7

A Ziegler-Nichols Replacement

7.1 Introduction

Since PID controllers are so common it is useful to have simple tuning rules that can be applied to a wide range of processes. This is testified to by the longevity of the Ziegler-Nichols rules. They have been used for more than half a century even though they have severe drawbacks. In this chapter we present new tuning methods in the spirit of Ziegler and Nichols.

Control system design is a rich problem, as was discussed in Section 4.2. Any design problem should take account of load disturbances, measurement noise, robustness, and set-point following. When developing the simple rules we will follow the main ideas used by Ziegler and Nichols. We will thus focus on load disturbances by maximizing integral gain, but we will depart from Ziegler and Nichols by also adding a robustness constraint. In this chapter we have chosen to require that the joint sensitivity is larger than $M = 1.4$. Measurement noise is handled by detuning the controllers if the gains are too large, and set-point following is dealt with by set-point weighting.

The procedure we use is essentially the same as the one employed by Ziegler and Nichols. We will select a large batch of representative processes. This includes a wide variety of systems with essentially monotone step responses, which are typically encountered in process control. Controllers for each process in the batch are then obtained by applying the MIGO design described in Section 6.8, which is based on the criteria given above. Having obtained the controller parameters we will then try to find correlations with normalized process parameters. The simple tuning rules obtained are called AMIGO, which stands for Approximate MIGO design.

The procedure shows that it is indeed possible to obtain simple tuning formulas. A major result is that it is necessary to use more process information than used by Ziegler and Nichols. Tuning based on step responses can be based on FOTD models. It is necessary to use all *three* process parameters K_p, L, and T and not just two parameters $a = K_pL/T$ and L, as was suggested by Ziegler and Nichols. For PI control it is possible to obtain tuning rules that are close to the optimal rules for the whole test batch. For PID control rules that are close to optimal can be obtained for balanced and delay-dominated processes. For lag-dominated processes it is necessary to have better process information. It is, however, possible to obtain efficient rules for balanced and delay dominant processes.

For the frequency response method where Ziegler and Nichols characterized the process by two parameters, K_{180} and T_{180}, we have shown that it is necessary to add a third parameter, e.g., the static gain K_p. Even with these parameters it is not possible to obtain rules that are close to optimal for all processes in the test batch. It is, however, possible to obtain conservative tuning rules both for PI and PID controllers.

The design method used can give high controller gains for processes that are lag dominated. This may result in large variations in the control signal due to noise. In some cases, it may therefore be necessary to make a trade-off between attenuation of load disturbances and injection of measurement noise. This can be accomplished by detuning the controllers. Methods for doing this are also included in this chapter.

Analysis of all the controllers in the test batch has also given much insight into PI and PID control. It is shown that derivative action only gives moderate improvement for balanced and delay-dominated processes but that very large improvements can be obtained for lag-dominated processes. It is also shown that there is a wide range of processes where it is advantageous to have $T_i < 4T_d$. Notice that controllers implemented on series form do not permit this. Nice formulas that give the ratio of the average residence time for open- and closed-loop systems are also given. This makes it possible to estimate the closed-loop response times that can be expected.

In the next section, the test batch is presented. Using this batch, AMIGO tuning rules based on step response experiments are derived for PI controllers in Section 7.3 and PID controllers in Section 7.4. AMIGO tuning rules based on frequency response experiments are presented in Section 7.5. More efficient tuning rules for PID control of lag-dominant processes can be obtained if a second order model is used. This is discussed in Section 7.6. In Section 7.7, the MIGO and AMIGO rules are compared for three different processes, one lag-dominant, one balanced, and one delay dominant, respectively. Section 7.8 and Section 7.9 treat noise filtering and high-frequency gain reduction of the controllers by detuning.

7.2 The Test Batch

PID control is not suitable for all processes. In [Hägglund and Åström, 2002] it is suggested that the class of processes where PID is suitable can be characterized as having essentially monotone step responses. One way to characterize such processes is to introduce the monotonicity index:

$$\alpha = \frac{\int_0^\infty g(t)dt}{\int_0^\infty |g(t)|dt}, \tag{7.1}$$

where g is the impulse response of the system. Systems with $\alpha = 1$ have monotone step responses, and systems with $\alpha > 0.8$ are considered essentially monotone. The tuning rules presented in this paper are derived using a test batch of essentially monotone processes.

The following 134 processes were used to derive the tuning rules:

$$
\begin{aligned}
P_1(s) &= \frac{e^{-s}}{1+sT},\\
T &= 0.02, 0.05, 0.1, 0.2, 0.3, 0.5, 0.7, 1,\\
&\quad 1.3, 1.5, 2, 4, 6, 8, 10, 20, 50, 100, 200, 500, 1000\\
P_2(s) &= \frac{e^{-s}}{(1+sT)^2},\\
T &= 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.5, 0.7, 1,\\
&\quad 1.3, 1.5, 2, 4, 6, 8, 10, 20, 50, 100, 200, 500\\
P_3(s) &= \frac{1}{(s+1)(1+sT)^2},\\
T &= 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 2, 5, 10\\
P_4(s) &= \frac{1}{(s+1)^n},\\
n &= 3,\ 4,\ 5,\ 6,\ 7,\ 8\\
P_5(s) &= \frac{1}{(1+s)(1+\alpha s)(1+\alpha^2 s)(1+\alpha^3 s)},\\
\alpha &= 0.1, 0.2, 0.3, 0, 4, 0.5, 0.6, 0.7, 0.8, 0.9\\
P_6(s) &= \frac{1}{s(1+sT_1)}e^{-sL_1},\\
L_1 &= 0.01, 0.02, 0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0, \quad T_1 + L_1 = 1\\
P_7(s) &= \frac{T}{(1+sT)(1+sT_1)}e^{-sL_1}, \qquad T_1 + L_1 = 1,\\
T &= 1, 2, 5, 10 \qquad L_1 = 0.01, 0.02, 0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0\\
P_8(s) &= \frac{1-\alpha s}{(s+1)^3},\\
\alpha &= 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1\\
P_9(s) &= \frac{1}{(s+1)((sT)^2 + 1.4sT + 1)},\\
T &= 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0.
\end{aligned}
\tag{7.2}
$$

The processes are representative for many of the processes encountered in process control. The test batch includes both delay-dominated, lag-dominated, and integrating processes. All processes have monotone step responses except P_8 and P_9. The parameters range for processes P_8 and P_9 are chosen so that the systems are essentially monotone with $\alpha \geq 0.8$. The normalized time delay ranges from 0 to 1 for the process P_1 but only from 0.14 to 1 for P_2. Process P_6 is integrating, and therefore $\tau = 0$. The rest of the processes have values of τ in the range $0 < \tau < 0.5$.

7.3 PI Control

The processes in test batch (7.2) are first approximated by the simple FOTD model

$$P(s) = \frac{K_p}{1+sT}e^{-sL}, \tag{7.3}$$

where K_p is the static gain, T the time constant (also called lag), and L the time delay. Processes with integration are approximated by the model

$$P(s) = \frac{K_v}{s}e^{-sL}, \tag{7.4}$$

where K_v is the velocity gain and L the time delay. The model (7.4) can be regarded as the limit of (7.3) as K_p and T go to infinity in such a way that $K_p/T = K_v$ is constant. The parameters of (7.3) and (7.4) can be determined from a step response experiments using the methods presented in Section 2.7.

The tuning rules were obtained in the following way. The MIGO design method (see Section 6.8) with $M = 1.4$ was applied to all processes in the test batch (7.2). This gave the PI controller parameters K and T_i. The AMIGO rules were then obtained by finding relations between the controller parameters and the process parameters.

Figure 7.1 illustrates the relations between the controller parameters and the process parameters for all processes in the test batch. The controller gain is normalized by multiplying it either with the static process gain K_p or with the parameter $a = K_pL/T = K_vL$. The integral time is normalized by dividing it by T or by L. The parameters for the integrating processes P_6 are only normalized with a and L since K_p and T are infinite for these processes. The controller parameters in Figure 7.1 are plotted versus the normalized dead time $\tau = L/(L+T)$.

The figure shows that there is a good correlation between the normalized controller parameters and normalized time delay. This indicates that it is possible to develop good tuning rules based on the FOTD model. Notice, however, that there are significant variations in the parameters with the normalized time delay τ.

Ziegler and Nichols tried to find rules that do not depend on τ. Figure 7.1 shows that the normalized parameters KK_p, aK, T_i/T, and T_i/L vary as much as two orders of magnitude with τ. It is thus not possible to find efficient rules that do not depend on τ.

The solid lines in Figure 7.1 correspond to the AMIGO tuning formula,

$$\begin{aligned} K &= \frac{0.15}{K_p} + \left(0.35 - \frac{LT}{(L+T)^2}\right)\frac{T}{K_pL} \\ T_i &= 0.35L + \frac{13LT^2}{T^2+12LT+7L^2}, \end{aligned} \tag{7.5}$$

and the dotted lines show the limits for 15 percent variations in the controller parameters. Almost all processes included in the test batch fall within these limits.

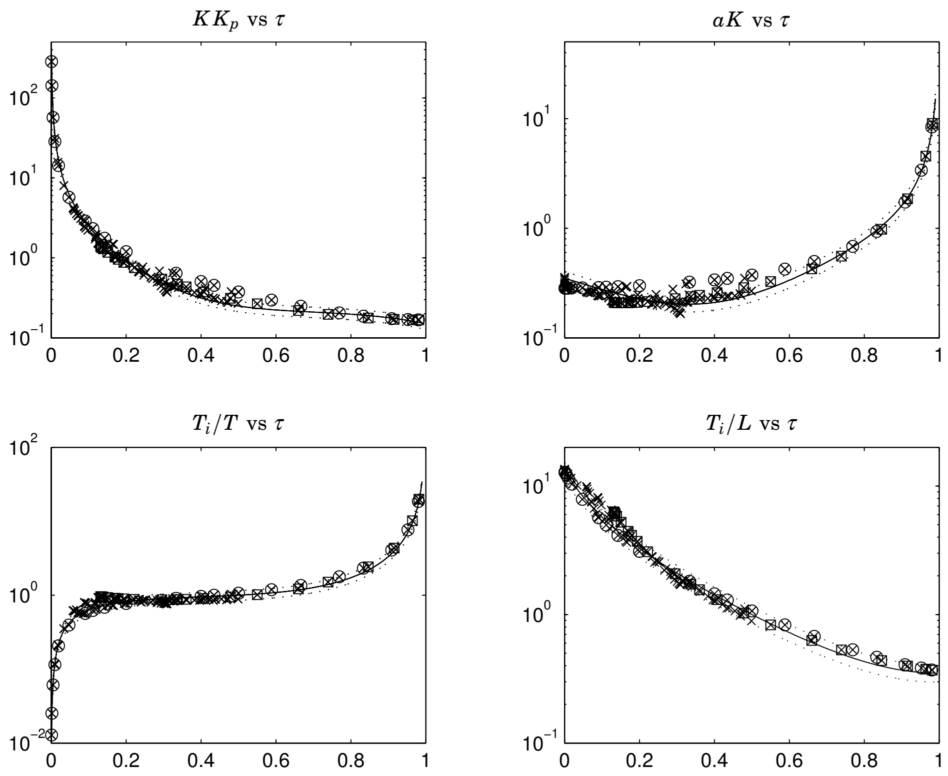

Figure 7.1 Normalized PI controller parameters plotted versus normalized time delay τ. The solid lines correspond to the AMIGO design rule (7.5), and the dotted lines indicate 15 percent parameter variations. The circles mark parameters obtained from process P_1, and squares parameters obtained from process P_2.

For integrating processes, K_p and T go to infinity and $K_p/T = K_v$. Therefore, the AMIGO tuning rules (7.5) can be simplified to

$$\begin{aligned} K &= \frac{0.35}{K_v L} \\ T_i &= 13.4L. \end{aligned} \tag{7.6}$$

for integrating processes.

The tuning rule (7.5) can be seen as a replacement for the Ziegler-Nichols' step response method for PI control. Notice that the rule was designed for the sensitivity $M = 1.4$. Similar rules can be found for other values of the design parameter.

Set-Point Weighting

The MIGO design method also gives suitable values of b. It is determined so that the resonance peak of the transfer function between set point and process output becomes close to one, as discussed in Section 5.3. Figure 7.2 shows the values of the b-parameter for the test batch (7.2).

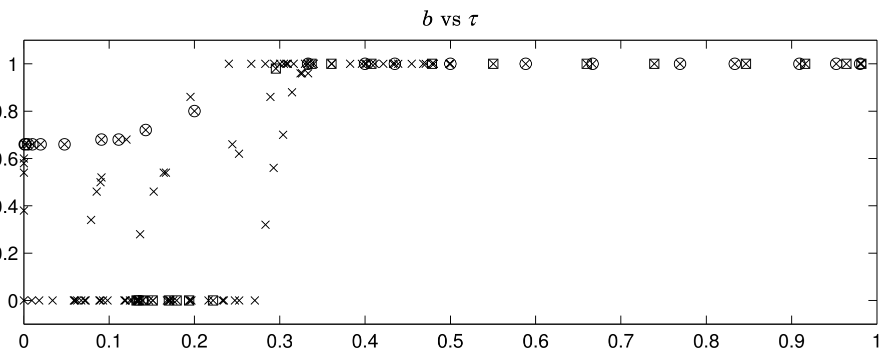

Figure 7.2 Set-point weighting as a function of τ for the test batch (7.2). The circles mark parameters obtained from the process P_1, and the squares mark parameters obtained from the process P_2.

Figure 7.2 shows that the correlation with parameter τ is not as good as for the feedback parameters and that there is a larger difference between the pure FOTD model P_1 and the other processes. The set-point weight should be $b = 1$ for processes with $\tau > 0.3$.

7.4 PID Control

Figure 7.3 illustrates the relations between the PID controller parameters obtained from the MIGO design and the process parameters for all processes in the test batch. The robustness criterion $M = 1.4$ is used together with the additional constraint $\partial k_i/\partial k = 0$; see Section 6.8. The normalized controller parameters in Figure 7.3 are plotted versus the normalized dead time τ.

