Systems Engineering/Process control L9

The PID controller

- The algorithm
- Frequency analysis
- Practical modifications
- Tuning methods

Reading: *Systems Engineering and Process Control*: 9.1–9.6

The PID controller

> "Based on a survey of over eleven thousand controllers in the refining, chemicals and pulp and paper industries, 97% of regulatory controllers utilize PID feedback."
>
> [Desborough and Miller, 2002]

"School-book form":

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right)$$

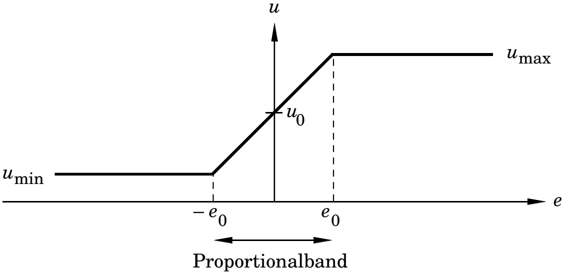
Transfer function:

$$G_c(s) = K\left(1 + \frac{1}{sT_i} + sT_d\right)$$

The P part

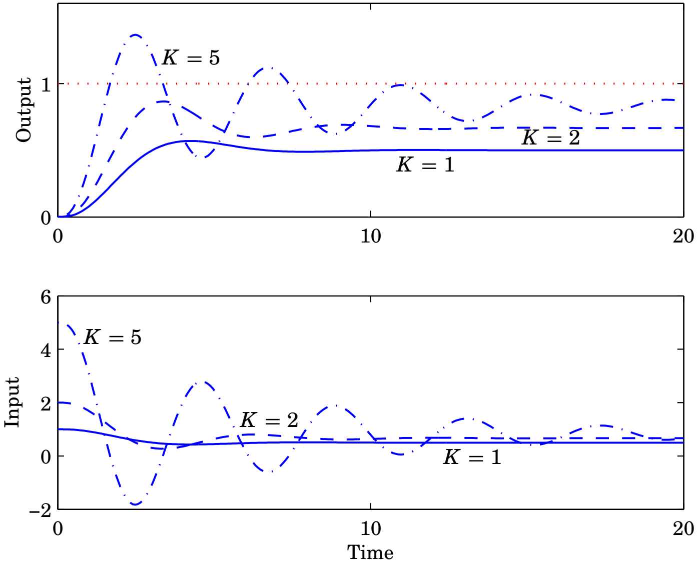
- P controller:

$$u = K(r - y) + u_0 = Ke + u_0$$



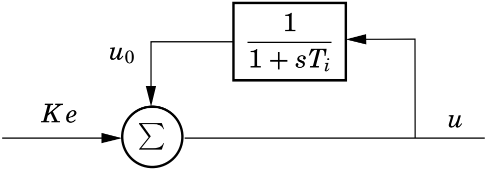
- u_0 can be chosen to eliminate stationary error at setpoint

Example: P control of $G_p(s) = (s+1)^{-3}$



The I part

- Introduce automatic/online/dynamic selection of u_0:

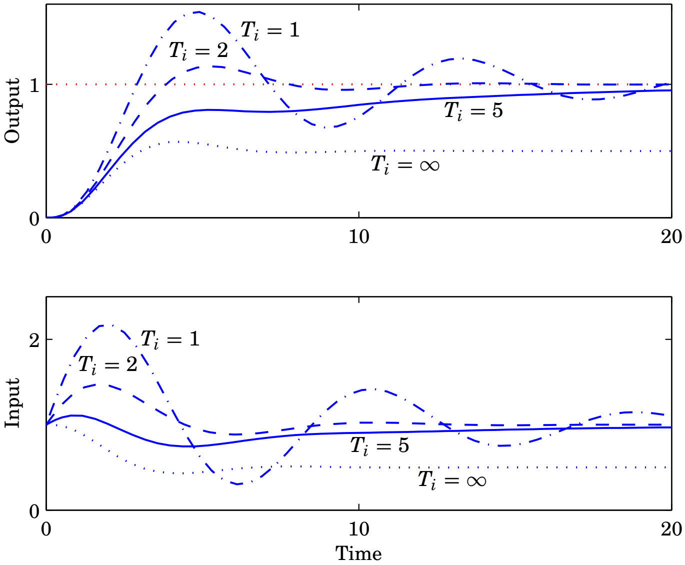


$$U(s) = KE(s) + \frac{1}{1+sT_i}U(s)$$

$$U(s) = K\left(1 + \frac{1}{sT_i}\right)E(s)$$

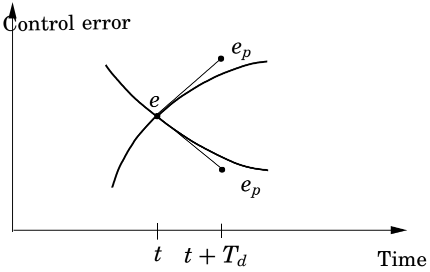
- Assume stationarity: How does u and u_0 relate? What is e?

