PID Tuning
using the SIMC rules

Sigurd Skogestad
NTNU, Trondheim, Norway

---

**PID controller**

- Time domain ("ideal" PID)
  \[ u(t) = u_0 + K_c' \left( e(t) + \frac{1}{\tau_i} \int_0^t e(t^*) dt^* + \tau_D' \frac{de(t)}{dt} \right) \]

- Laplace domain ("ideal"/"parallel" form)
  \[ c(s) = K_c' \left( 1 + \frac{1}{\tau_i s} + \tau_D' s \right) \]

- For our purposes. Simpler with cascade form
  \[ c(s) = K_c \frac{(\tau_I s + 1)(\tau_D s + 1)}{\tau_I s} \]

- Usually \( \tau_D = 0 \). Then the two forms are identical.

- Only two parameters left (\( K_c \) and \( \tau_i \))

- How difficult can it be to tune???
  - Surprisingly difficult without systematic approach!
Tuning of PID controllers

- SIMC tuning rules ("Skogestad IMC")(*)
- Main message: Can usually do much better by taking a systematic approach
- Key: Look at initial part of step response
  - Initial slope: $k' = k/\tau_1$
- One tuning rule! PI-control:
  \[
  K_c = \frac{1}{k'} \cdot \frac{1}{(\theta + \tau_c)} \\
  \tau_I = \min(\tau_1, 4(\tau_c + \theta))
  \]
  - $\tau_c \geq \theta$: desired closed-loop response time (tuning parameter)
  - For robustness select: $\tau_c \geq 0$

(Also reprinted in MIC)
(*) "Probably the best simple PID tuning rules in the world"

Need a model for tuning

- Model: Dynamic effect of change in input $u$ (MV) on output $y$ (CV)
- First-order + delay model for PI-control
  \[
  G'(s) = \frac{k}{\tau_1 s + 1} \cdot e^{-\theta s}
  \]
- Second-order model for PID-control
  \[
  G'(s) = \frac{k}{(\tau_1 s + 1)(\tau_2 s + 1)} \cdot e^{-\theta s}
  \]
  - Recommend: Use second-order model (PID control) only if $\tau_2 \geq \theta$
1. Step response experiment

- Make step change in one $u$ (MV) at a time
- Record the output ($s$) $y$ (CV)

MODEL, Approach 1

- $\theta$: Delay - Time where output does not change
- $\tau_1$: Time constant - Additional time to reach 63% of final change
- $k = \Delta y(\infty)/\Delta u$: Steady-state gain
Step response integrating process

MODEL, Approach 1

2. Model reduction of more complicated model

- Start with complicated stable model on the form

\[ G_0(s) = k_0 \frac{T_1 s + 1}{\tau_1 s + 1} \frac{T_2 s + 1}{\tau_2 s + 1} \ldots e^{-\theta_0 s} \]

- Want to get a simplified model on the form

\[ G(s) = k \left( \frac{1}{\tau_1 s + 1} \frac{1}{\tau_2 s + 1} \right) e^{-\theta s} \]

- Most important parameter is the “effective” delay \( \theta \)
- Use second-order model only if \( \tau_2 > \theta \)
MODEL, Approach 2

**OBTAINING THE EFFECTIVE DELAY $\theta$**

Basis (Taylor approximation):

$$e^{-\theta s} \approx 1 - \theta s \quad \text{and} \quad e^{\theta s} = \frac{1}{1 + \theta s}$$

**Effective delay =**

"true" delay

+ inverse response time constant(s)

+ half of the largest neglected time constant (the "half rule")

(this is to avoid being too conservative)

+ all smaller high-order time constants

The "other half" of the largest neglected time constant is added to $\tau_1$
(or to $\tau_2$ if use second-order model).

MODEL, Approach 2

**Example**

The second-order process

$$g_0(s) = \frac{1}{(1s + 1)(0.6s + 1)}$$

is approximated as a first-order with delay process

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

with

$$k = 1; \quad \tau_1 = 1 + 0.6/2 = 1.3; \quad \theta = 0.6/2 = 0.3;$$
Example 2

\[ g_0(s) = k \frac{(-0.3s + 1)(0.08s + 1)}{(2s + 1)(1s + 1)(0.4s + 1)(0.2s + 1)(0.05s + 1)^3} \]

is approximated as a first-order delay process with

\[ \tau_1 = 2 + 1/2 = 2.5 \]
\[ \theta = 1/2 + 0.4 + 0.2 + 3 \cdot 0.05 + 0.3 - 0.08 = 1.47 \]

or as a second-order delay process with

\[ \tau_1 = 2 \]
\[ \tau_2 = 1 + 0.4/2 = 1.2 \]
\[ \theta = 0.4/2 + 0.2 + 3 \cdot 0.05 + 0.3 - 0.08 = 0.77 \]

Derivation of SIMC-PID tuning rules

- PI-controller (based on first-order model)

\[ c(s) = K_c \left( 1 + \frac{1}{\tau_I s} \right) = K_c \frac{\tau_I s + 1}{\tau_I s} \]

- For second-order model add D-action.
  