The figure indicates that the variations of the normalized controller parameters are several orders of magnitude. We can thus conclude that it is not possible to find good universal tuning rules that do not depend on the normalized time delay τ. Recall that Ziegler and Nichols suggested the rules $aK = 1.2$, $T_i = 2L$, and $T_d = 0.5L$. Figure 7.3 shows that these parameters are only suitable for very few processes in the test batch.

The controller parameters for processes P_1 are marked with circles and those for P_2 are marked with squares in Figure 7.3. For $\tau < 0.5$, the gains for P_1 are typically smaller than for the other processes, and the integral time is larger. This is the opposite to what happened for PI control; see Figure 7.1. Process P_2 has a gain that is larger and an integral time that is shorter than for the other processes.

For PI control, it was possible to obtain simple tuning rules, where the controller parameters obtained from the AMIGO rules differed less than 15 percent from those obtained from the MIGO rules for most processes in the test batch. Figure 7.3 indicates that universal tuning rules for PID control can be obtained only for $\tau \geq 0.5$.

For $\tau < 0.5$ there is a significant spread of the normalized parameters. This

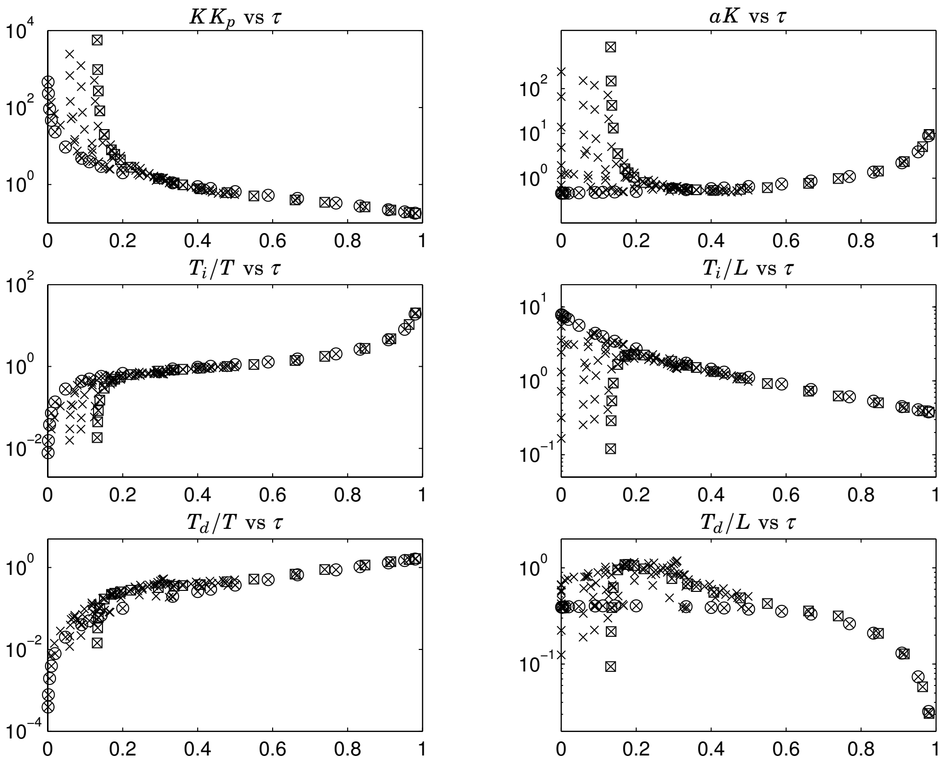

Figure 7.3 Normalized PID controller parameters as a function of the normalized time delay τ. The controllers for the process P_1 are marked with ○ and controllers for P_2 with □.

implies that it is not possible to find universal tuning rules for lag-dominated processes. Notice that the gain and the integral time are well defined for $0.3 < \tau < 0.5$ but that there is a considerable variation of the normalized derivative time in that interval.

Because of the large spread in parameter values for $\tau < 0.5$ it is worthwhile to model the process more accurately to obtain good tuning of PID controllers. The process models (7.3) and (7.4) model stable processes with three parameters and integrating processes with two parameters. In practice, it is very difficult to obtain more process parameters from the simple step response experiment. A step response experiment is thus not sufficient to tune PID controllers with $\tau < 0.5$ accurately.

However, it is possible to find *conservative* tuning rules for $\tau < 0.5$ by choosing controllers with parameters that correspond to the lowest gains and the largest integral times of Figure 7.3. Before developing such rules, we will discuss the reason why universal tuning rules for lag-dominant processes cannot be found.

Problems with FODT Structure

The criterion used is to maximize integral gain k_i. The fundamental limitations are given by the *true* time delay of the process, which we denote L_0. The integral gain is proportional to the gain crossover frequency ω_{gc} of the closed-loop system. The gain crossover frequency ω_{gc} is typically limited to

$$\omega_{gc} L_0 < 0.5.$$

When a process is approximated by the FOTD model the apparent time delay L is longer than the true time delay L_0 because lags are approximated by additional time delays. This implies that the integral gain obtained for the FOTD model will be lower than for a design based on the true model. The situation is particularly pronounced for systems with small τ.

Consider PI control of first-order systems, i.e., processes with the transfer functions

$$P(s) = \frac{K_p}{1+sT} \quad \text{or} \quad P(s) = \frac{K_v}{s}.$$

Since these systems do not have time delays there is no dynamics limitation, and arbitrarily high integral gain can be obtained. Since these processes can be matched perfectly by the models (7.3) and (7.4), the design rule reflects this property. The process parameters are $L = 0$, $a = 0$, and $\tau = 0$, and both the design method MIGO and the approximate AMIGO rule (7.5) give infinite integral gains.

Consider PID control of second-order systems with the transfer functions

$$P(s) = \frac{K_v}{s(1+sT_1)} \quad \text{and} \quad P(s) = \frac{K_p}{(1+sT_1)(1+sT_2)}.$$

Since the systems does not have time delays it is possible to have controllers with arbitrarily high integral gains. The first transfer function has $\tau = 0$. The second process has values of τ in the range $0 \le \tau < 0.13$, where $\tau = 0.13$ corresponds to $T_1 = T_2$. When these transfer functions are approximated with a FOTD model one of the time constants will be approximated with a time delay. Since the approximating model has a time delay there will be limitations in the integral gain.

We can thus conclude that for $\tau < 0.13$ there are processes in the test batch that permit infinitely large integral gains. This explains the wide spread of controller parameters for small τ. The spread is infinitely large for $\tau < 0.13$, and it decreases for larger τ. Therefore, for small τ improved modeling gives a significant benefit.

One way to avoid the difficulty is to use a more complicated model, such as

$$P(s) = \frac{b_1 s + b_2 s}{s^2 + a_1 s + a_2} e^{-sL}.$$

It is, however, very difficult to estimate the parameters of this model accurately from a simple step response experiment. Design rules for models having five parameters may also be cumbersome. Since the problem occurs for small values of τ it may be possible to approximate the process with

$$P(s) = \frac{K_v}{s(1+sT)} e^{-sL},$$

which only has three parameters. Instead of developing tuning rules for more complicated models it may be better to simply compute the controller parameters based on the estimated model.

Conservative Tuning Rules (AMIGO)

Figure 7.3 shows that it is not possible to find optimal tuning rules for PID controllers that are based on the simple process models (7.3) or (7.4). It is, however, possible to find conservative robust tuning rules with lower performance. The rules are close to the MIGO design for the process P_1, i.e., the process that gives the lowest controller gain and the longest integral time; see Figure 7.3.

The suggested AMIGO tuning rules for PID controllers are

$$\begin{aligned} K &= \frac{1}{K_p}\left(0.2 + 0.45\frac{T}{L}\right) \\ T_i &= \frac{0.4L + 0.8T}{L + 0.1T} L \\ T_d &= \frac{0.5LT}{0.3L + T}. \end{aligned} \tag{7.7}$$

For integrating processes, Equation 7.7 can be written as

$$\begin{aligned} K &= 0.45/K_v \\ T_i &= 8L \\ T_d &= 0.5L. \end{aligned} \tag{7.8}$$

Figure 7.4 compares the tuning rule (7.7) with the controller parameters given in Figure 7.3. The tuning rule (7.7) describes the controller gain K well for a process with $\tau > 0.3$. For small τ, the controller gain is well fitted to processes P_1, but the AMIGO rule underestimates the gain for other processes.

The integral time T_i is well described by the tuning rule (7.7) for $\tau > 0.2$. For small τ, the integral time is well fitted to processes P_1, but the AMIGO rule overestimates it for other processes.

The tuning rule (7.7) describes the derivative time T_d well for process with $\tau > 0.5$. In the range $0.3 < \tau < 0.5$ the derivative time can be up to a factor of 2 larger than the value given by the AMIGO rule. If the values of the derivative time for the AMIGO rule are used in this range the robustness is decreased; the value of M may be reduced by about 15 percent. For $\tau < 0.3$, the AMIGO tuning rule gives a derivative time that sometimes is shorter and sometimes longer than the one obtained by MIGO. Despite this, it appears that AMIGO gives a conservative tuning for all processes in the test batch, mainly because of the decreased controller gain and increased integral time.

The tuning rule (7.7) has the same structure as the Cohen-Coon method, but the parameters differ significantly.

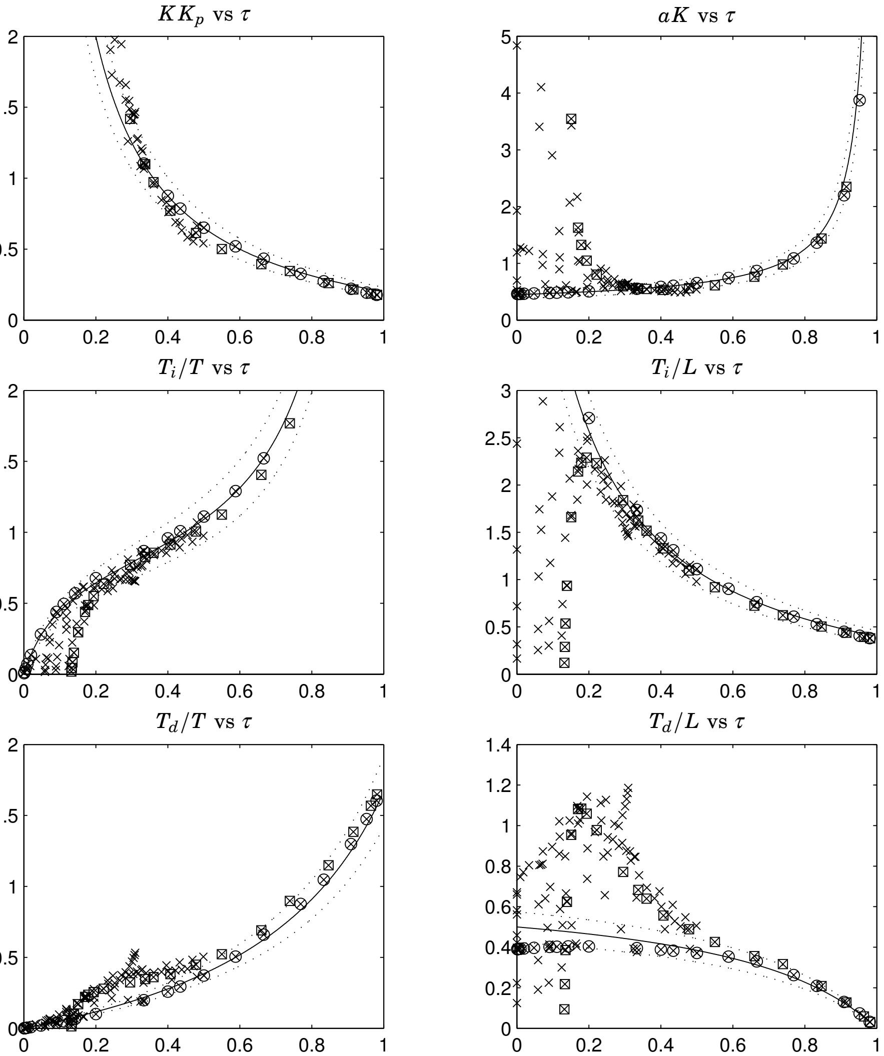

Figure 7.4 Normalized controller parameters as a function of normalized time delay τ. The solid line corresponds to the tuning rule (7.7), and the dotted lines indicate 15 percent parameter variations. The circles mark parameters obtained from the process P_1, and the squares mark parameters obtained from the process P_2.

Robustness

Figure 7.5 shows the Nyquist curves of the loop transfer functions obtained when the processes in the test batch (7.2) are controlled with the PID controllers tuned with the conservative AMIGO rule (7.7). When using MIGO all Nyquist curves are outside the M-circle. With AMIGO there are some processes where the Nyquist curves are inside the circle. An investigation shows that the derivative action is too small in these cases; compare with the curves of T_d/L vs τ in Figure 7.4. The increase of M is at most about 15 percent

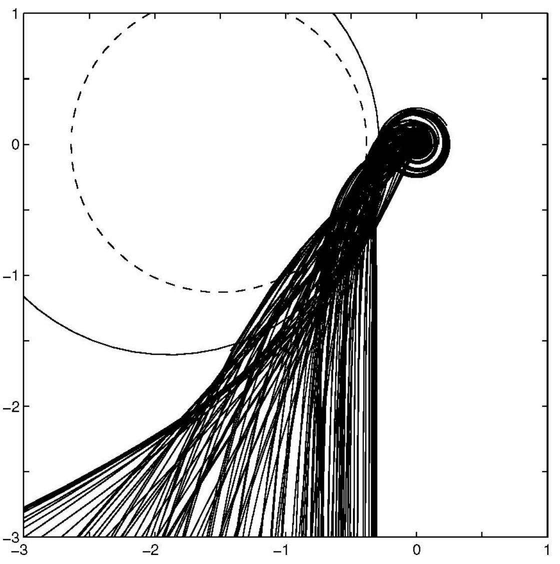

Figure 7.5 Nyquist curves of loop transfer functions obtained when PID controllers tuned according to (7.7) are applied to the test batch (7.2). The solid circle corresponds $M = 1.4$, and the dashed to a circle where M is increased by 15 percent.

