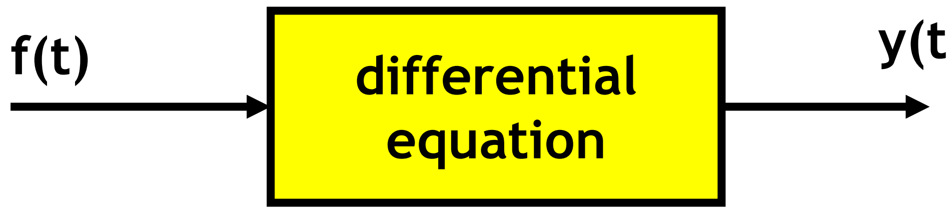
ROCO218: Control Engineering
Dr Ian Howard

Lecture 4

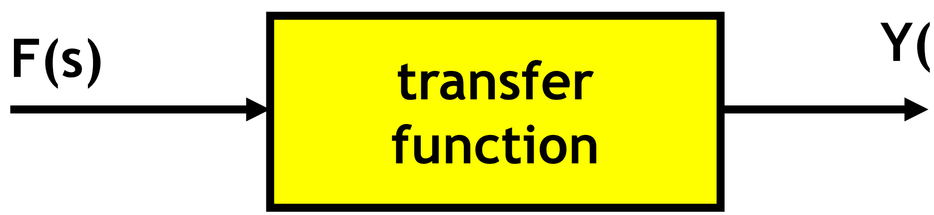
Transfer Functions

Transfer functions

- A differential equation is an equation which contains derivative terms
- Differential equation can often be used to describe a dynamical system in the time domain
- For example, a differential equation can capture the relationship between the force input f(t) and output position x(t) of a mass-spring damper

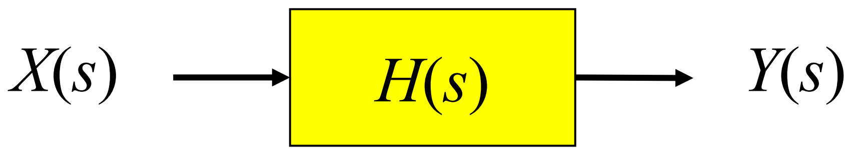


- A transfer function is an expression that describes a system the s-domain that relates the output to the input
- To get to the s-domain we use the Laplace transform



Transfer functions

- Consider a system in the s-domain with input X(s) and output Y(s)



- Given the s-domain input X(s) and output Y(s) the transfer function H(s) is given by the ratio of the output Y(s) to the input X(s)

$$H(s) = Y(s) / X(s)$$

- The output of the system can Y(s) therefore be found by multiplying the input X(s) with the transfer function H(s)

$$Y(s) = X(s)H(s)$$

For example, if the input signal is a step then

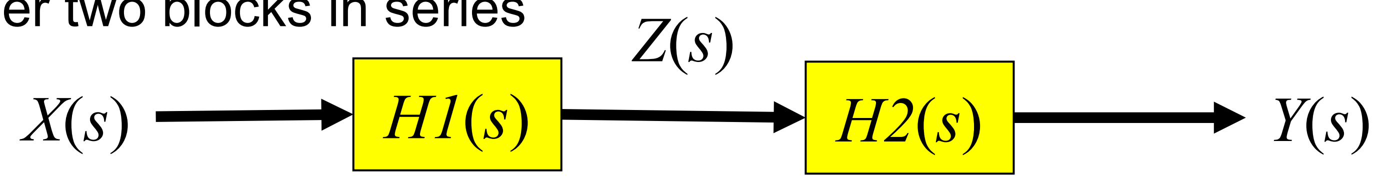
$$X(s) = 1/s$$

Therefore in this case

$$Y(s) = H(s)/s$$

Transfer functions in series

- We can use block diagrams to pictorially expresses flows and relationships between elements in system represented by their transfer functions
- Such signal flow graphs can often be simplify using simple rules
- Consider two blocks in series



Looking at midpoint we see that

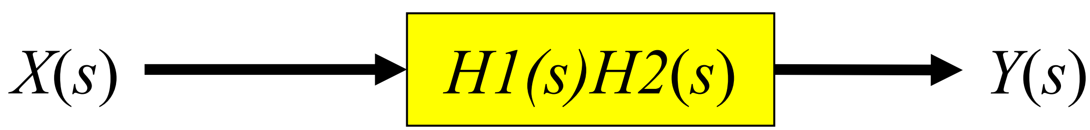
$$Z(s) = X(s)H1(s)$$

$$Y(s) = Z(s)H2(s)$$

Therefore

$$Y(s) = X(s)H1(s)H2(s)$$

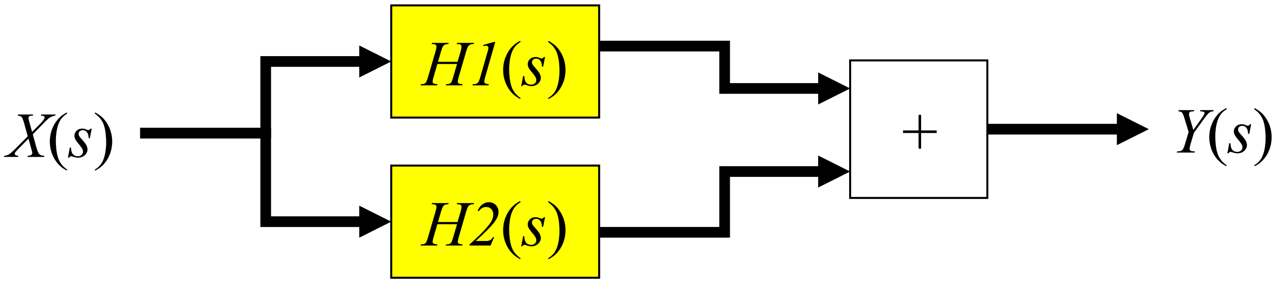
So we can represent H1(s) and H2(s) in series by a single block with a transfer function equal to their product H1(s)H2(2)



This result generalized to multiple blocks in series

Transfer functions in parallel

- Consider two blocks in parallel



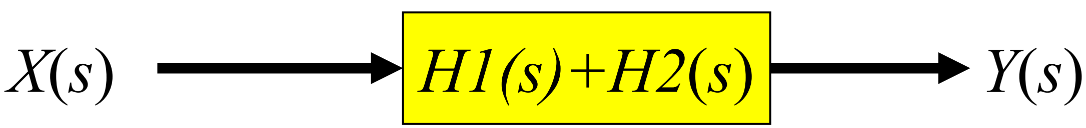
The output from the summer is

$$Y(s) = X(s)H1(s)+ X(s)H2(s$$

Therefore

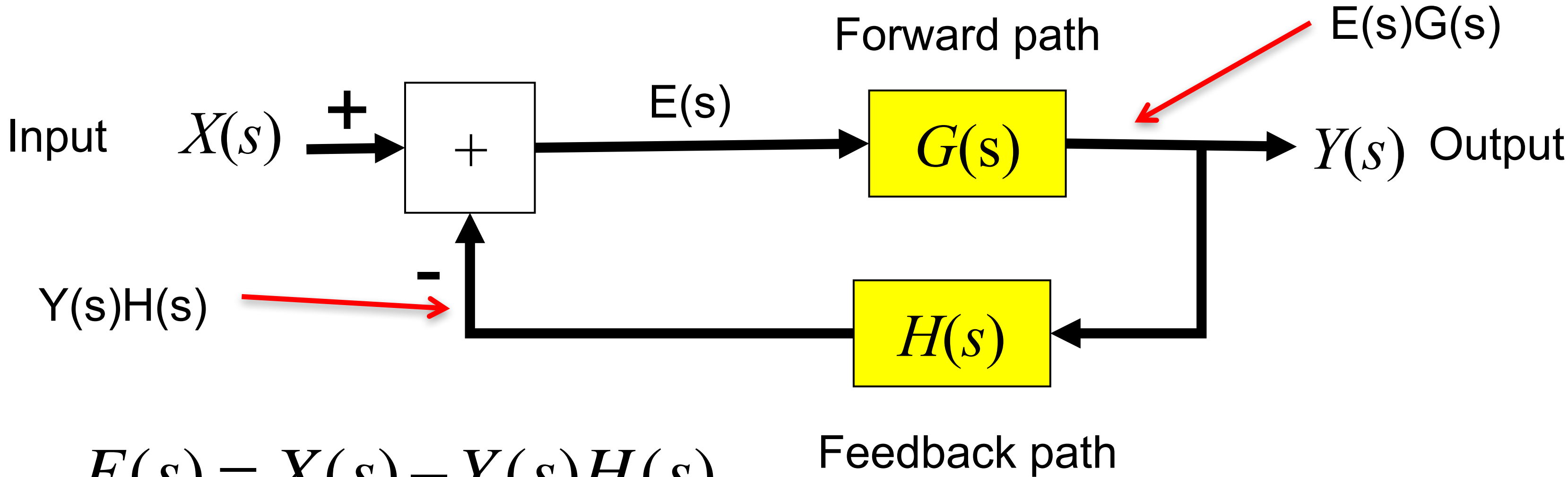
$$Y(s) = X(s)(H1(s)+H2(s))$$

So we can represent H1(s) and H2(s) in series by a single block with a transfer function equal to their sum H1(s) + H2(2)



This result generalized to multiple blocks in parallel and also to subtraction rather than addition

Closed loop system



$$E(s) = X(s) - Y(s)H(s)$$

$$Y(s) = E(s)G(s)$$

Substituting in error term E(s)

$$\Rightarrow Y(s) = \big(X(s) - Y(s)H(s)\big)G(s) \quad = X(s)G(s) - Y(s)H(s)G(s)$$

Collecting factors of Y(s) terms

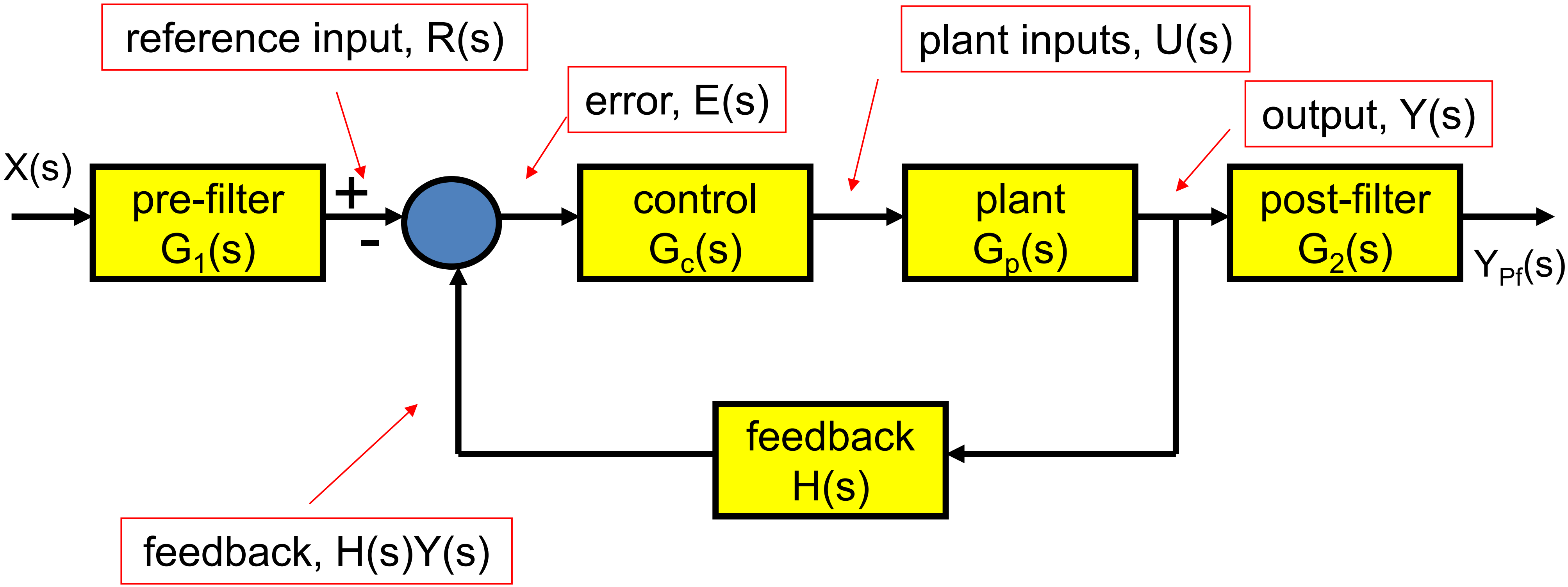
$$\Rightarrow Y(s)(1 + H(s)G(s)) = X(s)G(s)$$

Calculating the transfer function

$$\frac{Output}{Input} = \frac{Y(s)}{X(s)} \quad = \frac{G(s)}{\big(1 + G(s)H(s)\big)}$$

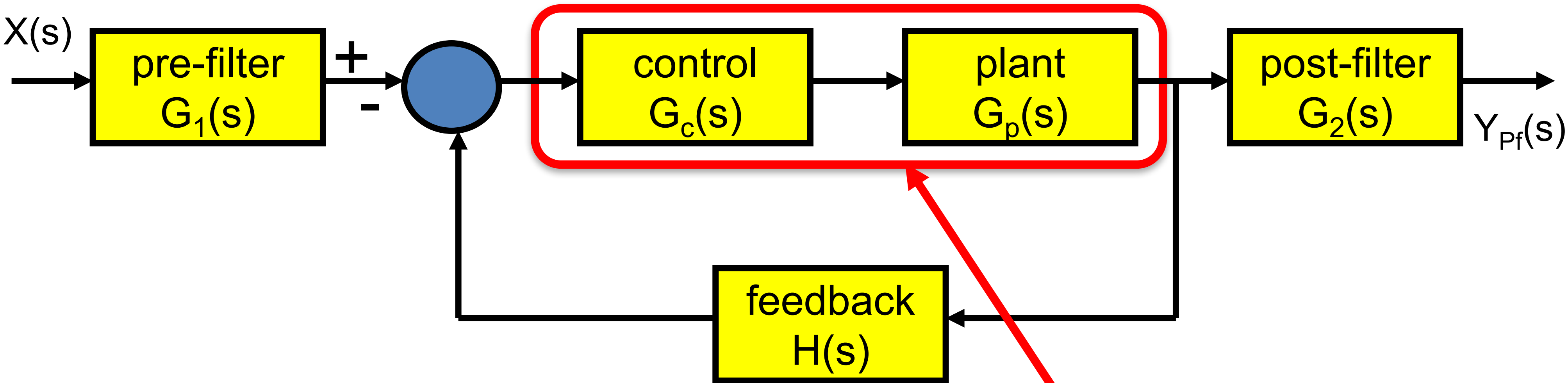
This is a very important result for feedback control!

