Journal of Process Control 20 (2010) 1220–1234

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

The setpoint overshoot method: A simple and fast closed-loop approach for PID tuning[☆]

Mohammad Shamsuzzoha, Sigurd Skogestad [*]

Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway

A R T I C L E   I N F O

Article history:
Received 8 April 2010
Received in revised form 16 August 2010
Accepted 16 August 2010

Keywords:
PID control
Tuning
On-line
Closed-loop
First-order with delay process
IMC tuning
Ziegler–Nichols

A B S T R A C T

A simple method has been developed for PID controller tuning of an unidentified process using closed-loop experiments. The proposed method requires one closed-loop step setpoint response experiment using a proportional only controller, and it mainly uses information about the first peak (overshoot) which is very easy to identify. The setpoint experiment is similar to the classical Ziegler–Nichols (1942) experiment, but the controller gain is typically about one half, so the system is not at the stability limit with sustained oscillations. Based on simulations for a range of first-order with delay processes, simple correlations have been derived to give PI controller settings similar to those of the SIMC tuning rules (Skogestad (2003) [6]). The recommended controller gain change is a function of the height of the first peak (overshoot), whereas the controller integral time is mainly a function of the time to reach the peak. The method includes a detuning factor that allows the user to adjust the final closed-loop response time and robustness. The proposed tuning method, originally derived for first-order with delay processes, has been tested on a wide range of other processes typical for process control applications and the results are comparable with the SIMC tunings using the open-loop model.

1. Introduction

The proportional integral (PI) controller is widely used in the process industries due to its simplicity, robustness and wide ranges of applicability in the regulatory control layer. On the basis of a survey of more than 11,000 controllers in the process industries, Desborough and Miller [1] report that more than 97% of the regulatory controllers utilise the PID algorithm. A recent survey [2] from Japan shows that the ratio of applications of PID control, conventional advanced control (feedforward, override, valve position control, gain-scheduled PID, etc.) and model predictive control is about 100:10:1. In addition, the vast majority of the PID controllers do not use derivative action. Even though the PI controller only has two adjustable parameters, it is not simple to find good settings and many controllers are poorly tuned. One reason is that quite tedious plant tests may be needed to obtain improved controller settings. The objective of this paper is to derive a method which is simpler to use than the present ones.

Most tuning approaches are based on an open-loop plant model (g); typically given in terms of the plant's gain (k), time constant (τ) and time delay (θ); see O'Dwyer [3] for an extensive list of methods. Given a plant model g, one popular approach to obtain the controller is direct synthesis [4] which includes the IMC-PID tuning method of Rivera et al. [5]. The original direct synthesis approaches, like that of Rivera et al. [5], give very good performance for setpoint changes, but give sluggish responses to input (load) disturbances for lag-dominant (including integrating) processes with τ/θ larger than about 10. To improve load disturbance rejection, Skogestad [6] proposed the modified SIMC method where the integral time is reduced for processes with a large value of the process time constant τ. The SIMC rule has one tuning parameter, the closed-loop time constant τ_c, and for "fast and robust" control is recommended to choose $\tau_c = \theta$, where θ is the (effective) time delay.

However, these approaches require that one first obtains an open-loop model (g) of the process. There are two problems here. First, an open-loop experiment, for example a step test, is normally needed to get the required process data. This may be time consuming and may result in undesirable output changes. Second, approximations are involved in obtaining the process model g from the open-loop data.

In this paper, the objective is to derive controller tunings based on closed-loop experiments. The simplest is to directly obtain the controller from the closed-loop data, without explicitly obtaining an open-loop model g. This is the approach of the classical Ziegler–Nichols method [7] which requires very little information about the process; namely, the ultimate controller gain (K_u) and the period of oscillations (P_u) which are obtained from a sin-

[☆] This is an extended version of a paper presented at the IFAC Symposium on Dynamics and Control of Process Systems (DYCOPS) in Belgium in July 2010.

[*] Corresponding author. Tel.: +47 7359 4154; fax: +47 7359 4080.
E-mail addresses: skoge@ntnu.no, sigurd.skogestad@chemeng.ntnu.no (S. Skogestad).

0959-1524/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jprocont.2010.08.003

gle experiment. For a PI-controller the recommended settings are $K_c = 0.45K_u$ and $\tau_I = 0.83P_u$. However, there are several disadvantages. First, the system needs to be brought its limit of instability and a number of trials may be needed to bring the system to this point. To avoid this problem one may induce sustained oscillation with an on-off controller using the relay method of Åström and Hägglund [8]. However, this requires that the feature of switching to on/off-control has been installed in the system. Another disadvantage is that the Ziegler–Nichols [7] tunings do not work well on all processes. It is well known that the recommended settings are quite aggressive for lag-dominant (integrating) processes [9] and quite slow for delay-dominant process [6]. To get better robustness for the lag-dominant (integrating) processes, Tyreus and Luyben [9] proposed to use less aggressive settings ($K_c = 0.313K_u$ and $\tau_I = 2.2P_u$), but this makes the response even slower for delay-dominant processes [6]. This is a fundamental problem of the Ziegler–Nichols [7] method because it uses only two pieces of information about the process (K_u, P_u), which correspond to the critical point on the Nyquist curve. This does allow one to distinguish, for example, between a lag-dominant and a delay-dominant process. A fix is to use additional closed-loop experiments, for example an experiment with an integrating controller [15]. A third disadvantage of the Ziegler–Nichols [7] method is that it can only be used on processes for which the phase lag exceeds $-180°$ at high frequencies. For example, it does not work on a simple second-order process.

Therefore, there is need of an alternative closed-loop approach for plant testing and controller tuning which avoids the instability concern during the closed-loop experiment, reduces the number of trails, and works for a wider range of processes. The proposed new method satisfies these concerns:

1. The method uses a single closed-loop experiment with proportional only control. This is similar to the Ziegler-Nichols [7] method, but the process is not forced to its stability limit and it requires less trial-and-error adjustment of the P-controller gain to get to the desired closed-loop response.
2. Of the many parameters that can be obtained from the closed-loop setpoint response, the simplest to observe is the time (t_p) and magnitude (overshoot) of the first peak (see Fig. 1) which is the main information used in the proposed method.
3. The proposed method works well on a wider range of processes than the Ziegler–Nichols [7] method. In particular, it works well also for delay-dominant processes. This is because it that makes use of a third piece of information, namely the relative steady-state change $b = y(\infty)/y_s$.
4. The method applies to processes that give overshoot with proportional only control. This is less restrictive than the Ziegler–Nichols [7] method, which requires sustained oscillations. Thus, unlike the Ziegler–Nichols method, the method works on a simple second-order process.

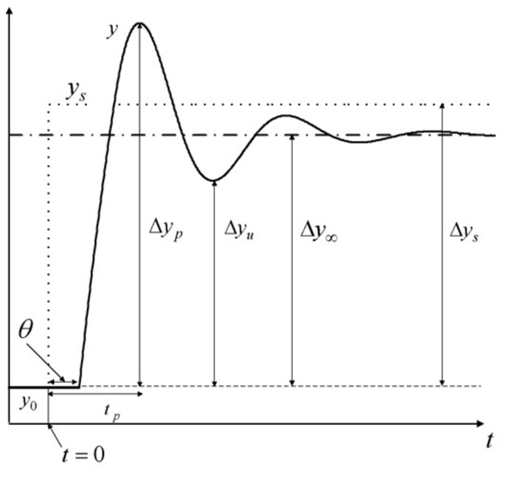

Fig. 1. Closed-loop step setpoint response with P-only control.

In summary, the proposed method is simpler in use than existing approaches and allows the process to be kept under closed-loop control.

2. SIMC PI tuning rules

In Fig. 2 we show the block diagram of a conventional feedback control system, where g denotes the process transfer function and c the feedback controller. The other variables are the manipulated variable u, the measured and controlled output variable y, the setpoint y_s, and the disturbance d which is here assumed to be a "load disturbance" at the plant input. The closed-loop transfer functions from the setpoint and load disturbance to the output are:

$$y = \frac{cg}{1+cg}y_s + \frac{g}{1+cg}d \tag{1}$$

In process control, a first-order process with time delay is a common representation of the process dynamics:

$$g(s) = \frac{ke^{-\theta s}}{\tau s + 1} \tag{2}$$

Here k is the process gain, τ the dominant lag time constant and θ the effective time delay. Most processes in the process industries can be satisfactorily controlled using a PI controller:

$$c(s) = K_c\left(1 + \frac{1}{\tau_I s}\right) \tag{3}$$

which in the time domain corresponds to

$$u(t) = K_c e(t) + \frac{K_c}{\tau_I}\int_0^t e(t)\,dt \tag{4}$$

where $e = y_s - y$. The PI controller has two adjustable parameters, the proportional gain K_c and the integral time τ_I. The ratio $K_I = K_c/\tau_I$ is known as the integral gain.

The SIMC tuning rule [6] is analytically based and widely used in the process industry. For the process in Eq. (2), the SIMC tuning rule gives

$$K_c = \frac{\tau}{k(\tau_c + \theta)} \tag{5}$$

$$\tau_I = \min\{\tau, \quad 4(\tau_c + \theta)\} \tag{6}$$

Note that the original IMC tuning rule [5] always uses $\tau_I = \tau$, but the SIMC rule increases the integral contribution for close-to integrating processes (with τ large) to avoid poor performance (slow settling) to load disturbance. There is one adjustable tuning parameter, the closed-loop time constant (τ_c), which is selected to give the

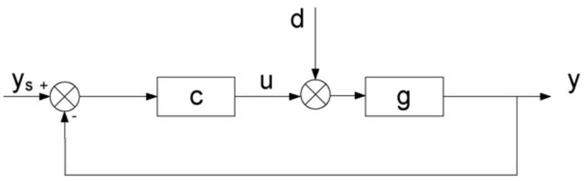

Fig. 2. Block diagram of feedback control system.

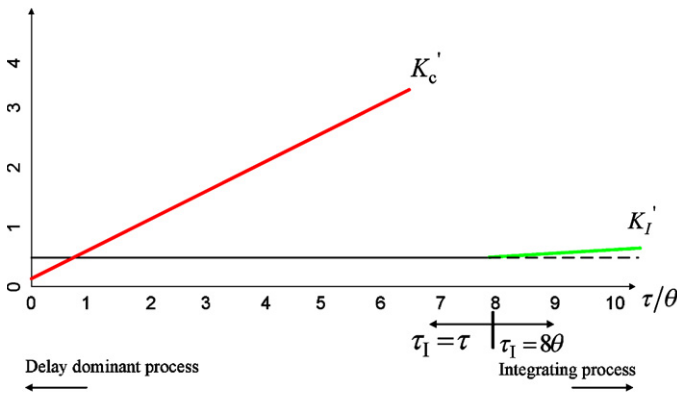

Fig. 3. Scaled proportional and integral gain for SIMC tuning rule.

desired trade-off between performance and robustness. Initially, this study is based on the “fast and robust” setting

$$\tau_c = \theta \tag{7}$$

which gives a good trade-off between performance and robustness. In terms of robustness, this choice gives a gain margin is about 3 and a sensitivity peak (M_s-value) of about 1.6. On dimensionless form, the SIMC tuning rules with $\tau_c = \theta$ become

$$K_c' = kK_c = 0.5\frac{\tau}{\theta} \tag{8}$$

$$K_I' = \frac{kK_c}{\tau_I/\theta} = \max\left(0.5, \frac{1}{16}\frac{\tau}{\theta}\right) \tag{9}$$

The dimensionless gains K_c' and K_I' are plotted as a function of τ/θ in Fig. 3. We note that the integral term (K_I') is relatively more important for delay dominant processes ($\tau/\theta < 1$), while the proportional term K_c' is more significant for processes with a smaller time delay. These insights are useful for the next step when we want to derive tuning rules based on the closed-loop setpoint response.

