

Chapter 10

Tuning of PID controllers

10.1 Introduction

This chapter describes two methods for calculating proper values of the PID parameters K_p , T_i and T_d , i.e. controller tuning. These two methods are:

- **The Good Gain method** [3] which is a simple experimental method which can be used without any knowledge about the process to be controlled. (Of course, if you have a process model, you can use the Good Gain method on a simulator in stead of on the physical process.)
- **Skogestad's method** [7] which is a model-based method. It is assumed that you have mathematical model of the process (a transfer function model). It does not matter how you have derived the transfer function – it can stem from a model derived from physical principles (as described in Ch. 3), or from calculation of model parameters (e.g. gain, time-constant and time-delay) from an experimental response, typically a step response experiment with the process (step on the process input).

There is a large number of tuning methods [1], but it is my view that the above methods will cover most practical cases. What about the famous Ziegler-Nichols' methods – the Ultimate Gain method (or Closed-Loop method) and the Process Reaction curve method (the Open-Loop method)?[8] The Good Gain method has actually many similarities with the Ultimate Gain method, but the latter method has one serious

drawback, namely it requires the control loop to be brought to the limit of stability during the tuning, while the Good Gain method requires a stable loop during the tuning. The Ziegler-Nichols' Open-Loop method is similar to a special case of Skogestad's method, and Skogestad's method is more applicable. So, I have decided not to include the Ziegler-Nichols' methods in this book.¹

10.2 The Good Gain method

Before going into the procedure of controller tuning with the Good Gain method [3], let's look at what is the aim of the controller tuning. If possible, we would like to obtain both of the following for the control system:

- Fast responses, and
- Good stability

Unfortunately, for most practical processes being controlled with a PID controller, these two wishes can not be achieved simultaneously. In other words:

- The faster responses, the worse stability, and
- The better stability, the slower responses.

For a control system, it is more important that it has good stability than being fast. So, we specify:

Acceptable stability (good stability, but not too good as it gives too slow responses)

Figure 10.1 illustrates the above. It shows the response in the process output variable due to a step change of the setpoint. (The responses correspond to three different controller gains in a simulated control system.)

¹However, both Ziegler-Nichols' methods are described in articles available at <http://techteach.no>.

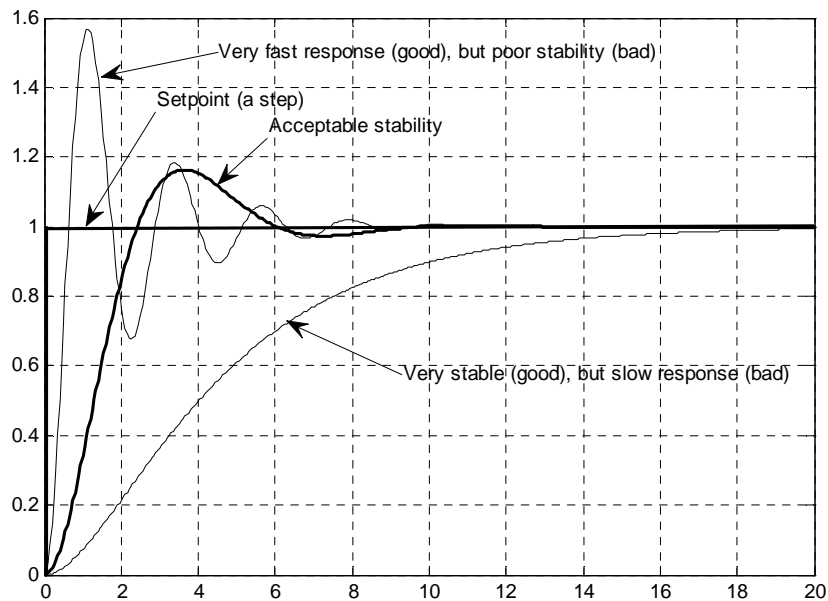


Figure 10.1: In controller tuning we want to obtain acceptable stability of the control system.

What is “acceptable stability” more specifically? There exists no single definition. One simple yet useful definition is as follows. Assume a positive step change of the setpoint. *Acceptable stability is when the undershoot that follows the first overshoot of the response is small, or barely observable.* See Figure 10.1. (If the step change is negative, the terms undershoot and overshoot are interchanged, of course.)

As an alternative to observing the response after a step change of the setpoint, you can regard the response after a step change of the process disturbance. The definition of acceptable stability is the same as for the setpoint change, i.e. that the undershoot (or overshoot – depending on the sign of the disturbance step change) that follows the first overshoot (or undershoot) is small, or barely observable.

The Good Gain method aims at obtaining acceptable stability as explained above. It is a simple method which has proven to give good results on laboratory processes and on simulators. The method is based on experiments on a real or simulated control system, see Figure 10.2. The procedure described below assumes a PI controller, which is the most commonly used controller function. However, a comment about how to

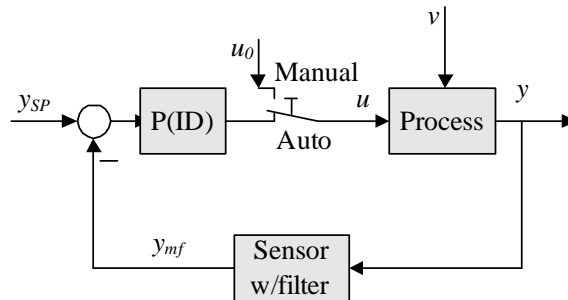


Figure 10.2: The Good Gain method for PID tuning is applied to the established control system.

include the D-term, so that the controller becomes a PID controller, is also given.

1. Bring the process to or close to the normal or specified operation point by adjusting the nominal control signal u_0 (with the controller in manual mode).
2. Ensure that the controller is a P controller with $K_p = 0$ (set $T_i = \infty$ and $T_d = 0$). Increase K_p until the control loop gets good (satisfactory) stability as seen in the response in the measurement signal after e.g. a step in the setpoint or in the disturbance (exciting with a step in the disturbance may be impossible on a real system, but it is possible in a simulator). If you do not want to start with $K_p = 0$, you can try $K_p = 1$ (which is a good initial guess in many cases) and then increase or decrease the K_p value until you observe some overshoot and a barely observable undershoot (or vice versa if you apply a setpoint step change the opposite way, i.e. a negative step change), see Figure 10.3. This kind of response is assumed to represent good stability of the control system. This gain value is denoted K_{pGG} .

It is important that *the control signal is not driven to any saturation limit* (maximum or minimum value) during the experiment. If such limits are reached the K_p value may not be a good one – probably too large to provide good stability when the control system is in normal operation. So, you should apply a relatively small step change of the setpoint (e.g. 5% of the setpoint range), but not so small that the response drowns in noise.

3. Set the integral time T_i equal to

$$\underline{T_i = 1.5T_{ou}} \quad (10.1)$$

where T_{ou} is the time between the overshoot and the undershoot of the step response (a step in the setpoint) with the P controller, see Figure 10.3.² Note that for most systems (those which does not contain a pure integrator) there will be offset from setpoint because the controller during the tuning is just a P controller.