Typical block diagram for simple feedback controller



- Let us compute the overall transfer function of this system
- To do so we first simplify the structure

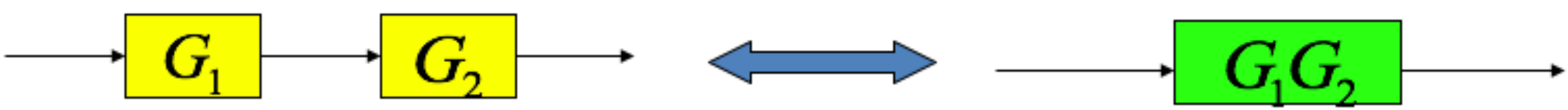
Typical block diagram for simple feedback controller



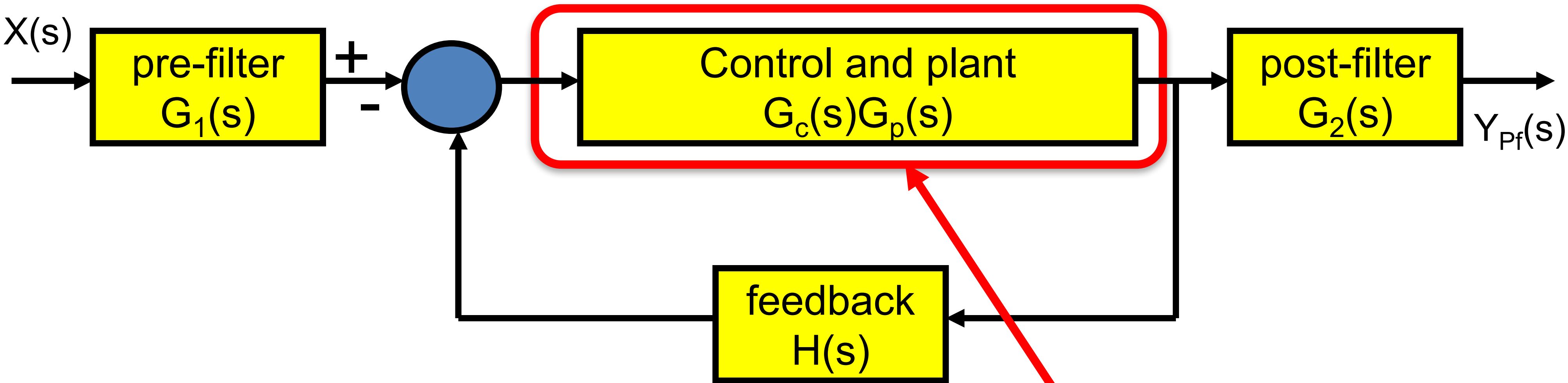
- Let us compute the overall transfer function of this system

- To do so we first simplify the structure

- Lets consider the control and the plant
- Can be combined into a single block



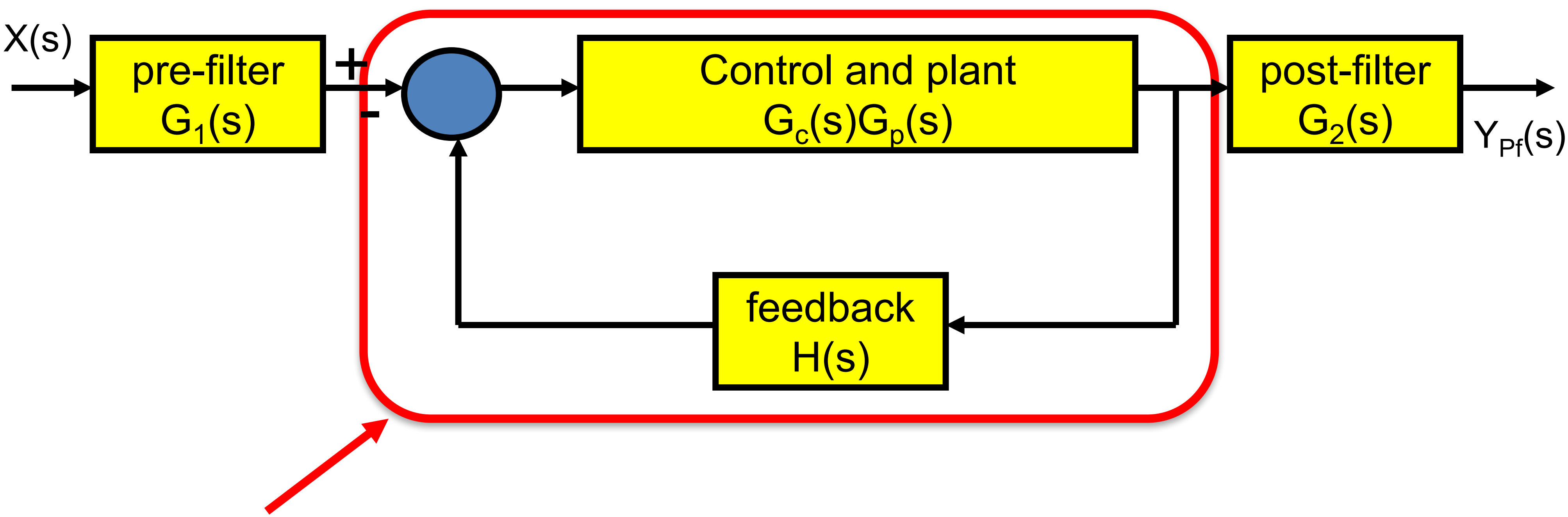
Typical block diagram for simple feedback controller



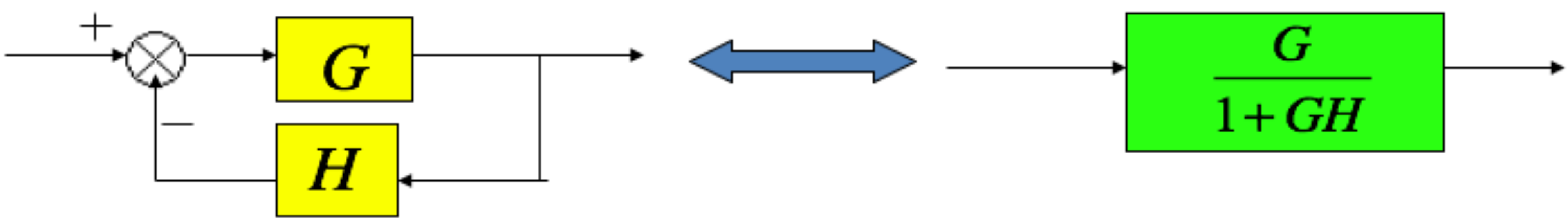
- Let us compute the overall transfer function of this system
- To do so we first simplify the structure

- Lets consider the control and the plant
- Can be combined into a single block

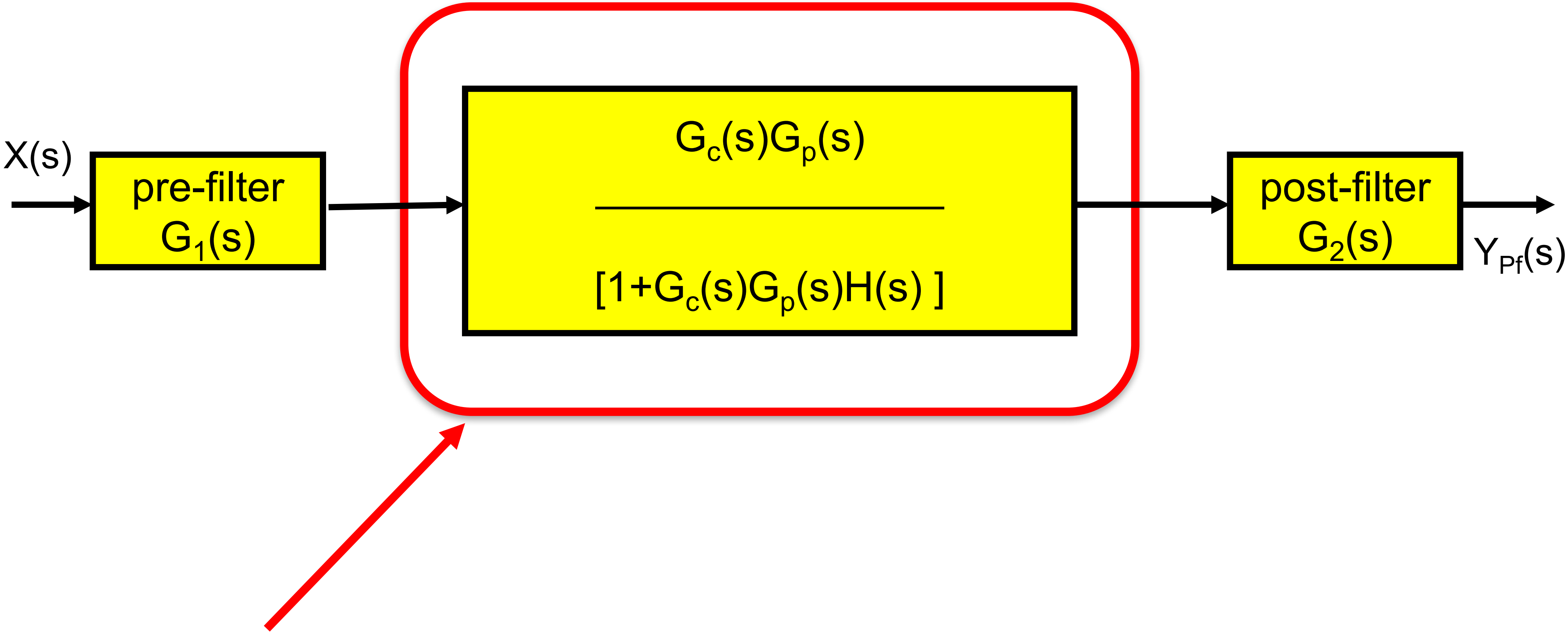
Typical block diagram for simple feedback controller



