TEKNILLINEN KORKEAKOULU
Systeemitekniikan laboratorio

PID Controllers: Theory, Design and Tuning

Lecture content

- Introduction
- Basics of PID controllers
- Tuning of PID controllers
- Optimization in Matlab
- Auto tuning

PID-controllers: introduction

- By far the most popular controller
- In process control >95 controllers are of PI(D)-type
- Good for linear process control
- Relatively easy to understand (important reason for wide popularity)
- Still many of the PID-control loops are poorly tuned...

Typical paper mill

- Over 2000-500 control loops
- 97 % PI-controllers
- Only 20% of PI-controllers work well decreasing process variability
- Reason for poor performance:
  - 30% poor tuning
  - 30% valve problems
  - 20 % variety of problems (e.g sensor problems, bad choice of sampling rates...)

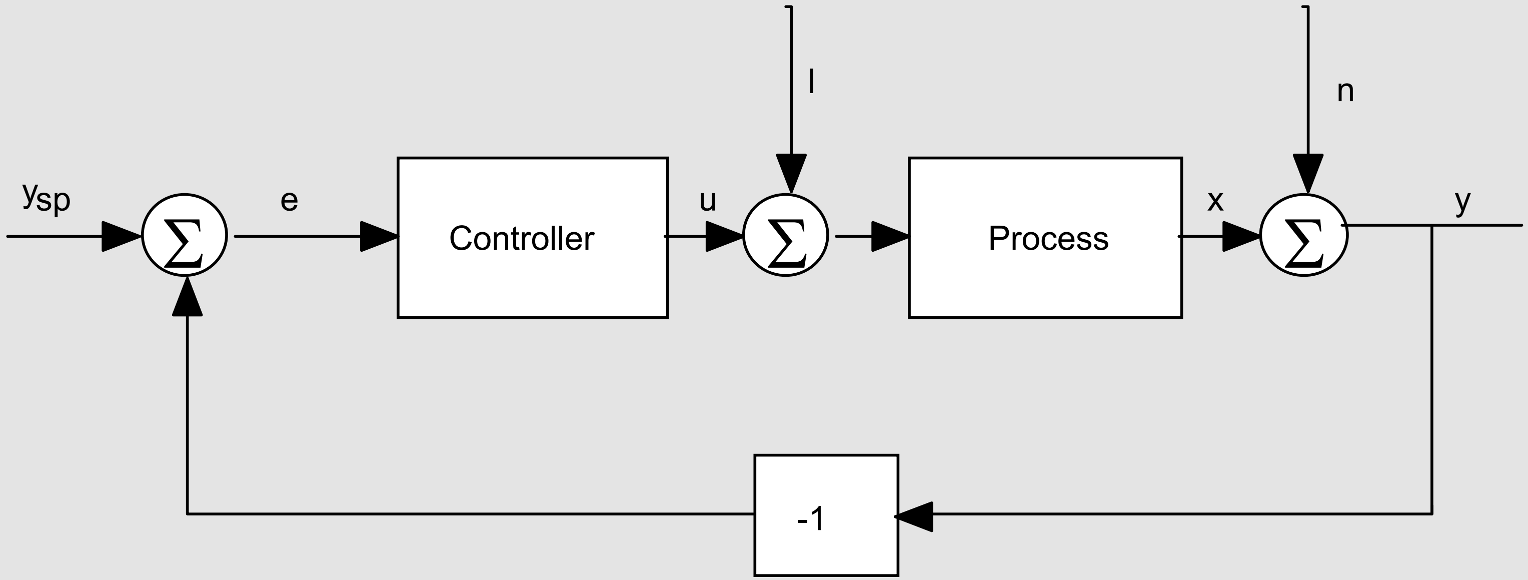
PID-controller

- Today most of the PID controllers are microprocessor based
- DAMATROL MC100: digital single-loop unit controller which is used, for example, as PID controller, ratio controller or manual control station.
- Often PID controllers are integrated directly into actuators (e.g valves, servos)

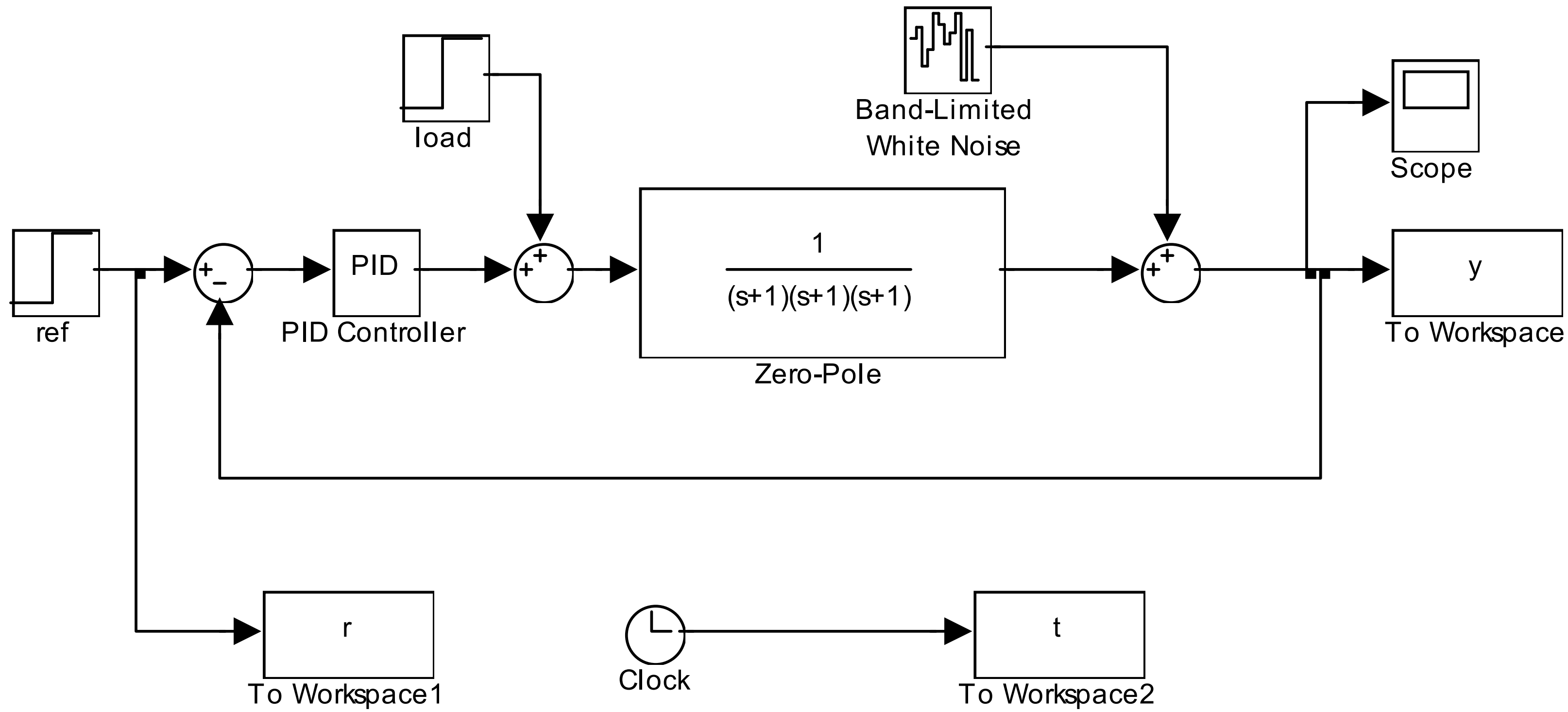
Simple idea of feedback...

- Principle of *negative* feedback:

  ”Increase the manipulated variable when process variable is smaller than the setpoint and decrease the manipulated variable when the process variable is larger than the setpoint”



Simulink model



PID control

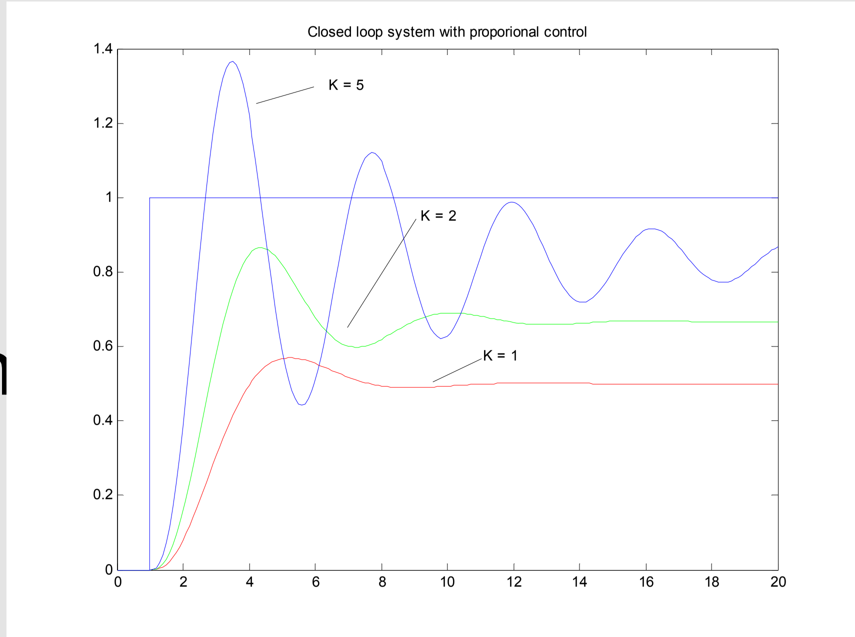
- ”textbook” version of PID
- Control variable *u* is a sum of:
  - *P-term (proportional to e)*
  - *I-term (proportional to integral of e)*
  - *D-term (proportional to derivatice of e)*
- *Controller parameters:*
  - *K = proportional gain*
  - *Ti = integral time*
  - *Td= derivative time*

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

Proportional action

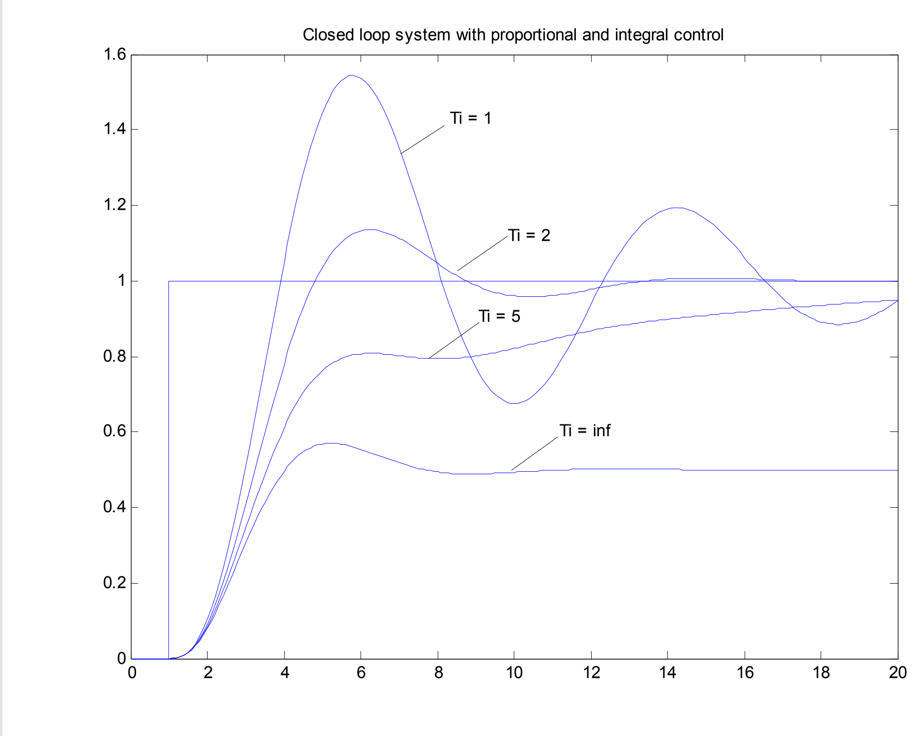
- High value of gain makes the system more insensitive to load disturbance
- Too large a gain makes the system more sensitive to measurement noise
- Steady-state error decreases when gain increases
- Oscillation however often increases

$$G = \frac{1}{(s+1)^3}$$



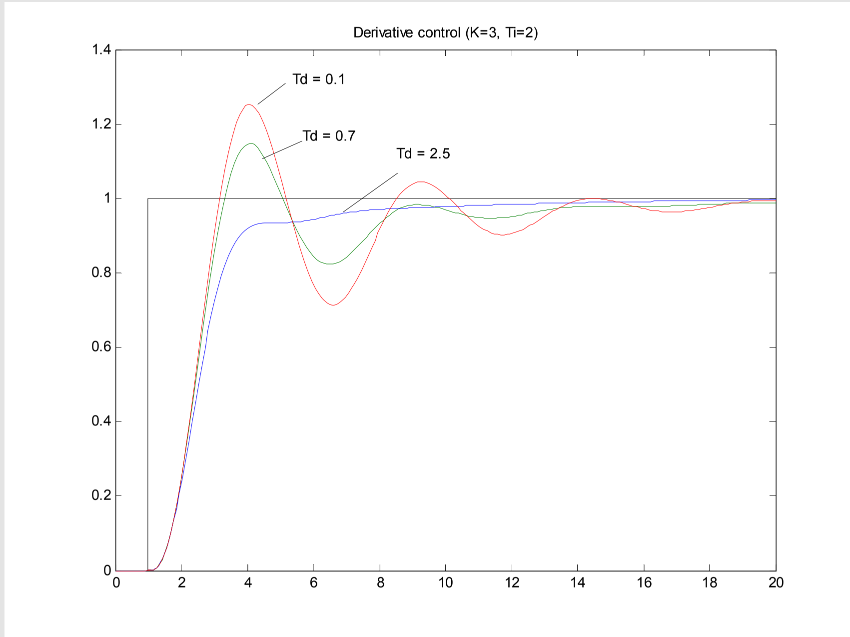
Integral action

- Integral term removes steady state error
- Short integration time often leads to oscillation
- Long integration time common in process control



Derivative action

- Derivative term can *predict* output
- Fast and stable response
- Noise can make derivative control problematic
- Also long delays are problematic when using derivative term



Derivative action

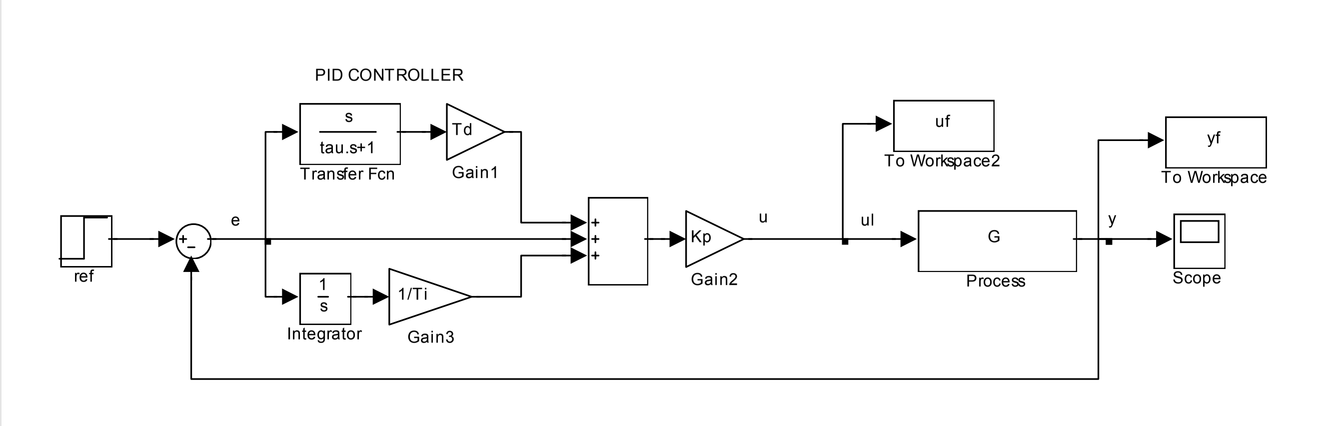
- Fast changes in reference signal result high derivatives→control signal saturates
- Fixes:
  - computing derivatives from process output
  - using filtered derivative term (this is used often in real applications)

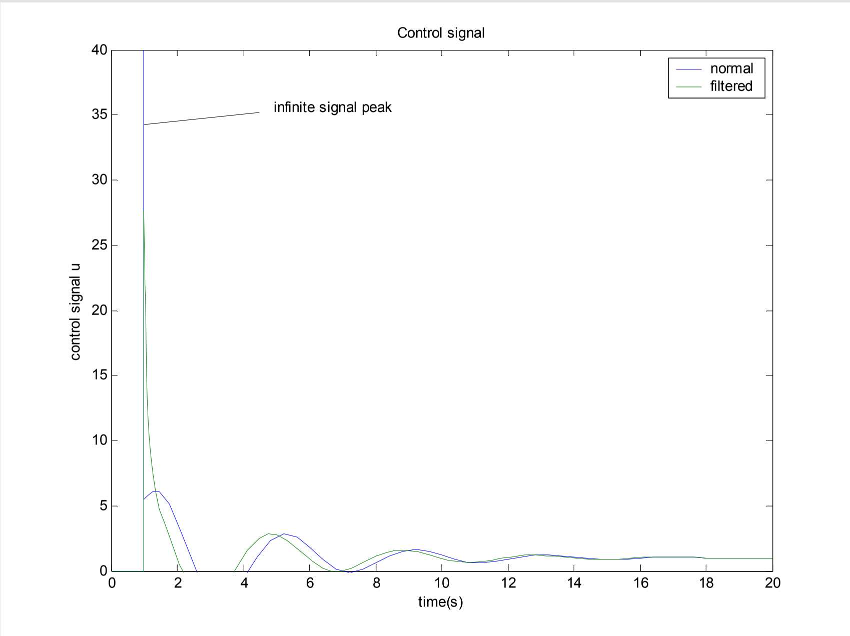
Filtered derivative term:

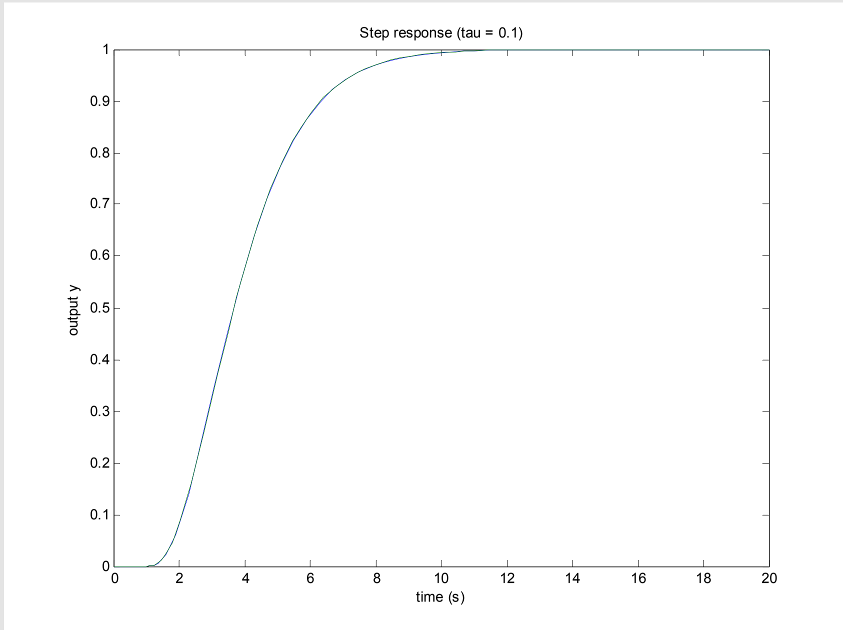
- Benefits:
  - easy to implement in practise
  - more insensitive to noise than normal derivative term
  - corresponds with derivation of low pass filtered signal
  - by choosing tau small
    system has same response as by using normal derivation

$$\begin{cases} G_{s,1}(s) = s \\ G_{s,2}(s) = \dfrac{s}{\tau_s s + 1} \end{cases}, \lim\left\{G_{s,2}(s)\right\} = s = G_{s,1}(s)$$

Filtered derivative

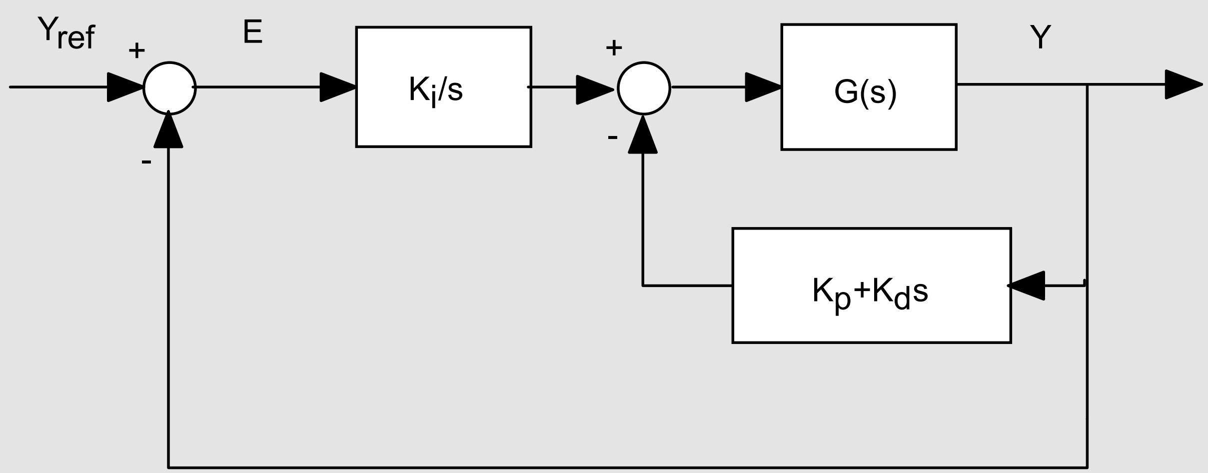






Output derivation – Tachometer feedback

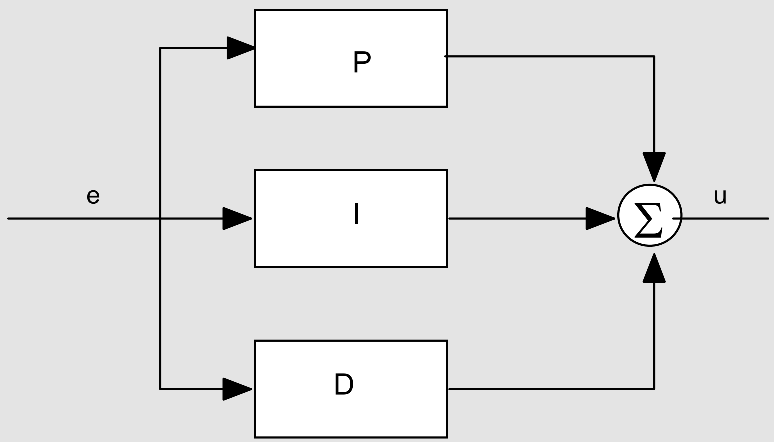
- Use process output for derivation and gain
- No zeros to controlled closed-loop system (prevents overshoots)

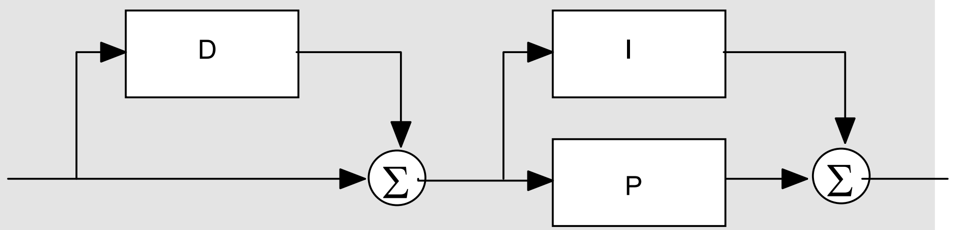


Alternative representations

- Non-interacting & interacting
- Non-interacting more general
- Interacting common in commercial controllers (said to be easier to tune manually)

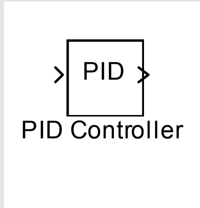
$$G(s) = K\left(1 + \frac{1}{sT_i} + sT_d\right) \qquad G'(s) = K'\left(1 + \frac{1}{sT_i'}\right)\left(1 + sT_d'\right)$$



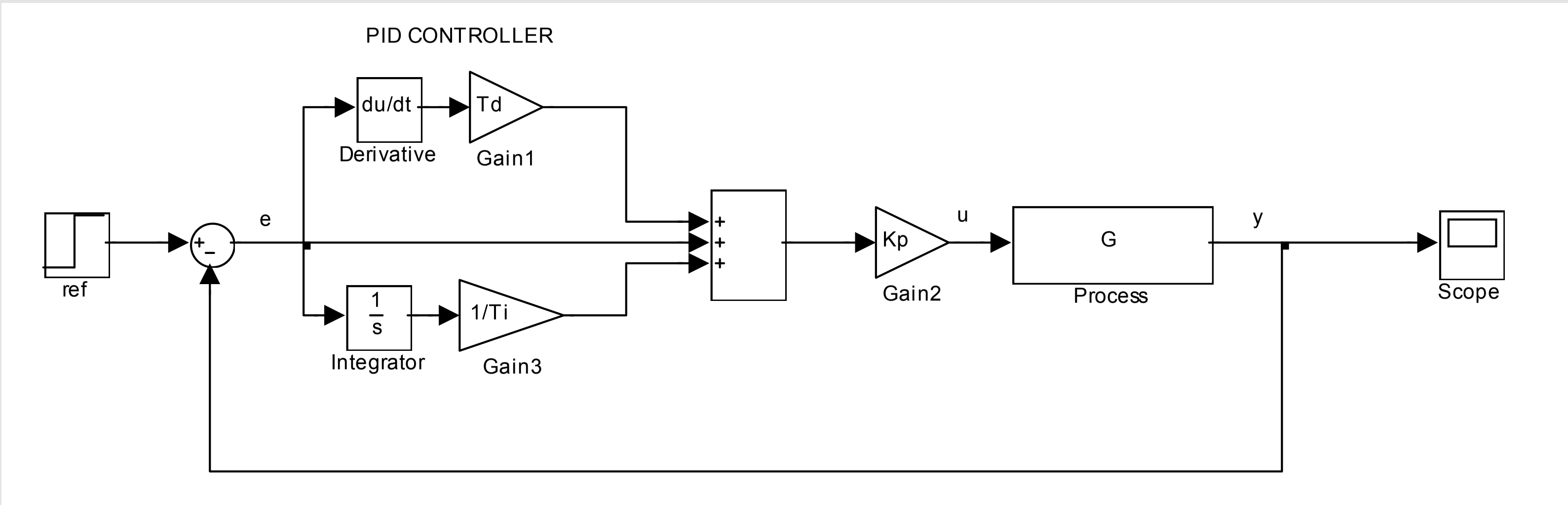


SIMULINK PID-controller

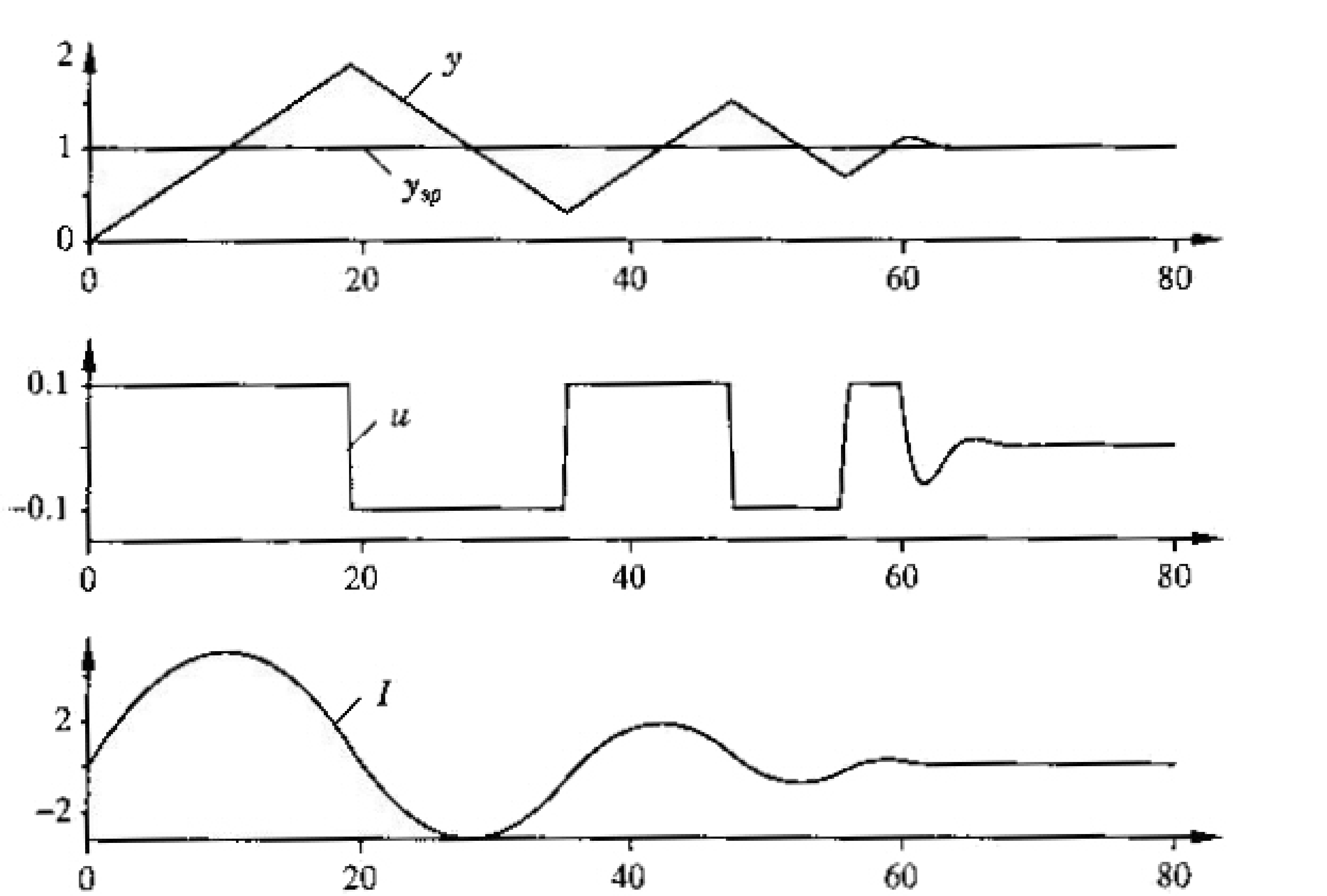
- Simulink PID-block is of form: $$K_p + K_i \frac{1}{s} + T_d s$$



- ”text book” version is:



Integrator windup



Integrator windup

- All actuators have limitations:
  - limited speed
  - valve opening
- If the control variable reaches actuator limits → feedback loop is broken!
- Still error will continue to integrate → very large integral term (”wind up”)
- Large transients when the actuator saturates

Integrator windup

- Integrator action must be stopped when output saturates!
- Solutions:
  - setpoint limitation (limit performance, windup caused by disturbances?)
  - incremental algorihms
  - back calculation and tracking
  - conditional integration

When is PI control sufficient?

- Often derivative action switched off
- Dominant dynamics are of the 1. order
- For example:
  - level control in single tank
  - stirred tank with perfect mixing...
- When tight control not needed

→ PI-control adequate

When is PID control sufficient?

- Dominant dynamics are of the 2. order
- PID speeds up the response versus PI
  - damping improved
  - higher gain can be used to speed up transient response
- For example:
  - temperature control

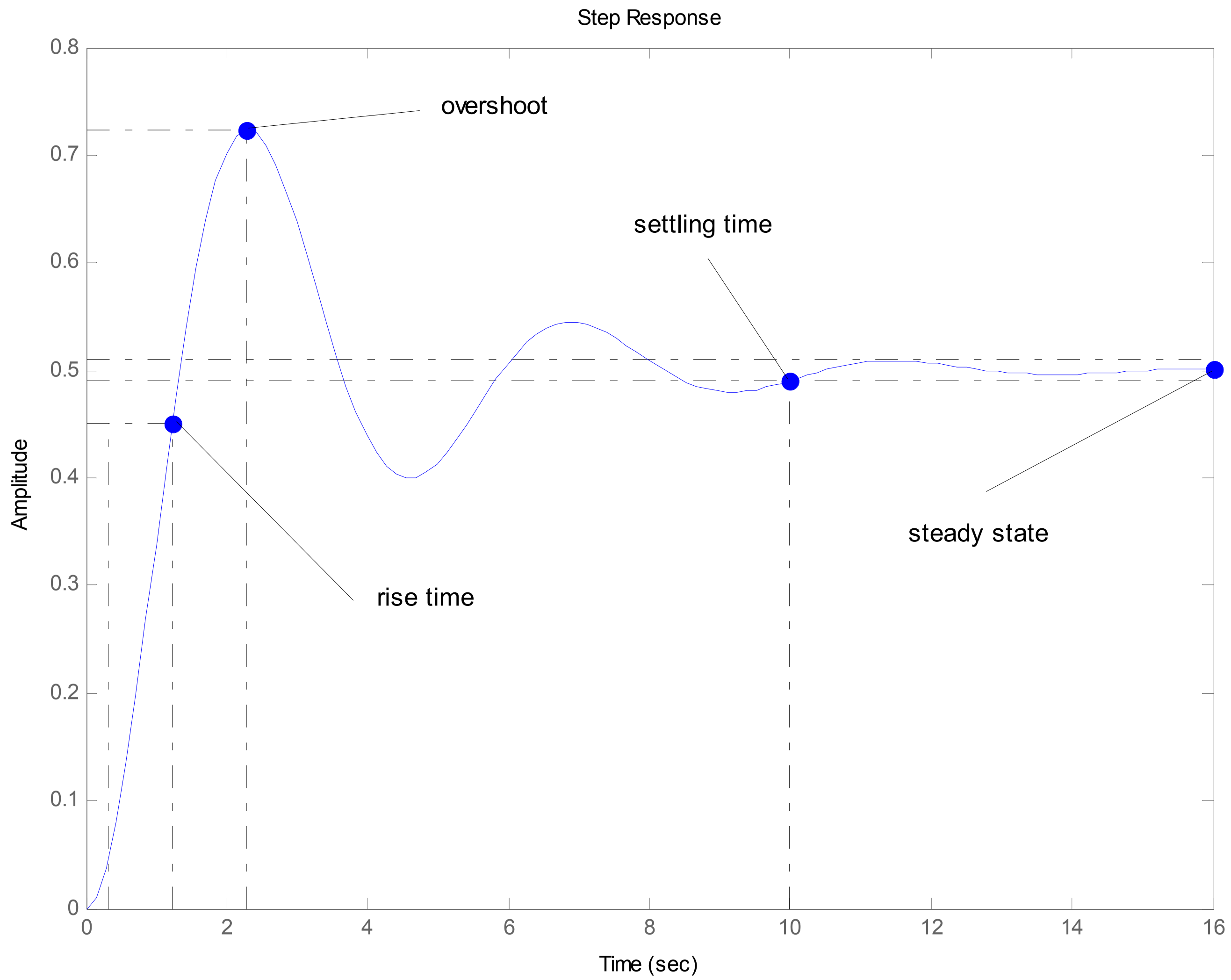
When PID control is insufficient?

