Dominant Pole Placement Design of PI Regulators

Åström, Karl Johan

1988

Document Version:
Publisher's PDF, also known as Version of record

CODEN: LUTFD2/(TFRT-7381)/1-37/(1988)

Dominant Pole Placement Design of PI Regulators

K J Åström

Department of Automatic Control
Lund Institute of Technology
February 1988

Dominant Pole Placement Design of PI Regulators

K J Åström

Abstract
A systematic design method based on the idea of positioning dominating poles is developed in this report where the method is also used to develop tuning methods for simple PI regulators. The results can also be used as a tool to assess achievable control performance in a knowledge based control system.

Department of Automatic Control
Lund Institute of Technology

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
Report

Date of issue
February 1988

Document Number
CODEN: LUTFD2/(TFRT-7381)/1-37/(1988)

Author(s)
K J Åström

Supervisor

Sponsoring organisation

Title and subtitle
Dominant Pole Placemnt Design of PI Regulators

Abstract

A systematic design method based on the idea of positioning dominating poles is developed in this report where the method is also used to develop tuning methods for simple PI regulators. The results can also be used as a tool to assess achievable control performance in a knowledge based control system.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
37

Recipient's notes

Security classification

1. Introduction

There are many different ways to approach the design problem for automatic control systems. One possibility is to postulate a regulator with a given structure and to adjust the parameters until the desired specifications are achieved. The adjustments can be made based on a mathematical model or as on-line adjustments. The classical root locus method is a typical example. The drawback with schemes of this type is that when using them it is difficult to assess what can be achieved and particularly if drastic improvements are possible by increasing the regulator complexity.

Another set of methods is based on characterization of a mathematical model of the process and specifications in terms of an optimization criterion or a specified closed loop behaviour. The design method typically results in a regulator whose complexity matches the model complexity. A typical example is a design based on observers and state feedback. The advantages of these methods are that the design methods are simple and that limitations in terms of nonminimum phase dynamics show up clearly. One drawback is, however, that the essential interplay between specifications and regulator complexity does not show up. Another disadvantage is that it is often too much to specify all the poles of a closed loop system including the observer.

In this paper we develop design methods which strike a middle ground. The key idea is to position only the dominant poles. It is straightforward to find a regulator which positions a few closed loop poles. The key difficulty is to ensure that these poles are also dominating. This can be explored using conventional frequency response methods. The analysis reveals an interesting connection between specifications on the closed loop system and regulator complexity.

The result is used to develop design methods for simple PID regulators. They are also used to assess achievable control performance.

The paper is organized as follows. The design method is presented in Section 2. In the following sections it is applied to systems with different types of dynamics.

2. The Design Method

The design method can be described as follows. A regulator with a given structure and adjustable parameters is chosen. A number of desired closed loop poles $p_1, p_2, \ldots, p_k$ are chosen. The regulator parameters are then chosen in such a way that the closed loop system has the desired poles. Finally it is attempted to ensure that the selected poles are dominating.

2.1 PI Control

The method is illustrated by design of PI regulators. Let the process to be controlled have the transfer function $G(s)$. The characteristic equation of the closed loop system under PI control is

$$F(s) = 1 + \left(k + \frac{k_i}{s}\right) G(s) = 0 \tag{2.1}$$

Two dominant poles can be specified when there are two adjustable parameters. It is natural to choose these poles as

$$p = -\sigma \pm i\omega = -\zeta\omega_0 \pm i\omega_0\sqrt{1-\zeta^2} \tag{2.2}$$

Require that the function F has zeros for $s = p_1$ and $s = p_2$ we get

$$1 + \left(k + \frac{k_i}{-\sigma + i\omega}\right) G(-\sigma + i\omega) = 0$$

or

$$\left(1 + \frac{k(-\sigma - i\omega)}{\sigma^2 + \omega^2}\right)(A + iB) = 0$$

where

$$\begin{aligned} A &= ReG(-\sigma + i\omega) \\ B &= J_mG(-\sigma + i\omega) \end{aligned} \tag{2.3}$$

Equating real and imaginary parts we get

$$\begin{cases} (\sigma^2 + \omega^2)Ak - (\sigma A - \omega B)k_i + \sigma^2 + \omega_2 = 0 \\ (\sigma^2 + \omega^2)Bk - (\omega A + \sigma B)k_i \end{cases}$$

These equations have the solution

$$\begin{aligned} k &= -\frac{\omega A + \sigma B}{\omega(A^2 + B^2)} = -\frac{A\sqrt{1-\zeta^2} + \zeta B}{(A^2 + B^2)\sqrt{1-\zeta^2}} \\ k_i &= -\frac{(\sigma^2 + \omega^2)B}{\omega(A^2 + B^2)} = -\frac{\omega_0 B}{(A^2 + B^2)\sqrt{1-\zeta^2}} \end{aligned} \tag{2.4}$$

The integration time is given by

$$T_i = \frac{k}{k_i} = \frac{\omega A + \sigma B}{(\sigma^2 + \omega^2)B} = \frac{A\sqrt{1-\zeta^2} + \zeta B}{\omega_0 B} \tag{2.5}$$

2.2 Pole Domination

The solution (2.4) always exists if $\zeta \neq 1$ and $A^2 + B^2 \neq 0$. Under these conditions regulator parameters can always be found such that the closed loop characteristic equation has the zeros p_1 and p_2 given by (2.2). This does, however, not imply that the closed loop system is stable or that the chosen poles are dominating. These issues have to be investigated by other methods. If the process gain is positive a necessary condition for stability is that the integrator gain k_i is also positive.

With given regulator parameters the stability of the closed loop can be investigated by the Nyquist criterion. The robustness with respect to unmodeled dynamics can also be assessed. The amplitude margin and the phase margin can be used as crude measures.

A direct approach is to calculate all the closed loop poles. This does, however, require detailed knowledge of the transfer function of the process. Another possibility is to compute the crossover frequency. For a second order system with two dominating poles the following relation holds between the undamped natural frequency ω_0 and the crossover frequency ω_c

$$\omega_c = \omega_0 \sqrt{\sqrt{4\zeta^4 + 1} - 2\zeta^2} \tag{2.6}$$

This relation can be used to judge if the poles are dominating.

2.3 Achievable Performance

The conditions for pole domination can normally only be satisfied if the closed loop bandwidth is chosen in a certain range. This range can be estimated by a simple argument. Consider the characteristic equation

$$1 + G_R(s)G(s) = 0$$

Hence

$$\arg G(s) = -\pi - \arg G_R(s)$$

With PI control the complexity of G_R is severely limited. This implies that the conditions can only be satisfied for a restricted range in the s-plane.

With pure integral control we get

$$G_R\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = -\frac{k_i}{\omega_0}\left(\zeta + i\sqrt{1-\zeta^2}\right)$$

Hence

$$\arg G_R = -\pi + \alpha$$

when

$$\alpha = atan\frac{\sqrt{1-\zeta^2}}{\zeta}$$

The frequency where integrating control can be used is thus given by

$$\arg G\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = -\alpha$$

With pure proportional control we have $\arg G_R = 0$. With dominant pole design and proportional control the frequency ω_0 must thus be chosen as

$$\arg G\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = -\pi$$

With pure derivative control we get

$$G_R\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = k_d\omega_0\left(-\zeta + i\sqrt{1-\zeta^2}\right) = \pi - \alpha$$

The frequency where pure derivative control can be used is thus given by

$$\arg G\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = -2\pi + \alpha$$

By exploring the values of

$$\arg G\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right)$$

we can thus determine the frequency ranges when I, PI and PID control are appropriate.

3. First Order Lag

The design method will first be applied to a process described by a first order lag

$$G(s) = \frac{1}{1+sT} \tag{3.1}$$

There is no loss in generality to assume unit gain. We have

$$G(-\sigma + i\omega) = \frac{1}{1-\sigma T + i\omega T} = \frac{1-\sigma T - i\omega T}{(1-\sigma T)^2 + \omega^2 T^2}$$

Since $-\pi < \arg G < 0$ we find that any bandwidth specification can be satisfied using PI control. With PI control the closed loop system is also of second order which means that there are no other poles apart from the dominating pole. Application of the design method gives

$$\begin{aligned} k &= 2\zeta\omega_0 T - 1 \\ k_i &= \omega_0^2 T \\ T_i &= \frac{2\zeta\omega_0 T - 1}{\omega_0^2 T} \end{aligned} \tag{3.2}$$

The integrator gain k_i is always positive the proportional gain is positive if $\omega_0 T > 1/2\zeta$.

Notice that this case is very special since there are no limitations to the achieveable bandwidth. In practice the bandwidth will, however, be limited by noise and actuator saturation. The consequences of modeling errors can be determined in several different ways. If there is an additional time constant the model (3.1) is replaced by

$$G(S) = \frac{1}{(1+sT)(1+s\tau)} \tag{3.3}$$

If we allow a phaseshift of at most 0.1 rad at $\omega_0\sqrt{1-\zeta^2}$ we find that the maximum bandwidth is limited to

$$\omega_0\tau < \frac{0.1}{\sqrt{1-\zeta^2}} \tag{3.4}$$

Another way to estimate the maximum bandwidth is to apply the design procedure and to determine the largest value of L that will give a stable closed loop system. Such a calculation gives

$$\omega_0\tau < 2\zeta\,\frac{\tau + L}{T} \tag{3.5}$$

This bound is conservative since only stability is required.

