Chapter 7

THE IMC-BASED PID PROCEDURE

In chapters 5 and 6 we developed a transparent framework for control system design: the Internal Model Control (IMC) structure. One nice thing about the IMC procedure, is that it results in a controller with a single tuning parameter, the IMC filter ($\lambda$). For a system which is “minimum-phase”, $\lambda$ is equivalent to a closed-loop time constant (the “speed of response” of the closed-loop system). Although the IMC procedure is clear and IMC is easily implemented, the most common industrial controller is still the PID controller. The purpose of this chapter is to show that the IMC block diagram can be rearranged to the form of a standard feedback control diagram. We will find that the IMC law, for a number of common process transfer functions, is equivalent to PID-type feedback controllers.

After studying this chapter the student should be able to:

- Design an internal model controller, then find the equivalent feedback controller in standard form. Derive and use the results presented in Table 7.1
- Use a Padé approximation for time-delays in order to find a PID-type control law.
- Compare the IMC-based PI, PID and improved PI controllers for first-order + time-delay processes. Use Table 7.2.
- Use Table 7.3 to find PID-type controllers for unstable processes.

The major sections of this chapter are:

7.1 Background
7.2 Equivalent Feedback Form of IMC
7.3 IMC-based Design for Delay-Free Processes
7.4 IMC-based Design for Systems with Time-Delays
7.5 Summary of IMC-based Design for Stable Processes
7.6 IMC-based Design for Unstable Processes
7.7 Summary of IMC-based Design for Unstable Processes
7.8 Summary
7.1 BACKGROUND

As we will show in this chapter, the IMC structure can be rearranged to the standard feedback structure.

Question: Why do we care about IMC, if we can show that it can be rearranged into the standard feedback structure?

Answer: Because the process model is explicitly used in the control system design procedure. The standard feedback structure uses the process model in an implicit fashion, that is, PID tuning parameters are “tweaked” on a transfer function model, but it is not always clear how the process model effects the tuning decision. In the IMC formulation, the controller, q(s), is based directly on the “good” part of the process transfer function. The IMC formulation generally results in only one tuning parameter, the closed loop time constant (λ, the IMC filter factor). The PID tuning parameters are then a function of this closed-loop time constant. The selection of the closed-loop time constant is directly related to the robustness (sensitivity to model error) of the closed-loop system.

The reader should realize that the IMC-based PID controller presented in this chapter will not give the same results as the IMC strategy when there are process time delays, because the IMC-based PID procedure uses a Padé approximation for deadtime, while the IMC strategy uses the exact representation for deadtime.

7.2 THE EQUIVALENT FEEDBACK FORM TO IMC

In this section we derive the feedback equivalence to IMC by using block diagram manipulation. Begin with the IMC structure shown in Figure 7.1; the point of comparison between the model and process output can be moved as shown in Figure 7.2.

![Figure 7.1. IMC Structure](image-url)
7.2 Equivalent Feedback Form to IMC

Figure 7.2. Cosmetic Change in IMC Structure

Figure 7.2 can be rearranged to the form of Figure 7.3.

Figure 7.3. Rearrangement of IMC Structure

The arrangement shown inside the dotted line of Figure 7.3 is shown below in Figure 7.4.

Figure 7.4. Inner-Loop of the Rearranged IMC Structure shown in Figure 7.5.

Figure 7.4 can be rearranged to the form of Figure 7.5.
Figure 7.5. Equivalent Block to Figure 7.4.

Notice that \( r(s) - y(s) \) is simply the error term used by a standard feedback controller. Therefore, we have found that the IMC structure can be rearranged to the feedback control (FBC) structure, as shown in Figure 7.6. This reformulation is advantageous because we will find that a PID controller often results when the IMC design procedure is used. Also, the standard IMC block diagram cannot be used for unstable systems, so this feedback form must be used for those cases.

Figure 7.6. Standard Feedback Diagram Illustrating the Equivalence with Internal Model Control. The feedback controller, \( g_c(s) \), contains both the internal model, \( \tilde{g}_p(s) \), and internal model controller, \( q(s) \).

Now, we can use the IMC design procedure to help us design a standard feedback controller. The standard feedback controller is a function of the internal model, \( \tilde{g}_p(s) \), and internal model controller, \( q(s) \), as shown in equation (7.1).

The standard feedback controller which is equivalent to IMC is

\[
g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)}
\]  

We will refer to equation (7.1) as the IMC-based PID relationship because the form of \( g_c(s) \) is often that of a PID controller. The IMC-Based PID procedure is similar to the IMC procedure of the previous handout, with some additional steps. Step 1 of the IMC-Based PID procedure contains steps 1-3 of the IMC procedure. One major difference is that the IMC-Based procedure will, many times, not require that the controller be proper. Also, the process deadtime will be approximated using the Padé procedure, in order to arrive at an equivalent PID-type controller. Because of the Padé approximation for deadtime, the IMC-based PID controller will not perform as well as IMC for processes with time-delays.
The IMC-Based PID Control Design Procedure

The following steps are used in the IMC-based PID control system design:

1. Find the IMC controller transfer function, \( q(s) \), which includes a filter, \( f(s) \), to make \( q(s) \) semi-proper or to give it derivative action (order of the numerator of \( q(s) \) is one order greater than the denominator of \( q(s) \)). Notice that this is a major difference from the IMC procedure. Here, in the IMC-based procedure, we may allow \( q(s) \) to be improper, in order to find an equivalent PID controller. The bad news is - you must know the answer that you are looking for, before you can decide whether to make \( q(s) \) proper or improper in this procedure. We will generally guide you along the correct path.

2. Find the equivalent standard feedback controller using the transformation

\[
g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)}
\]

write this in the form of a ratio between two polynomials.

3. Show this in PID form and find \( k_c, \tau_i, \tau_d \). Sometimes this procedure results in a PID controller cascaded with a lag term \( \tau_f \):

\[
g_c(s) = k_c \left[ \frac{\tau_i \tau_D s^2 + \tau_i s + 1}{\tau_f s + 1} \right]
\]

4. Perform closed-loop simulations for both the perfect model case and cases with model mismatch. Choose the desired value for \( \lambda \) as a trade-off between performance and robustness.

7.3 IMC-BASED FEEDBACK DESIGN FOR DELAY-FREE PROCESSES

The procedure outlined in the previous section will be illustrated by way of two examples: (1) a first-order process and (2) a second-order process. For simplicity we will drop the (\( ~ \)) notation on all of the process model parameters.