- System has high order dynamics
- System is time variant
- Long delays
- Non-linear process
- MIMO/MISO system with strong cross depencies

Controller design

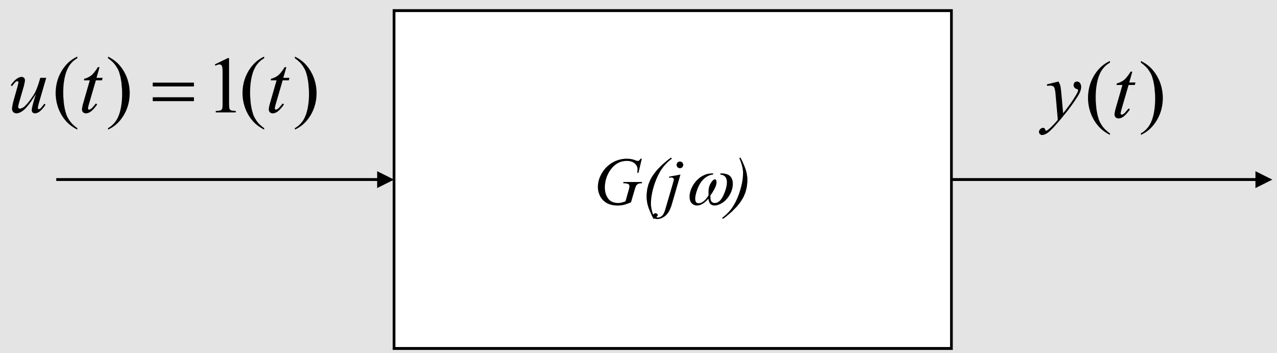
- Problem: how to determine the parameters?
- Tuning is a trade-offs between:
  - load disturbance attenuation
  - effects of measurement noise
  - robustness to process variations
  - response to setpoint change
  - model requirements
  - (computational requirements)

Performance criteria

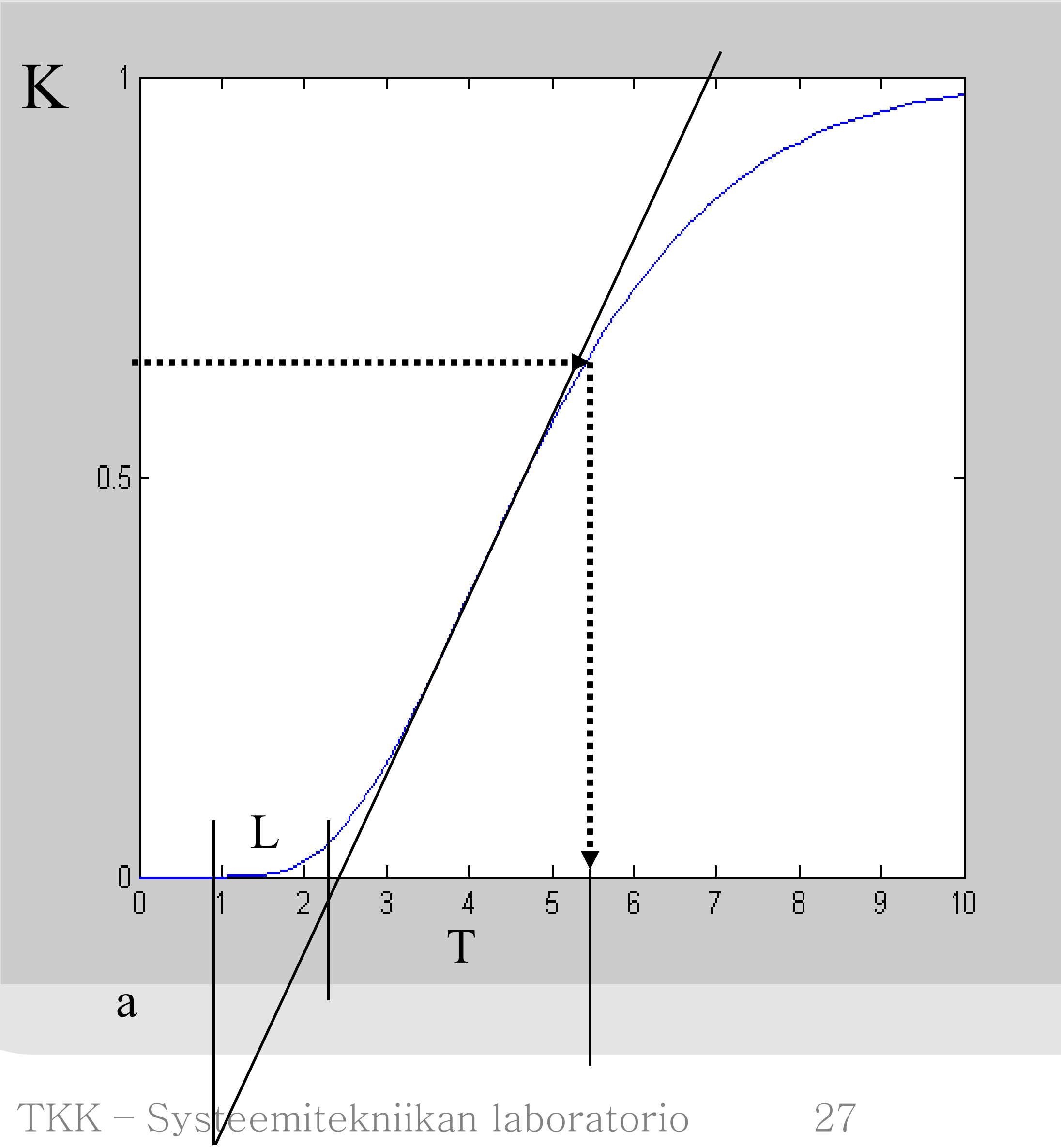


Open loop tuning

- Open loop
- Identify plant dynamics
- Let $u(t)$ be a unit step ('good' test signal)
- Measure output



Open loop tuning



$$G(s) = \frac{K}{1 + sT} e^{-sL}$$

From figure

$K=1$

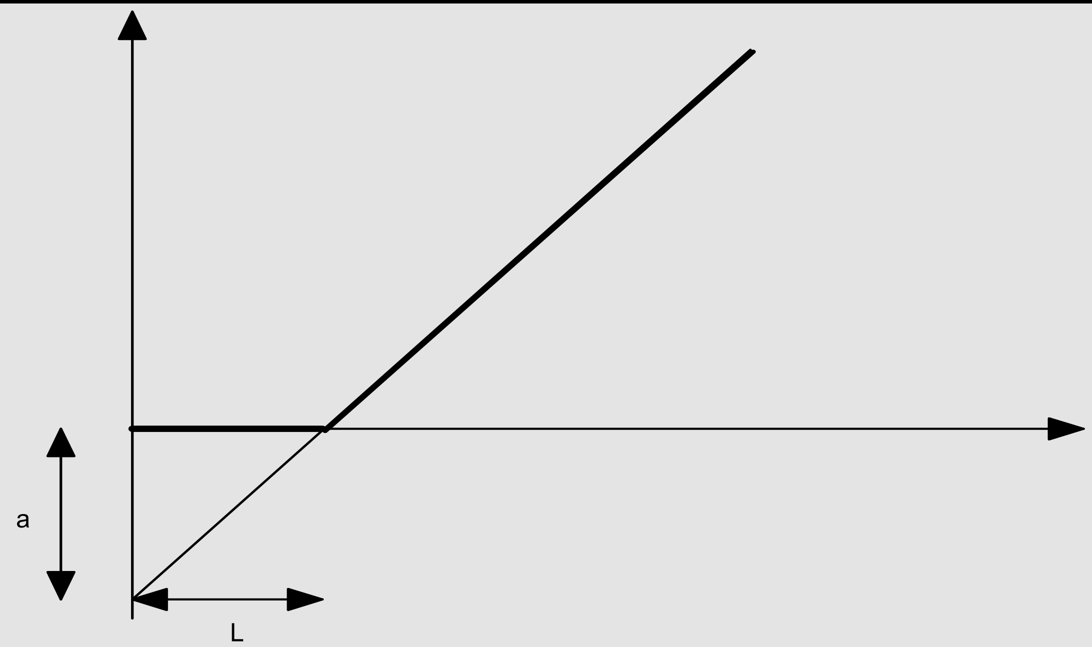
$L=1.3\ s$

$T=4.4\ s$

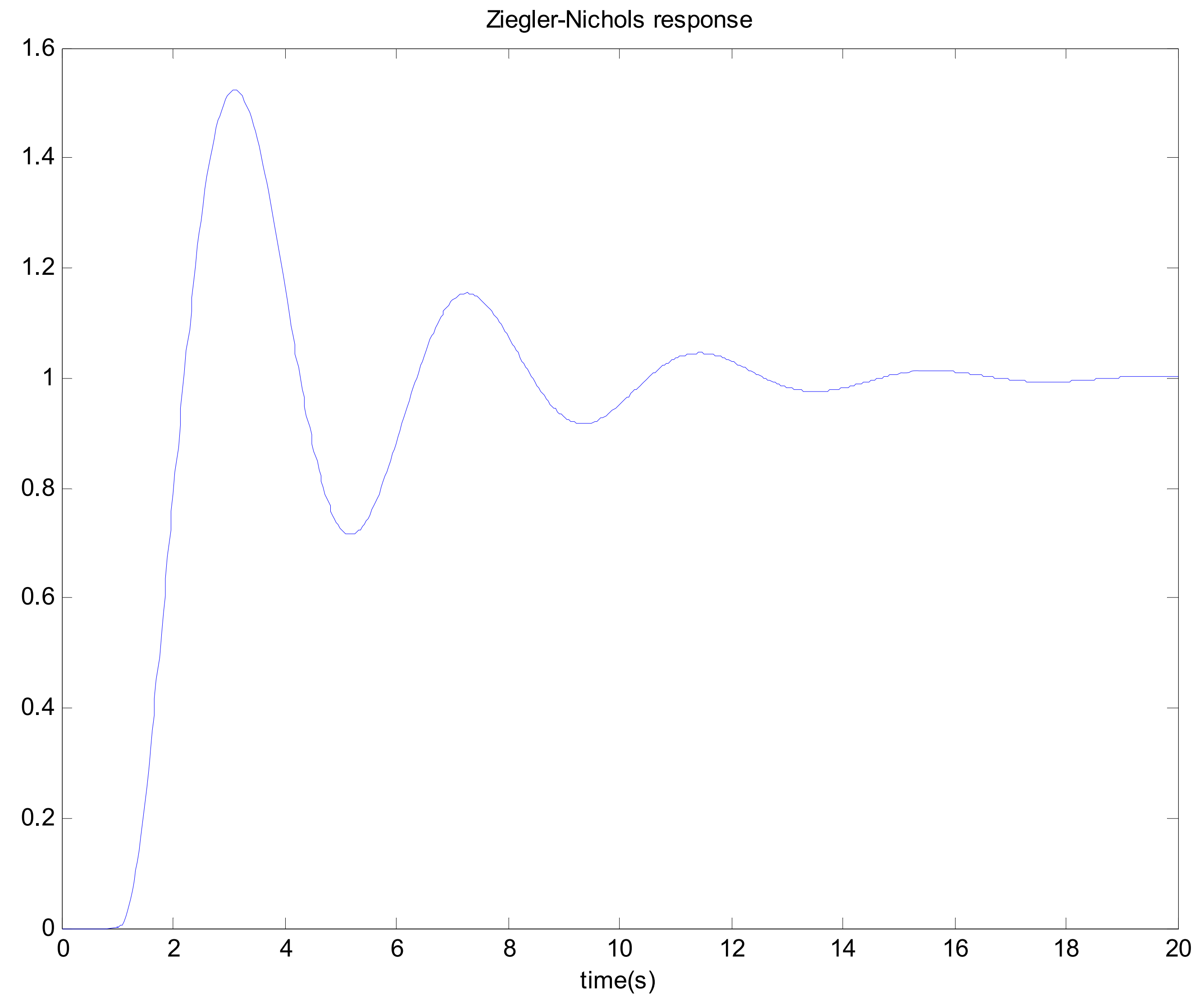
$a=0.4$

Ziegler-Nichols

Controller	K	Ti	Td	Tp
P	*1/a*			*4L*
PI	*0.9/a*	*3L*		*5.7L*
PID	*1.2/a*	*2L*	*L/2*	*3.4L*



Ziegler-Nichols: response



Ziegler-Nichols analysis

- Decay ratio is close to one quarter
- Overshoot is quite large
- Simple and widely used
- Often insufficient→necessary to have more data about process dynamics
- Gives a starting point for fine tuning
- Also frequency response method can be used

Ziegler-Nichols

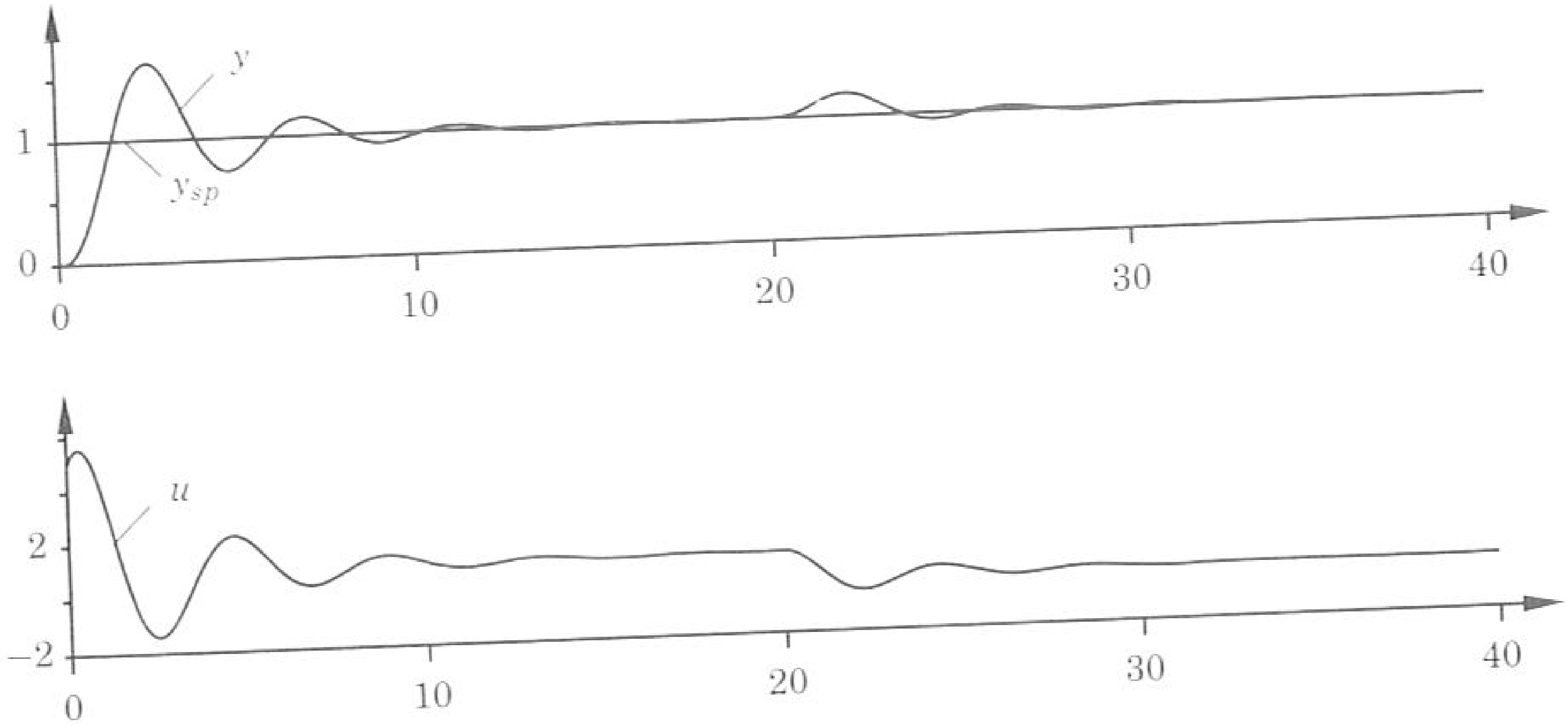


Figure 6.2 Set-point and load disturbance response of a process with transfer function $1/(s+1)^3$ controlled by a PID controller tuned with the Ziegler-Nichols step response method. The diagrams show set point y_{sp}, process output y, and control signal u.