Additional insight can be obtained by applying the design method to the model (3.3). This gives

$$\begin{aligned} k &= \omega_0\tau\omega_0 T(1-4\zeta^2) + 2\zeta\,\frac{T+L}{\tau}\omega_0 T - 1 \\ k_i &= \omega_0^2(T+\tau) - 2\zeta\omega_0^3\tau T \\ T_i &= \frac{k}{k_i} \end{aligned} \tag{3.6}$$

The characteristic equation of the closed loop system is of third order. It has the zeros $-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}$ and $-\alpha\omega$ where

$$\alpha = \frac{\tau + L}{\omega_0 L \tau} - 2\zeta \tag{3.7}$$

The value of α must be sufficiently large if the complex poles should be dominating. Requiring that $\alpha > 1$ we get

$$\omega_0\tau < \frac{T+\tau}{(1+2\zeta)T} \tag{3.8}$$

A comparison with (3.4) shows that the bandwidth can be increased by improved tuning which is matched to the dynamics. With $\tau \ll T$ and $\zeta = 0.707$ equation (3.8) gives the bound $\omega_0\tau < 0.4$ which should be compared with the bound $\omega_0\tau < 0.14$ given by the design based on a first order model.

4. Time Delay and First Order Lag

In this Section the design method will be applied to a process with the transfer function

$$G(s) = \frac{1}{1+sT}e^{-sL} \tag{4.1}$$

Again there is no lack of generality to assume that the process has unit gain. We have

$$G(-\sigma + i\omega) = \frac{1}{1-\sigma T + i\omega T}e^{\sigma L - i\omega L} = A + iB$$

where

$$A = \frac{(1-\sigma T)\cos\omega L + \omega T \sin\omega L}{(1-\sigma T)^2 + \omega^2 T^2}e^{\sigma L}$$

$$B = -\frac{\omega T\cos\omega L + (1-\sigma T)\sin\omega L}{(1-\sigma T)^2 + \omega^2 T^2}e^{\sigma} L$$

The regulator parameters are then obtained from the equations (2.4) and (2.5). Applying these formulas we get

$$k = \left[(2\sigma T - 1)\cos\omega L + \frac{\sigma + \omega^2 T - \sigma^2 T}{\omega}\sin\omega L\right]e^{-\sigma L}$$

$$k_i = \frac{(\sigma^2+\omega^2)[\omega T\cos\omega L + (1-\sigma T)\sin\omega L]e^{-\sigma L}}{\omega}$$

$$T_i = \frac{\omega(2\sigma T - 1)\cos\omega L + (\sigma + \omega^2 T - \sigma^2 T)\sin\omega L}{(\sigma^2+\omega^2)[\omega T\cos\omega L + (1-\sigma T)\sin\omega L]}$$

The achievable control performance can be estimated by evaluating the function $G(\omega_o(-\zeta + i\sqrt{1-\zeta^2}))$ for different values of ω_0. A few numerical examples serve as illustrations.

EXAMPLE 4.1
Consider the case $T = L = 1$. Let the relative damping be chosen as $\zeta = 0.707$. The achievable bandwidth can be estimated by evaluating the frequencies when

$$\arg G(\omega_0(-\zeta + i\sqrt{1-\zeta^2})) = \alpha$$

for α =45, 180 and 315. This shows that pure integral control gives $\omega_0 = 0.47$ rad/s, pure proportional control $\omega_0 = 1.9$ and pure derivative control gives 4.7 rad/s. It thus appears worth while to use a PI regulator since it allows a bandwidth increase with at most a factor of 4 compared to an I controller. The static loop gain under proportional control is only 0.36. A regulator with such a low gain is useless. It is therefore necessary to also have integral action.

Table 4.1 shows the regulator parameters k, k_i and T_i for different values of the design parameter ω_0. Additional data is given in Appendix A. To find the values of ω_0 where the complex poles are dominating first observe that with PI control the closed loop transfer function has a pole p_3 and a zero $z_1 = 1/T_i$ on the real axis. The values of p_3 and z_1 for different values of the design parameter ω_0 are also shown in Table 4.1. There is of course some arbitariness in a notion of domination. Somewhat arbitrarily we will consider the case when the pole p_3 and z_1 coincide as a separator. This implies that

Table 4.1 Regulator parameters for $L = 1$ and $T = 1$.

ω_0	k	k_i	T_i	ωT_i	p_3	z_1	ω_c
0.4	−0.104	0.201	−0.517	−0.146	0.034	−1.933	0.202
0.5	0.050	0.276	0.182	0.064	6.544	5.503	0.276
0.6	0.179	0.346	0.517	0.219	3.439	1.935	0.352
0.7	0.284	0.408	0.697	0.345	2.519	1.435	0.425
0.8	0.367	0.457	0.805	0.455	1.966	1.242	0.491
0.9	0.431	0.490	0.878	0.559	1.573	1.139	0.543
1.0	0.476	0.508	0.937	0.663	1.269	1.067	0.577
1.1	0.504	0.508	0.994	0.773	1.024	1.006	0.588
1.2	0.518	0.490	1.058	0.898	0.820	0.945	0.573
1.3	0.520	0.456	1.141	1.049	0.646	0.877	0.534
1.4	0.511	0.406	1.258	1.245	0.497	0.795	0.472
1.5	0.492	0.342	1.440	1.527	0.367	0.694	0.392
1.6	0.465	0.265	1.759	1.990	0.253	0.569	0.299
1.7	0.433	0.177	2.446	2.940	0.152	0.409	0.196
1.8	0.395	0.081	4.905	6.244	0.063	0.204	0.088
1.9	0.354	−0.022	−15.778	−21.200	−0.016	−0.063	−0.024

the complex poles are dominating if $\omega_0 \leq 1.1$ rad/s. Furthermore also notice that the crossover frequency ω_c and the integral gain also have their largest values for $\omega_0 - 1.1$ rad/s. The value of the crossover frequency $\omega_c = 0.59$ is close to the value 0.71 predicted by (2.6). The reason is that the pole p_3 is so close to the zero z_1 that the system response is close to the response of a second order system. With PI control we can thus achieve a bandwidth $\omega_0 = 1.1$ rad/s while an integrating regulator only gives $\omega_0 = 0.5$ rad/s. Some additional parameters which can be used to judge the control design are also given in Table 4.1A and Table 4.1B. Notice that the open loop process dynamics has a phaseshift of about 80° at the desired bandwidth. Notice that ωT_i is also an approximation of the ratio of proportional and integral action. At the frequency ω the ratio of the gains is precisely $k\omega/k_i = \omega T_i$. It thus follows from the table that at $\omega = 0.47$ the regulator has pure integral action. At $\omega = 1.8$ the proportional gain dominates. Figure 4.1 shows the responses to command inputs and load disturbances for some of the regulator designs. Notice that the conclusion drawn from the analysis of Table 4.1 is well supported by the time behaviour. Case C in the simulation shows clearly that the design poles are not dominating for $\omega_0 = 1.5$. The response will in fact have a significant contribution from a pole on the real axis.

We will now consider another example where time delay L is much larger than the time constant.

EXAMPLE 4.2

Consider the case $L = 1$ and $T = 0.1$. This is close to a pure delay process. Estimating the achievable bandwidth we find that with pure integral control we have $\omega = 1.0$ with pure proportional $\omega = 3.9$ and with pure derivative control $\omega = 6.7$. The static loop gain for pure proportional control is as low as 0.05 which clearly indicates that a significant integral action is required. Hence we find that it looks worthwhile to a PI controller but that integral action only gives a marginal improvement. The regulator parameters obtained for different bandwidth ω are given in Table 4.2A and Table 4.2B. The table shows that pole domination is obtained for $\omega_0 = 1.8$. The integral gain k_i also has its largest value for this ω_0. For this value of the design parameter the crossover frequency is $\omega_c = 0.6$. The predicted value of the crossover frequency assuming that the model is of second order is $\hat{\omega}_c = 1.2$rad/s. This is considerably higher

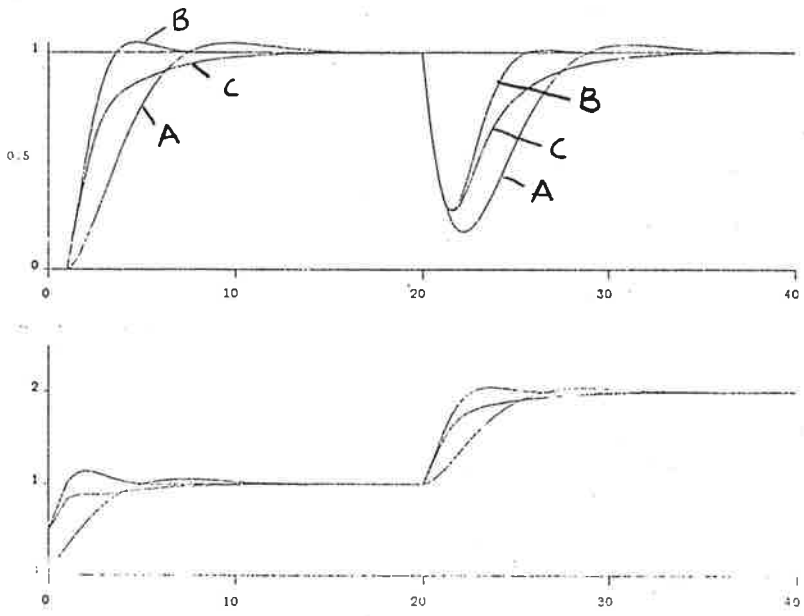

Figure 4.1 Responses to step changes in command inputs and load for the regulators with different ω. The regulators correspond to $\omega_0 = 0.5$ (A), $\omega_0 = 1.0$ (B) and $\omega_0 = 1.5$ (C).

