PID Tuning using the SIMC rules

Sigurd Skogestad

NTNU, Trondheim, Norway

MODEL

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

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

- Second-order model for PID-control

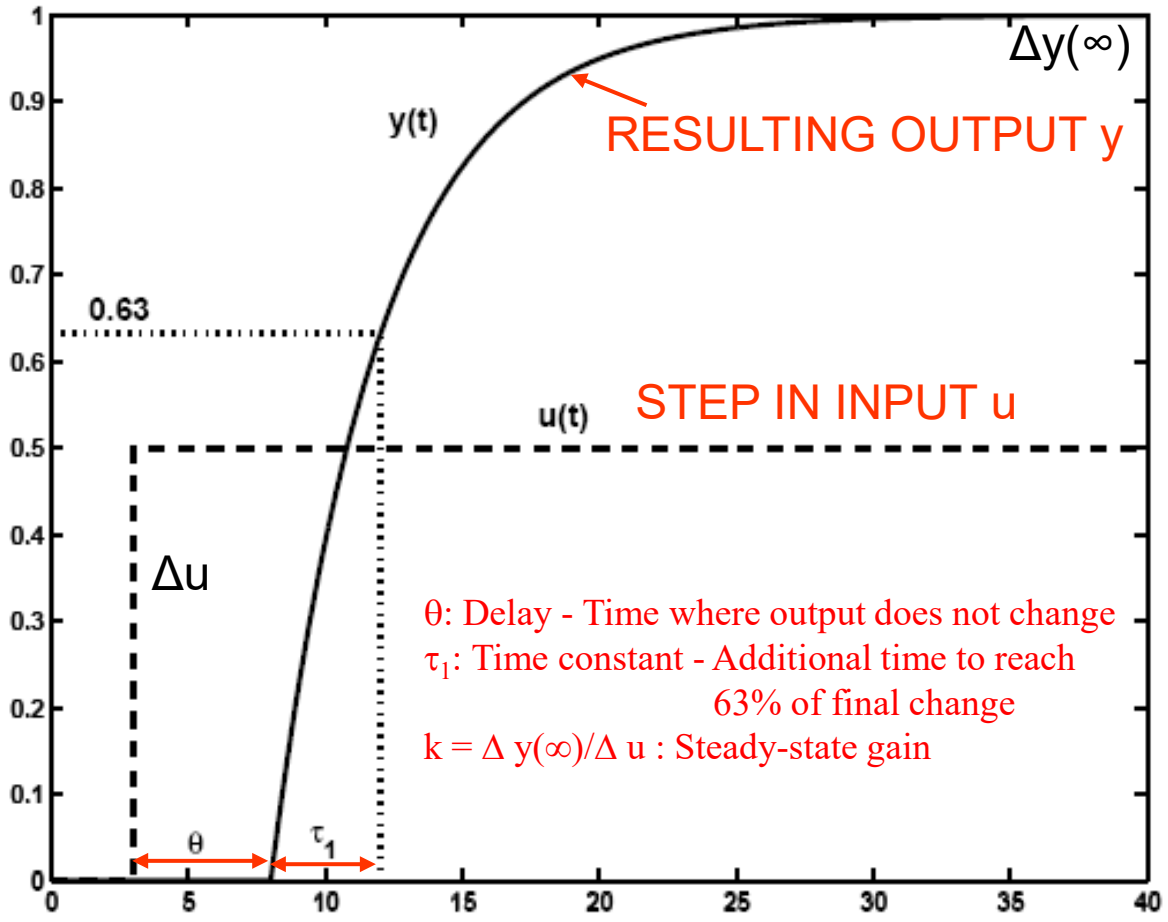
$$G(s) = \frac{k}{(\tau_1 s+1)(\tau_2 s+1)}\ e^{-\theta s}$$

  - Recommend: Use second-order model (PID control) only if ¿$_2$>μ

1. Step response experiment

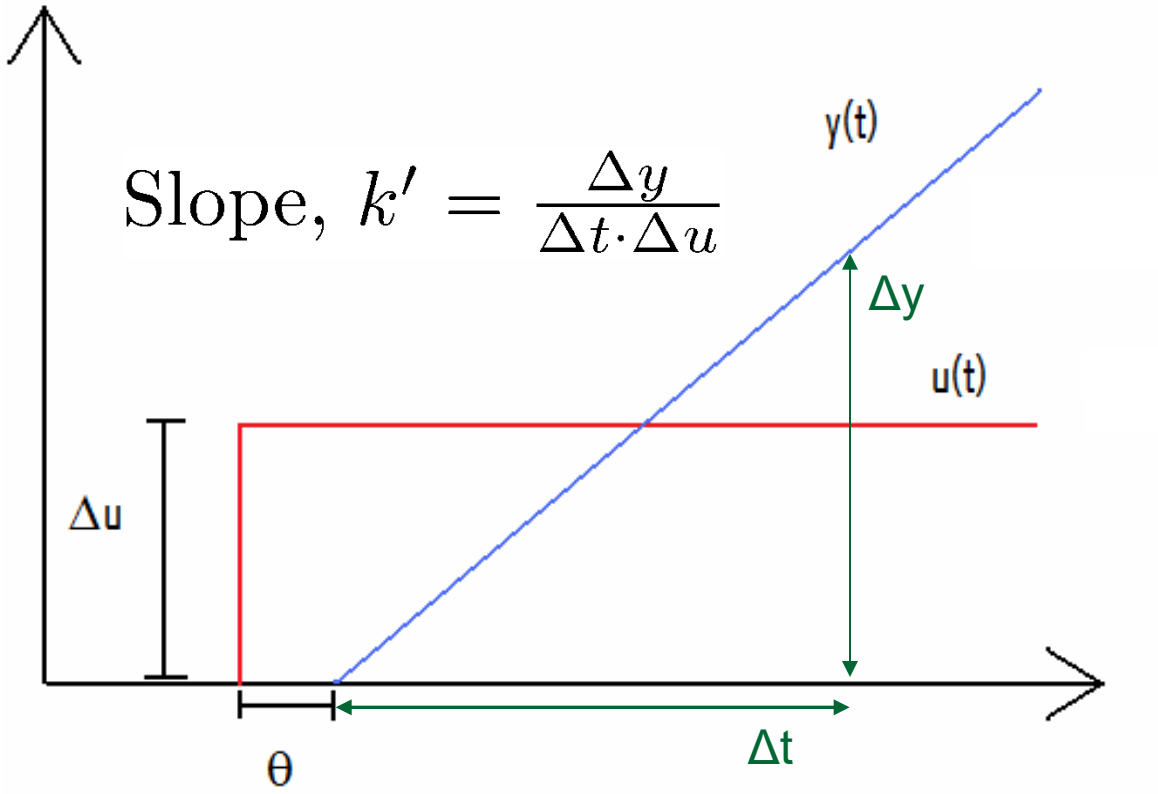
- Make step change in one u (MV) at a time
- Record the output (s) y (CV)

Step response first-order process



If the output y keeps ramping (with little sign of flattening) after about 4 times the delay θ then you can use an integrating process model (so you don't need τ_1).

Step response integrating process



Slope k' = «integrating process gain»

2. Model reduction of more complicated model

- Start with complicated stable model on the form

$$G_0(s) = k_0 \frac{(T_{10}s+1)(T_{20}s+1)\cdots}{(\tau_{10}s+1)(\tau_{20}s+1)\cdots} e^{-\theta_0 s}$$

- Want to get a simplified model on the form

$$G(s) = \frac{k}{(\tau_1 s+1)(\tau_2 s+1)} e^{-\theta s}$$

- Most important parameter is the “effective” delay θ
- Use second-order model only if ¿$_2$>μ

MODEL, Approach 2

OBTAINING THE EFFECTIVE DELAY θ

Basis (Taylor approximation):

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

Effective delay =

"true" delay

+ inverse reponse 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 τ_1 (or to τ_2 if use second-order model).

Details:

- **Half rule**: the largest neglected (denominator) time constant (lag) is distributed evenly to the effective delay and the smallest retained time constant.

In summary, let the original model be in the form

$$\frac{\prod_j \left(-T_{j0}^{\text{inv}} + 1\right)}{\prod_i \tau_{i0} s + 1} e^{-\theta_0 s} \tag{9}$$

where the lags τ_{i0} are ordered according to their magnitude, and $T_{j0}^{\text{inv}} > 0$ denote the inverse response (negative numerator) time constants. Then, according to the half-rule, to obtain a first-order model $e^{-\theta s}/(\tau_1 s + 1)$, we use

$$\tau_1 = \tau_{10} + \frac{\tau_{20}}{2}; \qquad \theta = \theta_0 + \frac{\tau_{20}}{2} + \sum_{i \geqslant 3} \tau_{i0} + \sum_j T_{j0}^{\text{inv}} + \frac{h}{2} \tag{10}$$

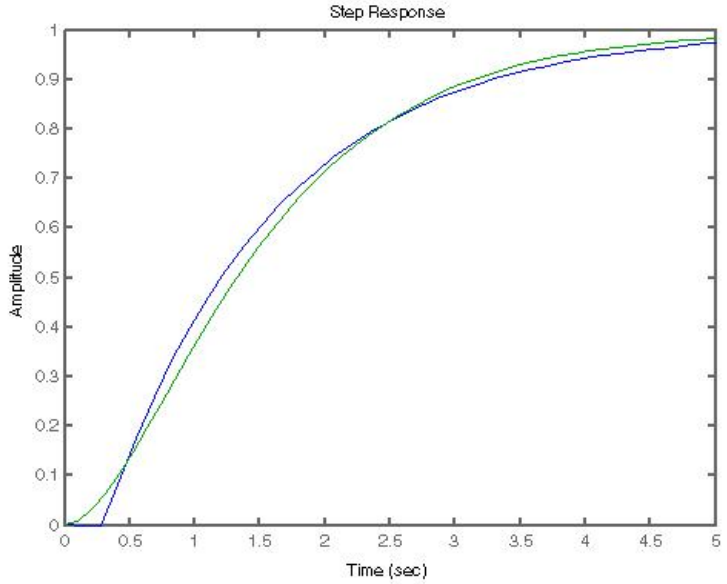
and, to obtain a second-order model (4), we use

$$\tau_1 = \tau_{10}; \qquad \tau_2 = \tau_{20} + \frac{\tau_{30}}{2}; \qquad \theta = \theta_0 + \frac{\tau_{30}}{2} + \sum_{i \geqslant 4} \tau_{i0} + \sum_j T_{j0}^{\text{inv}} + \frac{h}{2} \tag{11}$$

where h is the sampling period (for cases with digital implementation).

The main basis for the empirical half-rule is to maintain the robustness of the proposed PI- and PID-tuning rules, as is justified by the examples later.