- Now consider the feedback loop
- Can be combined into a single block

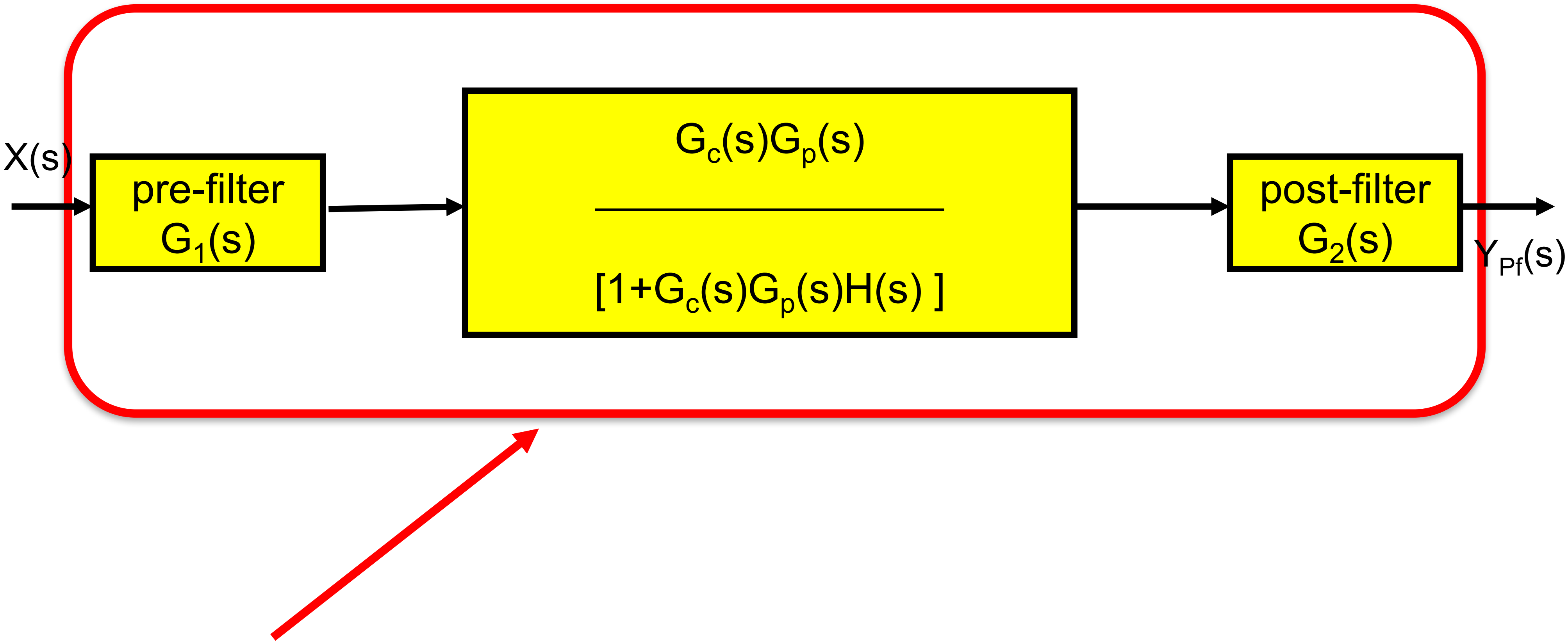


Typical block diagram for simple feedback controller

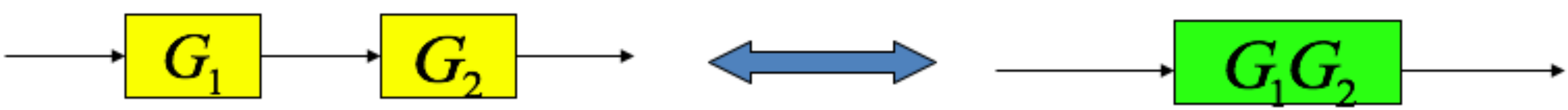


- Now consider the feedback loop
- Can be combined into a single block

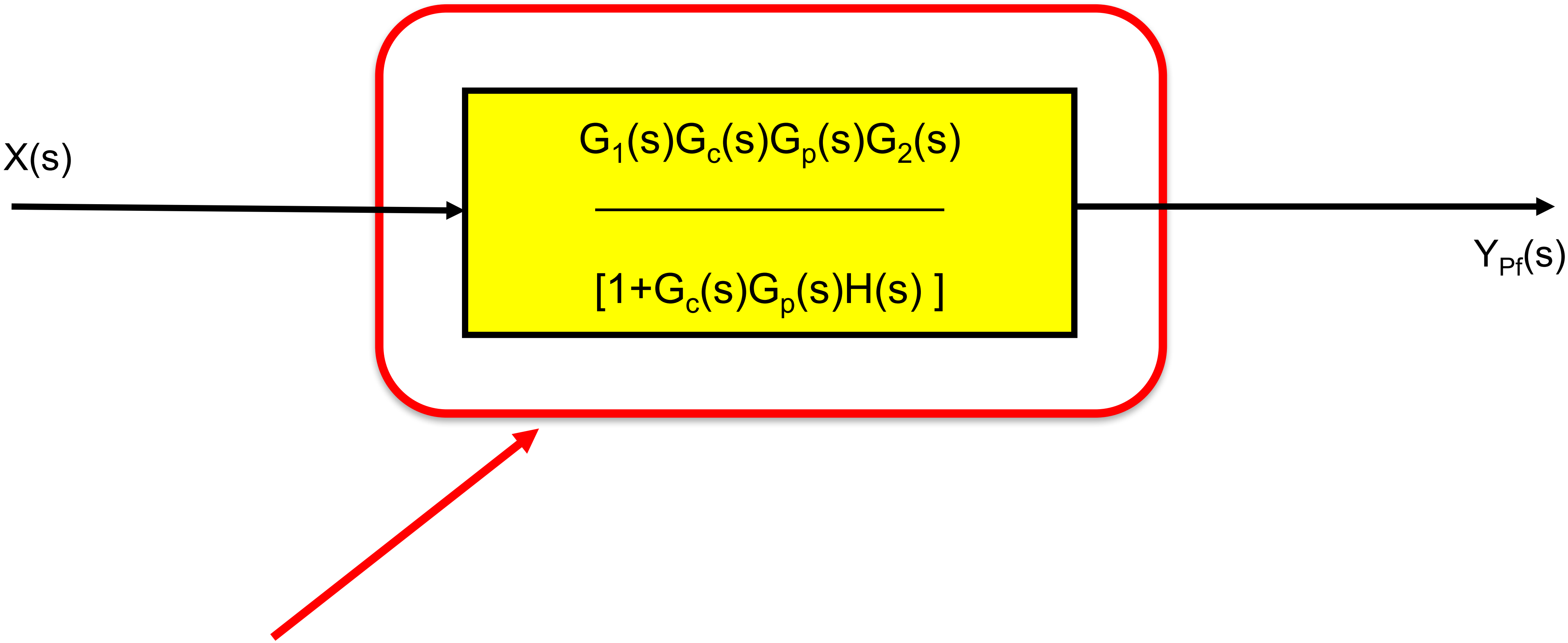
Typical block diagram for simple feedback controller



- Now consider the three blocks in series
- These can be combined into a single block



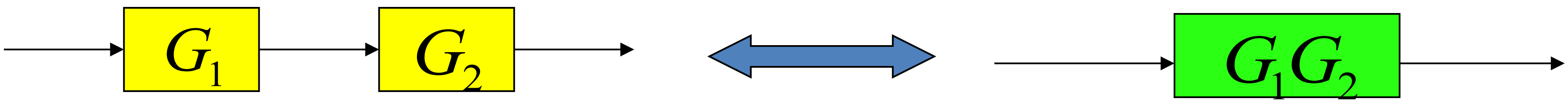
Typical block diagram for simple feedback controller

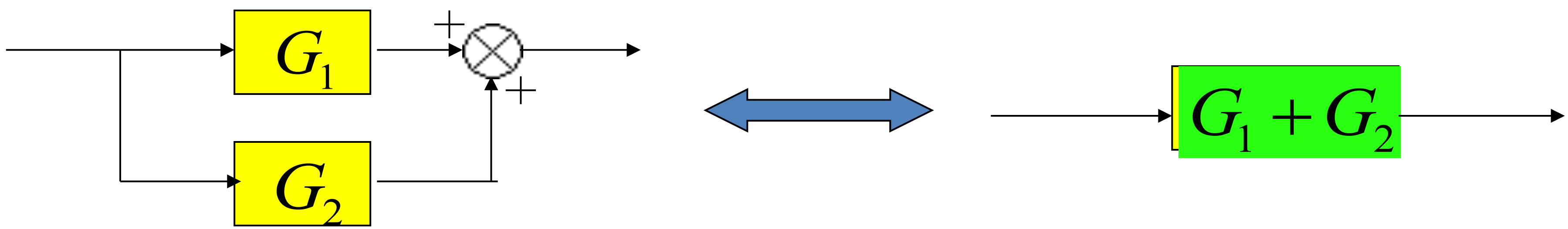


- Now consider the three blocks in series
- These can be combined into a single block

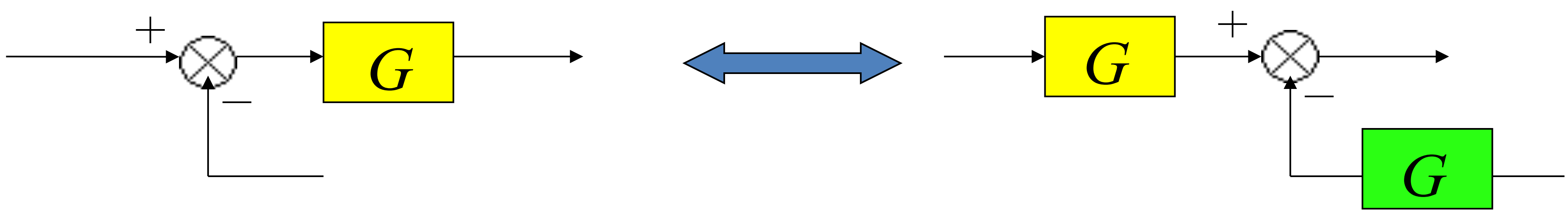
Block diagram reduction summary

1. Combining blocks in cascade or in parallel



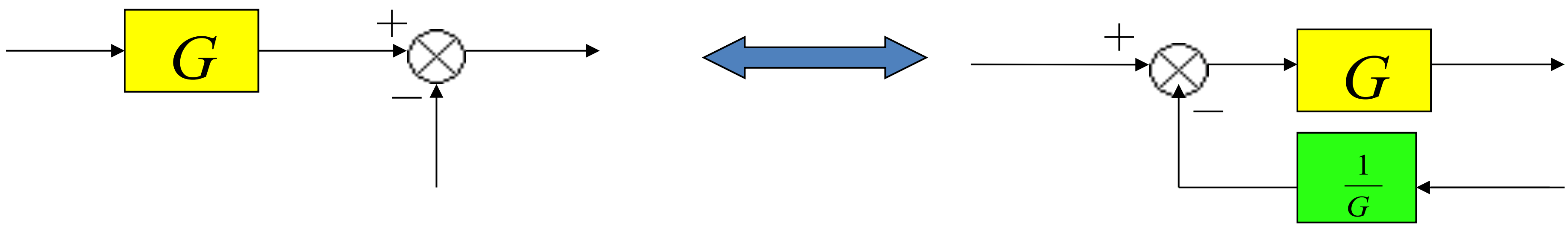


2. Moving a summing point from behind a block

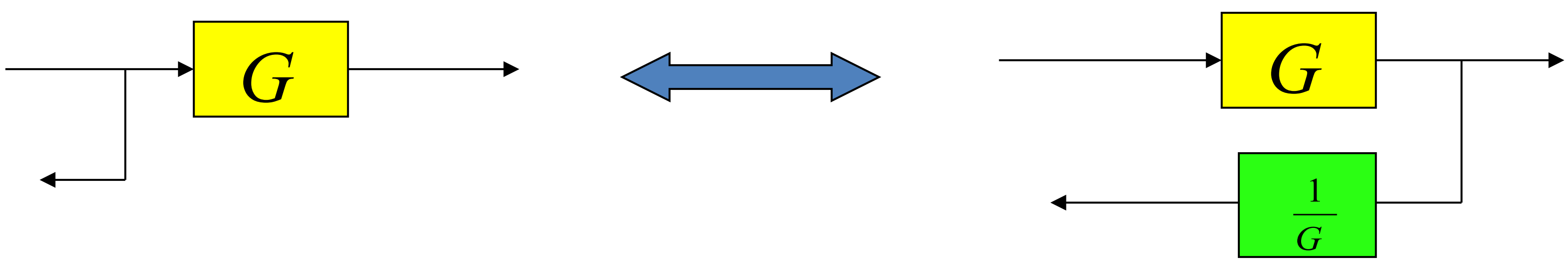


Block diagram reduction summary

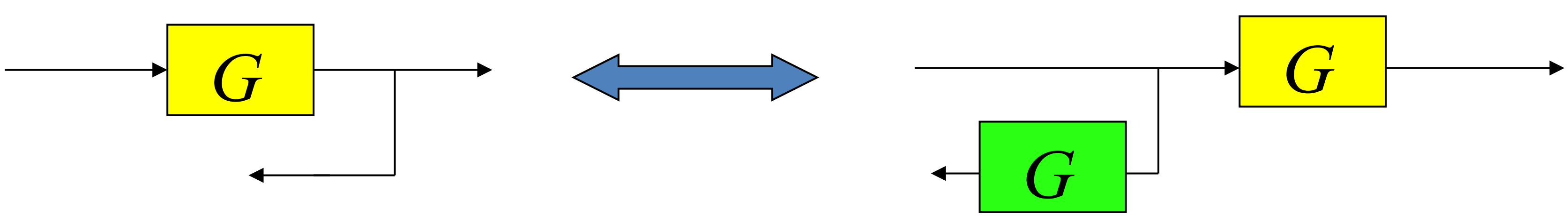
3. Moving a summing point ahead of a block



4. Moving a pickoff point from behind a block

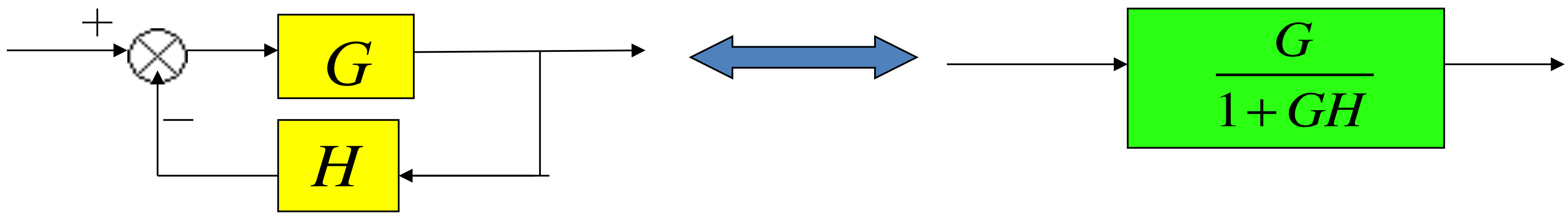


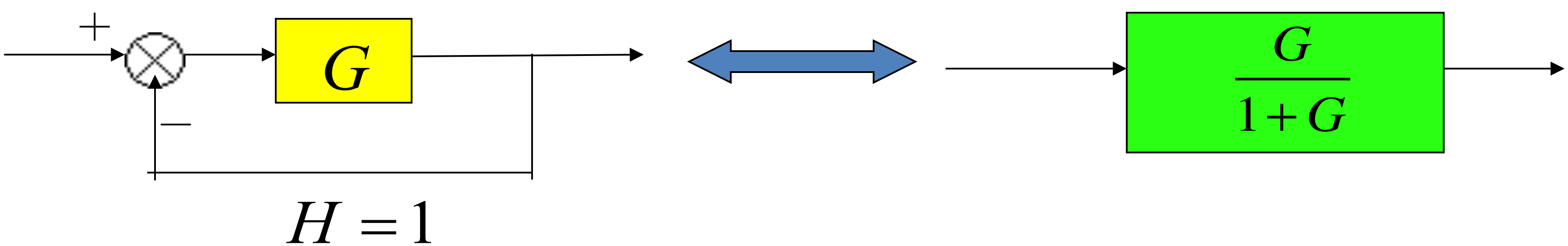
5. Moving a pickoff point from ahead of a block