Table 4.2 Regulator parameters for $L = 1$ and $T = 0.1$.

ω_0	k	k_i	T_i	ωT_i	p_1	z_1	ω_c
1.0	−0.002	0.458	−0.003		7.123		0.458
1.2	0.086	0.540	0.160	0.136	6.204	6.259	0.542
1.4	0.147	0.594	0.248	0.245	3.617	4.039	0.601
1.6	0.186	0.621	0.299	0.338	2.536	3.343	0.632
1.8	0.206	0.621	0.332	0.423	1.883	3.010	0.635
2.0	0.213	0.598	0.356	0.503	1.435	2.809	0.612
2.2	0.209	0.556	0.376	0.584	1.103	2.662	0.569
2.4	0.197	0.499	0.395	0.670	0.846	2.532	0.509
2.6	0.180	0.432	0.417	0.766	0.642	2.399	0.439
2.8	0.160	0.359	0.445	0.881	0.479	2.247	0.364
3.0	0.138	0.285	0.485	1.029	0.347	2.061	0.288
3.2	0.117	0.213	0.548	1.241	0.240	1.824	0.214
3.4	0.095	0.144	0.661	1.589	0.154	1.513	0.145
3.6	0.076	0.083	0.916	2.332	0.084	1.092	0.083
3.8	0.058	0.029	2.004	5.385	0.028	0.499	0.029
4.0	0.043	−0.016	−2.628				

than the actual crossover frequency. The reason for the discrepancy is that the closed loop system has a pole $p_3 = -1.9$ and a zero $z_1 = -2.8$ which significantly influences the bandwidth. The benefits by adding proportional action is thus marginal in this case as is seen by comparing the responses for A and B. The crossover frequency increases from 0.46 rad/s to 0.64 rad/s. The figure shows clearly how dominance is lost when the bandwidth is increased.

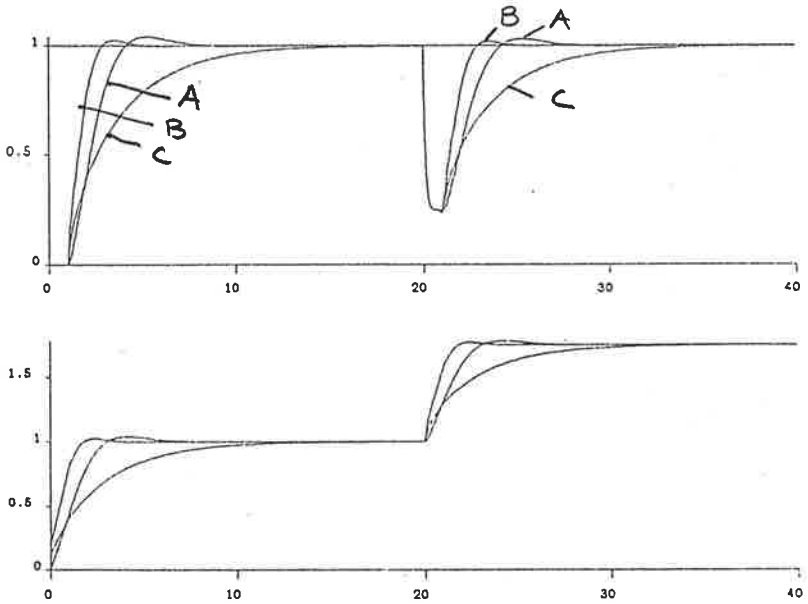

Figure 4.2 Responses to step changes in command inputs and load for the regulators with different ω. The regulators correspond to $\omega_0 = 1.2$ (A), $\omega_0 = 1.8$ (B) and $\omega_0 = 3.0$ (C).

5. Many Equal Lags

In this Section the design method will be applied to a process with the transfer function

$$G(s) = \frac{1}{(s+1)^n}$$

We get

$$G\left(-\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}\right) = r^{-n}(\cos n\varphi - i\sin n\varphi)$$

where

$$r = \sqrt{1 - 2\zeta\omega_0 + \omega_0^2}$$

$$\varphi = \begin{cases} \operatorname{atan}\frac{\omega\sqrt{1-\zeta^2}}{1-\zeta\omega} & \zeta\omega \leq 1 \\ \pi - \operatorname{atan}\frac{\omega\sqrt{1-\zeta^2}}{\zeta\omega-1} & \zeta\omega > 1 \end{cases}$$

Hence

$$A = r^{-n}\cos n\varphi$$

$$B = -r^{-n}\sin n\varphi$$

The regulator parameters are given by (2.4) and (2.5). This gives

$$k = \frac{r^n \sin(n\varphi - \alpha)}{\sqrt{1-\zeta^2}}$$

$$k_i = \frac{r^n\omega_0 \sin n\varphi}{\sqrt{1-\zeta^2}}$$

$$T_i = \frac{\sin(n\varphi - \alpha)}{\omega_0 \sin n\varphi}$$

where

$$\alpha = \operatorname{atan}\frac{\sqrt{1-\zeta^2}}{\zeta}$$

The closed loop system is stable if $k_i > 0$. For $n > 1$ this implies that $\varphi < \pi/n$ i.e.

$$\omega < \frac{\tan\frac{\pi}{n}}{\zeta\tan\frac{\pi}{n} + \sqrt{1-\zeta^2}} \qquad n > 1$$

The values of the regulator parameters for different values of ω_0 are given in Table 4.3. The table indicates that pure integral can be used if the bandwidth is 0.17. Higher bandwidth can be achieved by PI control. Since the steady state loop gain is at most 0.32 for pure proportional it is necessary to have a significant integral action. To select a reasonable value of the design parameter we will apply the dominant pole criterion. This gives $\omega_0 = 0.3$ rad/s. For this value we have $p_3 \approx \omega_0$. The integral gain k_i and the crossover frequency ω_c also have their largest value for this ω_0. The ratio of proportional to integral control is 0.5. Figure 5.1 shows the response to steps in command and load

Table 5.3 Regulator parameters for a process with six identical lags.

ω_0	k	k_i	T_i	ωT_i	p_3	z_1	ω_c
0.15	−0.058	0.073	−0.787	−0.083	0.000	−1.271	0.075
0.20	0.118	0.102	1.161	0.164	0.861	0.861	0.106
0.25	0.234	0.120	1.948	0.344	0.468	0.513	0.131
0.30	0.297	0.125	2.374	0.504	0.306	0.421	0.140
0.35	0.315	0.116	2.725	0.675	0.201	0.367	0.129
0.40	0.295	0.092	3.207	0.907	0.122	0.312	0.100
0.45	0.248	0.058	4.307	1.371	0.062	0.232	0.060
0.50	0.181	0.016	11.556	4.086	0.014	0.087	0.016
0.55	0.103	−0.029	−3.565	−1.386	−0.023	−0.281	−0.029

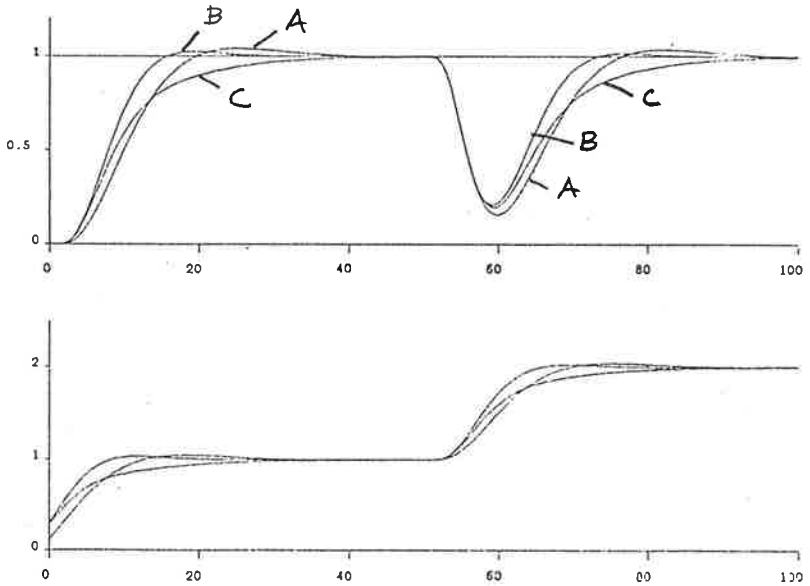

Figure 5.1 Responses to step changes in command inputs and load for the regulators with different ω. The regulators correspond to $\omega_0 = 0.20$ (A), $\omega_0 = 0.30$ (B) and$\omega_0 = 0.40$ (C).

for some of the regulators in Table 5.1. The results indicate that the choice is reasonable and that nothing is gained by choosing a regulator with higher bandwidth. The reason is that the design poles are then no longer dominating. This is verified by a direct calculation of the closed loop poles.

6. Many Different Lags

In this Section the design method will be applied to a process with the transfer function

$$G(s) = \frac{1}{(1+s)(1+\alpha s)(1+\alpha^2 s)\ldots(1+\alpha^{n-1}s)}$$

where $\alpha < 1$. The regulator parameters obtained for $\alpha = 0.1$ are given in Table 6.1.

Table 6.4 Regulator parameters for a four lag process with $\alpha = 0.1$.