Example 1



The second-order process

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

Half rule

is approximated as a first-order with delay process

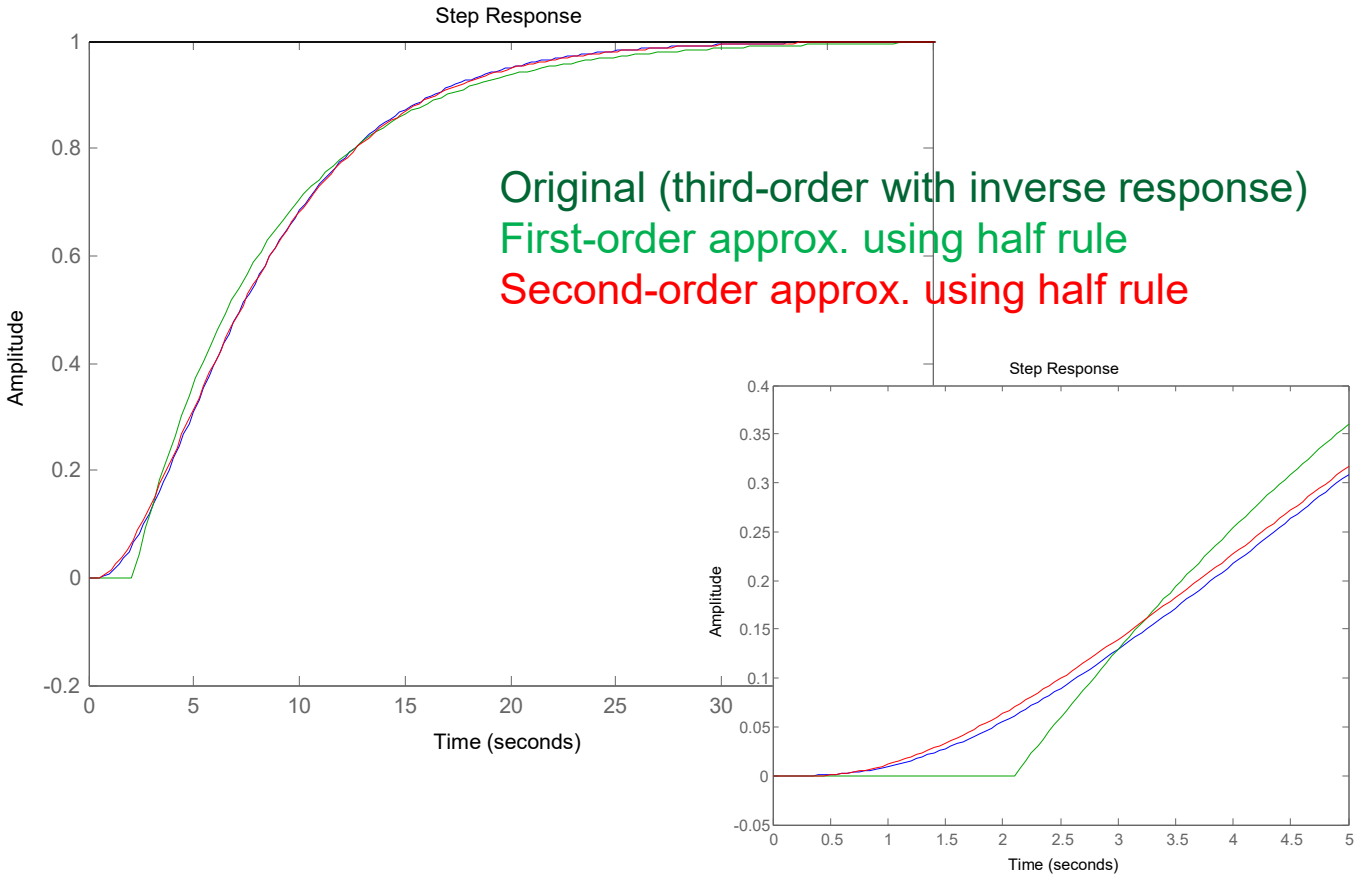
$$g(s) = k\frac{e^{-\theta s}}{\tau_1 s + 1}$$

with

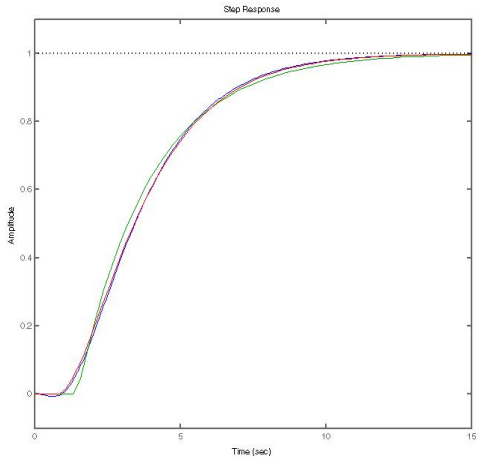
$$k = 1; \quad \tau_1 = 1 + 0.6/2 = 1.3; \quad \theta = 0.6/2 = 0.3;$$

Example 2

```
s=tf('s')
g=(-0.1*s+1)/[(5*s+1)*(3*s+1)*(0.5*s+1)]
g1 = exp(-2.1*s)/(6.5*s+1)
g2 = exp(-0.35*s)/[(5*s+1)*(3.25*s+1)]
step(g,g1,g2)
```



MODEL, Approach 2



Example 3

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

half rule

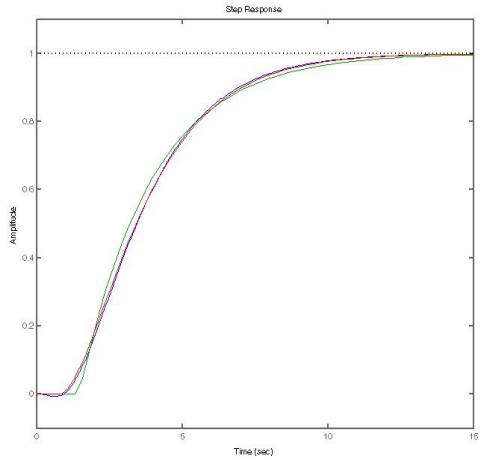
is approximated as a first-order delay process with

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

or as a second-order delay process with

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

Comment: The subtraction of T0=0.08 from the effective delay follows from the approximation (0.08s+1)/(0.2s+1) $\approx \frac{1}{(0.2-0.08)s+1}$ (rule T3).
Alternatively, we could have used the approximation (0.08s+1)/(0.05s+1) ≈ 1 (rule T1b) which would reduce the effective delay by 0.05 (instead of 0.08). In any case, it only has a small effect om the effective delay, so it does not matter much for the final result.



Example 3

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

half rule

is approximated as a first-order delay process with

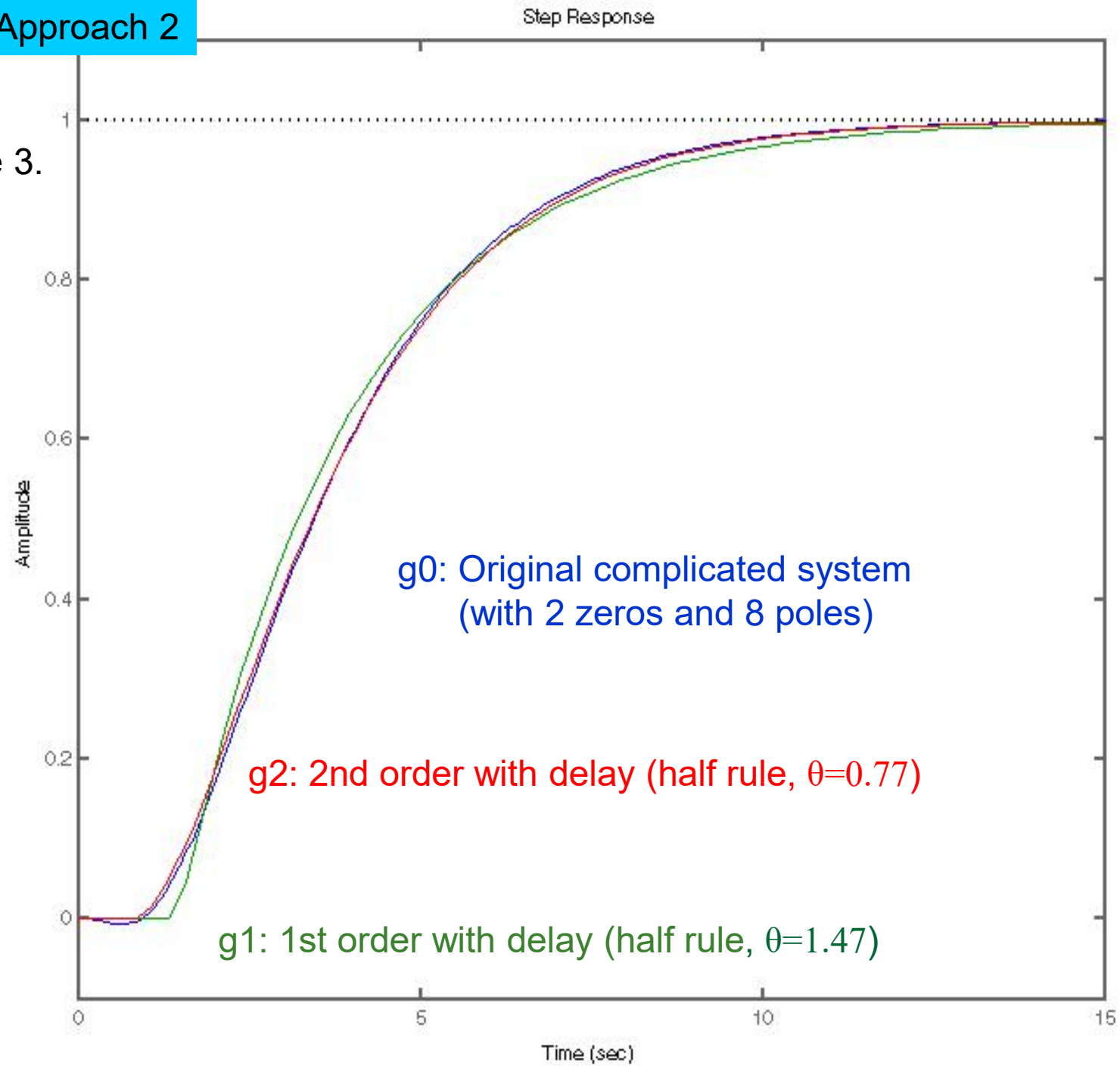
$$\tau_1 = 2 + 1/2 = 2.5$$
$$\theta = 1/2 + 0.4 + 0.2 + 3 \cdot 0.05 + 0.3 - 0.08 = 1.47$$

or as a second-order delay process with

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

MODEL, Approach 2

Example 3.