Chien, Hrones & Reswick Method

- Modification of Ziegler-Nichols method
- Better dampled closed-loop step response
- Parameter tables for:
  - quickest response without overshoot
  - quickest response with 20% overshoot
- Different parameters for tuning:
  - setpoint response
  - load disturbance

CHR for setpoint response

- T = time constant

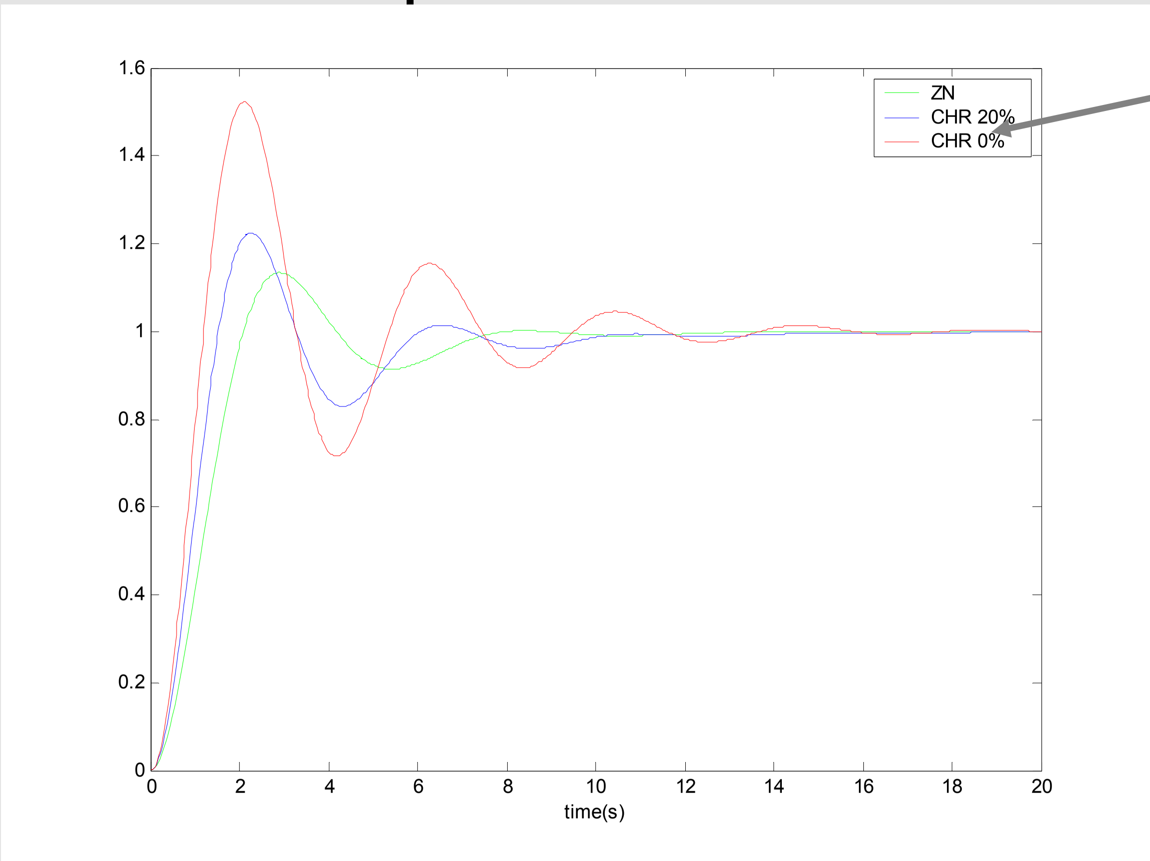
PID Controller parameters;							
Chien, Rhones Reswick setpoint response method							
		0 %			20 %	overshoot	
Control	K	Ti	Td		K	Ti	Td
P	0.3/a				0.7/a		
PI	0.35 a	1.2T			0.6 a	T	
PID	0.6/a	T	0.5L		0.95/a	1.4T	0.47L

CHR for load disturbance resp.

PID Controller parameters;							
Chien, Rhones Reswick load disturbance response method							
		0 %			20 %	overshoot	
Control	K	Ti	Td		K	Ti	Td
P	0.3/a				0.7/a		
PI	0.6 a	4L			0.7 a	2.3L	
PID	0.95/a	2.4L	0.42L		1.2/a	2L	0.42L

Z-N vs. CHR

- Overshoot exist but response is much better when CHR parameters are used



väärin

$$G(s) = \frac{1}{(s+1)^3}$$

Analytical tuning methods

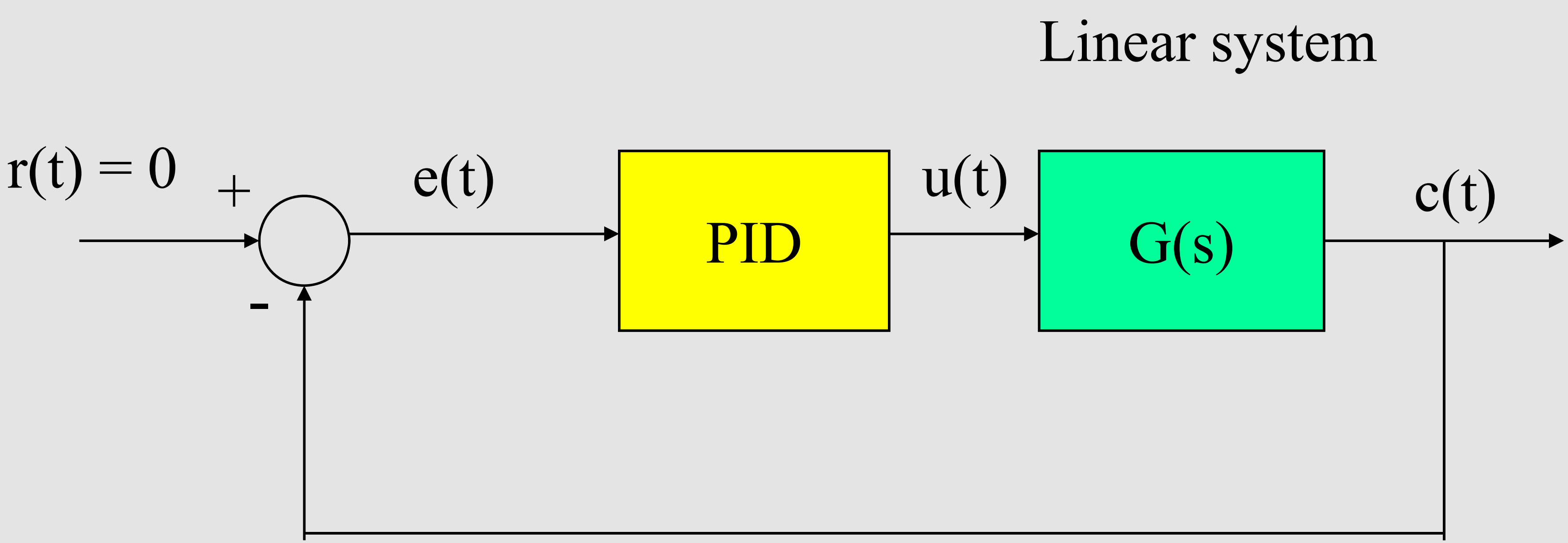
G_p (process transfer function)

G_c (controller transfer function)

$$\rightarrow \mathrm{G}_0 = \frac{G_p G_c}{1 + G_p G_c} \quad \text{(closed loop transfer function)}$$

$$\rightarrow \mathrm{G}_c = \frac{1}{G_p} \cdot \frac{G_0}{1 - G_0} \quad \text{(closed loop transfer function)}$$

Closed-loop tuning



1. Set $T_i = \infty$ and $T_d = 0$.
2. Increase K_p until the system oscillates to obtain critical gain K_{cr} and critical period T_{cr} *(or frequency)*

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

Closed-loop tuning

PID Controller parameters;			
Ziegler-Nichols frequency response method			
Control	K	Ti	Td
P	0.5Kcr=2.3		
PI	0.4Kcr=1.9	0.8Tcr=1.8	
PID	0.6Kcr=2.8	0.5Tcr=1.1	0.125Tcr=0.3

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

Frequency response method

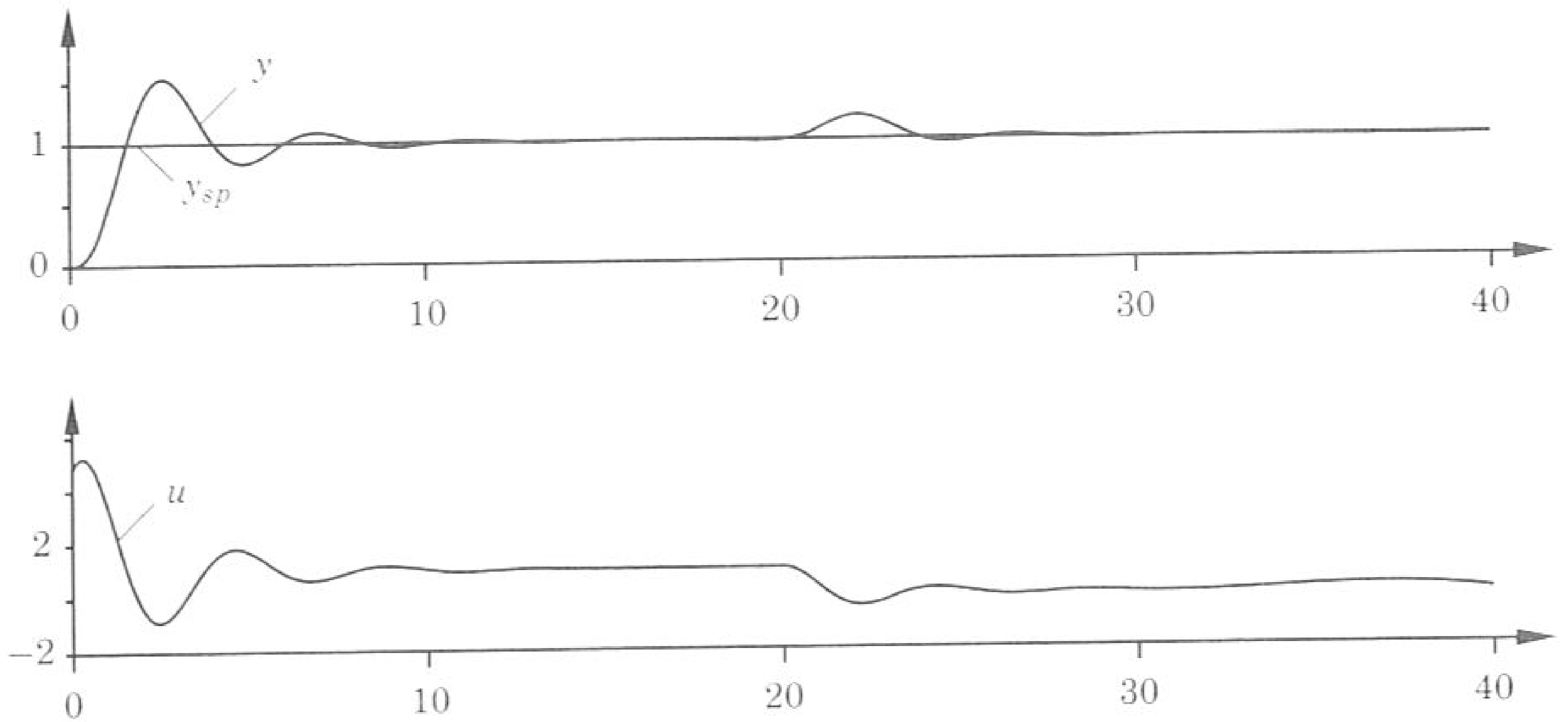


Figure 6.3 Set-point and load disturbance response of a process with the transfer function $1/(s+1)^3$ controlled by a PID controller that is tuned with the Ziegler-Nichols frequency response method. The diagrams show set point y_{sp}, process output y, and control signal u.

Frequency response interpretation

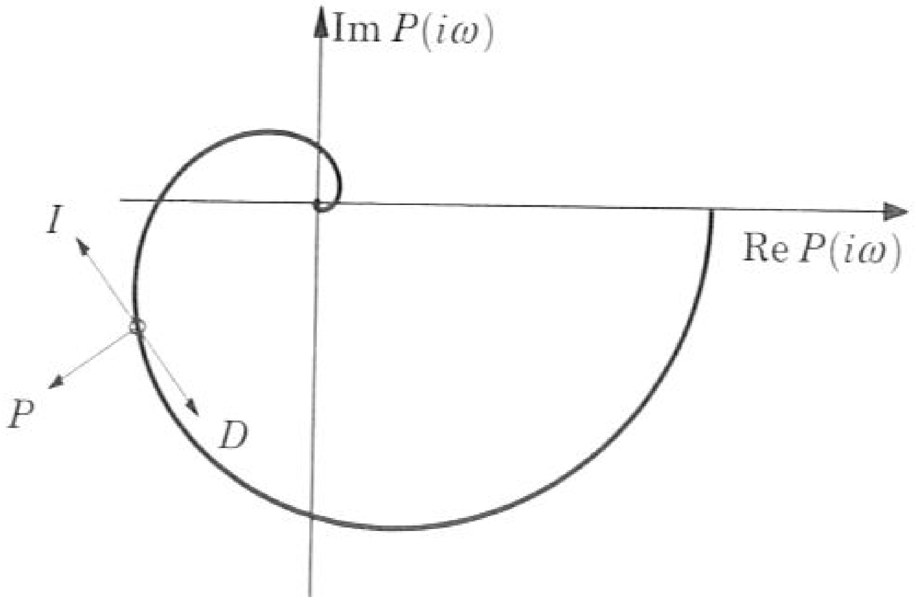


Figure 6.4 Illustrates that a point on the Nyquist curve of the process transfer function may be moved to another position by PID control. The point marked with a circle may be moved in the directions $P(i\omega)$, $-iP(i\omega)$, and $iP(i\omega)$ by changing the proportional, integral, and derivative gain, respectively.

Frequency response – phase margin

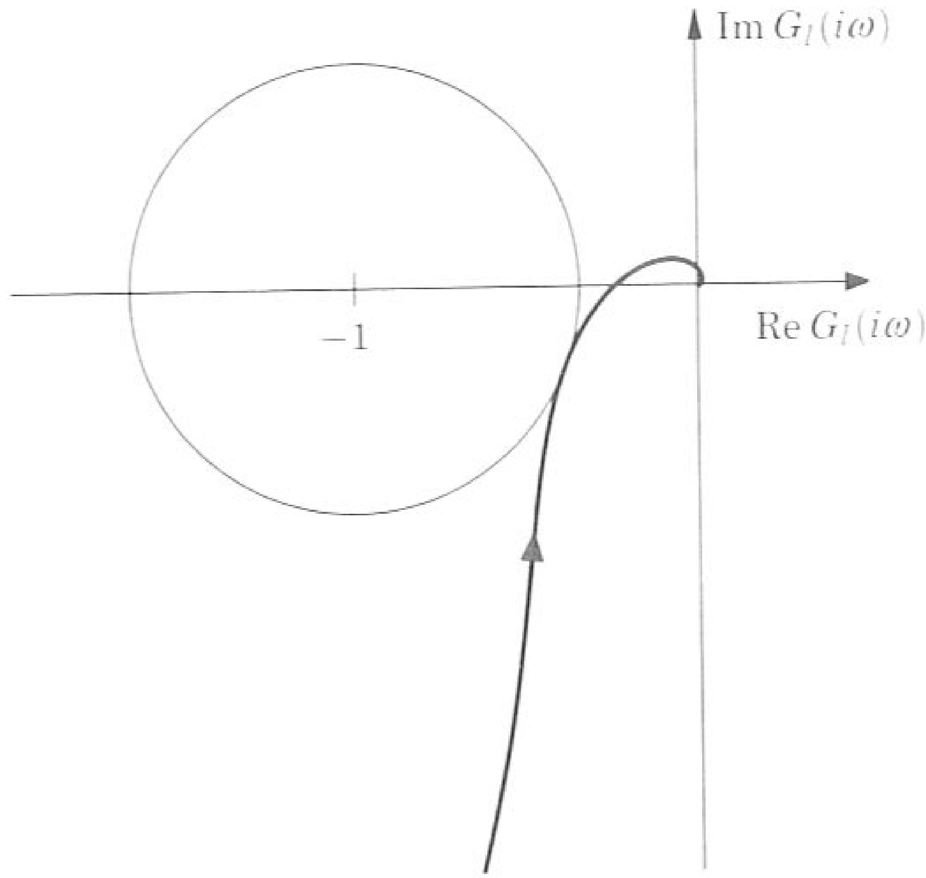


Figure 6.5 Nyquist plot for the loop transfer function G_l for PI control of the process $P(s) = e^{-\sqrt{s}}$. The controller was designed to give the phase margin of 60°.