ω_0	k	k_i	T_i	ωT_i	p_3	z_1	ω_c
0.5	−0.243	0.258	−0.940				
1.0	0.459	0.954	0.481				
1.5	1.104	1.970	0.560	0.595	0.345	1.784	1.6
2.0	1.694	3.194	0.530	0.750	0.032	1.886	2.0
2.5	2.227	4.510	0.494	0.873	4.730	2.025	2.6
3.0	2.705	5.809	0.466	0.988	5.205	2.148	3.0
3.5	3.126	6.980	0.448	1.109	5.676	2.233	3.4
4.0	3.491	7.915	0.441	1.248	4.926	2.267	3.7
4.5	3.801	8.508	0.447	1.422	4.182	2.238	3.9
5.0	4.054	8.654	0.469	1.657	3.445	2.134	4.1
5.5	4.252	8.249	0.515	2.005	2.713	1.940	4.2
6.0	4.393	7.194	0.611	2.591	1.988	1.637	4.2
6.5	4.479	5.387	0.831	3.822	1.268	1.203	4.2
7.0	4.509	2.731	1.651	8.172	0.544	0.606	4.1
7.5	4.483	−0.870	−5.152	−27.326	−0.154	−0.194	4.0
8.0	4.401	−5.512	−0.798				

With pure integral control we get a bandwidth $\omega_0 = 0.67$ rad/s. Table 6.1 indicates that the bandwidth can be increased significantly by introducing proportional action. Approximating the time constants 0.1, 0.01 and 0.001 by one time constant of 0.111 and using the estimate (3.8) for pole domination we get $\omega_0 < 4.6$ rad/s. The steady state loop gain is 3.8 at this frequency which implies that the integral action does not have to dominate. Table 6.1 shows the regulator gains and some quantities that are useful to judge pole dominance for different values of the design parameter ω_0. With the criterion $p_3 = \omega_0$ we find that $\omega_0 = 4.5$. For this value of ω_0 the zero is located at $z_1 = -2.2$. This zero is to the right of the dominant poles. It will thus result in an increased overshoot. The overshoot is easily reduced by a modification of the control law which puts the zero in $z_1 = -1/(bT_i)$. A value of $b < 1$ is required. This modification will also lead to an increased crossover frequency. For $\omega_0 = 4.5$ the value of ωT_i is 1.4 which means that proportional action dominates over integral action. Notice that the maximum of the crossover frequency is obtained for $\omega_0 = 6.5$ which is higher than the maximum value for dominance.

Figure 6.1 shows responses to stepchanges in command and load signals for different values of the design parameter ω_0. These responses agree well with the prediction made from the data in Table 6.1.

Notice that this case is different from the other cases because the performance can be increased drastically when an I regulator is replaced by a PI regulator. Also notice that there is a significant overshoot in the response to

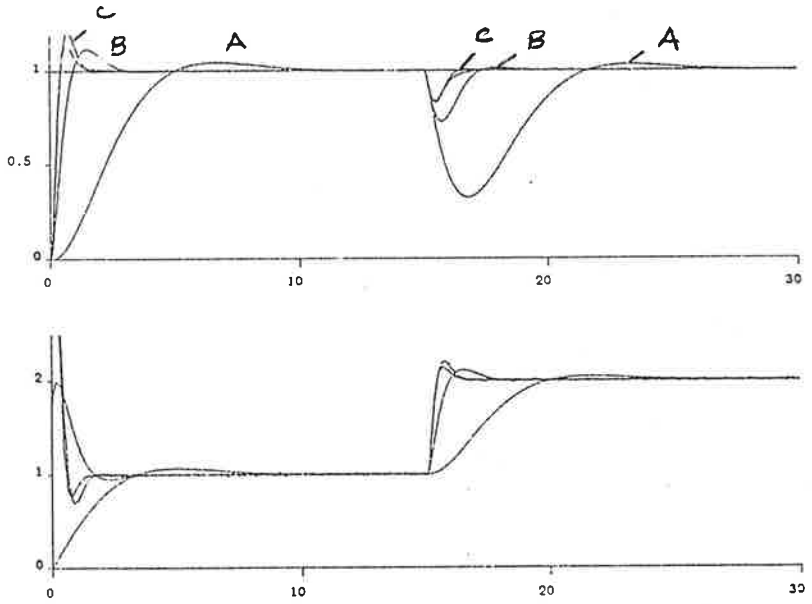

Figure 6.1 Responses to step changes in load and command signals for different PI regulators. The regulators correspond to $\omega_0 = 0.67$ (A), $\omega_0 = 2.0$ (B) and $\omega_0 = 4.5$ and 6.0(C).

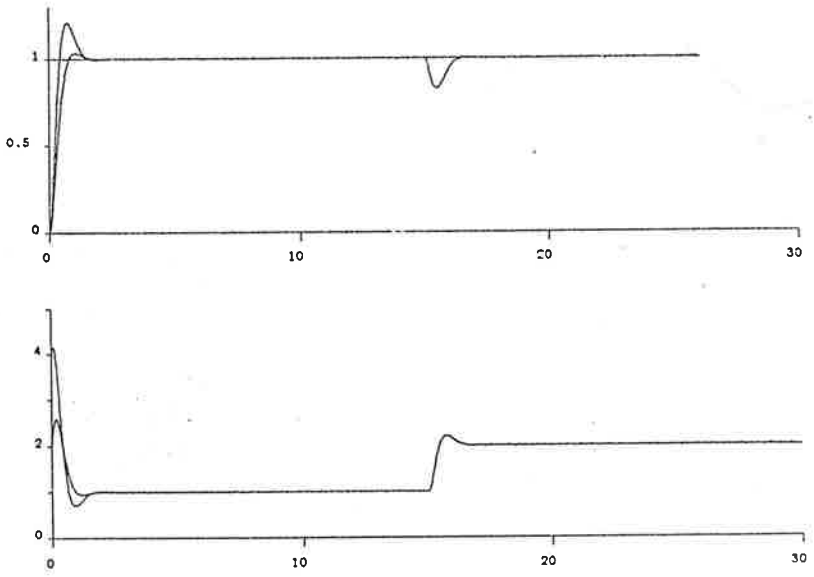

Figure 6.2 Response to step changes in load and command signals for a PI regulator with $\omega_0 = 4.5$ and $b = 0.5$.

step commands. This is caused by the zero z_1. The overshoot can be reduced by introducing the "b-modification" of the PI regulator. If b is chosen as z_1/p_3 the zero z_1 cancels the pole p_3 and the response is close to that of a second order system. In this particular case we have $z_1/p_3 = 0.5$. The response of such a regulator is shown in Figure 6.2. We can thus accept $\omega_0 T > 1$. Figure 6.1 shows responses to step changes in command and load signals.

7. System Identification

To apply the design method it is necessary to know the value of the transfer function of the process at $s = -\sigma + i\omega = -\zeta\omega_0 + i\omega_0\sqrt{1-\zeta^2}$. It is highly desirable to have a measurement technique which gives this value directly. In this Section we will give such a method.

Let u be the process input and y the process output and let U and Y denote the corresponding Laplace transforms. It then follows that

$$Y(s) = G(s)U(s) + H(s) \tag{7.1}$$

where G is the transfer function of the process and F the Laplace transform of a term which depends on the initial conditions of the process. The desired value of the transfer function can be obtained by calculating

$$\begin{aligned} Y(-\sigma + i\omega) &= \int_0^\infty e^{\sigma t} e^{-i\omega t} y(t) dt \\ U(-\sigma + i\omega) &= \int_0^\infty e^{\sigma t} e^{-i\omega t} u(t) dt \end{aligned} \tag{7.2}$$

If the process is initially at rest the desired value can then be calculated from (7.1).

We thus obtain the following method to determine $G(-\sigma+i\omega)$. Start with the system at rest i.e. $F = 0$. Apply a disturbance calculate $Y(-\sigma + i\omega)$ and $U(-\sigma + i\omega)$ from (7.2) and $G(-\sigma + i\omega)$ from (7.1). Notice that the weighting function increases exponentially. To ensure that the integrals converge it is thus necessary that both $u(t)$ and $y(t)$ decreases more rapidly than $e^{-\sigma t}$.

In open loop experiments $U(-\sigma + i\omega)$ will change exist if the input is of pulse type i.e. $u(t) = 0$ for $t \geq t_p$. The integral $Y(-\sigma + i\omega)$ will converge only if σ is chosen so that all process poles have real parts less than $-\sigma$. This can be used to explore the location of the process poles if the experiment is repeated for different σ.

The method can also be applied to closed loop experiments. The integrals will, however, converge only if σ is chosen so that all closed loop poles have real parts less than $-\sigma$. If the method is applied to a closed loop experiment it can thus be used to investigate pole dominance.

Notice that the calculations (7.2) can be performed in parallel for different values of ω_0. It is thus possible to determine several values of the transfer function from one experiment. The calculations can be performed in different ways. A straightforward method is to generate trigonometric functions and to use quadrature. Another method is to generate the signal $e^{\sigma t + i\omega t}$ from a dynamical system. The functions $x_1 = e^{\sigma t}\sin\omega t$ and $x_2 = e^{\sigma t}\cos\omega t$ can e.g. be generated as the states x_1 and x_2 of the differential equation

$$\frac{dx}{dt} = \begin{pmatrix} \sigma & \omega \\ -\omega & \sigma \end{pmatrix} x$$

with initial conditions $x_1(0) = 0$ and $x_2(0) = 1$. To explore possible divergence of the integrals it is useful to evaluate the integrals over each half period separately i.e.

$$Y_n = \int_{n\frac{\pi}{\omega}}^{(n+1)\frac{\pi}{\omega}} y(t)e^{\sigma t} \sin \omega t \, dt \tag{7.3}$$

The rate of decay of $y(t)$ can be estimated by looking at the ratios $|Y_{n+1}/Y_n|$. This can be used to estimate the dominant pole at the open loop or closed loop dynamics.