  For our purposes, simplest with the “series” (cascade) PID-form:

\[ c(s) = K_c \frac{(\tau_I s + 1)(\tau_D s + 1)}{\tau_I s} \]  \hspace{1cm} (1)
Basis: Direct synthesis (IMC)

\[ y = T \, y_s, \quad T(s) = \frac{gc}{1+gc} \]

Idea: Specify desired response: \( \left( \frac{y}{y_s} \right)_{\text{desired}} = T \)

and from this get the controller. ....... Algebra:

\[ c = \frac{1}{g} \cdot \frac{1}{T} - 1 \]

NOTE: Setting the steady-state gain = 1 in T will result in integral action in the controller!
IMC Tuning = Direct Synthesis

Algebra:

- Controller: \( c(s) = \frac{1}{g(s)} \cdot \frac{1}{\| \text{process} \|} - 1 \)
- Consider second-order with delay plant: \( g(s) = k \frac{e^{-\theta_s}}{(\tau_1 s + 1)(\tau_2 s + 1)} \)
- Desired first-order setpoint response: \( \left( \frac{y}{y_{\text{desired}}} \right) = \frac{1}{\tau_c s + 1} e^{-\theta_s} \)
- Gives a “Smith Predictor” controller: \( c(s) = \frac{\tau_{2s} + 1}{k(\tau_c s + 1 + e^{-\theta_s})} \)
- To get a PID-controller use \( e^{-\theta_s} \approx 1 - \theta_s \) and derive
  \[ c(s) = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{k(\tau_c s + 1)} \]
  which is a cascade form PID-controller with
  \[ K_c = \frac{1}{k\tau_c + \theta}; \quad \tau_I = \tau_1; \quad \tau_D = \tau_2 \]
- \( \tau_c \) is the sole tuning parameter

Integral time

- Found: Integral time = dominant time constant (\( \tau_I = \tau_1 \))
- Works well for setpoint changes
- Needs to be modified (reduced) for integrating disturbances

Example. “Almost-integrating process” with disturbance at input:
\( G(s) = e^{s}(30s + 1) \)
Original integral time \( \tau_I = 30 \) gives poor disturbance response
Try reducing it!
Integral Time

\[ \tau_I = \tau_c \]

Reduce \( \tau_I \) to this value:
\[ \tau_I = 4(\tau_c + \theta) = 8 \theta \]

Too large integral time: Poor disturbance rejection
Too small integral time: Slow oscillations

---

Integral time

- Want to reduce the integral time for “integrating” processes, but to avoid “slow oscillations” we must require:

\[ \tau_I \geq 4(\tau_C + \theta) \]

- Derivation:

\[ G(s) = k \frac{e^{-\tau}/s}{\tau_c s + 1} \approx \frac{k}{s} \quad \text{where} \quad k' = \frac{k}{\tau} \]

Closed-loop poles:
\[ 1 + GCC = 0 \Rightarrow 1 + \frac{k}{s} K_C \left(1 + \frac{1}{\tau_C s}\right) = 0 \Rightarrow \tau_C s^2 + k' K_C \tau_C s + k' K_C = 0 \]

To avoid oscillations we must not have complex poles:
\[ B^2 - 4AC \geq 0 \Rightarrow k'^2 K_C^2 \tau_C^2 - 4k' K_C \tau_C \geq 0 \Rightarrow k' K_C \tau_C \geq 4 \Rightarrow \tau_C \geq \frac{4}{k' K_C} \]

Inverted SIMC-rule for \( K_C = \frac{1}{k' \tau_C + \theta} \) then gives
\[ \tau_I \geq 4(\tau_c + \theta) \]
Conclusion: SIMC-PID Tuning Rules

For cascade form PID controller:

\[ K_c = \frac{1}{K_c + \theta} = \frac{1}{\tau_c + \theta} \]