Comparison

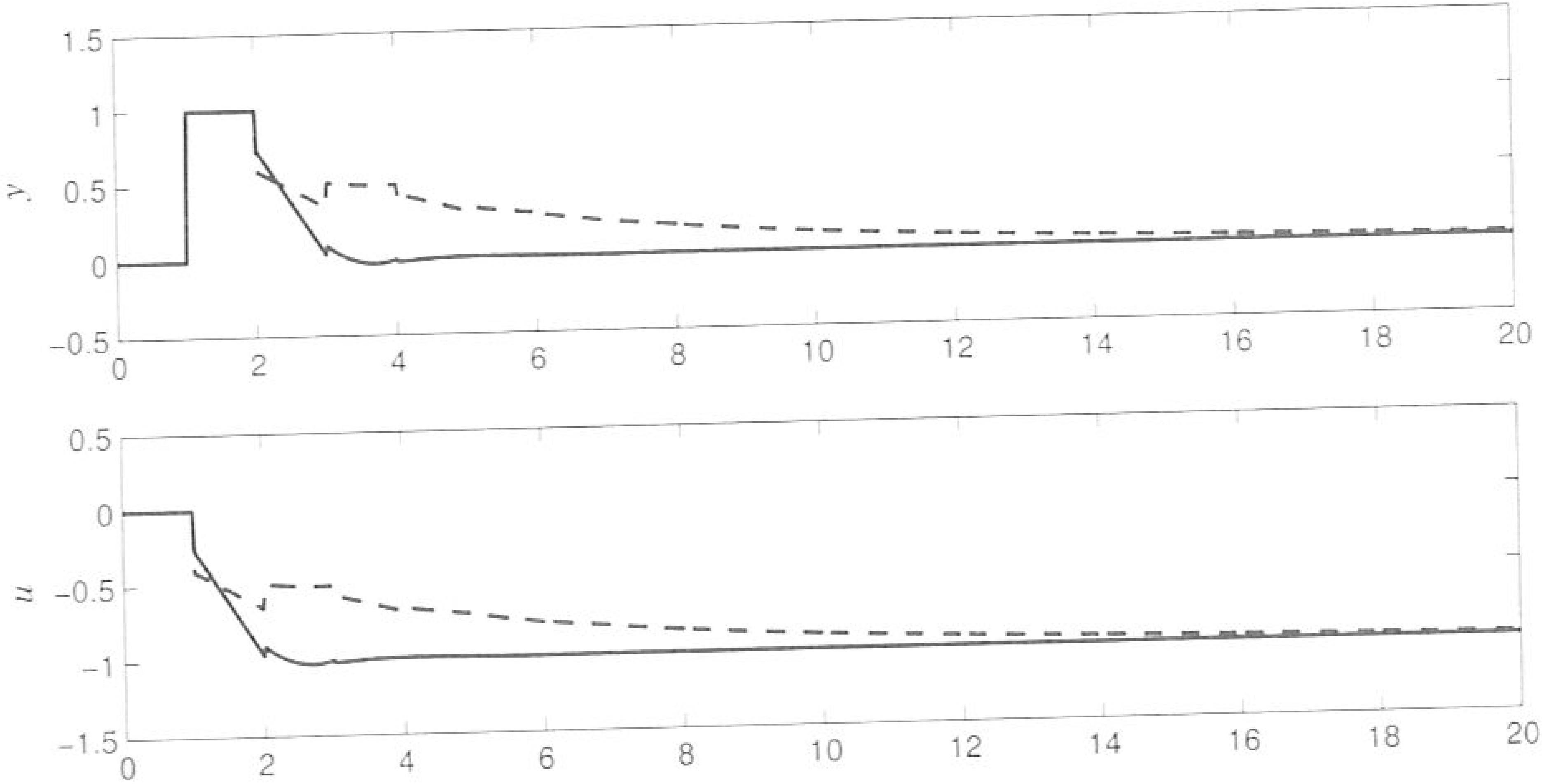
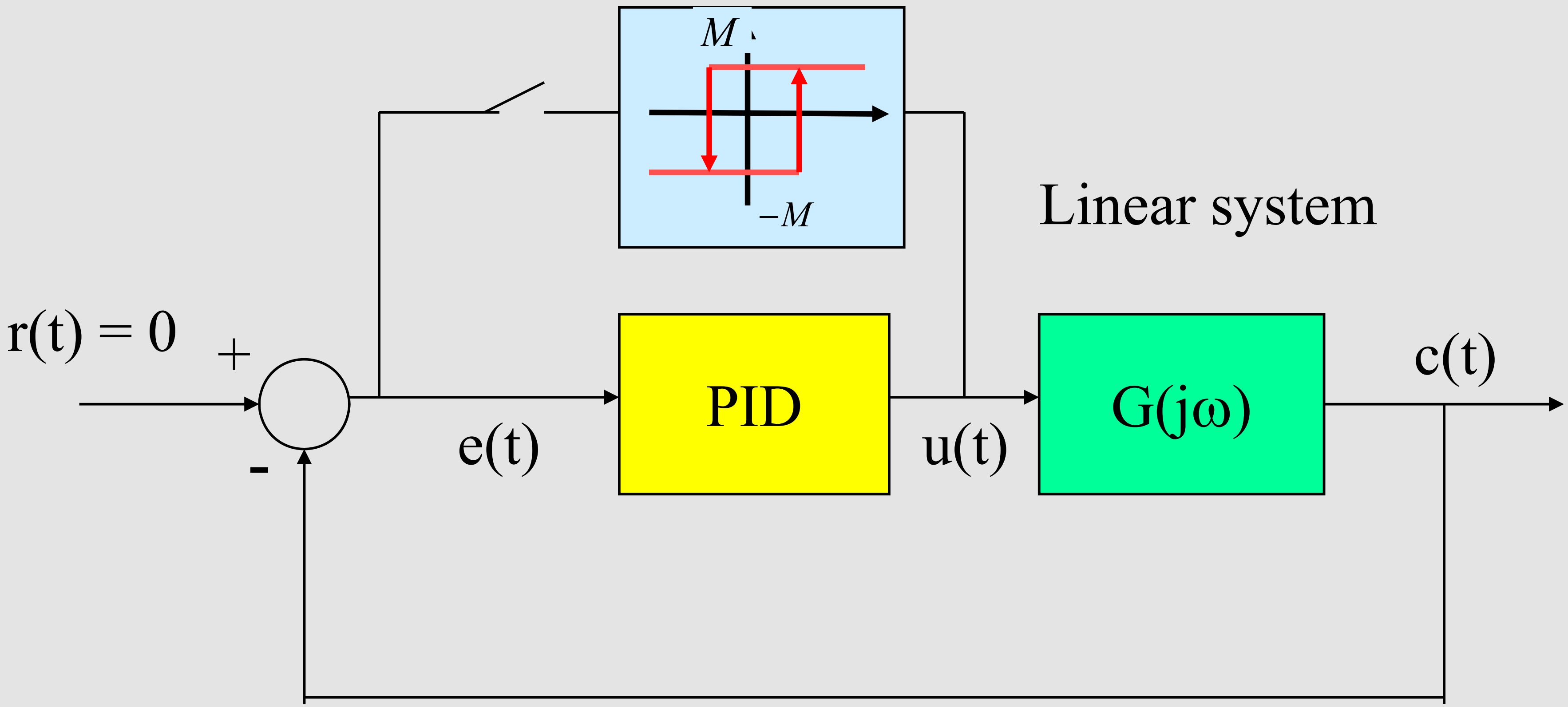


Figure 6.6 Responses to a load disturbance for a process with pure delay ($L = 1$) with PI controllers tuned by Ziegler-Nichols frequency response method (dashed) and a proper method (solid).

Relay tuning

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



First relay on, to obtain critical gain K_{cr} and critical period T_{cr}

Rule-based empirical tuning

- Increasing proportional gain decreases stability
- Error decays more rapidly if integration time is decreased
- Decreasing integration time decreases stability
- Increasing derivative time improves stability

Tuning maps

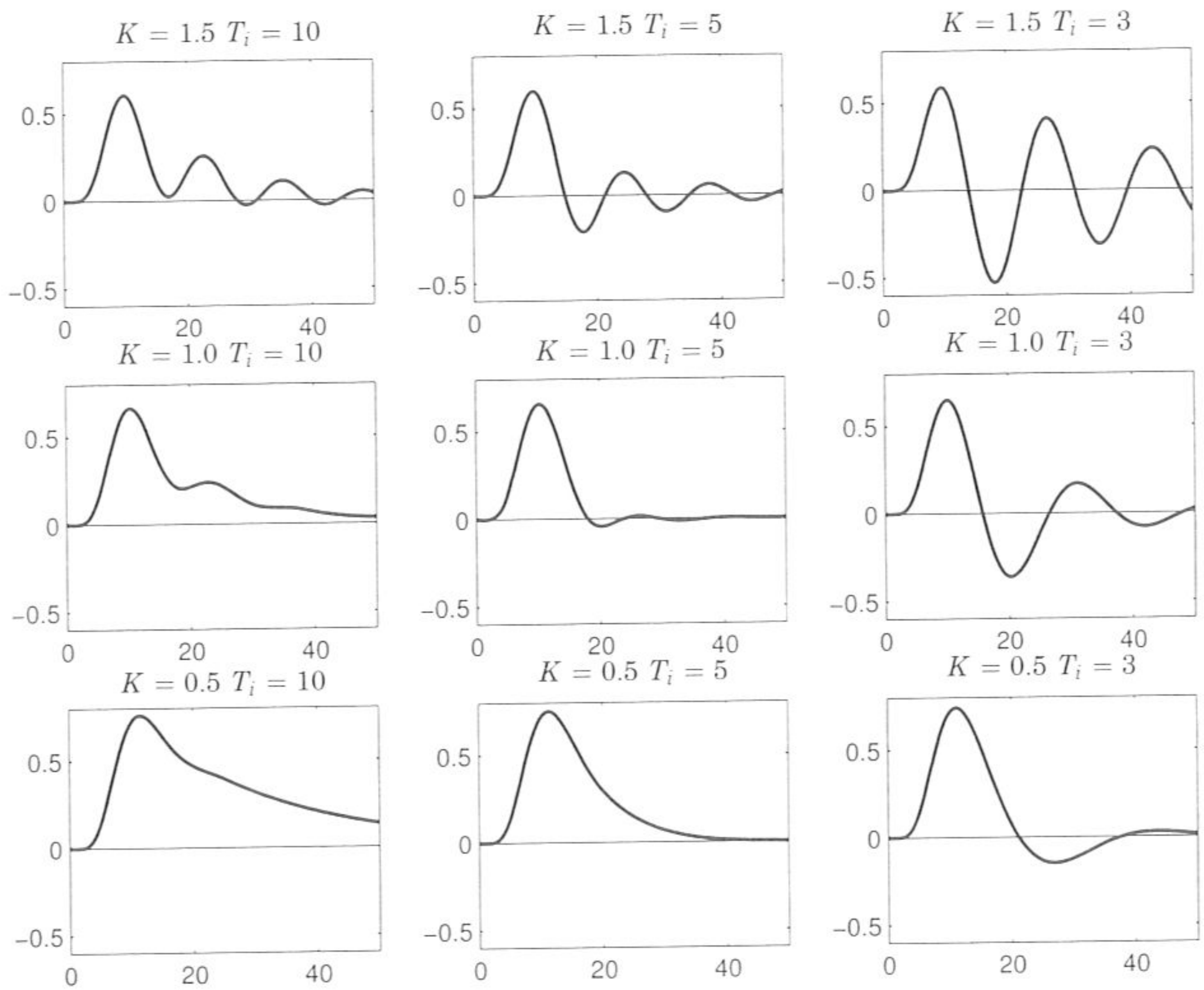


Figure 6.7 Tuning map for PID control of a process with the transfer function $P(s) = (s+1)^{-8}$. The figure shows the responses to a unit step disturbance at the process input. Parameter T_d has the value 1.9.

Tuning map

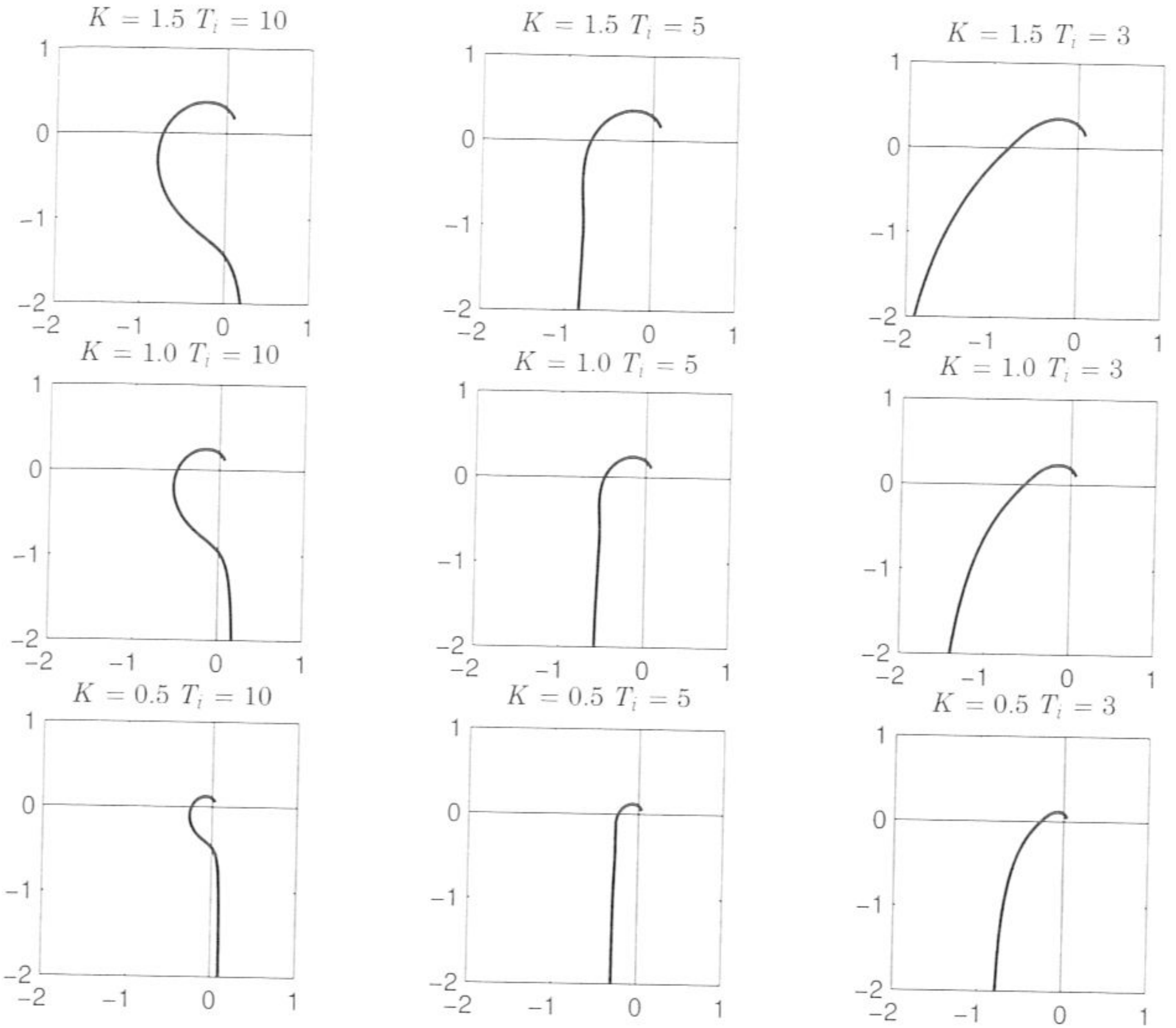


Figure 6.8 Tuning map for PID control of a process with the transfer function $P(s) = (s+1)^{-8}$. The figure shows the Nyquist plots of the loop transfer functions. Parameter T_d has the value 1.9.

Counterintuitive

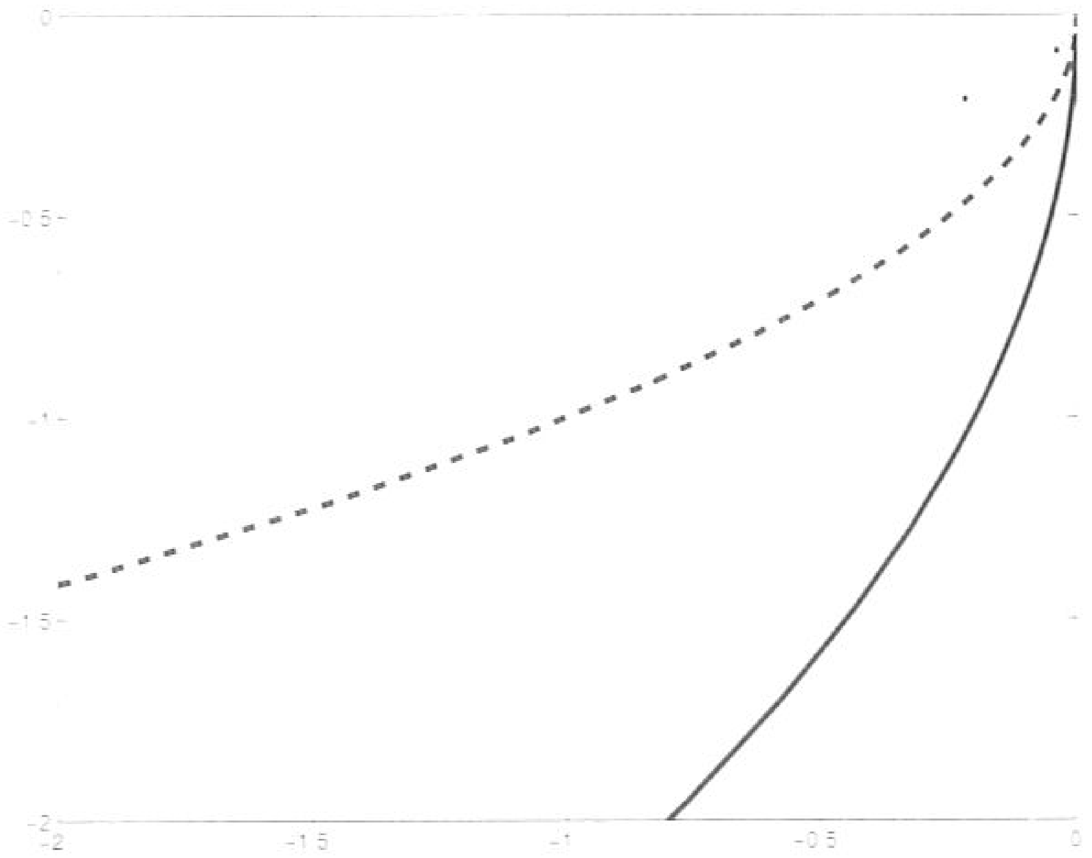


Figure 6.9 Nyquist curves for the loop transfer functions for an integrator with PI control. Integration time T_i is constant, and the gain has the values $K = 0.2$ (dotted), 1 (dashed), and 5 (solid). Notice the counterintuitive behavior that phase margin increases with increasing controller gain.

