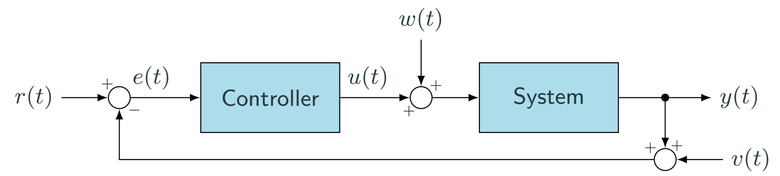
Control Systems I

Proportional, Integral, Derivative Controllers

Colin Jones

Laboratoire d'Automatique

Recall: The Control Loop

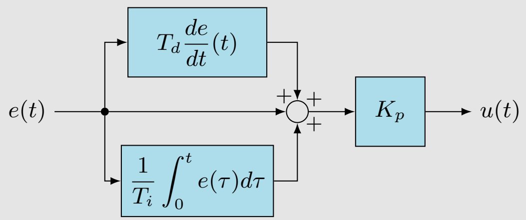


- Reference $r(t)$
- Error $e(t)$
- Input $u(t)$
- Input disturbance $w(t)$
- Measurement noise $v(t)$
- Ouput $y(t)$

Goal: Make $y(t) = r(t)$, no matter what $w(t)$, or $v(t)$ are

This Week: PID Control

PID - Proportional, Integral, Derivative Control



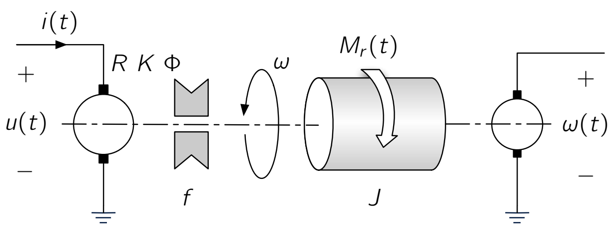
- Most common controller
- Used in billions of devices
- Very simple and effective

Goal: Drive error to zero and keep it there

$$\left.\begin{array}{ll} \text{P:} & u(t) = K_P e(t) \\ \text{I:} & u(t) = \int_0^t K_I e(\tau) d\tau \\ \text{D:} & u(t) = K_D \frac{de(t)}{dt} \end{array}\right\} \text{Zero if and only if error is zero and not changing}$$

Example

DC Motor Speed Control



Electrical dynamics:[1]

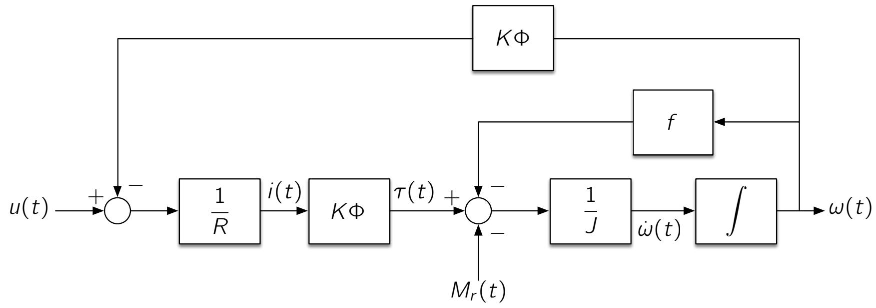
$$\underbrace{u(t)}_{\text{Voltage}} = \underbrace{v_{\text{emf}}}_{\text{Back-EMF}} + \underbrace{Ri(t)}_{\text{Resistance}} = K\Phi\omega(t) + Ri(t)$$

Mechanical dynamics:

$$\underbrace{\tau(i)}_{\text{Torque}} = K\Phi i(t) = \underbrace{J\dot{\omega}(t)}_{\text{Inertia}} + \underbrace{f\omega(t)}_{\text{Viscous friction}} + \underbrace{M_r(t)}_{\text{Parasitic torque}}$$

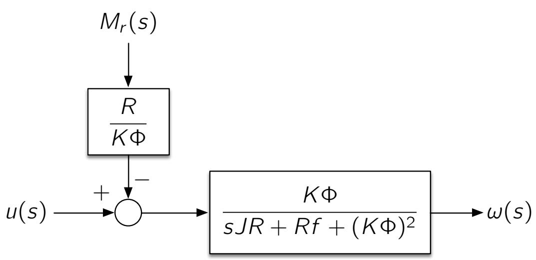
[1] Assuming that the motor inductance is negligible

Open-Loop Block Diagram



On the board: Simplify

Open-Loop Block Diagram



$$\omega(s) = \frac{K\Phi}{sJR + Rf + (K\Phi^2)}\left(u(s) - \frac{R}{K\Phi}M_r(s)\right)$$

Response to a step $u(s) = 1/s$

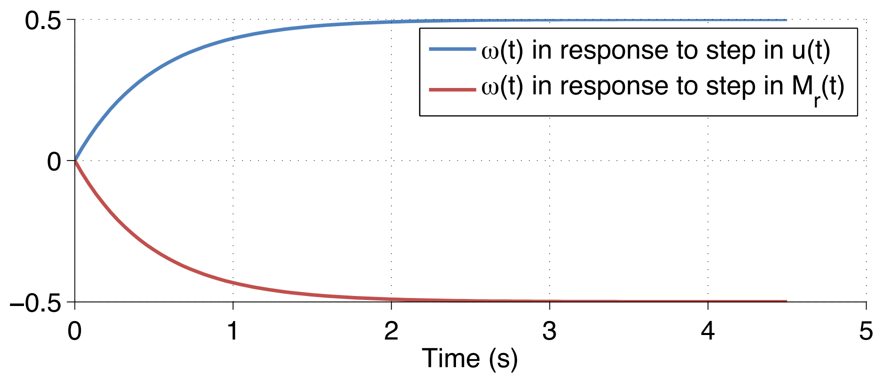
$$\omega(t) = \frac{K\Phi}{JR}\left(1 - e^{-\frac{Rf+(K\Phi)^2}{JR}t}\right)$$

Response to a step $M_r(s) = 1/s$

$$\omega(t) = -\frac{1}{J}\left(1 - e^{-\frac{Rf+(K\Phi)^2}{JR}t}\right)$$

[1] Comment on why we can drop gain on disturbance.

Open-loop System Response



Proportional Control

Proportional Control

Proportional Control

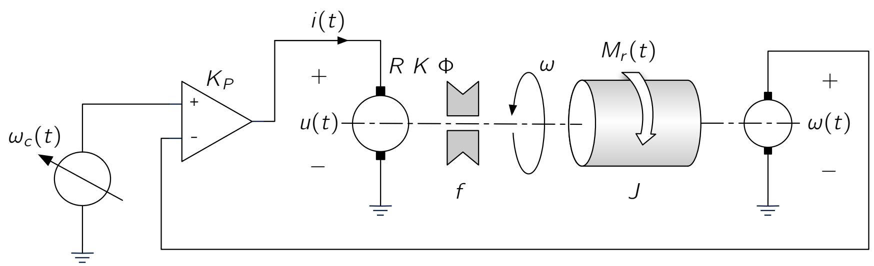
$$u(t) = K_P e(t) = K_P(y(t) - r(t))$$

Set the system input to be ***proportional*** to the ***error***

Intuition: Controller responds strongly to a large error and weakly to a small one

Only design choice: K_P
What impact does K_P have on the system behaviour?

Example: Motor Control

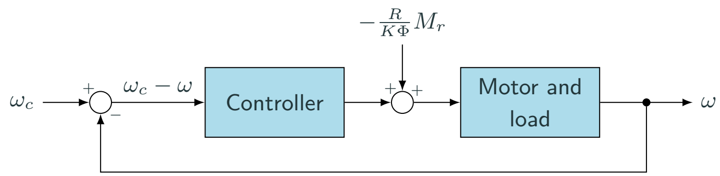


Recall:

$$\dot{\omega}(t) + \frac{1}{J}\left(f + \frac{(K\Phi)^2}{R}\right)\omega(t) = \frac{K\Phi}{JR}\left(u(t) - \frac{R}{K\Phi}M_r(t)\right)$$

Output: $\omega(t)$ speed of motor Input: $u(t)$ electrical current
J rotational inertia, R electrical resistance, f viscous friction, Φ inductance

Example: Block Diagram



System equation:

$$\dot{\omega}(t) + \frac{1}{J}\left(f + \frac{(K\Phi)^2}{R}\right)\omega(t) = \frac{K\Phi}{JR}\left(u(t) - \frac{R}{K\Phi}M_r(t)\right)$$

Controller equation:

$$u(t) = K_P(\omega_c(t) - \omega(t))$$

Intuition:

- Speed slower than desired ($\omega < \omega_c$): Increase current
- Speed faster than desired ($\omega > \omega_c$): Decrease current

Proportional Motor Speed Control

With the controller in place, the system equation is:[2]

$$\dot{\omega}(t) + \underbrace{\frac{1}{J}\left(f + \frac{(K\Phi)^2}{R}\right)}_{\alpha}\omega(t) = \underbrace{\frac{K\Phi}{JR}}_{\beta}K_p(\omega_c(t) - \omega(t))$$

$$\dot{\omega}(t) + \alpha\omega(t) = \beta K_p(\omega_c(t) - \omega(t))$$

Re-arranging gives:

$$\dot{\omega}(t) + (\alpha + \beta K_P)\omega(t) = \beta K_p \omega_c(t)$$

This is a standard first-order system.

[2] Note that we've assumed that the disturbance is zero here $M_r(t) = 0$.

Recall: Behaviour of First-Order Systems

$$\dot{x}(t) + \tau x(t) = \gamma v(t)$$

1. Take the Laplace transform:

$$sX(s) + \tau X(s) = \gamma V(s)$$
$$X(s)(s+\tau) = \gamma V(s)$$

2. Suppose the $v(t) = v_c$ for $t > 0$ for some constant v_c, then $V(s) = \frac{v_c}{s}$.

$$X(s) = \frac{\gamma}{s(s+\tau)}v_c$$

3. Take the inverse transform to compute the time-domain response

$$x(t) = \frac{\gamma}{\tau}v_c\mathcal{L}^{-1}\left\{\frac{1}{s} - \frac{1}{\tau + s}\right\} = \frac{\gamma}{\tau}v_c(1 - e^{-\tau t})$$

Response of Motor Under Proportional Control

$$\omega(t) = \frac{\beta K_P}{\alpha + \beta K_P}\bar{\omega}_c(1 - e^{-(\alpha+\beta K_P)t})$$

Take the constants to be: $J = f = K = \Phi = R = 1$.

$$\alpha = \frac{1}{J}\left(f + \frac{(K\Phi)^2}{R}\right) = 2 \qquad \beta = \frac{K\Phi}{JR} = 1$$

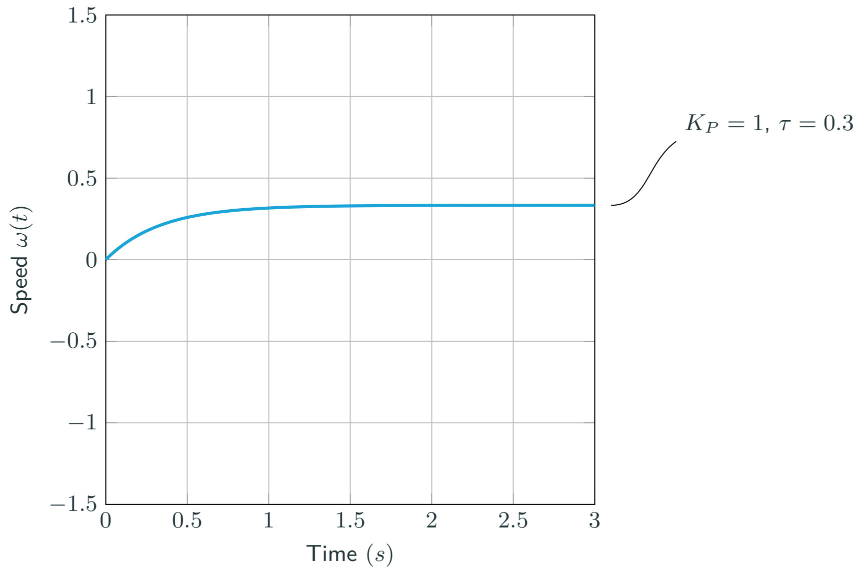
Suppose at time $t = 0$ a speed change is requested $\Rightarrow \bar{\omega}_c = 1$.

The time response is now:

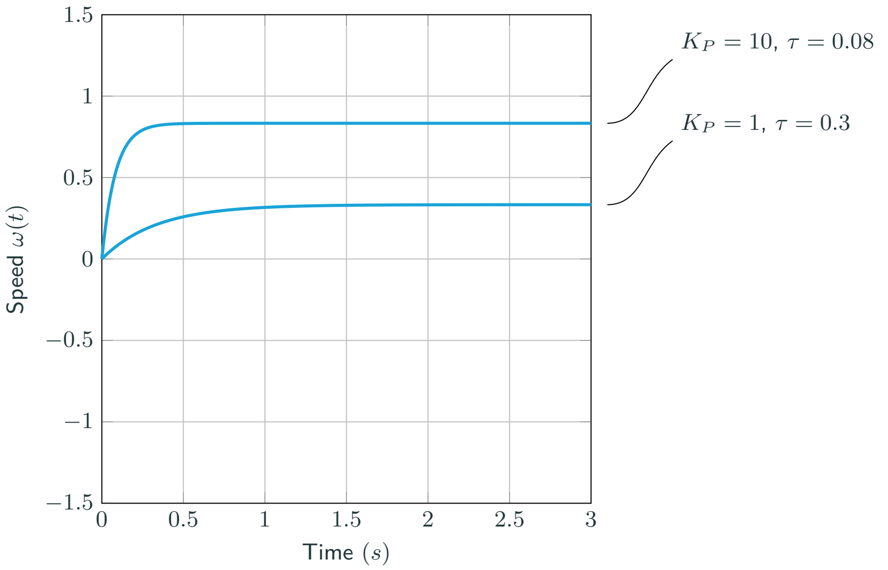
$$\omega(t) = \frac{K_P\bar{\omega}_c}{2 + K_P}\left(1 - e^{-(2+K_P)t}\right)$$

How should we choose K_P?