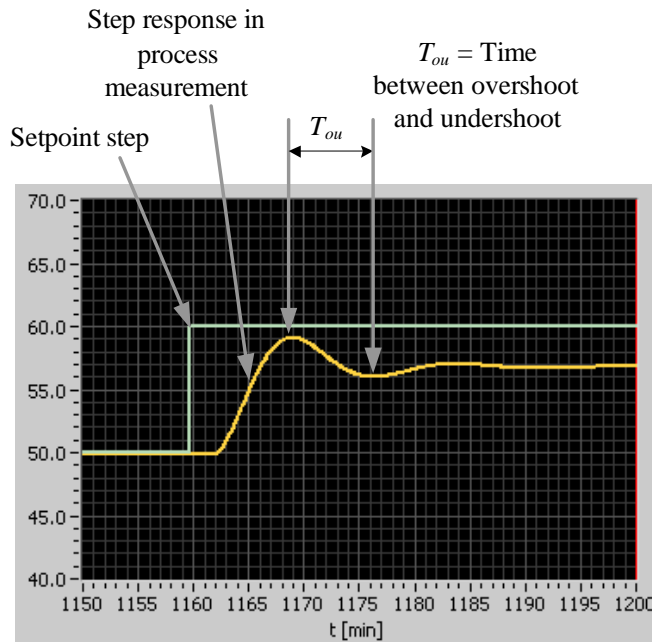


Figure 10.3: The Good Gain method: Reading off the time between the overshoot and the undershoot of the step response with P controller

4. Because of the introduction of the I-term, the loop with the PI controller in action will probably have somewhat reduced stability than with the P controller only. To compensate for this, the K_p can be reduced somewhat, e.g. to 80% of the original value. Hence,

$$\underline{K_p = 0.8K_{pGG}} \quad (10.2)$$

5. If you want to include the D-term, so that the controller becomes a PID controller³, you can try setting T_d as follows:

$$T_d = \frac{T_i}{4} \quad (10.3)$$

²Alternatively, you may apply a negative setpoint step, giving a similar response but downwards. In this case T_{ou} is time between the undershoot and the overshoot.

³But remember the drawbacks about the D-term, namely that it amplifies the measurement noise, causing a more noisy controller signal than with a PI controller.

which is the T_d - T_i relation that was used by Ziegler and Nichols [8].

6. You should check the stability of the control system with the above controller settings by applying a step change of the setpoint. If the stability is poor, try reducing the controller gain somewhat, possibly in combination with increasing the integral time.

Example 10.1 *PI controller tuning of a wood-chip level control system with the Good Gain Method*

I have used the Good Gain method to tune a PI controller on a simulator of the level control system for the wood-chip tank, cf. Figure 1.2. During the tuning I found

$$K_{pGG} = 1.5 \quad (10.4)$$

and

$$T_{ou} = 12 \text{ min} \quad (10.5)$$

The PI parameter values are

$$K_p = 0.8K_{pGG} = 0.8 \cdot 1.5 = 1.2 \quad (10.6)$$

$$T_i = 1.5T_{ou} = 1.5 \cdot 12 \text{ min} = 18 \text{ min} = 1080 \text{ s} \quad (10.7)$$

Figure 10.4 shows the resulting responses with a setpoint step at time 20 min and a disturbance step (outflow step from 1500 to 1800 kg/min) at time 120 min. The control system has good stability.

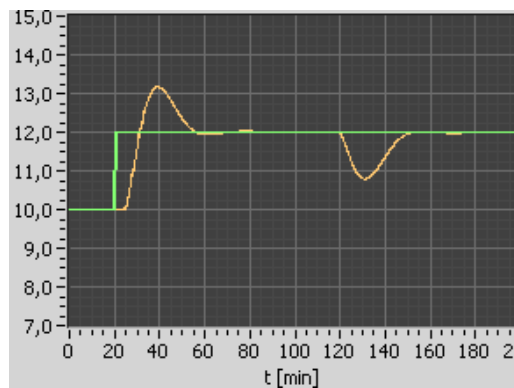


Figure 10.4: Example 10.1: Level control of the wood-chip tank with a PI controller.

[End of Example 10.1]

10.3 Skogestad's PID tuning method

10.3.1 The background of Skogestad's method

Skogestad's PID tuning method [7]⁴ is a model-based tuning method where *the controller parameters are expressed as functions of the process model parameters*. It is assumed that the control system has a transfer function block diagram as shown in Figure 10.5.

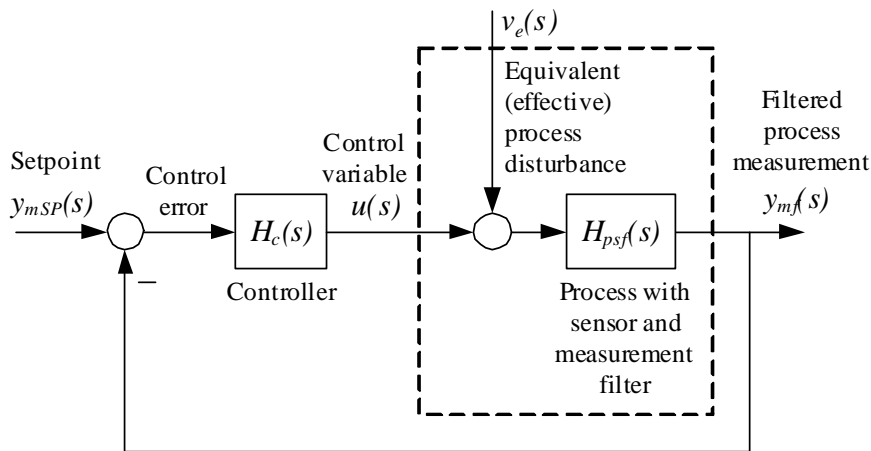


Figure 10.5: Block diagram of the control system in PID tuning with Skogestad's method

Comments to this block diagram:

- The transfer function $H_{psf}(s)$ is a combined transfer function of the process, the sensor, and the measurement lowpass filter. Thus, $H_{psf}(s)$ represents all the dynamics that the controller “feels”. For simplicity we may denote this transfer function the “process transfer function”, although it is a combined transfer function.
- The process transfer function can stem from a simple step-response experiment with the process. This is explained in Sec. 10.3.3.
- The block diagram shows a disturbance acting on the process. Information about this disturbance is not used in the tuning, but if you are going to test the tuning on a simulator to see how the control system compensates for a process disturbance, you should add a

⁴Named after the originator Prof. Sigurd Skogestad

disturbance at the point indicated in the block diagram, which is at the process input. It turns out that in most processes the dominating disturbance influences the process dynamically at the “same” point as the control variable. Such a disturbance is called an *input disturbance*. Here are a few examples:

- Liquid tank: The control variable controls the inflow. The outflow is a disturbance.
- Motor: The control variable controls the motor torque. The load torque is a disturbance.
- Thermal process: The control variable controls the power supply via a heating element. The power loss via heat transfer through the walls and heat outflow through the outlet are disturbances.

The design principle of Skogestad’s method is as follows. The control system *tracking transfer function* $T(s)$, which is the transfer function from the setpoint to the (filtered) process measurement, is *specified* as a first order transfer function with time delay:

$$T(s) = \frac{y_{mf}(s)}{y_{mSP}(s)} = \frac{1}{T_C s + 1} e^{-\tau s} \quad (10.8)$$

where T_C is the time-constant of the control system which *the user must specify*, and τ is the process time delay which is *given* by the process model (the method can however be used for processes without time delay, too). Figure 10.6 shows as an illustration the response in y_{mf} after a step in the setpoint y_{mSP} for (10.8).