In an on-line situation the method can be applied whenever there is a transient that is generated externally i.e. from set point changes or from manual control actions. To ensure an accurate estimate it is also necessary to make sure that the system is in steady state. An algorithm which detects that the system is in steady state is thus an essential ingredient.

To be able to use step changes it is also useful to high pass filter inputs and outputs before applying them to the algorithm. To have a reasonable precision of the estimate it is necessary that the process is properly excited. The value $|U(-\sigma + i\omega)|$ is a direct measure of excitation. The calculation should not be performed unless this value is sufficiently large.

8. Conclusions

A design procedure based on the positioning of the dominant poles of the closed loop system has been proposed. The design is based on knowledge of the plant transfer function at the dominant pole. The specifications are expressed as the relative damping ζ and frequency ω_0 of the dominant poles. The design method also gives several quantitites which are useful in order to reason about the feedback loop. The method gives a range of admissible bandwidths. The range is limited by the frequencies where the specifications can be satisfied by pure integral and pure proportional control. The numerical value of the static loop gain for pure proportional control tells if proportional control is sufficient or if integral action is needed to obtain a tolerable steady state error. The size of the range indicates if it is worthwhile to introduce proportional action or if pure integral control is sufficient. Useful information can also be derived from the variations of the proportional and derivative gains over the bandwidth range. The proportional gain k will be zero at the bandwidth corresponding to pure integral control. It increases with increasing frequency. In some cases it has a maximum. The integrating gain k_i is zero for the bandwidth corresponding to pure proportional control. It increases with decreasing bandwidth. In all cases considered so far it has a maximum for some frequency in the range. (Can this be proven?)

Several criteria for evaluating pole dominance have been investigated. These include calculation of poles and zeros to the right of the dominating poles and calculation of the crossover frequency. The crossover frequency is a good indicator which is also easy to compute. At least for the examples discussed the poles seem to be dominating at the low bandwidth range. It has been found empirically that the frequency where the integrating gain k_i has its maximum in a reasonable choice for the bandwidth. This choice of bandwidth is also close to the frequency where the open loop dynamics has a phaseshift of 90°C.

It appears that the design poles are not dominating for frequencies close to the frequency corresponding to pure proportional control. The maximum static loop gain and the quantity $\omega_0 T_i$ are also useful in order to evaluate domination. Can this be established theoretically? The design method can be applied to the standard model of a system with a time delay L and a time constant. In this case we can obtain simple analytical design rules which can be used in an auto-tuner. The design calculations for different cases are given in the Appendices.

Appendix A

In this and the following appendices we give programs used for the design calculations. We also give some quantities that are useful to assess the control design namely poles and zeros on the real axis (p_3, z_1), crossover frequency ω_c, phase margin *phm*, amplitude and phase slopes at crossover, amplitude margin *am*, the frequency *w*180 where the loop transfer function has a phase lag of 180°.

Poles on the Real Axis

Assume that the characteristic equation has a pole $s = -p$ on the negative real axis. Inserting this into the characteristic equation we get

$$1 + \left(k - \frac{k_i}{p}\right) G(-p) = 0$$

Hence

$$p = \frac{k_i G(-p)}{1 + kG(-p)}$$

where

$$G(-p) = \frac{e^{pL}}{1 - pT}$$

This equation is used to iterate for the real pole when k_i is small. For large values of k_i the iteration

$$p = \frac{1}{L} \ell n \frac{p(1 - pT)}{k_i - pk}$$

is used instead.

Crossover Frequency The crossover frequency is given by the condition

$$\left| \left(k + \frac{k_i}{i\omega}\right) G(i\omega) \right| = 1$$

Introduce

$$G(i\omega) = A(\omega) + iB(\omega)$$

Hence

$$\left(k^2 + \frac{k_i^2}{\omega^2}\right)(A^2 + B^2) = 1$$

Solving for ω we get

$$\omega = k_i \sqrt{\frac{A^2(\omega) + B^2(\omega)}{1 - k^2(a^2(\omega) + B^2(\omega))}}$$

This equation is used to iterate for the crossover frequency. The programs used to analyse this case are listed below.

Slopes at Crossover The amplitude slope is defined as

$$s_a = \frac{d \log |G(i\omega)|}{d \log \omega}$$

With

$$G(i\omega) = \frac{e^{-i\omega L}}{1 + i\omega T}$$

we get

$$\log |G(i\omega)| = -\frac{1}{2} \log(1 + \omega^2 T^2)$$

Hence

$$s_a = -\frac{\omega^2 T^2}{1 + \omega^2 T^2}$$

The phase slope is

$$s_{pn} = \frac{d \arg G(i\omega)}{d \log \omega}$$

With

$$\arg G(i\omega) = -\omega L - \arctan \omega T$$

Hence

$$s_{ph} = -\omega L - \frac{\omega T}{1 + \omega^2 T^2}$$

The calculation of these quantities is straightforward apart from a few cases where implicit equations have to be solved. The calculations will be presented in detail for one case in this appendix. For the remaining cases we will simply give the computer code.

```
program PIdesignLTX;
{Dominant pole design of PI regulator for process with lag T and delay T}
{This program calculates many auxiliary quantities}
{ which can be used to evaluate design methods}
 var
  w, z, k, T, L, Ti, ki, pi, arg : real;
  sq, wo, alfa, s, r, fi, x1, A, B : real;
  wTi, fiproc, rproc, phase, angle : real;
  p1, z1, p10, z10, func, wc, A1, B1, phm, w1, am, fix : real;
  s1, s2, aslope, pslope : real;
  ni : integer;
  design : boolean;
begin
 design := true;
 pi := 3.14159;
 z := 0.707;
 sq := sqrt(1 - z * z);
 T := 1;
 L := 1;
 writeln('Time delay = ', L : 4 : 1, '  Time constant = ', T : 4 : 1);
 wo := 0.40;
 if design then
  writeln(' wo   k     ki     p1     z1     p10    z10    Ti     wTi ')
 else
  writeln(' wo   wc    phm   aslope pslope w180   am     rproc   angle');

 repeat
  w := wo * sq;
  s := wo * z;
  r := exp(s * L) / sqrt((1 - s * T) * (1 - s * T) + w * T * w * T);
  x1 := arctan(w * T / (1 - s * T));
  if s * T < 1 then
   fi := -w * L - x1
  else
   fi := -w * L - (pi + x1);
  A := r * cos(fi);
  B := r * sin(fi);
  arg := 180 * fi / pi;
  k := -(w * A + s * B) / w / (A * A + B * B);
  ki := -(s * s + w * w) * B / w / (A * A + B * B);
  Ti := k / ki;

{Process gain and phase at w}
  wTi := w * Ti;
  fiproc := -w * L - arctan(w * T);
  rproc := 1 / sqrt(1 + w * T * w * T);
```

```
    writeln(wo : 4 : 1, k : 8 : 3, ki : 8 : 3, p1 : 8 : 3, z1 : 8 : 3, p10 : 8 : 3, z10 : 8 : 3, Ti : 8 : 3,
        wTi : 8 : 3)
  else
    writeln(wo : 4 : 1, wc : 8 : 3, phm : 6 : 1, aslope : 8 : 3, pslope : 8 : 3, w1 : 8 : 3, am : 8 : 3,
        rproc : 8 : 3, angle : 8 : 1);
  wo := wo + 0.20;
until ki < 0;


end.
```

Text

Time delay = 1.0 Time constant = 0.1

wo	k	ki	p1	z1	p10	z10	Ti	wTi
1.0	-0.002	0.458	7.123	-296.404	7.123	-296.404	-0.003	-0.002
1.2	0.086	0.540	6.204	6.259	5.170	5.216	0.160	0.136
1.4	0.147	0.594	3.617	4.039	2.584	2.885	0.248	0.245
1.6	0.186	0.621	2.536	3.343	1.585	2.089	0.299	0.338
1.8	0.206	0.621	1.883	3.010	1.046	1.672	0.332	0.423
2.0	0.213	0.598	1.435	2.809	0.718	1.405	0.356	0.503
2.2	0.209	0.556	1.103	2.662	0.501	1.210	0.376	0.584
2.4	0.197	0.499	0.846	2.532	0.352	1.055	0.395	0.670
2.6	0.180	0.432	0.642	2.399	0.247	0.923	0.417	0.766
2.8	0.160	0.359	0.479	2.247	0.171	0.802	0.445	0.881
3.0	0.138	0.285	0.347	2.061	0.116	0.687	0.485	1.029
3.2	0.117	0.213	0.240	1.824	0.075	0.570	0.548	1.241
3.4	0.095	0.144	0.154	1.513	0.045	0.445	0.661	1.589
3.6	0.076	0.083	0.084	1.092	0.023	0.303	0.916	2.332
3.8	0.058	0.029	0.028	0.499	0.007	0.131	2.004	5.385
4.0	0.043	-0.016	-0.015	-0.381	-0.004	-0.095	-2.628	-7.434