Example 7.1 IMC-Based PID Design for a First-Order Process

Find the PID-equivalent to IMC for a first-order process

\[
\tilde{g}_p(s) = \frac{k_p}{\tau_p s + 1}
\]

Step 1. Find the IMC controller transfer function, \( q(s) \), which includes a filter to make \( q(s) \) semi-proper.
\[ q(s) = \tilde{q}(s)f(s) = \tilde{g}_p^{-1}(s)f(s) = \frac{\tau_ps + 1}{k_p} \frac{1}{\lambda s + 1} \]

\[ q(s) = \frac{1}{k_p} \frac{\tau_ps + 1}{\lambda s + 1} \]  

**(Step 2)**. Find the equivalent standard feedback controller using the transformation

\[ g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)} = \frac{\frac{\tau_ps + 1}{k_p (\lambda s + 1)}}{1 - \frac{\tau_ps + 1}{k_p (\lambda s + 1)}} = \frac{\tau_ps + 1}{k_p \lambda s} \]  

 recall that the transfer function for a PI controller is

\[ g_c(s) = k_c \frac{\tau_ps + 1}{\tau_is} \]  

**(Step 3)**. Rearrange (7.4) to fit the form of (7.5). Multiplying (7.4) by \( \tau_p/\tau_p \), we find

\[ g_c(s) = \left( \frac{\tau_p}{k_p \lambda} \right) \frac{\tau_ps + 1}{\tau_ps} \]  

equating terms in (7.5) and (7.6), we find the following values for the PI tuning parameters

\[ k_c = \frac{\tau_p}{k_p \lambda} \]
\[ \tau_i = \tau_p \]  

The IMC-based PID design procedure for a first-order process has resulted in a PI control law. The major difference is that there are no longer two degrees of freedom in the tuning parameters \((k_c, \tau_i)\) - the IMC-based procedure shows that only the proportional gain needs to be adjusted. The integral time is simply set equal to the process time constant. Notice that the proportional gain is inversely related to \( \lambda \), which makes sense. If \( \lambda \) is small (closed loop is “fast”) the controller gain must be large. Similarly, if \( \lambda \) is large (closed-loop is “slow”) the controller gain must be small. Also notice that the same results were obtained using the *direct synthesis method* - a specified first-order closed-loop response for a first-order process lead to a PI controller with the parameter values in equation (7.7).

This procedure can be used to develop the equivalent PID (+ lag, in some cases) controller for a number of other transfer functions, as shown in Table 7.1. In the next example, we derive the PID controller for a second-order process.
Example 7.2 IMC-Based PID Design for a Second-Order Process

Find the PID-equivalent to IMC for a second-order process,

\[ \tilde{g}_p(s) = \frac{k_p}{(\tau_1s + 1)(\tau_2s + 1)} \]

**Step 1.** Find the IMC controller transfer function, \( q(s) \) - here we allow \( q(s) \) to be improper, because we wish to end up with a PID controller when we are

\[ q(s) = \tilde{g}(s)f(s) = \frac{(\tau_1s + 1)(\tau_2s + 1)}{k_p} \frac{1}{(\lambda s + 1)} \]  

(7.8)

**Step 2.** Find the equivalent standard feedback controller using the transformation

\[ g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)} = \frac{(\tau_1s + 1)(\tau_2s + 1)}{k_p(\lambda s + 1)} \frac{1}{(\tau_1s + 1)(\tau_2s + 1)} \frac{\tau_1\tau_2s^2 + (\tau_1 + \tau_2)s + 1}{k_p\lambda s} \]  

(7.9)

recall that the transfer function for a PID controller is

\[ g_c(s) = k_c\left[ \frac{\tau_1\tau_Ds^2 + \tau_1s + 1}{\tau_1s} \right] \]  

(7.10)

**Step 3.** Rearrange (7.9) to fit the form of (7.10). Multiplying (7.9) by \( (\tau_1 + \tau_2)/(\tau_1 + \tau_2) \), we find

\[ g_c(s) = \left( \frac{\tau_1 + \tau_2}{k_p\lambda} \right) \frac{\tau_1\tau_2s^2 + (\tau_1 + \tau_2)s + 1}{(\tau_1 + \tau_2)s} \]  

(7.11)

equating terms in (7.10) and (7.11), we find the following relationships

\[ k_c = \frac{(\tau_1 + \tau_2)}{k_p\lambda} \]

\[ \tau_f = \tau_1 + \tau_2 \]  

(7.12)

\[ \tau_D = \frac{\tau_1\tau_2}{\tau_1 + \tau_2} \]

The result for the previous example is also shown in Table 7.1. To develop a more complete understanding of this procedure, the reader should derive the parameters for some of the other relationships shown in Table 7.1. The reader should also note that the same results would be obtained using the direct synthesis approach, if the proper desired closed-loop transfer function is specified. The proper desired closed-loop transfer function appears clearly in the IMC procedure.
Notice that Table 7.1 is for process transfer functions that do not have a time-delay. The following section develops PID tuning relationships for a first-order + time-delay process.

### 7.4 IMC-BASED FEEDBACK DESIGN FOR PROCESSES WITH A TIME-DELAY

In order to arrive at a PID equivalent form for processes with a time-delay, we must make some approximation to the deadtime (notice that the IMC strategy does not use an approximation for deadtime. We are only doing this for the IMC-Based PID procedure). We will use either a zeroth or a first-order Padé approximation for deadtime.

---

**Example 7.3 IMC-Based PID Design for a First-Order + Deadtime Process**

Find the PID controller which approximates IMC for a first-order + time-delay process

\[
\tilde{g}_p(s) = \frac{k_p e^{-\theta \tau}}{\tau_p s + 1} \quad (7.13)
\]

**Step 1.** Use a first-order Padé approximation for deadtime \(e^{-\theta \tau} \approx -0.50s + 1\)

\[
\tilde{g}_p(s) = \frac{k_p e^{-\theta \tau}}{\tau_p s + 1} = \frac{k_p(-0.50s + 1)}{(\tau_p s + 1)(0.50s + 1)} \quad (7.14)
\]

**Step 2.** Factor out the non-invertible elements (this time, do not make the “bad” part “all-pass”)

\[
\tilde{g}_{\text{p-}}(s) = \frac{k_p}{(\tau_p s + 1)(0.50s + 1)}
\]

\[
\tilde{g}_{\text{p+}}(s) = -0.50s + 1 \quad (7.15)
\]

**Step 3.** Form the idealized controller

\[
\tilde{q}(s) = \frac{(\tau_p s + 1)(0.50s + 1)}{k_p} \quad (7.16)
\]

**Step 4.** Add the filter - this time we will not make \(q(s)\) proper, because a PID controller will not result. We use the “derivative” option, where we allow the numerator of \(q(s)\) to be one order higher than the denominator [**NOTE**: this is only done so that we will obtain a PID controller].