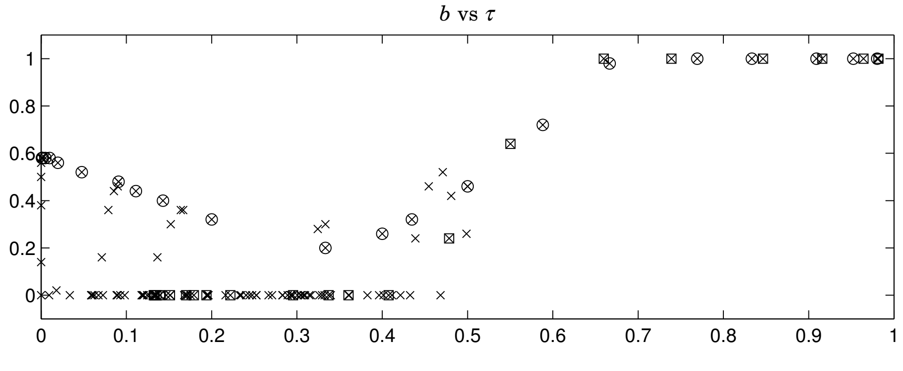

Figure 7.6 Set-point weighting as a function of τ for the test batch (7.2). The circles mark parameters obtained from the process P_1, and the squares mark parameters obtained from the process P_2.

with the AMIGO rule. If this increase is not acceptable derivative action can be increased or the gain can be decreased with about 15 percent.

Set-Point Weighting

Figure 7.6 shows the values of the b-parameter for the test batch (7.2).

The correlation between b and τ is not so good, but a conservative and simple rule is to choose b as

$$b = \begin{cases} 0, & \text{for } \tau \leq 0.5 \\ 1, & \text{for } \tau > 0.5. \end{cases} \tag{7.9}$$

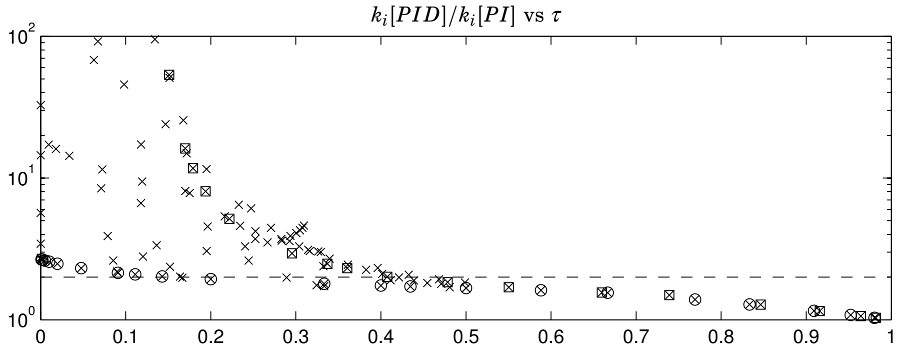

Figure 7.7 The ratio of integral gain with PID and PI control as a function of normalized time delay τ. The dashed line corresponds to the ratio $k_i[PID]/k_i[PI] = 2$. The controllers for the process P_1 are marked with circles and controllers for P_2 with squares.

The Benefits of Derivative Action

Since maximization of integral gain was chosen as design criterion we can judge the benefits of derivative action by the ratio of integral gain for PID and PI control. Figure 7.7 shows this ratio for the test batch, except for a few processes with a high ratio at small values of τ.

The figure shows that the benefits of derivative action are marginal for delay-dominated processes but that the benefits increase with decreasing τ. For $\tau = 0.5$ the integral gain can be doubled, and for values of $\tau < 0.15$ integral gain can be increased arbitrarily for some processes.

The Ratio T_i/T_d

The ratio T_i/T_d is of interest for several reasons. It is a measure of the relative importance of derivative and integral action. Many PID controllers are implemented in series form, which requires that the ratio be larger than 4. Many classical tuning rules therefore fix the ratio to 4. Figure 7.8 shows the ratio for the full test batch. The figure shows that there is a significant variation in the ratio T_i/T_d, particularly for small τ. The ratio is close to 2 for $0.5 < \tau < 0.9$, and it increases to infinity as τ approaches 1 because the derivative action is zero for processes with pure time delay.

Figure 7.8 also shows the ratio obtained by the AMIGO tuning rule (7.7). The ratio is less than four for processes with $0.3 < \tau < 0.9$, which means that the tuning rule cannot be used for controllers in series form for these processes. However, it appears that the changes of performance and robustness are marginal if the tuning rule (7.7) is modified so that $T_d = T_i/4$ for these processes. Figure 7.9 shows the Nyquist curves of the loop transfer functions obtained when the processes in the test batch with $0.3 < \tau < 0.9$ are tuned such that gain K and integral time T_i are obtained from (7.7), and the derivative time is obtained as $T_d = T_i/4$. The figure shows that the robustness is about the same as for (7.7); compare with Figure 7.5.

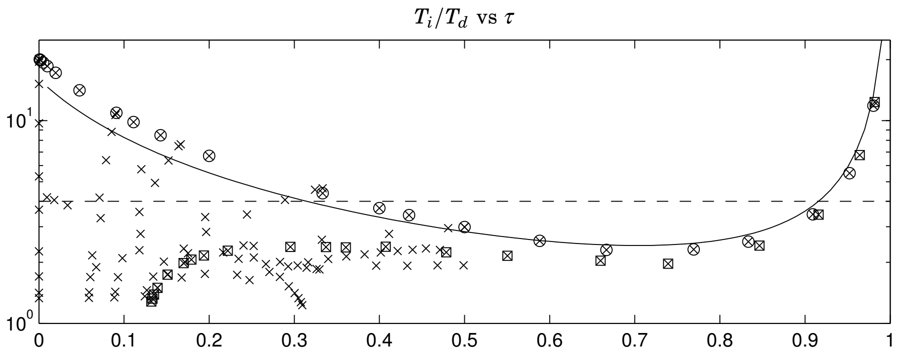

Figure 7.8 The ratio between T_i and T_d as a function of normalized time delay τ. Process P_1 is marked with circles and process P_2 with squares. The dashed line corresponds to the ratio $T_i/T_d = 4$, and the solid line to the ratio given by the AMIGO tuning rule (7.7).

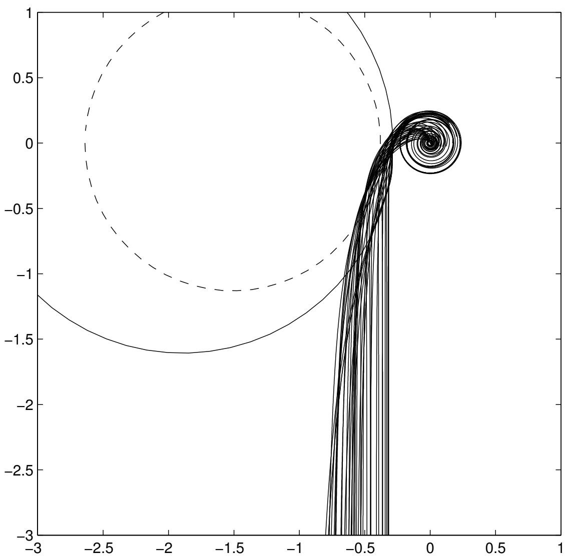

Figure 7.9 Nyquist curves of the loop transfer functions obtained from the processes in the test batch with $0.3 < \tau < 0.9$ when the controller is tuned with $T_d = T_i/4$.

The Average Residence Time

The parameter T_{63}, which is the time when the step response has reached 63 percent, a factor of $(1-1/e)$, of its steady-state value, is a reasonable measure of the response time for stable systems. It is easy to determine the parameter by simulation, but not by analytical calculations. For the FOTD process we have $T_{ar} = T_{63}$. The average residence time T_{ar} is in fact a good estimate of T_{63} for systems with essentially monotone step response. For all stable processes in the test batch we have $0.99 < T_{63}/T_{ar} < 1.08$.

The average residence time is easy to compute analytically. Consider the closed-loop system obtained when a process with transfer function $P(s)$ is con-

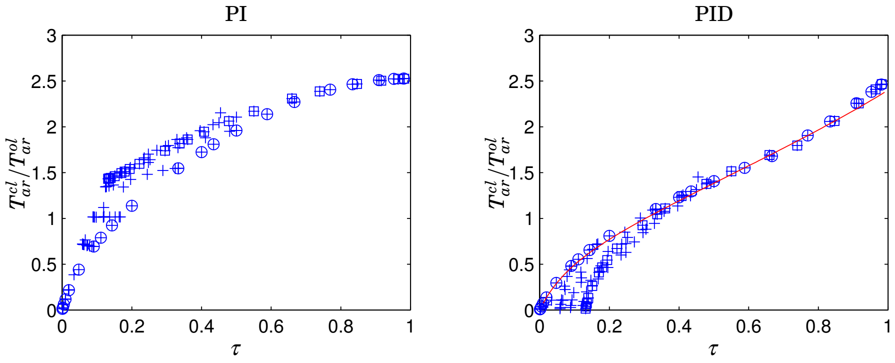

Figure 7.10 The ratio of the average residence time of the closed loop system and the open loop system for PI control left and PID control right.

trolled with a PID controller with set-point weighting. The closed-loop transfer function from set point to output is

$$G_{yy_{sp}}(s) = \frac{P(s)C_{ff}(s)}{1+P(s)C(s)},$$

where

$$C_{ff}(s) = bK + \frac{k_i}{s}.$$

Straightforward but tedious calculations give

$$T_{ar} = -\frac{G'_{yy_{sp}}(0)}{G_{yy_{sp}}(0)} = T_i\Big(1 - b + \frac{1}{KK_p}\Big). \tag{7.10}$$

Figure 7.10 shows the average residence times of the closed-loop system divided with the average residence time of the open-loop system. Figure 7.10 shows that for PID control the closed-loop system is faster than the open-loop system when $\tau < 0.3$ and slower for $\tau > 0.3$.

7.5 Frequency Response Methods

In this section we will investigate if it is possible to obtain simple tuning rules similar to the Ziegler-Nichols frequency response method.

Parameterization

Ziegler-Nichols characterized the processes by two parameters K_{180} and T_{180} when they developed their frequency response method for controller tuning. The Ziegler-Nichols tuning rules do not use sufficient information, and they give too aggressive tuning, which does not give robust closed-loop systems.

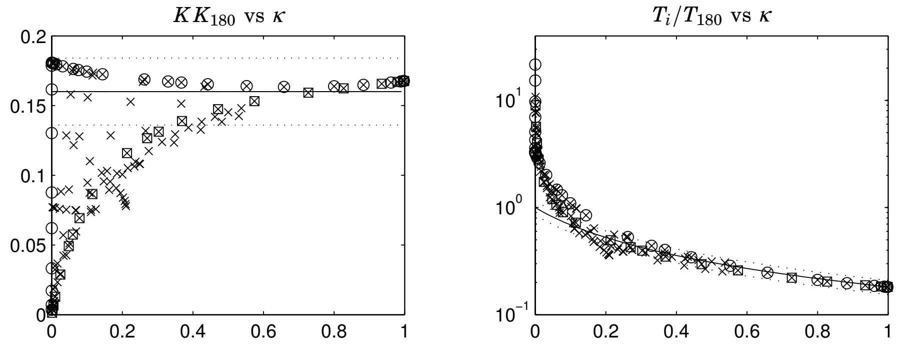

Figure 7.11 Normalized controller parameters plotted versus gain ratio κ for stable processes for $M = 1.4$. The solid lines correspond to the tuning rule (7.11), and the dotted lines indicate 15 percent variations from the rule. The circles mark data from process P_1, and squares data from P_2.

When investigating the step response method it was found that significant improvement could be obtained by including an additional third process parameter, the static process gain. In this section it will be investigated if similar improvements can be obtained for the frequency domain method.

For the step response method we used the normalized time delay τ as a parameter to characterize the process. The corresponding frequency domain parameter is the gain ratio $\kappa = K_{180}/K_p$.

PI Tuning Rules for Balanced and Delay-Dominated Processes

The MIGO design method has been applied to all processes in the test batch (7.2). Figure 7.11 shows the controller parameters obtained for $M = 1.4$. The figure shows that there is a significant spread of controller parameters for lag-dominant processes.

The Ziegler-Nichols tuning rules have constant values $KK_{180} = 0.4$ and $T_i/T_{180} = 0.8$, for all values of κ. Figure 7.11 shows that it may be reasonable to have a constant value KK_{180} for $\kappa > 0.5$, but not for smaller values of κ. The gain $KK_{180} = 0.4$ suggested by Ziegler and Nichols is clearly too high, which explains the poor robustness of their method. The integral time suggested by Ziegler and Nichols, $T_i = 0.8T_{180}$, is too high except for processes with very small values of κ.

Figure 7.11 shows that it is not possible to capture all data by one tuning rule. It may, however, be possible to obtain a rule for balanced and delay-dominated processes. Figure 7.11 shows the graphs corresponding to the following tuning rule.

$$\begin{aligned} KK_{180} &= 0.16 \\ \frac{T_i}{T_{180}} &= \frac{1}{1+4.5\kappa}. \end{aligned} \tag{7.11}$$

The tuning rule (7.11) is not appropriate for lag-dominant processes, but it gives controller parameters that are fairly close to the optimal for processes with $\kappa > 0.2$. Notice in particular that the ratio T_i/T_{180} is reduced by a factor

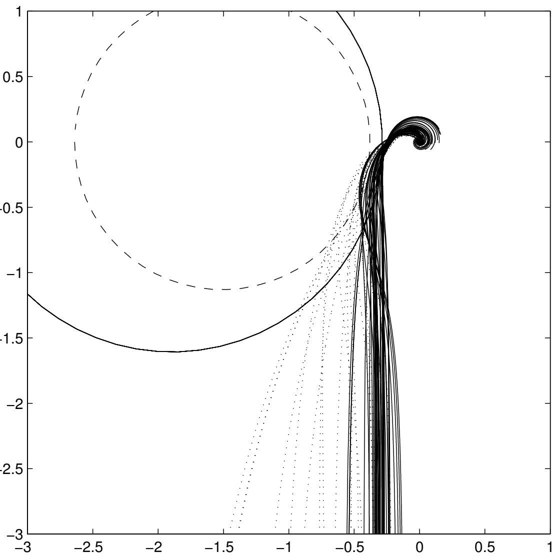

Figure 7.12 Nyquist curves of loop transfer functions obtained when PI controllers tuned according to (7.11) are applied to processes in the test batch with $\kappa > 0.1$. Transfer functions corresponding to processes with $0.1 < \kappa < 0.2$ are shown with dotted lines. The solid circle corresponds to $M = 1.4$ and the dashed to a circle where M is increased by 15 percent.

of three when κ increases from 0.2 to 1.