Text

wo	wc	phm	aslope	pslope	w180	am	rproc	angle
1.0	0.458	61.0	-1.002	-0.506	4.292	2.374	0.998	-44.6
1.2	0.542	60.8	-0.995	-0.510	1.666	1.855	0.996	-53.5
1.4	0.601	60.6	-0.982	-0.515	1.813	1.660	0.995	-62.4
1.6	0.632	60.9	-0.969	-0.512	1.900	1.570	0.994	-71.3
1.8	0.635	61.9	-0.961	-0.496	1.954	1.556	0.992	-80.2
2.0	0.612	63.7	-0.958	-0.465	1.991	1.605	0.990	-89.1
2.2	0.569	66.2	-0.960	-0.421	2.021	1.718	0.988	-98.0
2.4	0.509	69.3	-0.964	-0.367	2.049	1.903	0.986	-106.9
2.6	0.439	72.7	-0.970	-0.306	2.080	2.183	0.984	-115.8
2.8	0.364	76.3	-0.976	-0.242	2.118	2.599	0.981	-124.7
3.0	0.288	79.8	-0.982	-0.180	2.168	3.230	0.978	-133.5
3.2	0.214	83.2	-0.987	-0.120	2.237	4.228	0.975	-142.4
3.4	0.145	86.3	-0.991	-0.065	2.337	5.930	0.972	-151.3
3.6	0.083	89.1	-0.994	-0.016	2.484	9.170	0.969	-160.2
3.8	0.029	91.5	-0.997	0.026	2.695	15.923	0.966	-169.0
4.0	-0.016	93.5	-0.998	0.060	2.979	22.876	0.962	-177.9

```
    angle := 180 * fiproc / pi;

  {Pole on real axis}
    p1 := ki / (1 + k);
    ni := 1;
    repeat
      func := exp(p1 * L) / (1 - p1 * T);
      p1 := ki * func / (1 + k * func);
      ni := ni + 1;
    until ni = 30;
  {It is important to use a sufficiently large value of ni}
  {This iteration gives good values for large wo but poorer for small}

    z1 := 1 / Ti;
    p10 := p1 / wo;
    z10 := z1 / wo;

  {Crossover frequency and phase margin}
    wc := w;
    ni := 1;
    repeat
      ni := ni + 1;
      A1 := (cos(wc * L) + wc * T * sin(wc * L)) / sqrt(1 + wc * wc * T * T);
      B1 := (sin(wc * L) - wc * T * cos(wc * L)) / sqrt(1 + wc * wc * T * T);
      wc := ki * sqrt((A1 * A1 + B1 * B1) / (1 - k * k * (A1 * A1 + B1 * B1)));
    until ni = 10;
    fix := -pi / 2 + arctan(k * wc / ki);
    phm := 180 / pi * (pi / 2 - wc * L - arctan(wc * T) + arctan(k * wc / ki));

  {Amplitude and phase slopes at crossover}
    s1 := wc * T;
    s2 := wc * k;
    pslope := -s1 / (1 + s1 * s1) - wc * L + ki * s2 / (ki * ki + s2 * s2);
    aslope := -s1 * s1 / (1 + s1 * s1) - ki * ki / (s2 * s2 + ki * ki);

  {Amplitude margin and w180}
    w1 := w;
    ni := 1;
    repeat
      w1 := (pi - arctan(ki / w1 / k) - arctan(w1 * T)) / L;
      ni := ni + 1;
    until ni = 10;
    am := sqrt(1 + w1 * w1 * T * T) / sqrt(k * k + ki * ki / w1 * w1);

  {Write result}
    if design then
```

```
program PIdesignLTX;
{Dominant pole design of PI regulator for process with lag T and delay T}
{This program calculates many auxiliary quantities}
{ which can be used to evaluate design methods}
 var
  w, z, k, T, L, Ti, ki, pi, arg : real;
  sq, wo, alfa, s, r, fi, x1, A, B : real;
  wTi, fiproc, rproc, phase, angle : real;
  p1, z1, p10, z10, func, wc, A1, B1, phm, w1, am, fix : real;
  s1, s2, aslope, pslope : real;
  ni : integer;
  design : boolean;
begin
 design := true;
 pi := 3.14159;
 z := 0.707;
 sq := sqrt(1 - z * z);
 T := 0.1;
 L := 1;
 writeln('Time delay = ', L : 4 : 1, '  Time constant = ', T : 4 : 1);
 wo := 1.0;
 if design then
  writeln(' wo   k     ki     p1     z1     p10    z10    Ti     wTi ')
 else
  writeln(' wo   wc    phm   aslope pslope w180   am     rproc   angle');

 repeat
  w := wo * sq;
  s := wo * z;
  r := exp(s * L) / sqrt((1 - s * T) * (1 - s * T) + w * T * w * T);
  x1 := arctan(w * T / (1 - s * T));
  if s * T < 1 then
   fi := -w * L - x1
  else
   fi := -w * L - (pi + x1);
  A := r * cos(fi);
  B := r * sin(fi);
  arg := 180 * fi / pi;
  k := -(w * A + s * B) / w / (A * A + B * B);
  ki := -(s * s + w * w) * B / w / (A * A + B * B);
  Ti := k / ki;

{Process gain and phase at w}
  wTi := w * Ti;
  fiproc := -w * L - arctan(w * T);
  rproc := 1 / sqrt(1 + w * T * w * T);
```

```
    z1 := 1 / Ti;
    p10 := p1 / wo;
    px := p1 / z1;

{Crossover frequency and phase margin}
    wc := w;
    ni := 1;
    repeat
      ni := ni + 1;
      A1 := (cos(wc * L) + wc * T * sin(wc * L)) / sqrt(1 + wc * wc * T * T);
      B1 := (sin(wc * L) - wc * T * cos(wc * L)) / sqrt(1 + wc * wc * T * T);
      wc := ki * sqrt((A1 * A1 + B1 * B1) / (1 - k * k * (A1 * A1 + B1 * B1)));
    until ni = 10;
    wx := wc / (wo * sqrt(sqrt(4 * z * z * z * z + 1) - 2 * z * z));
    fix := -pi / 2 + arctan(k * wc / ki);
    phm := 180 / pi * (pi / 2 - wc * L - arctan(wc * T) + arctan(k * wc / ki));

{Write result}
    writeln(wo : 4 : 1, k : 8 : 3, ki : 8 : 3, wx : 8 : 3, p10 : 8 : 3, px : 8 : 3, Ti : 8 : 3, wTi : 8 : 3);
    wo := wo + 0.10;
  until ki < 0;


end.
```

Text

wo	k	ki	wc/wce	p10	p1/z1	Ti	wTi
0.4	-0.104	0.201	0.785	14.748	-3.051	-0.517	-0.146
0.5	0.050	0.276	0.858	2.625	0.238	0.182	0.064
0.6	0.179	0.346	0.911	13.195	4.092	0.517	0.219
0.7	0.284	0.408	0.944	3.523	1.719	0.697	0.345
0.8	0.367	0.457	0.953	2.458	1.582	0.805	0.455
0.9	0.431	0.490	0.938	1.684	1.331	0.878	0.559
1.0	0.476	0.508	0.897	1.189	1.114	0.937	0.663
1.1	0.504	0.508	0.830	0.884	0.966	0.994	0.773
1.2	0.518	0.490	0.742	0.665	0.844	1.058	0.898
1.3	0.520	0.456	0.638	0.493	0.731	1.141	1.049
1.4	0.511	0.406	0.524	0.354	0.624	1.258	1.245
1.5	0.492	0.342	0.406	0.245	0.529	1.440	1.527
1.6	0.465	0.265	0.290	0.158	0.445	1.759	1.990
1.7	0.433	0.177	0.179	0.090	0.373	2.446	2.940
1.8	0.395	0.081	0.076	0.035	0.310	4.905	6.244
1.9	0.354	-0.022	-0.020	-0.009	0.255	-15.778	-21.200

```
program PIdesignLT;
{Dominant pole design of PI regulator for process with lag T and delay T}
 var
  w, z, k, T, L, Ti, ki, pi, arg : real;
  sq, wo, alfa, s, r, fi, x1, A, B : real;
  wTi, fiproc, rproc, phase, angle : real;
  p1, z1, p10, px, func, wc, A1, B1, phm, w1, am, fix : real;
  wx : real;
  ni : integer;

begin
 pi := 3.14159;
 z := 0.707;
 sq := sqrt(1 - z * z);
 T := 1;
 L := 1;
 writeln('Time delay = ', L : 4 : 1, '  Time constant = ', T : 4 : 1);
 wo := 0.40;

 writeln(' wo   k     ki    wc/wce p10   p1/z1  Ti    wTi ');

 repeat
  w := wo * sq;
  s := wo * z;
  r := exp(s * L) / sqrt((1 - s * T) * (1 - s * T) + w * T * w * T);
  x1 := arctan(w * T / (1 - s * T));
  if s * T < 1 then
   fi := -w * L - x1
  else
   fi := -w * L - (pi + x1);
  A := r * cos(fi);
  B := r * sin(fi);
  arg := 180 * fi / pi;
  k := -(w * A + s * B) / w / (A * A + B * B);
  ki := -(s * s + w * w) * B / w / (A * A + B * B);
  Ti := k / ki;
  wTi := w * Ti;

{Pole on real axis}
  p1 := ki / (1 + k);
  ni := 1;
  repeat
   func := exp(p1 * L) / (1 - p1 * T);
   p1 := ki * func / (1 + k * func);
   ni := ni + 1;
  until ni = 10;
```

```
    writeln(wo : 4 : 1, k : 8 : 3, ki : 8 : 3, p1 : 8 : 3, z1 : 8 : 3, p10 : 8 : 3, z10 : 8 : 3, Ti : 8 : 3,
        wTi : 8 : 3)
  else
    writeln(wo : 4 : 1, wc : 8 : 3, phm : 6 : 1, aslope : 8 : 3, pslope : 8 : 3, w1 : 8 : 3, am : 8 : 3,
        rproc : 8 : 3, angle : 8 : 1);
  wo := wo + 0.10;
until ki < 0;

end.
```