Example: PI control of $G_p(s) = (s+1)^{-3}$ $(K = 1)$



The D part

- A P controller gives the same control in both these cases:



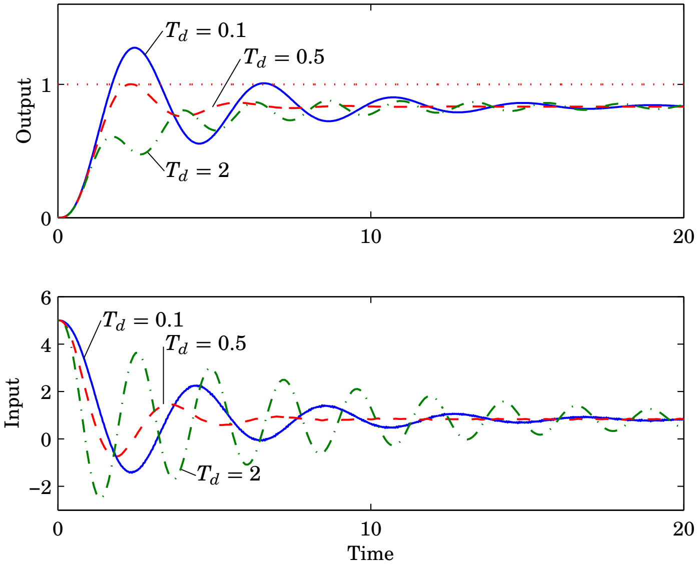
- Predicted error:

$$e_p(t + T_d) \approx e(t) + T_d \frac{de(t)}{dt}$$

- PD controller:

$$u(t) = K \left(e(t) + T_d \frac{de(t)}{dt} \right)$$

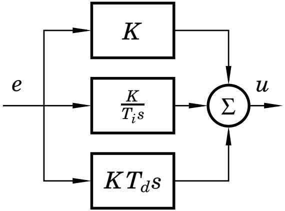
Example: PD control of $G_p(s) = (s+1)^{-3}$ $(K=5)$



Parallel and serial form

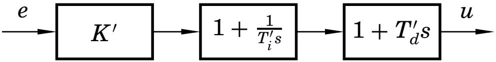
- PID controller on standard form (parallel form):

$$G_c(s) = K + \frac{K}{sT_i} + sKT_d$$



- PID controller on serial form (common in industry):

$$G_c'(s) = K'\left(1 + \frac{1}{sT_i'}\right)(1 + sT_d')$$



Parallel and serial form

Transformation parallel form $\leftrightarrow$ serial form:

$$K = K' \frac{T_i' + T_d'}{T_i'} \qquad K' = \frac{K}{2}\left(1 + \sqrt{1 - \frac{4T_d}{T_i}}\right)$$

$$T_i = T_i' + T_d' \qquad T_i' = \frac{T_i}{2}\left(1 + \sqrt{1 - \frac{4T_d}{T_i}}\right)$$

$$T_d = \frac{T_i' T_d'}{T_i' + T_d'} \qquad T_d' = \frac{T_i}{2}\left(1 - \sqrt{1 - \frac{4T_d}{T_i}}\right)$$

- Identical parameters for PI and PD controller
- Parallel $\rightarrow$ serial only possible if $T_i \geq 4T_d$
    - Parallel form more general

