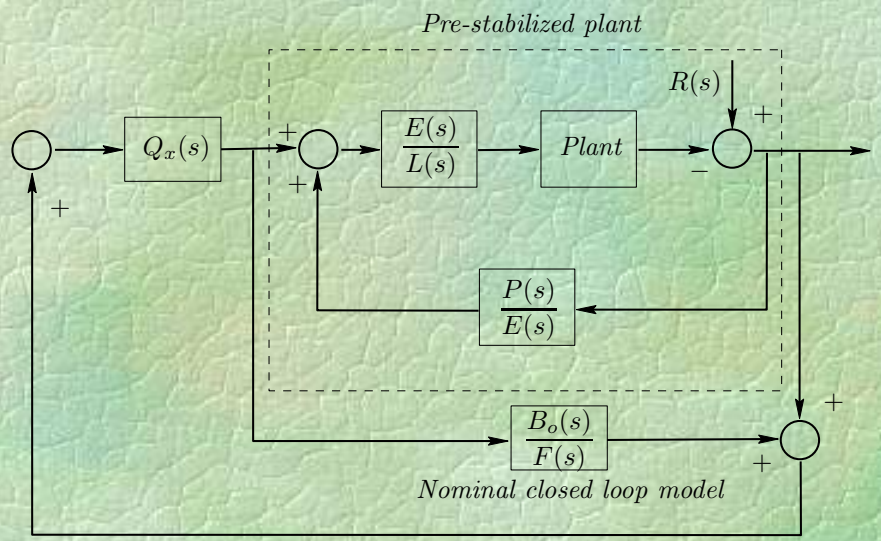


Figure 15.10: *Q interpretation for a pre-stabilized plant*

A simple calculation shows that the equivalent unity feedback controller is

$$\overline{C}(s) = \frac{Q_x(s)E(s)F(s) + P(s)(F(s) - Q_x(s)B_o(s))}{L(s)(F(s) - Q_x(s)B_o(s))}$$

Using the expression $A_0(s)L(s) + B_0(s)P(s) = E(s)F(s)$, the above expression can be simplified to

$$\overline{C}(s) = \frac{Q_x(s)A_o(s)L(s) + P(s)F(s)}{L(s)F(s) - L(s)Q_x(s)B_o(s)}$$

This is very close to the parameterization of all stabilizing controllers for a not necessarily stable open loop plant. The exact connection is described in the following result:

Lemma 15.7: Connecting pre-stabilization with the affine parameterization.

Consider the control structures shown in Figures 15.9 and 15.10

(*i*) Whenever $Q_x(s)$ is stable, then Figure 15.10 can be redrawn as in Figure 15.9 where $Q_u(s)$ takes the particular value

$$Q_u(s) = \frac{Q_x(s)L(s)}{F(s)}$$

(*ii*) Under the mildly restrictive condition that $\frac{Q_u(s)}{L(s)}$ is stable, then Figure 15.9 can be redrawn as in Figure 15.10 where $Q_x(s)$ takes the particular value

$$Q_x(s) = \frac{F(s)Q_u(s)}{L(s)}$$

Part (*i*) of the above result is unsurprising, since the loop in Figure 15.10 is clearly stable for $Q_x(s)$ stable, and hence by Lemma 15.6 the controller can be expressed as in Figure 15.9 for some stable $Q_u(s)$. The converse given in part (*ii*) is more interesting since it shows that there exist structures of the type shown in Figure 15.9 which cannot be expressed as in Figure 15.10.

Summary

- The previous part of the book established that closed loop properties are interlocked in a network of trade offs. Hence, tuning for one property automatically impacts on other properties. This necessitates an understanding of the interrelations and conscious trade-off decisions.
- The fundamental laws of trade-off presented in previous chapters allow one to both identify unachievable specifications as well as to establish where further effort is warranted or wasted.

- However, when pushing a design maximally towards a subtle trade-off, the earlier formulation of the fundamental laws falls short because it is difficult to push the performance of a design by tuning in terms of controller numerator and denominator: The impact on the trade-off determining sensitivity-poles and zeros is very nonlinear, complex and subtle.