\[
q(s) = \tilde{q}(s)f(s) = \tilde{g}_{\text{p-}}^{-1}(s)f(s) = \frac{(\tau_p s + 1)(0.50s + 1)}{k_p} \frac{1}{\lambda_s + 1} \quad (7.17)
\]
Now, find the PID equivalent. Recall that

\[ g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)} = \frac{\tilde{q}(s)f(s)}{1 - \tilde{g}_p(s)\tilde{q}(s)f(s)} \]

\[ g_c(s) = \frac{\tilde{q}(s)f(s)}{1 - \tilde{g}_p(s)\tilde{g}_p^{-1}(s)f(s)} = \frac{\tilde{q}(s)f(s)}{1 - \tilde{g}_p(s)f(s)} = \left(1 - \frac{1}{k_p}\right)\frac{(\tau_p s + 1)(0.5\theta s + 1)}{(\lambda + 0.5\theta)s} \]  

(7.18)

we can expand the numerator term to find

\[ g_c(s) = \left(1 - \frac{1}{k_p}\right)\frac{0.5\tau_p \theta s^2 + (\tau_p + 0.5\theta)s + 1}{(\lambda + 0.5\theta)s} \]  

(7.19)

we can multiply (7.19) by \( \tau_p + 0.5\theta / \tau_p + 0.5\theta \) to find the PID parameters

\[ k_c = \frac{(\tau_p + 0.5\theta)}{k_p(\lambda + 0.5\theta)} \]

\[ \tau_i = \tau_p + 0.5\theta \]

\[ \tau_d = \frac{\tau_p \theta}{2\tau_p + \theta} \]  

(7.20)

The IMC-based PID controller design procedure has resulted in a PID controller, when the process is first-order + deadtime. Remember that a Padé approximation for deadtime was used in this development, meaning that the filter factor cannot be made arbitrarily small, therefore there will be performance limitations to the IMC-based PID strategy that do not occur in the IMC strategy. Rivera et al. (1986) recommend that \( \lambda > 0.8\theta \) because of the model uncertainty due to the Padé approximation.

In example 7.3 the “all-pass” formulation was not used. The reader should show that the use of an “all-pass” in the factorization will lead to a PID controller in series with a first order lag. The parameters, in this case, are shown as the first entry in Table 7.2. Morari and Zafiriou (1989) recommend \( \lambda > 0.25\theta \) for the PID + lag formulation.

Example 7.4 IMC-Based PI Design for a First-Order + Time-delay Process

In Example 7.3 we showed that a first-order Padé approximation for deadtime leads to a PID controller. In this example the zero-order Padé approximation is used, that is

\[ \tilde{g}_p(s) = \frac{k_p e^{-\alpha\tau}}{\tau_p s + 1} \approx \frac{k_p}{(\tau_p s + 1)} \]  

(7.21)

The IMC-based tuning parameters for a first-order process have already been derived in example 7.1, and are shown here for convenience
Chapter 7. IMC-Based PID

\[ k_c = \frac{\tau_p}{k_p \lambda} \]  
\[ \tau_I = \tau_p \]  

(7.22)

Since deadtime has been neglected, creating quite a bit of model error, Morari and Zafiriou (1989) recommend that \( \lambda > 1.7\theta \).

Example 7.5 Improved IMC-Based PI Design for a First-Order + Deadtime Process

In Example 7.4 the zero-order Padé approximation for deadtime was used (that is, the time-delay was neglected). Here, we again neglect the time-delay, but increase the time constant to help approximate the time delay.

\[ \tilde{g}_p(s) = \frac{k_p e^{-\alpha \tau}}{\tau_p s + 1} = \frac{k_p}{(\tau_p + 0.5\theta)s + 1} \]  

(7.23)

Again, the IMC-based tuning parameters for a first-order process have already been derived in example 7.1, and lead to the following

\[ k_c = \frac{\tau_p + 0.5\theta}{k_p \lambda} \]  
\[ \tau_I = \tau_p + 0.5\theta \]  

(7.24)

since deadtime has been approximated, Morari and Zafiriou (1989) recommend that \( \lambda > 1.7\theta \) for the improved PI tuning parameters. We refer to this as the improved PI tuning parameters because these parameters are more robust than the parameters derived in Example 7.4 which were based on a zero-order Padé approximation in order to derive a PI controller. Table 7.2 provides a concise summary of the various IMC-based PI(D) controllers. In Example 7.6 we compare the closed-loop performance of the various IMC-based controllers with “pure” IMC.

Example 7.6 Comparison of IMC and IMC-based PI and PID Controllers for First-Order + deadtime Processes

Consider the first-order + deadtime process

\[ \tilde{g}_p(s) = \frac{1e^{-5s}}{10s + 1} \]

After factoring out the time-delay, we find the internal model controller

\[ q(s) = \frac{10s + 1}{\lambda s + 1} \]

We will compare the performance, via simulation, of the IMC strategy with the various IMC-based PI(D) controllers shown in the table below.
## Controller Design for Time-Delay Processes

### Table: Controller Formulas

<table>
<thead>
<tr>
<th>Controller</th>
<th>$k_c$</th>
<th>$\tau_i$</th>
<th>$\tau_d$</th>
<th>$\tau_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID+lag</td>
<td>$\frac{12.5}{5+\lambda}$</td>
<td>12.5</td>
<td>2</td>
<td>$\frac{5\lambda}{10+2\lambda}$</td>
</tr>
<tr>
<td>PID</td>
<td>$\frac{12.5}{2.5+\lambda}$</td>
<td>12.5</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>PI</td>
<td>$\frac{10}{\lambda}$</td>
<td>10</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>&quot;Improved&quot; PI</td>
<td>$\frac{12.5}{\lambda}$</td>
<td>12.5</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The closed-loop behavior of IMC and the various IMC-based PI(D) controllers, as a function of $\lambda$, is compared in Figures 7.7a - 7.7c. The closed-loop system is unstable for the PI controllers, when $\lambda = 2.5$ (Figure 7.7a). Notice that the PID controller results in “spikes” in the process output; this is an artifact of the “ideal PID” controller, and cannot occur in practice, since a perfect derivative cannot be implemented. The PID with a filter (lag) has much more desirable performance. Figures 7.7b and 7.7c show that the PI controllers are stable when $\lambda$ is increased. Also, notice that the PID with a filter has almost identical performance to IMC. The results of this simulation study are:

- For $\lambda = 2.5$ the IMC-based PI controllers are unstable. Notice from Table 7.2 that the recommended range for $\lambda$ is $\lambda > 1.5*5$, or $\lambda > 8.5$.
- The IMC-based PID with filter performs almost as well as IMC. From Table 7.2 the recommended range for the PID+filter $\lambda$ is $\lambda > 0.25*5$, or $\lambda > 1.25$.
- From Table 7.2 the recommended range for the PID is $\lambda > 0.8*5$, or $\lambda > 4$.
- An additional recommendation from Table 7.2 is that $\lambda > 2\tau_p$; for this example the requirement is that $\lambda > 2$.