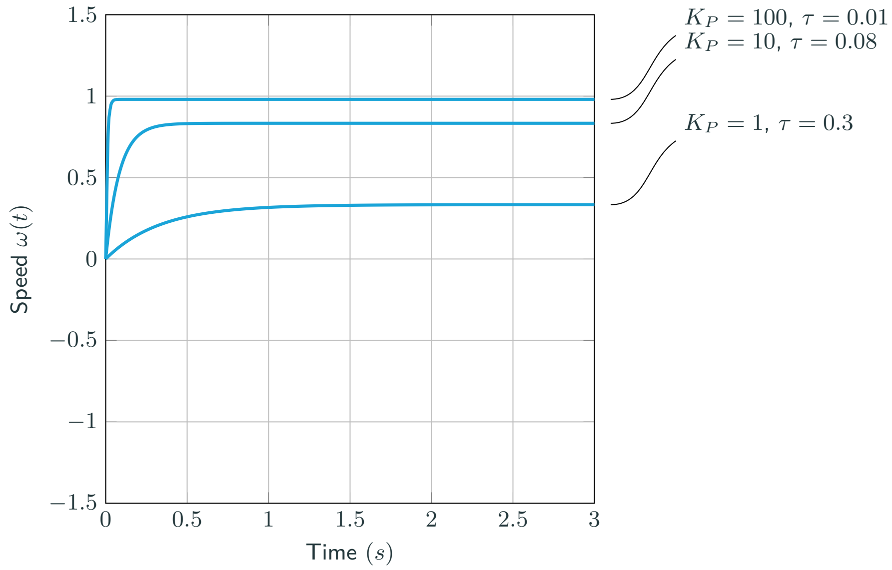
Response of Motor Under Proportional Control



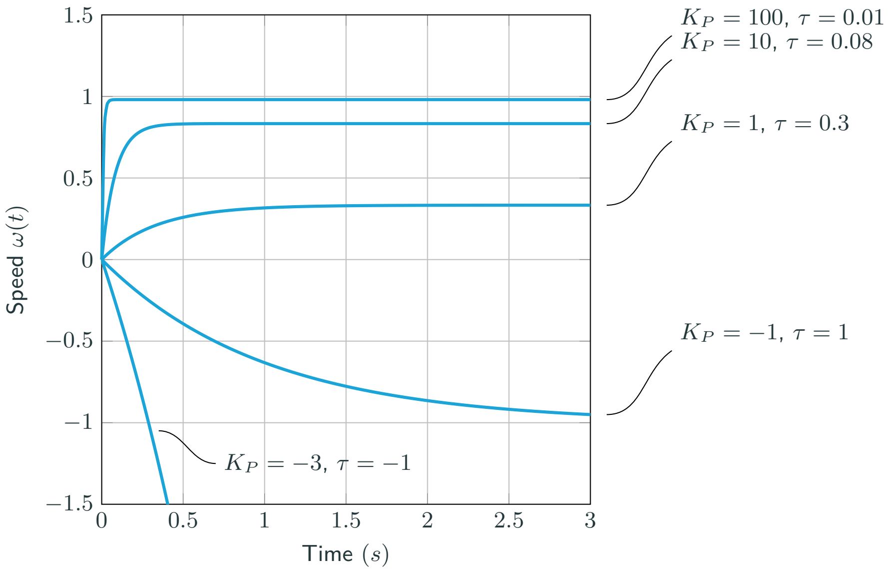
Response of Motor Under Proportional Control



Response of Motor Under Proportional Control



Response of Motor Under Proportional Control



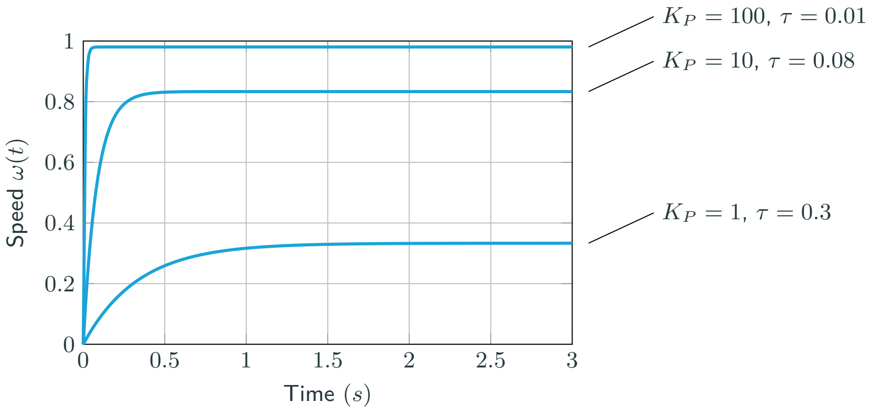
Impact of Proportional Gain

- Stability
  - An incorrect gain can cause the system to be unstable
- Transient response
  - A larger gain will normally cause the system to react more quickly
  - Larger gain $\rightarrow$ larger input. However, you do not have unlimited input authority!
- Steady-state offset
  - Many systems will have a steady-state offset with only proportional control

$$\lim_{t\to\infty} \omega(t) = \lim_{t\to\infty} \frac{K_P \bar{\omega}_c}{2 + K_P}(1 - e^{-(2+K_P)t}) = \frac{K_P}{2 + K_P}\bar{\omega}_c \neq \bar{\omega}_c$$

Another component needed to ensure steady-state error is zero $\rightarrow$ Integrator

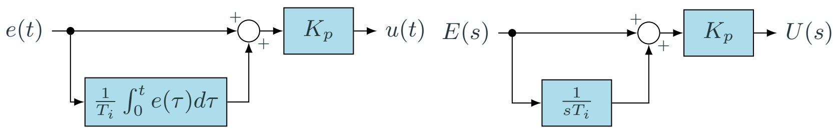
Why Not Choose the Maximum K_P?



- Faster response requires a faster actuator
- Need more input authority ('stronger' actuator)
- You may just be amplifying noise (more later)

Proportional Integral Control

Proportional Integral (PI) Control



Proportional Integral Control

$$u(t) = K_P\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau\right) = K_P e(t) + K_i \int_0^t e(\tau)d\tau)$$

where $K_i := \frac{K_P}{T_i}$

$$U(s) = K_p\left(1 + \frac{1}{T_i s}\right)E(s) = \left(K_P + \frac{K_i}{s}\right)E(s)$$

- Input is proportional to the integral of the error
- Intuition: Control input continues to grow until the error goes to zero

Final Value Theorem

How to compute the steady-state value of a signal?

Final Value Theorem

If and only if the linear time invariant system producing $x(t)$ is stable, then

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} sX(s)$$

The system must be stable!

- If it's not, then the FVT will give you the wrong answer (it won't predict an unbounded, or oscillatory response)

Final Value Theorem - Proof Sketch

First: Recall the Laplace transform of the derivative

$$\mathcal{L}\left(\frac{\mathrm{d}x(t)}{\mathrm{d}t}\right) = \int_0^\infty \frac{\mathrm{d}x(t)}{\mathrm{d}t} e^{-st} dt$$

Final Value Theorem - Proof Sketch

First: Recall the Laplace transform of the derivative

$$\mathcal{L}\left(\frac{\mathrm{d}x(t)}{\mathrm{d}t}\right) = \int_0^\infty \frac{\mathrm{d}x(t)}{\mathrm{d}t} e^{-st} dt$$

$$= x(t)e^{-st}\Big|_0^\infty - (-s)\int_0^\infty x(t)e^{-st}dt \qquad \text{Integration by parts}$$

Final Value Theorem - Proof Sketch

First: Recall the Laplace transform of the derivative

$$\mathcal{L}\left(\frac{\mathrm{d}x(t)}{\mathrm{d}t}\right) = \int_0^\infty \frac{\mathrm{d}x(t)}{\mathrm{d}t} e^{-st} dt$$

$$= x(t)e^{-st}\Big|_0^\infty - (-s)\int_0^\infty x(t)e^{-st}dt \qquad \text{Integration by parts}$$

$$= \underbrace{\lim_{t\to\infty} x(t)e^{-st}}_{=0\ x(t)\ \text{is stable}} - x(0) + s\mathcal{L}\left(x(t)\right)$$

Final Value Theorem - Proof Sketch

First: Recall the Laplace transform of the derivative

$$\mathcal{L}\left(\frac{\mathrm{d}x(t)}{\mathrm{d}t}\right) = \int_0^\infty \frac{\mathrm{d}x(t)}{\mathrm{d}t} e^{-st} dt$$

$$= x(t)e^{-st}\Big|_0^\infty - (-s)\int_0^\infty x(t)e^{-st}dt \qquad \text{Integration by parts}$$

$$= \underbrace{\lim_{t\to\infty} x(t)e^{-st}}_{=0\ x(t)\ \text{is stable}} - x(0) + s\mathcal{L}\left(x(t)\right)$$

$$= -x(0) + sX(s)$$

Final Value Theorem - Proof Sketch

Second: What happens when we take $s \to 0$?

$$\lim_{s\to 0}\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}e^{-st}dt = \lim_{s\to 0} -x(0) + sX(s)$$

Final Value Theorem - Proof Sketch

Second: What happens when we take $s \to 0$?

$$\lim_{s\to 0}\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}e^{-st}dt = \lim_{s\to 0} -x(0) + sX(s)$$

$$\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}dt = -x(0) + \lim_{s\to 0} sX(s)$$

Final Value Theorem - Proof Sketch

Second: What happens when we take $s \to 0$?

$$\lim_{s\to 0}\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}e^{-st}dt = \lim_{s\to 0} -x(0) + sX(s)$$

$$\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}dt = -x(0) + \lim_{s\to 0} sX(s)$$

$$\lim_{t\to\infty} x(t) - x(0) = -x(0) + \lim_{s\to 0} sX(s)$$

Final Value Theorem - Proof Sketch

Second: What happens when we take $s \to 0$?

$$\lim_{s\to 0}\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}e^{-st}dt = \lim_{s\to 0} -x(0) + sX(s)$$

$$\int_0^\infty \frac{\mathsf{d}x(t)}{\mathsf{d}t}dt = -x(0) + \lim_{s\to 0} sX(s)$$

$$\lim_{t\to\infty} x(t) - x(0) = -x(0) + \lim_{s\to 0} sX(s)$$

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} sX(s)$$

There is a similar relation between the limit as t goes to zero, and s goes to infinity.

Example: Motor Control

$$\dot{\omega}(t) + \alpha\omega(t) = \beta u(t)$$

Control input: $u(t) = K_P(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau) \;\rightarrow\; U(s) = K_P(1 + \frac{1}{T_i s})E(s)$

$$(s+\alpha)\Omega(s) = \beta K_p \left(1 + \frac{1}{T_i s}\right)(\Omega_c(s) - \Omega(s))$$

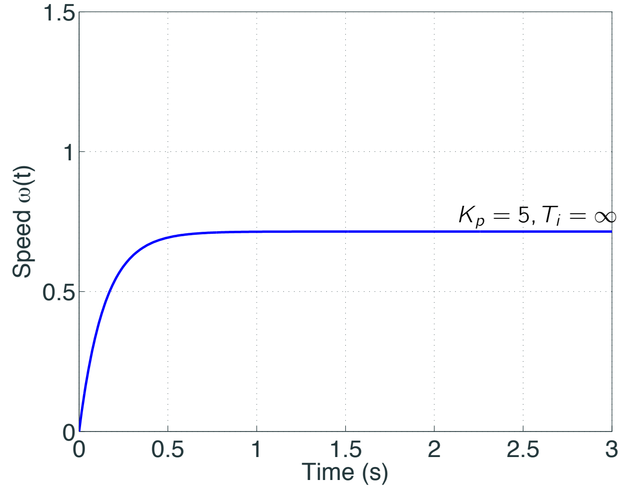
$$\Omega(s) = \frac{\beta K_p (T_i s + 1)}{T_i s^2 + T_i(\alpha + \beta K_p)s + \beta K_p}\Omega_c(s)$$

Steady-state error in response to a step in the command: $\Omega_c(s) = \frac{\bar{w}_c}{s}$:

$$\begin{aligned}\lim_{t\to\infty} w(s) &= \lim_{s\to 0} s\Omega(s) \\ &= \lim_{s\to 0} s \frac{\beta K_p (T_i s + 1)}{T_i s^2 + T_i(\alpha + \beta K_p)s + \beta K_p} \frac{\bar{w}_c}{s} \\ &= \bar{w}_c\end{aligned}$$