Figure 7.12 shows the Nyquist curves obtained for all processes in the test batch with $\kappa > 0.1$ when the tuning rule (7.11) is used. The figure shows that all loop transfer functions are close to the M-circle.

PID Tuning Rules for Balanced and Delay-Dominated Processes

Parameters of PID controllers for all the processes in the test batch (7.2) were computed using the MIGO design with the constraints described in the previous section. The design parameter was chosen to $M = 1.4$.

Figure 7.13 illustrates the relations between the controller parameters obtained from the MIGO design and the process parameters for all processes in the test batch.

The controller parameters for processes P_1 are marked with circles, and those for P_2 are marked with squares in Figure 7.13. For $\kappa < 0.3$, the gain for P_1 is typically smaller than for the other processes, and the integral time is larger. This is opposite to what happened for PI control. Process P_2 has a gain that is larger and an integral time that is shorter than for most other processes.

The figure indicates that the variations of the normalized controller parameters are more than an order of magnitude. Therefore, it is not possible to find good universal tuning rules that do not depend on the gain ratio κ. Ziegler and Nichols suggested the rule $KK_{180} = 0.6$, $T_i/T_{180} = 0.5$, and $T_d/T_{180} = 0.125$. The rule is indicated by the dashed lines in Figure 7.13. The Ziegler-Nichols

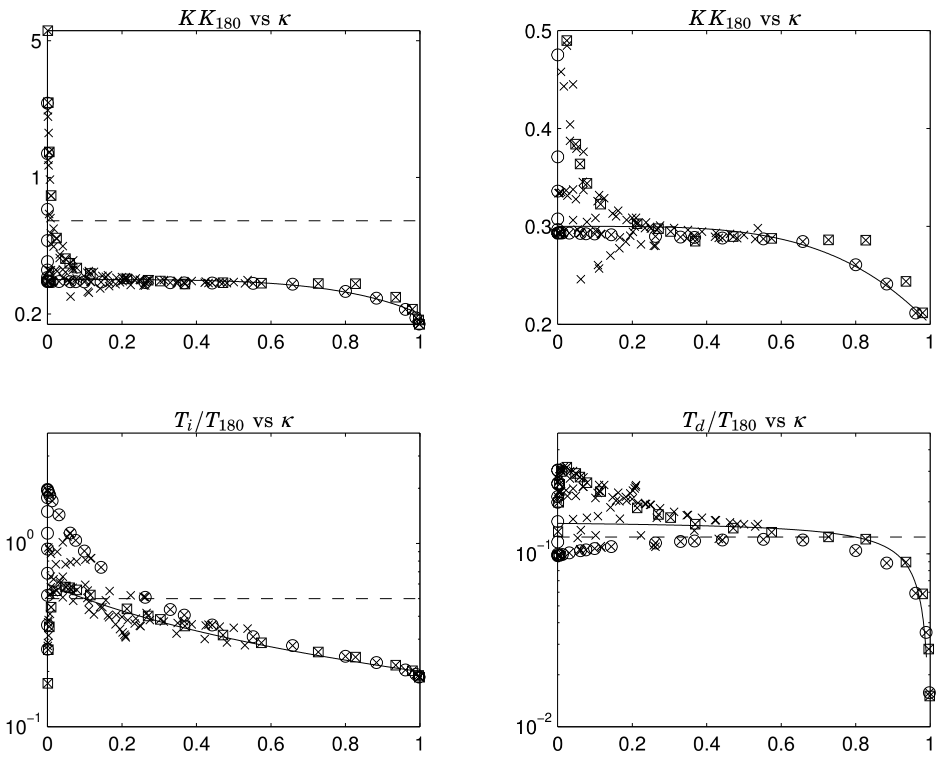

Figure 7.13 Normalized PID controller parameters as a function of the gain ratio κ. Parameters obtained for process P_1 are marked with circles, and parameters obtained for process P_2 with squares. The dashed lines indicate Ziegler-Nichols' tuning rule and the solid lines corresponds to the rule (7.12).

rule is only suitable for very few processes in the test batch. The controller gain is too high except for some processes with very small values of κ.

Even if Figure 7.13 indicates that it is not possible capture all data by one tuning rule it is clear that a good tuning rule can be found for balanced and delay-dominated processes. Figure 7.13 shows the graphs corresponding to the following tuning rule as solid lines.

$$\begin{aligned} K &= (0.3 - 0.1\kappa^4)/K_{180} \\ T_i &= \frac{0.6}{1+2\kappa} T_{180} \\ T_d &= \frac{0.15(1-\kappa)}{1-0.95\kappa} T_{180}. \end{aligned} \tag{7.12}$$

The tuning rule (7.12) is not appropriate for lag-dominant processes, but it gives controller parameters that are fairly close to the optimal for processes with $\kappa > 0.2$. Figure 7.14 shows the Nyquist curves obtained for all processes in the test batch with $\kappa > 0.1$ when the tuning rule (7.12) is used. The figure shows that all loop transfer functions remain fairly close to the M-circle.

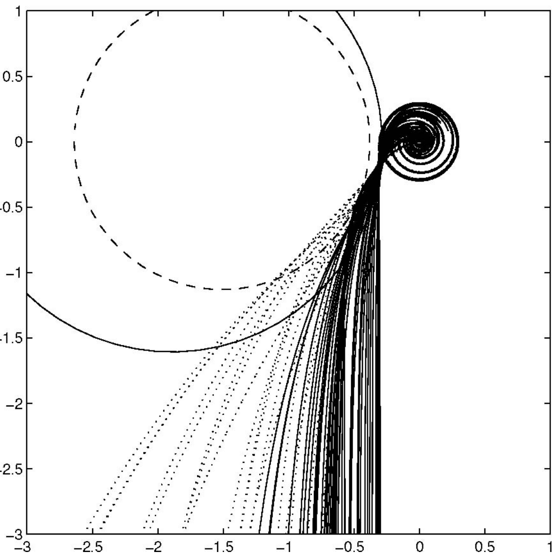

Figure 7.14 Nyquist curves of loop transfer functions obtained when PID controllers tuned according to (7.12) are applied to processes in the test batch with $\kappa > 0.1$. Transfer functions corresponding to processes with $0.1 < \kappa < 0.2$ are shown with dotted lines. The solid circle corresponds to $M = 1.4$, and the dashed to a circle where M is increased by 15 percent.

7.6 PID Control Based on Second-Order Model

In this section, tuning rules based on the SOTD model (2.47) are presented. The SOTD model may be obtained using the combined step and frequency response method presented in Section 2.7.

Figure 7.15 shows controller parameters K, $k_i = K/T_i$, and $k_d = KT_d$ for all processes in the test batch except the integrating process, plotted against the normalized time delay $\tau_1 = L_1/T_{63}$. Figures 7.16 shows the controller parameters T_i and T_d with different normalizations. Notice that the scales are also different. A comparison with Figure 7.3, where the simpler FOTD model is used, shows a significant improvement, particularly for small normalized time delays. This is not surprising because the achievable performance is primarily given by the time delay, and the improvement is mainly due to improved estimates of the true time delay.

The figures show that there is a considerable span of the parameter values. In Figure 7.15 the parameters KL_1/T_{63}, k_iT_{63}, and k_iL_1 range over two decades. The range of variation is larger for other normalizations; for example, the parameter KK_p ranges over five decades. Also notice that there is a spread in the values, particularly in k_d and T_d. This means that we cannot expect to find nice formulas where the normalized parameters are functions only of τ_1.

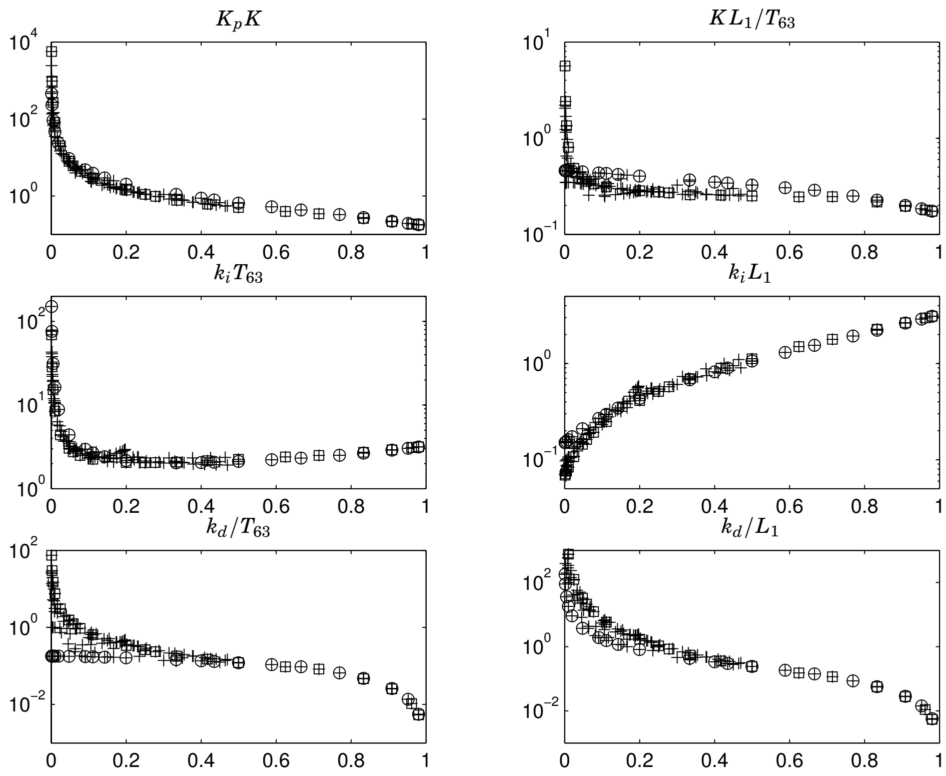

Figure 7.15 Normalized controller parameters K, k_i, and k_d for the processes in the test batch plotted versus τ_1. Data for the processes P_1 are marked with circles, and those for P_2 with squares.

Structure of Tuning Formulas

To get insight into suitable parameterizations we will consider some special systems.

For delay-dominated processes where $L_1 \gg T_1 > T_2$ the model (2.47) can be approximated by

$$P(s) = K_p e^{-sL_1}.$$

Derivative action cannot be used for this process. Designing a PI controller for the process we find

$$C(s) = K + \frac{k_i}{s} = \frac{0.1677}{K_p} + \frac{0.4618}{sL_1K_p},$$

where the numerical values are given for design with $M = 1.4$. Neglecting the time delay and using the numerical values of the controller parameters we find that the closed-loop system is of first order with the pole $sL_1 = -0.4$.

If the process dynamics is a time delay with a small lag,

$$P(s) = \frac{K_p}{1+sT}e^{-sL},$$

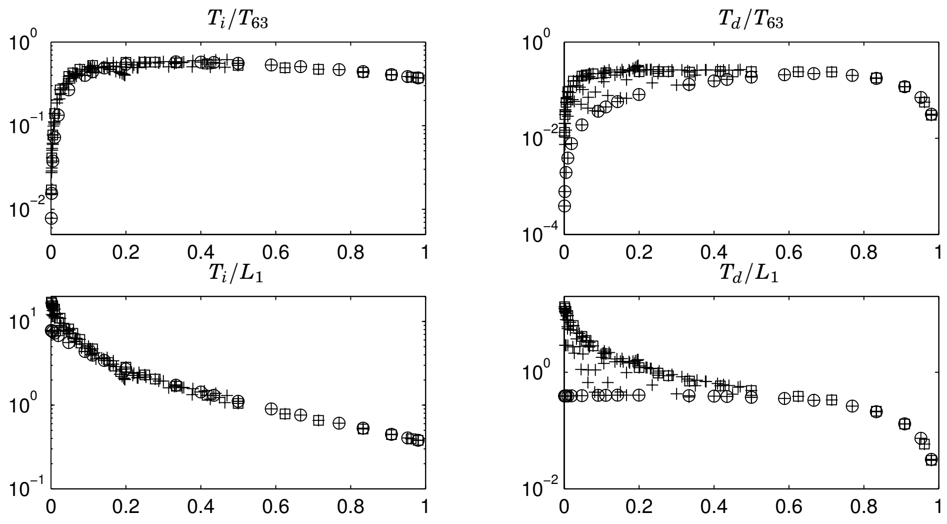

Figure 7.16 Normalized controller parameters T_i and T_d for the processes in the test batch plotted versus τ_1. Data for the processes P_1 is marked with circles, and those for P_2 with squares.

we find that the loop transfer functions under PID control with derivative gain k_d approaches

$$G_l(s) \approx \frac{k_d K_p}{T} e^{-sL}.$$

The Nyquist curve of this transfer function is a circle around the origin with radius $k_d K_p/T$. The design criterion using the combined sensitivity requires that the radius is less than $(M-1)/M$. The largest permissible derivative gain is thus

$$k_d = \frac{T}{K_p}\frac{M-1}{M}.$$

For delay-dominated processes the derivative time is thus proportional to the lag.