Text

Time delay = 1.0 Time constant = 1.0

wo	k	ki	p1	z1	p10	z10	Ti	wTi
0.4	-0.104	0.201	0.034	-1.933	0.086	-4.833	-0.517	-0.146
0.5	0.050	0.276	6.544	5.503	13.088	11.006	0.182	0.064
0.6	0.179	0.346	3.439	1.935	5.732	3.224	0.517	0.219
0.7	0.284	0.408	2.519	1.435	3.599	2.049	0.697	0.345
0.8	0.367	0.457	1.966	1.242	2.458	1.553	0.805	0.455
0.9	0.431	0.490	1.573	1.139	1.747	1.265	0.878	0.559
1.0	0.476	0.508	1.269	1.067	1.269	1.067	0.937	0.663
1.1	0.504	0.508	1.024	1.006	0.931	0.915	0.994	0.773
1.2	0.518	0.490	0.820	0.945	0.683	0.788	1.058	0.898
1.3	0.520	0.456	0.646	0.877	0.497	0.674	1.141	1.049
1.4	0.511	0.406	0.497	0.795	0.355	0.568	1.258	1.245
1.5	0.492	0.342	0.367	0.694	0.245	0.463	1.440	1.527
1.6	0.465	0.265	0.253	0.569	0.158	0.355	1.759	1.990
1.7	0.433	0.177	0.152	0.409	0.090	0.241	2.446	2.940
1.8	0.395	0.081	0.063	0.204	0.035	0.113	4.905	6.244
1.9	0.354	-0.022	-0.016	-0.063	-0.009	-0.033	-15.778	-21.200

Text

wo	wc	phm	aslope	pslope	w180	am	rproc	angle
0.4	0.202	61.0	-1.028	-0.500	2.583	12.231	0.962	-32.0
0.5	0.276	61.6	-1.068	-0.483	0.974	4.979	0.943	-39.7
0.6	0.352	60.8	-1.078	-0.489	1.248	4.104	0.921	-47.3
0.7	0.425	59.1	-1.072	-0.513	1.393	3.451	0.896	-54.7
0.8	0.491	57.3	-1.059	-0.545	1.466	3.029	0.870	-61.9
0.9	0.543	55.9	-1.042	-0.574	1.510	2.775	0.844	-68.9
1.0	0.577	55.4	-1.024	-0.592	1.541	2.641	0.816	-75.8
1.1	0.588	56.2	-1.002	-0.589	1.568	2.599	0.789	-82.5
1.2	0.573	58.6	-0.978	-0.561	1.596	2.640	0.762	-88.9
1.3	0.534	62.7	-0.951	-0.505	1.628	2.762	0.736	-95.3
1.4	0.472	68.4	-0.922	-0.419	1.666	2.979	0.711	-101.4
1.5	0.392	75.6	-0.891	-0.304	1.714	3.314	0.686	-107.5
1.6	0.299	84.0	-0.865	-0.161	1.774	3.804	0.662	-113.4
1.7	0.196	93.3	-0.850	0.005	1.849	4.498	0.639	-119.1
1.8	0.088	103.2	-0.852	0.188	1.942	5.419	0.618	-124.8
1.9	-0.024	113.4	-0.876	0.379	2.055	6.451	0.597	-130.3

```
    angle := 180 * fiproc / pi;

{Pole on real axis}
    p1 := ki / (1 + k);
    ni := 1;
    repeat
      func := exp(p1 * L) / (1 - p1 * T);
      p1 := ki * func / (1 + k * func);
      ni := ni + 1;
    until ni = 30;
{It is important to use a sufficiently large value of ni}
{This iteration gives good values for large wo but poorer for small}

    z1 := 1 / Ti;
    p10 := p1 / wo;
    z10 := z1 / wo;

{Crossover frequency and phase margin}
    wc := w;
    ni := 1;
    repeat
      ni := ni + 1;
      A1 := (cos(wc * L) + wc * T * sin(wc * L)) / sqrt(1 + wc * wc * T * T);
      B1 := (sin(wc * L) - wc * T * cos(wc * L)) / sqrt(1 + wc * wc * T * T);
      wc := ki * sqrt((A1 * A1 + B1 * B1) / (1 - k * k * (A1 * A1 + B1 * B1)));
    until ni = 10;
    fix := -pi / 2 + arctan(k * wc / ki);
    phm := 180 / pi * (pi / 2 - wc * L - arctan(wc * T) + arctan(k * wc / ki));

{Amplitude and phase slopes at crossover}
    s1 := wc * T;
    s2 := wc * k;
    pslope := -s1 / (1 + s1 * s1) - wc * L + ki * s2 / (ki * ki + s2 * s2);
    aslope := -s1 * s1 / (1 + s1 * s1) - ki * ki / (s2 * s2 + ki * ki);

{Amplitude margin and w180}
    w1 := w;
    ni := 1;
    repeat
      w1 := (pi - arctan(ki / w1 / k) - arctan(w1 * T)) / L;
      ni := ni + 1;
    until ni = 10;
    am := sqrt(1 + w1 * w1 * T * T) / sqrt(k * k + ki * ki / w1 * w1);

{Write result}
    if design then
```

Appendix B - N Equal Lags

```
program PIdesignLT;
{Dominant pole design of PI regulator for process n equal lags T}
 var
  w, z, k, T, Ti, ki, pi, arg : real;
  sq, wo, alfa, s, r, fi, fi1, x1, A, B : real;
  wTi, fiproc, fireg, rproc, rreg, phase, r1, dist, angle : real;
  p1, z1, p10, z10, func, wc, A1, B1, phm, w1, am, fix : real;
  s1, s2, aslope, pslope, wfi : real;
  design : boolean;
  n, ni : integer;

begin
 pi := 3.14159;
 z := 0.707;
 sq := sqrt(1 - z * z);
 T := 1;
 n := 6;
 wo := 0.150;
 design := false;

 writeln('System with', n : 3 : 1, ' equal lags');
 if design then
  writeln(' wo    k     ki    p1    z1    p10   z10   Ti    wTi ')
 else
  writeln(' wo    wc    phm   aslope pslope w180  am    rproc  angle');

 repeat
{Compute regulator gains k and ki}
  w := wo * sq;
  s := wo * z;
  r := 1 / sqrt((1 - s) * (1 - s) + w * w);
  r := exp(n * ln(r));
  if s < 1 then
   fi := -arctan(w / (1 - s))
  else
   fi := -(pi + arctan(w / (1 - s)));
  fi := n * fi;
  A := r * cos(fi);
  B := r * sin(fi);
  k := -(w * A + s * B) / w / (A * A + B * B);
  ki := -(s * s + w * w) * B / w / (A * A + B * B);
  Ti := k / ki;
  wTi := w * Ti;

{Calculate process gain and phase}
  rproc := exp(-n / 2 * ln(1 + w * w));
```

```
    fiproc := -n * arctan(w);
    angle := 180 * fiproc / pi;

{Pole on real axis}
    p1 := ki / (1 + k);
    ni := 1;
    repeat
      if p1 < 1 then
        func := exp(-n * ln(1 - p1))
      else
        func := exp(-n * ln(p1 - 1));
      p1 := ki * func / (1 + k * func);
      ni := ni + 1;
    until ni = 30;

    z1 := 1 / Ti;
    p10 := p1 / wo;
    z10 := z1 / wo;


{Crossover frequency and phase margin}
    wc := w;
    ni := 1;
    repeat
      ni := ni + 1;
      r1 := exp(-n * ln(1 + wc * wc * T * T));
      wc := ki / sqrt(r1 - k * k);
    until ni = 10;
    phm := 180 / pi * (pi / 2 - n * arctan(wc * T) + arctan(k * wc / ki));

{Amplitude and phase slopes at crossover}
    s1 := wc * T;
    s2 := wc * k;
    pslope := -n * s1 / (1 + s1 * s1) + ki * s2 / (ki * ki + s2 * s2);
    aslope := -n * s1 * s1 / (1 + s1 * s1) - ki * ki / (s2 * s2 + ki * ki);

{Amplitude margin and w180}
    w1 := w;
    ni := 1;
    repeat
      wfi := (pi / 2 + arctan(w1 * k / ki)) / n;
      w1 := 1 / T * sin(wfi) / cos(wfi);
      ni := ni + 1;
    until ni = 10;
    am := exp(n / 2 * ln(1 + w1 * w1 * T * T)) / sqrt(k * k + ki * ki / w1 / w1);
```

```
{Write result}
  if design then
   writeln(wo : 5 : 2, k : 8 : 3, ki : 8 : 3, p1 : 8 : 3, z1 : 8 : 3, p10 : 8 : 3, z10 : 8 : 3, Ti : 8 : 3,
        wTi : 8 : 3)
  else
   writeln(wo : 5 : 2, wc : 8 : 3, phm : 6 : 1, aslope : 8 : 3, pslope : 8 : 3, w1 : 8 : 3, am : 8 : 3,
        rproc : 8 : 3, angle : 8 : 1);
  wo := wo + 0.05;
 until ki < 0;

end.
```