3. Closed-loop setpoint experiment

As mentioned earlier, the objective is to base the controller tuning on closed-loop data. The simplest closed-loop experiment is probably a setpoint step response where one maintains control of the process, including the change in the output variable. From the setpoint experiment (Fig. 1) one may observe many values, like rise time, period of oscillations, magnitudes and times of overshoots and undershoots, etc. Of all these values, the simplest to observe is the magnitude and time (t_p) of the (first) overshoot, and this information is therefore the basis for the proposed method.

We propose the following procedure:

Step 1. Switch the controller to P-only mode (for example, increase the integral time τ_I to its maximum value or set the integral gain K_I to 0). In an industrial system, with bumpless transfer, the switch should not upset the process.
Step 2. Make a setpoint change with a P-only controller. The P-controller gain K_{c0} used in the experiment does not really matter as long as the response oscillates sufficiently with an overshoot between 0.10 (10%) and 0.60 (60%); about 0.30 (30%) is a good value. Most likely, unless the original controller was tightly tuned, one will need to increase the controller gain to get a sufficiently large overshoot. Note that the controller gain to get 30% overshoot is about half of the “ultimate” controller gain needed in the Ziegler–Nichols closed-loop experiment.
Step 3. From the closed-loop setpoint response experiment, obtain the following values (see Fig. 3):

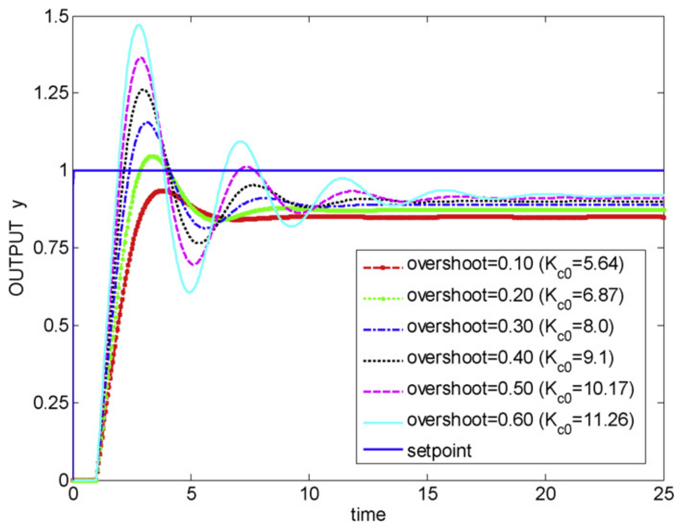

Fig. 4. Step setpoint responses with various overshoots for first-order plus time delay process, $g = e^{-s}/(10s+1)$.

- Controller gain, K_{c0}
- Overshoot = $(\Delta y_p - \Delta y_\infty)/\Delta y_\infty$
- Time from setpoint change to reach peak output (overshoot), t_p
- Relative steady state output change, $b = \Delta y_\infty/\Delta y_s$.

Here the output variable changes are:

$\Delta y_s = y_s - y_0$: setpoint change;
$\Delta y_p = y_p - y_0$: peak output change (at time t_p);
$\Delta y_\infty = y_\infty - y_0$: steady-state output change after setpoint step test.

To find Δy_∞ one needs to wait for the response to settle, which may take some time if the overshoot is relatively large (typically, 0.3 or larger). In such cases, one may stop the experiment when the setpoint response reaches its first minimum (undershoot) and record the corresponding output, Δy_u. As shown in Appendix A, one can then estimate the steady-state change from the following correlation:

$$\Delta y_\infty = 0.45(\Delta y_p + \Delta y_u) \tag{10}$$

Note that (10) involves deviations from the original steady state y_0; in terms of the actual variables we have $y_\infty = 0.45(y_p + y_u) + 0.1y_0$.

To illustrate the use of the closed-loop setpoint experiment, we show in Fig. 4 closed-loop responses for a typical process with a unit time delay ($\theta = 1$) and a ten times larger time constant ($\tau = 10$):

$$g(s) = \frac{e^{-s}}{10s+1} \tag{11}$$

The responses in Fig. 4 are for six different controller gains K_{c0}, which result in overshoots of 0.10, 0.20, 0.30, 0.40, 0.50 and 0.60, respectively. As expected, the closed-loop response gets faster and more oscillatory as the overshoot increases. Note that small overshoots (less than 0.10) are not shown. The main reason is that it is difficult in practice to obtain from experimental data accurate values of the overshoot and corresponding time if the overshoot is too small. Also, large overshoots (larger than about 0.6) are not shown, because these give a long settling time and require more excessive input changes. For these reasons we recommend using an “intermediate” overshoot of about 0.3 (30%) for the closed-loop setpoint experiment.

Fig. 5 shows setpoint responses when the P-controller gain K_{c0} has been adjusted to give an overshoot of 0.3 for a wide range of

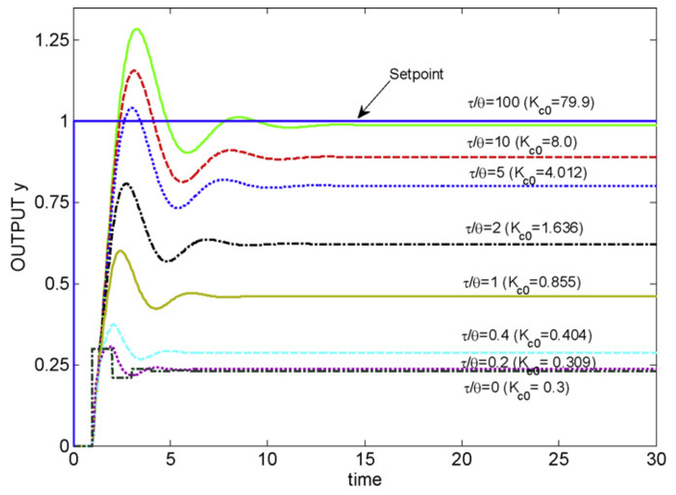

Fig. 5. Step setpoint responses with overshoot of 0.3 (30%) for eight first-order plus time delay processes with τ/θ ranging from 0 to 100 ($g = e^{-\theta s}/(\tau s + 1)$, $\theta = 1$).

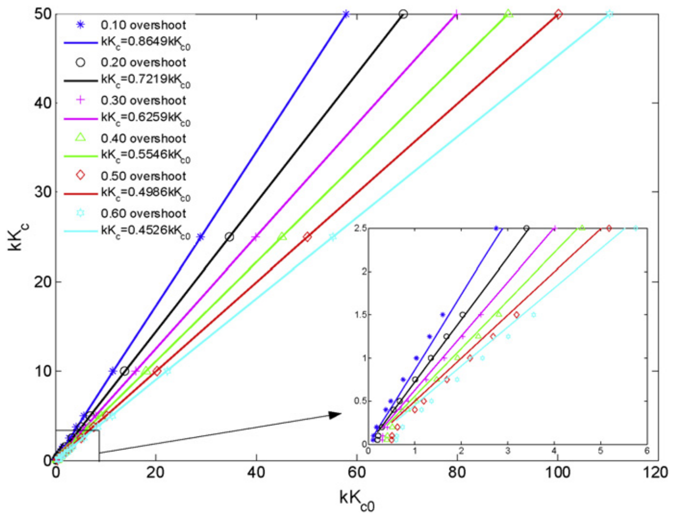

Fig. 6. Relationship between P-controller gain kK_{c0} used in setpoint experiment and corresponding SIMC controller gain kK_c.

first-order plus delay processes with a unit time delay ($\theta = 1$),

$$g(s) = \frac{e^{-s}}{\tau s + 1} \tag{12}$$

The process time constant τ varies from 0 (pure delay process) to 100 (almost integrating process). The time to reach the first peak (t_p) increases somewhat as we increase τ, but the most striking difference is that the steady-state output change (b-value) approaches 1 as we increase τ. Thus, the b-value provides an indirect measure of the value of τ/θ, and we will make use of this observation below.

4. Correlation between setpoint response and SIMC-settings

As mentioned in Section 1, a two-step procedure could be used where first the closed-loop setpoint response experiment is used to determine the open-loop model parameters (k, τ, θ) and next the SIMC-rules (or others) are used to derive PI-settings. However, the objective of this paper is to provide a more direct approach similar to the Ziegler–Nichols [7] closed-loop method.

Thus, the goal is to derive a correlation, preferably as simple as possible, between the setpoint response data (Fig. 1) and the SIMC PI-settings in Eqs. (5) and (6), initially with the choice $\tau_c = \theta$. For this purpose, we considered 15 first-order with delay models

$$g(s) = \frac{ke^{-\theta s}}{\tau s + 1} \tag{13}$$

that cover a wide range of processes; from delay-dominant to lag-dominant (integrating):

$$\frac{\tau}{\theta} = 0.1,\quad 0.2,\quad 0.4,\quad 0.8,\quad 1.0,\quad 1.5,\quad 2.0,\quad 2.5,\quad 3.0,\quad 5.0,\quad 7.5,\quad 10.0,\quad 20.0,\quad 50.0,\quad 100.0$$

Since we can always scale time with respect to the time delay (θ) and since the closed-loop response depends on the product of the process and controller gains (kK_c) we have without loss of generality used in all simulations $k = 1$ and $\theta = 1$.

For each of the 15 process models (values of τ/θ), we obtained the SIMC PI-settings (K_c and τ_I) using Eqs. (5) and (6) with the choice $\tau_c = \theta$. Furthermore, for each of the 15 processes we generated 6 closed-loop step setpoint responses (Figs. 4 and 5) using P-controllers that give different fractional overshoots.

$$\text{overshoot} = 0.10,\quad 0.20,\quad 0.30,\quad 0.40,\quad 0.50 \quad \text{and} \quad 0.60$$

In total, we then have 90 setpoint responses, and for each of these we record four data: the P-controller gain K_{c0} used in the experiment, the fractional overshoot, the time to reach the overshoot (t_p), and the relative steady-state change, $b = \Delta y_\infty/\Delta y_s$.

4.1. Controller gain (K_c)

We first seek a relationship between the above four data and the corresponding SIMC-controller gain K_c. Recall that with PI-control, the recommended Ziegler–Nichols [7] controller gain for any process is $K_c/K_u = 0.45$, where K_u is the "ultimate" controller gain that gives persistent oscillations in the Ziegler–Nichols experiment. As mentioned, this is generally viewed to be too aggressive and Tyreus and Luyben [9] recommend $K_c/K_u = 0.31$. Note that K_u is similar to our K_{c0}, and since the Ziegler–Nichols experiment is similar to a setpoint response with about 100% overshoot, one may hope that we may use a similar simple relationship. Indeed, as illustrated in Fig. 6, where we plot kK_c (scaled SIMC PI-controller gain for the process) as a function of kK_{c0} (scaled experimental controller gain for the same process) for the 90 setpoint experiments, the ratio

$$\frac{K_c}{K_{c0}} = A \tag{14}$$

is approximately constant for a fixed value of the overshoot, independent of the value of τ/θ. In Fig. 7 we plot the value of A, obtained as the best fit of the slopes of the lines in Fig. 6, as a function of the overshoot. The ratio A is found to vary from 0.85 for overshoot = 0.1, to 0.62 for overshoot = 0.3, and 0.45 for overshoot = 0.6. The following equation (solid line in Fig. 7) fits the data well

$$A = [1.152(\text{overshoot})^2 - 1.607(\text{overshoot}) + 1.0] \tag{15}$$

The correlation in Eq. (15) is based on data with overshoots between 0.1 and 0.6 and should not be extended outside this range. To compare with the Ziegler–Nichols [7] ultimate gain approach, note that a value of A of about 0.31 [9] seems reasonable if we imagine extending Fig. 7 to overshoots over 100%.