Example 4. Integrating process

$$g_0(s) = \frac{k'}{s(\tau_{20}s+1)}$$

Half rule gives

$$g(s) = \frac{k'e^{-\theta s}}{s} \text{ with } \theta = \frac{\tau_{20}}{2}$$

Proof:

Note that integrating process corresponds to an infinite time constant

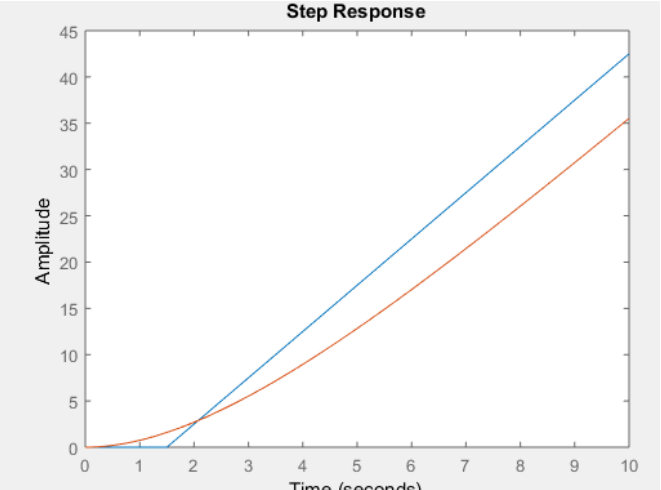
Write

$$g_0(s) = \frac{k'\tau_1}{\tau_1 s(\tau_{20}s+1)} = \frac{k'\tau_1}{(\tau_1 s+1)(\tau_{20}s+1)} \text{ where } \tau_1 \to \infty$$

and then apply half rule as normal, noting that $\tau_1 + \frac{\tau_{20}}{2} \approx \tau_1$:

$$g(s) \approx \frac{k'\tau_1 e^{-\frac{\tau_{20}}{2}s}}{(\tau_1+\frac{\tau_{20}}{2})s} = k'\frac{e^{-\frac{\tau_{20}}{2}s}}{s}$$

Example. g0 = 5/(s*(3*s+1)),
g = 5*exp(-1.5*s)/s,
step(g,g0,10)



Doesn't look so good
But it's OK

Approximation of LHP-zeros (you are not expected to remember this)

$$\frac{T_0 s+1}{\tau_0 s+1} \approx \begin{cases} T_0/\tau_0 & \text{for } T_0 \geq \tau_0 \geq \tau_c \quad \text{(Rule T1)}, \\ T_0/\tau_c & \text{for } T_0 \geq \tau_c \geq \tau_0 \quad \text{(Rule T1a)}, \\ 1 & \text{for } \tau_c \geq T_0 \geq \tau_0 \quad \text{(Rule T1b)}, \\ T_0/\tau_0 & \text{for } \tau_0 \geq T_0 \geq 5\tau_c \quad \text{(Rule T2)}, \\ \frac{(\tilde{\tau}_0/\tau_0)}{(\tilde{\tau}_0 - T_0)s+1} & \text{for } \tilde{\tau}_0 \stackrel{\text{def}}{=} \min(\tau_0, 5\tau_c) \geq T_0 \quad \text{(Rule T3)}. \end{cases}$$

τ_c = desired closed-loop time constant

Example E3. *For the process (Example 4 in (Astrom* et al. *1998))*

g0

$$g_0(s) = \frac{2(15s+1)}{(20s+1)(s+1)(0.1s+1)^2} \qquad (13)$$

we first introduce from Rule T2 the approximation

$$\frac{15s+1}{20s+1} \approx \frac{15s}{20s} = 0.75$$

(Rule T2 applies since $T_0 = 15$ is larger than 5θ, where θ is computed below). Using the half rule, the process may then be approximated as a first-order time delay model with

g1

$$k = 2 \cdot 0.75 = 1.5; \quad \theta = 0.1 + \frac{0.1}{2} = 0.15; \quad \tau_1 = 1 + \frac{0.1}{2} = 1.05$$

or as a second-order time delay model with

g2

$$k = 1.5; \quad \theta = \frac{0.1}{2} = 0.05; \quad \tau_1 = 1; \quad \tau_2 = 0.1 + \frac{0.1}{2} = 0.15$$

PID-controller (from 2nd order model) will give performance improvement because $\tau_2 > \theta$

Normally, we should approximate T_0 by a "close-by" τ_0.

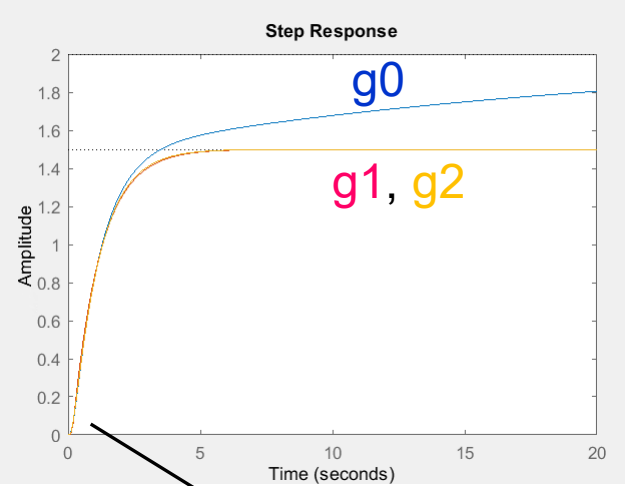
- BUT: The goal is to use the model for control purposes, so we should keep (i.e., not approximate) the τ which is closest to the desired τ_c.

In Example E3, we have two possible values for τ_0, namely 20 and 1. Since T_0=15, it seems clear that we should select the closest $\tau_0 = 20$ and use rule T2.

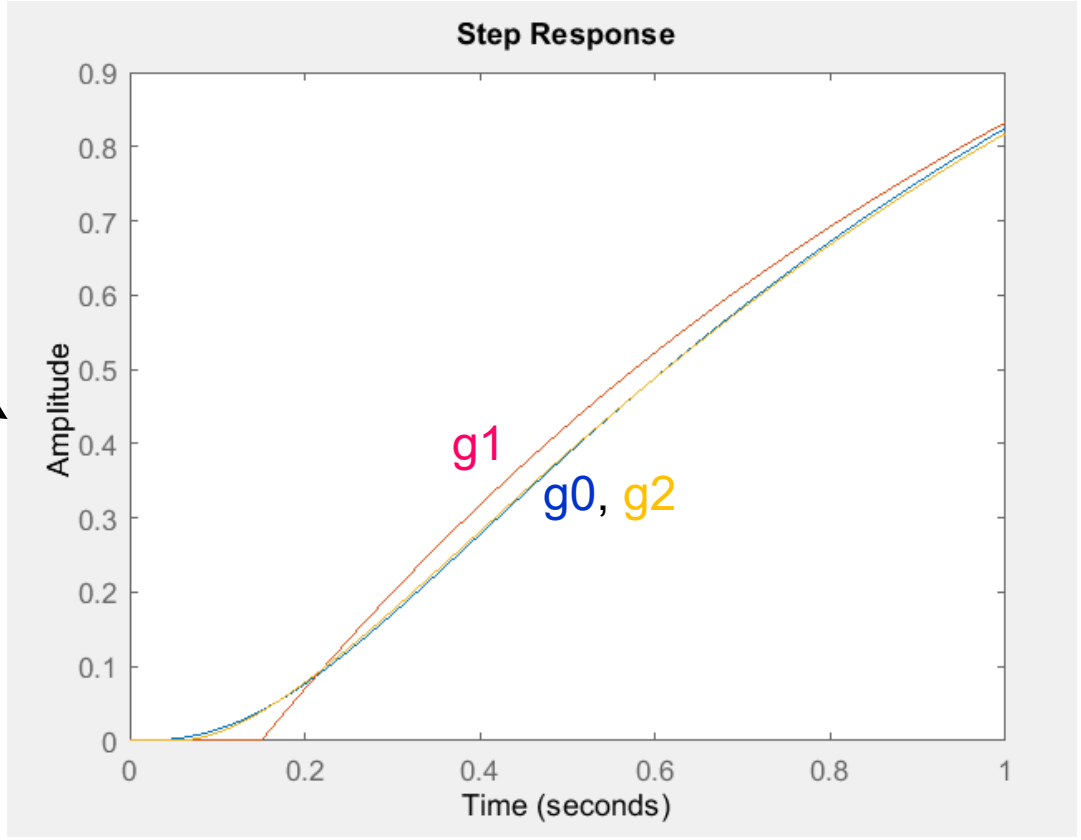
- But what if T_0=2, maybe selecting $\tau_0 = 1$ is better (and using rule T1)?
- No, this is not clear. Since $\tau_c = \theta$ is between 0.05 (PID) and 0.15 (PI), we may want to keep $\tau = 1$ which is closest to τ_c, that is, also in this case select $\tau_0 = 20$ (and use rule T2)
- This may seem surprising, but. in any case, it turns out that it will not matter very much for the PI/PID-tunings for this example (try!), because k/tau1 (and thus Kc) will not change much and because tauI = min(tau,4(tauc+theta)).
- Of course, if T_0 gets very close to 1, then we should select $\tau_0 = 1$.

Generally, the LHP-zeros approximation rules results in acceptable (robust) PI/PID-settings, but not necessarily the "optimal" settings.
See: Exam 2022, Problem 1 and Problem 5d

Step response (without control)



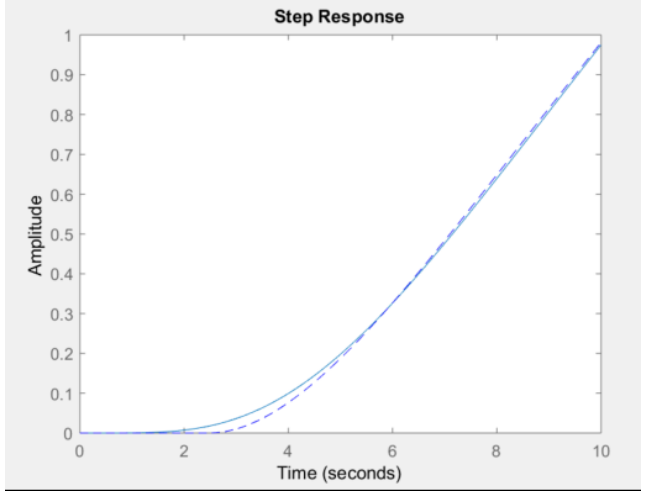
g0 = 2*(15*s+1)/((20*s+1)*(s+1)*(0.1*s+1)^2)
g1=1.5*exp(-0.15*s)/(1.05*s+1)
g2=1.5*exp(-0.05*s)/((s+1)*(0.15*s+1))
step(g0,g1,g2,1)



Note: It's the initial response that matters for feedback control (time from 0 to about 5*tauc)

Alternative: Obtain model from data numerically. Here: use procest (matlab)- but procest seems not to be reliable when I study other cases

```
% We generate some artifical data from a high-order model
s=tf('s')
G = 3*(1-0.1*s)/((10*s+1)*(3*s+1)*((s+1)^3))

Ts=1;     % sampling time 1 s  (Comment: This may be too long; could make shorter to fit only initial response)
t = Ts*[0:109]';
u = [zeros(10,1); ones(100,1)];  % Step response
y = lsim(G,u,t);

% Now fit the data to a second-order plus delay model using Matlab
data=iddata(y,u,Ts);  %
type=('P2D')          % P2D =  2nd order model + delay
sys = procest(data,type)

% Compare the two models

k=sys.Kp; tau1=sys.Tp1; tau2=sys.Tp2; Td=sys.Td;
Gfit = k*exp(-Td*s)/((tau1*s+1)*(tau2*s+1))
step(G,Gfit,'--')
figure(2),step(G,Gfit,'--',10)
```



OUTPUT FROM procest (MATLAB):