If the system is stable, then there is ***no steady-state offset***

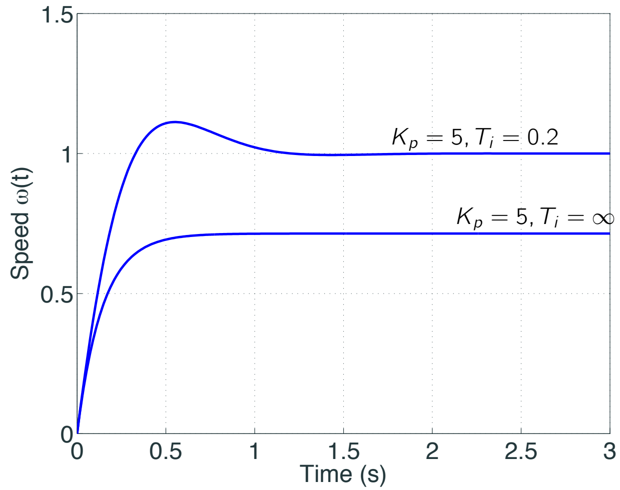
Motor Speed Control



System response to a speed change command $\bar{\omega}_c = 1$

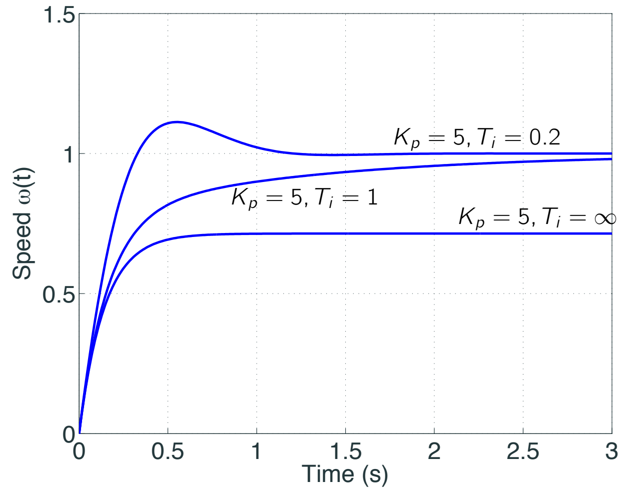
- No integrator $\rightarrow$ system settles at the wrong speed

Motor Speed Control



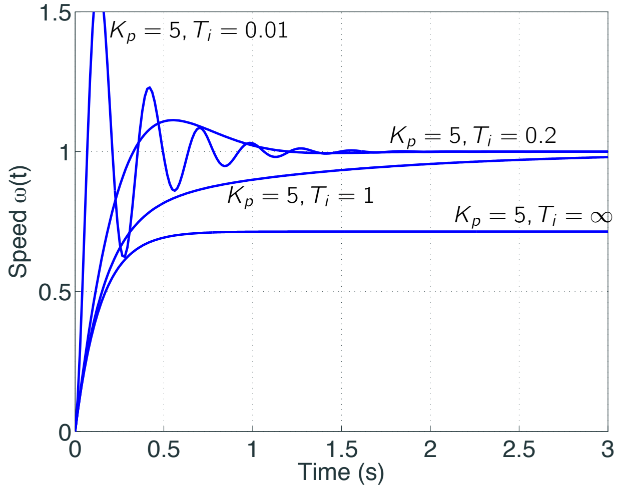
System response to a speed change command $\bar{\omega}_c = 1$

Motor Speed Control



System response to a speed change command $\bar{\omega}_c = 1$

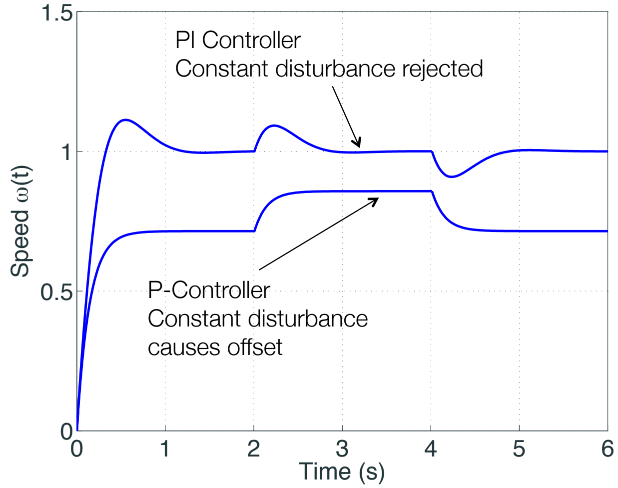
Motor Speed Control



System response to a speed change command $\bar{\omega}_c = 1$

- Tuning the system is now more complex (more later)

Rejection of Constant Disturbances

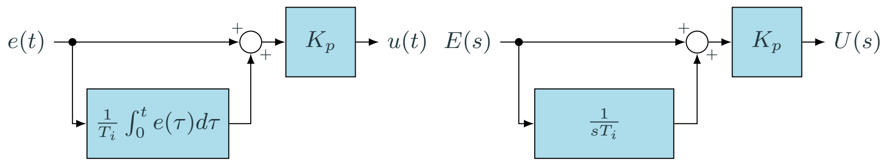


- Disturbance impacts the system from $t = 2$ to $t = 4$
- The integrator rejects the disturbance and keep the system at the setpoint

Interactive Simulations

External example 1.29

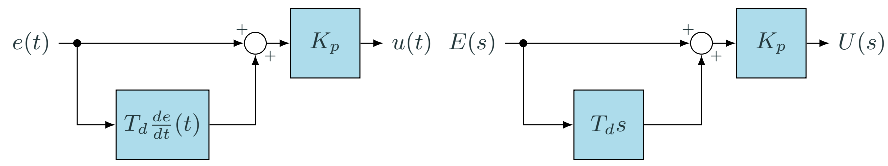
PI Control - Summary



- Steady-state offset
  - Integrator ensures zero offset (more details later)
- Stability
  - Adding an integrator can easily destabilize the system
- Transient response
  - Tuning is now more complex (more details later)

Proportional Derivative Control

Proportional Derivative (PD) Control



Proportional Derivative Control

$$u(t) = K_P\left(e(t) + T_d\frac{\mathrm{d}e}{\mathrm{d}t}(t)\right) = K_P e(t) + K_d\frac{\mathrm{d}e}{\mathrm{d}t}(t)$$

where $K_d := K_P T_d$

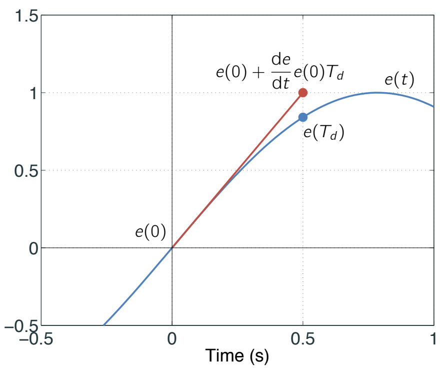
$$U(s) = K_P(1 + T_d s)E(s) = (K_P + K_d s)E(s)$$

- Input is proportional to the derivative of the error
- Intuition: React to fast disturbances more quickly than slow ones

PD Control : An Interpretation

Consider the value of the error T_d seconds into the future:

$$e(t + T_d) \approx e(t) + \frac{\mathrm{d}e}{\mathrm{d}t}(t)T_d$$



One interpretation: Feedback on an estimate of the future error

Motor Control Example

We now want to control the position θ of the motor:

$$\ddot{\theta}(t) + \alpha\dot{\theta}(t) = \beta u(t) \qquad u(t) = K_P\left(e(t) + T_d\frac{\mathrm{d}e}{\mathrm{d}t}(t)\right)$$

$$= K_P\left(\theta_c(t) - \theta(t) - T_d\frac{\mathrm{d}\theta}{\mathrm{d}t}(t)\right)^{3}$$

Take the Laplace transform:

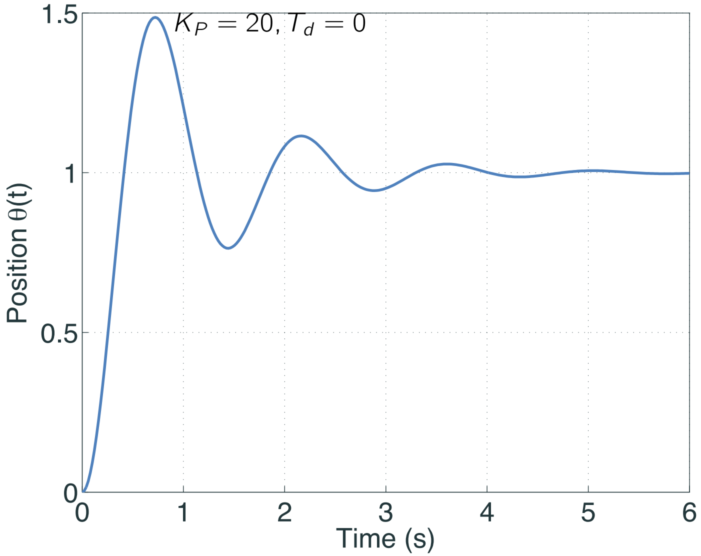
$$(s^2 + \alpha s)\Theta(s) = \beta K_P\Theta_c(s) - \beta K_p(1 + T_d s)\Theta(s)$$

$$\Theta(s) = \frac{\beta K_P}{s^2 + (\alpha + \beta K_P T_d)s + \beta K_P}\Theta_c(s)$$

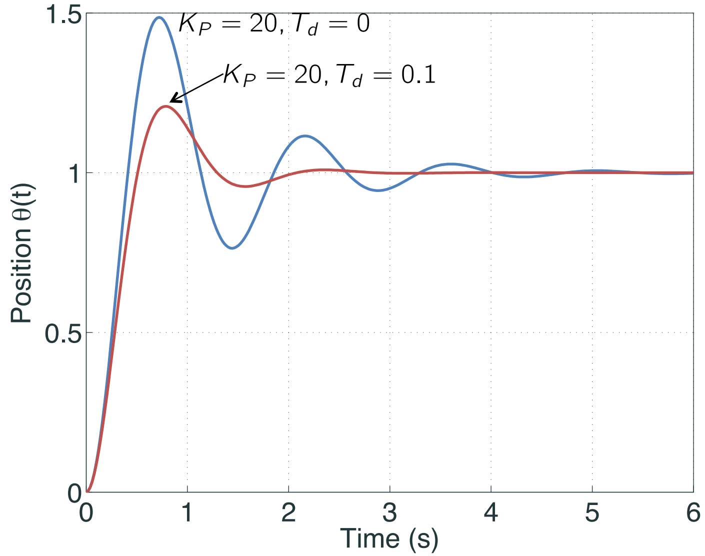
The gain T_d impacts the ***damping*** of the closed-loop system. (More later)

[3] Note that the derivative of $\theta_c(t)$ is assumed to be zero here

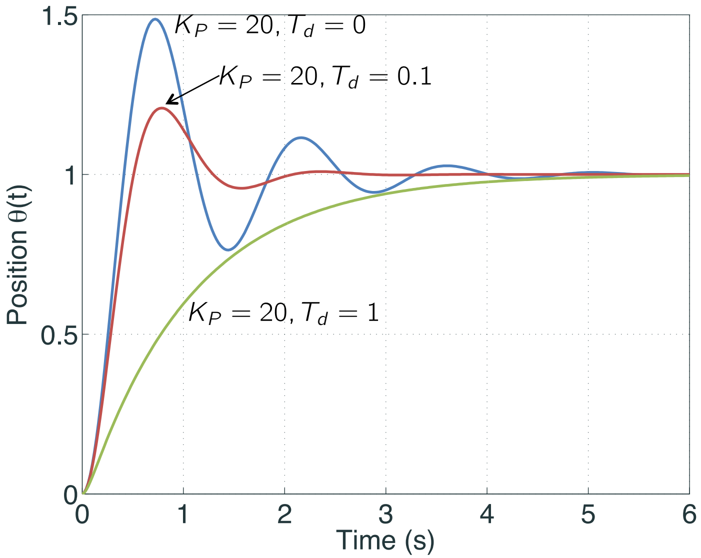
Response of Closed-Loop System to PD Control



Response of Closed-Loop System to PD Control



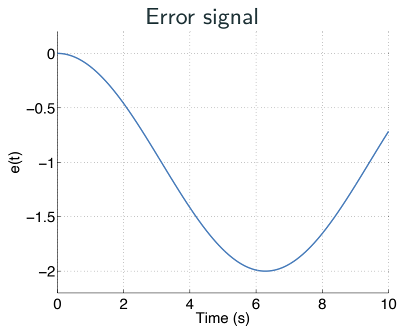
Response of Closed-Loop System to PD Control

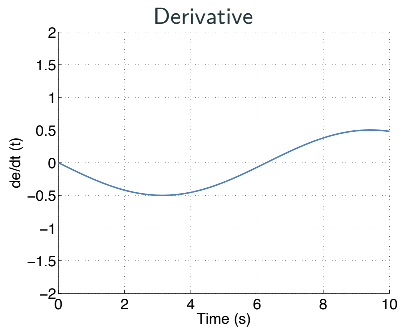


Implementing Derivative Action

$$T_d \frac{\mathrm{d}e}{\mathrm{d}t}(t) \qquad\qquad T_d s E(s) \approx \frac{T_d s}{\frac{T_d}{N} s + 1} E(s)$$

- Not a proper expression, and cannot be implemented in a circuit
- Digital approximation: $u(t) \approx \frac{e(t) - e(t-\Delta)}{\Delta}$

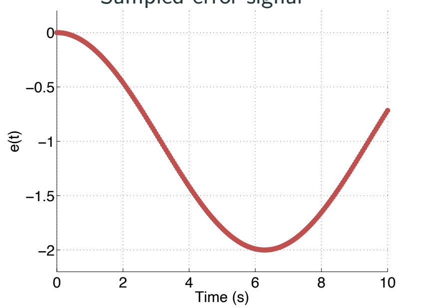


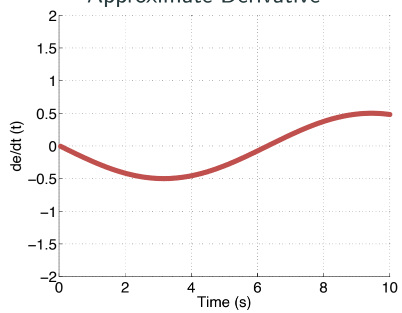


Implementing Derivative Action

$$T_d \frac{\mathrm{d}e}{\mathrm{d}t}(t) \qquad\qquad T_d s E(s) \approx \frac{T_d s}{\frac{T_d}{N} s + 1} E(s)$$