Actually, a closer look at Fig. 6 reveals that a constant slope, use of Eqs. (14) and (15), only fits the data well for a scaled controller gain $K_c' = kK_c$ greater than about 0.5. Fortunately, a good fit of the controller gain K_c is not so important for delay-dominant processes ($\tau/\theta < 1$) where $K_c' < 0.5$, because we recall from the discussion of the SIMC rules (Fig. 2) that the integral gain K_I is more important for such processes. This is discussed in more detail below.

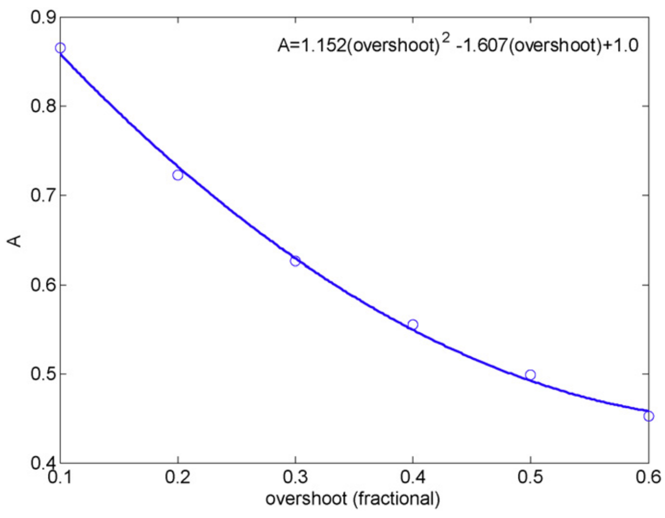

Fig. 7. Variation of A with overshoot using data (slopes) from Fig. 6.

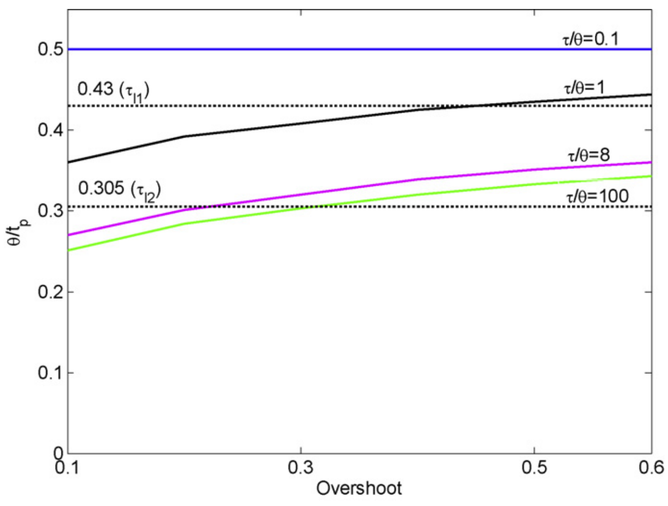

Fig. 8. Ratio of process time delay (θ) and setpoint overshoot time (t_p) as a function of overshoot for four first-order with delay processes (solid lines). Dotted lines: values of θ/t_p used in final correlations.

4.2. Integral time (τ_I)

Next, we want to find a simple correlation for the integral time. We follow the SIMC tuning formula in Eq. (6) and use the minimum from two cases:

- Case (1). $\tau_{I1} = \tau$ for processes with a relatively large delay θ.
- Case (2). $\tau_{I2} = 4(\tau_c + \theta)$ for processes with a relatively small delay θ including integrating processes. In the following we consider the nominal choice $\tau_c = \theta$ which gives $\tau_{I2} = 8\theta$, but we will later show how to reintroduce a detuning factor.

Case (1): We do not know the value of the dominant time constant τ. However, τ enters into the SIMC rule for the controller gain in Eq. (8), so we can insert $\tau = \tau_{I1}$ into Eq. (8) and solve for τ_{I1} to get:

$$\tau_{I1} = 2kK_c\theta \tag{16}$$

Rewrite kK_c as

$$kK_c = \frac{K_c}{K_{c0}} kK_{c0} \tag{17}$$

Here, $K_c/K_{c0} = A$ where A is given as a function of the overshoot in Eq. (15). The value of the loop gain kK_{c0} for the P-control setpoint experiment is given from the value of b:

$$kK_{c0} = \left|\frac{b}{(1-b)}\right| \tag{18}$$

To prove this, note from Eq. (1) that the closed-loop setpoint response is $\Delta y/\Delta y_s = gc/(1+gc)$ and with a P-controller with gain K_{c0}, the steady-state value is $\Delta y_\infty/\Delta y_s = kK_{c0}/(1+kK_{c0}) = b$ and we derive Eq. (18). The absolute value is included to avoid problems if $b > 1$, as may occur because of inaccurate data or for an unstable process.

To sum up, we have derived the following expression for the integral time for a delay-dominant process (with $\tau/\theta < 8$):

$$\tau_{I1} = 2A\left|\frac{b}{(1-b)}\right|\theta \tag{19}$$

Case (2): For a lag-dominant (including integrating) process with $\tau/\theta > 8$, the nominal SIMC integral time SIMC is

$$\tau_{I2} = 8\theta \tag{20}$$

Eqs. (19) and (20) for the integral time have all known parameters except the effective time delay θ. One could obtain θ the directly from the closed-loop setpoint experiment, but this is generally not easy. Fortunately, as shown in Fig. 8, there is a reasonably good correlation between θ and the setpoint peak time t_p which is much easier to observe.

Case (1): For processes with a relatively large time delay ($\tau/\theta < 8$), the ratio θ/t_p varies between 0.27 (for $\tau/\theta = 8$ with overshoot = 0.1) and 0.5 (for $\tau/\theta = 0.1$ with all overshoots). For the intermediate overshoot of 0.3, the ratio θ/t_p varies between 0.32 and 0.50. A conservative choice would be to use $\theta = 0.5t_p$ because this gives the largest integral time. However, to improve performance for processes with smaller time delays, we propose to use $\theta = 0.43t_p$ which is only 14% lower than 0.50 (the worst case).

In conclusion, we have for a process with a relatively large time delay:

$$\tau_{I1} = 0.86A\left|\frac{b}{(1-b)}\right|t_p \tag{21}$$

Case (2): For $\tau/\theta > 8$ we see from Fig. 8 that the ratio θ/t_p varies between 0.25 (for $\tau/\theta = 100$ with overshoot = 0.1) and 0.36 (for $\tau/\theta = 8$ with overshoot 0.6). We select to use the average value $\theta = 0.305t_p$ which is only 15% lower than 0.36 (the worst case). Also note that for the intermediate overshoot of 0.3, the ratio θ/t_p varies between 0.30 and 0.32. In summary, we have for a lag-dominant process

$$\tau_{I2} = 2.44t_p \tag{22}$$

4.3. Conclusion

For the nominal choice $\tau_c = \theta$, the integral time τ_I is obtained as the minimum of the above two values:

$$\tau_I = \min(\tau_{I1}, \tau_{I2}) = \min\left(0.86A\left|\frac{b}{(1-b)}\right|t_p,\quad 2.44t_p\right) \tag{23}$$

Remark. In effect, we are here estimating the effective delay θ from t_p only, by using $\theta = 0.43t_p$ and $\theta = 0.305t_p$ for obtaining τ_{I1} and τ_{I2}, respectively. It is possible get a better fit to the SIMC-value of τ_I by making θ also a function of, for example, the overshoot and the b-value, but we have here chosen to keep it simple.

5. Final choice of the controller settings (detuning)

So far we have derived nominal controller settings that correspond to a closed-loop time constant equal to the effective delay

($\tau_c = \theta$). However, in many cases one may want to use less aggressive (detuned) settings ($\tau_c > \theta$), or one may even want to speed up the response ($\tau_c < \theta$). To this end, we want to introduce a detuning factor F, where $F > 1$ corresponds less aggressive settings and $F < 1$ to more aggressive settings [12].

To find out how the factor F should be included in the expressions for the controller gain and integral time we go back to the SIMC settings in Eqs. (5) and (6). The nominal SIMC setting for $\tau_c = \theta$ considered so far (denoted * for clarity) are

$$K_c^* = \left(\frac{0.5\tau}{k\theta}\right), \quad \tau_I^* = \min(\tau, \tau_{12}^*) \quad \text{where} \quad \tau_{12}^* = 8\theta$$

The general formulas (5) and (6) can then be rewritten as

$$K_c = \frac{K_c^*}{F} \tag{24}$$

$$\tau_I = \min(\tau, \tau_{12}^* F) \tag{25}$$

where

$$F = \frac{\tau_c + \theta}{2\theta} \tag{26}$$

Note that $\tau_c = \theta$ gives $F = 1$. Formulas (24) and (25) can now be used to generalize the proposed settings in Eqs. (14) and (23), which are based $\tau_c = \theta$. In conclusion, the final tuning formulas for the proposed "Setpoint Overshoot Method" method are:

$$K_c = \frac{K_{c0}A}{F} \tag{27}$$

$$\tau_I = \min\left(0.86A\left|\frac{b}{(1-b)}\right|t_p, \quad 2.44t_pF\right) \tag{28}$$

where $A = [1.152(\text{overshoot})^2 - 1.607(\text{overshoot}) + 1.0]$ and F is a detuning parameter. $F = 1$ gives the "fast and robust" SIMC settings corresponding to $\tau_c = \theta$, see Eq. (26). To detune the response and get more robustness one selects $F > 1$, but in special cases one may select $F < 1$ to speed up the closed-loop response.

6. Analysis and simulation

Closed-loop simulations have been conducted for 33 different processes and the proposed tuning procedure provides in all cases acceptable controller settings with respect to both performance and robustness.

For each process, PI-settings were obtained based on step response experiments with three different overshoot (about 0.1, 0.3 and 0.6) and compared with the SIMC settings.

The closed-loop performance is evaluated by introducing a unit step change in both the set-point and load disturbance, i.e., ($y_s = 1$ and $d = 1$).

Output performance (y) is quantified by computing the integrated absolute error, $\text{IAE} = \int_0^\infty |y - y_s| dt$. Manipulated variable usage is quantified by calculating the total variation (TV) of the input (u), which is the sum of all its moves up and down. If we discretize the input signal as a sequence $[u_1, u_2, u_3, \ldots, u_i \ldots]$ then $\text{TV} = \sum_{i=1}^{\infty} |u_{i+1} - u_i|$. Note also that TV is the integral of the absolute value of the derivative of the input, $\text{TV} = \int_0^\infty |du/dt| dt$, so TV is a good measure of the smoothness. To evaluate the robustness, we compute the maximum closed-loop sensitivity, defined as $M_s = \max_\omega |1/[1 + gc(j\omega)]|$. Since M_s is the inverse of the shortest distance from the Nyquist curve of the loop transfer function to the critical point (−1, 0), a small M_s-value indicates that the control system has a large stability margin. We want IAE, TV and M_s all to be small, but for a well tuned controller there is a trade-off, which means that a reduction in IAE implies an increase in TV and M_s (and *vice versa*).

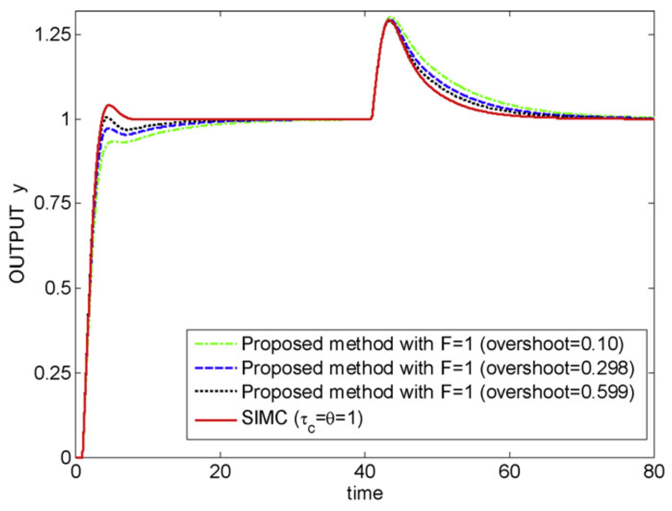

Fig. 9. Responses for PI-control of simple first-order process $g = e^{-s}/(5s + 1)$ (E17). Setpoint change at $t = 0$; load disturbance of magnitude 1 at $t = 40$.