### Summary of PI(D) Control of First-order + Time-delay Processes

Different assumptions have been to derive the PI and PID controllers shown in examples 7.3 to 7.5. A zeroth-order Padé approximation leads to a PI controller while a first-order Padé approximation leads to PID + lag (when an all-pass filter is used) and PID (when the all-pass is not used) controllers. Generally, the PID + lag controller will be easier to tune for robustness and will certainly be less sensitive to noise than the PID controller. Example 7.6 also showed that the PID + filter performance is almost identical to the “pure” IMC. This shows that the powerful IMC framework can be used to design PID-type controllers that can be implemented on industrial processes using existing (PID) control equipment.
Chapter 7. IMC-Based PID

Figure 7.7a. Comparison of IMC and various IMC-based PI(D) controllers for $\lambda = 2.5$. The PI controllers were unstable at this low value of $\lambda$.

Figure 7.7b. Comparison of IMC and various IMC-based PI(D) controllers. $\lambda = 5.0$. The unmarked curve is the PID+filter controller, which has almost identical performance to IMC.
Figure 7.7c. Comparison of IMC and various IMC-based PI(D) controllers for first-order + time-delay process. $\lambda = 10.0$. The unmarked curve is the PID+filter controller, which has almost identical performance to IMC.

7.5 SUMMARY OF IMC-BASED PID CONTROLLER DESIGN FOR STABLE PROCESSES

We have shown several examples where the IMC design procedure could be used to develop an equivalent PID-type control law. For stable processes with no time-delay, the IMC-based PID procedure gives the exact same feedback performance as IMC. For stable processes with a time-delay the IMC-based PID procedure will not give exactly the same performance as IMC, because a Padé approximation for deadtime is used in the controller design.

We want this following point to be clear to the reader. For process transfer functions without time-delays, the IMC-based PID controller will yield exactly the same performance as IMC. This will occur if no approximation has to be made in the process model to find a feedback form which is equivalent to PID. If an approximation (such as Padé) is made in the IMC-based PID strategy which is not made in the IMC strategy, then the performance will not be the same.

Table 7.1 provides a summary of the PID tuning parameters for systems without a time-delay, while Table 7.2 summarizes the PID tuning parameters for the stable first-order + time-delay process.
<table>
<thead>
<tr>
<th>( g_p(s) )</th>
<th>( g_{CL}(s) )</th>
<th>( k_c )</th>
<th>( \tau_I )</th>
<th>( \tau_D )</th>
<th>( \tau_F )</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>( \frac{k_p}{\tau_p s+1} )</td>
<td>( \frac{1}{\lambda s+1} )</td>
<td>( \frac{\tau_p}{k_p \lambda} )</td>
<td>( \tau_p )</td>
<td>—</td>
</tr>
<tr>
<td>B</td>
<td>( \frac{k_p}{(\tau_1 s+1)(\tau_2 s+1)} )</td>
<td>( \frac{1}{\lambda s+1} )</td>
<td>( \frac{\tau_1+\tau_2}{k_p \lambda} )</td>
<td>( \tau_1+\tau_2 )</td>
<td>( \frac{\tau_1 \tau_2}{\tau_1+\tau_2} )</td>
</tr>
<tr>
<td>C</td>
<td>( \frac{k_p}{\tau^2 s^2+2\zeta \tau s+1} )</td>
<td>( \frac{1}{\lambda s+1} )</td>
<td>( \frac{2\zeta \tau}{k_p \lambda} )</td>
<td>( 2\zeta \tau )</td>
<td>( \tau )</td>
</tr>
<tr>
<td>D</td>
<td>( \frac{k_p}{\tau^2 s^2+2\zeta \tau s+1} )</td>
<td>( \frac{1}{(\lambda s+1)^2} )</td>
<td>( \frac{\zeta \tau}{k_p \lambda} )</td>
<td>( 2\zeta \tau )</td>
<td>( \frac{\tau}{2\zeta} )</td>
</tr>
<tr>
<td>E</td>
<td>( \frac{k_p (-\beta s+1)}{\tau^2 s^2+2\zeta \tau s+1} )</td>
<td>( \frac{(-\beta s+1)}{(\beta s+1)(\lambda s+1)} )</td>
<td>( \frac{2\zeta \tau}{k_p (2\beta+\lambda)} )</td>
<td>( 2\zeta \tau )</td>
<td>( \tau )</td>
</tr>
<tr>
<td>F</td>
<td>( \frac{k_p (-\beta s+1)}{\tau^2 s^2+2\zeta \tau s+1} )</td>
<td>( \frac{(-\beta s+1)}{(\lambda s+1)} )</td>
<td>( \frac{2\zeta \tau}{k_p (\beta+\lambda)} )</td>
<td>( 2\zeta \tau )</td>
<td>( \tau )</td>
</tr>
<tr>
<td>G</td>
<td>( \frac{k_p}{s} )</td>
<td>( \frac{1}{\lambda s+1} )</td>
<td>( \frac{1}{k_p \lambda} )</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>H</td>
<td>( \frac{k_p}{s(\tau_p s+1)} )</td>
<td>( \frac{1}{\lambda s+1} )</td>
<td>( \frac{1}{k_p \lambda} )</td>
<td>—</td>
<td>( \tau_p )</td>
</tr>
</tbody>
</table>

**Note:** Parameters for other process transfer functions are given in Rivera et al. (1986) and Morari and Zafiriou (1989)

In E and F it is assumed that \( \beta > 0 \) (inverse response, right-half-plane zeros)

The controller for D and E is PID + lag. \( g_c(s) = k_c \left[ \frac{\tau_1 \tau_D s^2 + \tau_1 s + 1}{\tau_1 s} - \frac{1}{\tau_2 s + 1} \right] \)
Table 7.2 PID Tuning Parameters for First-Order + Time Delay Processes

<table>
<thead>
<tr>
<th>Controller</th>
<th>$k_c$</th>
<th>$\tau_I$</th>
<th>$\tau_D$</th>
<th>$\tau_F$</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID $\tau_p + \theta$</td>
<td>$\tau_p + \theta$</td>
<td>$\tau_p \theta$</td>
<td>$\frac{\theta \lambda}{2(\theta + \lambda)}$</td>
<td>(1)</td>
<td></td>
</tr>
<tr>
<td>PID $\tau_p + \frac{\theta}{\theta_\tau}$</td>
<td>$\tau_p + \frac{\theta}{\theta_\tau}$</td>
<td>$\tau_p \theta$</td>
<td>$\frac{\theta \lambda}{2(\theta + \lambda)}$</td>
<td>(2)</td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>$\frac{\tau_p}{k_p \lambda}$</td>
<td>$\tau_p$</td>
<td>—</td>
<td>—</td>
<td>(3)</td>
</tr>
<tr>
<td>Improved PI</td>
<td>$\frac{\tau_p + \theta}{k_p \lambda}$</td>
<td>$\tau_p + \theta$</td>
<td>—</td>
<td>—</td>
<td>(4)</td>
</tr>
</tbody>
</table>

$$g_p(s) = \frac{k_p e^{-\theta s}}{\tau_p s + 1}$$

$$g_c(s) = k_c \left[ \frac{\tau_I \tau_D s^2 + \tau_I s + 1}{\tau_I s} \right] \left[ \frac{1}{\tau_F s + 1} \right]$$

(1) With an “all-pass” factorization and semi-proper $q(s)$. Recommended $\lambda > 0.25 \theta$.