(1)

\[ \tau_I = \min\{\tau_1, \frac{4}{K_c}\} = \min\{\tau_1, 4(\tau_c + \theta)\} \]

(2)

\[ \tau_D = \tau_2 \]

(3)

Derivation:

1. First-order setpoint response with response time \( \tau_c \) (IMC-tuning = "Direct synthesis")

2. Reduce integral time to get better disturbance rejection for slow or integrating process (but avoid slow cycling \( \Rightarrow \tau_I \geq \frac{4}{K_c} \))

One tuning parameter: \( \tau_c \)

SIMC-tunings

Some special cases

<table>
<thead>
<tr>
<th>Process</th>
<th>( g(s) )</th>
<th>( K_c )</th>
<th>( \tau_I )</th>
<th>( \tau_D )</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \min{\tau_1, 4(\tau_c + \theta)} )</td>
<td>-</td>
</tr>
<tr>
<td>Second-order, eq.(4)</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \min{\tau_1, 4(\tau_c + \theta)} )</td>
<td>( \tau_2 )</td>
</tr>
<tr>
<td>Pure time delay(^{(1)})</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>0 ( (*) )</td>
<td>-</td>
</tr>
<tr>
<td>Integrating(^{(2)})</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>-</td>
</tr>
<tr>
<td>Integrating with lag ( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \tau_2 )</td>
</tr>
<tr>
<td>Double integrating(^{(3)})</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
<td>( \frac{1}{\tau_c + \theta} )</td>
</tr>
</tbody>
</table>

Table 1: SIMC PID-settings (23)-(25) for some special cases of (4) (with \( \tau_c \) as a tuning parameter).

(1) The pure time delay process is a special case of a first-order process with \( \tau_1 = 0 \).

(2) The integrating process is a special case of a first-order process with \( \tau_1 \to \infty \).

(3) For the double integrating process, integral action has been added according to eq.(27).

(4) The derivative time is for the series form PID controller in eq.(1).

\( (*) \) Pure integral controller \( c(s) = \frac{K_c}{s} \) with \( K_I = \frac{\tau_c}{\tau_1} \).

One tuning parameter: \( \tau_c \)
6.3 Ideal PID controller

The settings given in this paper \((K_c, \tau_1, \tau_D)\) are for the series (cascade, “interacting”) form PID controller in (1). To derive the corresponding settings for the ideal (parallel, “non-interacting”) form PID controller

\[
\text{Ideal PID: } \quad C(s) = K_c \left(1 + \frac{1}{\tau_c s} + \tau_D s\right) = \frac{K_c}{\tau_c s} (\tau_c s^2 + 1) \quad (35)
\]

we use the following translation formulas

\[
K_c = K_c \left(\frac{1 + \tau_D}{\tau_D}\right); \quad \tau_c = \tau_c \left(1 + \frac{\tau_D}{\tau_D}\right); \quad \tau_D = \frac{\tau_c}{1 + \frac{n}{\theta}} \quad (36)
\]

The SIMC-PID series settings in (29)-(31) then correspond to the following SIMC ideal-PID settings \((\tau_c = \theta)\):

\[
\begin{align*}
\tau_1 \leq 8\theta: & \quad K_c = 0.5 \frac{(\tau_c - 1)}{\theta}; \quad \tau_c = \tau_1 + \tau_c; \quad \tau_D = \frac{\tau_1}{1 + \frac{n}{\theta}} \\
\tau_1 \geq 8\theta: & \quad K_c = 0.5 \frac{\tau_1}{k \theta} \left(1 + \frac{\tau_1}{\theta}\right); \quad \tau_c = 8\theta + \tau_1; \quad \tau_D = \frac{\tau_1}{1 + \frac{n}{\theta}} \quad (37)
\end{align*}
\]

We see that the rules are much more complicated when we use the ideal form.

Example. Consider the second-order process \(g(s) = e^{-\theta}/(s + 1)^2\) \((E9)\) with the \(k = 1, \theta = 1, \tau_1 = 1\) and \(\tau_2 = 1\). The series-form SIMC settings are \(K_c = 0.5, \tau_1 = 1\) and \(\tau_2 = 1\). The corresponding settings for the ideal PID controller in (35) are \(K_c = 1, \tau_c = 2\) and \(\tau_D = 0.5\). The robustness margins with these settings are given by the first column in Table 2.