The results for the 33 example processes, which include the 15 examples (E1–E15) from Skogestad [6] and some additional examples with oscillating and unstable plant dynamics, are listed in Table 1. For first-order processes (E14, E15, E16), a small delay must be added (E14a, E15a, E16a) to be able to get the closed-loop overshoot needed to apply the proposed method. All results are without detuning ($F = 1$). The complete simulation results for all cases are available in a technical report [11].

As expected, when the method is tested on first-order plus delay processes, similar to those used to develop the method, the responses are similar to the SIMC-responses, independent of the value of the overshoot. Typical cases are E17 (first-order with delay), E21 (pure time delay) and E24 (integrating with delay); see Figs. 9–11. For models that are *not* first-order plus delay (typical cases are E1, E5 and E8; see Figs. 12–14), the agreement with the SIMC-method is best for the intermediate overshoot (around 0.3). A small overshoot (around 0.1) typically give "slower" and more robust PI-settings, whereas a large overshoot (around 0.6) gives more aggressive PI-settings. In some sense this is good, because it means that a more "careful" step response results in more "careful" tunings. Also note that the user always has the option to use the

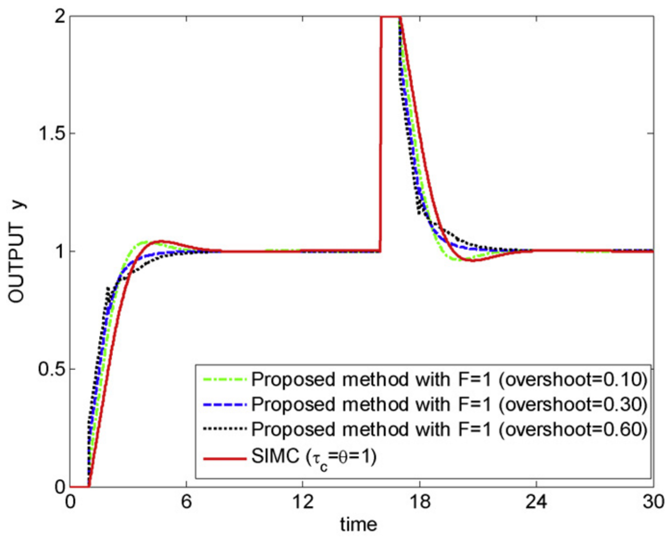

Fig. 10. Responses for PI-control of pure time delay process $g = e^{-s}$ (E21), setpoint change at $t = 0$; load disturbance of magnitude 1 at $t = 15$.

Table 1
PI controller setting for proposed method ($F = 1$) and comparison with SIMC method ($\tau_c = \theta_{\mathrm{effective}}$ obtained using half rule).

Case	Process model	P-control setpoint experiment							Resulting PI-controller					
									Setpoint			Load disturbance		
		K_{c0}	Overshoot	t_p	b	K_c	τ_I	M_s	IAE (y)	TV (u)	Overshoot (y)	IAE (y)	TV (u)	Peak value (y)
E1	$\frac{1}{(s+1)(0.2s+1)}$	5.0	0.127	0.710	0.833	4.074	1.732	1.33	0.44	8.0	0.03	0.43	1.17	0.20
		15.0	0.322	0.393	0.937	9.031	0.958	1.74	0.30	23.72	0.29	0.11	1.81	0.11
		40.0	0.508	0.230	0.976	19.23	0.561	2.62	0.33	74.74	0.53	0.03	3.05	0.06
		SIMC	–	–	–	5.50	0.80	1.56	0.36	12.65	0.23	0.15	1.55	0.15
E2	$\frac{(-0.3s+1)(0.08s+1)}{(2s+1)(s+1)(0.4s+1)(0.2s+1)(0.05s+1)^3}$	0.85	0.131	5.31	0.46	0.688	3.141	1.41	4.56	1.20	0	4.57	1.01	0.60
		1.50	0.303	4.450	0.60	0.929	3.562	1.56	3.83	1.76	0	3.83	1.09	0.56
		2.50	0.567	3.886	0.714	1.148	3.836	1.73	3.43	2.43	0.04	3.35	1.28	0.54
		SIMC	–	–	–	0.85	2.50	1.66	3.56	1.90	0.1	2.97	1.26	0.57
E3	$\frac{2(15s+1)}{(20s+1)(s+1)(0.1s+1)^2}$	2.75	0.107	0.711	0.846	2.315	1.734	1.48	0.55	4.74	0.03	0.75	1.20	0.35
		5.0	0.314	0.527	0.909	3.043	1.287	1.70	0.43	7.16	0.17	0.43	1.48	0.30
		9.50	0.599	0.402	0.950	4.283	0.981	2.18	0.45	12.88	0.36	0.23	2.14	0.24
		SIMC	–	–	–	2.33	1.05	1.55	0.47	4.96	0.12	0.45	1.29	0.34
E4	$\frac{1}{(s+1)^4}$	0.50	0.10	6.60	0.333	0.426	2.416	1.36	5.66	1.0	0	5.66	1.0	0.69
		1.25	0.304	5.250	0.556	0.773	3.489	1.56	4.50	1.49	0	4.50	1.09	0.62
		2.50	0.598	4.414	0.714	1.128	4.281	1.85	3.99	2.56	0.06	3.80	1.44	0.56
		SIMC	–	–	–	0.30	1.50	1.46	5.59	1.15	0.05	5.40	1.10	0.71
E5	$\frac{1}{(s+1)(0.2s+1)(0.04s+1)(0.008s+1)}$	3.0	0.104	0.922	0.750	2.534	2.005	1.33	0.79	4.76	0	0.79	1.08	0.28
		6.50	0.292	0.615	0.867	4.093	1.50	1.59	0.46	9.13	0.13	0.37	1.42	0.21
		15.0	0.599	0.429	0.938	6.766	1.048	2.18	0.47	20.96	0.37	0.16	2.28	0.16
		SIMC	–	–	–	3.72	1.10	1.59	0.45	8.26	0.16	0.30	1.45	0.22
E6	$\frac{(0.17s+1)^2}{s(s+1)^2(0.028s+1)}$	0.40	0.110	7.636	1.0	0.335	18.63	1.48	6.14	0.75	0.23	55.66	1.46	2.91
		0.80	0.301	4.987	1.0	0.496	12.169	1.77	4.74	1.29	0.37	24.51	1.81	2.13
		2.0	0.576	3.199	1.0	0.913	7.805	2.73	4.71	3.57	0.58	8.55	3.08	1.32
		SIMC	–	–	–	0.296	13.5	1.48	6.50	0.67	0.28	45.61	1.55	3.09
E7	$\frac{-2s+1}{(s+1)^3}$	0.22	0.111	6.717	0.180	0.184	1.062	1.96	7.42	1.41	0.16	8.92	1.63	1.06
		0.40	0.309	5.979	0.286	0.245	1.263	2.13	7.04	1.57	0.21	8.62	1.83	1.08
		0.60	0.604	5.595	0.375	0.270	1.298	2.28	7.13	1.73	0.26	8.75	2.01	1.10
		SIMC	–	–	–	0.214	1.50	1.66	6.78	1.05	0.02	8.34	1.28	1.03
E8	$\frac{1}{s(s+1)^2}$	0.32	0.106	8.985	1.0	0.270	21.923	1.51	7.22	0.60	0.23	81.26	1.46	3.62
		0.58	0.307	6.188	1.0	0.357	15.10	1.75	6.21	0.90	0.35	42.33	1.72	2.92
		1.15	0.610	4.492	1.0	0.516	10.961	2.30	5.61	1.67	0.52	21.26	2.50	2.25
		SIMC	–	–	–	0.330	12.0	1.76	6.45	0.84	0.38	36.36	1.78	3.02
E9	$\frac{e^{-s}}{(s+1)^2}$	0.50	0.116	4.55	0.333	0.415	1.622	1.43	3.91	1.0	0	3.91	1.0	0.73
		1.0	0.321	3.85	0.50	0.603	1.995	1.58	3.31	1.27	0	3.31	1.04	0.69
		1.70	0.623	3.453	0.63	0.758	2.251	1.74	3.03	1.69	0.03	2.97	1.19	0.67
		SIMC	–	–	–	0.50	1.50	1.61	3.38	1.32	0.06	3.14	1.15	0.71
E10	$\frac{e^{-s}}{(20s+1)(2s+1)}$	4.50	0.11	11.65	0.818	3.769	28.426	1.42	7.54	7.32	0.01	7.54	1.10	0.21
		8.0	0.301	8.425	0.889	4.966	20.557	1.62	5.92	10.99	0.14	4.14	1.34	0.18
		14.0	0.594	6.594	0.933	6.326	16.088	1.90	6.28	16.42	0.28	2.54	1.72	0.16
		SIMC	–	–	–	5.25	16.0	1.72	6.34	12.31	0.22	3.05	1.49	0.17
E11	$\frac{(-s+1)e^{-s}}{(6s+1)(2s+1)^2}$	0.70	0.119	16.62	0.412	0.577	8.25	1.41	14.3	1.08	0	14.32	1.01	0.65
		1.40	0.344	13.67	0.583	0.817	9.602	1.59	11.72	1.60	0	11.78	1.09	0.61
		2.20	0.608	12.28	0.687	0.987	10.423	1.74	10.78	2.11	0.03	10.59	1.25	0.58
		SIMC	–	–	–	0.70	7.0	1.63	11.5	1.59	0.08	10.14	1.20	0.62