Frequency analysis of PID controller

Frequency function for PID controller on serial form:

$$G_c'(i\omega) = \frac{K'}{i\omega T_i'}(1 + i\omega T_i')(1 + i\omega T_d')$$

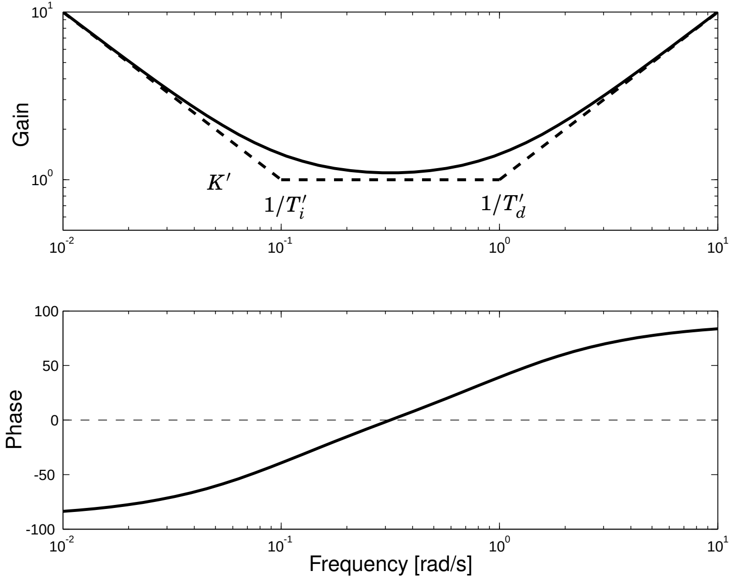
- For low frequencies (small ω):

$$|G_c'(i\omega)| \approx \frac{K'}{\omega T_i'}$$

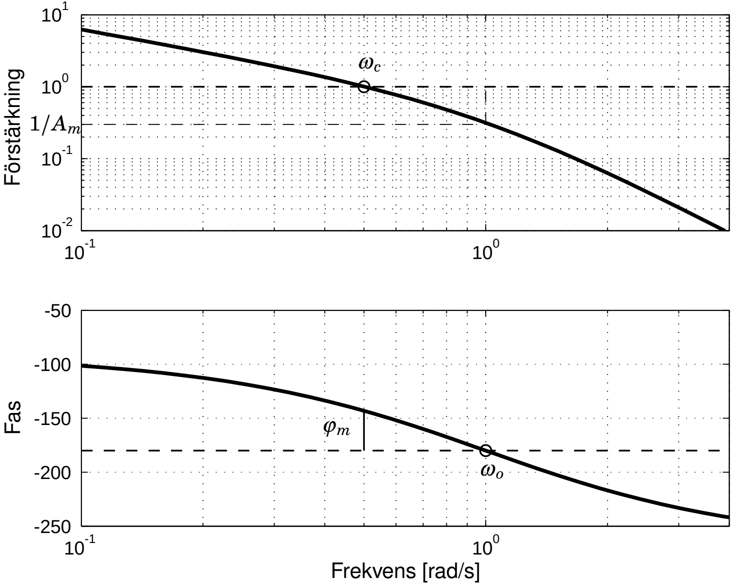
$$\arg G_c'(i\omega) \approx -90^\circ$$

- Zero at $s = -1/T_i'$ bends amplitude curve up and increases phase with 90° around $\omega = 1/T_i'$
- The same holds for the zero at $s = -1/T_d'$

Frequency analysis of PID controller



Repetition: Amplitude and phase margin



Frequency analysis of PID controller

The P part:

- Affects gain at all frequencies
- Higher gain $\Rightarrow$ faster system but worse margins

The I part:

- Increases gain and reduces phase for low frequencies
- Eliminates low frequency (constant) control errors but gives worse phase margin

The D part:

- Increases gain and phase at high frequencies
- Gives better phase margin (to a limit) but amplifies noise

Practical modifications of PID controllers

School-book form:

$$e(t) = r(t) - y(t)$$

$$u(t) = \underbrace{Ke(t)}_{P(t)} + \underbrace{\frac{K}{T_i}\int_0^t e(\tau)d\tau}_{I(t)} + \underbrace{KT_d\frac{de(t)}{dt}}_{D(t)}$$

Modifications:

- The P part: reference weighting
- The I part: anti-windup
- The D part: reference weighting and limited gain

Modification of P part

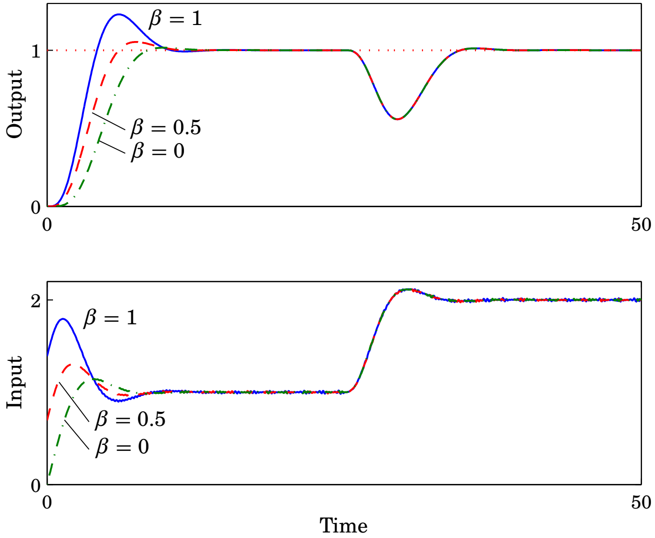
- Introduce reference weighting β:

$$P(t) = K\big(\beta r(t) - y(t)\big), \quad 0 \leq \beta \leq 1$$

- Can be used to limit overshoot after reference changes (moves a zero in closed-loop system)
- Note! Works only if also I part used

Example: Reference weighting with PI control

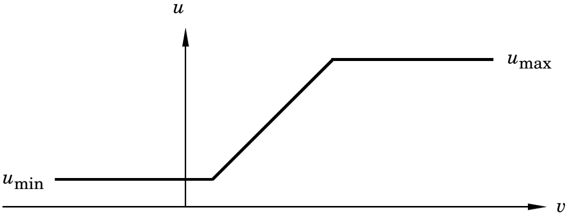
(reference change at $t = 0$, load disturbance at $t = 25$):



Modification of I part

Input is always limited in practice ($u_{\min} \leq u \leq u_{\max}$)

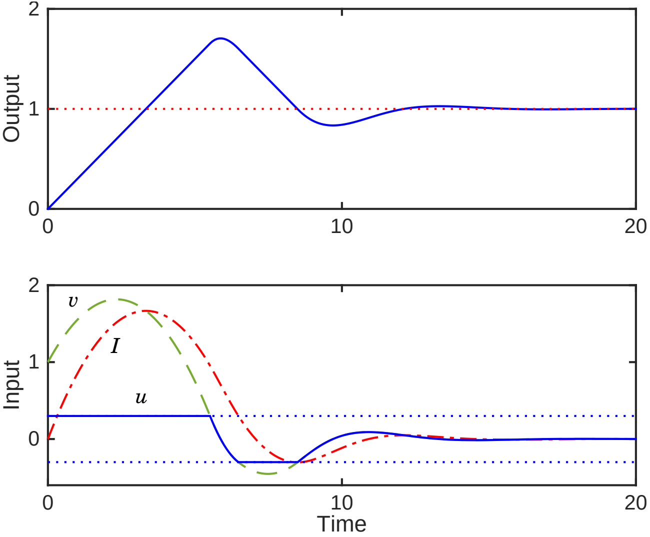
- Let v be the input the controller *wants* to use
- Let u be the input the controller *can* use



Integrator windup: I part keeps growing when signal saturated

Example: PI control with integrator windup

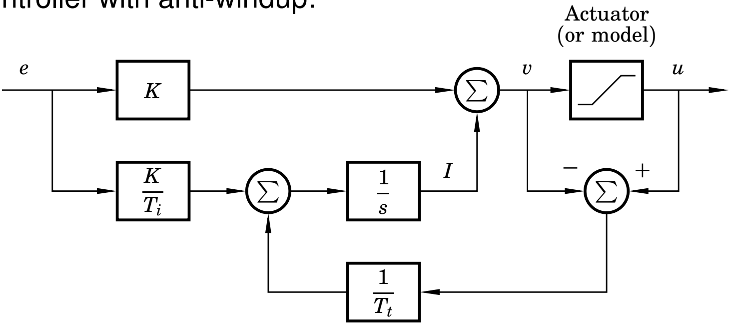
$G_p(s) = 1/s, \; K = T_i = 1, \; -0.3 \leq u \leq 0.3$:



Anti-windup

$$I(t) = \int_0^t \left(\frac{K}{T_i} e(\tau) + \frac{1}{T_t} \big(u(\tau) - v(\tau)\big) \right) d\tau$$

PI controller with anti-windup:

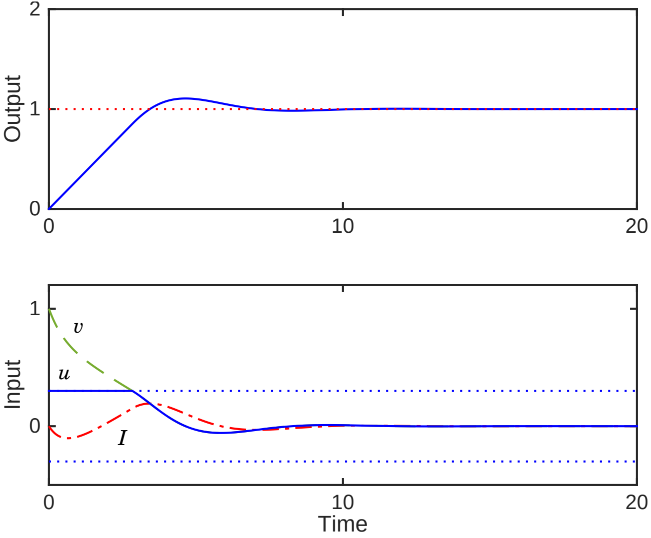


Rule of thumb for constant T_t:

- PI controller: $T_t = 0.5T_i$
- PID controller: $T_t = \sqrt{T_i T_d}$

Example: PI control with anti-windup

Same example as before, but with anti-windup ($T_t = 0.5$):



Modification of D part

- Reference weighting: derivate only measurement, not reference

$$D(t) = -KT_d \frac{dy(t)}{dt}$$

- Limit gain with low-pass filter (extra pole):

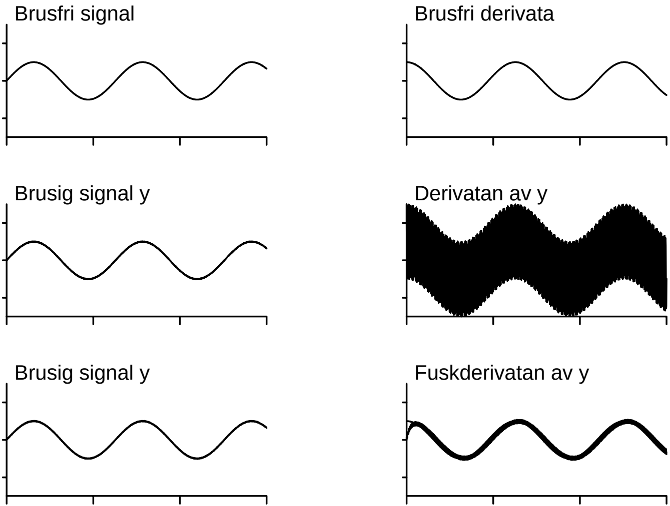
$$D(s) = -\frac{sKT_d}{1 + sT_d/N} Y(s)$$

(“fuskderivata”)

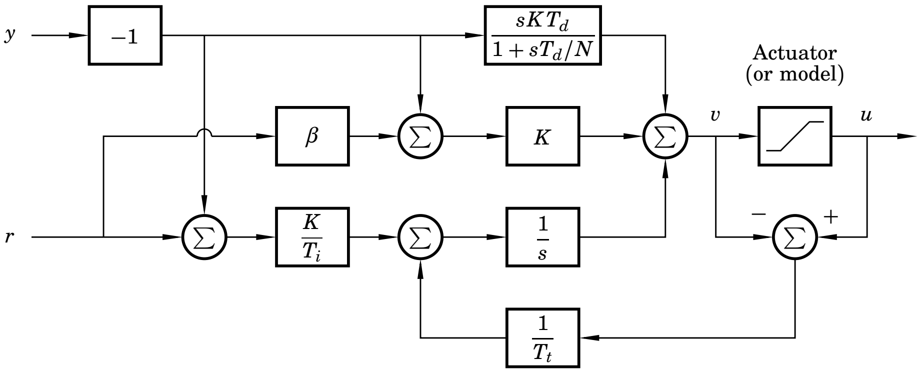
Maximal derivative gain N typically chosen in interval 5–20

Example: Limited derivative gain

$y(t) = \sin t + 0.01 \sin 100t, \;\; T_d = 1, \; N = 5$



Summary: Practical modifications



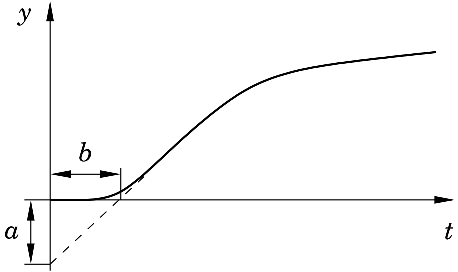
(More to think about: bumpless transfer between manual/automatic control, bumpless parameter changes, sampling filters, sampling, ...)