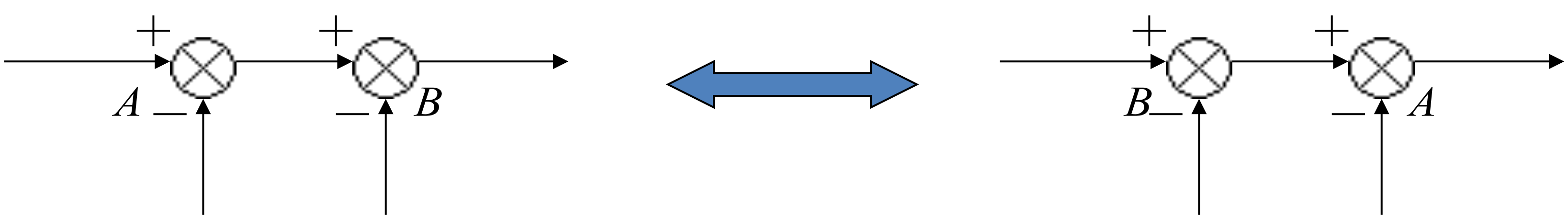
Block diagram reduction summary

6. Eliminating a feedback loop





7. Swap order of two neighboring summing points



ROCO218: Control Engineering
Dr Ian Howard

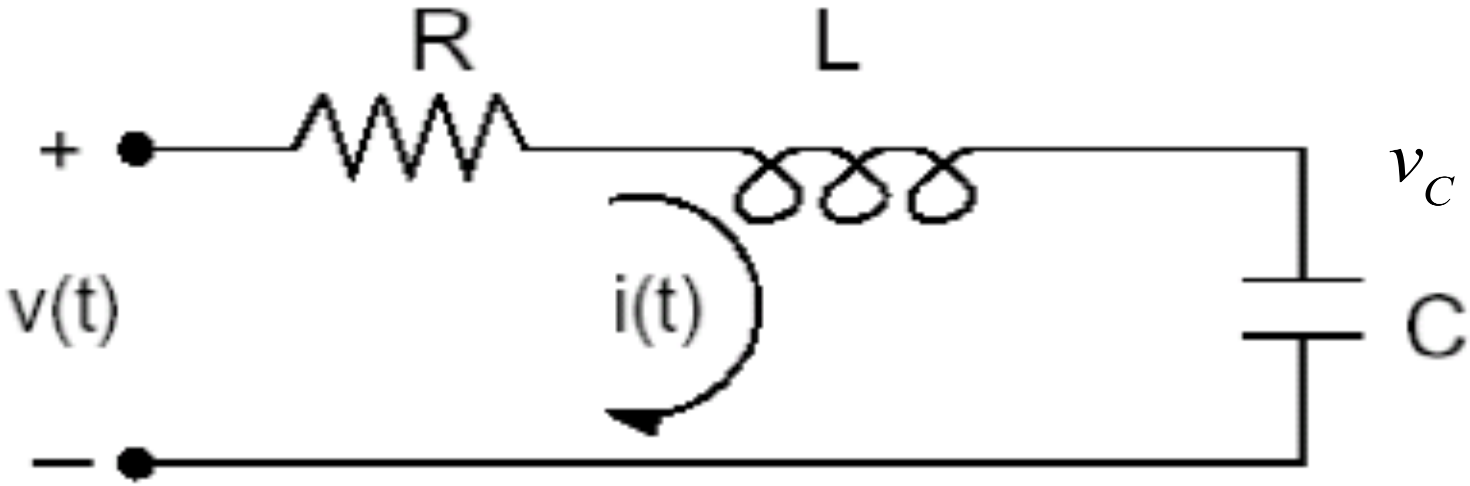
Lecture 4

Transfer function RLC example

Series RLC circuit

$$v_R = iR$$

$$v_L = L\frac{di}{dt}$$



$$v_C = \frac{1}{C}\int i(t).dt$$

Adding up the contributions to overall applied voltage

$$v(t) = iR + L\frac{di}{dt} + \frac{1}{C}\int i(t).dt$$

Series RLC circuit in the s-domain

From differential equation of voltage

$$v(t) = iR + L\frac{di}{dt} + \frac{1}{C}\int i(t).dt$$

Taking Laplace transforms

$$V(s) = RI(s) + LsI(s) + \frac{1}{sC}I(s) \quad = I(s)(R + Ls + \frac{1}{sC})$$

Similarly for capacitor voltage

$$v_c(t) = \frac{1}{C}\int i(t).dt \qquad \Rightarrow V_c(s) = \frac{1}{sC}I(s)$$

Series RLC circuit in the s-domain

Transfer function is given by

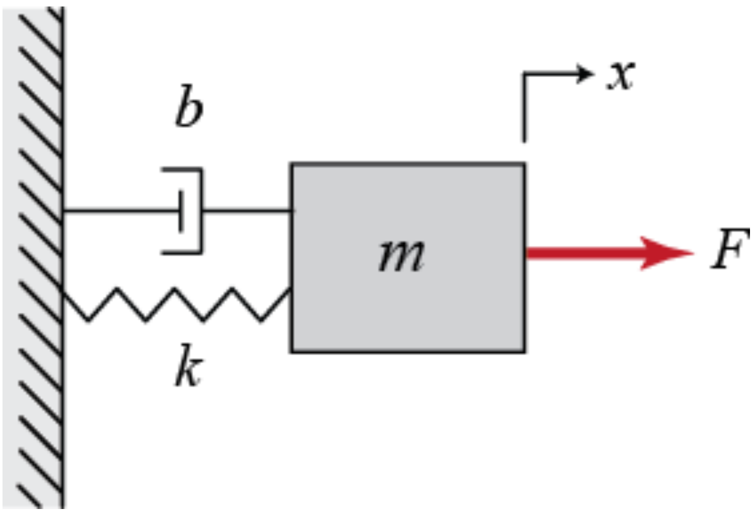
$$\frac{V_c(s)}{V(s)} = \frac{\frac{1}{sC} I(s)}{I(s)(R + Ls + \frac{1}{sC})}$$

Cancelling current term

$$= \frac{\frac{1}{sC}}{R + Ls + \frac{1}{sC}} \qquad = \frac{\frac{1}{LC}}{s^2 + s\frac{R}{L} + \frac{1}{LC}}$$

Analysis of a simple mechanical system

- Consider the mass-spring-damper system shown below:



Use these parameters
M = 0.5 kg
b = 5 N s/m
k = 10 N/m
F = 1 N

- Describe using differential equation of motion
- Balancing the forces gives:

$$F(t) = m\frac{d^2x(t)}{dt^2} + b\frac{dx(t)}{dt} + kx(t)$$

Analysis of a simple mechanical system

So given

$$F(t) = m\frac{d^2x(t)}{dt^2} + b\frac{dx(t)}{dt} + kx(t)$$

Remember in the general case the Laplace transform of differentials include initial conditions

$$\mathfrak{L}\{y'\} = sY(s) - y(0)$$
$$\mathfrak{L}\{y''\} = s^2Y(s) - sy(0) - y'(0)$$

The Laplace transform gives

$$F(s) = m\left[s^2X(s) - sx(0) - \frac{dx}{dt}(0)\right] + b\left[sX(s) - x(0)\right] + kX(s)$$

Analysis of a simple mechanical system

So given

$$F(s) = m\left[s^2 X(s) - sx(0) - \frac{dx}{dt}(0)\right] + b\left[sX(s) - x(0)\right] + kX(s)$$

Setting initial conditions to zero

$$F(s) = ms^2 X(s) + bsX(s) + kX(s) \quad = X(s)\left(ms^2 + bs + k\right)$$

$$\Rightarrow \frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

Use these parameters
M = 0.5 kg
b = 5 N s/m
k = 10 N/m
F = 1 N

$$H(s) = \frac{1}{0.5s^2 + 5s + 10}$$

ROCO218: Control Engineering
Dr Ian Howard

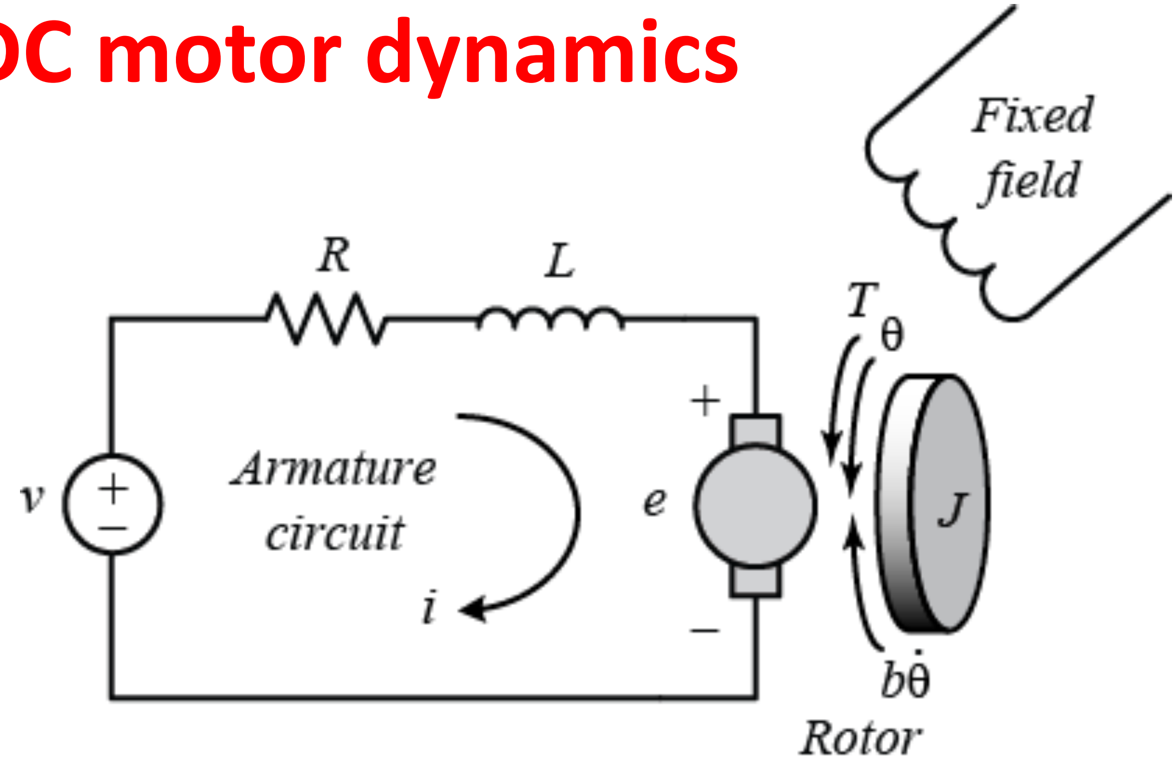
Lecture 4
Modelling a DC motor with transfer function