From the block diagram shown in Figure 10.5 the tracking transfer function is, cf. the Feedback rule in Figure 5.2,

$$T(s) = \frac{H_c(s)H_{psf}(s)}{1 + H_c(s)H_{psf}(s)} \quad (10.9)$$

Setting (10.9) equal to (10.8) gives

$$\frac{H_c(s)H_{psf}(s)}{1 + H_c(s)H_{psf}(s)} = \frac{1}{T_C s + 1} e^{-\tau s} \quad (10.10)$$

Here, the only unknown is the controller transfer function, $H_c(s)$. By making some proper simplifying approximations to the time delay term, the controller becomes a PID controller or a PI controller for the process transfer function assumed.

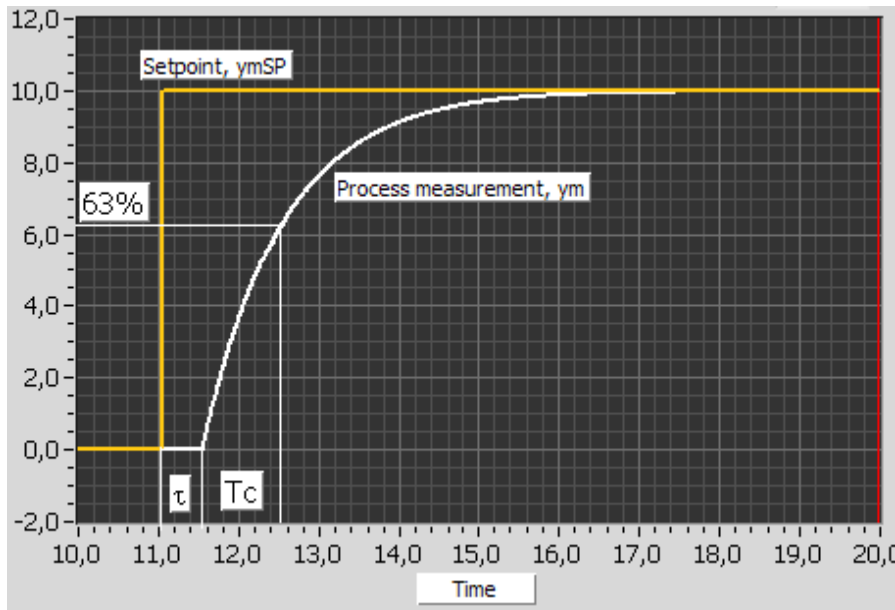


Figure 10.6: Step response of the specified tracking transfer function (10.8) in Skogestad's PID tuning method

10.3.2 The tuning formulas in Skogestad's method

Skogestad's tuning formulas for a number of processes are shown in Table 10.1.⁵

Process type	$H_{psf}(s)$ (process)	K_p	T_i	T_d
Integrator + delay	$\frac{K}{s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	0
Time-constant + delay	$\frac{K}{Ts+1} e^{-\tau s}$	$\frac{T}{K(T_C + \tau)}$	$\min [T, c(T_C + \tau)]$	0
Integr + time-const + del.	$\frac{K}{(Ts+1)s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	T
Two time-const + delay	$\frac{K}{(T_1s+1)(T_2s+1)} e^{-\tau s}$	$\frac{T_1}{K(T_C + \tau)}$	$\min [T_1, c(T_C + \tau)]$	T_2
Double integrator + delay	$\frac{K}{s^2} e^{-\tau s}$	$\frac{1}{4K(T_C + \tau)^2}$	$4(T_C + \tau)$	$4(T_C + \tau)$

Table 10.1: Skogestad's formulas for PI(D) tuning.

For the "Two time-constant + delay" process in Table 10.1 T_1 is the largest and T_2 is the smallest time-constant.⁶

Originally, Skogestad defined the factor c in Table 10.1 as 4. This gives

⁵In the table, "min" means the minimum value (of the two alternative values).

⁶[7] also describes methods for model reduction so that more complicated models can be approximated with one of the models shown in Table 10.1.

good setpoint tracking. But the disturbance compensation may become quite sluggish. To obtain faster disturbance compensation, I suggest [3]

$$c = 2 \quad (10.11)$$

The drawback of such a reduction of c is that there will be somewhat more overshoot in the setpoint step respons, and that the stability of the control loop will be somewhat reduced. Also, the robustness against changes of process parameters (e.g. increase of process gain and increase of process time-delay) will be somewhat reduced.

Skogestad suggests using

$$T_C = \tau \quad (10.12)$$

for T_C in Table 10.1 – unless you have reasons for a different specification of T_C .

Example 10.2 Control of first order system with time delay

Let us try Skogestad's method for tuning a PI controller for the (combined) process transfer function

$$H_{psf}(s) = \frac{K}{Ts + 1} e^{-\tau s} \quad (10.13)$$

(time-constant with time-delay) where

$$K = 1; T = 1 \text{ s}; \tau = 0.5 \text{ s} \quad (10.14)$$

We use (10.12):

$$T_C = \tau = 0.5 \text{ s} \quad (10.15)$$

The controller parameters are as follows, cf. Table 10.1:

$$K_p = \frac{T}{K(T_C + \tau)} = \frac{1}{1 \cdot (0.5 + 0.5)} = 1 \quad (10.16)$$

$$T_i = \min[T, c(T_C + \tau)] \quad (10.17)$$

$$= \min[1, 2(0.5 + 0.5)] \quad (10.18)$$

$$= \min[1, 2] \quad (10.19)$$

$$= 1 \text{ s} \quad (10.20)$$

$$T_d = 0 \quad (10.21)$$

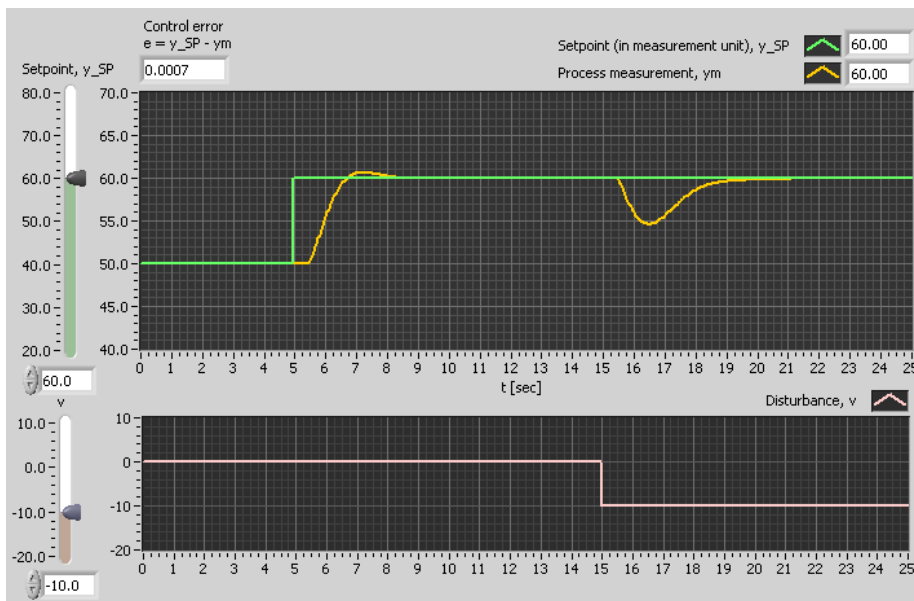


Figure 10.7: Example 10.2: Simulated responses in the control system with Skogestad's controller tuning

Figure 10.7 shows control system responses with the above PID settings. At time 5 sec the setpoint is changed as a step, and at time 15 sec the disturbance is changed as a step. The responses, and in particular the stability of the control systems, seem ok.