Process model with transfer function:

$$G(s) = \frac{Kp}{(1+Tp1*s)(1+Tp2*s)} * \exp(-Td*s)$$

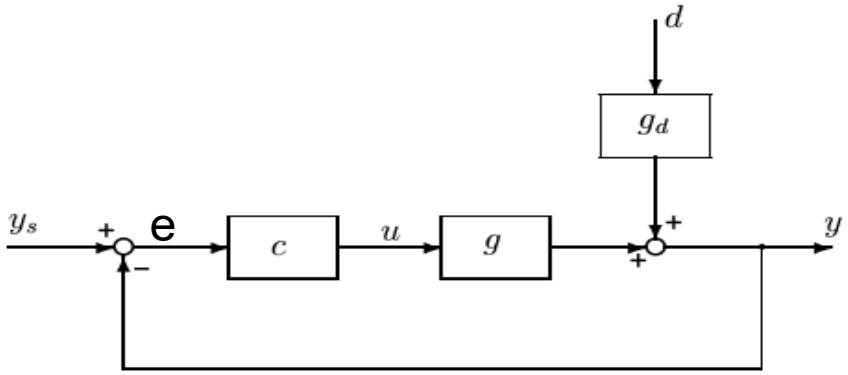
Kp = 2.9986
Tp1 = 9.6838
Tp2 = 3.9211,
Td = 2.478

Half rule: Gives similar result

Half rule:

Kp = 3
Tp1 = 10
Tp2=3+0.5=3.5
Td=0.5+2*1+0.1=2.6

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)

$$c(s) = K_c'(1 + \frac{1}{\tau_I' s} + \tau_D' s)$$

- For our purposes. Simpler with cascade/series form

$$c(s) = K_c \frac{(\tau_I s+1)(\tau_D s+1)}{\tau_I s}$$

$$K_c' = K_c\left(1+\frac{\tau_D}{\tau_I}\right); \quad \tau_I' = \tau_I\left(1+\frac{\tau_D}{\tau_I}\right); \quad \tau_D' = \frac{\tau_D}{1+\frac{\tau_D}{\tau_I}}$$

- Usually τ_D=0. Then the two forms are identical.
- Only two parameters left (K_c and τ_I)
- How difficult can it be to tune???
  - Surprisingly difficult without systematic approach!

Let's start with the CONCLUSION

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!

$$G(s) = \frac{k}{(\tau_1 s+1)(\tau_2 s+1)}\, e^{-\theta s}$$

For cascade-form PID controller:

$$K_c = \frac{1}{k'} \cdot \frac{1}{(\theta+\tau_c)}$$
$$\tau_I = \min(\tau_1, 4(\tau_c+\theta))$$
$$\tau_D = \tau_2$$

- τ_c: desired closed-loop response time (tuning parameter)
- For robustness select: $\tau_c \geqslant \theta$

Reference: S. Skogestad, "Simple analytic rules for model reduction and PID controller design", *J.Proc.Control*, Vol. 13, 291-309, 2003 (Also reprinted in MIC)

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

Derivation of SIMC-PID tuning rules

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

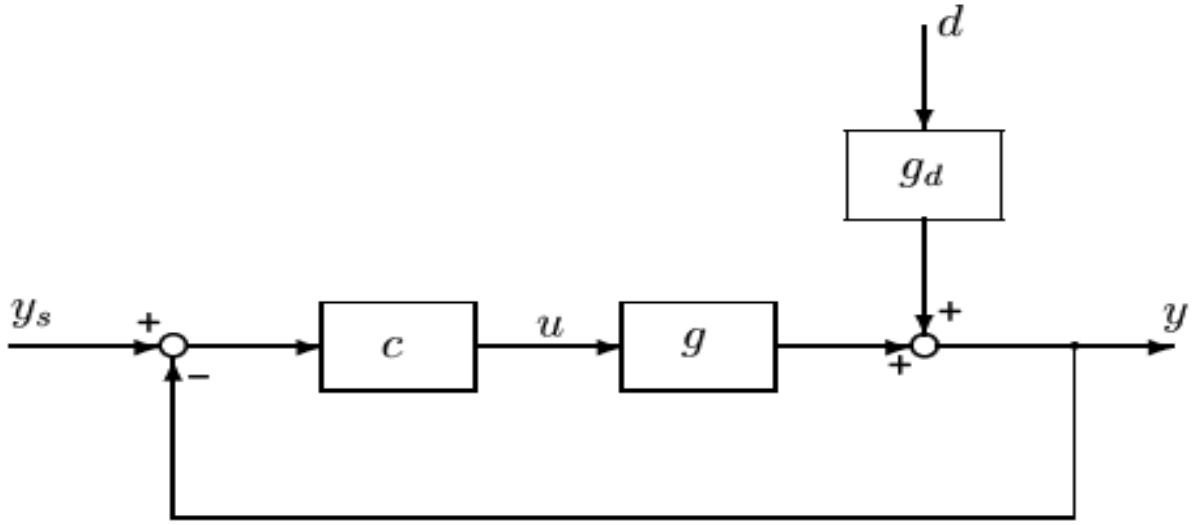
$$c(s) = K_c(1 + \frac{1}{\tau_I s}) = K_c \frac{\tau_I s + 1}{\tau_I s}$$

- For second-order model add D-action.

  For our purposes, simplest with the "series" (cascade) PID-form:

$$c(s) = K_c \frac{(\tau_I s + 1)(\tau_D s + 1)}{\tau_I s} \qquad (1)$$

Basis: Direct synthesis (IMC)



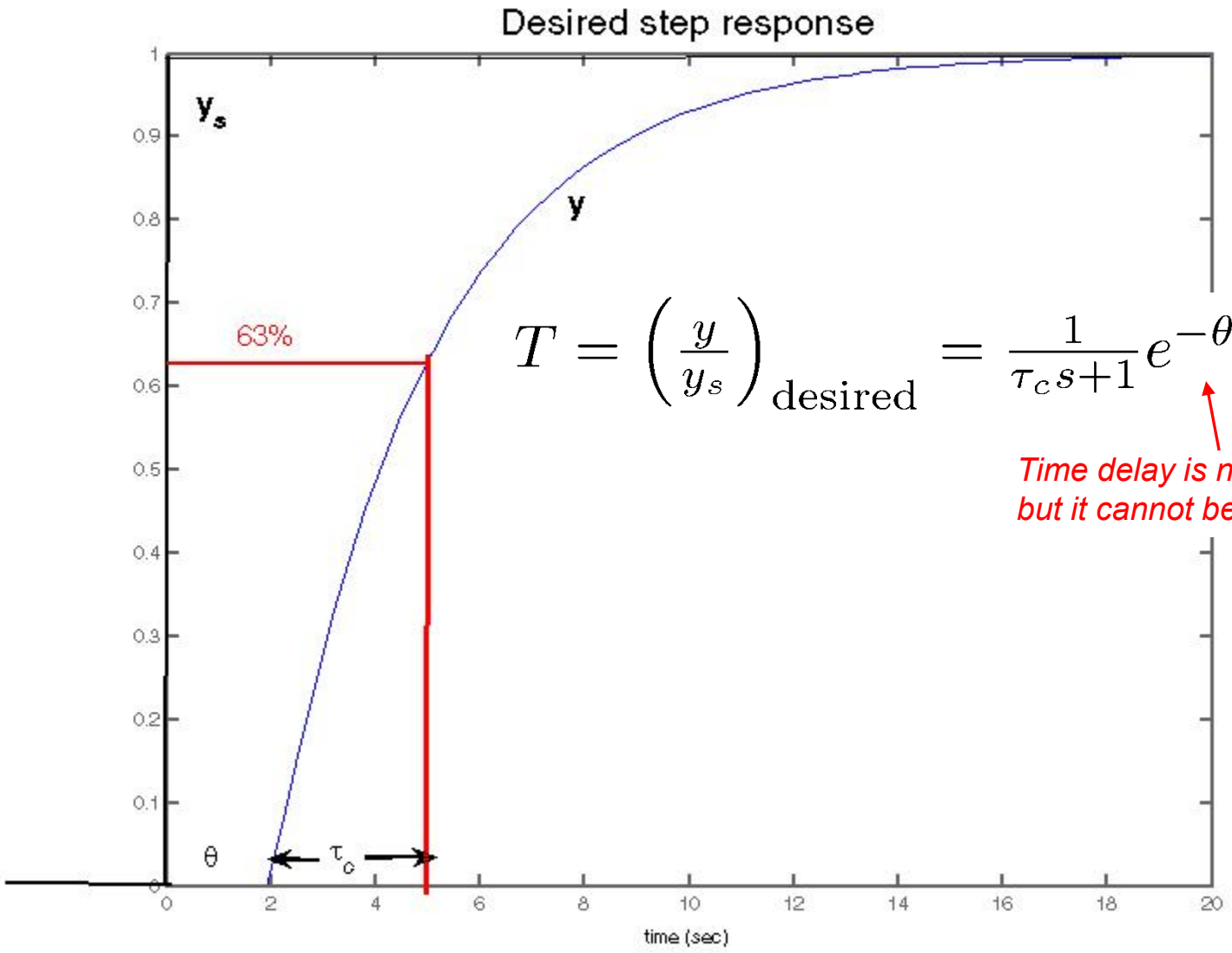
Closed-loop response to setpoint change

$$y = T\ y_s;\ \ T(s) = \frac{gc}{1+gc}$$

Idea: Specify desired response: $(y/y_s)_{\text{desired}} = T$

and from this get the controller. ……. Algebra: $c = \frac{1}{g} \cdot \frac{1}{\frac{1}{T}-1}$

Note: Process g has time delay (θ)



$$T = \left(\frac{y}{y_s}\right)_{\text{desired}} = \frac{1}{\tau_c s+1} e^{-\theta s}$$

Time delay is not really desired but it cannot be avoided

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

IMC Tuning = Direct Synthesis

Algebra:

- Controller: $c(s) = \frac{1}{g(s)} \cdot \frac{1}{\frac{1}{(y/y_s)_{\text{desired}}} - 1}$
- 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: $\left(\frac{y}{y_s}\right)_{\text{desired}} = \frac{1}{\tau_c s+1}e^{-\theta s}$
- Gives a "Smith Predictor" controller: $c(s) = \frac{(\tau_1 s+1)(\tau_2 s+1)}{k}\frac{1}{(\tau_c s+1-e^{-\theta s})}$
- To get a PID-controller use $e^{-\theta s} \approx 1 - \theta s$ and derive

$$c(s) = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{k}\frac{1}{(\tau_c + \theta)s}$$

which is a cascade form PID-controller with

$$K_c = \frac{1}{k}\frac{\tau_1}{\tau_c + \theta}; \quad \tau_I = \tau_1; \quad \tau_D = \tau_2$$