---

**Selection of tuning parameter \(\tau_c\)**

Two main cases

1. **TIGHT CONTROL (\(\tau_c\) small):** Want “fastest possible control” subject to having good robustness
   - Want tight control of active constraints (“squeeze and shift”)

2. **SMOOTH CONTROL (\(\tau_c\) large):** Want “slowest possible control” subject to acceptable disturbance rejection
   - Want smooth control if fast setpoint tracking is not required, for example, levels and unconstrained (“self-optimizing”) variables
TIGHT CONTROL

TUNING FOR FAST RESPONSE WITH GOOD ROBUSTNESS

SIMC: \( \tau_c = \theta \)

Gives:

\[
K_c = \frac{0.5 \tau_1}{\theta} = \frac{0.5}{\frac{k}{k'} \cdot \theta}
\]

\( \tau_I = \min \{ \tau_1, 8\theta \} \)

\( \tau_D = \tau_2 \)

Try to memorize!

Gain margin about 3

<table>
<thead>
<tr>
<th>Process p(s)</th>
<th>( \zeta )</th>
<th>( \omega_n )</th>
<th>( \zeta' )</th>
<th>( \omega'_n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller gain, ( K_c )</td>
<td>3.14</td>
<td>2.06</td>
<td>0.14°</td>
<td>46°</td>
</tr>
<tr>
<td>Integral time, ( \tau_c )</td>
<td>2.34</td>
<td>1.59</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Gain margin (GM)</td>
<td>1.59</td>
<td>1.70</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Phase margin (PM)</td>
<td>1.57</td>
<td>1.49</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>Phase crossover frequency, ( \omega_n )</td>
<td>1.56</td>
<td>1.49</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>Sensitivity peak, (</td>
<td>S</td>
<td>)</td>
<td>1.59</td>
<td>1.70</td>
</tr>
<tr>
<td>Complementary sensitivity peak, (</td>
<td>1/S</td>
<td>)</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Phase crossover frequency, ( \omega_n' )</td>
<td>1.57</td>
<td>1.49</td>
<td>0.50</td>
<td>0.55</td>
</tr>
</tbody>
</table>

Table 1: Robustness margins for first-order and integrating delay process using SIMC-tunings in (5) and (6) (\( \tau_c = \theta \)). The same margins apply to second-order processes if we choose \( \tau_D = \tau_c \).

TIGHT CONTROL

Typical closed-loop SIMC responses with the choice \( \tau_c = \theta \)

Figure 1: Responses using SIMC settings for the five time delay processes in Table 3 (\( \tau_c = \theta \)).

Unit step response at \( t = 0 \); Unit load disturbance at \( t = 20 \).

Simulations are without derivative action on the setpoint.

Parameter values: \( \theta = 3, k = 1, k' = 1, \theta = 1, \theta \).
Example E2 (Further continued) We want to derive PI and PID settings for the process

\[ y(s) = \frac{(-0.3s + 1)(0.08s + 1)}{(2s + 1)(3s + 1)(0.4s + 1)(0.2s + 1)(0.05s + 1)} \]

using the SIMC tuning rules with the "default" recommendation \( \tau_c = \theta \). From the closed-loop step response, we obtain in a previous example a first-order model with parameters \( k = 0.999, \theta = 1.67, \tau_1 = 3.00 \) (2.10). The resulting SIMC PI settings with \( \tau_c = \theta = 1.67 \) are

\[ P_{12} : \quad K_c = 0.904, \quad \tau_f = 3. \]

From the full-order model \( g(s) \) and the half rule, we obtained in a previous example a first-order model with parameters \( k = 1, \theta = 1.47, \tau_1 = 2.5 \). The resulting SIMC PI settings with \( \tau_c = \theta = 1.47 \) are

\[ P_{12, \text{ half}} : \quad K_c = 0.850, \quad \tau_f = 2.5. \]

From the full-order model \( g(s) \) and the half rule, we obtained a second-order model with parameters \( k = 1, \theta = 0.77, \tau_2 = 2, \tau_1 = 1.2 \). The resulting SIMC PID settings with \( \tau_c = \theta = 0.77 \) are

\[ \text{Series PID} : \quad K_c = 1.299, \quad \tau_1 = 2, \quad \tau_D = 1.2. \]

The corresponding settings with a more common ideal (parallel form) PID controller are obtained by computing \( f = 1 + \tau_D/\tau_1 = 1.60 \), and we have

Ideal PID : \[ K_c' = K_c f = 1.60, \quad \tau_1' = \tau_1 f = 3.2, \quad \tau_D' = \tau_D f = 0.75. \quad (5.30) \]