[End of Example 10.2]

You may wonder: Given a process model as in Table 10.1. Does Skogestad's method give better control than if the controller was tuned with some other method, e.g. the Good Gain method? There is no unique answer to that question, but my impression is that Skogestad's method in general works fine. If you have a mathematical model of the process to be controlled, you should always simulate the system with alternative controller tunings. The benefit of Skogestad's method is that you do not have to perform trial-and-error simulations to tune the controller. The parameters come directly from the process model and the specified control system response time. Still, you should run simulations to check the performance.

10.3.3 How to find model parameters from experiments

The values of the parameters of the transfer functions in Table 10.1 can be found from a mathematical model based on physical principles, cf. Chapter 3. The parameter values can also be found from a step-response experiment with the process. This is shown for the model *Integrator with time-delay* and *Time-constant with time-delay* in the following respective figures. (The theory of calculating these responses is covered by Chapter 6.)

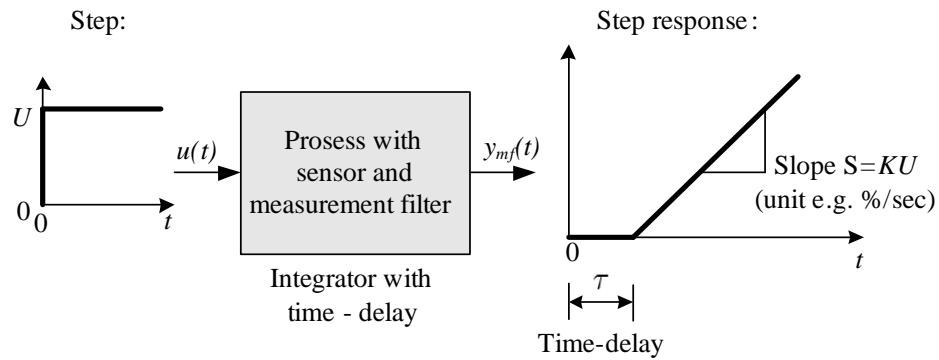


Figure 10.8: How the transfer function parameters K and τ appear in the step response of an *Integrator with time-delay* process

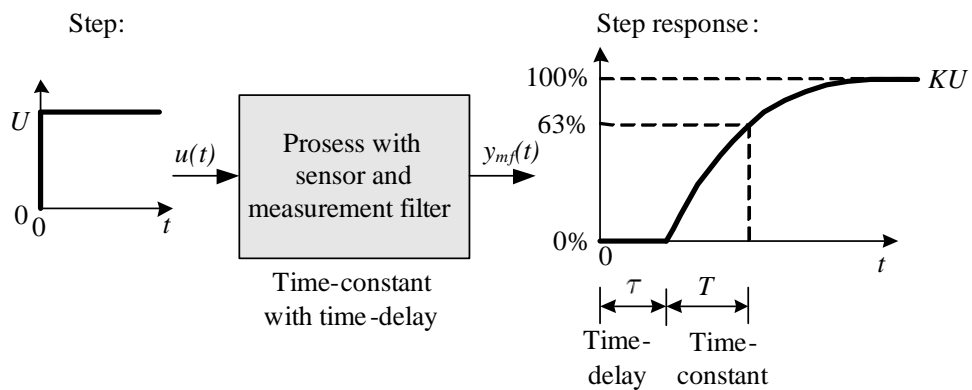


Figure 10.9: How the transfer function parameters K , T , and τ appear in the step response of a *Time-constant with time-delay* process

10.3.4 Transformation from serial to parallel PID settings

Skogestad's formulas assumes a *serial* PID controller function (alternatively denoted cascade PID controller) which has the following transfer function:

$$u(s) = K_{ps} \frac{(T_{is}s + 1)(T_{ds}s + 1)}{T_{is}s} e(s) \quad (10.22)$$

where K_{ps} , T_{is} , and T_{ds} are the controller parameters. If your controller actually implements a *parallel* PID controller (as in the PID controllers in LabVIEW PID Control Toolkit and in the Matlab/Simulink PID controllers), which has the following transfer function:

$$u(s) = \left[K_{pp} + \frac{K_{ip}}{T_{ip}s} + K_{pd}s \right] e(s) \quad (10.23)$$

then you should transform from serial PID settings to parallel PID settings. If you do not implement these transformations, the control system may behave unnecessarily different from the specified response. The serial-to-parallel transformations are as follows:

$$K_{pp} = K_{ps} \left(1 + \frac{T_{ds}}{T_{is}} \right) \quad (10.24)$$

$$T_{ip} = T_{is} \left(1 + \frac{T_{ds}}{T_{is}} \right) \quad (10.25)$$

$$T_{dp} = T_{ds} \frac{1}{1 + \frac{T_{ds}}{T_{is}}} \quad (10.26)$$

Note: The parallel and serial PI controllers are identical (since $T_d = 0$ in a PI controller). Therefore, the above transformations are not relevant for PI controller, only for PID controllers.

10.3.5 When the process has no time-delay

What if the process $H_p(s)$ is *without time-delay*? Then you can not specify T_C according to (10.12) since that would give $T_C = 0$ (zero response time of the control system). You must specify T_C to some reasonable value larger than zero. If you do not know what could be a reasonable value, you can simulate the control system for various values of T_C . If the control signal (controller output signal) is changing too quickly, or often reaches the maximum and minimum values for reasonable changes of the setpoint

or the disturbance, the controller is too aggressive, and you can try increasing T_C . If you don't want to simulate, then just try setting $T_C = T/2$ where T is the dominating (largest) time-constant of the process (assuming the process is a time-constant system, of course).

For the double integrator (without time-delay) I have seen in simulations that the actual response-time (or 63% rise-time) of the closed-loop system may be about twice the specified time-constant T_C . Consequently, you can set T_C to about half of the response-time you actually want to obtain.

10.4 Auto-tuning

Auto-tuning is automatic tuning of controller parameters in one experiment. It is common that commercial controllers offers auto-tuning. The operator starts the auto-tuning via some button or menu choice on the controller. The controller then executes automatically a pre-planned experiment on the uncontrolled process or on the control system depending on the auto-tuning method implemented. Below are described a couple of auto-tuning methods.

Auto-tuning based on relay tuning

The relay method for tuning PID controllers is used as the basis of auto-tuning in some commercial controllers.⁷ The principle of this method is as follows: When the auto-tuning phase is started, a relay controller is used as the controller in the control loop, see Figure 10.10. The relay controller is simply an On/Off controller. It sets the control signal to a high (On) value when the control error is positive, and to a low (Off) value when the control error is negative. This controller creates automatically sustained oscillations in control loop, and from the amplitude and the period of these oscillations proper PID controller parameters are calculated by an algorithm in the controller. Only a few periods are needed for the autotuner to have enough information to accomplish the tuning. The autotuner activates the tuned PID controller automatically after the tuning has finished.