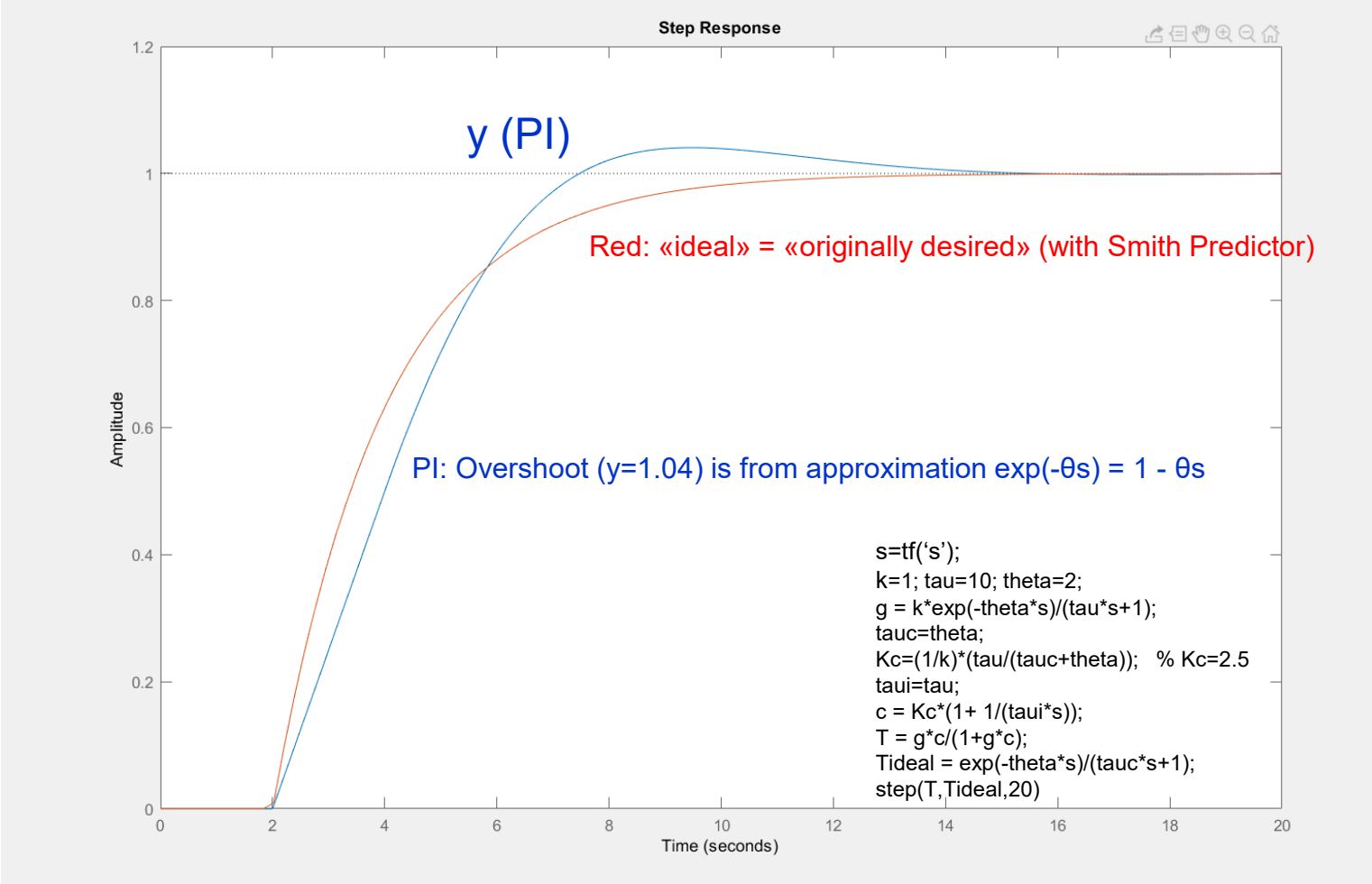
- τ_c is the sole tuning parameter

IMC-tuning is the same as "Lambda-tuning": τ_c is sometimes called λ

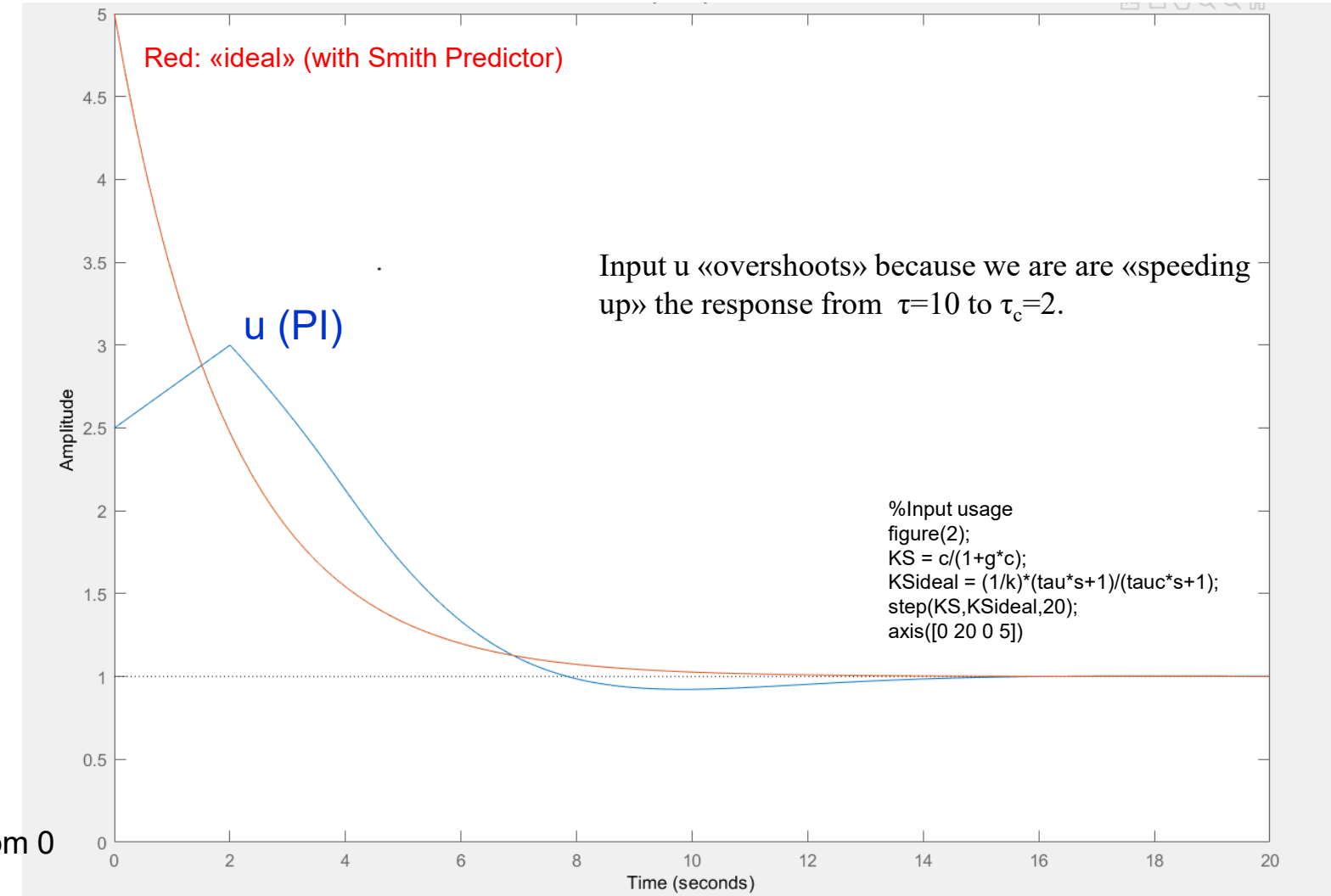
Surprisingly, this PID-controller is generally better, or at least more robust with respect to changes in the time delay θ, than the Smith Predictor controller from which it was derived. We are lucky ☺.

Reference: Chriss Grimholt and Sigurd Skogestad. "Should we forget the Smith Predictor?" (2018)
In 3rd IFAC conference on Advances in PID control, Ghent, Belgium, 9-11 May 2018. *In IFAC papers Online (2018)* .

Example step setpoint response (with choice $\tau_c=\theta=2$)



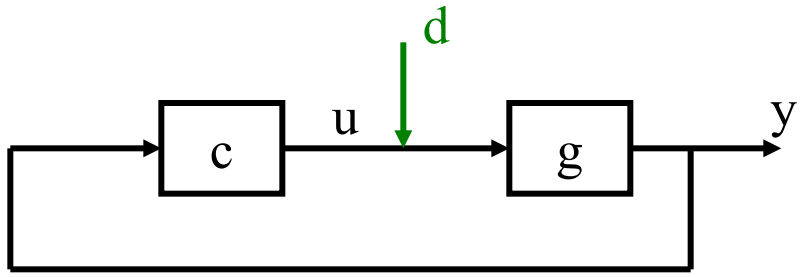
Input usage for setpoint response



Input starts from 0

Integral time

- Found: Integral time = dominant time constant ($\tau_I = \tau_1$)
- Gives P-controller for integrating process ($\tau_I = \infty$)
  - This works well for setpoint changes
  - But: τ_I needs to be modified (reduced) for integrating disturbances



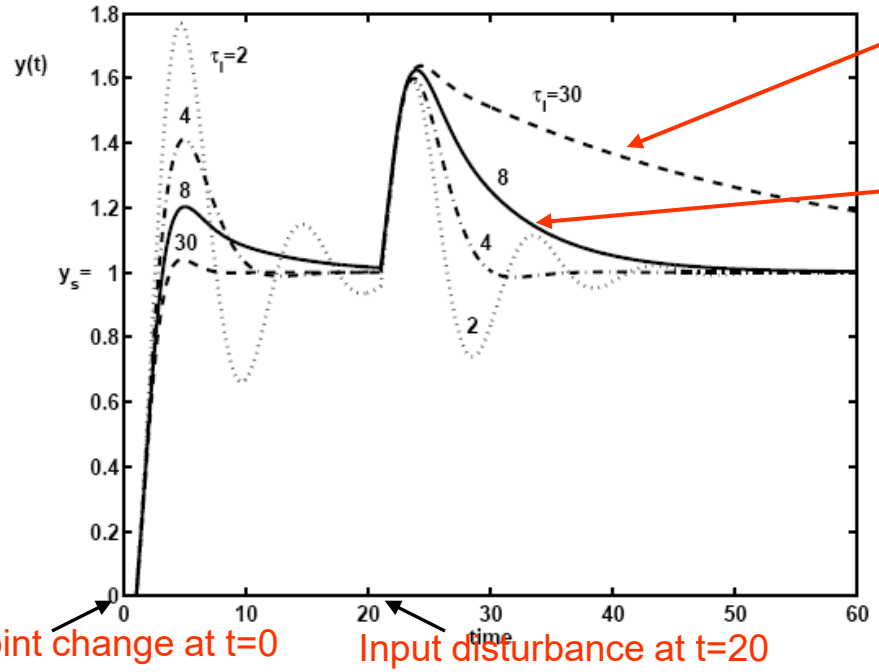
Example. "Almost-integrating process" with disturbance at input:

$G(s) = e^{-s}/(30s+1)$

Original integral time $\tau_I = 30$ gives poor disturbance response

Try reducing it!

Effect of decreasing Integral Time



Setpoint change at t=0

Input disturbance at t=20

$\tau_I = \tau_1$

Reduce τ_I to this value:
$\tau_I = 4\ (\tau_c+\theta) = 8\ \theta$

SIMC modification:
Decrease integral time to improve disturbance rejection for slow processes (with large τ)!

Figure 2: Effect of changing the integral time τ_I for PI-control of "slow" process $g(s) = e^{-s}/(30s+1)$ with $K_c = 15$. Load disturbance of magnitude 10 occurs at $t = 20$.