E12	$\frac{(6s+1)(3s+1)e^{-0.3s}}{(10s+1)(8s+1)(s+1)}$	12.0	0.128	0.89	0.923	9.766	2.171	1.81	0.90	23.40	0.08	0.23	1.32	0.09
		15.0	0.308	0.836	0.938	9.220	2.040	1.75	0.92	21.54	0.06	0.23	1.26	0.09
		20.0	0.609	0.792	0.952	8.971	1.933	1.72	0.93	20.79	0.06	0.22	1.24	0.09
		SIMC	–	–	–	7.40	1.0	1.66	1.07	18.28	0.19	0.15	1.39	0.10
E13	$\frac{(2s+1)e^{-s}}{(10s+1)(0.5s+1)}$	4.0	0.142	2.25	0.80	3.179	5.490	1.87	2.70	7.44	0.04	1.74	1.28	0.23
		4.75	0.302	2.20	0.826	2.943	5.367	1.76	2.88	6.60	0.04	1.85	1.20	0.23
		6.0	0.570	2.143	0.857	2.75	5.068	1.68	3.03	6.01	0.05	1.88	1.15	0.23
		SIMC	–	–	–	2.88	4.50	1.74	2.86	6.56	0.06	1.62	1.20	0.23
E14	$\frac{-s+1}{s}$	No oscillation with	–	–	–	–	–	–	–	–	–			
		P-controller, proposed	–	–	–	–	–	–	–	–	–			
		method does not apply	–	–	–	–	–	–	–	–	–			
		SIMC	–	–	–	0.50	8.0	2.0	2.92	2.04	0.20	17.3	3.40	1.87
E14 (a)	$\frac{(-s+1)e^{-0.1s}}{s}$	0.65	0.106	1.887	1.0	0.548	4.604	2.48	2.74	2.31	0.40	9.91	3.81	2.01
		0.70	0.285	1.655	1.0	0.445	4.037	2.01	3.58	1.74	0.45	11.63	3.40	2.17
		0.75	0.558	1.584	1.0	0.347	3.866	1.84	4.89	1.28	0.47	16.56	3.00	2.42
		SIMC	–	–	–	0.455	8.80	1.95	3.38	1.58	0.21	20.69	2.69	2.09
E15	$\frac{-s+1}{(s+1)}$	No oscillation with	–	–	–	–	–	–	–	–	–			
		P-controller, proposed	–	–	–	–	–	–	–	–	–			
		method does not apply	–	–	–	–	–	–	–	–	–			
		SIMC	–	–	–	0.50	1.0	2.0	2.0	1.02	0	2.85	3.0	0.89
E15 (a)	$\frac{(-s+1)e^{-0.2s}}{(s+1)}$	0.42	0.107	1.75	0.296	0.353	0.532	2.88	2.83	2.56	0.44	3.96	3.25	1.20
		0.51	0.31	1.55	0.338	0.314	0.418	3.90	4.12	3.88	0.67	5.26	4.41	1.26
		0.60	0.605	1.50	0.375	0.270	0.348	4.64	5.24	4.83	0.77	6.36	5.29	1.27
		SIMC	–	–	–	0.417	1.0	1.88	1.86	1.02	0	3.26	2.25	1.03
E16	$\frac{1}{(s+1)}$	No oscillation with	–	–	–	–	–	–	–	–	–			
		P-controller, proposed	–	–	–	–	–	–	–	–	–			
		method does not apply	–	–	–	–	–	–	–	–	–			
		SIMC	–	–	–	$\tau_c = \theta_{\text{effective}} = 0$ so SIMC-rule does not apply (gives infinite controller gain)								
E16 (a)	$\frac{e^{-0.05s}}{(s+1)}$	12.0	0.123	0.217	0.923	9.833	0.530	1.61	0.15	21.49	0.12	0.054	1.24	0.092
		16.0	0.309	0.174	0.941	9.819	0.425	1.63	0.16	22.08	0.17	0.043	1.32	0.091
		22.50	0.612	0.146	0.957	10.082	0.355	1.68	0.17	23.55	0.22	0.035	1.42	0.090
		SIMC	–	–	–	10.0	0.40	1.65	0.16	22.83	0.18	0.04	1.36	0.091
E17	$\frac{e^{-s}}{(5s+1)}$	2.75	0.10	3.60	0.733	2.338	7.240	1.50	3.09	4.49	0	3.09	1.0	0.30
		4.0	0.298	3.049	0.80	2.494	6.538	1.56	2.62	4.96	0	2.62	1.04	0.29
		5.75	0.599	2.705	0.852	2.592	6.030	1.60	2.33	5.29	0	2.33	1.07	0.29
		SIMC	–	–	–	2.50	5.0	1.59	2.17	5.16	0.04	2.0	1.08	0.29
E18	$\frac{e^{-s}}{(s+1)}$	0.50	0.109	2.75	0.333	0.419	0.991	1.47	2.39	1.06	0.01	2.37	1.01	0.74
		0.90	0.326	2.40	0.474	0.538	1.111	1.58	2.09	1.23	0.01	2.06	1.03	0.72
		1.35	0.593	2.25	0.575	0.610	1.181	1.66	1.99	1.39	0.02	1.94	1.08	0.72
		SIMC	–	–	–	0.50	1.0	1.59	2.17	1.26	0.04	2.06	1.08	0.72
E19	$\frac{e^{-s}}{(0.2s+1)}$	0.12	0.113	2.0	0.107	0.10	0.172	1.76	2.16	1.26	0.11	2.19	1.25	0.99
		0.30	0.292	2.0	0.231	0.189	0.325	1.67	1.88	1.12	0.05	1.87	1.10	0.99
		0.60	0.590	2.0	0.375	0.272	0.467	1.66	1.72	1.01	0	1.72	1.01	0.99
		SIMC	–	–	–	0.10	0.20	1.59	2.17	1.09	0.04	2.16	1.08	0.99

Table 1 (*Continued*)

Case	Process model	P-control setpoint experiment							Resulting PI-controller					
		K_{c0}	Overshoot	t_p	b	K_c	τ_I	M_s	Setpoint			Load disturbance		
									IAE (y)	TV (u)	Overshoot (y)	IAE (y)	TV (u)	Peak value (y)
E20	$\frac{e^{-s}}{(0.05s+1)^2}$	0.10	0.10	2.0	0.091	0.085	0.146	1.70	2.01	1.17	0.08	2.01	1.17	1.0
		0.30	0.30	2.0	0.231	0.187	0.321	1.61	1.74	1.02	0.01	1.74	1.01	1.0
		0.60	0.60	2.0	0.375	0.270	0.464	1.64	1.72	1.01	0	1.72	1.00	1.0
		SIMC	–	–	–	0.037	0.075	1.59	2.22	1.09	0.04	2.22	1.08	1.0
E21	e^{-s}	0.10	0.10	2.0[a]	0.091	0.085	0.146	1.60	1.85	1.09	0.04	1.85	1.08	1.0
		0.30	0.30	2.0	0.231	0.187	0.321	1.53	1.72	1.07	0	1.72	1.02	1.0
		0.60	0.60	2.0	0.375	0.270	0.465	1.59	1.72	1.16	0	1.72	1.14	1.0
		SIMC	–	–	–	$K_c = 0$, $K_c/\tau_I = 0.50$		1.59	2.17	1.08	0.04	2.17	1.08	1.0
E22	$\frac{100e^{-s}}{100s+1}$	0.60	0.118	3.911	0.984	0.496	9.544	1.66	3.79	1.15	0.22	19.24	1.43	1.95
		0.80	0.301	3.293	0.988	0.496	8.034	1.68	3.79	1.18	0.25	16.19	1.50	1.94
		1.10	0.598	2.913	0.991	0.496	7.108	1.71	3.79	1.22	0.28	14.33	1.56	1.93
		SIMC	–	–	–	0.50	8.0	1.69	3.78	1.20	0.26	16.0	1.51	1.93
E23	$\frac{(10s+1)e^{-s}}{s(2s+1)}$	0.22	0.136	2.633	1.0	0.177	6.425	2.14	3.73	0.51	0.13	39.48	1.69	5.35
		0.26	0.303	2.563	1.0	0.161	6.255	1.96	3.85	0.43	0.08	42.74	1.51	5.45
		0.33	0.595	2.442	1.0	0.149	5.958	1.84	3.96	0.39	0.09	44.50	1.40	5.55
		SIMC	–	–	–	0.10	2.0	1.67	3.96	0.30	0.23	24.92	1.46	5.93
E24	$\frac{e^{-s}}{s}$	0.59	0.108	3.976	1.0	0.495	9.702	1.67	3.98	1.16	0.24	19.6	1.47	1.99
		0.80	0.302	3.282	1.0	0.496	8.008	1.70	3.94	1.21	0.28	16.15	1.55	1.97
		1.10	0.60	2.909	1.0	0.496	7.098	1.72	3.92	1.24	0.30	14.31	1.61	1.96
		SIMC	–	–	–	0.50	8.0	1.70	3.92	1.22	0.28	16.0	1.55	1.96
E25	$\frac{(s+6)^2}{s(s+1)^2(s+36)}$	0.41	0.117	7.51	1.0	0.339	18.323	1.49	6.09	0.76	0.24	53.93	1.48	2.89
		0.80	0.304	4.989	1.0	0.495	12.173	1.77	4.76	1.29	0.37	24.61	1.81	2.14
		1.90	0.566	3.305	1.0	0.873	8.064	2.64	4.67	3.39	0.57	9.09	2.99	1.36
		SIMC	–	–	–	0.417	9.60	1.74	5.18	1.07	0.39	23.04	1.82	2.36
E26	$\frac{-1.6(-0.5s+1)}{s(3s+1)}$	−0.13	0.116	14.937	1.0	−0.108	36.447	1.49	12.07	0.25	0.24	338.2	1.49	−9.10
		−0.25	0.296	9.685	1.0	−0.156	23.632	1.77	9.46	0.41	0.37	151.3	1.82	−6.80
		−0.50	0.554	6.653	1.0	−0.232	16.232	2.31	8.83	0.78	0.53	70.16	2.53	−4.99
		SIMC	–	–	–	−0.156	16.0	1.93	9.71	0.45	0.46	102.5	2.08	−6.53
E27	$\frac{e^{-s}}{s^2}$	Not possible to stabilize with PI controller												
E28	$\frac{(-2s+1)}{(s+1)^3}$	0.22	0.111	6.717	0.180	0.184	1.062	1.96	7.41	1.41	0.16	8.92	1.63	1.06
		0.40	0.309	5.979	0.286	0.246	1.263	2.14	7.04	1.56	0.21	8.62	1.83	1.08
		0.60	0.604	5.595	0.375	0.270	1.298	2.28	7.13	1.73	0.26	8.75	2.01	1.10
		SIMC	–	–	–	0.214	1.50	1.66	6.78	1.05	0.02	8.34	1.28	1.03
E29	$\frac{(-s+1)e^{-2s}}{(s+1)^5}$	0.17	0.111	12.787	0.145	0.142	1.562	1.71	13.52	1.24	0.10	13.47	1.24	0.95
		0.40	0.304	11.987	0.286	0.247	2.547	1.70	11.66	1.17	0.07	11.63	1.18	0.94
		0.73	0.595	11.487	0.422	0.330	3.256	1.75	10.59	1.17	0.05	10.55	1.17	0.94
		SIMC	–	–	–	0.115	1.50	1.57	14.14	1.09	0.04	14.19	1.10	0.95
E30	$\frac{9}{(s+1)(s^2+2s+9)}$	0.55	0.101	1.63	0.355	0.467	0.643	1.42	1.47	1.09	0.01	1.43	1.03	0.68
		1.25	0.322	1.40	0.556	0.752	0.905	1.72	1.26	1.57	0.01	1.23	1.21	0.63
		2.30	0.581	1.24	0.697	1.048	1.118	2.21	1.19	2.8	0.05	1.08	1.83	0.58
		SIMC	–	–	–	No SIMC PI rule	–	–	–	–	–	–	–	–
E31	$\frac{9}{(s+1)(s^2+s+9)}$	0.30	0.111	1.55	0.231	0.251	0.334	1.58	1.60	1.24	0.08	1.56	1.25	0.81
		0.75	0.31	1.40	0.429	0.460	0.554	2.18	1.53	1.53	0.08	1.60	1.77	0.76
		1.10	0.457	3.326	0.524	0.557	1.607	2.13	2.86	1.44	0	2.86	1.66	0.77
		SIMC	–	–	–	No SIMC PI rule	–	–	–	–	–	–	–	–

E32	$\frac{(s^2+2s+9)(-2s+1)(s+1)e^{-2s}}{(s^2+0.5s+1)(5s+1)^2}$	0.07	0.112	18.132	0.387	0.058	8.198	1.46	15.4	0.12	0	143.2	1.08	6.16
		0.12	0.301	15.043	0.519	0.074	8.667	1.61	12.74	0.16	0.01	119.4	1.17	5.98
		0.18	0.583	13.71	0.618	0.082	8.684	1.70	12.18	0.18	0.05	108.9	1.27	5.91
		SIMC	–	–	–	No SIMC PI rule	–	–	–	–	–	–	–	–
33	$\frac{e^{-s}}{(5s-1)}$	3.10	0.10	4.647	1.476	2.636	10.54	2.12	7.69	9.48	0.87	4.0	2.73	0.56
		4.0	0.30	3.671	1.333	2.487	7.852	2.33	7.96	10.15	0.96	3.81	3.12	0.58
		5.30	0.607	3.164	1.233	2.379	6.475	2.67	9.03	11.12	1.03	4.18	3.63	0.59
		SIMC	–	–	–	No SIMC PI rule		–	–	–	–	–	–	–

[a] For pure time delay process (E21), obtain t_p as end time of peak (or add a small time constant in simulation).