Revision: DC motor dynamics

Motor torque T_m is given by the current multiplied by the torque constant of the motor

$$T_m = K_t i(t)$$

Mechanical resisting torque T_r is given by inertia and viscous friction

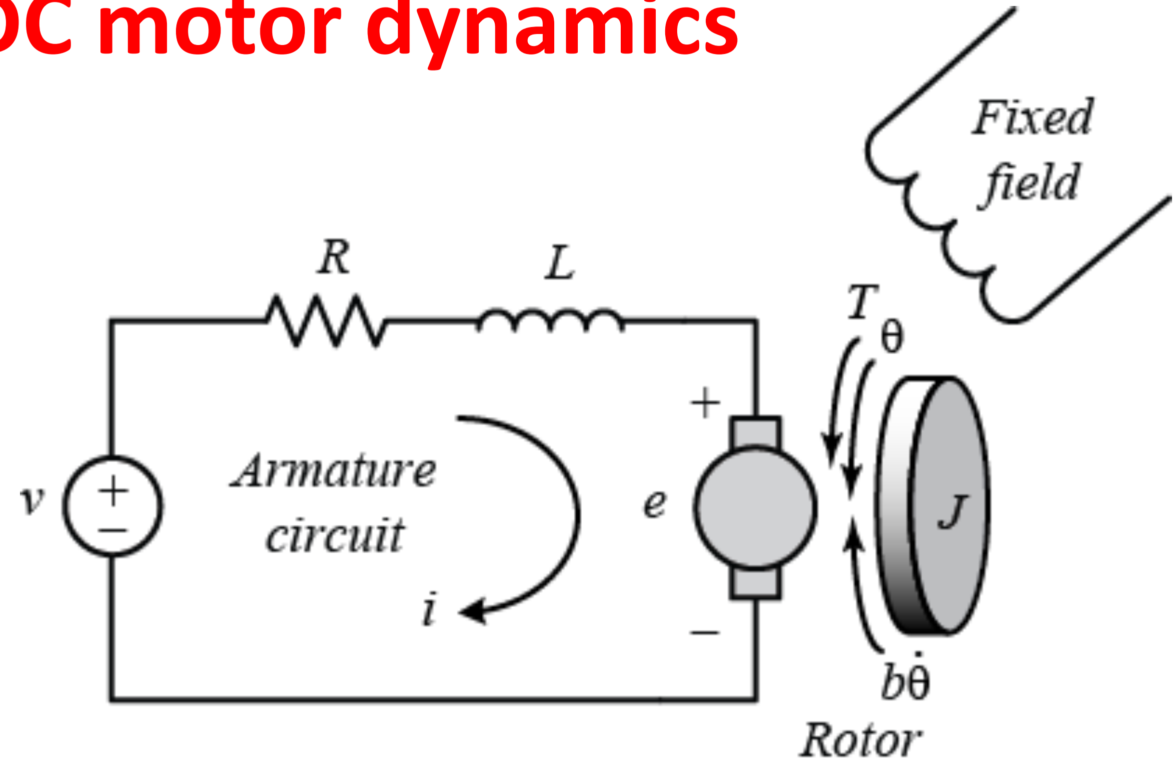


$$T_r = b\frac{d\theta}{dt} + J\frac{d^2\theta}{dt^2}$$

Therefore equating the two terms gives torque equation

$$K_t i(t) = b\frac{d\theta}{dt} + J\frac{d^2\theta}{dt^2}$$

Revision: DC motor dynamics



Summing voltages around the circuit gives voltage equation

$$v(t) = i(t)R + L\frac{di}{dt} + K_e\frac{d\theta}{dt}$$

$$\Rightarrow L\frac{di}{dt} = -K_e\frac{d\theta}{dt} - i(t)R + v(t)$$

$$\Rightarrow \frac{di}{dt} = -\frac{K_e}{L}\frac{d\theta}{dt} - \frac{R}{L}i(t) + \frac{1}{L}v(t)$$

Motor solution using Laplace transformations

Taking Laplace transforms of the differential equations that describe the motor mechanical dynamics:

$$K_t i(t) = b\frac{d\theta}{dt} + J\frac{d^2\theta}{dt^2}$$

Becomes

$$K_t I(s) = bs\Theta(s) + Js^2\Theta(s) \quad = s(b + Js)\Theta(s)$$

$$\Rightarrow I(s) = \frac{s(b + Js)\Theta(s)}{K_t}$$

Motor solution using Laplace transformations

Taking Laplace transforms of the differential equations that describe the motor voltages:

$$v(t) = i(t)R + L\frac{di}{dt} + K_e\frac{d\theta}{dt}$$

Becomes

$$V(s) = I(s)R + LsI(s) + K_e s\Theta(s)$$

$$\Rightarrow V(s) = I(s)\left[R + Ls\right] + K_e s\Theta(s)$$

Motor solution using Laplace transformations

Substituting

$$I(s)=\frac{s(b+Js)\Theta(s)}{K_t}$$ Into $$V(s)=I(s)\left[R+Ls\right]+K_e s\Theta(s)$$

and thereby laminating current I(s)

$$\Rightarrow V(s)=\frac{s(b+Js)\Theta(s)}{K_t}\left[R+Ls\right]+K_e s\Theta(s)$$

$$\Rightarrow V(s)=\Theta(s)\left(\frac{s(b+Js)\left[R+Ls\right]}{K_t}+K_e s\right)$$

Motor solution using Laplace transformations

So from

$$V(s) = \Theta(s)\left(\frac{s(b+Js)[R+Ls]}{K_t} + K_e s\right)$$

setting $K_t = K_e = K$ gives

$$V(s) = \Theta(s)\left(\frac{s(b+Js)[R+Ls]}{K} + Ks\right)$$

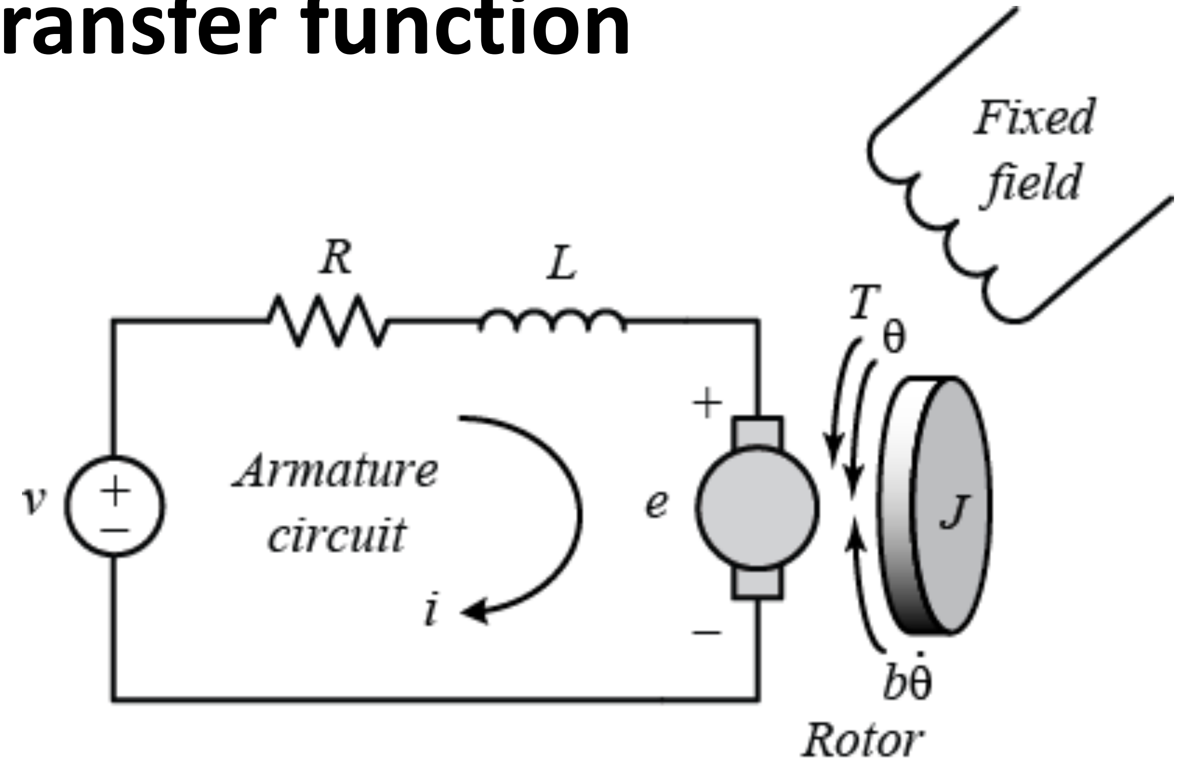
factoring out 1/K

$$V(s) = \frac{\Theta(s)}{K}\left(s(b+Js)[R+Ls] + K^2 s\right)$$

$$\Rightarrow \frac{\Theta(s)}{V(s)} = \frac{K}{s\left[(Js+b)(Ls+R)+K^2\right]}$$

Motor transfer function

Result of a transfer response output position for a DC electric motor given its input voltage



$$\frac{\Theta(s)}{V(s)} = \frac{K}{s\left[(Js+b)(Ls+R)+K^2\right]}$$

Can differentiate this expression to get the transfer function for speed by multiplying by the transfer function of a differentiator (=s)

$$\frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{\left[(Js+b)(Ls+R)+K^2\right]}$$

Simulink model simulation transfer function

Expand the expressing into powers of s so can be directly used with Matlab tf function.

$$\frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{\left[(Js+b)(Ls+R)+K^2\right]}$$

Becomes

$$\frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{\left[JLs^2+(JR+bL)s+\left(bR+K^2\right)\right]}$$

Use Matlab to calculate Simulink parameters

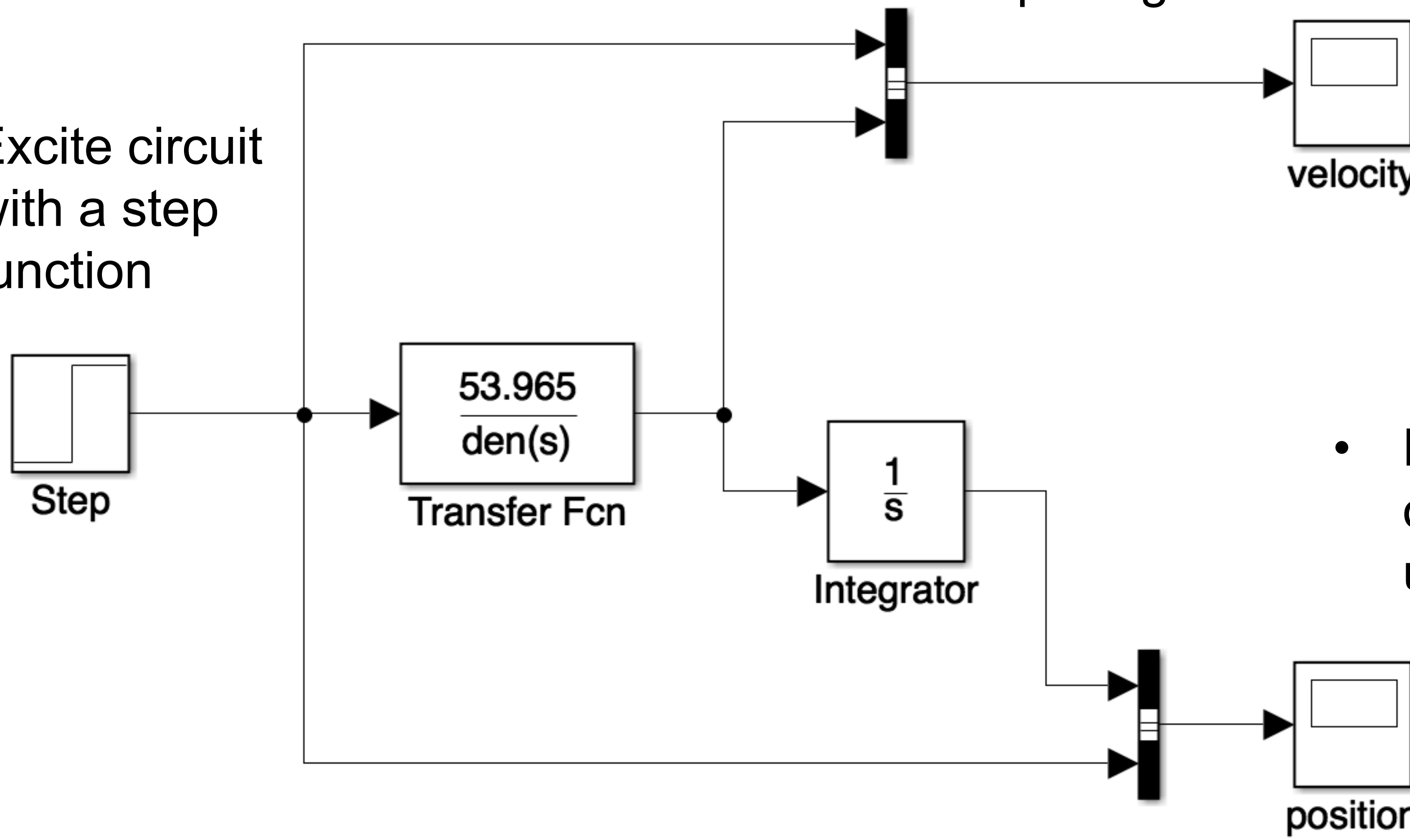
Use real motor parameter values:

$R = 8.152\ \Omega$
$L = 3.252$ mH
$J = 0.04248$ kg.m2
$b = 0$
$K_t = 53.965$ N.m/A
$K_e = 53.965$ V/rad/sec

```
% enter motor kit parameters
R = 8.152;
L = 0.003252;
J = 0.04248;
b = 0;
K = 53.965;


%  use the tf function to create the object
% num and den are the real- or complex-valued row vectors of numerator and
% denominator coefficients ordered in descending powers of s.
% get gain
G=1;
num = K*G;
den = [(J*L) (J*R + b*L) (b*R + K^2)];
mtrTF = tf(num, den);
disp('num');
disp(num(1));

disp('den');
disp(den(1));
disp(den(2));
disp(den(3));
```