The PID controller for the process

$$P(s) = \frac{K_p}{sT} e^{-sL_1}$$

is

$$C(s) = K + \frac{k_i}{s} + k_d s = \frac{0.4603T}{K_p L_1} + \frac{0.05841T}{sK_p L_1^2} + \frac{0.1796sT}{K_p},$$

where the numerical values are given for design with $M = 1.4$. Neglecting the time delay and using the numerical values of the controller parameters we find that the closed loop system is of second order with the poles $sL_1 = -0.2 \pm 0.11i$; the dominant pole is thus $\omega_d = 0.2$.

The PID controller for the process

$$P(s) = \frac{K_p}{s^2 T_1 T_2} e^{-sL_1}$$

is

$$C(s) = K + \frac{k_i}{s} + k_d s = \frac{0.02140 T_1 T_2}{K_p L_1^2} + \frac{0.001218 T_1 T_2}{K_p L_1^3 s} + \frac{0.3 T_1 T_2 s}{K_p L_1},$$

where the numerical values are given for design with $M = 1.4$. Neglecting the time delay and using the numerical values of the controller parameters we find that the closed-loop system is of third order with the poles $sL_1 = -0.23$ and $sL_1 = 0.035 \pm 0.064i$; the dominant pole is thus $\omega_d = 0.07$.

Parameterization

Based on the special cases given above it is reasonable to try tuning formulas having the form

$$\begin{aligned} K_p K &= \alpha_1 + \alpha_2 \frac{T_1}{L_1} + \alpha_3 \frac{T_2}{L_1} + \alpha_4 \frac{T_1 T_2}{L_1^2} \\ K_p k_i &= \beta_1 \frac{1}{L_1} + \beta_2 \frac{T_1}{L_1^2} + \beta_3 \frac{T_2}{L_1^2} + \beta_4 \frac{T_1 T_2}{L_1^3} \\ K_p k_d &= \Big(\gamma_1 L_1 + \gamma_2 T_1 + \gamma_3 T_2 + \gamma_4 \frac{T_1 T_2}{L_1}\Big) \frac{T_1 + T_2}{T_1 + T_2 + L_1}. \end{aligned} \tag{7.13}$$

This will match the controllers for the special cases. The coefficients of proportional and integral gain are simply obtained by adding the coefficients for the prototype processes. Because of the structure of the formula this will automatically give an interpolation between processes with pure delay and double integrator with delay. This procedure will not work for the derivative gain. In this case we have simply taken the weighted average with weights L_1 and $T_1 + T_2$.

Making a least squares fit of the parameters in (7.13) using the parameters of the test batch gives the results in Table 7.1.

Final Parameters

It seems reasonable to make the following approximations.

$$\begin{aligned} &\alpha_1 = 0.19 \quad \alpha_2 = 0.37 \quad \alpha_3 = 0.18 \quad \alpha_4 = 0.02 \\ &\beta_1 = 0.48 \quad \beta_2 = 0.03 \quad \beta_3 = -0.0007 \quad \beta_4 = 0.0012 \\ &\gamma_1 = 0.29 \quad \gamma_2 = 0.16 \quad \gamma_3 = 0.20 \quad \gamma_4 = 0.28. \end{aligned} \tag{7.14}$$

The parameters β_3 and β_4 are quite small. This means that integral gain is essentially determined by parameters K_p, T_1, and L_1. This explains why there is a good correlation in the data for integral gain in Figure 7.15. The correlation for proportional gain in Figure 7.15 is good but not as good as for k_i, because the parameters α_3 and α_4 are larger. The correlation is poor for k_d because parameters γ_3 and γ_4 are large.

Integrating Processes

To investigate that the formulas also work for integrating processes we will investigate the process P_6. A model for integrating processes can be obtained

Table 7.1 Parameters fitted to the tuning formula for different data sets; $P^\dagger$ denotes all processes except the integrating process P_6.

Par	P_1	P_2	P_1, P_2	$P^\dagger$	e^{-s}	e^{-s}/s	e^{-s}/s^2
α_1	0.1755	0.1815	0.1823	0.1903	0.1677	-	-
α_2	0.4649	−0.0215	0.4607	0.3698	-	0.4603	-
α_3	0	0.6816	0.0930	0.1777	-	-	-
α_4	0	0.0210	0.0211	0.0196	-	-	0.02140
β_1	0.5062	0.4613	0.4800	0.4767	0.4618	-	-
β_2	0.0587	−0.2028	0.0596	0.0310	-	0.05841	-
β_3	0	0.2877	−0.0367	0.0017	-	-	-
β_4	0	0.0013	0.0013	0.0012	-	-	0.001218
γ_1	0.3026	0.2864	0.2971	0.2918	-	-	-
γ_2	0.1805	0.0590	0.1814	0.1654	-	0.1796	-
γ_3	0	0.2464	0.0814	0.2033	-	-	-
γ_4	0	0.3090	0.3096	0.2772	-	-	0.3

by taking the limit of

$$P(s) = \frac{K_p}{(1+sT_1)(1+sT_2)} e^{-sL_1}$$

as K_p and T_1 goes to infinity in such a way that $K_p/T_1 = K_v$. The model then becomes

$$P(s) = \frac{K_v}{s(1+sT_2)} e^{-sL_1}.$$

The tuning formula (7.13) becomes

$$\begin{aligned} K_v K &= \alpha_2 \frac{1}{L_1} + \alpha_4 \frac{T_2}{L_1^2} \\ K_v k_i &= \beta_2 \frac{1}{L_1^2} + \beta_4 \frac{T_2}{L_1^3} \\ K_v k_d &= \gamma_2 + \gamma_4 \frac{T_2}{L_1}. \end{aligned} \tag{7.15}$$

Validation

Figure 7.17 shows the Nyquist curves of the loop transfer functions obtained when the processes in the test batch (7.2) are controlled with the PID controllers tuned with the rules (7.13), (7.15), and (7.14). When using MIGO all Nyquist curves are outside the M-circle in the figure. With the approximative rule there are some processes where the Nyquist curves are inside the circle. The increase of M is, however, less than 15 percent for all processes in the test batch.

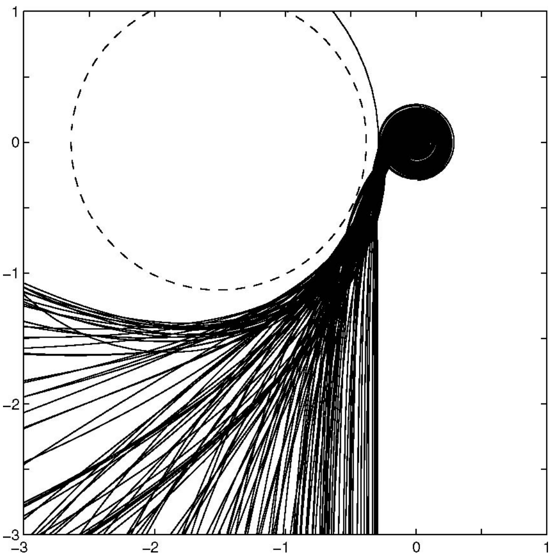

Figure 7.17 Nyquist curves of loop transfer functions obtained when PID controllers tuned according to (7.13), (7.15), and (7.14) are applied to the test batch (7.2). The solid circle corresponds $M = 1.4$, and the dashed to a circle where M is increased by 15 percent.

7.7 Comparison of the Methods

This section presents a few examples that illustrate the AMIGO method and compares it with the MIGO designs for PI and PID controllers. Three examples are given, one with a lag-dominant process, one with a delay-dominant process, and one with a process with balanced lag and delay.

EXAMPLE 7.1—LAG-DOMINATED DYNAMICS
Consider a process with the transfer function

$$P(s) = \frac{1}{(1+s)(1+0.1s)(1+0.01s)(1+0.001s)}.$$

Fitting the model (7.3) to the process we find that the apparent time delay and time constants are $L = 0.075$ and $T = 1.04$, which gives $\tau = 0.067$. The dynamics are thus lag dominated. The corresponding frequency response data needed for the AMIGO design are $K_{180} = 0.0091$ and $T_{180} = 0.199$. Since the static gain is $K_p = 1$, the gain ratio becomes $\kappa = K_{180}/K_p = 0.0091$. Since the process is lag dominant with $\kappa < 0.1$, the AMIGO rules based on frequency response data cannot be used for this process. Fitting the second-order model (2.47) gives the parameters $T_1 = 0.980$, $T_2 = 0.108$, and $L_1 = 0.010$.

The controller parameters obtained from the MIGO and AMIGO tuning rules are presented in Table 7.2.

Figure 7.18 shows the responses of the system to changes in set point and load disturbances when the controllers are tuned with the MIGO and AMIGO design. The figure shows that the AMIGO rule gives responses that are close

Table 7.2 Controller parameters obtained from the MIGO and AMIGO tuning rules for the lag-dominant process in Example 7.1.

Controller	Design	K	T_i	T_d	b	k_i
PI	MIGO	3.56	0.660		0	5.39
	AMIGO–step	4.13	0.539		0	7.66
PID	MIGO	56.9	0.115	0.0605	0	495
	AMIGO–step	6.44	0.361	0.0367	0	17.8
	AMIGO–step+frequency	59.6	0.127	0.0523	0	468

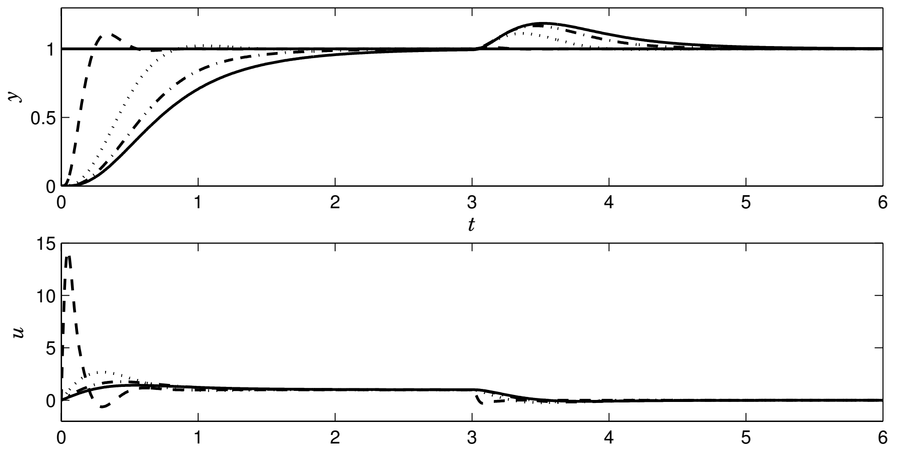

Figure 7.18 Responses to a unit step change at time 0 in set point and a unit load step at time 3 for PI controllers designed by MIGO (solid line) and AMIGO (dash-dotted line), and PID controllers designed by MIGO (dashed line) and AMIGO-step (dotted line) for the lag-dominant process in Example 7.1.

to the MIGO rule for PI control. However, since this is a lag-dominant process, the AMIGO tuning rule for PID control is conservative compared to the MIGO rule. This is obvious in the figure.

The responses obtained using the SOTD model are not presented in the figure, but Table 7.2 shows that the controller parameters are close to the MIGO design.

Notice that the magnitudes of the control signals are about the same at load disturbances, but that there is a major difference in the response time. The differences in the responses clearly illustrate the importance of reacting quickly.

The example shows that derivative action can give drastic improvements in performance for lag-dominated processes. It also demonstrates that the control performance can be increased considerably by obtaining better process models than (7.3). □

Table 7.3 Controller parameters obtained from the MIGO and AMIGO tuning rules for the process with balanced lag and delay in Example 7.2.

Controller	Design	K	T_i	T_d	b	k_i
PI	MIGO	0.432	2.43		1	0.178
	AMIGO–step	0.414	2.66		0	0.156
	AMIGO–frequency	0.640	2.96			0.216
PID	MIGO	1.19	2.22	1.21	0	0.536
	AMIGO–step	1.12	2.40	0.619	0	0.467
	AMIGO–frequency	1.20	2.51	0.927		0.478
	AMIGO–step+frequency	1.15	2.17	1.32	0	0.506

Next we will consider a process where the lag and the delay are balanced.

EXAMPLE 7.2—BALANCED LAG AND DELAY
Consider a process with the transfer function

$$P(s) = \frac{1}{(s+1)^4}.$$

Fitting the model (7.3) to the process we find that the apparent time delay and time constants are $L = 1.42$ and $T = 2.90$. Hence, $L/T = 0.5$ and $\tau = 0.33$. The frequency response data needed for the AMIGO design are $K_{180} = 0.250$ and $T_{180} = 6.28$. The gain ratio becomes $\kappa = K_{180}/K_p = 0.25$. Fitting the second-order model (2.47) gives the parameters $T_1 = 1.73$, $T_2 = 1.73$, and $L_1 = 1.05$.

The controller parameters obtained from the MIGO and AMIGO tuning rules are presented in Table 7.3.

Figure 7.19 shows the responses of the system to changes in set point and load disturbances when the MIGO and AMIGO-step designs are used. The figure shows that the load disturbance responses obtained by MIGO and AMIGO are quite similar, which can be expected because of the similarity of the controller parameters. The difference in set-point response between the MIGO and AMIGO design is caused by the different set-point weightings b of the two designs.

The integral gain k_i is about three times higher for PID control than for PI control. This is in accordance with Figure 7.7. □

Finally, we will consider an example where the dynamics are dominated by the time delay.

EXAMPLE 7.3—DELAY-DOMINATED DYNAMICS
Consider a process with the transfer function

$$P(s) = \frac{1}{(1+0.05s)^2}e^{-s}.$$

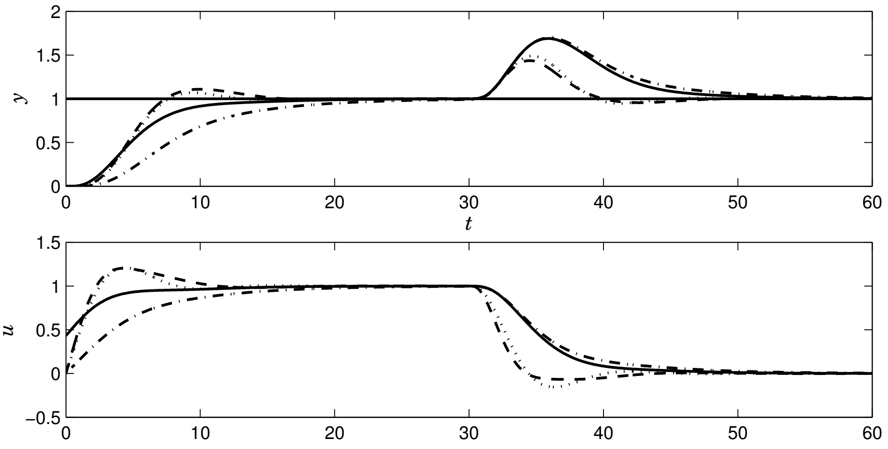

Figure 7.19 Responses to a unit step change at time 0 in set point and a unit load step at time 30 for PI controllers designed by MIGO (solid line) and AMIGO-step (dash-dotted line), and PID controllers designed by MIGO (dashed line) and AMIGO-step (dotted line) for the process with balanced lag and delay in Example 7.2.