- Not a proper expression, and cannot be implemented in a circuit
- Digital approximation: $u(t) \approx \frac{e(t)-e(t-\Delta)}{\Delta}$

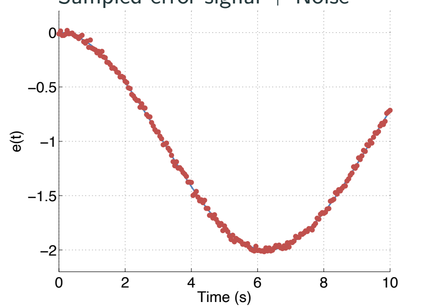


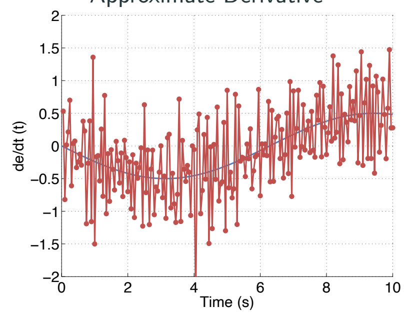


Implementing Derivative Action

$$T_d \frac{\mathrm{d}e}{\mathrm{d}t}(t) \qquad\qquad T_d s E(s) \approx \frac{T_d s}{\frac{T_d}{N} s + 1} E(s)$$

- Not a proper expression, and cannot be implemented in a circuit
- Digital approximation: $u(t) \approx \frac{e(t)-e(t-\Delta)}{\Delta}$

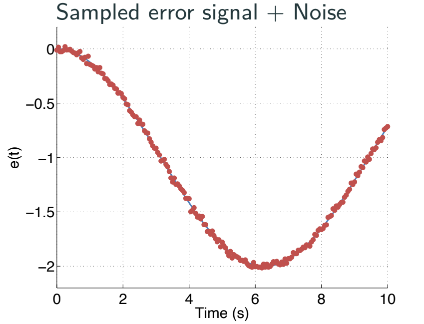


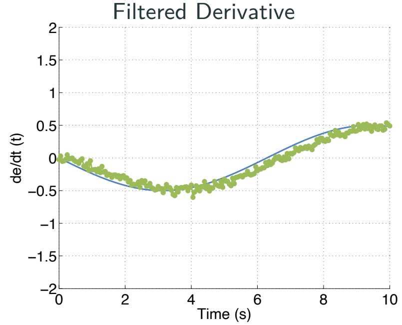


Implementing Derivative Action

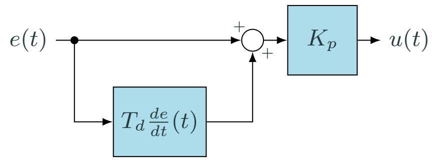
$$T_d \frac{\mathrm{d}e}{\mathrm{d}t}(t) \qquad\qquad T_d s E(s) \approx \frac{T_d s}{\frac{T_d}{N} s + 1} E(s)$$

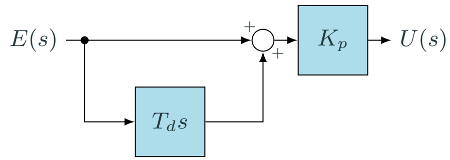
- Not a proper expression, and cannot be implemented in a circuit
- Digital approximation: $u(t) \approx \frac{e(t)-e(t-\Delta)}{\Delta}$





PD Control - Summary



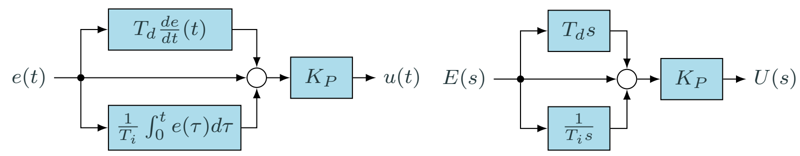


- Stability
  - Can add extra damping to the system.
  - Intuition: Acts to reduce velocity
- Transient response
  - Tuning is now more complex (more details later)
- Robustness
  - Operates on ***high-frequencies*** more than lower-frequencies
  - Will amplify high-frequency noise acting on the system

  ⇒ Derivative controllers are always combined with low-pass filters

Proportional Integral Derivative Control

Proportional Integral Derivative (PID) Control



Proportional Integral Derivative (PID) Control

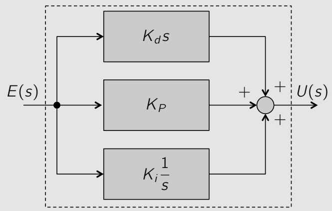
$$u(t) = K_P\left(e(t) + T_d\frac{\mathrm{d}e}{\mathrm{d}t}(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau\right)$$
$$= K_P e(t) + K_d\frac{\mathrm{d}e}{\mathrm{d}t}(t) + K_i\int_0^t e(\tau)d\tau$$

Or in the Laplace domain:

$$U(s) = K_P\left(1 + T_d s + \frac{1}{T_i s}\right)E(s) = \left(K_P + K_d s + K_i\frac{1}{s}\right)E(s)$$

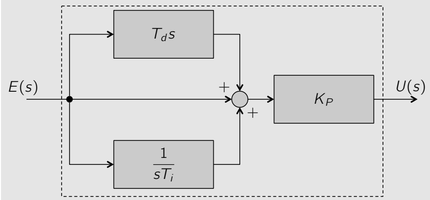
Many Equivalent Formulations

Parallel Formulation



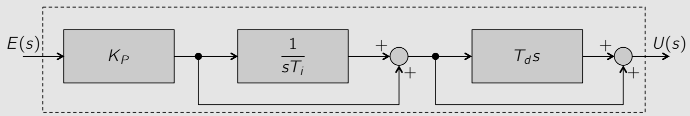
$$U(s) = K_P + K_d s + K_i\frac{1}{s}$$

Mixed Formulation



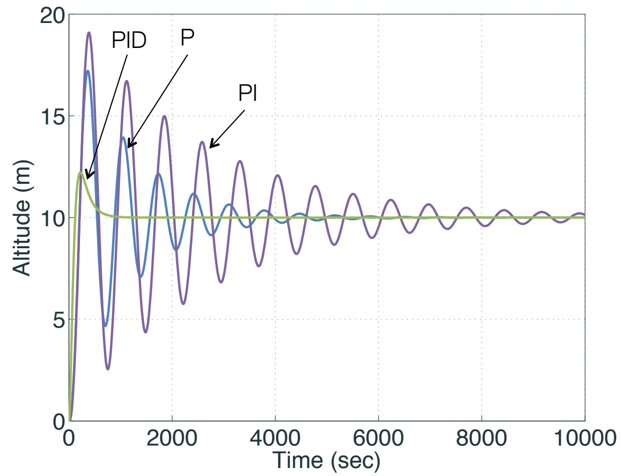
$$U(s) = K_P\left(1 + T_d s + \frac{1}{T_i s}\right)$$

Series Formulations



$$U(s) = K_P\left(1 + \frac{1}{T_i s}\right)(1 + T_d s)$$

Balloon Altitude Control - Closed-Loop Response



PID Control

Proportional
- Sets the 'aggressiveness' of your system.
- Higher generally means that the system will respond more strongly to disturbances

Integral
- Added to ensure zero steady-state offset
- Not necessary if your system already has 'integral action'
- Danger: Can easily de-stabilize the system

Derivative
- Increase the damping of the system - improve stability
- Can amplify high-frequency noise
- Less often used

Ziegler-Nichols Tuning

Tuning: How to choose the parameters K_P, T_i and T_d??

$\Rightarrow$ $1,637$ books on "PID Control" on Amazon

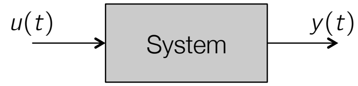
Common approaches:

Factory defaults	$\rightarrow$	Very common practice!
Fiddle until it works	$\rightarrow$	Can be effective if not very complex (and stable)
Model-based approaches	$\rightarrow$	Good initial settings for delicate, unstable systems
Automatic tuning	$\rightarrow$	Effective in specific settings
Experimental tuning	$\rightarrow$	Structured, simple and effective

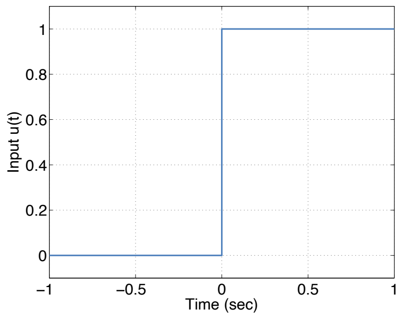
The most common form of experimental tuning: Ziegler-Nichols

Note a lot of intuition why this works... primarily based on experience

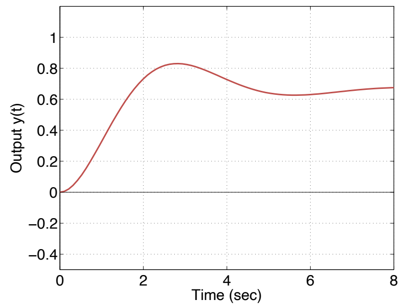
Ziegler-Nichols First Method: Stable Systems



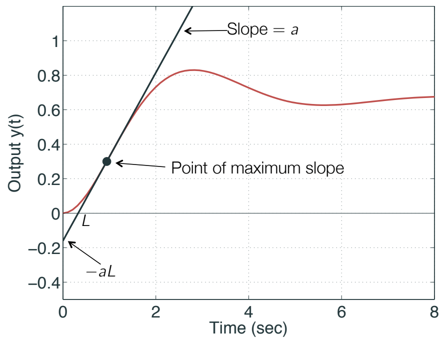
(1) Step input



(2) Analyze response



Ziegler-Nichols First Method: Stable Systems



Type	K_P	T_i	T_d
P	$\frac{1}{aL}$		
PI	$\frac{0.9}{aL}$	$3.3L$	
PID	$\frac{1.2}{aL}$	$2L$	$0.5L$

$$u(t) = K_P e(t)$$

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

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

Example - Balloon Velocity Control

Spirit of Freedom

Equations of motion:

$$\delta\dot{T} + \frac{1}{\tau_1}\delta T = \delta q$$

$$\tau_2 \dot{v} + v = a\delta T$$

$\delta T =$ deviation of the hot-air temperature from the equilibrium temperature where buoyant force equals weight

$v =$ vertical velocity of the balloon

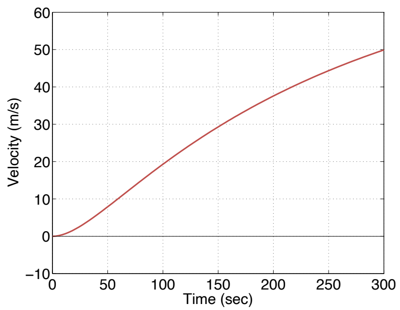
$\delta q =$ deviation in the burner heating rate from the equilibrium rate

Balloon parameters:

$\tau_1 = 250$ sec $\quad \tau_2 = 25$ sec $\quad a = 0.3$ m/(sec·°C)

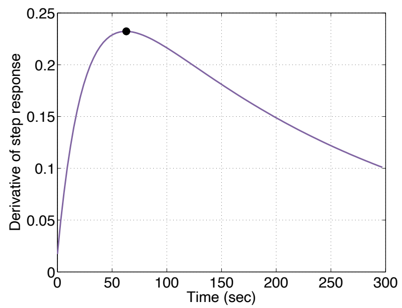
Balloon - Step Response

Tuning procedure: Turn the burner on full and measure vertical velocity.



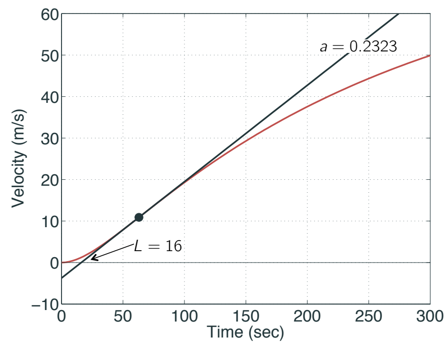
Step response

→



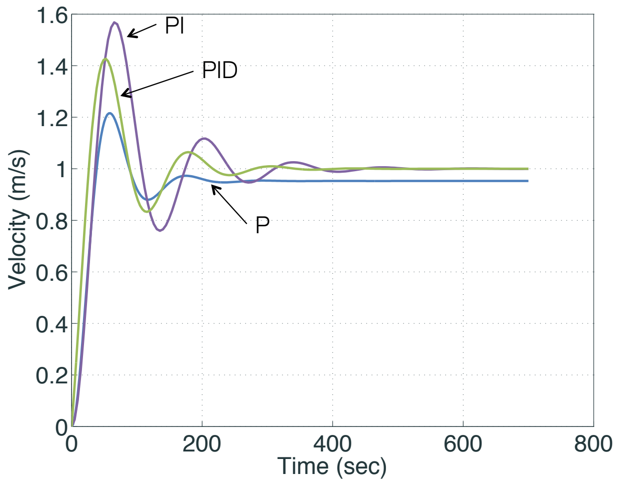
Derivative (acceleration)

Balloon - Zieger-Nichols Parameters



Type	K_P	T_i	T_d
P	$\frac{1}{aL} = 0.27$		
PI	$\frac{0.9}{aL} = 0.24$	$3.3L = 53.03$	
PID	$\frac{1.2}{aL} = 0.32$	$2L = 32.14$	$0.5L = 8.03$

Balloon - Closed-Loop Reponse

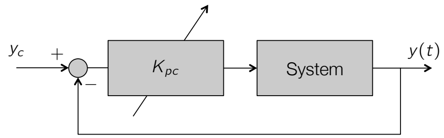


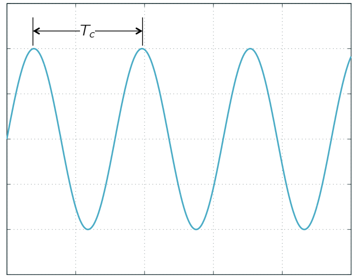
Zieger-Nichols tuning is often quite aggressive.