- This shortcoming raises the need for an alternative controller representation that
  - allows one to design more explicitly in terms of the quantities of interest (the sensitivities),
  - makes stability explicit, and
  - makes the impact of the controller on the trade-offs explicit.
- This need is met by the affine parameterization, also known as *Youla parameterization.*

- Summary of results for stable systems:
  - $C = Q(1-QG_0)^{-1}$, where the design is carried out by designing the transfer function Q.
  - Nominal sensitivities:

$$T_o = QG_o$$
$$S_o = 1 - QG_o$$
$$S_{io} = (1 - QG_o)\,G_o$$
$$S_{uo} = Q$$

  - Achieved sensitivities:

$$S_\Delta = \frac{1}{1 + QG_oG_\Delta} = \frac{1}{1 + QG_\epsilon}$$
$$T = QGS_\Delta$$
$$S = S_oS_\Delta$$
$$S_i = GS_oS_\Delta$$
$$S_u = QS_\Delta$$

- Observe the following advantages of the affine parameterization:
  - nominal stability is explicit
  - the known quantity G_0 and the quantity sought by the control engineer (Q) occur in the highly insightful relation $T_0 = QG_0$ (multiplicative in the frequency domain); whether a designer chooses to work in this quantity from the beginning or prefers to start with a synthesis technique and then convert, the simple multiplicative relation QG_0 provides deep insights into the trade-offs of a particular problem and provides a very direct means of pushing the design by shaping Q.
  - The sensitivities are affine in Q, which is a great advantage for synthesis techniques relying on numerical minimization of a criterion (*see Chapter 16 for a detailed discussion of optimization methods which exploit this parameterization*).

- The following points are important to avoid some common misconceptions:
  - the associated trade-offs are not a consequence of the affine parameterization: they are perfectly general and hold for any linear time invariant controller including LQR, PID, pole placement based, H_∞, etc.
  - we have used the affine parameterization to make the general trade-offs more visible and to provide a direct means for the control engineer to make trade-off decisions; this should not be confused with synthesis techniques that make particular choices in the affine parameterization to synthesize a controller.
  - The fact that Q must approximate the inverse of the model at frequencies where the sensitivity is meant to be small is perfectly general and highlights the fundamental importance of inversion in control. This does not necessarily mean that the controller, C, must contain this approximate inverse as a factor and should not be confused with the pros and cons of that particular design choice.

- PI and PID design based on affine parameterization.
  - 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.
  - 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.
  - All trade-offs and insights of the previous chapters also apply to PID based control loops.

- 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 (*Chapter 3*).
- The chapter provides explicit formulas for first-order, time-delay second order and integrating processes.
- 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.

- Since the PI(D) parameter formulas are provided explicitly in terms of physical process parameters, the PI(D) gains can be scheduled to measurably changing parameters without extra effort (*it is possible, for example, to schedule for a speed-dependent time-delay*).
- The approach does not preempt the design choice of canceling or shifting the open-loop poles - both are possible and associated with different trade-offs

- 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 analyzed 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 parameterization with the particular synthesis technique of the Smith-predictor.

- Summary of results for unstable systems:
  - All stabilizing controllers for an unstable plant have the form

$$C(s) = \frac{\frac{P(s)}{E(s)} + Q_u(s)\frac{A_o(s)}{E(s)}}{\frac{L(s)}{E(s)} - Q_u(s)\frac{B_o(s)}{E(s)}}$$

    where $Q_u(s)$ is any proper rational stable transfer function.
  - Polynomials $A_0(s)$, $B(s)$, $E(s)$, $P(s)$ and $L(s)$ satisfy

$$A_o(s)L(s) + B_o(s)P(s) = E(s)F(s)$$

    where $E(s)$ and $F(s)$ are polynomials of suitable degrees which are arbitrary, save that they must have desirable zeros.
  - Any stabilizing controller can be used to obtain an initial set of polynomials {E(s), P(s), L(s)}