(2) Without an “all-pass” factorization and improper $q(s)$. Recommended $\lambda > 0.8 \theta$.

(3) With zero-order Padé approximation ($e^{-\theta s} \approx 1$). Recommended $\lambda > 1.7 \theta$.

(4) With the approximation $\frac{k_p e^{-\theta s}}{\tau_p s + 1} \approx \frac{k_p}{\theta (\tau_p + \frac{\theta}{\theta_\tau}) s + 1}$. Recommended $\lambda > 1.7 \theta$.

In all cases it is recommended that $\lambda > 0.2 \tau_p$.

7.6 IMC-BASED PID CONTROLLER DESIGN FOR UNSTABLE PROCESSES
The IMC procedure must be modified for unstable processes. Rotstein and Lewin (1991) have used the procedure developed by Morari and Zafiriou (1989) to find IMC-based PID controllers for unstable processes. The modification to the procedure shown in sections 7.2 and 7.3 is to use a slightly more complicated filter transfer function.

1. Find the IMC controller transfer function, \( q(s) \), which includes a filter, \( f(s) \), to make \( q(s) \) semi-proper. An additional requirement is that the value of \( f(s) \) at \( s = p_u \) (where \( p_u \) is an unstable pole) must be 1. That is

\[
f(s = p_u) = 1
\]

Morari and Zafiriou (1989) recommend a filter function which has the form

\[
f(s) = \frac{\gamma s + 1}{\lambda s + 1^m}
\]

where \( n \) is chosen to make \( q(s) \) proper (usually semi-proper). A value of \( \gamma \) is found which satisfies the filter requirement \( f(s = p_u) = 1 \).

2. Find the equivalent standard feedback controller using the transformation

\[
g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)}
\]

write this in the form of a ratio between two polynomials.

3. Show this in PID form and find \( k_c, \tau_i, \tau_d \). Sometimes this procedure results in a PID controller cascaded with a lag term \( \tau_f \):

\[
g_c(s) = k_c \left[ \frac{\tau_i \tau_d s^2 + \tau_i s + 1}{\tau_i s} \right] \left[ \frac{1}{\tau_f s + 1} \right]
\]

4. Perform closed-loop simulations for both the perfect model case and cases with model mismatch. Choose the desired value for \( \lambda \) as a trade-off between performance and robustness.

Example 7.7 illustrates this procedure for a first-order unstable process.

---

Example 7.7 IMC-based PID Design for a First-Order Unstable Process

Find the IMC-based PID controller for a first-order unstable process

\[
\tilde{g}_p(s) = \frac{k}{-\tau_u s + 1}
\]

where \( \tau_u \) is given a positive value. The pole, \( p_u \) is \( 1/\tau_u \).

Step 1. Find the IMC controller transfer function, \( q(s) \)
7.6 IMC-Based Feedback Design for Unstable Processes

\[ q(s) = \hat{q}(s)f(s) = \tilde{g}_p^{-1}(s)f(s) = \frac{-\tau_s s + 1}{k_p} \frac{\gamma s + 1}{(\lambda s + 1)^2} \]  

(7.26)

Note that we have selected a second-order polynomial in the filter to make the controller, \( q(s) \), semi-proper. Now we solve for \( \gamma \) so that \( f(s = p_u) = 1 \)

\[ f(s) = \frac{\gamma s + 1}{(\lambda s + 1)^2} \]

\[ f\left( s = \frac{1}{\tau_u} \right) = \frac{\gamma (1/\tau_u) + 1}{(\lambda (1/\tau_u) + 1)^2} = 1 \]

so

\[ \gamma \left( \frac{1}{\tau_u} \right) + 1 = \left( \lambda \left( \frac{1}{\tau_u} \right) + 1 \right)^2 \]

solving for \( \gamma \) we find

\[ \gamma = \lambda \left( \frac{\lambda}{2} + 2 \right) \]  

(7.27)

**Step 2.** Find the equivalent standard feedback controller using the transformation

\[ g_c(s) = \frac{q(s)}{1 - \tilde{g}_p(s)q(s)} \]

after a lengthy bit of algebra (Appendix 1)

\[ g_c(s) = \frac{\gamma}{k_p(2\lambda - \gamma)} \frac{(\gamma s + 1)}{\gamma s} \]  

(7.28)

**Step 3.** This is in the form of a PI controller, where

\[ k_c = \frac{\gamma}{k_p(2\lambda - \gamma)} \]  

(7.29)

\[ \tau_i = \gamma \]

or (after some more algebra for \( k_c \))

\[ k_c = \frac{-(\lambda + 2\tau_u)}{k_p \lambda} \]  

(7.30)

\[ \tau_i = \lambda \left( \frac{\lambda}{\tau_u} + 2 \right) \]
As a numerical example, consider \( g_p(s) = \frac{1}{-s + 1} \). The closed loop output responses for various values of \( \lambda \) are shown in Figure 7.8 while the manipulated variable responses are shown in Figure 7.9. Notice that we do not achieve the nice overdamped-type of closed-loop output responses that we were able to obtain with open-loop stable processes. The reader should show that the closed-loop relationship for this system is

\[
y(s) = \frac{g_c(s)g_p(s)}{1 + g_c(s)g_p(s)} r(s) = \frac{\gamma s + 1}{(\lambda s + 1)^2} r(s)
\]

which has overshoot if \( \gamma > \lambda \) (this is always the case for this system (see eqn 7.27)).

**Figure 7.8.** Closed-loop Output Responses for a Step Setpoint Change. Various values of \( \lambda \).
In Figure 7.8 we notice that the closed-loop response had overshoot. A response to a setpoint change, without overshoot, can be obtained by including a setpoint filter, as shown in Figure 7.10. The setpoint filter is

\[ f_{sp}(s) = \frac{1}{\gamma s + 1} \]

which yields the following closed-loop response for a setpoint change

\[ y(s) = \frac{g_c(s)g_p(s)}{1 + g_c(s)g_p(s)} f_{sp}(s)r(s) = \frac{1}{(\lambda s + 1)^2} r(s) \]
The results for several unstable process transfer functions are shown in Table 7.3. See Rotstein and Lewin (1991) for a discussion of the effect of deadtime and model uncertainty on the control of unstable processes.