Fig. 5.6 Closed-loop responses for process E2 using SIMC PI and PID-tunings with \( \tau_c = \theta \). Setpoint change at \( r = 0 \) and input (load) disturbance at \( r = 10 \). For the PID controller, D-action is only on the feedback signal, i.e., not on the setpoint \( y_0 \).
**SIMC: Tuning parameter ($\tau_c$) correlates nicely with robustness measures**

**$M_s$**

1.6

**GM**

3

**PM**

$\theta = \frac{\Delta \theta}{\theta}$

60°

**DM**

$\theta = \frac{\Delta \theta}{\theta}$


**SMOOTH CONTROL**

**Tuning for smooth control**

- Tuning parameter: $\tau_c = $ desired closed-loop response time
- Selecting $\tau_c=0$ if we need “tight control” of $y$.
- Other cases: “Smooth control” of $y$ is sufficient, so select $\tau_c > 0$ for
  - slower control
  - smoother input usage
  - less disturbing effect on rest of the plant
  - less sensitivity to measurement noise
  - better robustness
- Question: Given that we require some disturbance rejection.
  - What is the largest possible value for $\tau_c$?
  - Or equivalently: What is the smallest possible value for $K_c$?
  - ANSWER: $K_{c,\text{min}} = \frac{u_d}{y_{\text{max}}}$
    - $u_d = $ input change to reject disturbance (steady-state)
    - $y_{\text{max}} = $ maximum desired output deviation
    - May obtain $u_d$ from historical data

Proof: Imagine using P-control only. Then we get at steady-state $u = K_c y_d$, where $y_d$ is the steady-state offset. With I-action we have no offset but the peak value of $y$ will be close to $y_d$. More detailed proof: S. Skogestad, "Tuning for smooth PID control with acceptable disturbance rejection", J. Eng. Chem. Res. 46 (23), 7677-7682 (2006).
**Conclusion PID tuning**

SIMC tuning rules

\[ K_c = \frac{1}{k} \cdot \left( \frac{1}{\theta + \tau_c} \right) \]

\[ \tau_I = \min(\tau_1, 4(\tau_c + \theta)) \]

1. **Tight control:** Select \( \tau_c = \theta \) corresponding to

\[ K_{c,\text{max}} = \frac{0.5}{k\theta} \]

2. **Smooth control:** Select \( K_c \). \( K_{c,\text{min}} = \left| \frac{u_0}{y_{\text{max}}} \right| \)

\( u_0 \) input change required to reject disturbance
\( y_{\text{max}} \) largest allowed output change

Note: Having selected \( K_c \) (or \( \tau_c \)), the integral time \( \tau_c \) should be selected as given above

3. **Derivative time:** Only for dominant second-order processes
Level control

- Level control often causes problems
- Typical story:
  - Level loop starts oscillating
  - Operator detunes by decreasing controller gain
  - Level loop oscillates even more
  - ......
- ???
- Explanation: Level is by itself unstable and requires control.

Level control: Can have both fast and slow oscillations

- Slow oscillations ($K_c$ too low): $P > 3\tau_I$
- Fast oscillations ($K_c$ too high): $P < 3\tau_I$

Here: Consider the less common slow oscillations
How avoid oscillating levels?

- Simplest: Use P-control only (no integral action)
- If you insist on integral action, then make sure the controller gain is sufficiently large
- If you have a level loop that is oscillating then use Sigurds rule (can be derived):

  To avoid oscillations, increase $K_c \cdot \tau_i$ by factor
  \[ f = 0.1 \cdot \frac{(P_0/\tau_0)^2}{1/\tau^2} \]
  where
  - $P_0$ = period of oscillations [s]
  - $\tau_0$ = original integral time [s]
  - $0.1 \approx 1/\pi^2$

Case study oscillating level

- We were called upon to solve a problem with oscillations in a distillation column
- Closer analysis: Problem was oscillating reboiler level in upstream column
- Use of Sigurd’s rule solved the problem
APPLICATION: RETUNING FOR INTEGRATING PROCESS

To avoid “slow” oscillations the product of the controller gain and integral time should be increased by factor $f \approx 0.1 \left( \frac{P_0}{\tau_1} \right)^2$.

Real Plant data:

Period of oscillations $P_0 = 0.85 \times 60 \text{ sec} = 51 \text{ min} \Rightarrow f = 0.1 \times (51/1)^2 \approx 260$