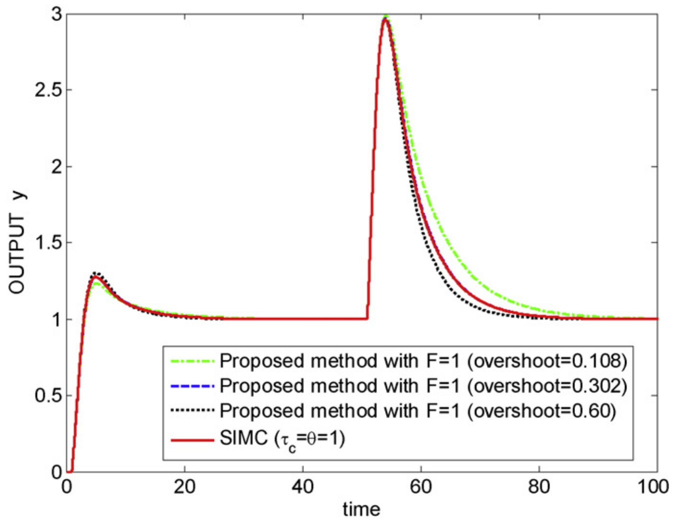

Fig. 11. Responses for PI-control of integrating process $g = e^{-s}/s$ (E24), setpoint change at $t = 0$; load disturbance of magnitude 1 at $t = 50$.

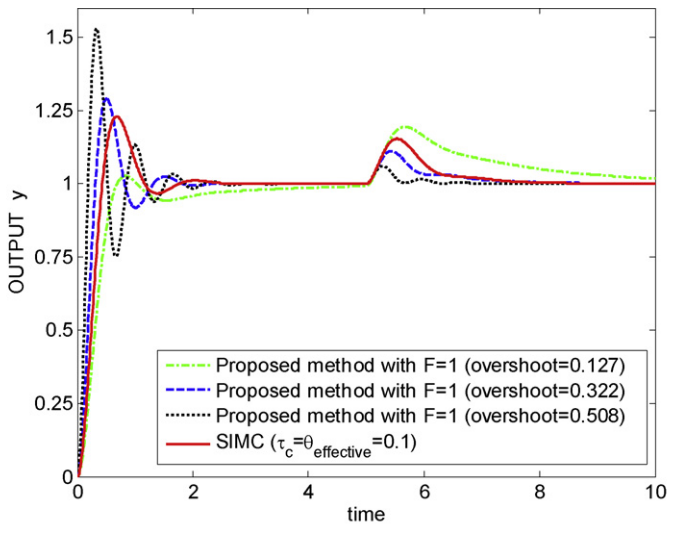

Fig. 12. Responses for PI-control of second-order process $g = 1/(s+1)(0.2s+1)$ (E1), setpoint change at $t = 0$; load disturbance of magnitude 1 at $t = 5$.

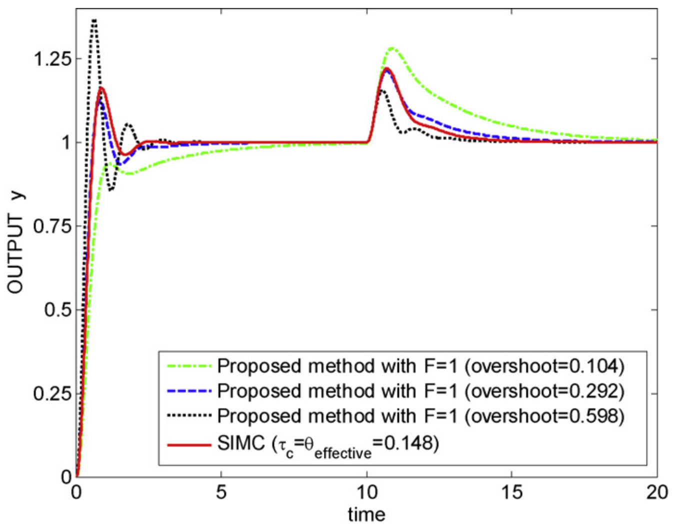

Fig. 13. Responses for PI-control of high-order process $g = 1/(s+1)(0.2s+1)(0.04s+1)(0.008s+1)$ (E5), setpoint change at $t = 0$; load disturbance of magnitude 1 at $t = 10$.

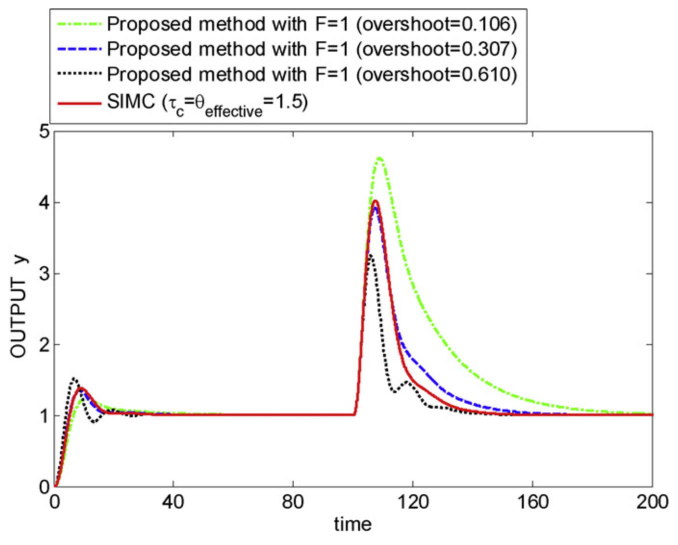

Fig. 14. Responses for PI-control of third-order integrating process $g=1/s(s+1)^2$ (E8), setpoint change at $t=0$; load disturbance of magnitude 1 at $t=100$.

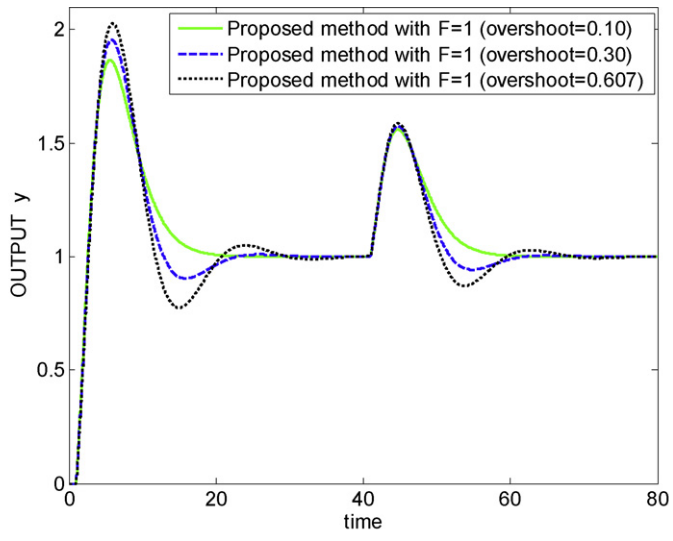

Fig. 15. Responses for PI-control of first-order unstable process $g=e^{-s}/(5s-1)$ (E33), setpoint change at $t=0$; load disturbance of magnitude 1 at $t=40$.

detuning factor F to correct the final tunings. Note that the proposed method works well on some unstable (E33; Fig. 15) and oscillating processes (E30, E31, E32) where there are no PI-rule for the SIMC method.

The effect of using the detuning factor F is illustrated in Fig. 16 using a simple second-order process (case E1). As expected, using $F>1$ results in more robust controller settings.

7. Derivative action (PID control)

The tuning rules derived in this paper are for PI control (3). In theory, one may better robustness and/or better output performance by adding derivative action (PID control). There are many PID implementations, and we here consider the "classical" PID controller on cascade form,

$$c_{\mathrm{PID}}(s)=K_c\left(1+\frac{1}{\tau_I s}\right)\frac{1+\tau_D s}{1+(\tau_D/N)s} \tag{29}$$

Here, τ_D is the derivative time and the filter parameter N is typically around 10 [3]. Because the addition of derivative action comes at

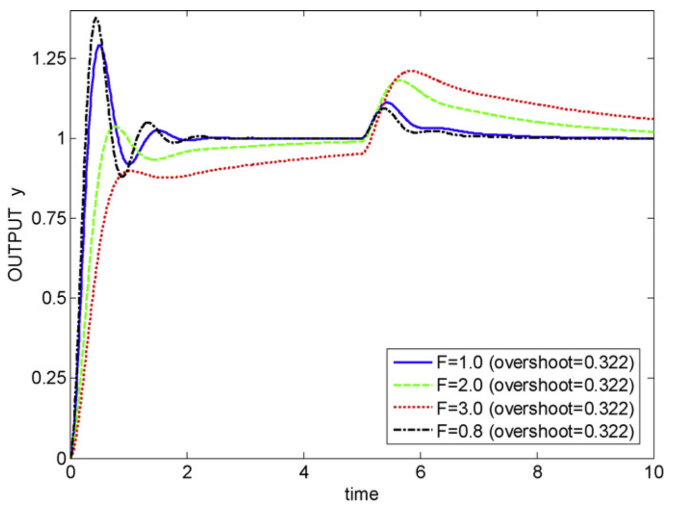

Fig. 16. Effect of detuning factor: responses for PI-control of second-order $g=1/(s+1)(0.2s+1)$ (E1), setpoint change at $t=0$; load disturbance of magnitude 1 at $t=5$.

the expense of a more complex controller, more sensitivity to measurement noise and more input usage, Skogestad [6] recommends that derivative action (PID control) is only justified for *dominant* second-order process,

$$g(s)=\frac{e^{-\theta s}}{(\tau_1 s+1)(\tau_2 s+1)} \tag{30}$$

where "dominant" means that the second-order time constant (τ_2) is larger than the effective time delay θ. Of the 33 processes in Table 1, this would apply to cases E1, E2, E3, E5, E6, E8, E10 and E27. One particular example is the double integrating process (E27, $\tau_2=\infty$), which actually can be stabilized only if we add derivative action to the PI controller.

Is it possible to use derivative action with the closed-loop tuning approach in this paper? Yes, it is, but one cannot just "add on" derivative action to the PI-controller, because also the controller gain and integral time needs to be changed to achieve the benefit of adding derivative action.

We suggest adding the derivative action beforehand by using a PD controller during setpoint experiment,

$$c_{\mathrm{PD}}(s)=K_{c0}\frac{1+\tau_D s}{1+(\tau_D/N)s} \tag{31}$$

The idea is to include the derivative term during the experiment and then include the same term in the final controller. It is like designing a PI controller for a modified process. Two disadvantages with this approach are:

1. We must make sure that the setpoint is differentiated. The "problem" is that in order to avoid the "derivative kick", many industrial PID implementations do not differentiate the setpoint (see Fig. 17), and this will give a too small overshoot in the PD setpoint experiment. There are two fixes to this problem: (a) temporarily use a PD-implementation with setpoint differentiation. (b) Alternatively, and this is better because it gives less input usage, record the "differentiated" output signal $y_D(s)=y(s)(1+\tau_D s)/(1+(\tau_D/N)s)$ (see Fig. 17) during the experiment.
2. We must decide on a value for the derivative time constant (τ_D) before the setpoint experiment.

What value should one select for τ_D?

With the cascade PID form in (29), Skogestad [6] recommends selecting τ_D equal to the dominant second-order time constant, i.e.,

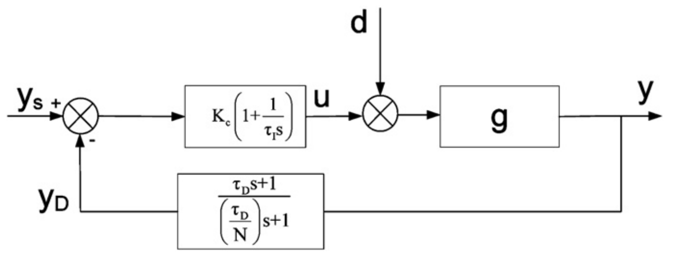
d
y_s
+
-
$K_c\left(1+\frac{1}{\tau_I s}\right)$
u
g
y
y_D
$\frac{\tau_D s+1}{\left(\frac{\tau_D}{N}\right)s+1}$

Fig. 17. Cascade implementation of PID controller without differentiation of setpoint.