<table>
<thead>
<tr>
<th>$g_p(s)$</th>
<th>$g_{CL}(s)$</th>
<th>$k_c$</th>
<th>$\tau_I$</th>
<th>$\tau_D$</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{k_p}{-\tau_u s + 1}$</td>
<td>$\frac{\gamma s + 1}{(\lambda s + 1)^2}$</td>
<td>$\frac{-(\lambda + 2\tau_u)}{k_p \lambda}$</td>
<td>$\gamma$</td>
<td>_</td>
<td>(1)</td>
</tr>
<tr>
<td>$\frac{k_p}{(-\tau_u s + 1)(\tau_p s + 1)}$</td>
<td>$\frac{\gamma s + 1}{(\lambda s + 1)^2}$</td>
<td>$\frac{-\tau_u(\gamma + \tau_p)}{k_p \lambda^2}$</td>
<td>$\gamma + \tau_p$</td>
<td>$\frac{\gamma \tau_p}{\gamma + \tau_p}$</td>
<td>(1)</td>
</tr>
<tr>
<td>$\frac{k_p(\tau_n s + 1)}{(-\tau_u s + 1)(\tau_p s + 1)}$</td>
<td>$\frac{\gamma s + 1}{(\lambda s + 1)^2}$</td>
<td>$\frac{-1}{k_p(1 + \frac{2\tau_u}{\lambda})}$</td>
<td>$\gamma$</td>
<td>(1, 2)</td>
<td></td>
</tr>
<tr>
<td>$\frac{k_p(\tau_n s + 1)}{(-\tau_u s + 1)(\tau_1 s + 1)(\tau_2 s + 1)}$</td>
<td>$\frac{\gamma s + 1}{(\lambda s + 1)^3}$</td>
<td>$\frac{\tau_1}{3\lambda - \gamma}$</td>
<td>$\tau_1$</td>
<td>(3)</td>
<td></td>
</tr>
</tbody>
</table>

(1) $\gamma = \lambda \left(\frac{\lambda}{\tau_u} + 2\right)$

(2) PI is cascaded with a lead/lag filter $\frac{\tau_p s + 1}{\tau_n s + 1}$

(3) PI is cascaded with a lead/lag filter $\frac{\tau_2 s + 1}{\tau_n s + 1}(\frac{\gamma s + 1}{\alpha s + 1})$

with $\gamma = \lambda \left[\frac{\lambda^2}{\tau_u^2} + \frac{3\lambda}{\tau_u} + 3\right]$ and $\alpha = \tau_u + \frac{3\lambda^2}{3\lambda - \gamma}$

### 7.7 SUMMARY OF IMC-BASED PID CONTROLLER DESIGN FOR UNSTABLE PROCESSES

A major tuning consideration by the student is that there are both upper and lower bounds on $\lambda$ to assure stability of an unstable process. This is in contrast to stable processes, where the closed-loop is guaranteed to be stable under model uncertainty, simply by increasing $\lambda$ to a large value (detuning the controller).

### 7.8 SUMMARY

We have shown how the IMC procedure can be used to design PID-type feedback controllers. If the process has no time-delay, and the inputs do not hit a constraint, then the IMC-based PID
controllers will have the same performance as IMC. If there is deadtime, then the IMC-based PID controllers will not perform as well as IMC, because of the Padé approximation for deadtime.

It is interesting to note that the IMC-based PID controllers for all of the transfer functions shown in Table 7.1 could have been designed using the direct synthesis method. The key issue in the direct synthesis method is the specification of the closed-loop response characteristic. If the process has a right-half-plane zero, then the specified closed-loop response must also have a right-half-plane zero. The IMC-based PID procedure provides a clear method for handling this.

The IMC design method must be modified to handle unstable processes. Also, the standard IMC structure cannot handle unstable processes, so the controller for an unstable process must be implemented in standard PID feedback form.

REFERENCES


STUDENT EXERCISES

1. Use the IMC-based PID design procedure to find the PID controller for a second order transfer function

$$\tilde{g}_p(s) = \frac{k_p}{\tau^2 s^2 + 2\zeta \tau s + 1}$$

*Hint:* First design the Internal Model Controller, assuming that you will allow the controller to be improper, that is, the order of the numerator is one higher than the denominator. What order filter do you find?

a. Find the PID parameters, $k_c, \tau_i, \tau_d$, as a function of the process model parameters, $k_p, \tau, \zeta$, and the filter factor, $\lambda$.

b. For a perfect model, plot the closed-loop response of $y$ to a step setpoint change in $r$ (as a function of $\lambda$). (Show your control loop diagram)

$\tau = 2$ min, $\zeta = 0.8$, $k_p = 5.25$ psig/gpm

c. Discuss the effect of uncertainty in $\zeta$ (show plots) as a function of $\lambda$.

2. Compare the response of the following first-order + time delay process, using IMC and IMC-based PID. Discuss the effect of $\lambda$ on the closed-loop stability for both systems.

$$\tilde{g}_p(s) = \frac{1e^{-10s}}{5s + 1}$$

Do you find that there is a minimum $\lambda$ required for the stability of the IMC-based PID strategy? How does this relate to the recommendations in Table 7.2. Is there a minimum $\lambda$ required for the stability of the IMC strategy?
3. In example 7.3 the “all-pass” formulation was not used for the first-order + time-delay process with a Padé approximation for deadtime. Show that the use of an “all-pass” in the factorization (and semi-proper $q(s)$) leads to a PID controller in series with a first order lag.

4. For the following inverse response process

$$\tilde{g}_p(s) = \frac{k_p(-\beta s + 1)}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

a. Find the IMC-based PID controller if no “all-pass” is used and the controller is improper.

b. Find the IMC-based PID controller (which must be cascaded with a first-order lag) if an “all-pass” is used and the controller is semi-proper.

5. For the following integrating process

$$\tilde{g}_p(s) = \frac{k_p}{s}$$

Show that the IMC-based PID procedure results in a P controller if $f(s) = \frac{1}{\lambda s + 1}$ and a PI controller if $f(s) = \frac{2\lambda s + 1}{(\lambda s + 1)^2}$.

6. The following process transfer function represents the relationship between boiler feedwater flowrate and steam drum level

$$\tilde{g}_p(s) = \frac{k_p(-\beta s + 1)}{s(\tau_p s + 1)}$$

Use the “all-pass” factorization method to show that the IMC-based PID controller is PD with a first-order lag.