⁷For example the ABB ECA600 PID controller and the Fuji PGX PID controller.

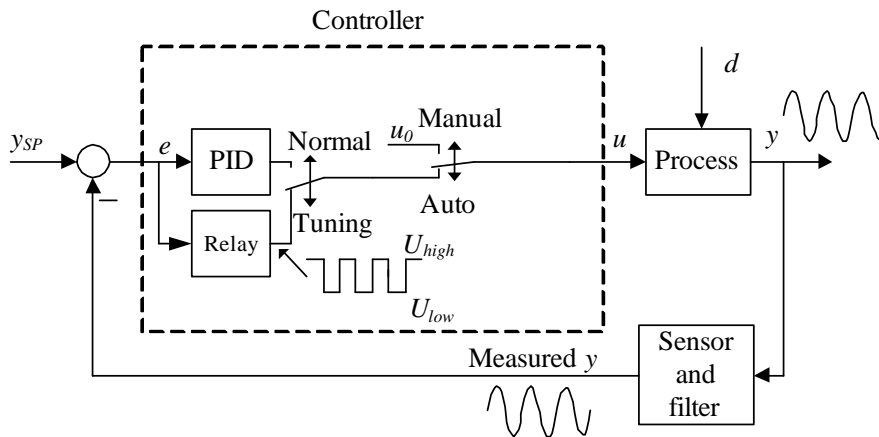


Figure 10.10: Relay control (On/Off control) used in autotuning

Model-based auto-tuning

Commercial software tools⁸ exist for auto-tuning based on an estimated process model developed from a sequence of logged data – or time-series – of the control variable u and process measurement y_m . The process model is a “black-box” input-output model in the form of a transfer function. The controller parameters are calculated automatically on the basis of the estimated process model. The time-series of u and y_m may be logged from the system being in closed loop or in open loop:

- **Closed loop**, with the control system being excited via the setpoint, y_{SP} , see Figure 10.11. The closed loop experiment may be used when the controller should be re-tuned, that is, the parameters should be optimized.
- **Open loop**, with the process, which in this case is *not* under control, being excited via the control variable, u , see Figure 10.12. This option must be made if there are no initial values of the controller parameters.

⁸E.g. MultiTune (Norwegian) and ExperTune

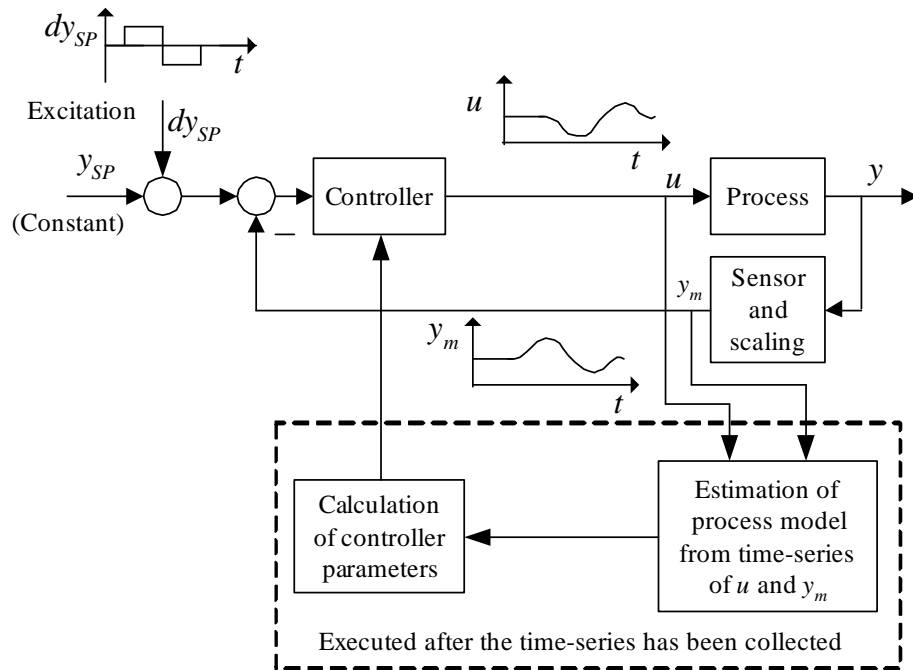


Figure 10.11: Auto-tuning based on closed loop excitation via the setpoint

10.5 PID tuning when process dynamics varies

10.5.1 Introduction

A well tuned PID controller has parameters which are adapted to the dynamic properties to the process, so that the control system becomes fast and stable. If the process dynamic properties varies without re-tuning the controller, the control system

- gets *reduced stability*, or
- becomes *more sluggish*.

Problems with variable process dynamics can be solved in the following alternative ways:

- **The controller is tuned in the most critical operation point**, so that when the process operates in a different operation point, the

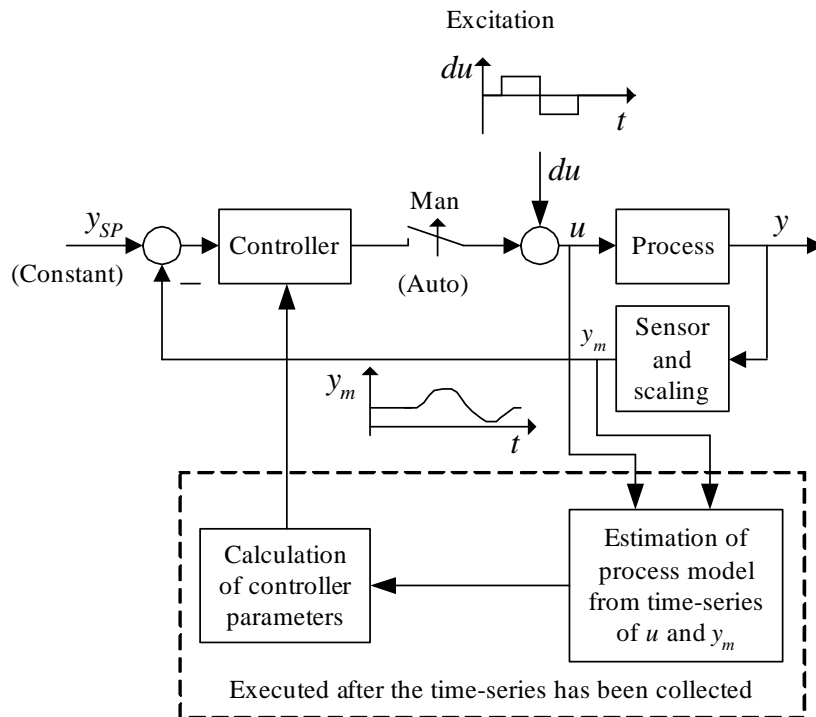


Figure 10.12: Auto-tuning based on open loop excitation via the control variable

stability of the control system is just better — at least the stability is not reduced. However, if the stability is too good the tracking quickness is reduced, giving more sluggish control.

- **The controller parameters are varied in the “opposite” direction of the variations of the process dynamics**, so that the performance of the control system is maintained, independent of the operation point. Some ways to vary the controller parameters are:
 - *Model-based PID parameter adjustment*, cf. Section 10.5.2.
 - *PID controller with gain scheduling*, cf. Section 10.5.3.
 - *Model-based adaptive controller*, cf. Section 10.5.4.