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" (not caused by the delay θ) we must require $k'K_c\tau_I \geq 4$, which with the SIMC-rule for K_c gives:

$$\tau_I \geq 4(\tau_C + \theta)$$

- Proof: Need $k'K_c\tau_I \geq 4$ to avoid slow oscillations (from last week):

$G(s) = k\frac{e^{-\theta s}}{\tau_1 s+1} \approx \frac{k'}{s}$ where $k' = \frac{k}{\tau_1}$; $C(s) = K_c\left(1 + \frac{1}{\tau_I s}\right)$

Closed-loop poles:

$1 + GC = 0 \Rightarrow 1 + \frac{k'}{s}K_c\left(1 + \frac{1}{\tau_I s}\right) = 0 \Rightarrow \tau_I s^2 + k'K_c\tau_I s + k'K_c = 0$

To avoid oscillations we must not have complex poles:

$B^2 - 4AC \geq 0 \Rightarrow k'^2K_c^2\tau_I^2 - 4k'K_c\tau_I \geq 0 \Rightarrow \boxed{k'K_c\tau_I \geq 4} \Rightarrow \tau_I \geq \frac{4}{k'K_c}$

Inserted SIMC-rule for $K_c = \frac{1}{k'}\frac{1}{\tau_c+\theta}$ then gives

$\tau_I \geq 4(\tau_c + \theta)$

Conclusion: SIMC-PID Tuning Rules

For cascade form PID controller:

$$K_c = \frac{1}{k}\frac{\tau_1}{\tau_c + \theta} = \frac{1}{k'} \cdot \frac{1}{\tau_c + \theta} \tag{1}$$

$$\tau_I = \min\{\tau_1, \frac{4}{k'\ K_c}\} = \min\{\tau_1, 4(\tau_c + \theta)\} \tag{2}$$

$$\tau_D = \tau_2 \tag{3}$$

Derivation:

1. First-order setpoint response with response time τ_c (IMC-tuning = "Direct synthesis")
2. Reduce integral time to get better disturbance rejection for slow or integrating process (but avoid slow cycling $\Rightarrow \tau_I \geq \frac{4}{k'\ K_c}$)

One tuning parameter: τ_c

SIMC-tunings

Some special cases

Process	$g(s)$	K_c	τ_I	$\tau_D^{(4)}$
First-order	$k\frac{e^{-\theta a}}{(\tau_1 s+1)}$	$\frac{1}{k}\frac{\tau_1}{\tau_c+\theta}$	$\min\{\tau_1, 4(\tau_c+\theta)\}$	-
Second-order, eq.(4)	$k\frac{e^{-\theta a}}{(\tau_1 s+1)(\tau_2 s+1)}$	$\frac{1}{k}\frac{\tau_1}{\tau_c+\theta}$	$\min\{\tau_1, 4(\tau_c+\theta)\}$	τ_2
Pure time delay[1]	$ke^{-\theta s}$	0	0 [(*)]	-
Integrating[2]	$k'\frac{e^{-\theta s}}{s}$	$\frac{1}{k'}\cdot\frac{1}{(\tau_c+\theta)}$	$4(\tau_c+\theta)$	-
Integrating with lag	$k'\frac{e^{-\theta s}}{s(\tau_2 s+1)}$	$\frac{1}{k'}\cdot\frac{1}{(\tau_c+\theta)}$	$4(\tau_c+\theta)$	τ_2
Double integrating[3]	$k''\frac{e^{-\theta s}}{s^2}$	$\frac{1}{k''}\cdot\frac{1}{4(\tau_c+\theta)^2}$	$4\ (\tau_c+\theta)$	$4\ (\tau_c+\theta)$

Table 1: SIMC PID-settings (23)-(25) for some special cases of (4) (with τ_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 $c(s) = \frac{K_I}{s}$ with $K_I \stackrel{\text{def}}{=} \frac{K_c}{\tau_I} = \frac{1}{k(\tau_c+\theta)}$.

One tuning parameter: τ_c

(1)(*) Note that we get pure I-controller for static process with delay.

Selection of tuning parameter τ_c

Two main cases

1. TIGHT CONTROL (τ_c small): Want "fastest possible control" subject to having good robustness
   - Want tight control of active constraints ("squeeze and shift")
   - Select $\tau_c = \theta$ (effective delay)

2. SMOOTH CONTROL (τ_c large): Want "slowest possible control" subject to acceptable disturbance rejection
   - Prefer smooth control if fast control is not required

TIGHT CONTROL

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

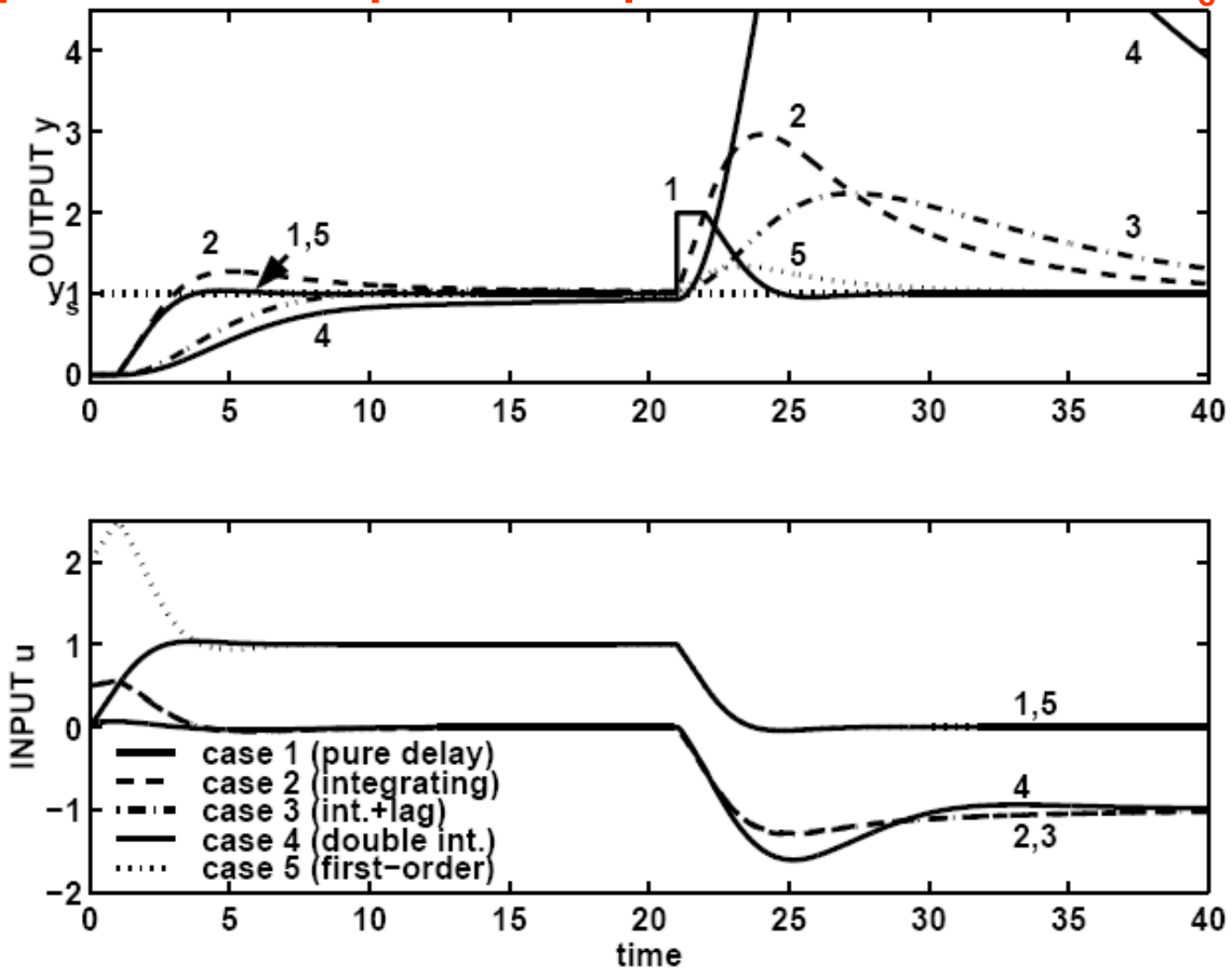


Figure 4: Responses using SIMC settings for the five time delay processes in Table 3 ($\tau_c = \theta$).
Unit setpoint change at $t = 0$; Unit load disturbance at $t = 20$.
Simulations are without derivative action on the setpoint.
Parameter values: $\theta = 1, k = 1, k' = 1, k'' = 1$.

TIGHT CONTROL

TUNING FOR FAST RESPONSE WITH GOOD ROBUSTNESS

$$\text{SIMC}: \quad \tau_c = \theta \tag{4}$$

Gives:

$$K_c = \frac{0.5}{k}\frac{\tau_1}{\theta} = \frac{0.5}{k'} \cdot \frac{1}{\theta} \tag{5}$$

$$\tau_I = \min\{\tau_1, 8\theta\} \tag{6}$$

$$\tau_D = \tau_2 \tag{7}$$

Gain margin about 3 (can increase process gain k by factor 3 before we get instability from «overreaction»)

Process $g(s)$ Controller gain, K_c Integral time, τ_I	$\frac{k}{\tau_1 s+1}e^{-\theta s}$ $\frac{0.5}{k}\frac{\tau_1}{\theta}$ τ_1	$\frac{k'}{s}e^{-\theta s}$ $\frac{0.5}{k'}\frac{1}{\theta}$ 8θ
Gain margin (GM)	3.14	2.96
Phase margin (PM)	61.4^o	46.9^o
Allowed time delay error, $\Delta\theta/\theta$	2.14	1.59
Sensitivity peak, M_s	1.59	1.70
Complementary sensitivity peak, M_t	1.00	1.30
Phase crossover frequency, $\omega_{180} \cdot \theta$	1.57	1.49
Gain crossover frequency, $\omega_c \cdot \theta$	0.50	0.51

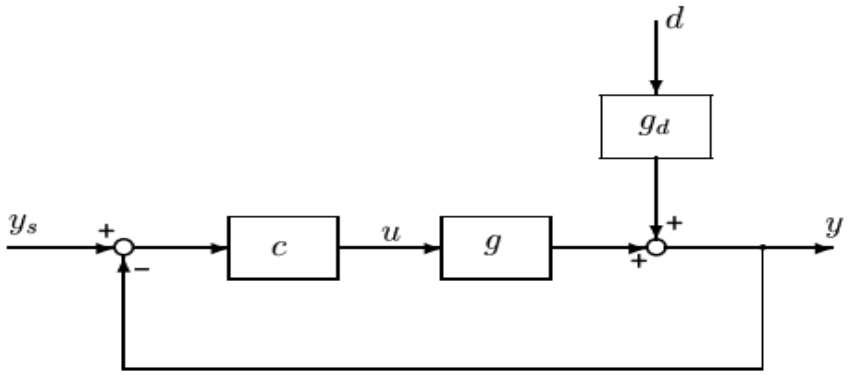
Table 1: Robustness margins for first-order and integrating delay process using SIMC-tunings in (5) and (6) ($\tau_c = \theta$). The same margins apply to second-order processes if we choose $\tau_D = \tau_2$.