Table 7.4 Controller parameters obtained from the MIGO and AMIGO tuning rules for the delay-dominant process in Example 7.3.

Controller	Design	K	T_i	T_d	b	k_i
PI	MIGO	0.170	0.404		1	0.421
	AMIGO–step	0.175	0.360		1	0.486
	AMIGO–frequency	0.163	0.407			0.400
PID	MIGO	0.216	0.444	0.129	1	0.486
	AMIGO–step	0.242	0.474	0.119	1	0.511
	AMIGO–frequency	0.212	0.446	0.0957		0.475
	AMIGO–step+frequency	0.218	0.453	0.129	1	0.481

Approximating the process with the model (7.3) gives the process parameters $L = 1.01$, $T = 0.0932$, and $\tau = 0.92$. The large value of τ shows that the process is delay dominated. The frequency response data needed for the AMIGO design is $K_{180} = 0.980$ and $T_{180} = 2.20$. The gain ratio becomes $\kappa = K_{180}/K_p = 0.98$. The process has the same structure as (2.47) so the parameters of this model become $T_1 = T_2 = 0.05$, and $L_1 = 1$.

The controller parameters obtained from the MIGO and AMIGO tuning rules are presented in Table 7.4.

Figure 7.20 shows the responses of the system to changes in set point and load disturbances. The responses obtained from the MIGO and AMIGO designs are similar. It also shows that there are small differences between PI and PID control, which was expected since the process is delay dominant. □

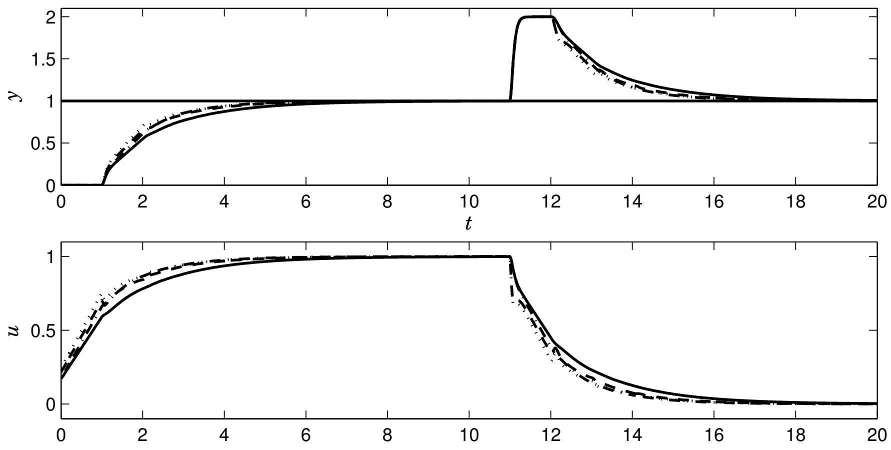

Figure 7.20 Responses to a unit step change at time 0 in set point and a unit load step at time 10 for PI controllers designed by MIGO (solid line) and AMIGO-step (dash-dotted line), and PID controllers designed by MIGO (dashed line) and AMIGO-step (dotted line) for the delay-dominant process in Example 7.3.

7.8 Measurement Noise and Filtering

So far we have focused on attenuation of load disturbances and robustness to process variations. In many cases it is also necessary to consider measurement noise. This is particularly the case for lag-dominated processes where maximization of integral gain gives controllers with high gain. Measurement noise can then create large control actions. In extreme cases the control signals can be so large that the actuator is saturated. The effect of measurement noise can be estimated from the transfer function from measurement noise to control signal:

$$G_{un} = -\frac{C}{1+PC}. \tag{7.16}$$

Since measurement noise typically has high frequencies, the high-frequency properties of the transfer function are particularly important.

The effect of measurement noise can be alleviated by filter the measurement signal as is shown in Figure 4.3. The transfer function from measurement noise to controller output is then

$$G_{un} = -\frac{CG_f}{1+PCG_f}. \tag{7.17}$$

A typical filter transfer function is given by

$$G_f(s) = \frac{1}{1+sT_f+(sT_f)^2/2}; \tag{7.18}$$

see (3.16). Adding a filter will reduce the robustness of the controller. It is easy to recover robustness by redesigning a controller for a process with the

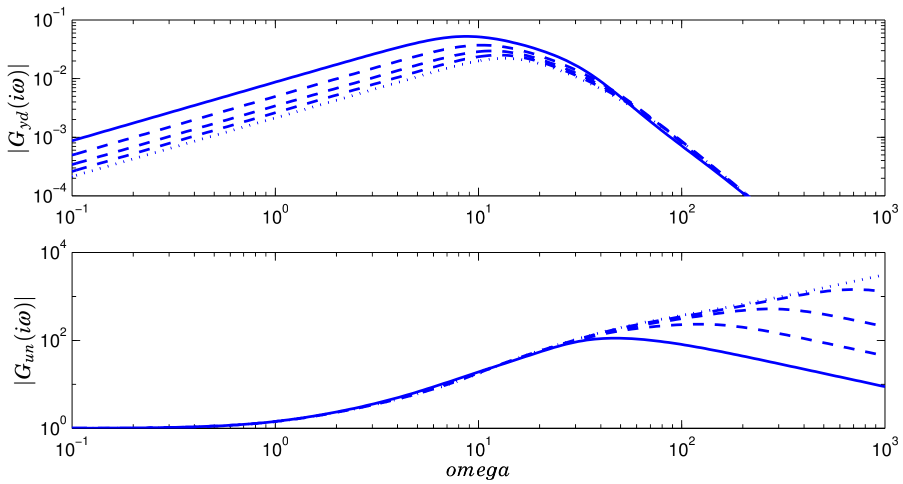

Figure 7.21 Gain curves of the transfer functions from load disturbance to process output (upper) and from measurement noise to controller output (lower). Curves for ideal PID control are shown in dotted lines, and for PID control with filtering with time constants $T_f = 0.002$, 0.005, and 0.010 in dashed lines, and for $T_f = 0.02$ in solid lines.

transfer function $P(s)G_f(s)$. The design procedure starts by designing an ideal PID controller for the process $P(s)$. The design gives guidance for choosing the filter time constant T_f; typically a fraction of the integral time for PI control or the derivative time for PID control. An ideal PID controller is then designed for the process $P(s)G_f(s)$, and the controller for the process $P(s)$ is then $C(s)G_f(s)$. If necessary, the procedure can be iterated a few times. Adding a filter improves attenuation of measurement noise at the cost of poorer load disturbance attenuation. The final design choice is thus a compromise. The procedure is illustrated by an example.

Example 7.4—Effect of Filtering.
Consider the lag-dominated system in Example 7.1. Table 7.2 shows that the MIGO design gives a controller with high gain, $k = 56.9$,, which gives good attenuation of load disturbances with integral gain $k_i = 495$. The transfer function from measurement noise to controller output has high gain at high frequencies, as is shown by the Bode plot in Figure 7.21. The derivative time is $T_f = 0.06$ and reasonable filter time constants are in the range $T_f = 0.002$–0.020.

To design controllers for the system $P(s)G_f(s)$ we approximate the transfer function using Skogestad's half-rule. Starting with the SOTD model used in Example 7.1 we account for filtering by adding $T_f/2$ to the time constant T_2 and to the time delay L_1. The combination or process $P(s)$ and filter $G_f(s)$ is then represented by the SOTD model (2.47) with $T_1 = 0.980$, $T_2 = 0.108+T_f/2$, and $L_1 = 0.010 + T_f/2$. Equation (7.13) then gives the controller parameters shown in Table 7.5. Controller gain decreases by a factor of 2 with increasing values of the filter constant, integral time increases by a factor of 2 and the derivative time increases by about 40 percent. Integral gain k_i decreases with

Table 7.5 Controller parameters obtained in Example 7.4. Compare with Example 7.1.

T_f	K	T_i	T_d	k_i	k_d/T_f	M_{un}
0.000	59.6	0.127	0.0523	468	∞	∞
0.002	52.6	0.138	0.0546	382	1436	1436.
0.005	44.7	0.153	0.0578	293	516	520
0.010	35.6	0.176	0.0624	203	222	234
0.020	25.1	0.220	0.0705	115	88.6	112

a factor of 4 and the largest high-frequency gain of G_{un} decreases with several orders of magnitude.

Table 7.5 also shows the largest gain, M_{un}, of the transfer function $G_{un}(s)$ and its estimate k_d/T_f given by (4.44). The simple estimate is remarkable accurate for small filter-time constants.

The properties of the different controllers are also illustrated in Figure 7.22 which shows the responses of the system to load disturbances and measurement noise for different controllers designed with different values of the filter-time constant T_f.

Notice that there are large variations in the control signal for $T_f = 0.002$ even if the noise in the process output is not too large. The reason for this is that the controller gain is quite large.

Figures 7.21 and 7.22 give a good illustration of the trade-off between attenuation of load disturbances and injection of measurement noise. The final trade-off is always subjective, but a moderate amount of filtering is always useful because the effect of measurement noise can be decreased significantly with only moderate increase of integral gain. In the particular case a value of T_f around 0.01 is a reasonable choice. □

7.9 Detuning

The AMIGO tuning rule lends itself naturally to detuning. For PI control, load disturbance rejection can be characterized by integral gain $k_i = K/T_i$. Amplification of measurement noise can be characterized by controller gain K. Since measurement noise typically has high frequencies the variation of the control signal generated by measurement noise is approximately $Kn(t)$ where $n(t)$ is the measurement noise.

Figure 7.23 shows the robustness domain of a PI controller for typical first-order processes. All gains in the white area satisfy the robustness condition that the combined sensitivities are less than $M = 1.4$. Any combination of controller parameter in that range is thus admissible from the point of view of robustness. Load disturbance attenuation is captured by the integral gain k_i. Assuming that load disturbances enter at the process input the transfer

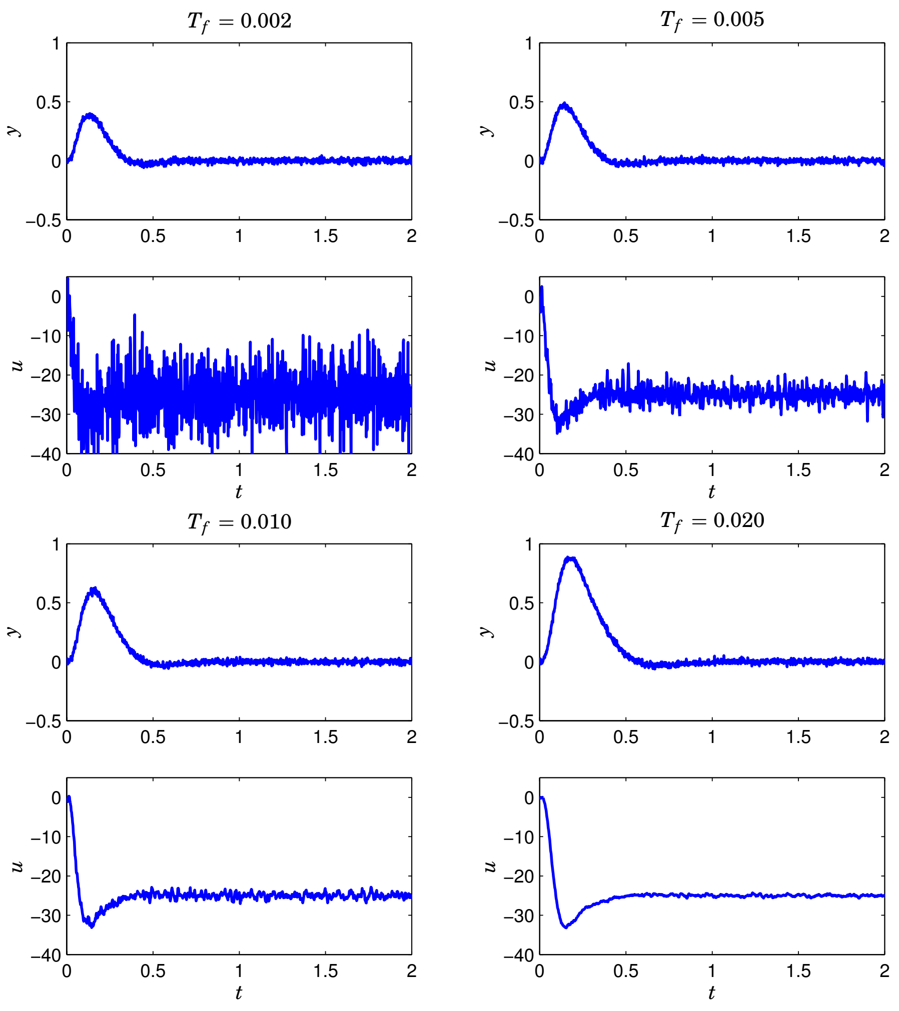

Figure 7.22 Simulation of PID control of the system in Example 7.4. The measurements are filtered with the second order filter (7.18) where time constants are $T_f = 0.002$, 0.005, 0.010, and 0.020. For each filter constant the controller parameters are chosen to maximize integral gain subject to the robustness constraint $M = 1.4$. A load disturbance of 25 is applied at time 0 and measurement noise is acting on the system.

function from load disturbances to process output is approximately given by

$$G(s) = \frac{s}{k_i}.$$

Load disturbance attenuation is thus inversely proportional to k_i. Measurement noise typically has high frequencies. For high frequencies the transfer function from measurement noise to the control signal is approximately given by

$$G(s) = K.$$

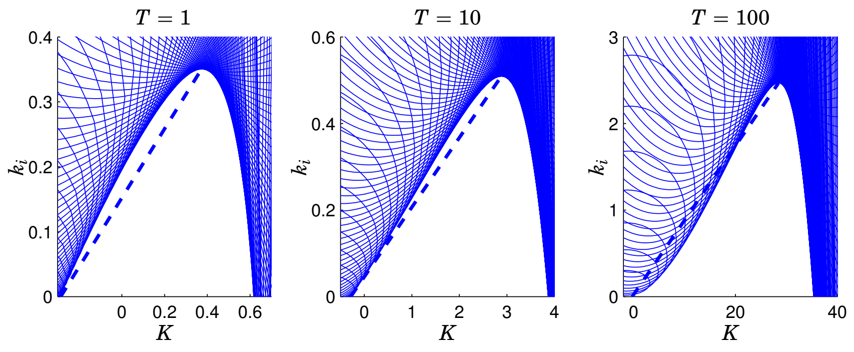

Figure 7.23 The sensitivity constraint for a system with $M = 1.4$ and the transfer function $P(s) = e^{-s}/(1+sT)$, with $T = 1$, 10 and 100.