Pole Placement

- Process must be modelled
- Define the desired closed -loop poles
- PID-controller gives desired closed-loop poles
- Process parameters max. 3→poles can be set freely using PID-controller

Pole Placement: example

- Process: $$G_p = \frac{K_p}{(1+sT)}$$
- Controller (PI): $$G_c = K(1+\frac{1}{sT_1})$$
- Closed loop system: $$G(s) = \frac{G_p G_c}{1+G_p G_c}$$
- Characteristic eq: $$1+G_c G_p = 0$$

→ $$s^2 + s\frac{1+K_p K}{T} + \frac{K_p K}{TT_i} = 0$$

Pole placement: example

- Desired close-loop poles characterized by relative damping (ζ) and frequency (ω):
- By making coefficient equal in characteristics equations and solving controller parameters gives:

$$s^2 + 2\xi\omega_0 s + \omega_0{}^2 = 0$$

$$K = \frac{2\xi\omega_0 T - 1}{K_p}$$

$$T_i = \frac{2\xi\omega_0 T - 1}{\omega_0{}^2 T}$$

Gang of six

$$\frac{PC}{1+PC} = \frac{(2\zeta\omega_0 - 1/T)s + \omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \qquad \frac{C}{1+PC} = \frac{K(s+1/T_i)(s+1/T)}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$$

$$\frac{P}{1+PC} = \frac{K_p s/T}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \qquad \frac{1}{1+PC} = \frac{s(s+1/T)}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$$

$$\frac{PC_{ff}}{1+PC} = \frac{b(2\zeta\omega_0 - 1/T)s + \omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \qquad \frac{C_{ff}}{1+PC} = \frac{K(bs+1/T_i)(s+1/T)}{s^2 + 2\zeta\omega_0 s + \omega_0^2}.$$

- Largest value of transfer function from load disturbance to process output

$$\max_{\omega} |G_{xd}(i\omega)| = \max_{\omega} \left| \frac{P(i\omega)}{1 + P(i\omega)C(i\omega)} \right| = \frac{K_p}{\omega_0 T \min(1,\zeta)}.$$

Choosing ω_o

$$\frac{1}{2\zeta} \leq \omega_0 T < \min\left(\frac{0.25}{T_e}, \frac{1 + K_p K_{max}}{2\zeta}\right).$$

- More sensitive when $\omega_o T$ is small

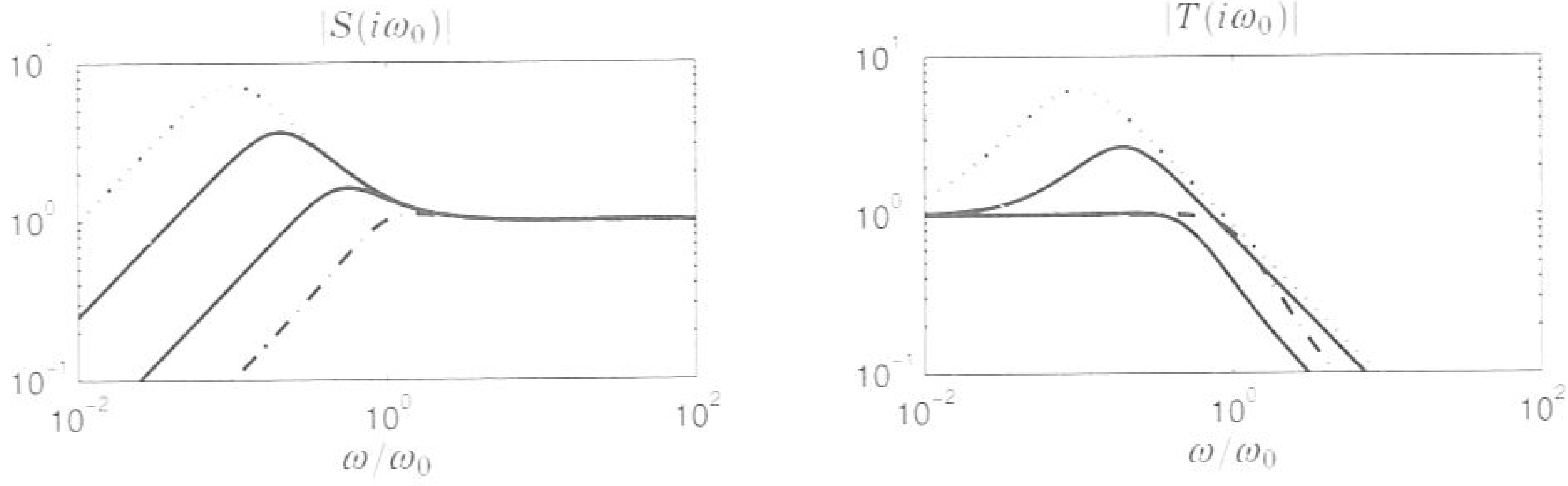


Figure 6.10 Gain curves of the sensitivity functions for $\zeta = 0.7$ and $\omega_0 T = 0.1, 0.2, 0.5,$ and 1. The dotted curve corresponds to $\omega_0 L = 0.1$ and the dash-dotted curve to $\omega_0 L = 1$.

Approximate models

- First order approx

$$P(s) = \frac{1}{1 + 1.26s}$$

$$P(s) = \frac{1}{(1+s)(1+0.2s)(1+0.05s)(1+0.01s)}.$$

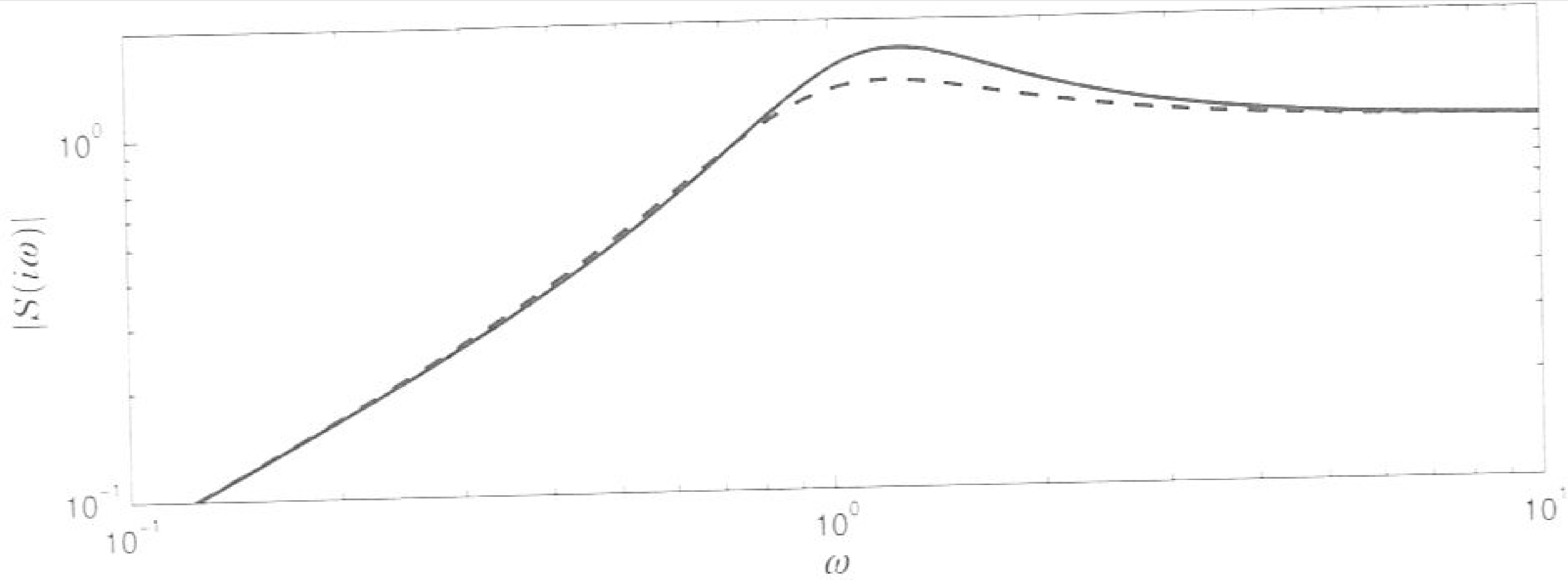


Figure 6.11 Sensitivity functions for the approximate system (dashed) and the true system in Example 6.11.

Approximate models

$$P(s) = \frac{1}{(1+s)(1+0.26s)}$$

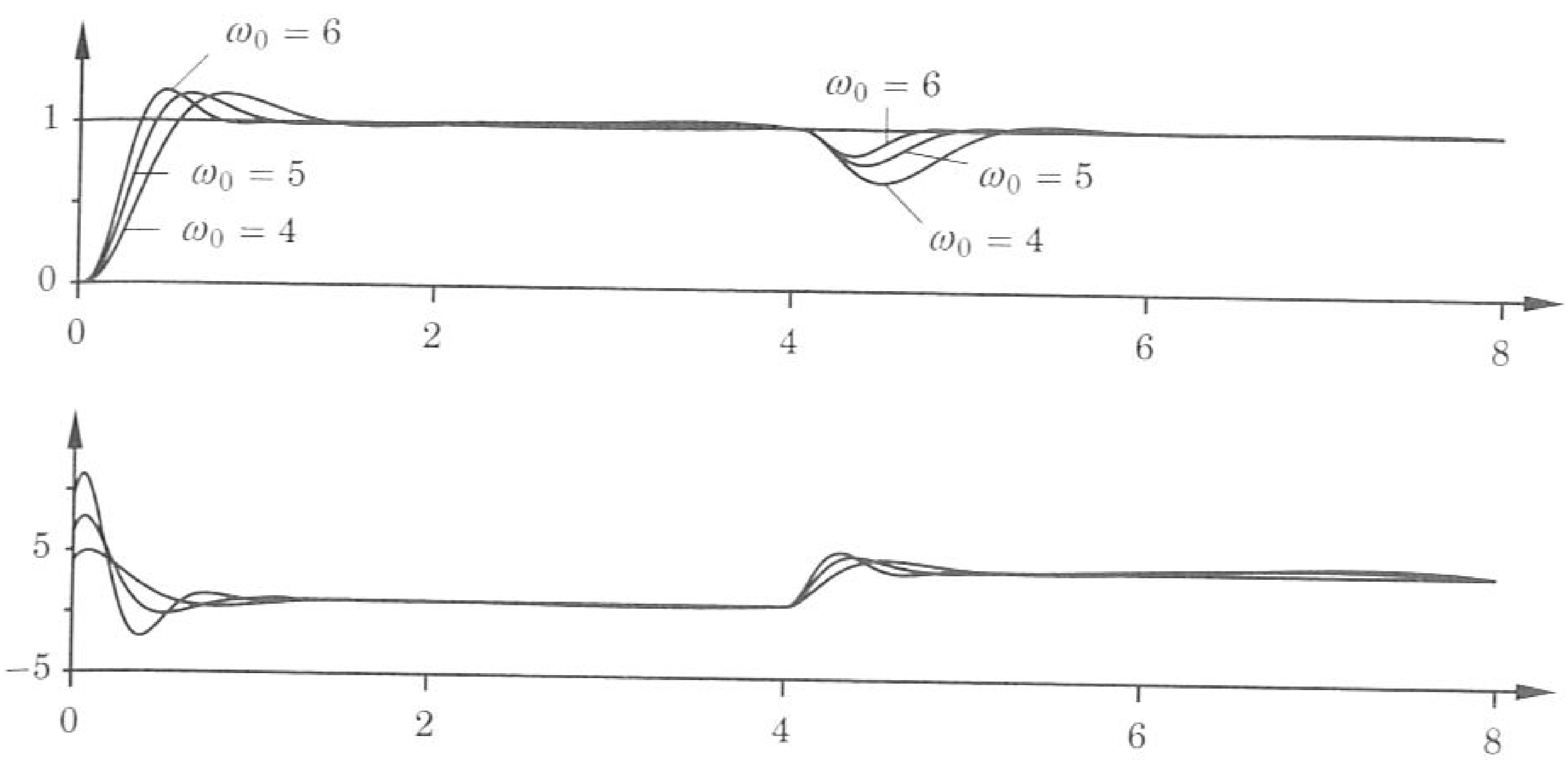


Figure 6.12 Set-point and load disturbance responses of the process with two poles controlled by a PID controller tuned according to Example 6.12. The responses for $\omega_0 = 4, 5$, and 6 are shown. The upper diagram shows set point $y_{sp} = 1$ and process output y, and the lower diagram shows control signal u.

Optimization in Matlab

- Minimize the cost function by simulating the controlled model
- Use *fminsearch* function to find optimized parameters for the PID controller
- Choose appropriate cost function carefully
- Don’t forget local minima

Analytical tuning methods

- λ –Tuning
- processes with long dead time
- desired closed-loop tf: $$G_0 = \frac{e^{-sL}}{1 + s\lambda T}$$
- controller transfer function: $$G_c = \frac{1 + sT}{K_p(1 + s\lambda T - e^{-sL})}$$
- λ < 1 faster response (smaller time constant)

Optimization in Matlab

1. Guess initial values for K_p^0, K_i^0, K_d^0, Set $n = 0$
2. Solve ***y****(t)* using simulation (SIMULINK) and also the value of cost function $J\left(K_p^n, K_i^n, K_d^n\right)$
3. Use optimization algorithm (*FMINSEARCH or FMINUNC*) to update the control parameters:

$$K_p^{n+1} = K_p^n + correction \text{ (depends on optimization algorithm)}$$

$$K_i^{n+1} = K_i^n + correction \text{ (depends on optimization algorithm)}$$

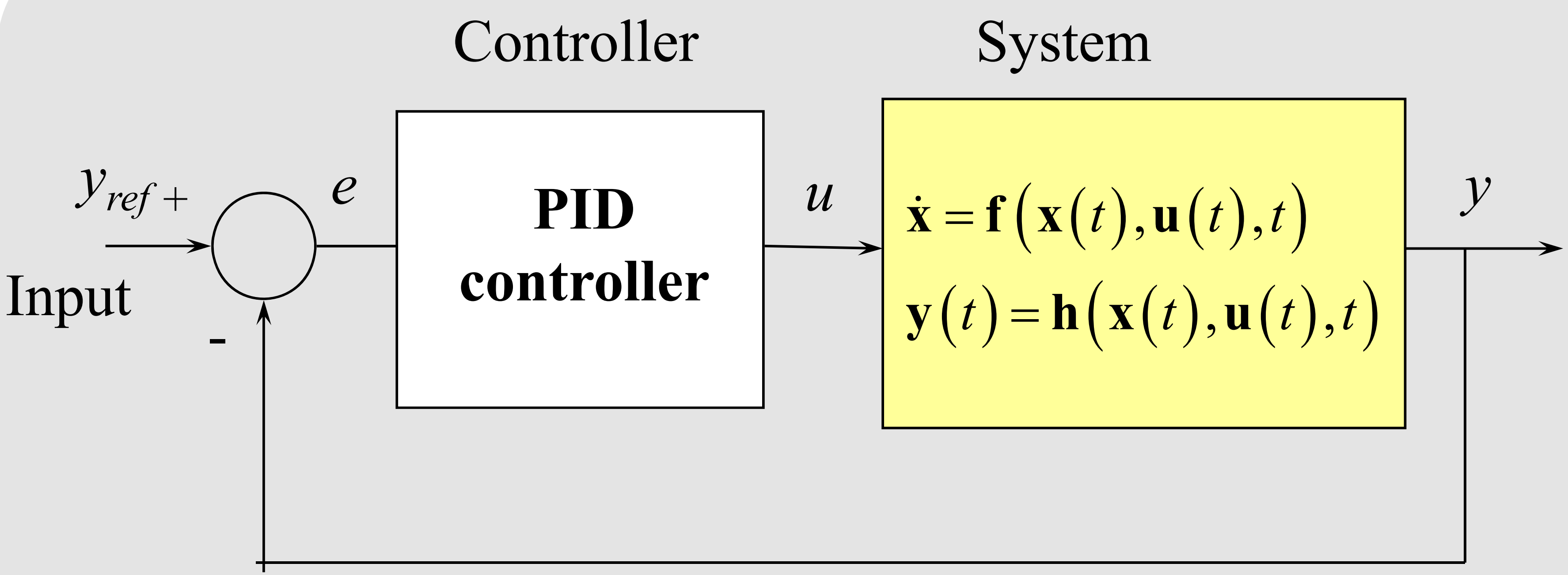
$$K_d^{n+1} = K_d^n + correction \text{ (depends on optimization algorithm)}$$

4. Check if minimum is reached

$$\left|J^{n+1} - J^n\right| < tolerance$$

If not, set *n* = *n+1* and go back to 2.

Optimization in Matlab



$$u(t) = K_p e(t) + K_i \int_0^t e(\alpha) d\alpha + K_d \frac{de}{dt}$$

PID controller

Parameters K_p, K_i, and K_d

Parameter vector $\mathbf{p} = [K_p, K_i, K_d]^T$

$$= [K_p, K_i, K_d] \begin{bmatrix} e(t) \\ \int_0^t e(\alpha) d\alpha \\ \frac{de}{dt} \end{bmatrix} = \mathbf{p}^T \begin{bmatrix} e(t) \\ \int_0^t e(\alpha) d\alpha \\ \frac{de}{dt} \end{bmatrix}$$