Example 2. Compare PI and PID

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

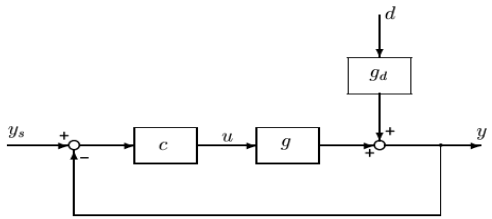
```
s=tf('s')
g=(-0.3*s+1)*(0.08*s+1)/((2*s+1)*(s+1)*(0.4*s+1)*(0.2*s+1)*(0.05*s+1)^3)
k=1;
tau1=2.5, tau2=0, theta=1.47, tauc=theta % 1st order
%tau1=2, tau2=1.2, theta=0.77, tauc=theta % 2nd order

Kc=(1/k)*tau1/(tauc+theta)        % Kc.   PI: 0.85  PID: 1.30
taui=min(tau1,4*(tauc+theta))     % taui. PI: 2.50  PID: 2
taud=tau2;                         % taud. PI: 0     PID: 1.2
cpi=Kc*(1+1/(taui*s));
cd=(taud*s+1)/(0.1*taud*s+1);
cpid=cpi*cd;
L = cpid*g
S=inv(1+L)
%setpoint response
Ty=g*cpi*S, Ty=minreal(Ty); % without D-action on setpoint
Tuy=cpi*S, Tuy=minreal(Tuy); % without D-action on setpoint
%Input disturbance
gd=g;
Td=gd*S; Td=minreal(Td);
Tud=-gd*cpid*S; Tud=minreal(Tud);
Typi=Ty; Tdpi=Td; Tuypi=Tuy; Tudpi=Tud;
%Typid=Ty; Tdpid=Td; Tuypid=Tuy; Tudpid=Tud;

figure(1),step(Typi,'blue',Typid,'blue--',Tuypi,'red',Tuypid,'red--',15)
figure(2),step(Tdpi,'blue',Tdpid,'blue--',Tudpi,'red',Tudpid,'red--',15)
```

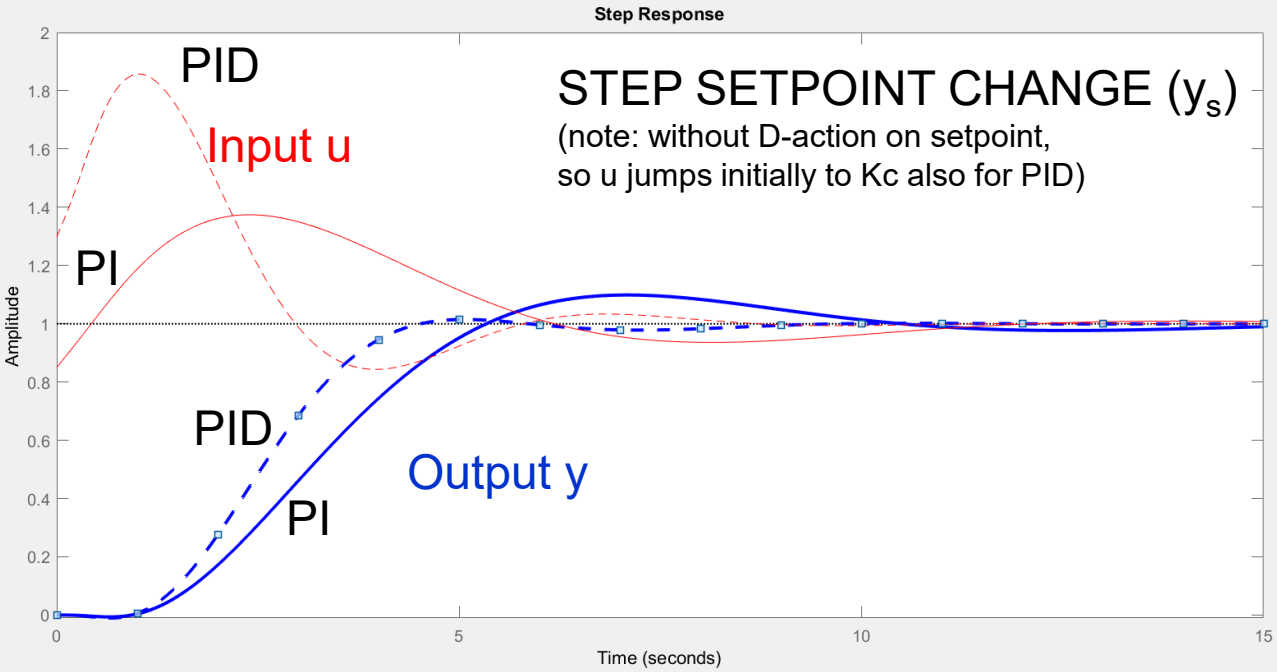
Note: tau2=1.2 > theta=0.77
so 2nd order and PID should give improvement compared to PI

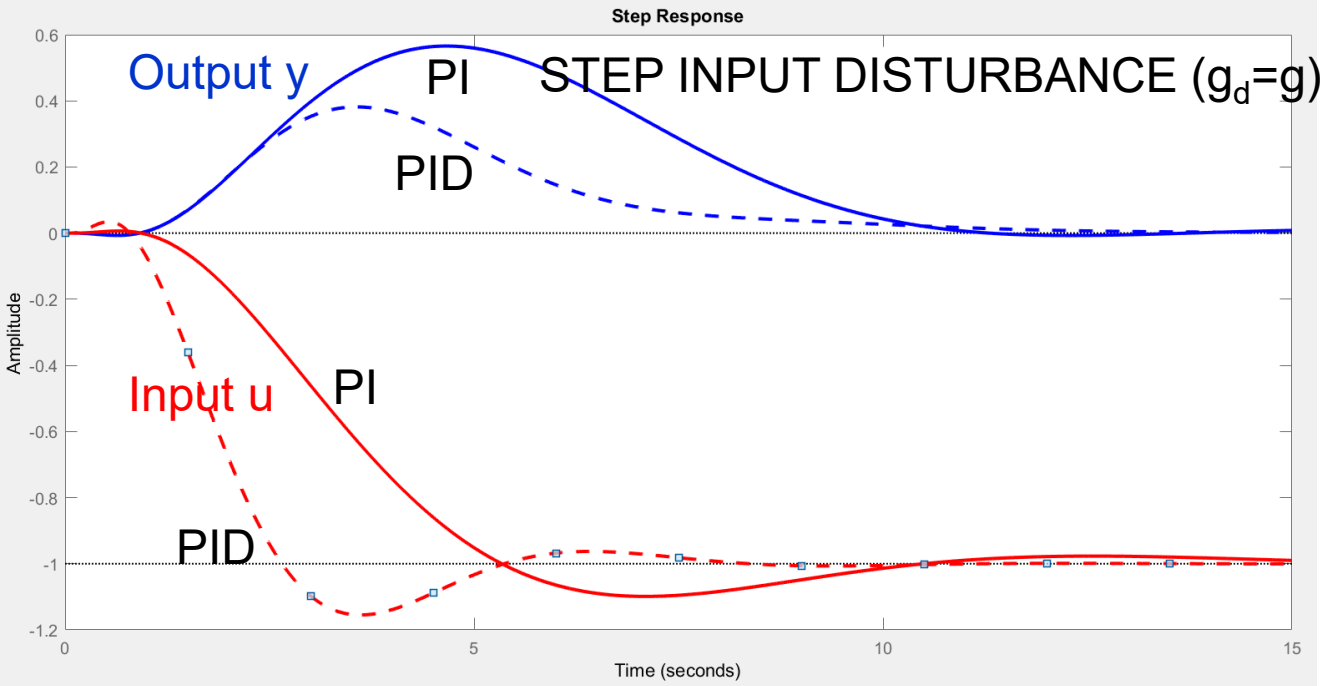


Example 2.

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







Conclusion:
PID is quite a lot better.
(expected since tau2=1.2
> theta=0.77)

SHOULD WE ADD **DERIVATIVE ACTION** TO COUNTERACT TIME DELAY?

First order with delay plant ($\tau_2 = 0$) with $\tau_c = \theta$:

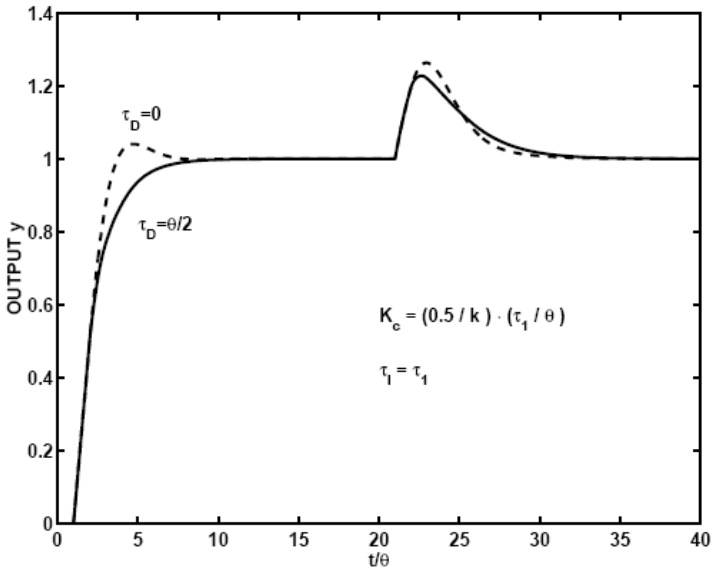


Figure 5: Setpoint change at $t = 0$. Load disturbance of magnitude 0.5 occurs at $t = 20$.

NO

- Observe: Derivative action (solid line) has only a minor effect.
- Conclusion: Use second-order model (and derivative action) only when $\tau_2 > \theta$ (approximately)

Comments

1. Derivative action (PID) can help a little to speed up response for a process with time delay (e.g. use $\tau_D = \theta/3$), but we then need to reduce τ_C (i.e., increase K_c) to get the performance benefit (e.g., reduce τ_C from θ to $\theta/2$).
We did not do this in the above simulation, so this is why the benefit of D-action is small.
2. Use derivative action (PID) for unstable processes, for example, a double integrating process (not so common in process control).