Injection of measurement noise is thus proportional to controller gain K. Since all values of K and k_i that satisfy the robustness requirement are given in Figure 7.23, it is straightforward to make a trade-off between load disturbance attenuation and injection of measurement noise.

Figure 7.23 indicates that variations of the control variable due to measurement noise can be reduced simply by reducing proportional gain. The penalty for this is poorer attenuation of load disturbances. A proper quantitative trade-off is easily done based on Figure 7.23. Instead of choosing the largest value of integral gain we should simply choose a combination of proportional and integral gain on the left border of the robustness region in the figure. Since the figure is not available when the simple tuning formulas are used we will develop an approximate formula for the left boundary of the robustness region.

A First Attempt

One possibility is to reduce the gains as indicated by the straight line in Figure 7.23. This line goes through the peak with parameters K^0 and k_i^0 obtained by the nominal design. When integral gain is zero the robustness boundary goes through the point

$$\begin{aligned} KK_p &= -1 + \frac{1}{M_s} = -\frac{M_s - 1}{M_s} = -\alpha \\ k_i &= 0. \end{aligned} \tag{7.19}$$

A line through this point and the extremum is

$$k_i = k_i^0 \frac{KK_p + \alpha}{K^0 K_p + \alpha}. \tag{7.20}$$

Notice that it is not useful to reduce proportional gain below the value $K = 0$, when the controller is reduced to a pure integral controller. Figure 7.23 shows that the formula (7.20) is conservative for $T = 1$ and $T = 10$ but not for $T = 100$ since the line will be partially outside the robustness boundary for this process. We will use the detuning formula (7.20) for processes with $\tau > 0.1$. For lag-dominant processes with $\tau < 0.1$, better approximations of the robustness boundary are required.

Figure 7.23 indicates that for $T = 100$ the lower left-hand side of the robustness boundary has the shape of a parabola. To obtain a better approximation of the left-hand boundary of the robustness region we will first consider a simple example where the robustness boundary can be computed explicitly. An integrator with delay is the extreme case of a lag-dominated process, but we will start by determining the robustness bound for an even simpler case.

A Pure Integrator

Consider a pure integrator

$$P(s) = \frac{1}{s}.$$

The loop transfer function with PI control is

$$G_l(s) = \frac{Ks + k_i}{s^2} = \frac{k_i}{s^2} + \frac{K}{s}.$$

Requiring that the loop transfer function is outside a circle with radius r and center at $-c$ gives

$$\Big|c + G_l(i\omega)\Big|^2 \geq r^2. \tag{7.21}$$

But

$$\begin{aligned}\Big|c + G_l(i\omega)\Big|^2 &= \Big|c - \frac{k_i}{\omega^2} - i\frac{K}{\omega}\Big|^2 = \Big(c - \frac{k_i}{\omega^2}\Big)^2 + \Big(\frac{K}{\omega}\Big)^2 \\ &= \frac{k_i^2}{\omega^4} + \frac{K^2 - 2ck_i}{\omega^2} + c^2 = \Big(\frac{k_i}{\omega^2} + \frac{K^2 - 2ck_i}{2k_i}\Big)^2 + c^2 - \Big(\frac{K^2 - 2ck_i}{2k_i}\Big)^2.\end{aligned}$$

The robustness condition can thus be written as

$$\Big|c + G_l(i\omega)\Big|^2 = \Big(\frac{k_i}{\omega^2} + \frac{K^2 - 2ck_i}{2k_i}\Big)^2 + c^2 - \Big(\frac{K^2 - 2ck_i}{2k_i}\Big)^2 \geq r^2.$$

The left-hand side has its smallest value for

$$\omega^2 = \frac{2k_i^2}{2ck_i - K^2},$$

where we require that $2ck_i \geq K^2$. The robustness condition thus imposes the following constraint between integral and proportional gain:

$$\Big(\frac{2ck_i - K^2}{2k_i}\Big)^2 \leq c^2 - r^2.$$

Equality is achieved for

$$\frac{2ck_i - K^2}{2k_i} = \sqrt{c^2 - r^2},$$

or

$$k_i = \frac{K^2}{2\big(c - \sqrt{c^2 - r^2}\big)} = \frac{K^2(c + \sqrt{c^2 - r^2})}{2r^2}, \tag{7.22}$$

which is a parabola in the K, k_i plane.

Non-normalized Variables

So far we have used scaled variables. If we consider a process with the transfer function

$$P(s) = \frac{K_v}{s} = \frac{K_p}{sT},$$

the equation becomes

$$k_i = \frac{K_p K^2 (c + \sqrt{c^2 - r^2})}{2Tr^2}.$$

For a design based on a constraint on M_s we have $c = 1$ and $r = 1/M_s$; hence

$$\frac{c + \sqrt{c^2 - r^2}}{2r^2} = \frac{M_s \big(M_s + \sqrt{M_s^2 - 1}\big)}{2}.$$

For a design with equal constraints on both sensitivity and complementary sensitivity we have

$$\begin{aligned} r &= \frac{2M - 1}{2M(M-1)} \\ c &= \frac{2M^2 - 2M + 1}{2M(M-1)}. \end{aligned}$$

This implies $c^2 - r^2 = 1$, and we get

$$\frac{c + \sqrt{c^2 - r^2}}{2r^2} = \frac{c + 1}{2r^2} = M(M-1).$$

Summarizing, we find that the robustness constraint for a pure integrator becomes

$$k_i = \beta \frac{K_p K^2}{T}, \tag{7.23}$$

where

$$\beta = \begin{cases} M_s \left(M_s + \sqrt{M_s^2 - 1} \right) /2 & \text{for design based on } M_s \\ M(M-1) & \text{for design based on } M. \end{cases} \tag{7.24}$$

Equation 7.23 implies that integral gain is reduced by the factor n^2 when gain is reduced by the factor n. Since $T_i = K/k_i$ we find that the integral time increases with the factor n.

The detuning rule (7.23) is derived for an integrator without time delay. To deal with the process (7.3) we first observe from Figure 7.23 and Equation 7.19 that the parabola passes through the point $KK_p = -\alpha$ for $k_i = 0$. For the process (7.3) the detuning rule (7.23) should therefore be replaced by

$$k_i = \beta \frac{(\alpha + KK_p)^2}{K_p(L + T)}, \tag{7.25}$$

where the time constant T has been replaced by the effective time constant $T + L$.

Combining the Results

We have obtained two formulas for detuning. The formula (7.20) based on linear extrapolation gives good results for processes with $\tau > 0.1$, and processes with $\tau < 0.1$ as long as the gain reduction is moderate. The formula (7.25) gives good results for strongly lag-dominated processes with large gain reduction. It is then natural to combine the formulas. This will give a good match to the left part of the robustness constraint in Figure 7.23.

The formulas (7.20) and (7.25) give the same result for

$$k_i^0 \frac{\alpha + KK_p}{\alpha + K^0K_p} = \beta \frac{(\alpha + KK_p)^2}{K_p(L+T)}$$

or

$$KK_p = \frac{k_i^0 K_p(L+T)}{\beta(\alpha + K^0K_p)} - \alpha. \tag{7.26}$$

Summarizing we obtain the following formula for detuning the PI controller. First choose a gain $K < K_0$. Then determine the integral gain in the following way. For process with $\tau > 0.1$, determine k_i from (7.20). For processes with $\tau < 0.1$, compute integral gain from

$$k_i = \begin{cases} k_i^0 \dfrac{\alpha + KK_p}{\alpha + K^0K_p} & \text{for } KK_p \geq \dfrac{k_i^0 K_p(L+T)}{\beta(\alpha + K^0K_p)} - \alpha \\ \beta \dfrac{(\alpha + KK_p)^2}{K_p(L+T)} & \text{for } KK_p < \dfrac{k_i^0 K_p(L+T)}{\beta(\alpha + K^0K_p)} - \alpha. \end{cases} \tag{7.27}$$

Notice that this equation is an approximation of the left-hand side of the robustness constraint in Figure 7.23.

Examples

The detuning rule (7.27) will be illustrated by some examples. First, we treat one single process with the structure (7.3).

EXAMPLE 7.5—DETUNING
A PI controller designed for the process

$$P(s) = \frac{1}{1 + 1000s} e^{-s} \tag{7.28}$$

using the AMIGO design (7.5) with the robustness constraint $M = 1.4$ has the controller parameters $K = 349$ and $T_i = 13.2$, which gives the integral gain $k_i = 26.4$. The process is almost an integrator with delay, $P(s) \approx 0.001e^{-s}/s$, with a normalized time delay $\tau \approx 0.001$. This explains the high gain in the controller. Figure 7.24 shows Nyquist curves of the loop transfer function, as well as the curves obtained when the gain is reduced by the factors 0.5, 0.1, 0.05, 0.01, and 0.005, respectively, using the detuning rule (7.27). The figure shows that the loop transfer functions of the detuned systems remain close to the robustness region.

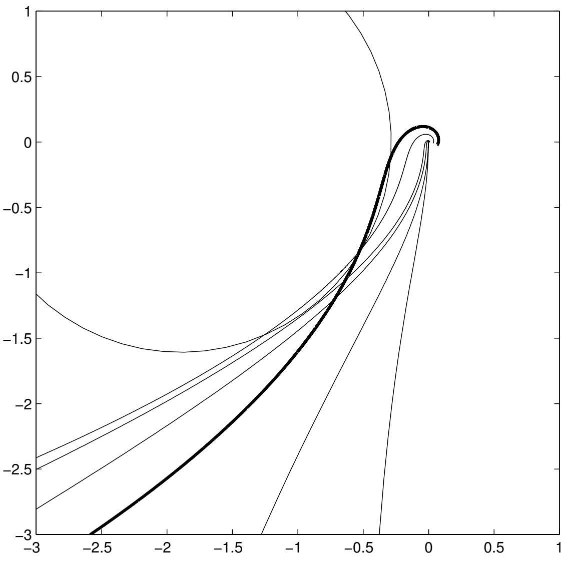

Figure 7.24 Nyquist curves for loop transfer functions for PI control of the process (7.28). The thick line corresponds to the optimal controller, and the thin lines to controllers where the gain is reduced by the factors 0.5, 0.1, 0.05, 0.01, and 0.005, respectively. The circle shows the robustness constraint $M = 1.4$.

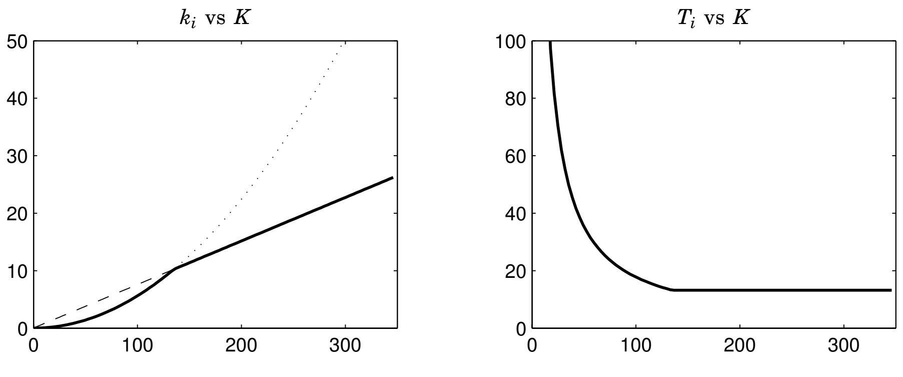

Figure 7.25 Relations between the reduced gain K and the integral gain k_i (left) and the integral time T_i (right). The dashed line corresponds to the detuning rule (7.20) and the dotted line to the rule (7.25)

Figure 7.25 shows how integral gain k_i and integral time T_i are changed when the gain is reduced. Notice that the integral time remains almost constant as long as the gain reduction is made according to the linear part of (7.27). The linear reduction is replaced by the quadratic reduction when the gain is lower than $K \approx 135$. The gain reduction at this point is $K/K^0 \approx 135/349 \approx 0.4$. □

In the next example, the detuning rule (7.27) is applied to a large test batch of processes.