7. Use the IMC-based Feedback Controller Design procedure to design a PID controller for the following process. Assume that the IMC filter is $f(s) = \frac{2\lambda s + 1}{(\lambda s + 1)^2}$

$$\tilde{g}_p(s) = \frac{k_p}{s(\tau_p s + 1)}$$

Find $k_c$, $\tau_I$ and $\tau_D$ - these will be a function of the process parameters and $\lambda$ (show all work). What is the closed-loop transfer function? Sketch the expected response behavior.

8. Consider the following first-order + time-delay process

$$\tilde{g}_p(s) = \frac{25e^{-20s}}{15s + 1}$$
Find the tuning parameters for the IMC-based PID controller (no lag). What is the maximum value of $k_c$ that you would recommend? Why?

9. Consider Example 7.2 where $q(s)$ was allowed to be improper. This yielded a PID controller. Now assume that $q(s)$ is forced to be proper (actually semi-proper). Find the resulting feedback controller. Elaborate on the control structure.

10. A stack gas scrubber has the following relationship between the fresh feedwater flowrate and the $SO_2$ concentration in the water leaving the scrubber

$$y(s) = \frac{-2(6s + 1)}{(10s + 1)(3s + 1)} u(s)$$

Use the IMC-based PID procedure to find the PID controller (+ first-order lag) for this process.

*Hint:* Let the IMC $q(s)$ be semi-proper (order of denominator = order of numerator). Which controller tuning parameters does $\lambda$ effect?

11. Use the IMC-based feedback controller design procedure to design a PID-type controller for the following process. Assume that $q(s)$ is semi-proper and $\tau_n > 0$.

$$\tilde{g}_p(s) = \frac{k_p(\tau_n s + 1)}{\tau_p^2 s^2 + 2\zeta\tau_p s + 1}$$

Find the PID tuning parameters (assuming ideal PID) as a function of the process parameters and $\lambda$ (show all work). What is the closed-loop transfer function? Sketch the expected response behavior.

12. Consider the closed-loop response for IMC when the model is not perfect. Show that there is no offset for a setpoint change for the following model and process transfer functions. Also, find the minimum value of $\lambda$ that assures closed-loop stability when

process model $\tilde{g}_p(s) = \frac{1}{10s + 1}$

process $\tilde{g}_p(s) = \frac{0.75(-s + 1)}{(2s + 1)(8s + 1)}$

13. Show that the IMC strategy cannot be implemented on an unstable process, and must be implemented in standard feedback form.

14. A styrene polymerization reactor is operated at an open-loop unstable point, and has the following input-output model

$$\tilde{g}_p(s) = \frac{-2}{-10s + 1}$$

Design the IMC-based PI controller for this system. Sketch the expected response for a setpoint change.

15. A vinyl-acetate polymerization reactor is operated at an open-loop unstable point, and has the following input-output model
\[ \hat{g}_p(s) = \frac{-2.5}{(-10s + 1)(2s + 1)} \]

a. Design the IMC-based PID controller for this system, that is, find \( k_c \), \( \tau_I \) and \( \tau_D \).

b. What is the closed-loop transfer function? Sketch the expected response for a step setpoint change.

16. Derive \( \gamma \) for the first three elements in Table 7.3. The value of the filter must be one at the location of the unstable pole.

17. A reactor is operated at an open-loop unstable point, and has the following input-output model

\[ \hat{g}_p(s) = \frac{-2(3s + 1)}{(-4s + 1)(5s + 1)} \]

a. Design the IMC-based PID controller (perhaps with a lead/lag filter) for this system, that is, find the tuning parameters.

b. What is the closed-loop transfer function? Sketch the expected response for a step setpoint change.

18. Show that a PI controller cannot stabilize the process \( \hat{g}_p(s) = \frac{k_p}{(-\tau_u s + 1)(\tau_p s + 1)} \) if \( \tau_u < \tau_p \).

(Hint: Use the Routh stability criterion) The IMC-based PID controller (entry 3 in Table 7.3) can handle this process.

Special SIMULINK Exercise

The objective of this assignment is to illustrate a practical problem where the IMC-based PID procedure can be used. The procedure for a stable process will be used in problem 1, while the procedure for an unstable process will be used in problem 2.

Although you will be using a linear controller, you will be implementing this controller on a nonlinear process. The inputs and outputs to/from this process are in physical variables, while the controller design procedure is based on deviation variables.

Background

Biochemical reactors are used in a wide variety of processes, from waste treatment to alcohol fermentation. The modeling equations for a bioreactor are:

\[ \frac{dx_1}{dt} = (\mu - D)x_1 \]

\[ \frac{dx_2}{dt} = D(x_{2f} - x_2) - \frac{\mu x_1}{Y} \]
with the specific growth rate for substrate inhibition kinetics:

\[
\mu = \frac{u_{\text{max}}x_2}{k_m + x_2 + k_1x_2^2}
\]

the state variables are:

\[
x_1 = \text{biomass concentration} = \text{mass of cells/volume}
\]

\[
x_2 = \text{substrate concentration} = \text{mass of substrate/volume}
\]

and the manipulated input is:

\[
D = \text{dilution rate} = \frac{F}{V} = \frac{\text{volumetric flowrate}}{\text{volume of reactor}}
\]

We will use the following parameters for this control study

\[
\mu_{\text{max}} = 0.53 \\
k_m = 0.12 \\
k_1 = 0.4545 \\
Y = 0.4
\]

with a steady-state dilution rate of \(D_s = 0.3\) and feed substrate concentration of \(x_{2fs} = 4.0\).

The nonlinear system has the following 3 steady-state solutions:

<table>
<thead>
<tr>
<th>Steady-state</th>
<th>biomass conc.</th>
<th>substrate conc.</th>
<th>stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equilibrium 1 - wash-out</td>
<td>(x_{1s} = 0)</td>
<td>(x_{2s} = 4.0)</td>
<td>stable</td>
</tr>
<tr>
<td>Equilibrium 2 - non-trivial</td>
<td>(x_{1s} = 0.995103)</td>
<td>(x_{2s} = 1.512243)</td>
<td>unstable (saddle point)</td>
</tr>
<tr>
<td>Equilibrium 3 - non-trivial</td>
<td>(x_{1s} = 1.530163)</td>
<td>(x_{2s} = 0.174593)</td>
<td>stable</td>
</tr>
</tbody>
</table>

The state-space A matrix is:

\[
A = \begin{bmatrix}
\mu_s - D_s & x_{1s}\mu'_s \\
-\mu'_s & -D_s - \frac{\mu'_sx_{1s}}{Y}
\end{bmatrix}
\]

where

\[
\mu'_s = \frac{\partial \mu}{\partial x_{2s}} = \frac{\mu_{\text{max}}k_m}{(k_m + x_{2s})^2}
\]

and the state-space B matrix is:

\[
B = \begin{bmatrix}
-x_{1s} \\
\frac{x_{2fs} - x_{2s}}{Y}
\end{bmatrix}
\]

where dilution rate is the manipulated variable.
In the following two problems, the biomass concentration is the measured output and dilution rate is the manipulated input. You will be doing all of your simulations with the nonlinear process, so the inputs and outputs from the process are in physical variable form, while the controller operates on deviation variables. Assume that the dilution rate is physically constrained between 0 and 0.6.