Commercial control equipment is available with options for gain scheduling and/or adaptive control.

10.5.2 PID parameter adjustment with Skogestad's method

Assume that you have tuned a PID or a PI controller for some process that has a transfer function equal to one of the transfer functions $H_{psf}(s)$ shown in Table 10.1. Assume then that some of the parameters of the process transfer function changes. How should the controller parameters be adjusted? The answer is given by Table 10.1 because it gives the controller parameters as functions of the process parameters. You can actually use this table as the basis for adjusting the PID parameters even if you used some other method than Skogestad's method for the initial tuning.

From Table 10.1 we can draw a number of general rules for adjusting the PID parameters:

Example 10.3 *Adjustment of PI controller parameters for integrator with time delay process*

Assume that the process transfer function is

$$H_{psf}(s) = \frac{K}{s} e^{-\tau s} \quad (10.27)$$

(integrator with time delay). According to Table 10.1, with the suggested specification $T_C = \tau$, the PI controller parameters are

$$K_p = \frac{1}{2K\tau} \quad (10.28)$$

$$T_i = k_1 2\tau \quad (10.29)$$

As an example, assume that the process gain K is increased to, say, twice its original value. How should the PI parameters be adjusted to maintain good behaviour of the control system? From (10.28) we see that K_p should be halved, and from (10.29) we see that T_i should remain unchanged.

As another example, assume that the process time delay τ is increased to, say, twice its original value. From (10.28) we see that K_p should be halved, and from (10.29) we see that T_i should get a doubled value. One concrete example of such a process change is the wood-chip tank. If the speed of the conveyor belt is halved, the time delay (transport delay) is doubled. And now you know how to quickly adjust the PI controller parameters if such a change of the conveyor belt speed should occur.⁹

[End of Example 10.3]

⁹What may happen if you do not adjust the controller parameters? The control system may get poor stability, or it may even become unstable.

10.5.3 Gain scheduling of PID parameters

Figure 10.13 shows the structure of a control system for a process which may have varying dynamic properties, for example a varying gain. The

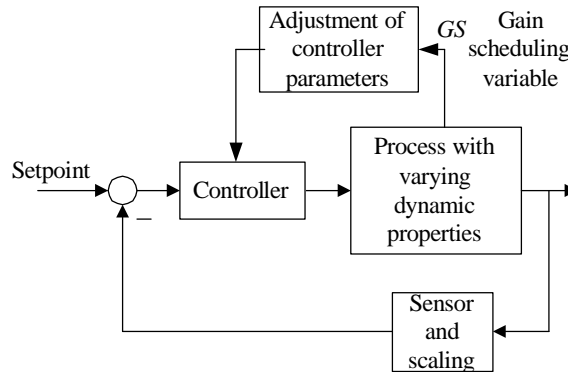


Figure 10.13: Control system for a process having varying dynamic properties. The GS variable expresses or represents the dynamic properties of the process.

Gain scheduling variable GS is some measured process variable which at every instant of time expresses or represents the dynamic properties of the process. As you will see in Example 10.4, GS may be the mass flow through a liquid tank.

Assume that proper values of the PID parameters K_p , T_i and T_d are found (using e.g. the Good Gain method) for a set of values of the GS variable. These PID parameter values can be stored in a parameter table – the gain schedule – as shown in Table 10.2. From this table proper PID parameters are given as functions of the gain scheduling variable, GS .

GS	K_p	T_i	T_d
GS_1	K_{p1}	T_{i1}	T_{d1}
GS_2	K_{p2}	T_{i2}	T_{d2}
GS_3	K_{p3}	T_{i3}	T_{d3}

Table 10.2: Gain schedule or parameter table of PID controller parameters.

There are several ways to express the PID parameters as functions of the GS variable:

- **Piecewise constant:** An interval is defined around each GS value in the parameter table. The controller parameters are kept constant as long as the GS value is within the interval. This is a simple

solution, but it seems nonetheless to be the most common solution in commercial controllers.

When the GS variable changes from one interval to another, the controller parameters are changed abruptly, see Figure 10.14 which illustrates this for K_p , but the situation is the same for T_i and T_d . In Figure 10.14 it is assumed that GS values toward the left are critical with respect to the stability of the control system. In other words: It is assumed that it is safe to keep K_p constant and equal to the K_p value in the left part of the interval.

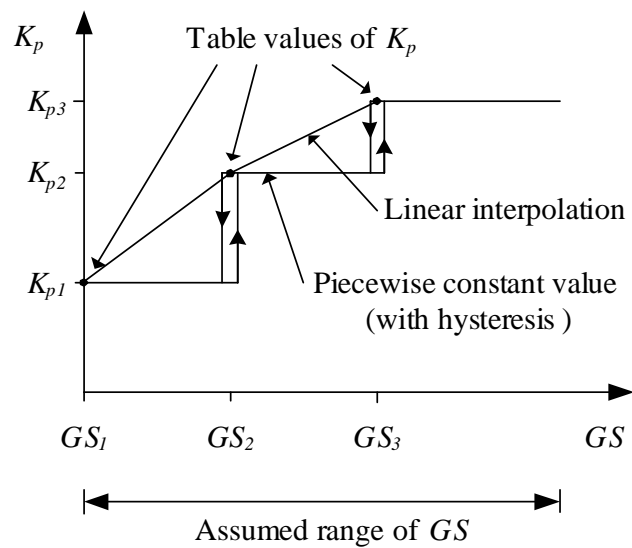


Figure 10.14: Two different ways to interpolate in a PID parameter table: Using piecewise constant values and linear interpolation

Using this solution there will be a disturbance in the form of a step in the control variable when the GS variable shifts from one interval to another, but this disturbance is probably of negligible practical importance for the process output variable. Noise in the GS variable may cause frequent changes of the PID parameters. This can be prevented by using a hysteresis, as shown in Figure 10.14.

- **Piecewise linear**, which means that a linear function is found relating the controller parameter (output variable) and the GS variable (input variable) between to adjacent sets of data in the table. The linear function is on the form

$$K_p = a \cdot GS + b \quad (10.30)$$

where a and b are found from the two corresponding data sets:

$$K_{p_1} = a \cdot GS_1 + b \quad (10.31)$$

$$K_{p_2} = a \cdot GS_2 + b \quad (10.32)$$

(Similar equations applies to the T_i parameter and the T_d parameter.) (10.31) and (10.32) constitute a set of two equations with two unknown variables, a and b (the solution is left to you).¹⁰

- **Other interpolations** may be used, too, for example a polynomial function fitted exactly to the data or fitted using the least squares method.

Example 10.4 *PID temperature control with gain scheduling during variable mass flow*

Figure 10.16 shows the front panel of a simulator for a temperature control system for a liquid tank with variable mass flow, w , through the tank. The control variable u controls the power to heating element. The temperature T is measured by a sensor which is placed some distance away from the heating element. There is a time delay from the control variable to measurement due to imperfect blending in the tank.

The process dynamics

We will initially, both in simulations and from analytical expressions, that the dynamic properties of the process *varies with the mass flow w* . The response in the temperature T after a step change in the control signal (which is proportional to the supplied power) is simulated for a large mass flow and a small mass flow. (Feedback temperature control is not active, thus open loop responses are shown.) The responses are shown in Figure 10.15. The simulations show that the following happens when the mass flow w is reduced (from 24 to 12 kg/min): *The gain process K is larger*. It can be shown that in addition, the time-constant T_t is larger, and the time delay τ is larger. (These terms assumes that system is a first order system with time delay. The simulator is based on such a model. The model is described below.)