Tuning methods for PID controllers

- Manual tuning (lab 1)
- Ziegler–Nichols methods
- The Lambda method
- Arresttidstrimning (project)
- Model-based tuning (lab 2)
- Relay methods
- Optimization-based methods
- …

Ziegler–Nichols step response method

Experiment on **open-loop** system, read a and b in step response:



Controller	K	T_i	T_d
P	$1/a$		
PI	$0.9/a$	$3b$	
PID	$1.2/a$	$2b$	$0.5b$

Ziegler–Nichols frequency method

(*Ziegler–Nichols' ultimate-sensitivity method*)

Experiment on **closed-loop** system

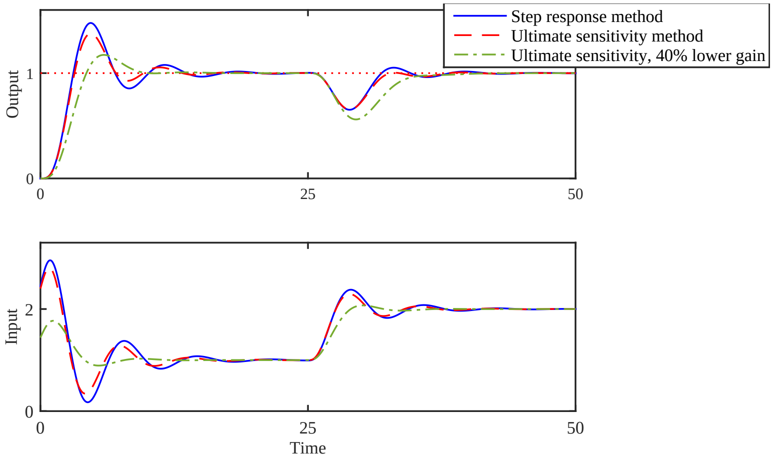
1. Disconnect I and D parts in PID controller
2. Increase K until oscillations with constant amplitude. This $K = K_0$.
3. Measure period time T_0 for oscillations.

Controller	K	T_i	T_d
P	$0.5K_0$		
PI	$0.45K_0$	$T_0/1.2$	
PID	$0.6K_0$	$T_0/2$	$T_0/8$

(Note that $T_0 = 2\pi/\omega_0$, where ω_0 is frequency that gives $-180°$ phase shift)

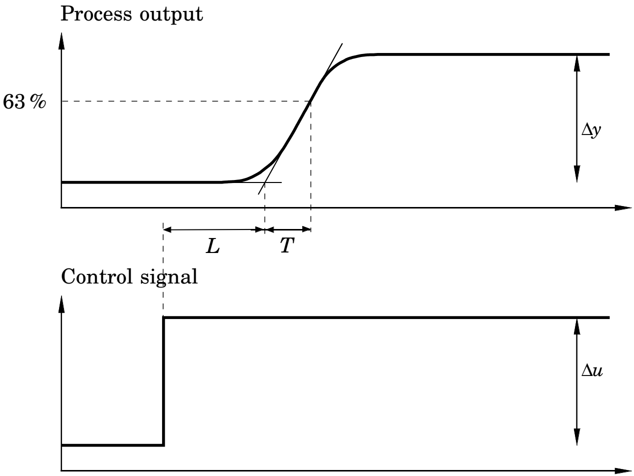
Ziegler–Nichols methods – warning

- Ziegler–Nichols' methods give aggressive control with bad damping
- Recommendation: K lowered with 30–50 % for better robustness
- Example: PID control of $G_p(s) = 1/(s+1)^4$:



Lambda method

1. Read deadtime L, time constant T and static gain $K_p = \frac{\Delta y}{\Delta u}$:



Lambda-method

2. Choose λ = desired time constant for closed-loop system
   - $\lambda = T$ common choice
   - $\lambda = 2T$ a bit slower for more robustness

3. PI controller:

$$K = \frac{1}{K_p} \frac{T}{L + \lambda}, \quad T_i = T$$

   PID controller (in serial form):

$$K' = \frac{1}{K_p} \frac{T}{L/2 + \lambda}, \quad T_i' = T, \quad T_d' = \frac{L}{2}$$

Model based tuning (Lab 2)

1. Find process transfer function $G_p(s)$
2. Choose controller type $G_c(s)$
3. Compute closed-loop system transfer function:

$$G(s) = \frac{G_p(s)G_c(s)}{1 + G_p(s)G_c(s)}$$

4. Choose controller parameters to place poles for $G(s)$ to achieve desired behavior (pole placement)