$\tau_D = \tau_2$. However, this requires that we already have some process knowledge.

In the absence of process knowledge, a reasonable lower starting value is

$$\tau_2 = 0.27\,t_p \tag{32}$$

where t_p is the peak time from an initial P-only setpoint experiment. The starting value is derived as follows: if the true model was second-order with a delay θ_0, then from the half rule [6] the effective delay for a first-order with delay model is $\theta = \theta_0 + \tau_2/2$, which gives $\tau_2 = 2(\theta - \theta_0)$. We do not know the original time delay θ_0, but it should be smaller than τ_2 to have a benefit of derivative action. To get a lower value for τ_2 (and be conservative), let us assume $\theta_0 = \tau_2$, which gives $\tau_2 = 0.67\theta$. This value works well also when the process is not dominant second order, because it is only slightly higher than the derivative time 0.5θ recommended to counteract the time delay in a first-order process (e.g., [5]). Next, from Fig. 8 the effective time delay θ is between $0.3t_p$ and $0.5t_p$, so let us use $\theta = 0.4t_p$, and we finally obtain $\tau_2 = 0.67\theta = 0.27t_p$.

When performing the setpoint experiment with the PD-controller, one should monitor the value of t_p (with approximately the same overshoot) and make sure that this it is significantly reduced compared to the P-only setpoint experiment. If this is not the case then the expected benefits of adding derivative action are small.

7.1. Proposed approach for PID control

The procedure for the setpoint experiment is unchanged, expect that we use a PD-controller.

Step 1(D). Use a PD-controller (31) and select a value for the derivative time τ_D. A good value is $\tau_D = \tau_2$ (second time constant), but if this is not known then a lower starting value is $\tau_D = 0.27t_p$, where t_p is the time to reach the peak using a P-only controller. Make sure that (a) the PD controller is set up such that the setpoint is differentiated or (b) record the differentiated output y_D.
Step 2. Adjust K_{c0} to get a setpoint overshoot between 10% and 60% (this step is unchanged, except that we use a PD-controller).
Step 3. Collect data for the setpoint experiment (unchanged).

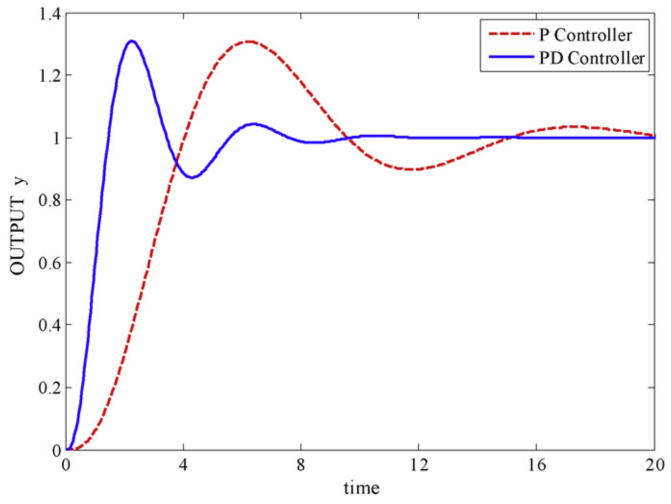

Fig. 18. Setpoint experiment with P and PD controller for 3rd order integrating process (E8). PD controller has filter parameter $N = 10$ and setpoint is differentiated.

The recommended formulas for K_c and τ_I are also unchanged, but note that this assumes that we use the "cascade" PID controller in (29). To use the common "ideal" PID controller

$$c(s) = K_c'\left(1 + \frac{1}{\tau_I' s} + \frac{\tau_D' s}{1 + (\tau_D'/N)s}\right) \tag{33}$$

we must modify the cascade settings by a factor $c = 1 + \tau_D/\tau_I$ by using the following translation formulas (e.g., [6]) (the effect of the filter parameter N has here been neglected, i.e., $N = \infty$ is assumed):

$$K_c' = cK_c, \qquad \tau_I' = c\tau_I, \qquad \tau_D' = \frac{\tau_D}{c} \tag{34}$$

7.2. Example PID control

We consider a third-order integrating process, $g(s) = 1/s(s+1)^2$ (case E8). From the "half rule" of Skogestad [6] this can be approximated by a second-order integrating process $g(s) = e^{-\theta s}/s\,(\tau_2 s + 1)$ with $\theta = 0.5$ and $\tau_2 = 1.5$. Since τ_2 is three times larger than the effective time delay θ, this is a "dominant" second-order process, so we expect significant benefits from adding derivative action.

We now follow the procedure for the setpoint experiment.

Step 1(D). Switch the controller to PD mode. With P-only control we found $t_p = 6.19$ for an overshoot of 30.7% (Tables 1 and 2) and based on this, we select $\tau_D = 0.27 \cdot 6.19 = 1.67$, which is close to the value $\tau_D = 1.5$ recommended by Skogestad [6] for this process. The filter parameter for the PD-controller (31) is set at $N = 10$.
Step 2. Setpoint experiments with overshoots of about 30% are shown in Fig. 18 for the P-controller ($K_{c0} = 0.58$, $\tau_D = 0$) and PD-controller ($K_{c0} = 1.54$, $\tau_D = 1.67$). The time to reach the first peak (t_p)

Table 2
P/PD setpoint experiment and resulting PI/PID controller for process $g(s) = 1/s(s+1)^2$ (E8).

Case	P/PD setpoint experiment					Resulting PI/PID controller								
	K_{c0}	τ_D	Overshoot	t_p	b	K_c	τ_I	M_s	Setpoint			Load disturbance		
									IAE (y)	TV (u)	Overshoot (y)	IAE (y)	TV (u)	Peak value (y)
E8	0.58	0	0.308	6.2	1.0	0.357	15.10	1.75	6.21	0.90	0.35	42.33	1.72	2.92
	1.54	1.67 ($N = 10$)	0.309[a]	2.25	1.0	0.945	5.49	1.49	2.68	1.51	0.081[b]	5.81	1.79	0.81

[a] The PD controller is with D-action on the setpoint.
[b] The PID controller is without D-action on the setpoint.

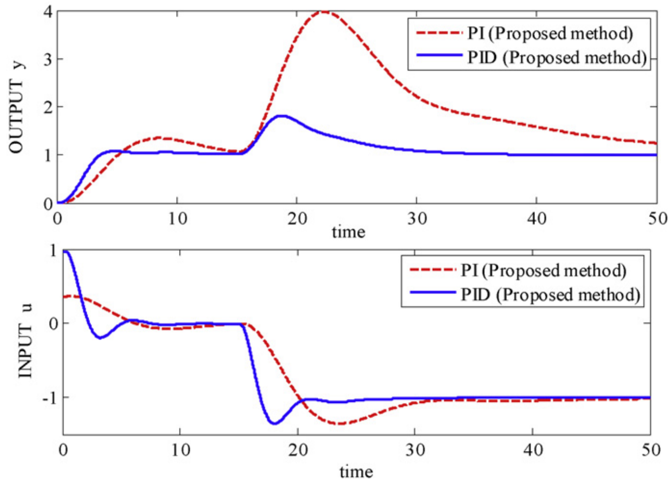

Fig. 19. Response with PI- and PID-controller for 3rd order integrating process (E8). PID controller has filter parameter $N = 10$ and setpoint is *not* differentiated.

is reduced from 6.2 (P-control) to 2.3 (PD-control) which indicates a large benefit of adding derivative action.
Step 3. The results of the setpoint experiment are summarized in Table 2. The resulting cascade PID settings with $F = 1$ are $K_c = 0.945$, $\tau_I = 5.49$ min and $\tau_D = 1.67$ min. If one uses the ideal form (33), then one must translate the values using (34) with $c = 1.30$.

In the closed-loop simulations in Fig. 19, we use the PID structure in Fig. 17 with $N = 10$ and no differentiation of the setpoint. Comparing the closed-loop responses for PI and PID-control in Fig. 19, we note that there is, as expected, a large benefit of adding derivative action for this process. This holds both for the setpoint and disturbance responses. The system is also more robust (sensitivity peak M_s is reduced from 1.75 to 1.49), but the input usage is somewhat larger and the sensitivity to measurement noise (not shown in the simulations) is larger.

8. Industrial verification

The proposed tuning method for PI control has been verified industrially at the Statoil refinery at Mongstad, Norway. They report [16] that the algorithm works well in most cases and they see it as a great advantage to be able to do the tuning in closed loop. The detuning factor was set to $F = 1$ in most cases. They have previously used the SIMC method [6] based on an open-loop step test, but this took longer time than the closed-loop test.

Results for the pressure control loop for the crude prefractionator are shown in Fig. 20 (setpoint experiment) and Fig. 21 (resulting closed-loop PI response). The output y (CV) is the pressure [barg] and the input u (MV) is the valve position [%]. The responses are a bit jerky because of infrequent updates of the pressure measurements.

From the setpoint experiment with $K_{c0} = 35$ we obtain the following data (Fig. 20):

$t_p = 541–516\ \text{s} = 25\ \text{s} = 0.417$ min
$y_0 = 1.805$ barg
$y_s = 1.700$ barg
$y_p = 1.671$ barg
$y_u = 1.741$ barg

and we obtain the output changes

$\Delta y_s = |y_s - y_0| = 0.105$ barg
$\Delta y_p = |y_p - y_0| = 0.134$ barg
$\Delta y_u = |y_u - y_0| = 0.064$ barg

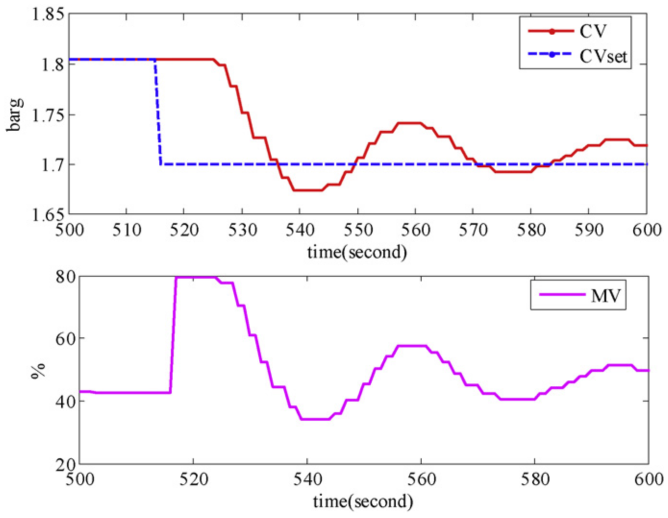

Fig. 20. Industrial verification. Setpoint experiment.

To save time, the experiment was not run to steady-state, but from (10) the predicted steady-state change is

$$\Delta y_\infty = 0.45\,(\Delta y_p + \Delta y_u) = 0.45(0.134 + 0.064) = 0.089\ \text{barg}$$

From this we get

$$\text{overshoot} = \frac{\Delta y_p - \Delta y_\infty}{\Delta y_\infty} = 0.506$$

$$\text{steady-state ratio, } b = \frac{\Delta y_\infty}{\Delta y_s} = 0.847$$

With this overshoot, we find from (15) the gain ratio $A = 0.48$, and with a selected detuning factor $F = 1.2$, the final settings are from (27) and (28):

$$K_c = K_{c0}A/F = 14.0$$

$$\tau_I = \min(0.95, \quad 1.22) = 0.95\ \text{min}$$

The final closed-loop response with PI control is shown in Fig. 21. The response is good, except for some apparent jerky behavior caused by the infrequent the measurement update.