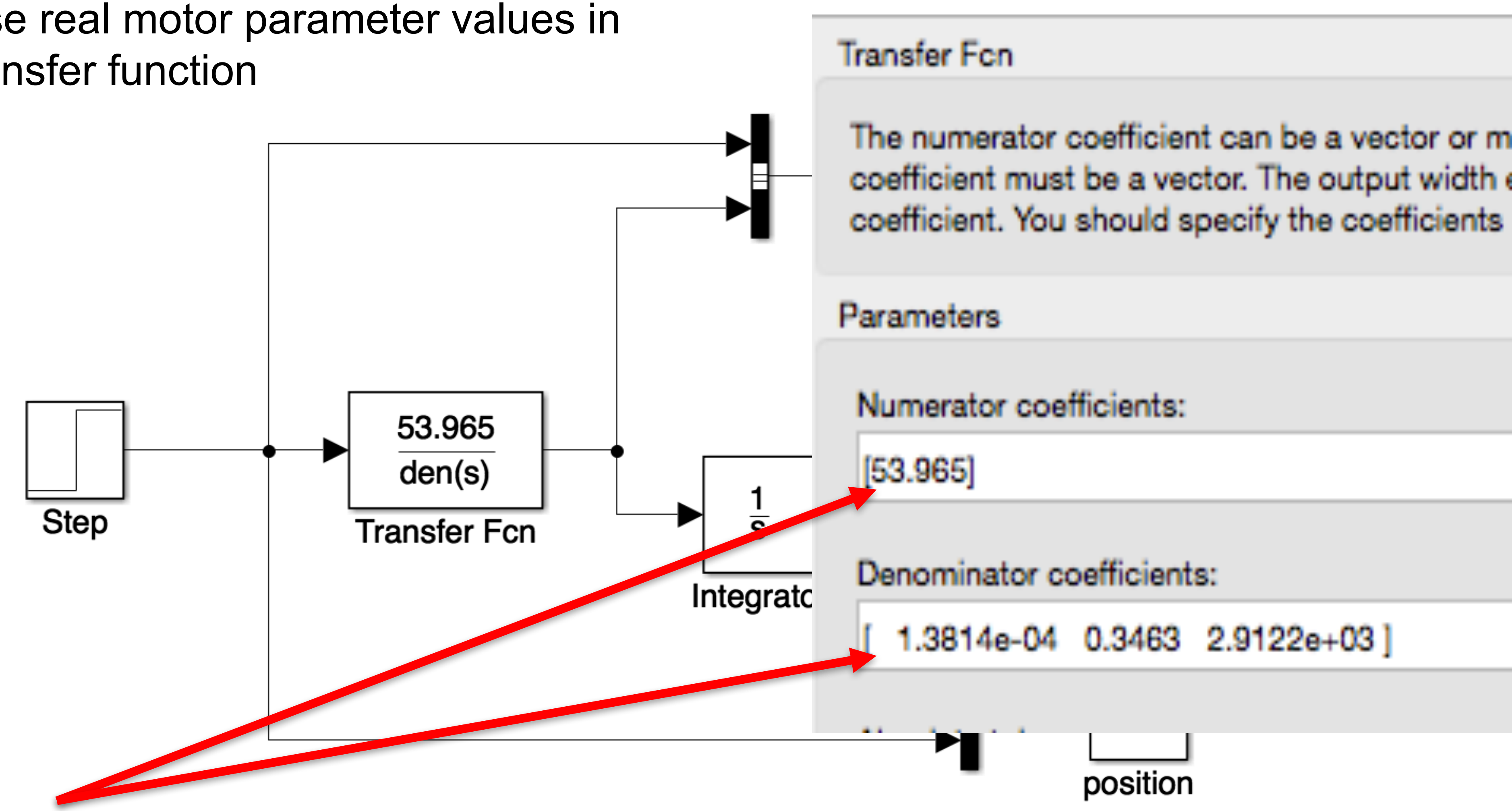
Building the Simulink model simulation

- Use multiplexers to joint step and output signals
- Excite circuit with a step function
- Display outputs using scopes



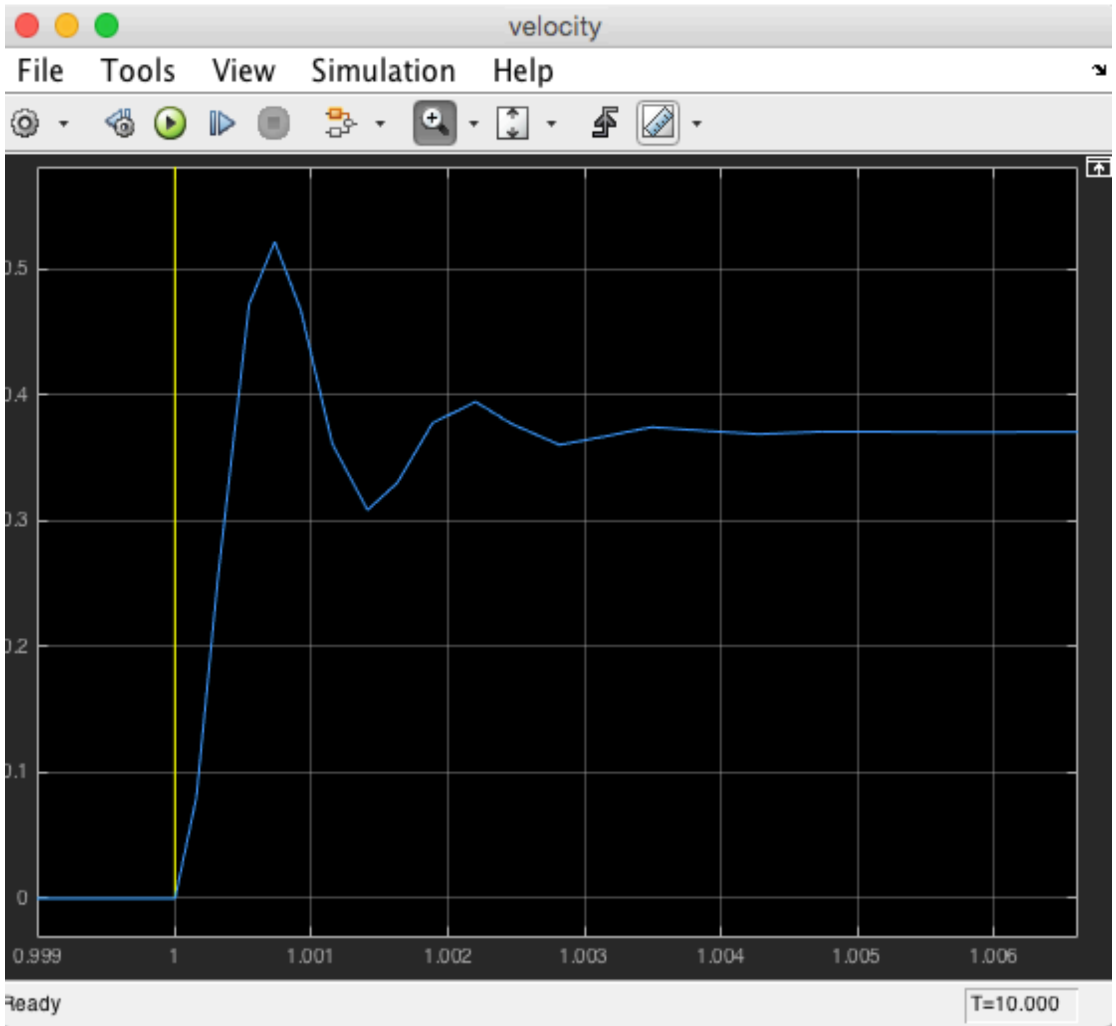
Enter transfer function parameters

- Use real motor parameter values in transfer function



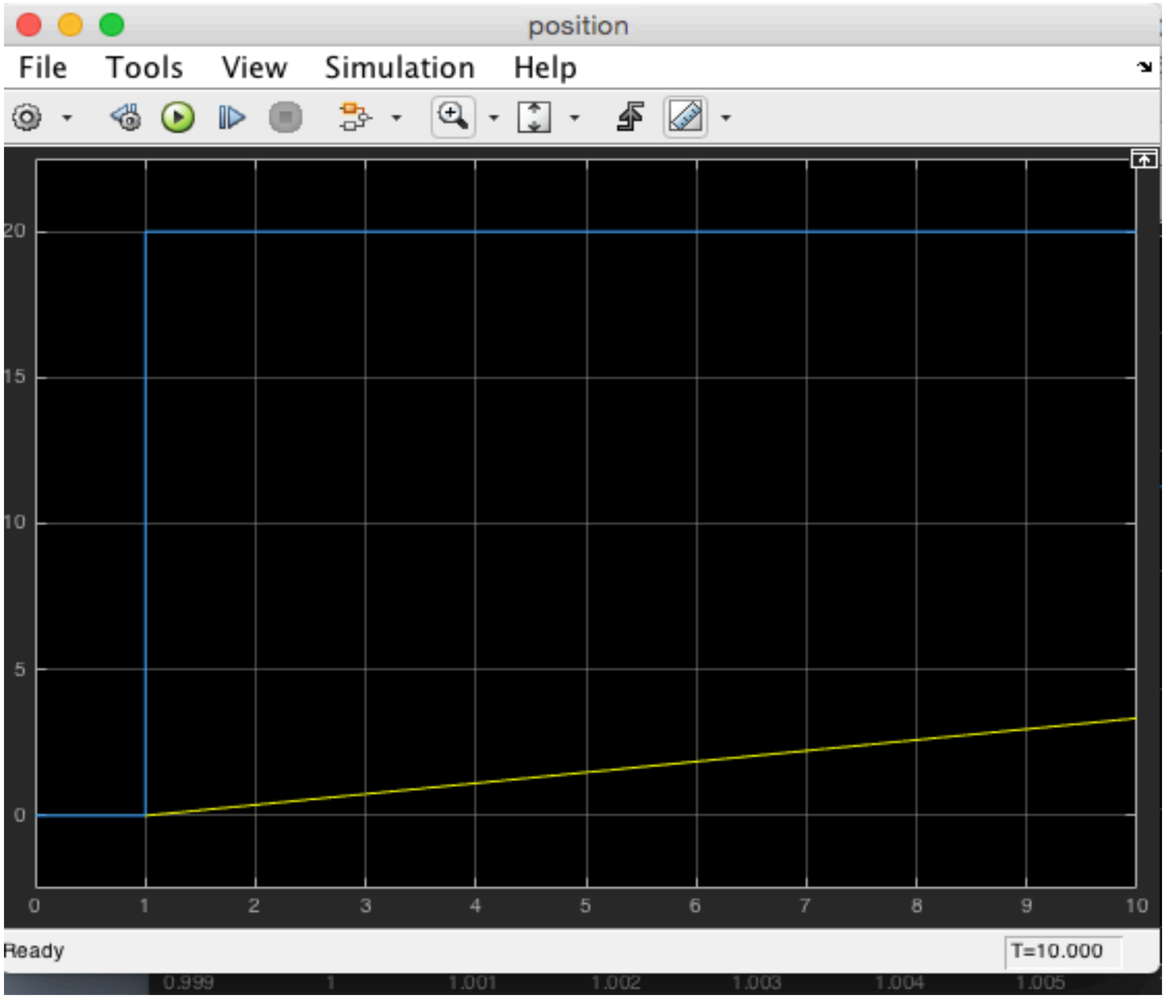
- Better to use Matlab to compute values in workspace and refer to them here is Simulink
- Makes it easy to change values without having to update numerical values

Examine angular speed plot



The speed plot when zoomed in will look like this.
Angular velocity settles time is about 4ms

Examine angular position plot



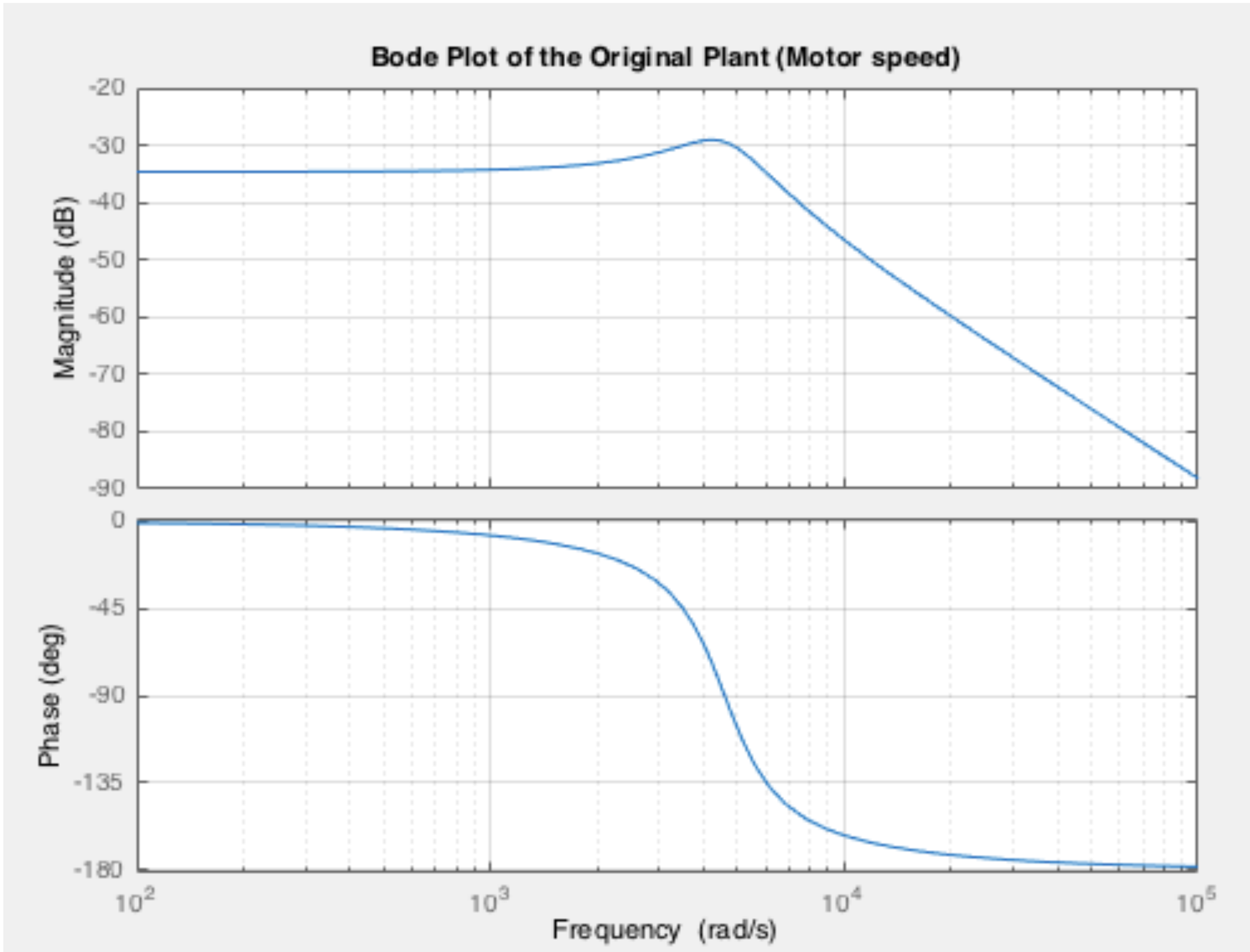
Shows linear increase in angular position when step function activated