Zieger-Nichols Second Method - Unstable Systems

Why two methods? Can't apply a 'step' to an unstable system!

Solution: Stabilize the system with proportional controller first, and then tune





Type	K_P	T_i	T_d
P	$0.5K_{pc}$		
PI	$0.45K_{pc}$	$0.83T_c$	
PID	$0.6K_{pc}$	$0.5T_c$	$0.125T_c$

Parameters:

- K_{pc}: Gain at which the system becomes unstable
- T_c: Period of oscillation

Example - Balloon Altitude Control

Spirit of Freedom

Equations of motion:

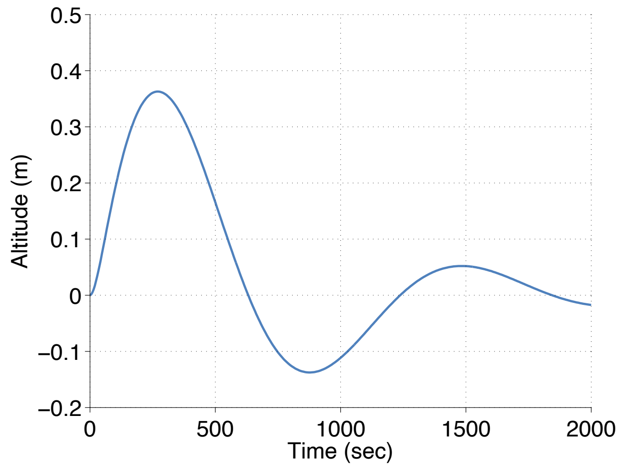
$$\delta\dot{T} + \frac{1}{\tau_1}\delta T = \delta q$$

$$\tau_2\ddot{z} + \dot{z} = a\delta T$$

$z =$ Altitude of balloon

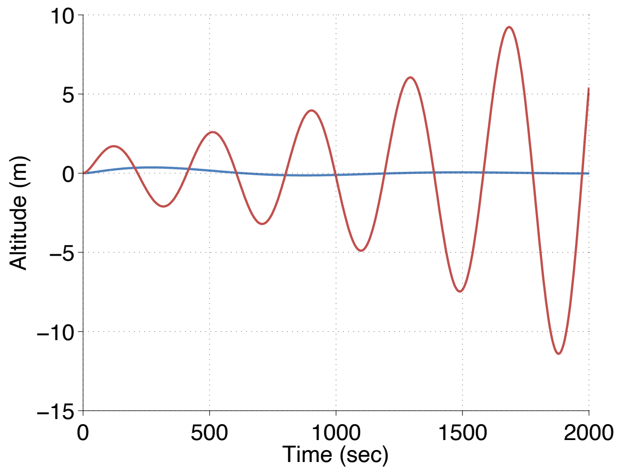
This is an unstable system.

Example - Balloon Altitude Control



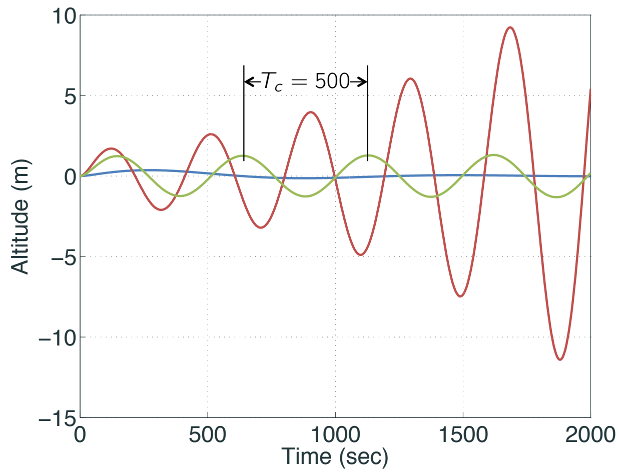
$K_P = 1 \times 10^{-4}$

Example - Balloon Altitude Control



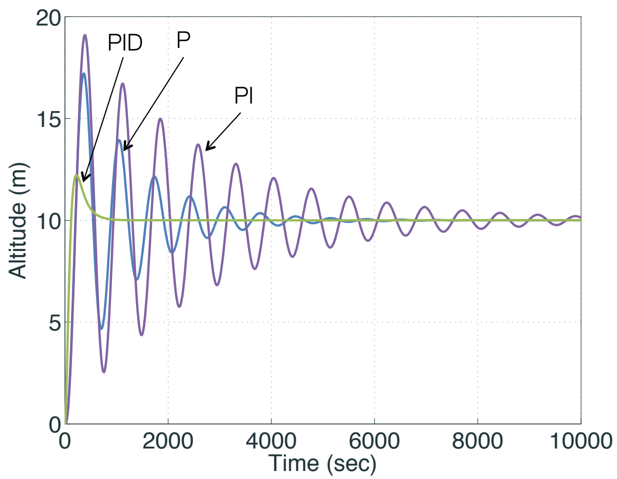
$K_P = 10 \times 10^{-4}$

Example - Balloon Altitude Control



$K_{PC} = 6 \times 10^{-4}$

Balloon Altitude Control - Closed-Loop Response



Zieger-Nichols Tuning - Summary

Simple method to determine reasonable PID tuning coefficients

- Method 1: Estimate delay and time constant from step response (stable systems)
- Method 2: Estimate gain at which the system becomes unstable, and the frequency of oscillation (unstable systems)
  - Limited to unstable systems that can be stabilized with a proportional controller

Limitations

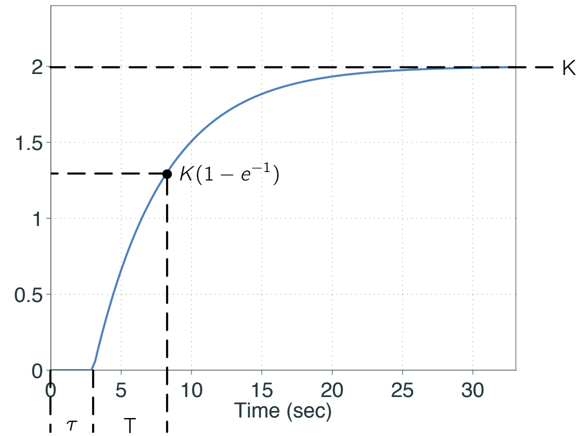
- Very simple, but also somewhat limited
- Based on information during the first portion of the step response - many systems are fast enough for more information to be available
- Fairly aggressive - normally good idea to reduce gains

Alternative Tuning Methods

Idea: Use More Information

Fit a parameterized curve to the step response:

$$P(s) = \frac{K}{sT+1}e^{-\tau s} \qquad p(t) = K(1 - e^{-\frac{t-\tau}{T}})$$



Choose a "Good" Set of Parameters

"Good" parameters for this Surrogate Model:

$$K_p = \frac{0.15\tau + 0.35T}{K\tau}$$
$$K_i = \frac{0.46\tau + 0.02T}{K\tau^2}$$

Idea: These gains give the same response for all surrogate model parameters

For the control structure:

$$C(s) = K_p + \frac{Ki}{s}$$

Note:

- Many other parameter values possible
- Several other surrogate models proposed

(Ziegler-Nichols parameters for same model: $K_p = 0.9T/K\tau$, $K_i = 0.5T/K\tau^2$)

Example: Balloon Velocity Control

Equations of motion:

$$\delta\dot{T} + \frac{1}{\tau_1}\delta T = \delta q$$
$$\tau_2\dot{v} + v = a\delta T$$

Compute transfer function:

$$\left(s + \frac{1}{\tau_1}\right)\delta T = \delta q \qquad (\tau_2 s + 1)v = a\delta T$$

$$v = \frac{a}{(\tau_2 s + 1)(s + 1/\tau_1)}\delta q = \frac{a}{\tau_2 s^2 + (1 + \tau_2/\tau_1)s + 1/\tau_1}\delta q$$

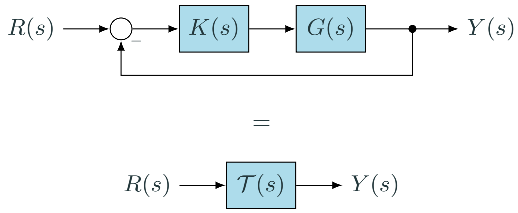
Balloon parameters:

$\tau_1 = 250$ sec $\qquad \tau_2 = 25$ sec $\qquad a = 0.3$ m/(sec·°C)

To Matlab!

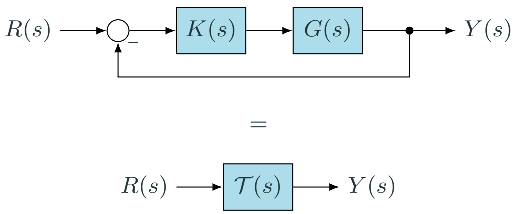
Model-Matching via PID

Model-matching



- The closed-loop system is a transfer function $\mathcal{T}(s)$ parameterized by $K(s)$
- Can we choose $K(s)$ to make the closed-loop system match a desired behaviour?

Model-matching

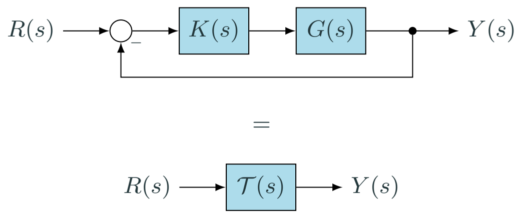


Compute $\mathcal{T}(s)$:

$$E(s) = R(s) - Y(s) \qquad\qquad Y(s) = G(s)K(s)E(s)$$

$$\Rightarrow \frac{Y(s)}{R(s)} = \frac{K(s)G(s)}{1 + K(s)G(s)} = \mathcal{T}(s)$$

Model-matching



$$\frac{Y(s)}{R(s)} = \frac{K(s)G(s)}{1 + K(s)G(s)} = \mathcal{T}(s)$$

$$\Rightarrow K(s) = \frac{\mathcal{T}(s)}{G(s)(1 - \mathcal{T}(s))}$$

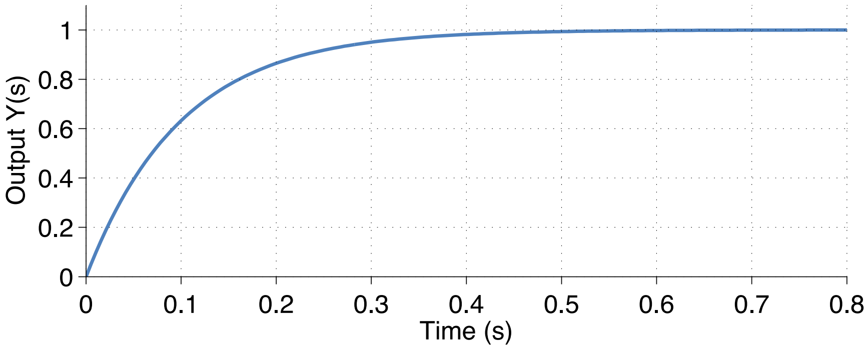
We can set $K(s)$ to give us the behaviour $\mathcal{T}(s)$.[a]

[a] There are a lot of limitations to this in general, which we will discuss later.

Matching a First-Order Response

Suppose we want to match the system

$$\mathcal{T}(s) = \frac{1}{\tau_m s + 1}$$



Step response of first-order system with time-constant $\tau_m = 0.1$

- Doesn't oscillate
- Gain of one

Controlling a First-Order System

Suppose that the system we're trying to control is

$$G(s) = \frac{\gamma}{\tau s + 1}$$

A system that moves when you 'push' it and:

- Does not oscillate
- Stops moving after some amount of time

Compute the controller:

$$K(s) = \frac{\mathcal{T}(s)}{G(s)(1 - \mathcal{T}(s))} = \frac{\frac{1}{\tau_m s+1}}{\frac{\gamma}{\tau s+1}\left(1 - \frac{1}{\tau_m s+1}\right)}$$
$$= \frac{\tau s + 1}{\gamma \tau_m s}$$
$$= \frac{\tau}{\gamma \tau_m}\left(1 + \frac{1}{\tau s}\right)$$

Controlling a First-Order System

$$K(s) = \frac{\tau}{\gamma \tau_m}\left(1 + \frac{1}{\tau s}\right)$$

This is a PI controller!

$$K_P = \frac{\tau}{\gamma \tau_m}$$
$$T_I = \tau$$

- We can choose how fast we want the closed-loop system to respond
- Simple 'tuning' procedure

First-Order System with Integral Action

Suppose we're controlling the system:

$$G(s) = \frac{\gamma}{\tau s + 1} \cdot \frac{1}{s}$$

A system that moves when you 'push' it and:

- Does not oscillate
- Continues moving at a constant speed forever

Compute the controller:

$$K(s) = \frac{\mathcal{T}(s)}{G(s)(1 - \mathcal{T}(s))} = \frac{\frac{1}{\tau_m s+1}}{\frac{\gamma}{s(\tau s+1)}\left(1 - \frac{1}{\tau_m s+1}\right)}$$
$$= \frac{1}{\gamma \tau_m}(\tau s + 1)$$

This is a PD controller

Second-Order System

Suppose we're controlling the system:

$$G(s) = \frac{\gamma}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

A system that moves when you 'push' it and:

- Does not oscillate
- Continues moving at a constant speed forever

Compute the controller:

$$K(s) = \frac{\tau_1 + \tau_2}{\gamma \tau_m}\left(1 + \frac{1}{(\tau_1 + \tau_2)s} + \frac{\tau_1 \tau_2}{\tau_1 + \tau_2} s\right)$$

This is a PID controller

Example - Balloon Velocity Control

Spirit of Freedom

Equations of motion:

$$\delta \dot{T} + \frac{1}{\tau_1}\delta T = \delta q$$

$$\tau_2 \dot{v} + v = a\delta T$$

δT = deviation of the hot-air temperature from the equilibrium temperature where buoyant force equals weight

v = vertical velocity of the balloon

δq = deviation in the burner heating rate from the equilibrium rate

Balloon parameters:

$\tau_1 = 250$ sec $\quad \tau_2 = 25$ sec $\quad a = 0.3$ m/(sec·°C)

Example - Balloon Velocity Control

Equations of motion:

$$\delta \dot{T} + \frac{1}{\tau_1}\delta T = \delta q$$

$$\tau_2 \dot{v} + v = a\delta T$$

Take Laplace transform:

$$\left.\begin{aligned} \delta T(s)\left(s + \frac{1}{\tau_1}\right) &= \delta Q(s) \\ V(s)(\tau_2 s + 1) &= a\delta T(s) \end{aligned}\right\} \rightarrow \frac{V(s)}{\delta Q(s)} = \frac{a\tau_1}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

Goal:

$$\mathcal{T}(s) = \frac{1}{\tau_m s + 1}$$

where $\tau_m = 10$s.