Cost criteria

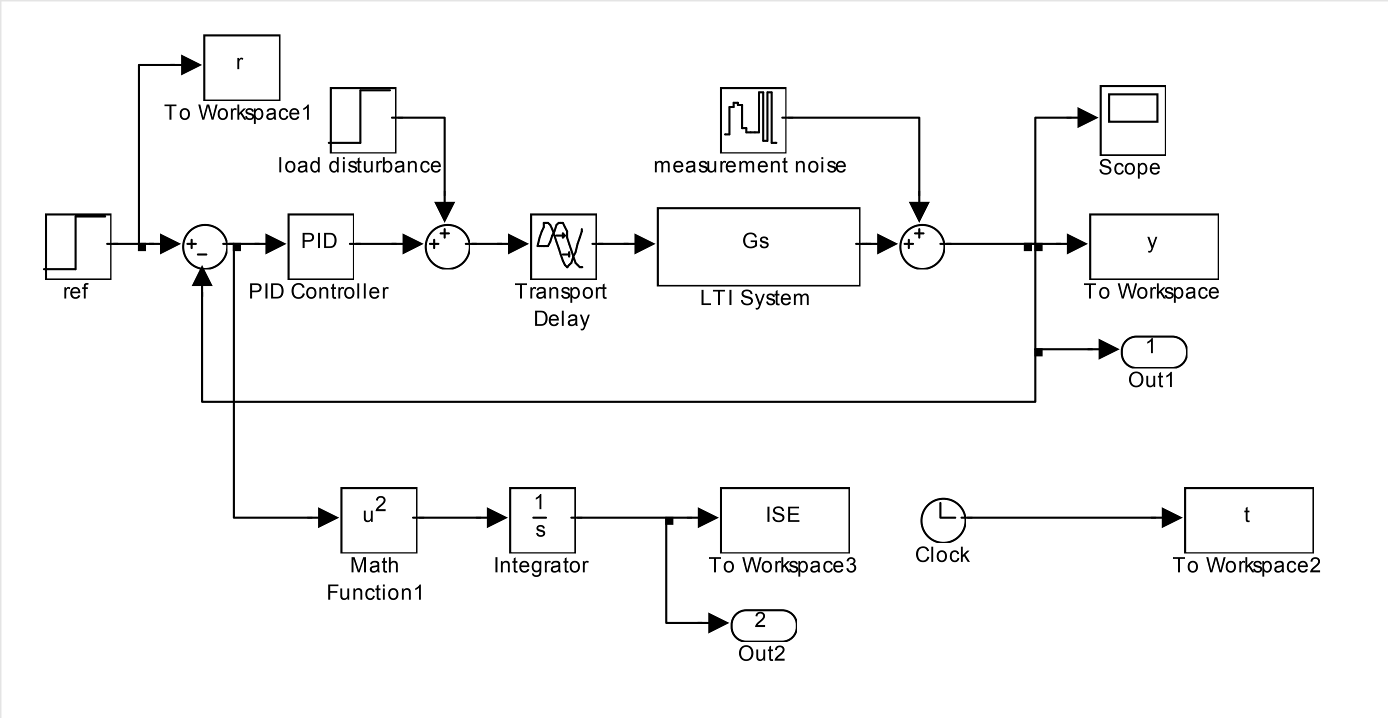
$$ITAE = \int_0^\infty t\left|e(t)\right| dt$$

$$ITE = \int_0^\infty te(t)dt$$

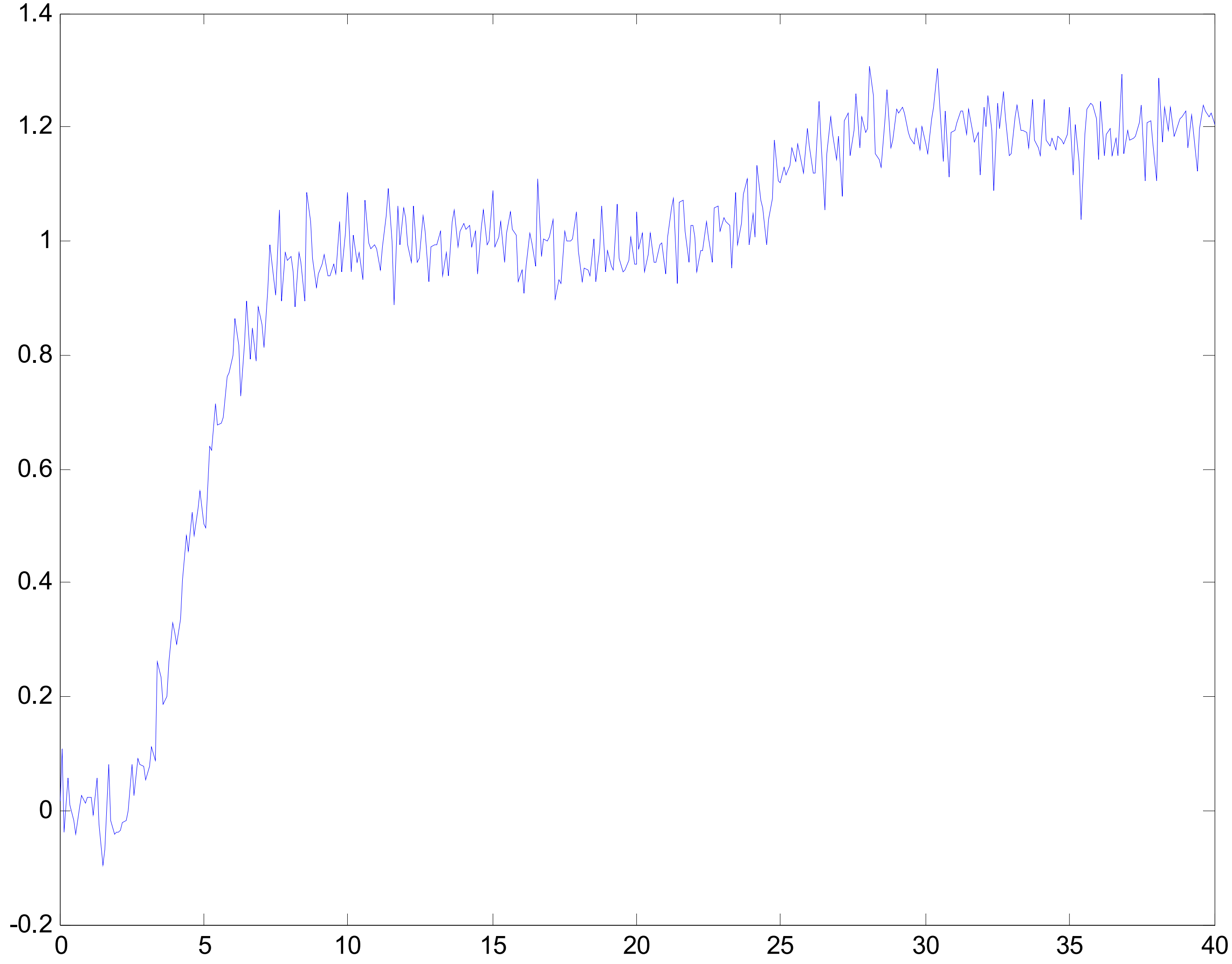
$$ITSE = \int_0^\infty te(t)^2\, dt$$

$$ISTE = \int_0^\infty t^2 e(t)^2\, dt$$

Example model



Open loop response



Cost function in Matlab

- %evaluate cost function from the simulink model
- function J = cost(inputs)
- %paramaters
- Tsim=40; %simulation time
- model= 'pidmodel'; %simulink model

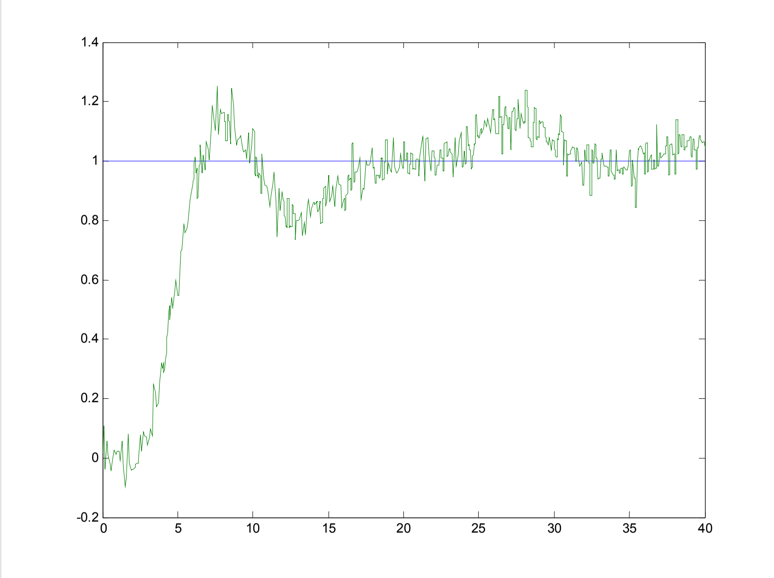
- %assign new parameters to workspace for simulation
- assignin('base','Kp',inputs(1));
- assignin('base','Ti',inputs(2));

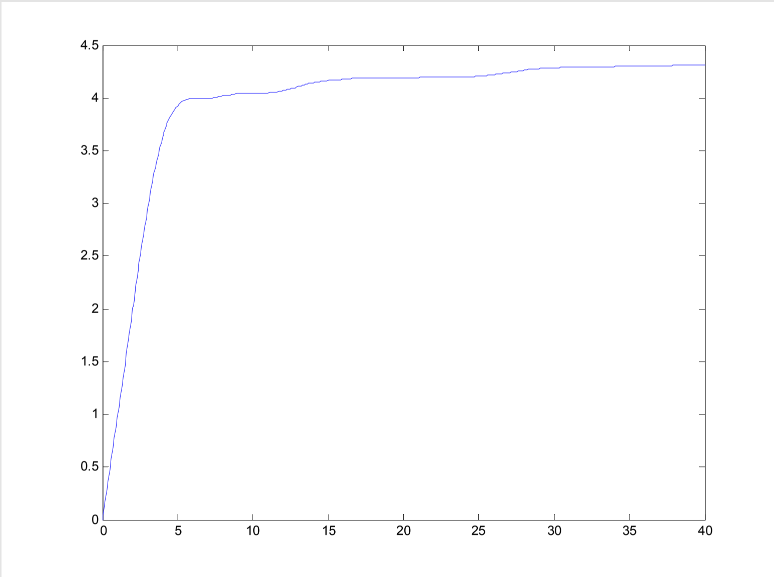
- %simulate
- [t,x,y]=sim(model,Tsim);

- %y(2) corresponds with cost
- J=max(y(:,2));

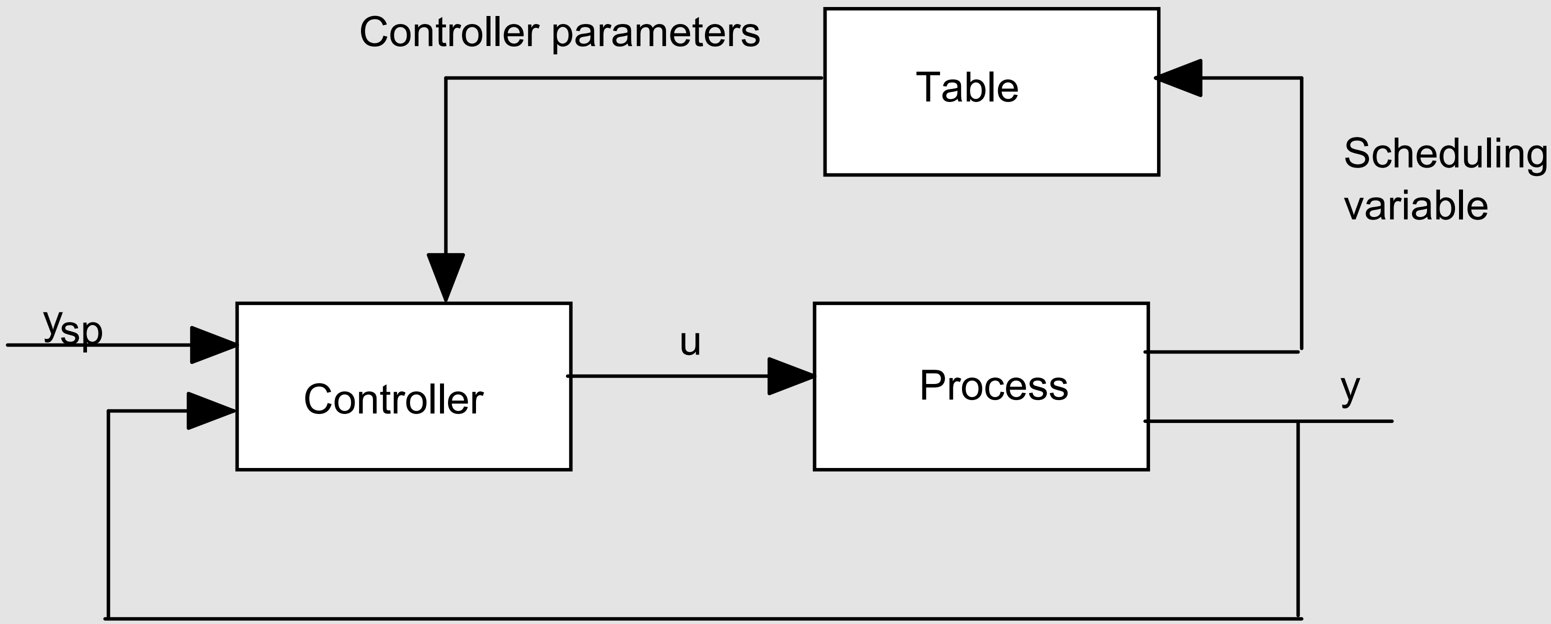
IN MATLAB:

- fminsearch('cost',[1 10])





Gain scheduling



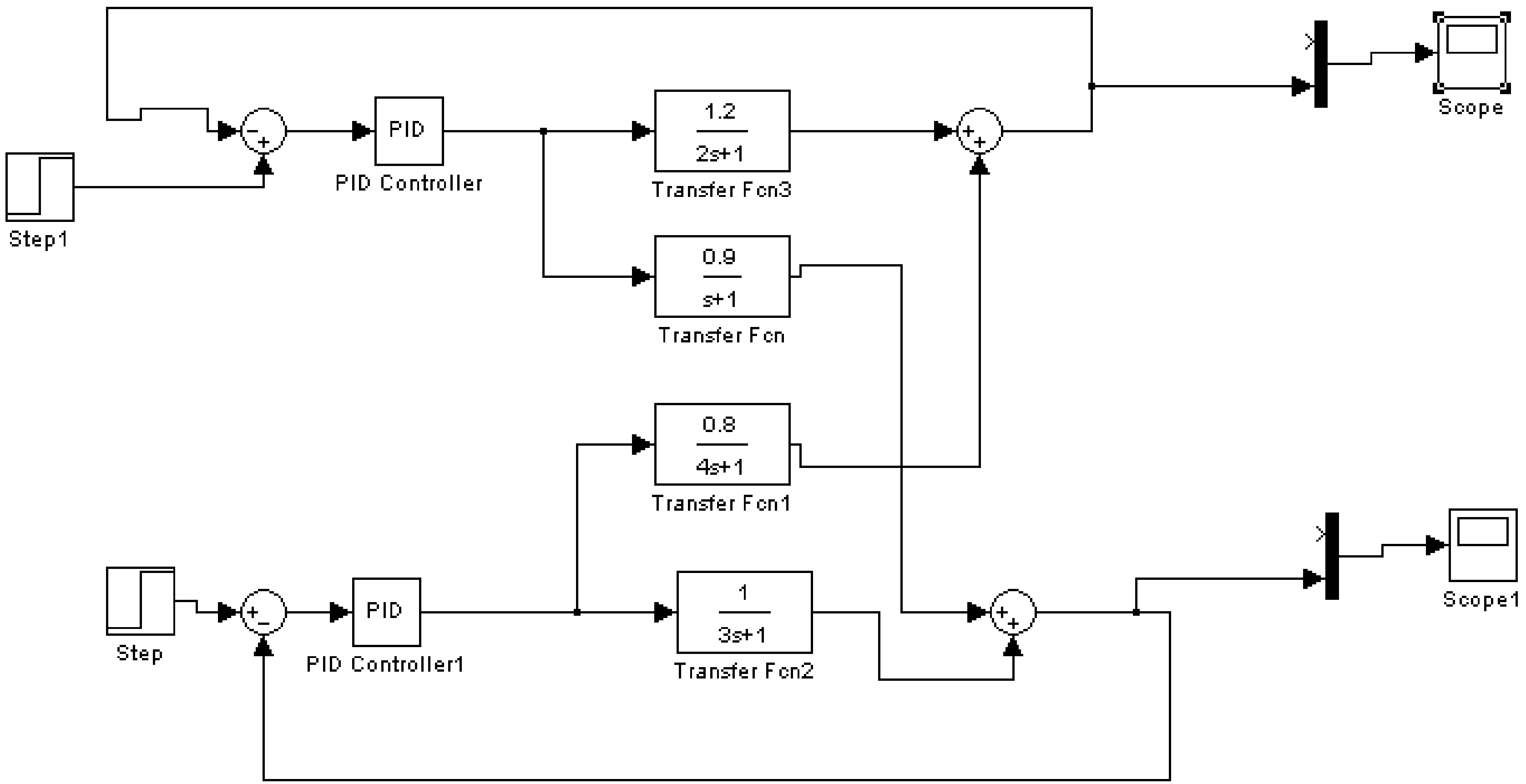
Rule-Based Methods

	Speed	Stability
K increases	increases	reduces
T_i increases	reduces	increases
T_d increases	increases	increases

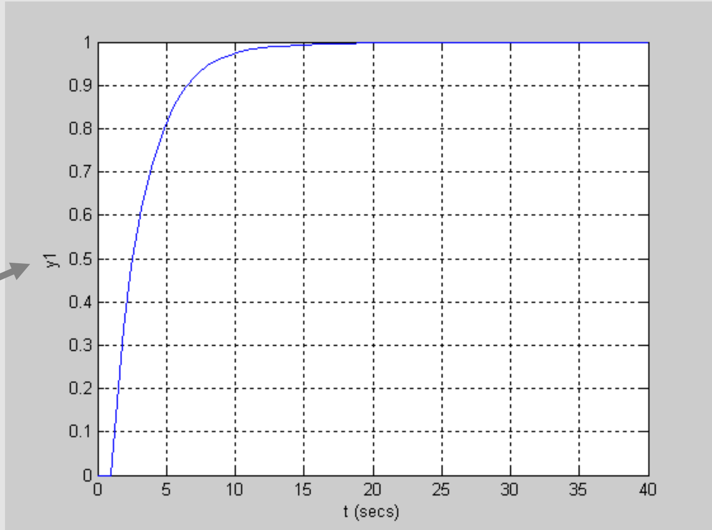
MIMO PID control- Decentralized

- Multiple Input Multiple Output (MIMO)