Bode plots of the transfer function

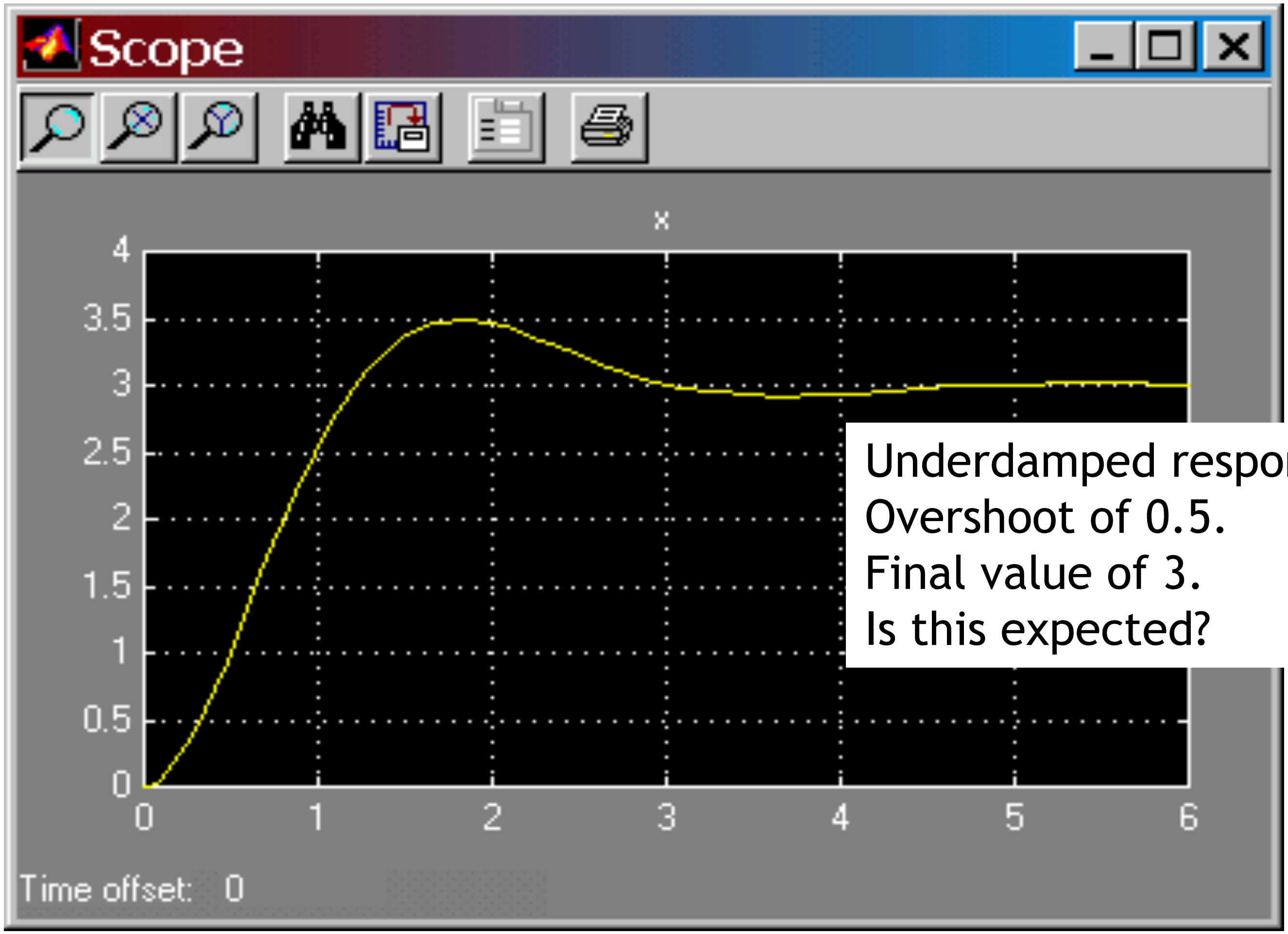
- It is easy to generate a Bode plot of transfer functions.
- We can use the Matlab bode function
- This plots the amplitude and phase response of the transfer function