Example - Balloon Velocity Control

$$K(s) = \frac{\tau_1 + \tau_2}{\gamma \tau_m}\left(1 + \frac{1}{(\tau_1 + \tau_2)s} + \frac{\tau_1 \tau_2}{\tau_1 + \tau_2} s\right)$$

Balloon parameters:

$\tau_1 = 250$ sec $\quad \tau_2 = 25$ sec $\quad a = 0.3$ m/(sec·°C)

Desired system parameters:

$$\tau_m = 10$$

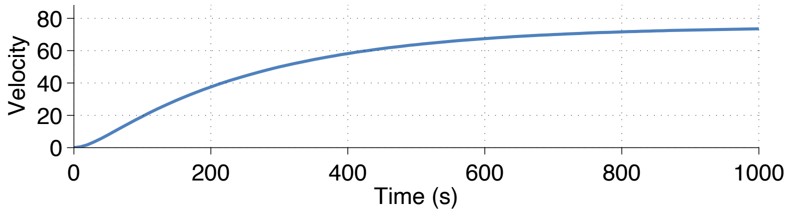
Resulting PID controller:

$$K(s) = \frac{275}{0.3 \cdot 10}\left(1 + \frac{1}{275s} + \frac{6250}{275}s\right)$$

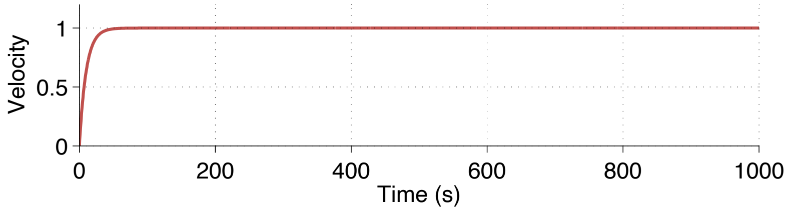
$K_P = 92 \quad T_i = 3 \quad T_d = 2083$

Example - Balloon Velocity Control

Open-loop behaviour



Closed-loop behaviour



Summary - Model Matching

The key idea:

- PID controller can make up to second order system behave as desired

- Many limitations on this statement:
  - Actuator limitations (speed, power, etc)
  - Physical constraints - may damage system if it's moved too fast, etc
- Many, many physical systems are approximately second order
  - Newton's law
  - Higher-order dynamics can often be ignored

Second Order Models

What are 'Good' Models?

Second-order systems are extremely common
(e.g., mass/spring/damper + Newton's law)

$$\ddot{x}(t) + 2\zeta\omega_n\dot{x}(t) + \omega_n^2 x(t) = \omega_n^2 u(t)$$

- ζ: Damping ratio
- ω_n: Natural frequency

The transfer function for this system is:

$$\frac{X(s)}{U(s)} = G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

What does the response of this system look like as a function of ζ and ω_n?

Second Order Systems

$$\frac{X(s)}{U(s)} = G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

where we assume that $\omega_n > 0$ and $\zeta > 0$.

Response to a unit step input $U(s) = \frac{1}{s}$:

$$X(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} U(s) = \frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)}$$

Note that the system has no steady-state offset for all ζ, ω_n:

$$\lim_{s\to 0} sX(s) = \lim_{s\to 0} s\frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} = \lim_{s\to 0} \frac{\omega_n^2}{\omega_n^2} = 1$$

Step Response

The roots of the ***characteristic polynomial*** $s^2 + 2\zeta\omega_n s + \omega_n^2$ are:

$$p = \omega_n(-\zeta \pm \sqrt{\zeta^2 - 1})$$

Three cases depending on damping ratio ζ:

1. $\zeta > 1$ Overdamped
2. $\zeta < 1$ Underdamped
3. $\zeta = 1$ Critically damped

Case One: Overdamped

When $\zeta > 1$ we call the system ***overdamped***

The system has two real, distinct poles p_1 and p_2

$$p_1 = \omega_n(-\zeta + \sqrt{\zeta^2 - 1}) \qquad p_2 = \omega_n(-\zeta - \sqrt{\zeta^2 - 1})$$

The partial-fraction expansion is:

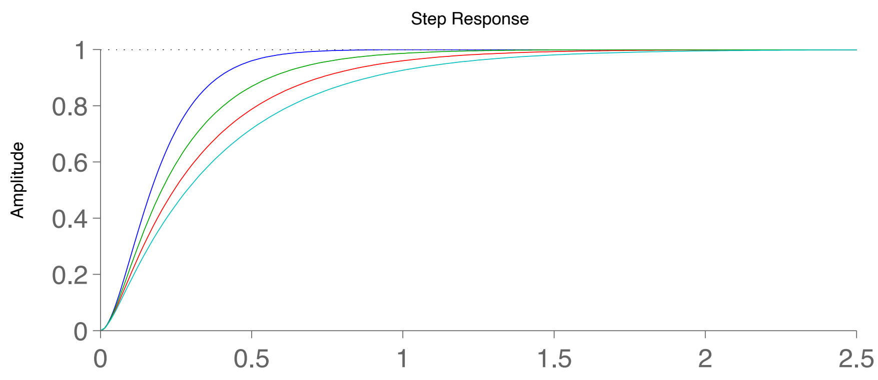
$$X(s) = \frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} = \frac{a_1}{s - p_1} + \frac{a_2}{s - p_2} + \frac{1}{s}$$

The inverse Laplace transform is:

$$x(t) = a_1 e^{p_1 t} + a_2 e^{p_2 t} + 1$$

Note that both p_1 and p_2 are negative, since $\zeta > 1$. Therefore both exponential terms decay.

Case One: Overdamped



Larger values of ζ have a slower response.

Case Two: Critically Damped

Assume $\zeta = 1$.
One repeated pole:

$$p_1 = p_2 = s = \omega_n(-\zeta \pm \sqrt{\zeta^2 - 1}) = \omega_n$$

The partial-fraction expansion is:

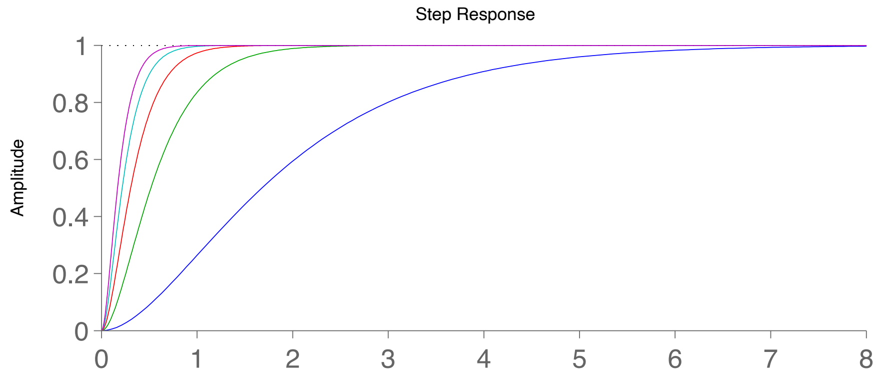
$$X(s) = \frac{\omega_n^2}{s(s+\omega_n)^2} = \frac{-1}{s+\omega_n} + \frac{-\omega_n}{(s+\omega_n)^2} + \frac{1}{s}$$

The inverse Laplace transform is:

$$-e^{-\omega_n t} - \omega_n t e^{-\omega_n t} + 1$$

Since $\omega_n > 0$, the exponential terms will always go to zero for all ω_n.

Case Two: Critically Damped



Larger values of ω_n have a faster response

Case Three: Underdamped

Assume $0 \leq \zeta < 1$

The poles are complex:

$$p = \omega_n(-\zeta \pm j\sqrt{1-\zeta^2})$$

The inverse Laplace transform from the table is[4]

$$x(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin\left(\omega_n\sqrt{1-\zeta^2}t + \theta\right)$$

where $\theta = \cos^{-1}\zeta$

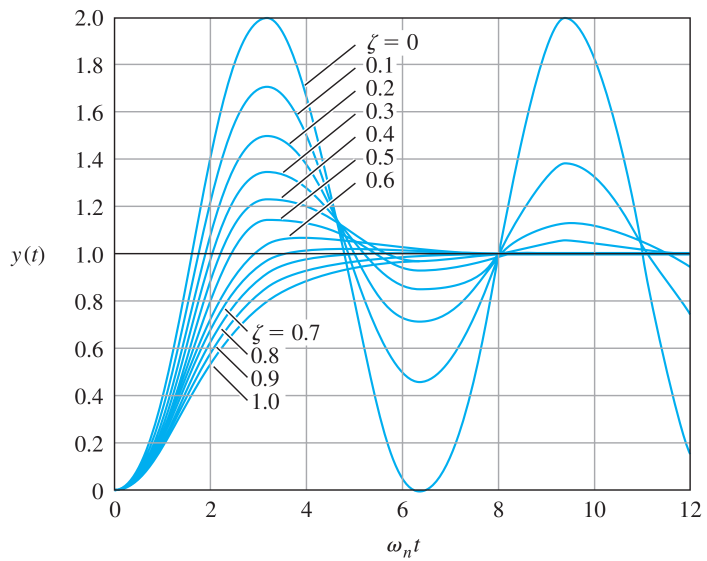
[4]Or you can derive from the frequency-shift property, and knowing the transform of the sine function.

Case Three: Underdamped

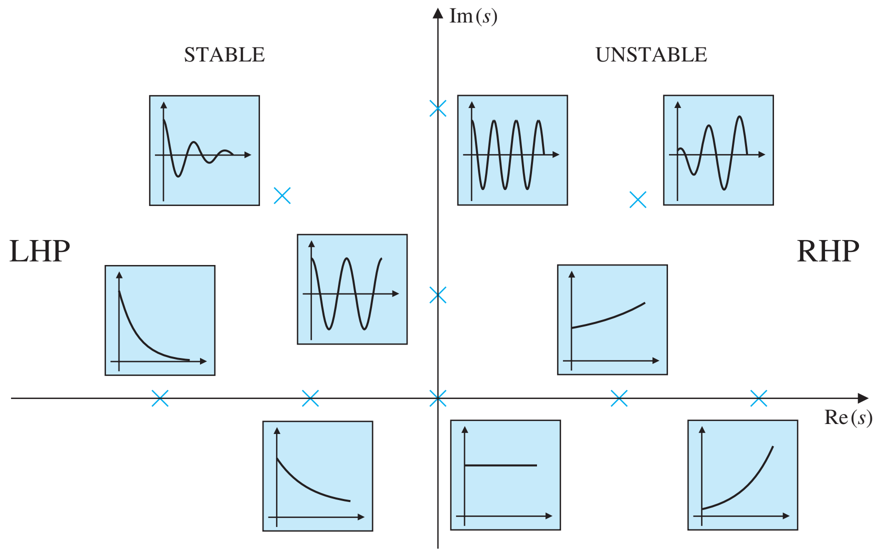
$$x(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin\left(\omega_n\sqrt{1-\zeta^2}t + \theta\right)$$

- The signal oscillates, but decays to one
- The frequency of oscillation is the damped frequency $\omega_d := \omega_n\sqrt{1-\zeta^2}$
- The signal decays at an exponential rate of $e^{-\sigma t}$, where $\sigma = \zeta\omega_n$

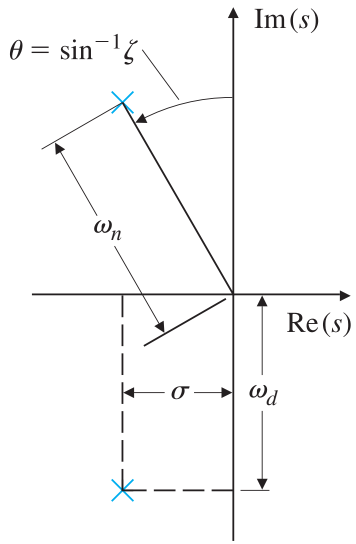
Case Three: Underdamped



(b)



Visualization: The Pole-Zero Diagram



Pole location determines the behaviour of the system

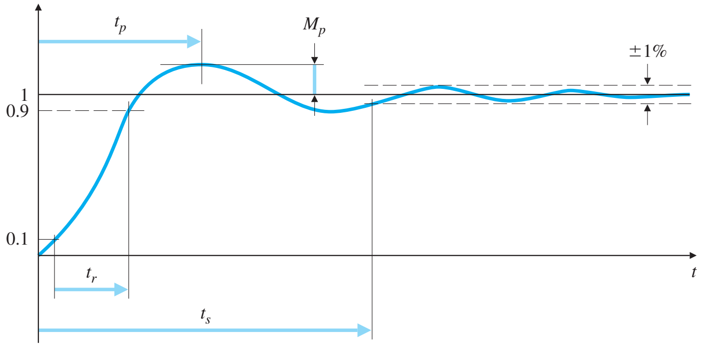
- Magnitude of the real component: decay rate
  - Larger: faster decay
- Magnitude of the complex component: frequency of oscillation
  - Larger: Faster oscillation
- Magnitude of the pole: natural frequency
- Angle of the pole: $\sin^{-1}\zeta$

What are good choices for pole locations?