Text

System with 6 equal lags

wo	k	ki	p1	z1	p10	z10	Ti	wTi
0.15	-0.058	0.073	0.000	-1.271	0.000	-8.476	-0.787	-0.083
0.20	0.118	0.102	0.861	0.861	4.305	4.305	1.161	0.164
0.25	0.234	0.120	0.468	0.513	1.872	2.053	1.948	0.344
0.30	0.297	0.125	0.306	0.421	1.021	1.404	2.374	0.504
0.35	0.315	0.116	0.201	0.367	0.573	1.048	2.725	0.675
0.40	0.295	0.092	0.122	0.312	0.306	0.780	3.207	0.907
0.45	0.248	0.058	0.062	0.232	0.138	0.516	4.307	1.371
0.50	0.181	0.016	0.014	0.087	0.028	0.173	11.556	4.086
0.55	0.103	-0.029	-0.023	-0.281	-0.042	-0.510	-3.565	-1.386

Text

System with 6 equal lags

wo	wc	phm	aslope	pslope	w180	am	rproc	angle
0.15	0.075	61.0	-1.030	-0.505	0.235	3.706	0.967	-36.3
0.20	0.106	60.7	-1.052	-0.509	0.336	4.224	0.942	-48.3
0.25	0.131	59.6	-1.040	-0.531	0.387	3.909	0.912	-60.2
0.30	0.140	60.6	-1.016	-0.524	0.412	3.766	0.876	-71.9
0.35	0.129	65.4	-0.988	-0.447	0.430	4.015	0.837	-83.4
0.40	0.100	73.6	-0.966	-0.302	0.450	4.833	0.794	-94.8
0.45	0.060	83.9	-0.959	-0.117	0.481	6.794	0.749	-105.9
0.50	0.016	94.9	-0.969	0.082	0.543	11.858	0.702	-116.8
0.55	-0.029	105.9	-0.994	0.276	0.172	5.548	0.655	-127.5

Appendix C - Many Different Lags

```
program PIdesignMutilLag;
{Dominant pole design of PI regulator }
{for process with multiple lags with geometric time constants}
{This program calculates many auxiliary quantities}
{ which can be used to evaluate design methods}
 var
  w, z, k, T, L, Ti, ki, pi, arg : real;
  sq, wo, w1, w2, alfa, s, s1, r, rp, fi, fip, x1, A, B : real;
  wTi, fireg, rreg, phase, r1, dist, angle : real;
  fiproc, rproc : real;
  p1, z1, p10, z10, func, wc, A1, B1, phm, am, fix : real;
  s2, aslope, pslope, error : real;
  m, n, ni : integer;
  design : boolean;

begin
 pi := 3.14159;
 z := 0.707;
 sq := sqrt(1 - z * z);
 alfa := 0.1;
 n := 4;
 wo := 1.5;
 design := true;

 writeln('System with ', n : 4 : 1, ' lags having ratio ', alfa : 4 : 1);
 if design then
  writeln(' wo   k     ki    p1    z1    p10   z10   Ti    wTi ')
 else
  writeln(' wo   wc  phm   aslope pslope w180  am    rproc  angle');


{Main loop)}
 repeat
  w := wo * sq;
  s := wo * z;
  r := 1;
  rp := 1;
  fi := 0;
  fip := 0;
  w1 := w;
  s1 := s;
  m := 0;

  repeat
   r := r / sqrt((1 - s1) * (1 - s1) + w1 * w1);
   rp := rp / sqrt(1 + w1 * w1);
```

```
      rp := 1;
      fip := 0;
      w1 := wc;
      m := 0;
      repeat
       rp := rp / sqrt(1 + w1 * w1);
       fip := fip - arctan(w1);
       w1 := alfa * w1;
       m := m + 1;
      until m = n;
      wc := sqrt((k * k + ki * ki / wc / wc) * rp * rp * (1 + wc * wc) - 1);
    until ni = 20;
    fix := -pi / 2 + arctan(k * wc / ki);
    phm := 180 / pi * (pi / 2 + arctan(wc * k / ki) + fip);

{Amplitude and phase slopes at crossover}
    s2 := wc * k;
    aslope := 0;
    pslope := 0;
    w1 := wc;
    m := 0;
    repeat
     aslope := aslope - w1 * w1 / (1 + w1 * w1);
     pslope := pslope - w1 / (1 + w1 * w1);
     w1 := alfa * w1;
     m := m + 1;
    until m = n;
    pslope := pslope + ki * s2 / (ki * ki + s2 * s2);
    aslope := aslope - ki * ki / (s2 * s2 + ki * ki);

{Amplitude margin and w180}
    w1 := w;
    ni := 1;
    repeat
     rp := 1;
     fip := 0;
     m := 0;
     w2 := w1;
     repeat
      rp := rp / sqrt(1 + w2 * w2);
      fip := fip + arctan(w2);
      w2 := alfa * w2;
      m := m + 1;
     until m = n;
     error := pi / 2 + arctan(k * w1 / ki) - fip;
     w2 := pi / 2 + arctan(k * w1 / ki) + arctan(w1 * alfa) - fip;
```

```
      if s1 < 1 then
       fi := fi - arctan(w1 / (1 - s1))
      else
       fi := fi - (pi - arctan(w1 / (s1 - 1)));
      fip := fip - arctan(w1);
      s1 := alfa * s1;
      w1 := alfa * w1;
      m := m + 1;
    until m = n;
    A := r * cos(fi);
    B := r * sin(fi);
    k := -(w * A + s * B) / w / (A * A + B * B);
    ki := -(s * s + w * w) * B / w / (A * A + B * B);
    Ti := k / ki;

  {Process gain and phase at w}
    wTi := w * Ti;
    fiproc := fip;
    rproc := rp;
    angle := 180 * fiproc / pi;

  {Pole on real axis}
    p1 := ki / (1 + k);
    ni := 1;
    repeat
     m := 0;
     w1 := 1;
     func := 1;
     repeat
      func := func / (1 - p1 * w1);
      w1 := w1 * alfa;
      m := m + 1;
     until m = n;
     p1 := ki * func / (1 + k * func);
     ni := ni + 1;
    until ni = 20;

    z1 := 1 / Ti;
    p10 := p1 / wo;
    z10 := z1 / wo;

  {Crossover frequency and phase margin}
    wc := w;
    ni := 1;
    repeat
     ni := ni + 1;
```

```
      w1 := sin(w2) / cos(w2) / alfa;
      ni := ni + 1;
    until ni = 20;
    am := 1 / rp * sqrt(k * k + ki * ki / w1 * w1);

  {Write result}
    if design then
      writeln(wo : 4 : 1, k : 8 : 3, ki : 8 : 3, p1 : 8 : 3, z1 : 8 : 3, p10 : 8 : 3, z10 : 8 : 3, Ti : 8 : 3,
            wTi : 8 : 3)
    else
      writeln(wo : 4 : 1, wc : 6 : 1, phm : 6 : 1, aslope : 8 : 3, pslope : 8 : 3, w1 : 8 : 1, am : 8 : 1,
            rproc : 8 : 3, angle : 8 : 1);
    wo := wo + 0.50;

  until ki < 0


end.
```

Text

System with 4 lags having ratio 0.1

wo	k	ki	p1	z1	p10	z10	Ti	wTi
1.5	1.104	1.970	0.345	1.784	0.230	1.189	0.560	0.595
2.0	1.694	3.194	0.032	1.886	0.016	0.943	0.530	0.750
2.5	2.227	4.510	4.730	2.025	1.892	0.810	0.494	0.873
3.0	2.705	5.809	5.205	2.148	1.735	0.716	0.466	0.988
3.5	3.126	6.980	5.676	2.233	1.622	0.638	0.448	1.109
4.0	3.491	7.915	4.926	2.267	1.231	0.567	0.441	1.248
4.5	3.801	8.508	4.182	2.238	0.929	0.497	0.447	1.422
5.0	4.054	8.654	3.445	2.134	0.689	0.427	0.469	1.657
5.5	4.252	8.249	2.713	1.940	0.493	0.353	0.515	2.005
6.0	4.393	7.194	1.988	1.637	0.331	0.273	0.611	2.591
6.5	4.479	5.387	1.268	1.203	0.195	0.185	0.831	3.822
7.0	4.509	2.731	0.554	0.606	0.079	0.087	1.651	8.172
7.5	4.483	-0.870	-0.154	-0.194	-0.021	-0.026	-5.152	-27.326

Text

System with 4 lags having ratio 0.1

wo	wc	phm	aslope	pslope	w180	am	rproc	angle
1.5	1.6	70.3	-1.300	-0.128	29.0	200.5	0.682	-53.4
2.0	2.0	60.6	-1.308	-0.115	28.6	323.3	0.572	-63.7
2.5	2.6	57.0	-1.315	-0.121	28.2	445.0	0.485	-71.6
3.0	3.0	54.2	-1.322	-0.136	28.0	558.4	0.417	-78.1
3.5	3.4	52.2	-1.327	-0.154	27.9	658.9	0.363	-83.5
4.0	3.7	51.0	-1.327	-0.173	27.8	741.5	0.321	-88.1
4.5	3.9	50.7	-1.319	-0.193	27.9	800.6	0.286	-92.2
5.0	4.1	51.3	-1.303	-0.217	28.1	828.5	0.256	-95.9
5.5	4.2	53.2	-1.274	-0.248	28.7	808.2	0.232	-99.3
6.0	4.2	56.6	-1.231	-0.292	30.9	672.3	0.211	-102.4
6.5	4.2	62.1	-1.173	-0.362	54.8	199.0	0.193	-105.3
7.0	4.1	70.5	-1.110	-0.481	-83.5	121.8	0.177	-108.0
7.5	4.0	-97.9	-1.086	-0.673	-11.7	3314.2	0.163	-110.6