**Problem 1.** Design an IMC-based PID controller to control the bioreactor at equilibrium point 3 - the stable non-trivial point. The steady-state (also use this as the initial condition for your simulations) is

$$x(0) = \begin{bmatrix} 1.530163 \\ 0.174593 \end{bmatrix}$$

At this operating point, the state space model is

$$A = \begin{bmatrix} 0 & 0.9056 \\ -0.75 & -2.5640 \end{bmatrix}$$

$$B = \begin{bmatrix} -1.5301 \\ 3.8255 \end{bmatrix}$$

use MATLAB to find that the eigenvalues are \(-0.3\) and \(-2.2640\), so the system is stable and the IMC-based PID method for stable systems can be used (Table 7.1). Find the transfer function relating the dilution rate to the biomass concentration, and use this for controller design. You may wish to use the MATLAB function `ss2tf` to find the process transfer function

$$g_p(s) = \frac{-1.5302s - 0.4590}{s^2 + 2.564s + 0.6792}$$

After placing the process model in gain and time constant form and recognizing pole-zero cancellation, you should find

$$g_p(s) = \frac{-0.6758}{0.4417s + 1}$$

Notice from Table 7.1, that the IMC-based controller is a PI controller.
a. Show how the response to a small setpoint change varies with $\lambda$ (show explicitly how the PID tuning parameters vary with $\lambda$). Suggested setpoint changes are from 1.53016 to 1.52 and from 1.53016 to 1.54.

b. For a particular value of $\lambda$, show how the magnitude of the setpoint change affects your response. This is where the nonlinearity comes into play. Try changing from the steady-state value of 1.53016 to 1.0, 1.4, 1.6 and 2.5.

c. Often measurements cannot be made instantaneously, and there will be a transport delay associated with the measurement. Use the transport delay function from the Nonlinear Block (although time-delay is not a nonlinearity) in SIMULINK. Use $\theta = 0.25$ and discuss how the time-delay affects your choice of $\lambda$.

**Problem 2.** Design an IMC-based PID controller to control the bioreactor at equilibrium point 2 - the unstable non-trivial point. The steady-state (also use this as the initial condition for your simulations) is

$$x(0) = \begin{bmatrix} 0.995103 \\ 1.512243 \end{bmatrix}$$

At this point, the state space model is

$$A = \begin{bmatrix} 0 & -0.0679 \\ -0.7500 & -0.1302 \end{bmatrix}$$

$$B = \begin{bmatrix} -0.9951 \\ 2.4878 \end{bmatrix}$$

use MATLAB to find that the eigenvalues are $-0.3$ and $0.169836$, so the system is unstable and the IMC-based PID method for unstable systems must be used (Table 7.3). Find the transfer function relating the dilution rate to the biomass concentration, and use this for controller design. You may wish to use the MATLAB function `ss2tf` to find the process transfer function

$$g_p(s) = \frac{-0.9951s - 0.2985}{s^2 + 0.1302s - 0.0509}$$

After placing the process model in gain and time constant form (and cancelling common poles and zeros), you should find

$$g_p(s) = \frac{5.8644}{-5.888s + 1}$$

That is, the transfer function has a RHP pole at $\frac{1}{5.888} = 0.1698$ which is consistent with the state space model. Notice that we can use the first entry in Table 7.3, which is a PI controller.

a. Show how the response to a small setpoint change varies with $\lambda$ (show explicitly how the PID tuning parameters vary with $\lambda$). I suggest setpoint changes from 0.995103 to 0.985 and 1.005.
b. For a particular value of $\lambda$, show how the magnitude of the setpoint change affects your response. This is where the nonlinearity comes into play. Try changing from the steady-state value of 0.995013 to 0.5, 0.75, 1.5 and 2.0.

c. Use the transport delay function from the Nonlinear block in SIMULINK. Use $\theta = 0.25$ and discuss how the time-delay affects your choice of $\lambda$.

*The bottom line:* You should find that it is very tough to control unstable systems which have measurement time-delays.
APPENDIX. ALGEBRA REQUIRED TO FIND PI CONTROLLER FOR EXAMPLE 7.7

\[
g_c(s) = \frac{q(s)}{1 - f_p(s)q(s)} = \frac{-\tau_s s + 1}{k_p (\lambda s + 1)^2} \frac{\gamma s + 1}{1 - \frac{k_p}{k_p} \frac{-\tau_s s + 1}{-\tau_s s + 1} \frac{\gamma s + 1}{(\lambda s + 1)^2}}
\]

\[
= \frac{1}{k_p} \frac{(-\tau_u s + 1)(\gamma s + 1)}{(\lambda s + 1)^2 - (\gamma s + 1)} = \frac{1}{k_p} \frac{(-\tau_u s + 1)(\gamma s + 1)}{\lambda^2 s^2 + 2\lambda s + 1 - \gamma s - 1}
\]

\[
= \frac{1}{k_p} \frac{(-\tau_u s + 1)(\gamma s + 1)}{\lambda^2 s^2 + (2\lambda - \gamma)s} = \frac{1}{k_p} \frac{(-\tau_u s + 1)(\gamma s + 1)}{\lambda^2 s^2 + (2\lambda - \gamma)s}
\]

\[
g_c(s) = \frac{1}{k_p} \frac{(-\tau_u s + 1)(\gamma s + 1)}{\gamma} \frac{\gamma}{2\lambda - \gamma} \frac{2\lambda - \gamma}{2\lambda - \gamma} = \frac{\gamma}{k_p(2\lambda - \gamma)(-\tau_u s + 1)} \frac{(\gamma s + 1)}{\gamma}
\]

but \( \gamma = \lambda \left( \frac{\tau}{\tau_u} + 2 \right) \)

so

\[
g_c(s) = \frac{\gamma}{k_p(2\lambda - \gamma)(-\tau_u s + 1)} \frac{(\gamma s + 1)}{\gamma} = \frac{\gamma}{k_p(2\lambda - \gamma)(-\tau_u s + 1)} \frac{(\gamma s + 1)}{\gamma}
\]

\[
g_c(s) = \frac{\gamma}{k_p(2\lambda - \gamma)(-\tau_u s + 1)} \frac{(\gamma s + 1)}{\gamma} - \tau u s + 1
\]

which is a PI controller.