To Matlab! `pzLocations`

- Impact of ω_d
- Impact of σ
- Impact of ζ

Characterization of Second Order Systems

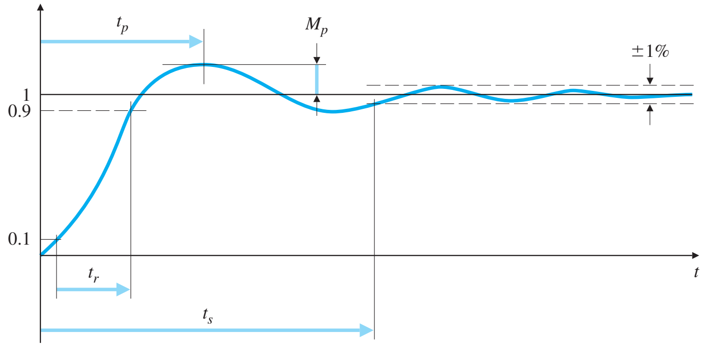


Peak time T_p. Time to get to the maximum value.

$$T_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} = \frac{\pi}{\omega_d}$$

e.g., constraint: $T_p \leq 1.5 \Leftrightarrow \omega_d \geq \frac{\pi}{1.5}$

Characterization of Second Order Systems

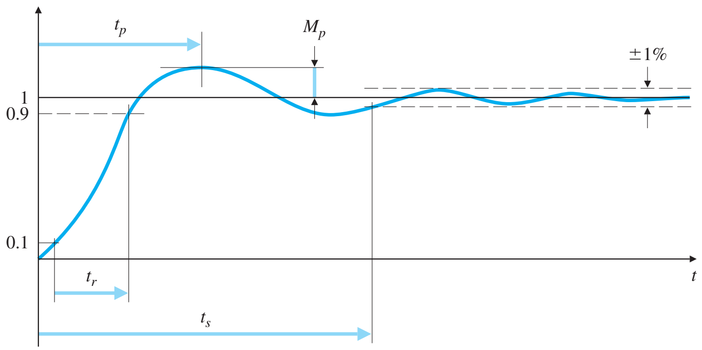


Percent overshoot $P.O.$.

$$P.O. := M_p \times 100\% = 100e^{-\zeta\pi/\sqrt{1-\zeta^2}}$$

e.g., constraint $M_p < 20\% \Leftrightarrow \zeta \geq -\frac{\ln(M_p)}{\sqrt{\ln(M_p)^2+\pi^2}} = 0.45$

Characterization of Second Order Systems

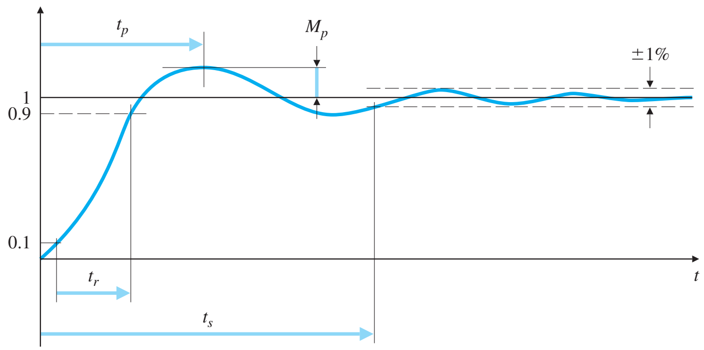


Settling time T_s. Time to settle to within δ percent of the steady-state value. e.g., if $\delta = 2\%$

$$T_s = \frac{-\log\delta}{\zeta\omega_n} = \frac{4}{\zeta\omega_n} = \frac{4}{\sigma}$$

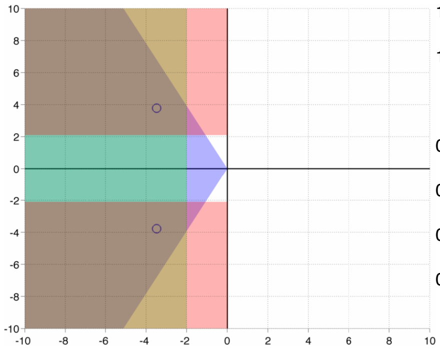
e.g., constraint: $T_s \leq 4 \Leftrightarrow \sigma \geq \frac{4}{T_s} = 1$

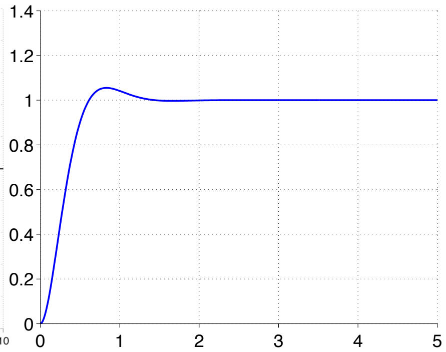
Characterization of Second Order Systems



Rise time T_r. Time to get to 90% of final value from 10%

Characterization of Second Order Systems





- $T_p \leq 1.5$
- $M_p \leq 20\%$
- $T_s \leq 4$s

Second-Order Models: Summary

$$\ddot{x}(t) + 2\zeta\omega_n\dot{x}(t) + \omega_n^2 x(t) = \omega_n^2 u(t) \qquad \frac{X(s)}{U(s)} = G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

- ζ: Damping ratio
- ω_n: Natural frequency

- Many systems can be described with such a model.
- If your system is higher order, the general behaviour can often be described by the *dominant poles* (the most unstable ones - those closest to the imaginary axis)
- Common performance parameters can be set by appropriate selection of ω_n and ζ.

Second-Order Models: Summary

$$\ddot{x}(t) + 2\zeta\omega_n\dot{x}(t) + \omega_n^2 x(t) = \omega_n^2 u(t) \qquad \frac{X(s)}{U(s)} = G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

- ζ: Damping ratio
- ω_n: Natural frequency

- Many systems can be described with such a model.
- If your system is higher order, the general behaviour can often be described by the *dominant poles* (the most unstable ones - those closest to the imaginary axis)
- Common performance parameters can be set by appropriate selection of ω_n and ζ.

How do we choose the PID weights so that we can meet specific criteria?

- Ziegler-Nichols tuning + manual adjustments (root locus)
- Model-matching
- Methods in later lectures (generally requires higher-order controllers)

Example

Example

Suppose that we have a system which takes a force, and outputs a position:

$$G(s) = \frac{V(s)}{U(s)} = \frac{21.53}{s^4 + 1.833s^3 + 70.28s^2 + 69.44s}$$

Control the position of this system using a PD controller such that:

- Over shoot is less than $M_P = 40\%$
- Settling time T_s is below 10s
- Peak-time T_p is below 4s

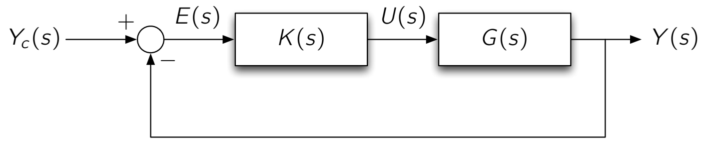
Note: The transfer function to velocity is

$$G'(s) = \frac{21.53}{s^3 + 1.833s^2 + 70.28s + 69.44}$$

There is already an integrator here, so we're using a PD controller.

Method 1: Root-Locus Design

Goal: Choose K_p so that our closed-loop poles are in the right place.
Idea: Plot the poles of the closed-loop system as a function of the gain K_p



The closed-loop system is:

$$Y(s) = G(s)K(s)(R(s) - Y(s)) \qquad \frac{Y(s)}{R(s)} = \frac{G(s)K(s)}{1 + G(s)K(s)}$$

Equivalently:

$$G(s) = \frac{A(s)}{B(s)} \qquad K(s) = \frac{C(s)}{D(s)} \qquad \Rightarrow \frac{Y(s)}{R(s)} = \frac{A(s)C(s)}{B(s)D(s) + A(s)C(s)}$$

Root-Locus Design

Our controller is:

$$K(s) = K_p(1 + T_d s)$$

Suppose we've chosen $T_d = 0.01$, and we're looking for a good K_p

Our closed-loop poles are given by the roots of the characteristic equation:

$$B(s)D(s) + A(s)C(s) = \\ s^4 + 1.833s^3 + 70.28s^2 + 69.44s + K_p 21.53(1 + 0.01s) := f(s)$$

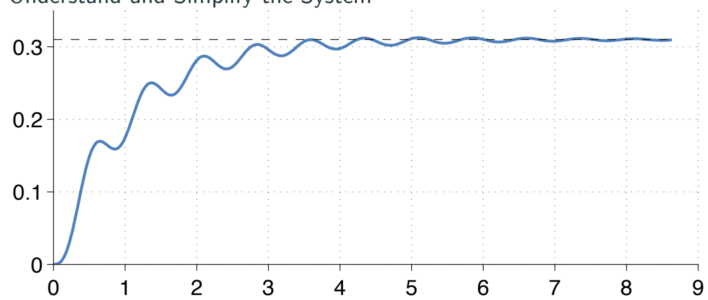
We can plot how the four poles of the closed-loop system move in response to changes in K_P. This is the root-locus diagram.

To Matlab! `sol_rlocus.m`

Method 2: Pole-Placement

Can we directly place the dominant poles of this system where we want?

Step 1: Understand and Simplify the System



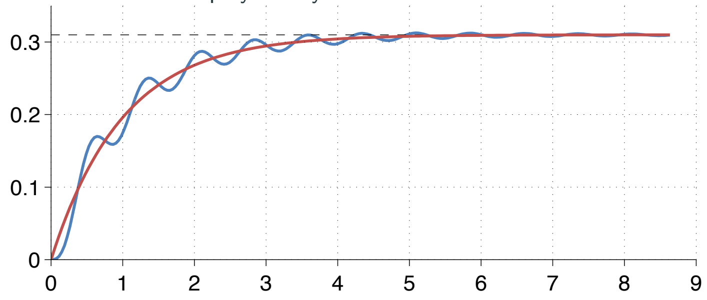
$$G'(s) = \frac{21.53}{s^3 + 1.833s^2 + 70.28s + 69.44}$$

System is complex, but there is clearly a ***dominant mode***

Method 2: Pole-Placement

Can we directly place the dominant poles of this system where we want?

Step 1: Understand and Simplify the System



Much simpler system that captures the main properties

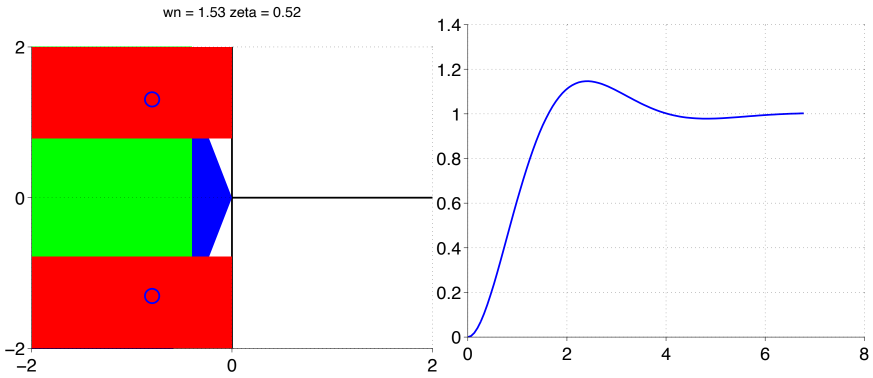
$$P(s) = \frac{0.31}{s+1} \approx G'(s)$$

Very common to neglect the 'higher order dynamics'

Target System

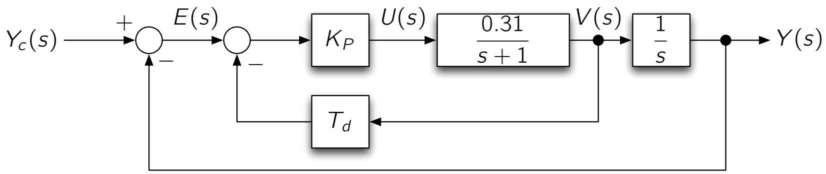
Compute a second order system that satisfies the specified conditions:

- Over shoot is less than $M_P = 40\%$
- Settling time T_s is below 10s
- Peak-time T_p is below 4s



$\zeta \approx 0.52$ $\omega_n \approx 1.53$

PD Control Structure



Closed-loop transfer function:

$$Y(s) = \frac{1}{s}\frac{0.31}{s+1}K_P(E(s) - T_d sY(s)) \qquad E(s) = R(s) - Y(s)$$

$$\frac{Y(s)}{R(s)} = \frac{0.31K_p}{s^2 + (1 + 0.31K_pT_d)s + 0.31K_p}$$

Two parameters to choose, and two parameters to set

$\therefore$ we can choose any response we like!