Let us see if the way the process dynamics seems to depend on the mass flow w as seen from the simulations, can be confirmed from a

¹⁰Note that both MATLAB/SIMULINK and LabVIEW have functions that implement linear interpolation between tabular data. Therefore gain scheduling can be easily implemented in these environments.

Responses in temperature T [°C] after step amplitude of 10% in control signal, u

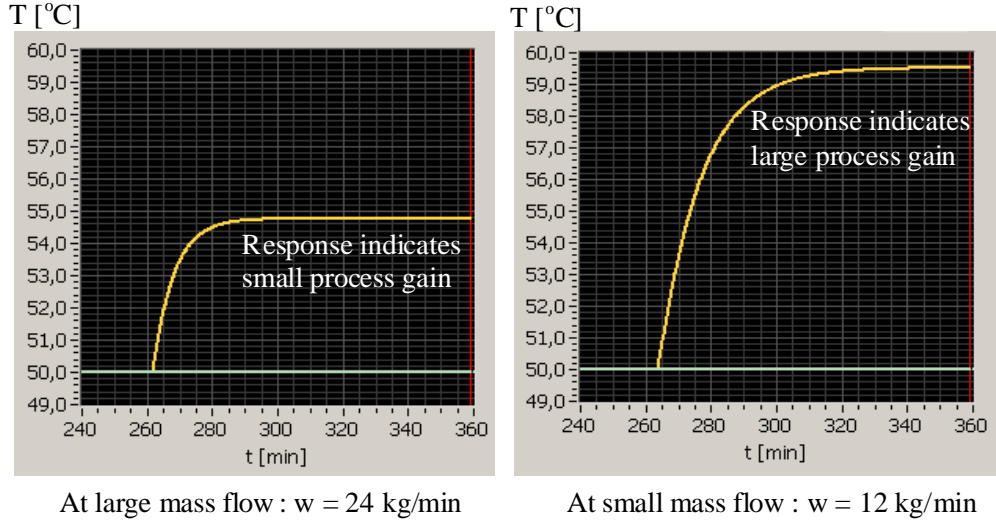


Figure 10.15: Responses in temperature T after a step in u of amplitude 10% at large mass flow and small mass flow

mathematical process model.¹¹ Assuming perfect stirring in the tank to have homogeneous conditions in the tank, we can set up the following energy balance for the liquid in the tank:

$$c\rho VT_1\dot{T}_1(t) = K_P u(t) + cw [T_{in}(t) - T_t(t)] \quad (10.33)$$

where T_1 [K] is the liquid temperature in the tank, T_{in} [K] is the inlet temperature, c [J/(kg K)] is the specific heat capacity, V [m³] is the liquid volume, ρ [kg/m³] is the density, w [kg/s] is the mass flow (same out as in), K_P [W/%] is the gain of the power amplifier, u [%] is the control variable, $c\rho VT_1$ is (the temperature dependent) energy in the tank. It is assumed that the tank is isolated, that is, there is no heat transfer through the walls to the environment. To make the model a little more realistic, we will include a time delay τ [s] to represent inhomogeneous conditions in the tank. Let us for simplicity assume that the time delay is inversely proportional to the mass flow. Thus, the temperature T at the sensor is

$$T(t) = T_1 \left(t - \frac{K_\tau}{w} \right) = T_1 (t - \tau) \quad (10.34)$$

where τ is the time delay and K_τ is a constant. It can be shown that the

¹¹Well, it would be strange if not. After all, we will be analyzing the same model as used in the simulator.

transfer function from control signal u to process variable T is

$$T(s) = \frac{K}{\underbrace{T_t s + 1}_{H_u(s)}} e^{-\tau s} u(s) \quad (10.35)$$

where

$$\text{Gain } K = \frac{K_P}{cw} \quad (10.36)$$

$$\text{Time-constant } T_t = \frac{\rho V}{w} \quad (10.37)$$

$$\text{Time delay } \tau = \frac{K_\tau}{w} \quad (10.38)$$

This confirms the observations in the simulations shown in Figure 10.15: Reduced mass flow w implies *larger process gain*, and larger time-constant and larger time delay.

Heat exchangers and blending tanks in a process line where the production rate or mass flow varies, have similar dynamic properties as the tank in this example.

Control without gain scheduling (with fixed parameters)

Let us look at temperature control of the tank. The mass flow w varies. In which operating point should the controller be tuned if we want to be sure that the stability of the control system is not reduced when w varies? In general the stability of a control loop is reduced if the gain increases and/or if the time delay of the loop increases. (10.36) and (10.38) show how the gain and time delay depends on the mass flow w . According to (10.36) and (10.38) the PID controller should be tuned at minimal w . If we do the opposite, that is, tune the controller at the maximum w , the control system may actually become unstable if w decreases.

Let us see if a simulation confirms the above analysis. Figure 10.16 shows a temperature control system. The PID controller is in the example tuned at the maximum w value, which here is assumed 24 kg/min.¹² The PID parameters are

$$K_p = 7.8; T_i = 3.8 \text{ min}; T_d = 0.9 \text{ min} \quad (10.39)$$

¹² Actually, the controller was tuned with the Ziegler-Nichols' Ultimate Gain method. This method is however not described in this book. The Good Gain method could have been used in stead.

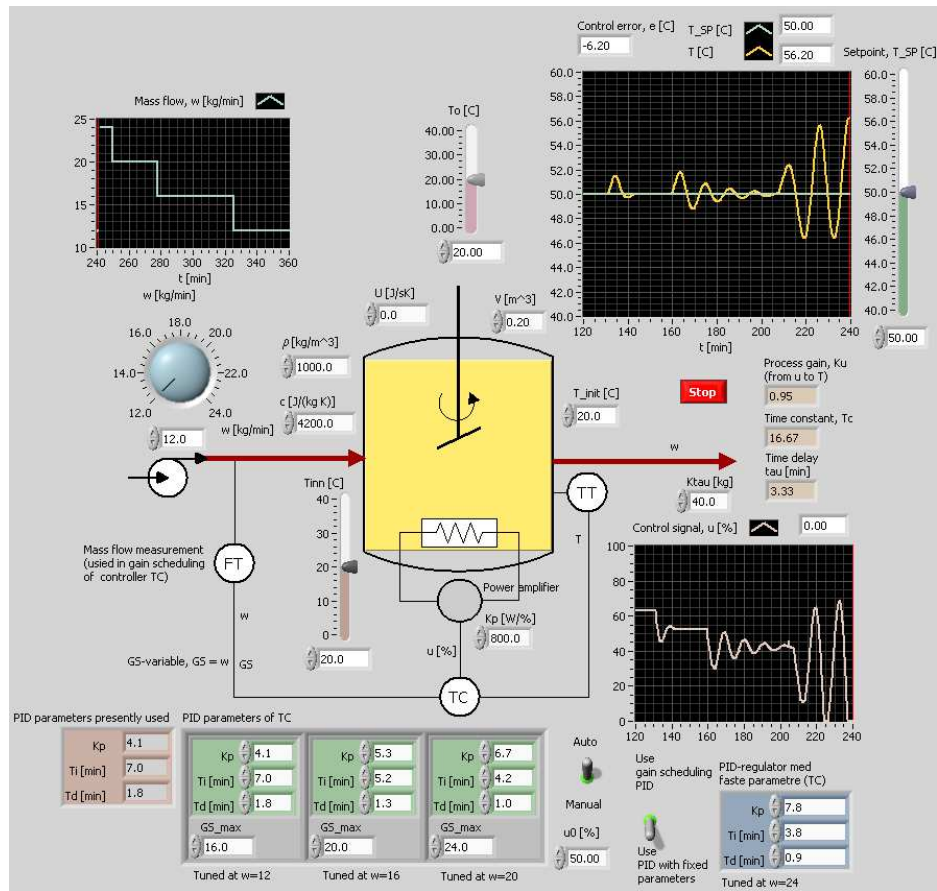


Figure 10.16: Example 10.4: Simulation of temperature control system with PID controller with fixed parameters tuned at maximum mass flow, which is $w = 24\text{kg/min}$