MIMO PID control- Decentralized



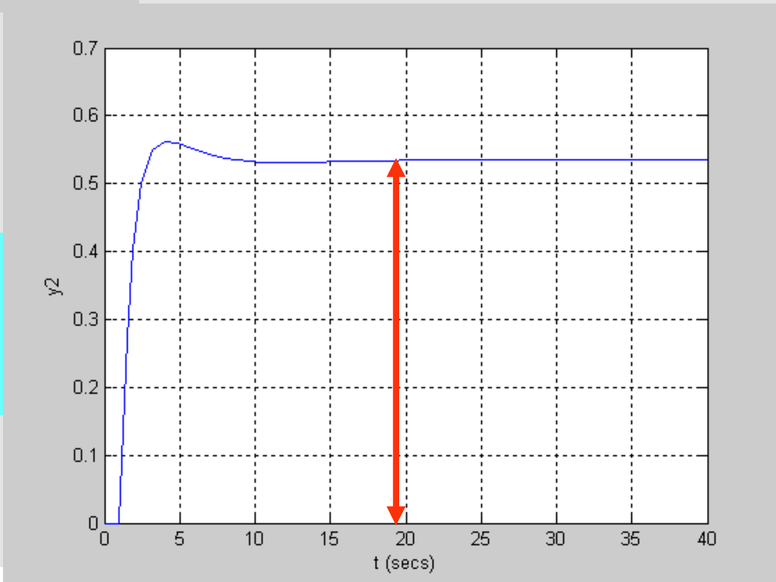
MIMO PID control- Decentralized



Lieslehto

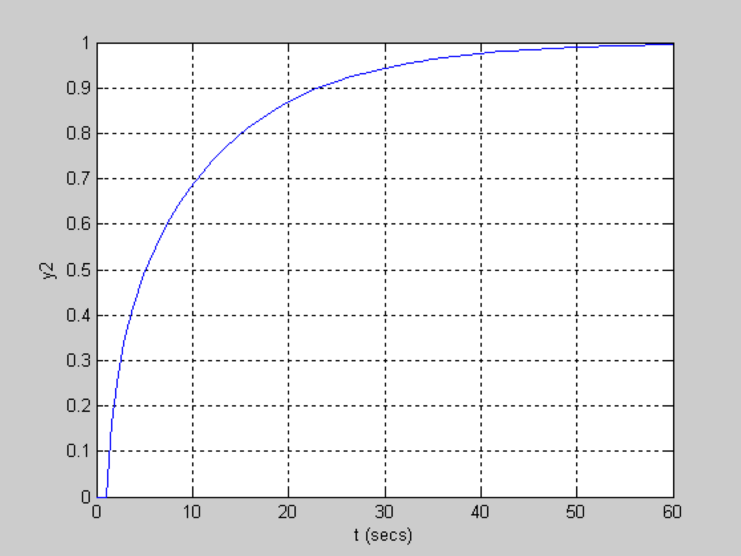
$K=0.83,\ T_i=2,\ I=0.5$

Unit step in ref1



Amount of interaction

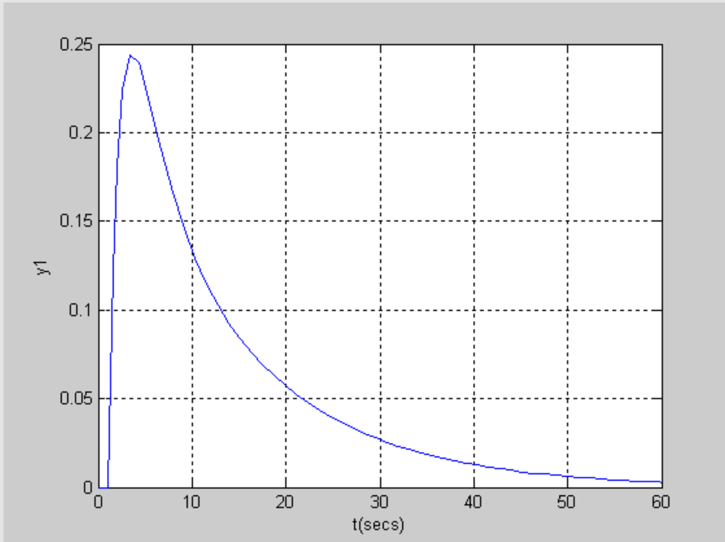
MIMO PID control- Decentralized



Lieslehto

$K=1.5,\ T_i=2,\ I=0.5$

Unit step in ref2



Amount of interaction

MIMO PID control - Centralized

State-space equation

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

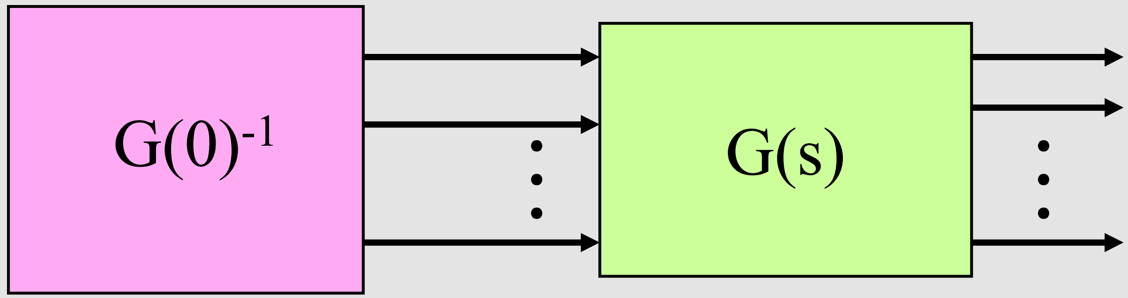
$$\mathbf{y} = \mathbf{C}\mathbf{x}$$

Transfer function

$$\mathbf{G}(\mathbf{s}) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$$

MIMO PID control - Centralized

Decoupling at steady state



MIMO PID control – Integral gain

$$G(0) = \begin{bmatrix} 1.2 & 0.9 \\ 0.8 & 1 \end{bmatrix}$$

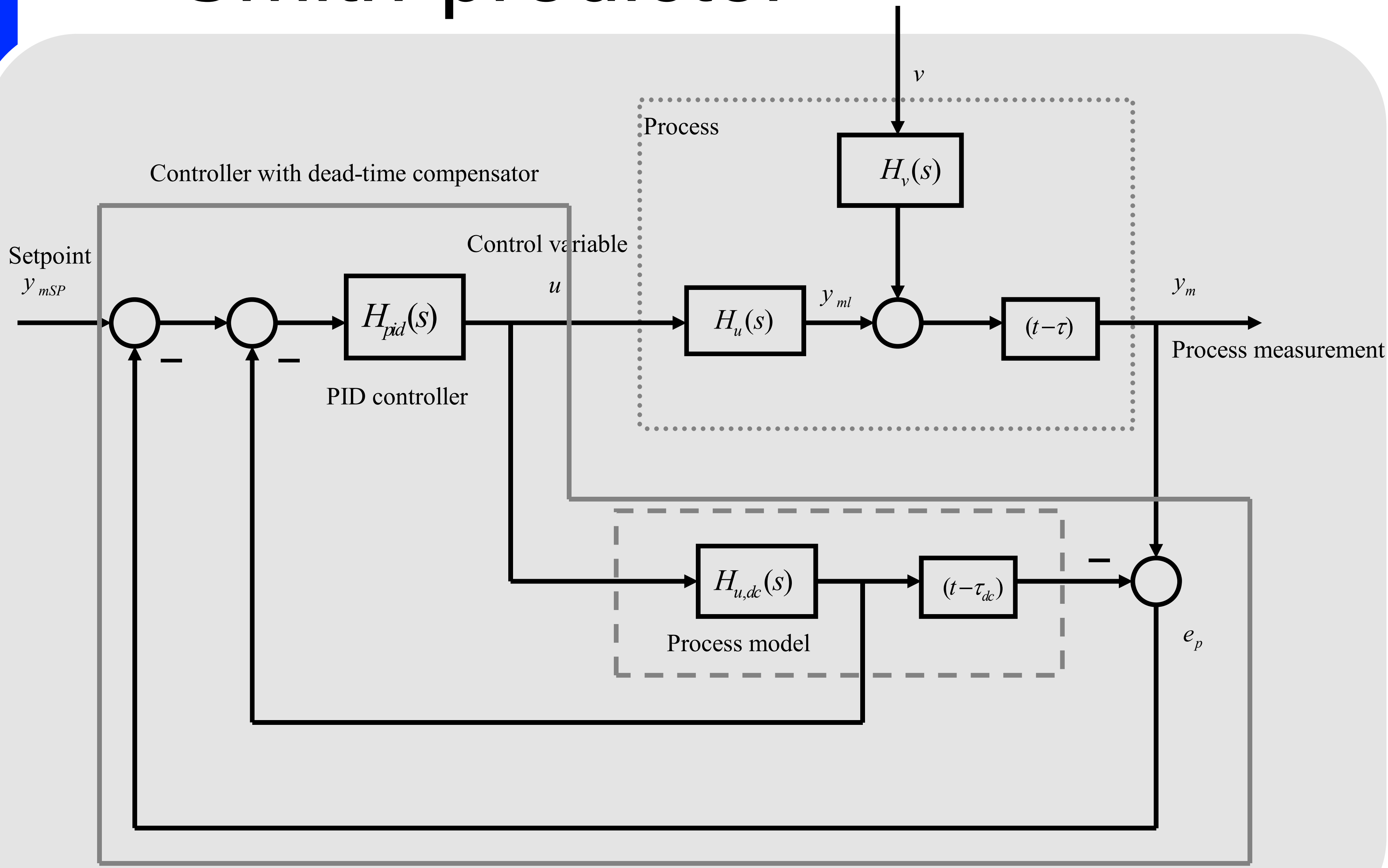
$$K_i = \varepsilon \left(G(0) \right)^{-1} = \varepsilon \begin{bmatrix} 1.2 & 0.9 \\ 0.8 & 1 \end{bmatrix}^{-1} = \varepsilon \begin{bmatrix} 2.1 & -1.9 \\ -1.7 & 2.5 \end{bmatrix}$$

MIMO PID control – Proportional gain

$$\lim_{s\to 0} sG(s) = \lim_{s\to 0} s\begin{bmatrix} \frac{1.2}{2s+1} & \frac{0.9}{s+1} \\ \frac{0.8}{4s+1} & \frac{1}{3s+1} \end{bmatrix} = \begin{bmatrix} 0.6 & 0.9 \\ 0.2 & 0.3 \end{bmatrix}$$

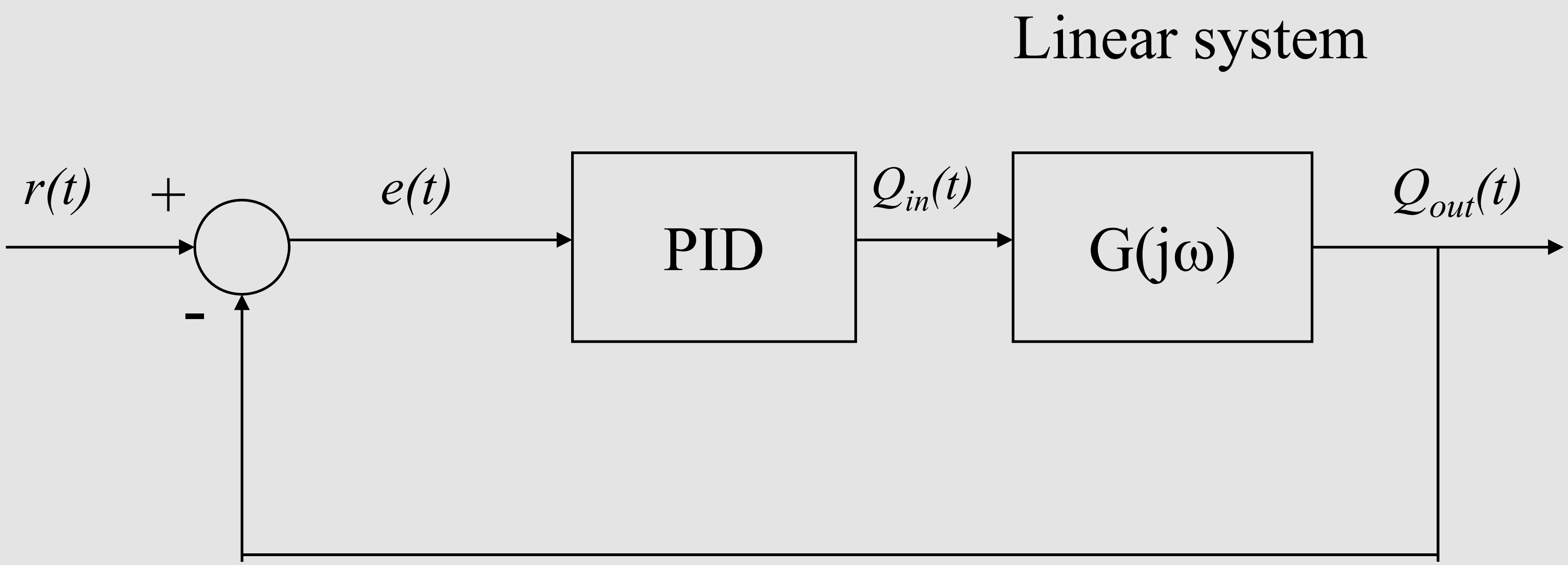
$$K_p = \delta\left(\lim_{s\to 0} sG(s)\right)^{-1} = \delta\begin{bmatrix} 0.6 & 0.9 \\ 0.2 & 0.33 \end{bmatrix}^{-1} = \delta\begin{bmatrix} 18.3 & -50 \\ -11 & 33 \end{bmatrix}$$

Smith-predictor

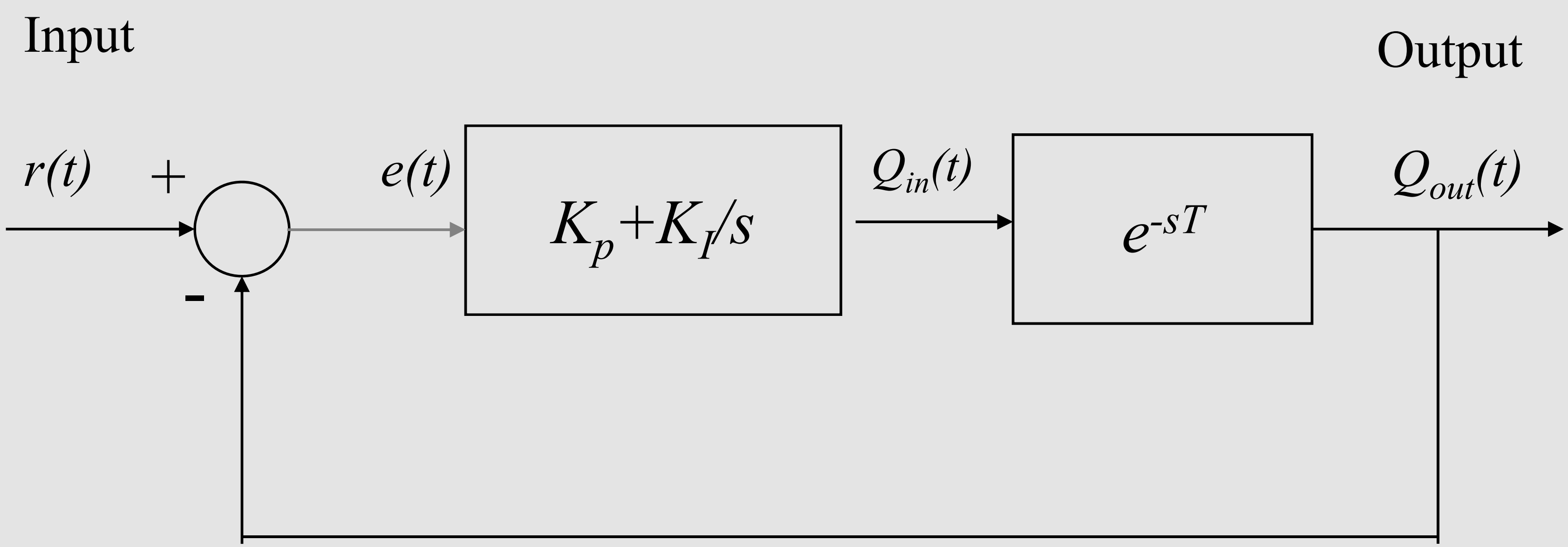


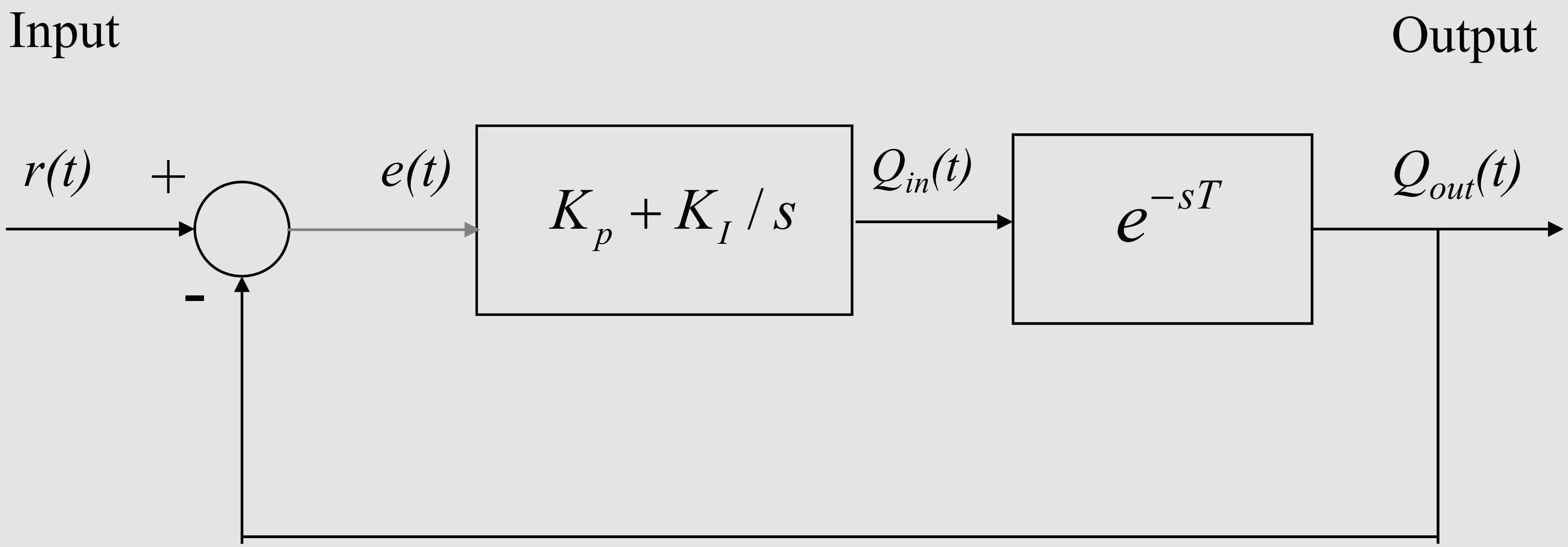
References

- K. Åström and T. Hägglund
  PID Controllers: Theory, Design, and Tuning (3rd ed) (2005)


Linear system
r(t)
+
-
e(t)
PID
Qin(t)
G(jω)
Qout(t)

$K_p + K_I / s$




Input
Output
r(t)
+
-
e(t)
Kp + KI / s
Qin(t)
e−sT
Qout(t)

Gain scheduling