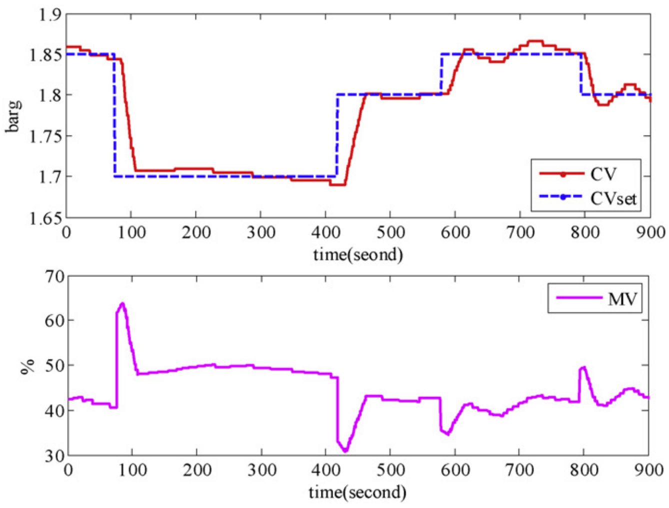

Fig. 21. Industrial verification. Final closed-loop PI response.

9. Discussion: comparison with related approaches

An obvious alternative to the proposed method is a two-step procedure where one first identifies an open-loop model g (with parameters k, θ and τ) from the closed-loop setpoint experiment, and then obtains the PI or PID controller using standard tuning rules (e.g., the SIMC rules [6]). In fact, from (18), we can obtain the following estimate of the process gain k,

$$k = K_{c0}^{-1}\left|\frac{b}{(1-b)}\right| \tag{35}$$

Similarly, from (21), we have the following estimate of the dominant time constant τ,

$$\tau = 0.86A(\text{overshoot})\left|\frac{b}{(1-b)}\right| t_p \tag{36}$$

where A(overshoot) is given in (16). Finally, for the effective time delay θ, we have the following estimate from Case (2):

$$\theta = 0.305t_p \tag{37}$$

One may use (35)–(37) also with other tuning rules, but note these estimates, and in particular A(overshoot) from (16), were derived with the goal of matching the SIMC PI settings. Nevertheless, the estimates (35)–(37) may be useful, for example, for gaining insight or when comparing or combining with information obtained from an open-loop experiment. However, if the goal is to use the setpoint experiment to obtain PI settings, then we recommend the one-step procedure with PI-settings obtained from (27) and (28), rather than the two-step procedure where one first obtains the open-loop model g.

A two-step procedure, based on a closed-loop setpoint experiment with a P-controller, was originally proposed by Yuwana and Seborg [10]. They identified a first-order with delay model by matching the closed-loop setpoint response with a standard oscillating second-order step response that results when the time delay is approximated by a first-order Pade approximation. They identified from the setpoint response the first overshoot, first undershoot and second overshoot, but the method may be modified to not use the second overshoot, as in the present paper. Yuwana and Seborg [10] then used the Ziegler–Nichols [7] tuning rules, which as mentioned in the introduction may give rather aggressive setting. We tried using the SIMC rules instead. This improves the performance, but nevertheless we found (see [11] for details) that the two-step approach of Yuwana and Seborg [10] gives results slightly inferior to the one-step approach proposed in this paper.

Veronesi and Visioli [13] recently published another two-step approach, where the idea is to assess and possibly retune an existing PI controller. From a closed-loop setpoint or disturbance response using the existing PI controller, they identify a first-order with delay model and time constant and use this to assess the closed-loop performance. If the performance is worse that should be expected, the controller is retuned, for example, using the SIMC method. So far the method has only been developed for integrating processes.

In another recent paper, Seki and Shigemasa [14] propose to retune the controller based comparing closed-loop responses obtained with two different controller settings.

10. Conclusion

A simple and new approach for PI controller tuning has been developed. It is based on a single closed-loop setpoint step experiment using a P-controller with gain K_{c0}. The PI-controller settings are then obtained directly from following three data from the setpoint experiment:

- Overshoot = $(\Delta y_p - \Delta y_\infty)/\Delta y_\infty$
- Time to reach overshoot (first peak), t_p
- Relative steady state output change, $b = \Delta y_\infty/\Delta y_s$.

If one does not want to wait for the system to reach steady state, one can use the estimate $\Delta y_\infty = 0.45(\Delta y_p + \Delta y_u)$ where Δy_u is the output change at the first undershoot.

The proposed PI tuning formulas for the proposed "Setpoint Overshoot Method" method are:

$$K_c = \frac{K_{c0}A}{F}$$

$$\tau_I = \min\left(0.86A\left|\frac{b}{(1-b)}\right| t_p,\quad 2.44t_pF\right)$$

where $A = [1.152(\text{overshoot})^2 - 1.607(\text{overshoot}) + 1.0]$.

The factor F is a detuning parameter. A good trade-off between robustness and speed of response is achieved with $F = 1$, but one may use $F > 1$ to get a smoother response with more robustness and less input usage.

The Setpoint Overshoot Method works well for a wide variety of the processes typical for process control, including the standard first-order plus delay processes as well as integrating, high-order, inverse response, unstable and oscillating process. The method gives a PI controller, but for dominant second-order processes where derivative action may give large benefits, one can use a PD-controller in the setpoint experiment, to end up with a PID controller.

Compared to the classical Ziegler–Nichols closed-loop method [7], including its relay tuning variants [3], the proposed overshoot method is faster and simpler to use and also gives better settings in most cases. The new overshoot method is therefore well suited for use in the process industries as has already been verified.

Acknowledgements

Are Skage at the Statoil Mongstad refinery provided the industrial data. Comments from Ivar J. Halvorsen and Vinay Kariwala are gratefully acknowledged.

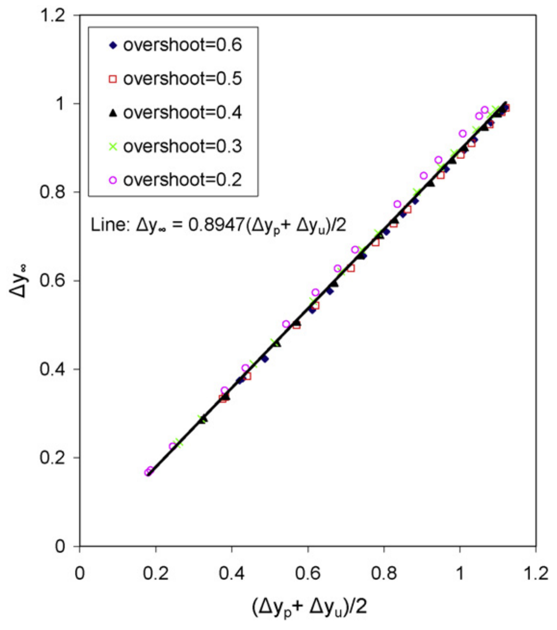

Fig. 22. Relationship between Δy_∞ and $(\Delta y_p + \Delta y_u)/2$ for 15 first-order with delay process with 5 different overshoots.

Appendix A.

The proposed method requires one to obtain the steady-state output change (Δy_∞) for the setpoint response, but it may take some time for the response to settle to the new steady state. To avoid this, we propose to estimate Δy_∞ from only Δy_p (first peak) and Δy_u (undershoot). Yuwana and Seborg (1982) derived an estimate of Δy_∞ that makes use of also the second peak, but achieve sufficient accuracy without this information. Since Δy_∞ must lie between Δy_p and Δy_u, a first try is to consider the average. In Fig. 22 we plot Δy_∞ as a function of the average $[(\Delta y_p + \Delta y_u)/2]$ for 15 first-order with delay process using 5 different overshoot (0.2, 0.3, 0.4, 0.5, 0.6). Somewhat surprisingly, we find for the 75 cases that there is almost a linear relationship with a coefficient of 0.895 corresponding to

$$\Delta y_\infty = \frac{0.895(\Delta y_p + \Delta y_u)}{2} \approx 0.45(\Delta y_p + \Delta y_u)$$

as given in Eq. (10). In Table 3, the correlation has been further tested on the 33 processes from Table 1. For an overshoot of about 30% we find that the deviation is less than 1% in most cases; the main exceptions are for processes with overshoot in the open-loop response (positive zeros; E12, E13, E24) and oscillations (E30, E31) where it underpredicts the b-value by as much as to −18% (E23).

Table 3
%Error in steady-state output Δy_∞ when using $\Delta y_\infty = 0.45(\Delta y_p + \Delta y_u)$.

Case	%Error in Δy_∞ (b value)	
	Overshoot ≈0.30	Overshoot ≈0.60
E1	−0.3	1.2
E2	−0.6	1.0
E3	−2.9	−2.5
E4	−1.1	−1.3
E5	−0.8	0.04
E6	−1.0	−1.2
E7	0.9	6.7
E8	−0.8	−1.0
E9	−0.1	1.2
E10	−0.5	0.9
E11	0.2	1.0
E12	−8.0	−7.3
E13	−13.9	−13.0
E14a	1.1	9.0
E15a	2.1	10.9
E16a	0.4	4.3
E17	−0.4	1.8
E18	0.4	2.8
E19	−0.4	1.8
E20	−0.6	0.8
E21	−0.6	0.8
E22	−0.3	1.7
E23	−18.2	−18.9
E24	−0.3	1.7
E25	−0.9	−1.1
E26	−0.6	1.5
E28	0.9	6.7
E29	0.03	2.9
E30	−6.2	−8.9
E31	−11.4	−11.2
E32	0.3	4.2
E33	−0.4	1.5

Appendix B. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.jprocont.2010.08.003.

References

[1] L.D. Desborough, R.M. Miller, Increasing customer value of industrial control performance monitoring—Honeywell's experience. Chemical Process Control-VI (Tuscon, Arizona, Jan. 2001), AIChE Symposium Series No. 326. vol. 98, USA, 2002.
[2] M. Kano, M. Ogawa, The state of art in advanced process control in Japan, in: IFAC Symposium ADCHEM 2009, July 2009, Istanbul, Turkey, 2009.
[3] A. O'Dwyer, Handbook of PI and PID Controller Tuning Rules, Imperial College Press, 2006.
[4] D.E. Seborg, T.F. Edgar, D.A. Mellichamp, Process Dynamics and Control, 2nd ed., John Wiley & Sons, New York, USA, 2004.
[5] D.E. Rivera, M. Morari, S. Skogestad, Internal model control. 4. PID controller design, Ind. Eng. Chem. Res. 25 (1) (1986) 252–265.
[6] S. Skogestad, Simple analytic rules for model reduction and PID controller tuning, J. Process Control 13 (2003) 291–309.
[7] J.G. Ziegler, N.B. Nichols, Optimum settings for automatic controllers, Trans. ASME 64 (1942) 759–768.
[8] K.J. Åström, T. Hägglund, Automatic tuning of simple regulators with specifications on phase and amplitude margins, Automatica 20 (1984) 645–651.
[9] B.D. Tyreus, W.L. Luyben, Tuning PI controllers for integrator/dead time processes, Ind. Eng. Chem. Res. (1992) 2625–2628.
[10] M. Yuwana, D.E. Seborg, A new method for on-line controller tuning, AIChE J. 28 (3) (1982) 434–440.
[11] M. Shamsuzzoha, S. Skogestad, Internal report with additional data for the setpoint overshoot method, 2010. Available at the home page of S. Skogestad. http://www.nt.ntnu.no/users/skoge/.
[12] W.L. Luyben, Simple method for tuning SISO controllers in multivariable systems, Ind. Eng. Chem. Process Des. Dev. 25 (3) (1986) 654–660.
[13] M. Veronesi, A. Visioli, Performance assessment and retuning of PID controllers for integral processes, J. Process Control 20 (2010) 261–269.
[14] H. Seki, T. Shigemasa, Retuning oscillatory PID control loops based on plant operation data, J. Process Control 20 (2010) 217–227.
[15] T.S. Schei, A method for closed loop automatic tuning of PID controllers, Automatica 20 (3) (1992) 587–591.
[16] A. Skage, Personal email communication, 2010.