Figure 10.16 shows what happens at a stepwise reduction of w : The stability becomes worse, and the control system becomes *unstable* at the minimal w value, which is 12kg/min .

Instead of using the PID parameters tuned at maximum w value, we can tune the PID controller at minimum w value, which is 12 kg/min . The parameters are then

$$K_p = 4.1; T_i = 7.0\text{ min}; T_d = 1.8\text{ min} \quad (10.40)$$

The control system will now be stable for all w values, but the system behaves sluggish at large w values. (Responses for this case is however not shown here.)

Control with gain scheduling

Let us see if gain scheduling maintains the stability for varying mass flow w . The PID parameters will be adjusted as a function of a measurement of w since the process dynamics varies with w . Thus, w is the gain scheduling variable, GS :

$$GS = w \quad (10.41)$$

A gain schedule consisting of three PID parameter value sets will be used. The PID controller are tuned at the following GS or w values: 12, 16 and 20 kg/min. These three PID parameter sets are shown down to the left in Figure 10.16. The PID parameters are held piecewise constant in the GS intervals. In each interval, the PID parameters are held fixed for an increasing $GS = w$ value, cf. Figure 10.14.¹³ Figure 10.17 shows the response in the temperature for decreasing values of w . The simulation shows that the *stability of the control system is maintained even if w decreases*.

Finally, assume that you have decided not to use gain scheduling, but instead a PID controller with fixed parameter settings. What is the most critical operating point, at which the controller should be tuned? Is it at maximum flow or at minimum flow?¹⁴

[End of Example 10.4]

10.5.4 Adaptive controller

In an adaptive control system, see Figure 10.18, a mathematical model of the process to be controlled is continuously estimated from samples of the control signal (u) and the process measurement (y_m). The model is typically a transfer function model. Typically, the structure of the model is fixed. The model parameters are estimated continuously using e.g. the least squares method. From the estimated process model the parameters of a PID controller (or of some other control function) are continuously calculated so that the control system achieves specified performance in form of for example stability margins, poles, bandwidth, or minimum variance of the process output variable[10]. Adaptive controllers are commercially available, for example the ECA60 controller (ABB).

¹³The simulator uses the inbuilt gain schedule in LabVIEW's PID Control Toolkit.

¹⁴The answer is minimum flow, because at minimum flow the process gain is at maximum, and also the time-delay (transport delay) is at maximum.

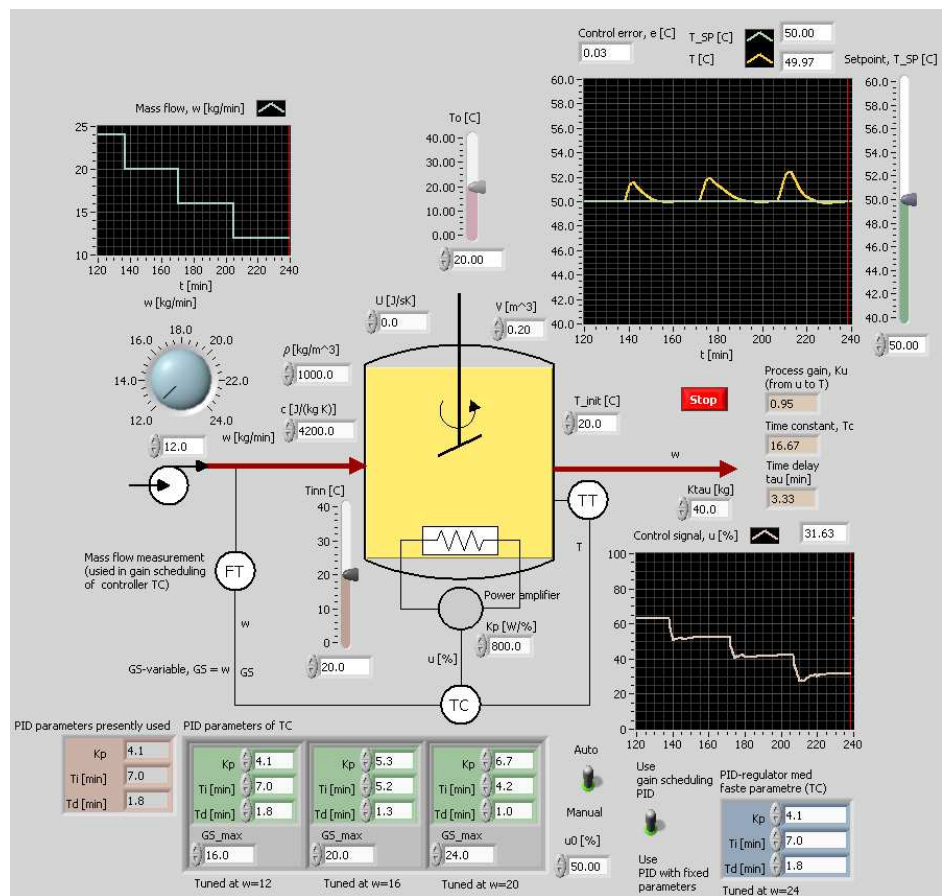


Figure 10.17: Example 10.4: Simulation of temperature control system with a gain schedule based PID controller

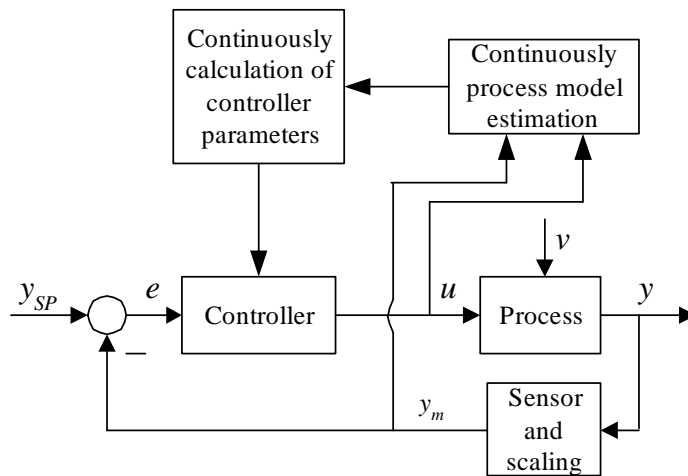


Figure 10.18: Adaptive control system