PD Control Structure

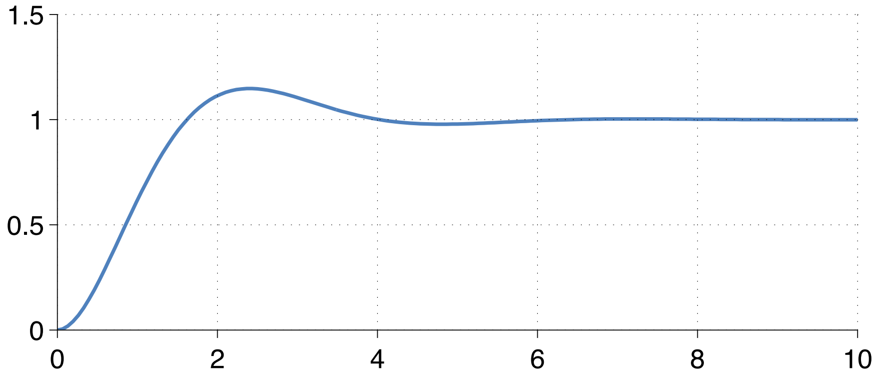
$$\frac{Y(s)}{R(s)} = \frac{0.31K_p}{s^2 + (1 + 0.31K_pT_d)s + 0.31K_p}$$

$$= \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad \text{Desired response}$$

where $\zeta \approx 0.52$, $\omega_n \approx 1.53$

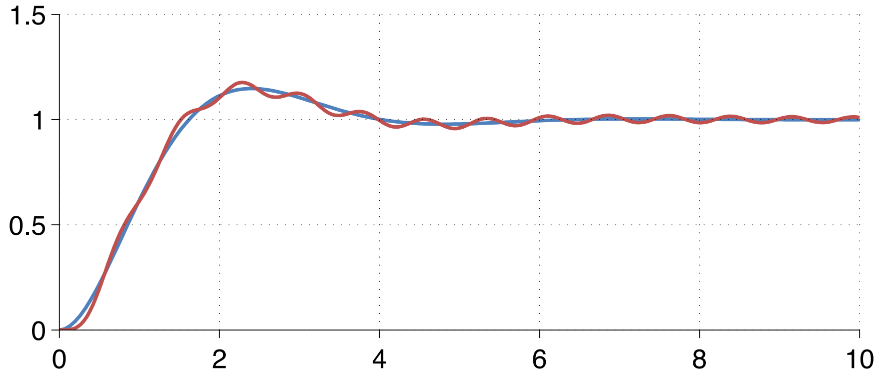
$$K_P = \frac{\omega_n^2}{0.31} = 7.55 \qquad T_d = \frac{2\zeta\omega_n - 1}{0.31K_P} = 0.25$$

Pole Placement Result



Closed-loop response of simplified system

Pole Placement Result



Real closed-loop system with controller

Anti-Windup

Input Constraints

All real systems have ***input constraints***

All the controllers you've seen assume that they do not

This is a problem!

Consider the simple system:

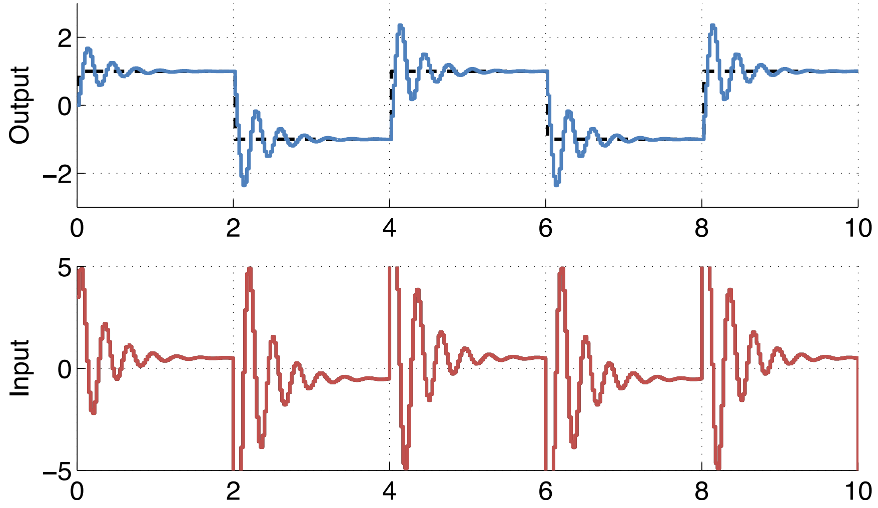
$$G(s) = \frac{100}{s+50}$$

with a PI controller

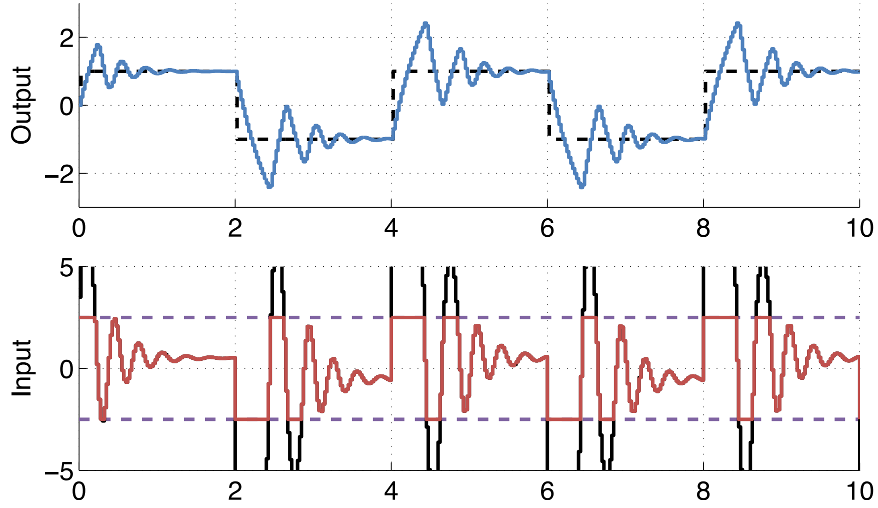
$$K(s) = K_P\left(1 + \frac{1}{T_i s}\right)$$

with $K_p = 3.5$ and $T_i = 0.01$.

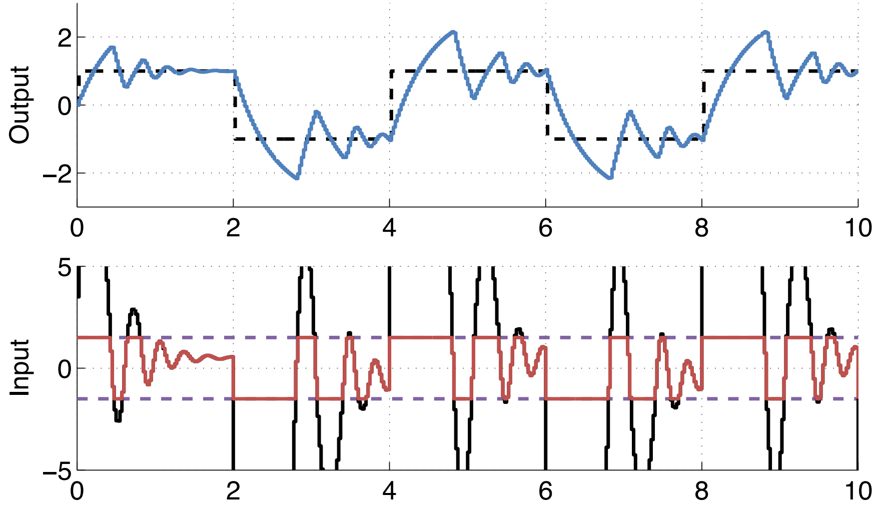
Example : Impact of Constraints



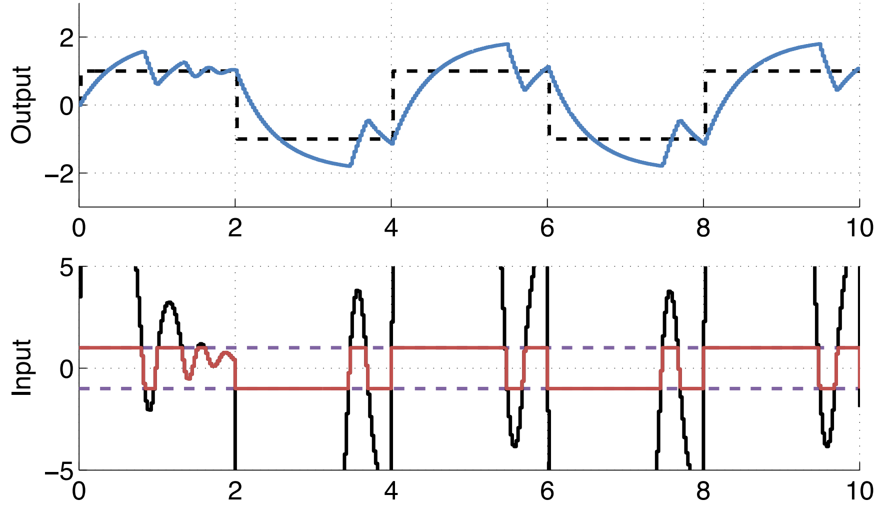
Example : Impact of Constraints



Example : Impact of Constraints



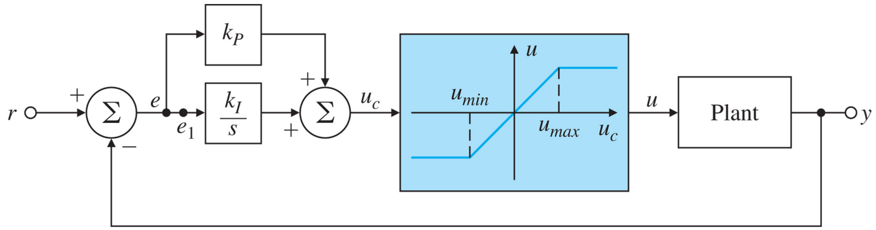
Example : Impact of Constraints



Saturation

No matter what we do, the input will satisfy the condition called ***saturation***:[5]

$$u(t) = \begin{cases} u_{\max} & \text{if } u(t) > u_{\max} \\ u(t) & \text{if } u(t) \in [u_{\min}, u_{\max}] \\ u_{\min} & \text{if } u(t) < u_{\min} \end{cases}$$

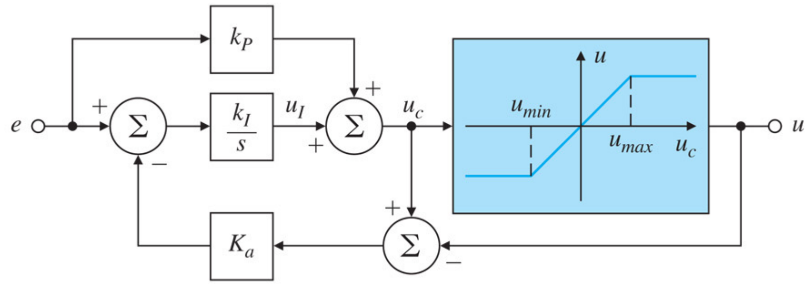


[5] We've written the saturation here as a symmetric term. It is also possible to have asymmetric saturation.

Anti-Windup

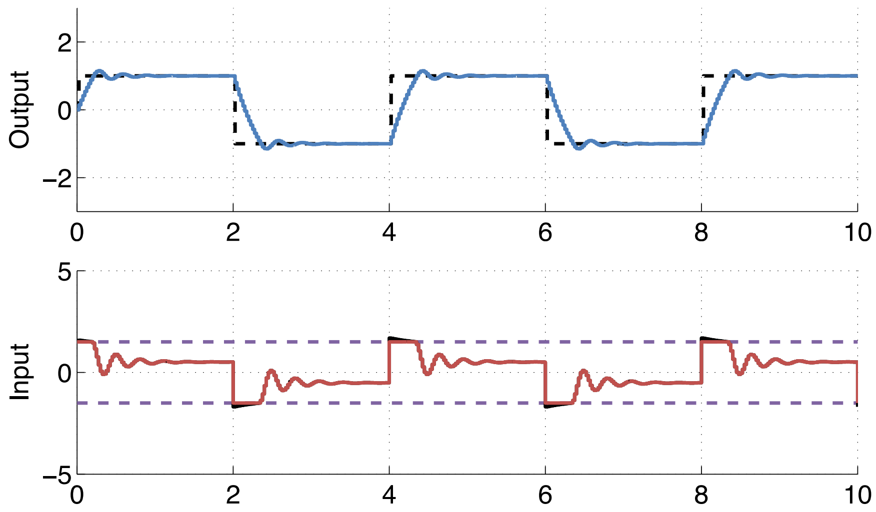
Preventing the integrator from growing or 'winding up' is called ***anti-windup***

Idea: Detect when saturation is active, and turn off the integrator



- Only impacts the system when constraints are active
- Relatively simple to tune
- Can be implemented in continuous-time (traditional reason)

Example : Impact of Constraints



PID - Summary

PID controllers are extremely useful:

- Used in the vast majority of simple systems
- Often the 'lowest-level' of control. More complex control built on top

A great deal of good literature available on tuning commercial PID controllers

Proportional ▪ Sets the 'aggressiveness' of your system

Integral ▪ Added to ensure zero steady-state offset

Derivative ▪ Increase the damping of the system - improve stability

Impact of PID terms:

PID Gain	Percent Overshoot	Settling Time	Steady-State Error
Increasing K_P	Increases	Minimal impact	Decreases
Increasing K_I	Increases	Increases	Zero steady-state error
Increasing K_d	Decreases	Decreases	No impact