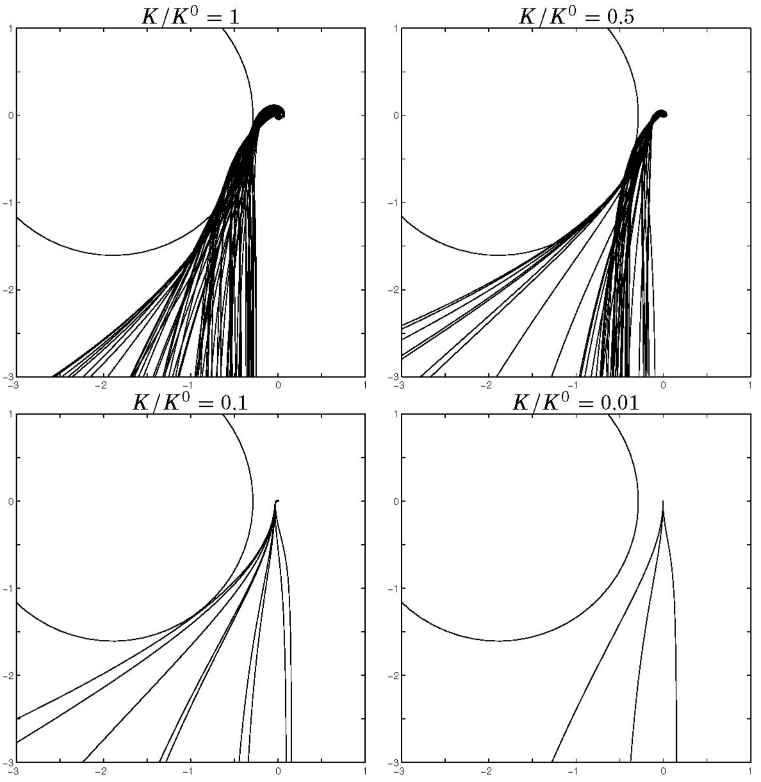

Figure 7.26 Nyquist curves of loop transfer functions where the controller is detuned using the rule (7.27). The circles show the robustness margin $M = 1.4$.

EXAMPLE 7.6—DETUNING APPLIED TO THE TEST BATCH

The detuning rule (7.27) has been applied to all processes in the test batch (7.2). Figure 7.26 shows the Nyquist plots of the loop transfer functions obtained when the PI controllers are detuned using (7.27). The figure shows four cases; the original loop and loops where the controller gain is reduced by factors 0.5, 0.1, and 0.01. Only those systems where the controller gain is larger than 0.5 are shown. This is the reason why only three cases are left when the controller gain is reduced by a factor of 0.01.

The example shows that the loop transfer functions remain close to the robustness region and that the detuning rule works well for the processes in the test batch. □

A Pole Placement Interpretation

There are situations when the response time and the bandwidth are of great importance. In this case the detuning problem can be solved using a simple pole placement approach. Neglecting the the time delay the loop transfer function

Table 7.6 Controller parameters, frequency and damping.

T	K^0	k_i^0	ω	ζ	ω^e	ζ^e
5	1.21	0.296	0.343	0.910	0.222	0.830
10	2.82	0.513	0.226	0.845	0.216	0.804
20	6.24	0.99	0.222	0.815	0.217	0.794
100	34	4.94	0.222	0.788	0.211	0.784

obtained when a PI controller is combined with the process model (7.3) becomes

$$G_l(s) = \frac{K_pKs + K_pk_i}{s(1+sT)}.$$

The characteristic polynomial is

$$s^2 + s\frac{1+K_pK}{T} + \frac{k_iK_p}{T}.$$

Comparing this with the standard polynomial $s^2 + 2\zeta\omega s + \omega^2$ we find

$$\begin{aligned} 1 + K_pK &= 2\zeta\omega T \\ K_pk_i &= \omega^2 T. \end{aligned} \tag{7.29}$$

Using the numerical values for the process (7.3) with $K_p = 1$ and $L = 1$ we get the values in Table 7.6 for different values of time constant T. The optimal controller parameters K^0 and k_i^0 are determined from the MIGO design with $M = 1.4$. The last two columns are the frequency and the damping when the time constant T is replaced by the effective time constant $T_e = T + L$. Notice that the frequency and the damping are practically constant for the whole range of parameters.

Another way to detune the controller is to use Equation 7.29 and reduce the natural frequency. This gives

$$\omega = \frac{1 + KK_p}{2\zeta T_e}$$

and

$$k_i = \frac{(1+KK_p)^2}{4\zeta^2 K_p T_e}. \tag{7.30}$$

This is similar but of somewhat different form than the parabolic expression in (7.27).

PID Control

For PID control, it is natural to start a high-frequency gain reduction by reducing the derivative gain. One way to do this is the following. Let K^{PID}, k_i^{PID}, and k_d^{PID} denote the gains for PID controllers obtained by the AMIGO tuning formula, and let K^{PI} and k_i^{PI} be the corresponding controller gains for the PI controller. Following the ideas used in PI control we will obtain detuned controller by linear interpolation. This gives

$$\begin{aligned} K &= K^{PI} + \frac{k_d}{k_d^{PID}}(K^{PID} - K^{PI}) \\ k_i &= k_i^{PI} + \frac{k_d}{k_d^{PID}}(k_i^{PID} - k_i^{PI}). \end{aligned} \tag{7.31}$$

This gives a natural way to detune the PID controller until it becomes a PI controller. If further gain reductions are required we can proceed as for PI controllers.

Example 7.7—Detuning Applied to the Test Batch
The detuning rule (7.31) has been applied to the test batch (7.2). Figure 7.27 shows the Nyquist plots of the loop transfer functions obtained when the PID controllers are detuned using (7.31). The figure shows four cases, the original loop tuned with the AMIGO tuning rules (7.7) and loops where the derivative gain is reduced by factors 0.1, 0.01, and 0. The last case gives a pure PI controller.

The example shows that the loop transfer functions remain close to the robustness region and that the detuning rule works well for the processes in the test batch. □

7.10 Summary

In this section it has been attempted to develop simple tuning rules in the spirit of the work done by Ziegler and Nichols in the 1940s. The goal has been to make rules that can be used both for manual tuning and in auto-tuners for a wide range of processes. The methods were developed by applying the techniques for robust loop shaping presented in Section 6.8 to a large test batch of representative processes. The controller parameters obtained were then correlated with simple features of process dynamics.

One interesting observations was that there are significant differences between processes with delay-dominated and lag-dominated dynamics. To capture this difference, process dynamics must be characterized by at least three parameters. Notice that Ziegler and Nichols used only two parameters. One possible choice is: process gain K_p, apparent time constant T, and apparent time delay L. These parameters can be obtained from a step response experiment. Section 2.7. The relative time delay $\tau = L/(L+T)$, which ranges from 0 to 1, is used for a crude characterization of dynamics. Processes with small τ are called lag dominated, processes with τ close to one are called delay dominated and processes with τ around 0.5 are called balanced.

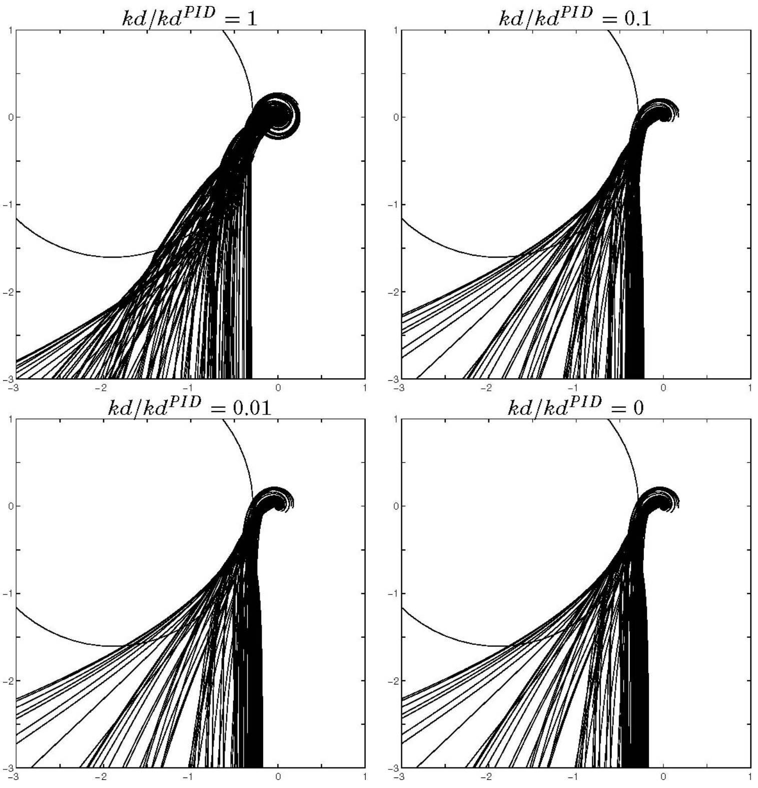

Figure 7.27 Nyquist curves of loop transfer functions where the controller is detuned using the rule (7.31). The circles show the robustness constraint $M = 1.4$.

Very satisfactory results were obtained for PI control, where the parameters from MIGO tuning can be matched with

$$\begin{aligned} K &= \frac{0.15}{K_p} + \left(0.35 - \frac{LT}{(L+T)^2}\right)\frac{T}{K_pL} \\ T_i &= 0.35L + \frac{13LT^2}{T^2 + 12LT + 7L^2}, \end{aligned} \tag{7.32}$$

for the full test batch. The tuning rule, which we called AMIGO (Approximate MIGO), gave good results for all processes in the test batch ranging from process with integration to processes with pure time delay.

The numerical values in (7.32) are based on a combined sensitivity $M = 1.4$. The form of the tuning rules are the same for other values of M but the numerical values of the coefficients are different.

For PID control of processes with $\tau > 0.3$ it was also possible to find the simple tuning rule

$$\begin{aligned} K &= \frac{1}{K_p}\left(0.2 + 0.45\frac{T}{L}\right) \\ T_i &= \frac{0.4L + 0.8T}{L + 0.1T}L \\ T_d &= \frac{0.5LT}{0.3L + T}. \end{aligned} \tag{7.33}$$

This tuning rule also gave a conservative tuning rule for lag-dominant processes. It can thus be used for the full range of processes provided that a conservative tuning is acceptable. Derivative action can give substantial benefits for lag-dominated processes. A quantitative estimate can be obtained by comparing the integral gains k_i of of (7.32) and (7.33).

For some lag-dominated processes it is possible to give tuning rules with much better performance than (7.33). When process dynamics is characterized by the parameters K_p, L, and T both time delay and small time constants are captured in L. For lag-dominated processes improved performance can be obtained if the time constant and the delay are separated by better modeling. For a model characterized by an SOTD model with four parameters the AMIGO tuning rule is

$$\begin{aligned} K_pK &= \alpha_1 + \alpha_2\frac{T_1}{L_1} + \alpha_3\frac{T_2}{L_1} + \alpha_4\frac{T_1T_2}{L_1^2} \\ K_pk_i &= \beta_1\frac{1}{L_1} + \beta_2\frac{T_1}{L_1^2} + \beta_3\frac{T_2}{L_1^2} + \beta_4\frac{T_1T_2}{L_1^3} \\ K_pk_d &= \left(\gamma_1L_1 + \gamma_2T_1 + \gamma_3T_2 + \gamma_4\frac{T_1T_2}{L_1}\right)\frac{T_1 + T_2}{T_1 + T_2 + L_1}, \end{aligned} \tag{7.34}$$

where the parameters are given by

$$\begin{array}{llll} \alpha_1 = 0.19 & \alpha_2 = 0.37 & \alpha_3 = 0.18 & \alpha_4 = 0.02 \\ \beta_1 = 0.48 & \beta_2 = 0.03 & \beta_3 = -0.0007 & \beta_4 = 0.0012 \\ \gamma_1 = 0.29 & \gamma_2 = 0.16 & \gamma_3 = 0.20 & \gamma_4 = 0.28. \end{array} \tag{7.35}$$

This tuning rule is similar to (7.33) for processes with balanced and delay-dominated dynamics, but it typically gives higher gain for lag-dominated processes. This tuning rule requires improved process models. It is difficult to obtain two time constants from a step response experiment. System identification or the combined frequency and step response methods described in Section 2.7 can be used.

Tuning rules based on frequency response data have also been developed. In this case the parameters were chosen as: process gain K_p, ultimate gain K_{180}, and ultimate period T_{180}. The parameter $\kappa = K_{180}/K_p$ was used to classify the processes. This choice matches what is used in auto-tuners based on relay feedback. The AMIGO tuning rule for PI controllers based on frequency response data is

$$\begin{aligned} KK_{180} &= 0.16 \\ \frac{T_i}{T_{180}} &= \frac{1}{1 + 4.5\kappa} \end{aligned} \tag{7.36}$$

and the tuning rules for PID controllers are

$$\begin{aligned} K &= (0.3 - 0.1\kappa^4)/K_{180} \\ T_i &= \frac{0.6}{1+2\kappa} T_{180} \\ T_d &= \frac{0.15(1-\kappa)}{1-0.95\kappa} T_{180}. \end{aligned} \tag{7.37}$$

These tuning rules give good tuning for balanced and delay-dominant processes with $\kappa > 0.2$, but are not appropriate for lag-dominant processes.

The AMIGO tuning rules optimize load disturbance attenuation with a specified robustness. Measurement noise can be dealt with by filtering the process output. There are significant advantages of using a second-order filter. The dynamics of the filter can be accounted for in a simple way by applying the AMIGO rules with $T_f/2$ added to T and L for the FOTD model and with $T_f/2$ added to T_2 and L for the SOTD model. In this way it is possible to make the trade-off between attenuation of load disturbances and injection of measurement noise.

A systematic method of detuning the controllers to give a specific controller gain has also been developed.

7.11 Notes and References

This chapter is based on work by the authors and their students. The motivation was to gain improved understanding in the information required to develop good tuning rules and to find tuning rules that can be used for manual and automatic tuning. The basis for the work is the robust design method (MIGO) which is developed in [Åström *et al.*, 1998] for PI control and in [Åström and Hägglund, 2001; Panagopoulos *et al.*, 2002] for PID control. In certain circumstances it is advantageous to have $T_i < 4T_d$ as has been noted by [Kristiansson and Lennartsson, 2002]. Tuning rules for that case are given by [Wallén *et al.*, 2002]. The MIGO method requires knowledge of the transfer function of the process. The AMIGO tuning rules for PI and PID presented in [Hägglund and Åström, 2002; Hägglund and Åström, 2004b; Hägglund and Åström, 2004a] can be applied when only features of step and frequency responses are known. Much of the material in the chapter have not been published before.