6.3 Ideal PID controller

The settings given in this paper (K_c, τ_I, τ_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

$$\text{Ideal PID}: \quad c'(s) = K'_c \left(1 + \frac{1}{\tau'_I s} + \tau'_D s\right) = \frac{K'_c}{\tau'_I s}\left(\tau'_I \tau'_D s^2 + \tau'_I s + 1\right) \tag{35}$$

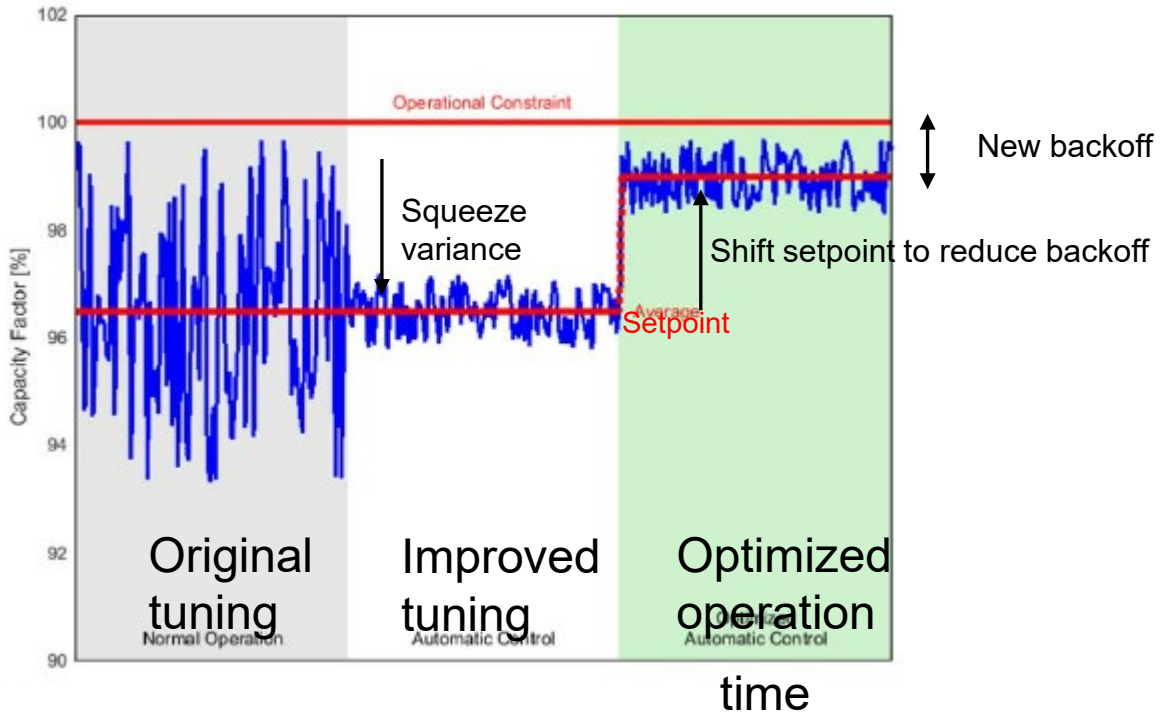
we use the following translation formulas

$$K'_c = K_c\left(1 + \frac{\tau_D}{\tau_I}\right); \quad \tau'_I = \tau_I\left(1 + \frac{\tau_D}{\tau_I}\right); \quad \tau'_D = \frac{\tau_D}{1 + \frac{\tau_D}{\tau_I}} \tag{36}$$

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

When do we need «tight control»?
For hard constraints where backoff is costly

«SQEEZE and SHIFT» RULE



Tuning for smooth control

- Tuning parameter: τ_c = desired closed-loop response time
- Selecting $\tau_c=\theta$ if we need "tight control" of y.
- Other cases: "Smooth control" of y is sufficient, so select $\tau_c > \theta$ 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 τ_c ?
  - ANSWER: $\tau_{c,max} = 1/\omega_d$ (where ω_d is defined as the frequence where $|g_d(j\omega_d)| = y_{max}/d_{max}$)

Proof: S. Skogestad, ``Tuning for smooth PID control with acceptable disturbance rejection", *Ind.Eng.Chem.Res*, **45** (23), 7817-7822 (2006).

Level control (integrating process)

- 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 (integrating process): Can have both fast and slow oscillations with PI-control

- Fast oscillations (K_c too high): $P < \pi \tau_I$
  - Caused by (effective) time delay
- Slow oscillations (K_c too low): $P > \pi \tau_I$
  - Caused by integral action in controller
  - Avoid slow oscillations: $k' K_c \tau_I \geq 4$.

P=period of oscillations = $2\pi/\omega$

How avoid slowly 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

$$f = 0.1 \cdot (P_0/\tau_{I0})^2$$

where

P_0 = period of oscillations [s]

τ_{I0} = original integral time [s]

$0.1 \approx 1/\pi^2$

$$P_0 = \frac{2\pi}{\sqrt{1-\zeta^2}}\tau_0 = \frac{2\pi}{\sqrt{1-\zeta^2}}\sqrt{\frac{\tau_{I0}}{k' K_{c0}}} \approx 2\pi\sqrt{\frac{\tau_{I0}}{k' K_{c0}}} \quad (39)$$

where we have assumed $\zeta^2 << 1$ (significant oscillations). Thus, from (39) the product of the original controller gain and integral time is approximately

$$K_{c0} \cdot \tau_{I0} = (2\pi)^2 \frac{1}{k'} \left(\frac{\tau_{I0}}{P_0}\right)^2$$

To avoid oscillations ($\zeta \geq 1$) with the new settings we must from (21) require $K_c\tau_I \geq 4/k'$, that is, we must require that

$$\frac{K_c \tau_I}{K_{c0}\tau_{I0}} \geq \frac{1}{\pi^2} \cdot \left(\frac{P_0}{\tau_{I0}}\right)^2 \quad (40)$$

Here $1/\pi^2 \approx 0.10$, so we have the **rule**:

- To avoid "slow" oscillations of period P_0 the product of the controller gain and integral time should be increased by a factor $f \approx 0.1(P_0/\tau_{I0})^2$.

Avoid slow oscillations: $k' K_C \tau_I \geq 4$

Case study oscillating level

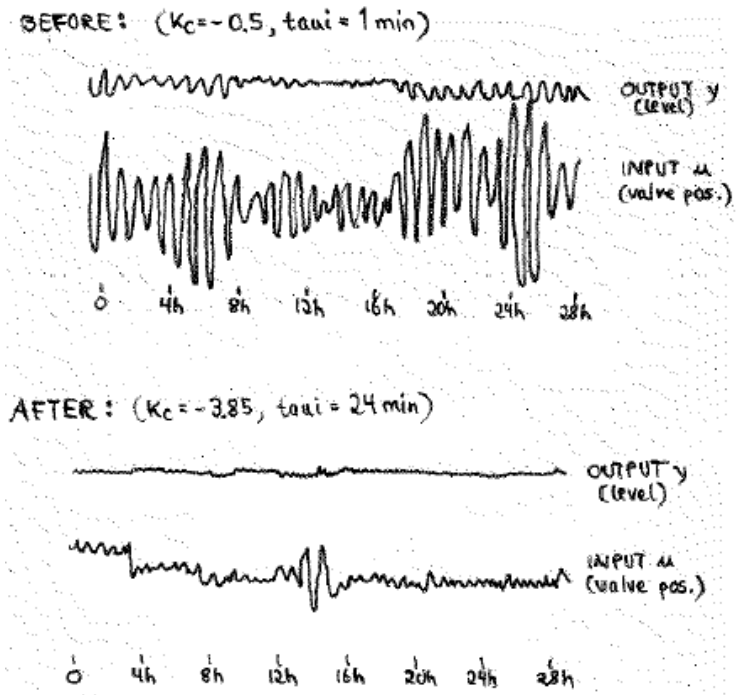
- We were called upon to solve a problem with oscillations in a distillation column
- Closer analysis: Problem was oscillating reboiler level in upstream column
- Use of Sigurd's rule solved the problem

APPLICATION: RETUNING FOR INTEGRATING PROCESS

To avoid "slow" oscillations the product of the controller gain and integral time should be increased by factor $f \approx 0.1(P_0/\tau_{I0})^2$.

Real Plant data:

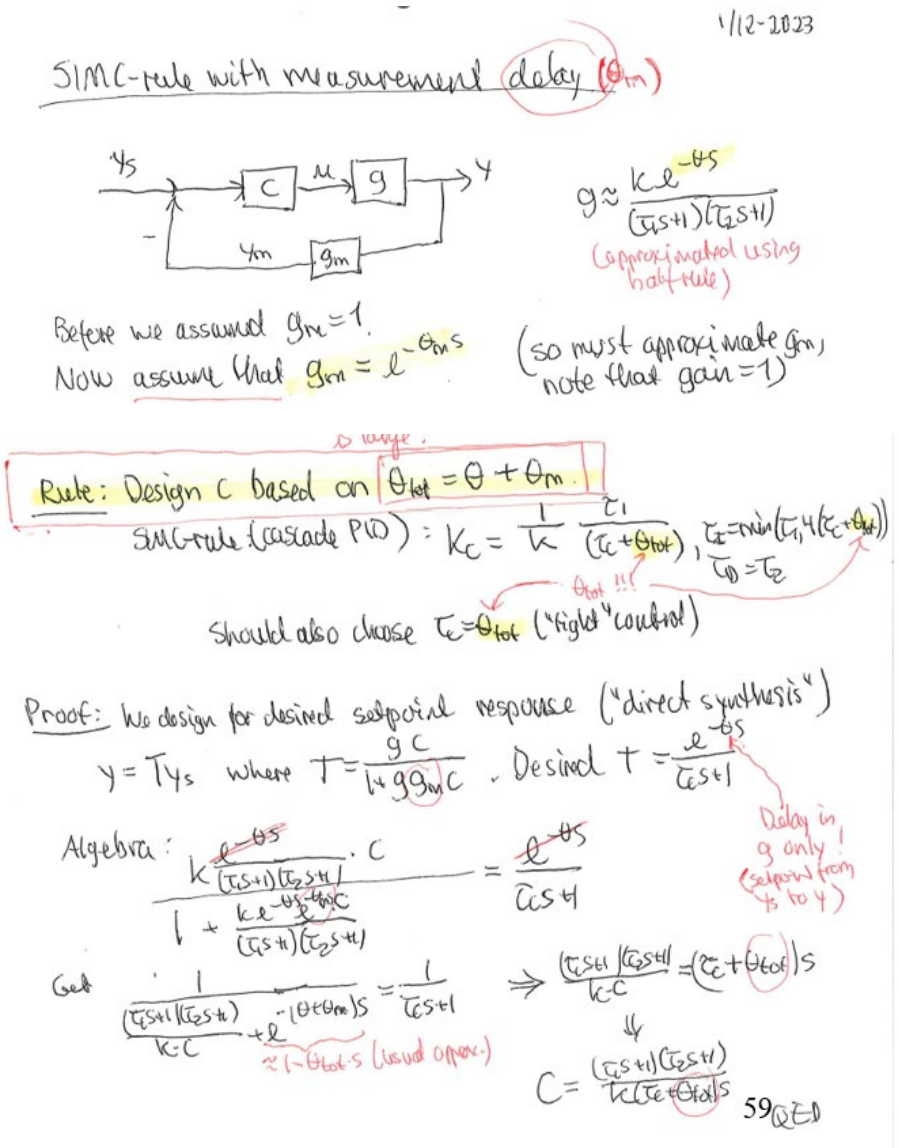
$$\text{Period of oscillations } P_0 = 0.85h = 51min \Rightarrow f = 0.1 \cdot (51/1)^2 = 260$$



SIMC-rule with measurement dynamics

Rule: Combine the measurement dynamics $g_m(s)$ and the process model g(s) and apply the SIMC-rules on gg_m. This applies both to the model approximation (half rule) to get a 1st or 2nd model and to the PI- or PID-tuning, including the choice of τ_c.

Proof: handwritten note



CONCLUSION

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!

For cascade-form PID controller:

$$K_c = \frac{1}{k'} \cdot \frac{1}{(\theta+\tau_c)}$$
$$\tau_I = \min(\tau_1, 4(\tau_c + \theta))$$
$$\tau_D = \tau_2$$

- τ_c: desired closed-loop response time (tuning parameter)
- For robustness select: $\tau_c \geqslant \theta$

Note: The delay θ includes any measurement delay

Reference: S. Skogestad, "Simple analytic rules for model reduction and PID controller design", *J.Proc.Control*, Vol. 13, 291-309, 2003 (Also reprinted in MIC)

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