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% generate Bode plot
figure
hold on
bode(mtrTF);
grid
title('Bode Plot of the Original Plant (Motor speed)')
```

Bode plots of the transfer function



Output results



Underdamped response.
Overshoot of 0.5.
Final value of 3.
Is this expected?

Interlude

10 minute break

ROCO218: Control Engineering
Dr Ian Howard

Lecture 4

Poles and zeros

Poles of a transfer function

- The poles of a transfer function are the values of s that make the function evaluate to infinity.
- The value of s can be visualized on the s-plane

- The poles are therefore the roots of the denominator polynomial
- For example: the expression

$$H(s) = \frac{10(s+2)}{(s+1)(s+3)}$$

- Has a pole at s = -1
- And a pole at s = -3

- Complex poles always appear in complex-conjugate pairs
- The transient response of system is determined by the location of poles

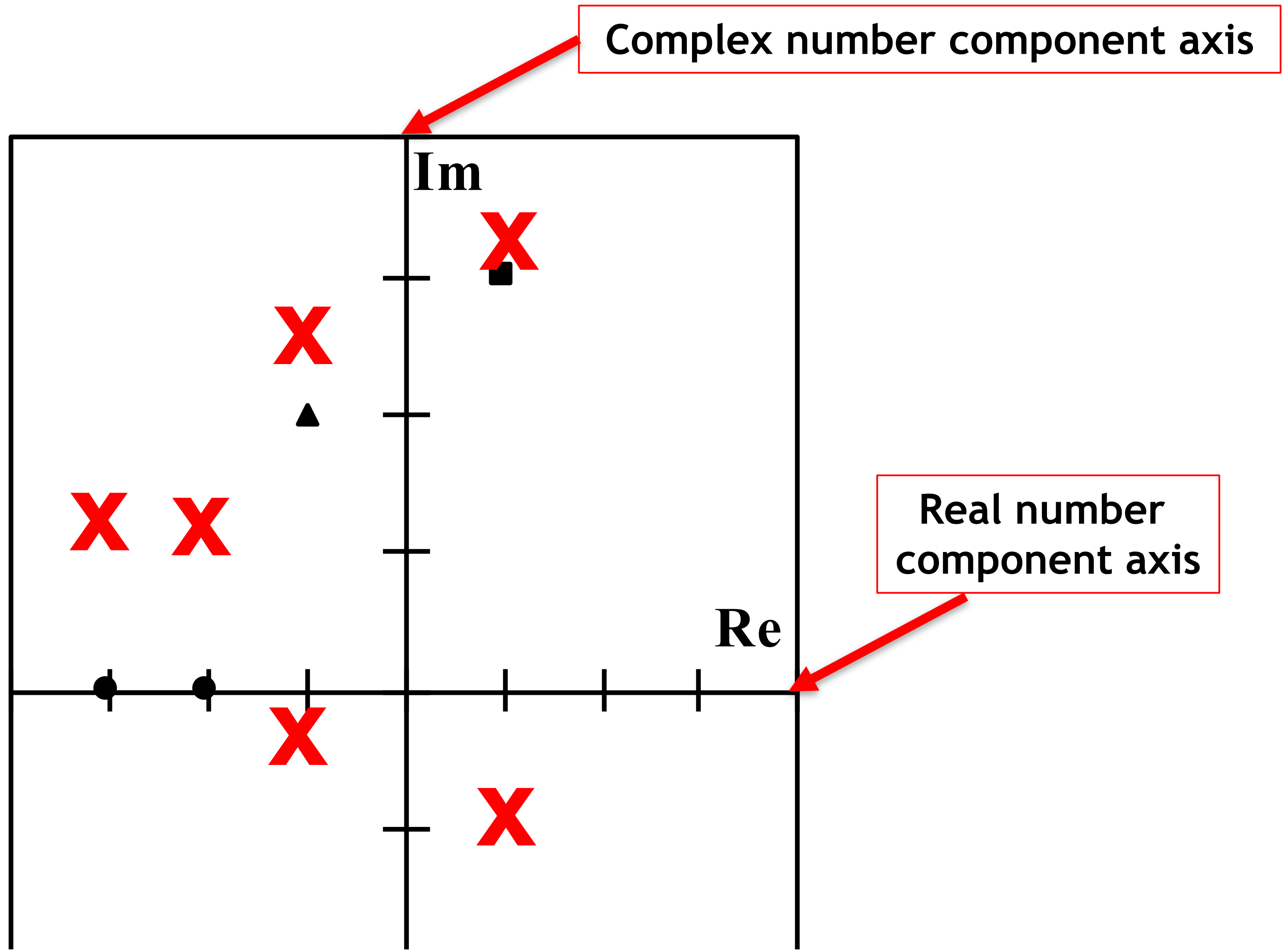
Zeros of a transfer function

- The zeros of a transfer function are the values of s that make the transfer function evaluate to zero.
- They are therefore the zeros of the numerator polynomial
- For example: The expression

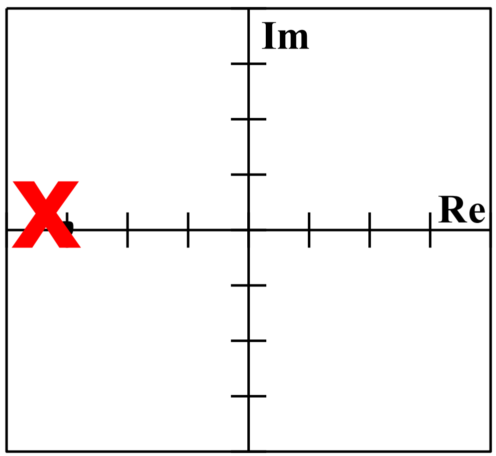
$$H(s) = \frac{10(s+2)}{(s+1)(s+3)}$$

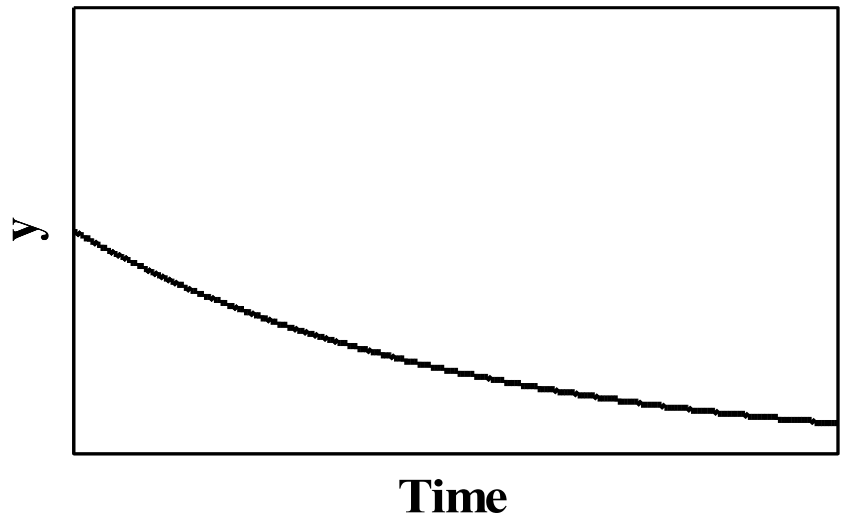
- has a zero at s = -2
- Complex zeros always appear in complex-conjugate pairs

Plotting poles on the complex plane



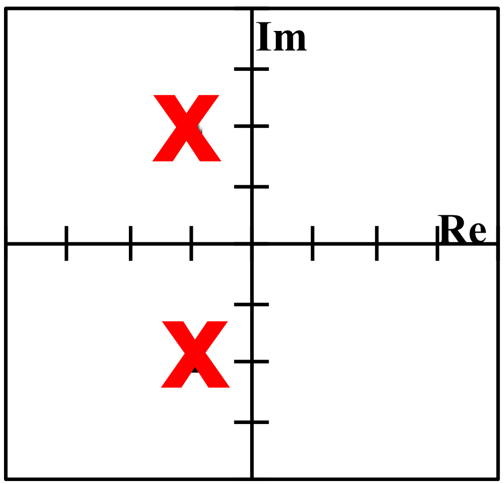
Exponential decay

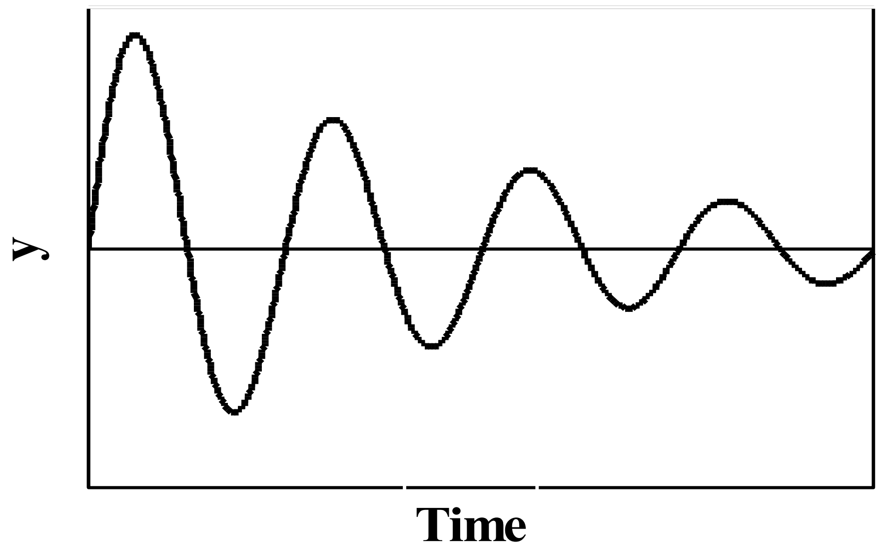




Effect of a -ve real pole
System exhibits exponential decay
The signal will eventually decay to a constant value of zero
Therefore the system is stable

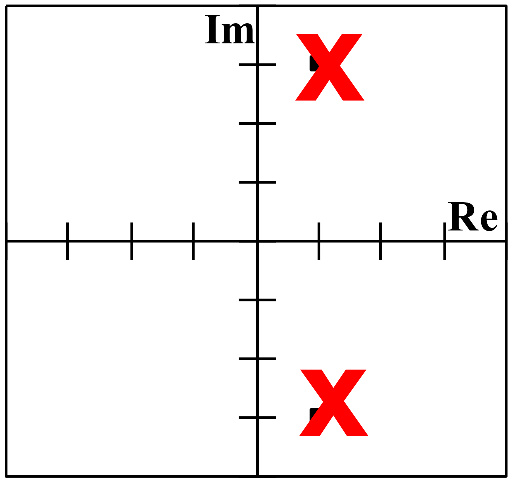
Damped sinusoid

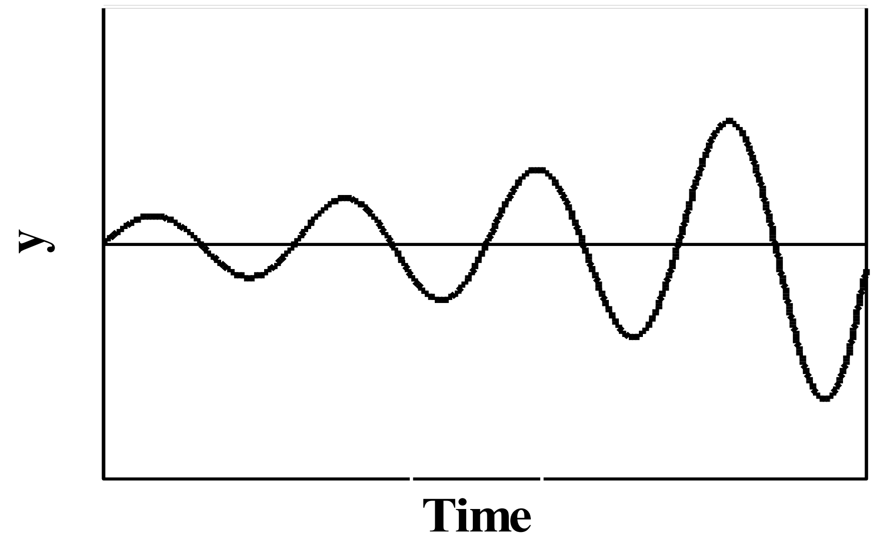




Effect of a -ve real part complex pole
System exhibits exponential damped sinusoidal decay
The signal will eventually decay to a constant value of zero
Therefore the system is stable

Exponentially growing sinusoid (unstable)





Effect of a +ve real part complex pole
System exhibits exponentially sinusoidal growth
The signal will eventually go to infinity (in a mathematical model)
Therefore the system is unstable

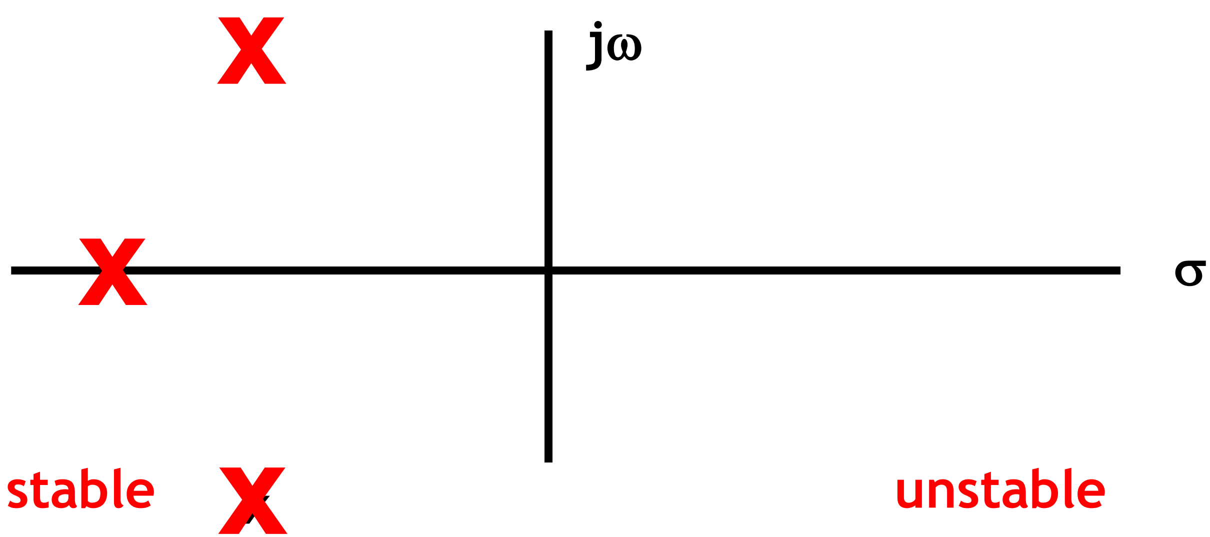
Stable behavior

A system is stable if bounded inputs produce bounded outputs

The complex s-plane is divided into two regions:

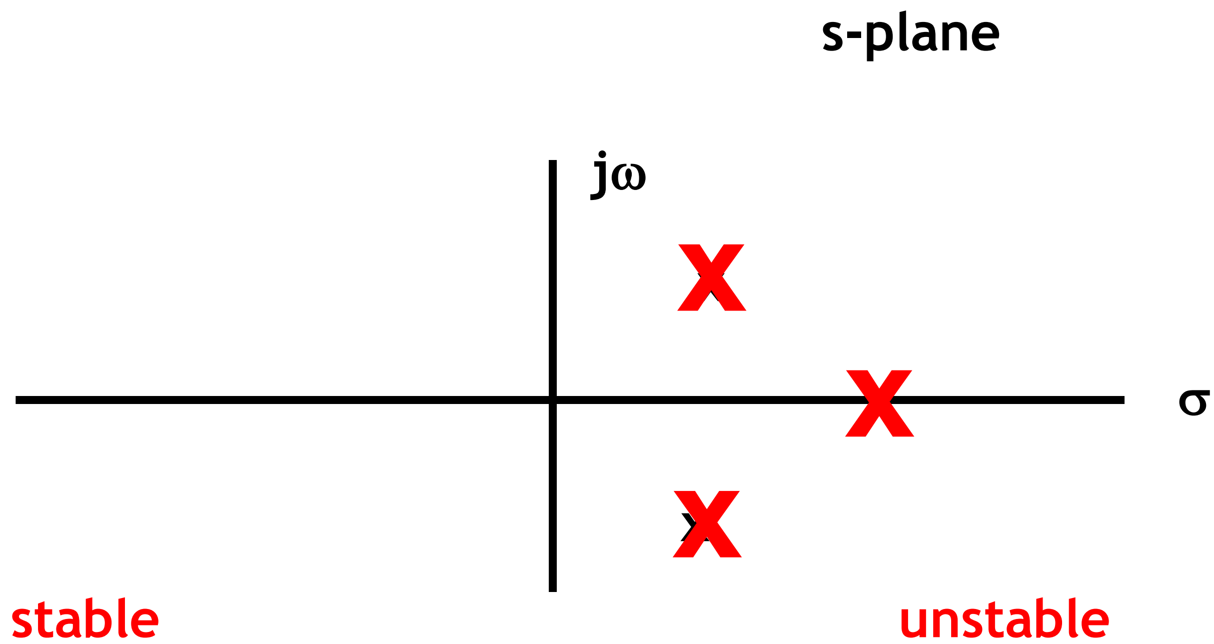
- Stable region, which is on the left half of the plane
- Unstable region, which is on the right half of the s-plane

s-plane



Unstable behavior

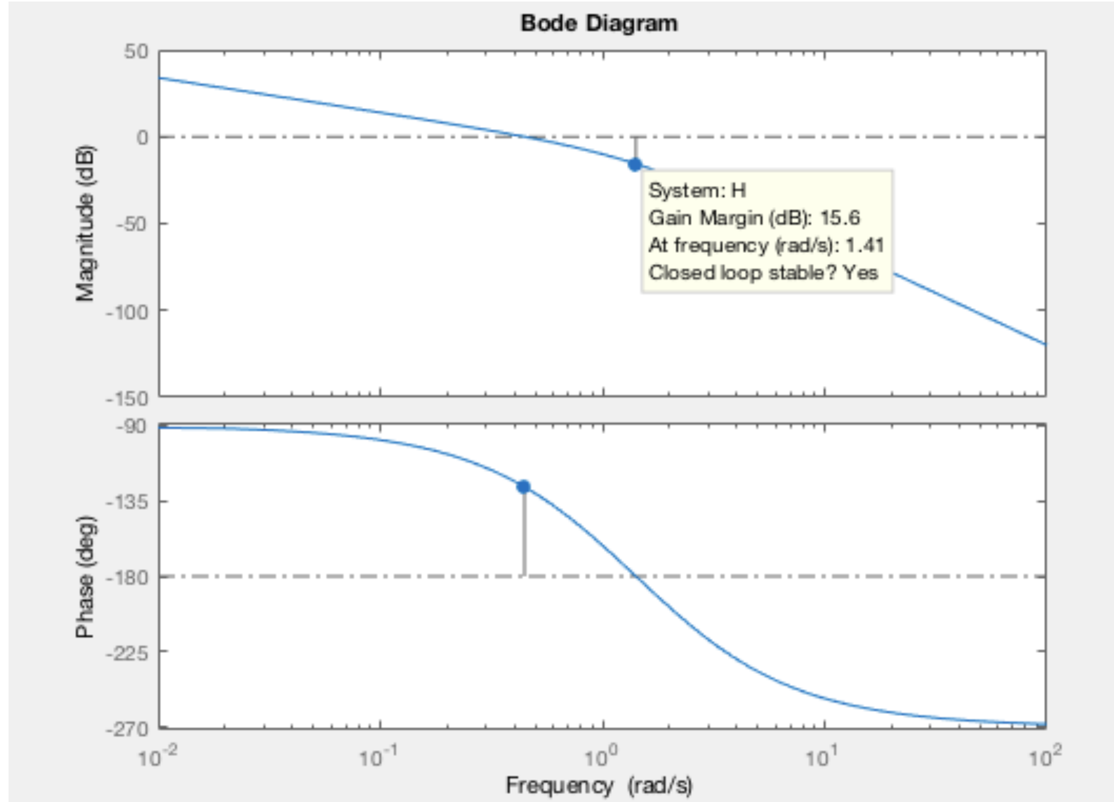
- If the output of a process grows without bound for a bounded input, the process is referred to a **unstable**
- If the real portion of any pole of a transfer function is positive, the process corresponding to the transfer function is unstable
- If any pole is located in the right half plane, the process is unstable.



Stable feedback system

Consider the open loop transfer function

$$H(s) = \frac{K}{s(s+1)(s+2)}$$



When K=1 this has the corresponding Bode plot

We see the gain margin is 15.6dB

This corresponds to a linear gain of 6.0256

Unity feedback losed loop transfer function is given by

$$CL(s) = \frac{\dfrac{K}{s(s+1)(s+2)}}{1 + \dfrac{K}{s(s+1)(s+2)}} = \frac{K}{s(s+1)(s+2)+K}$$

Stable feedback system

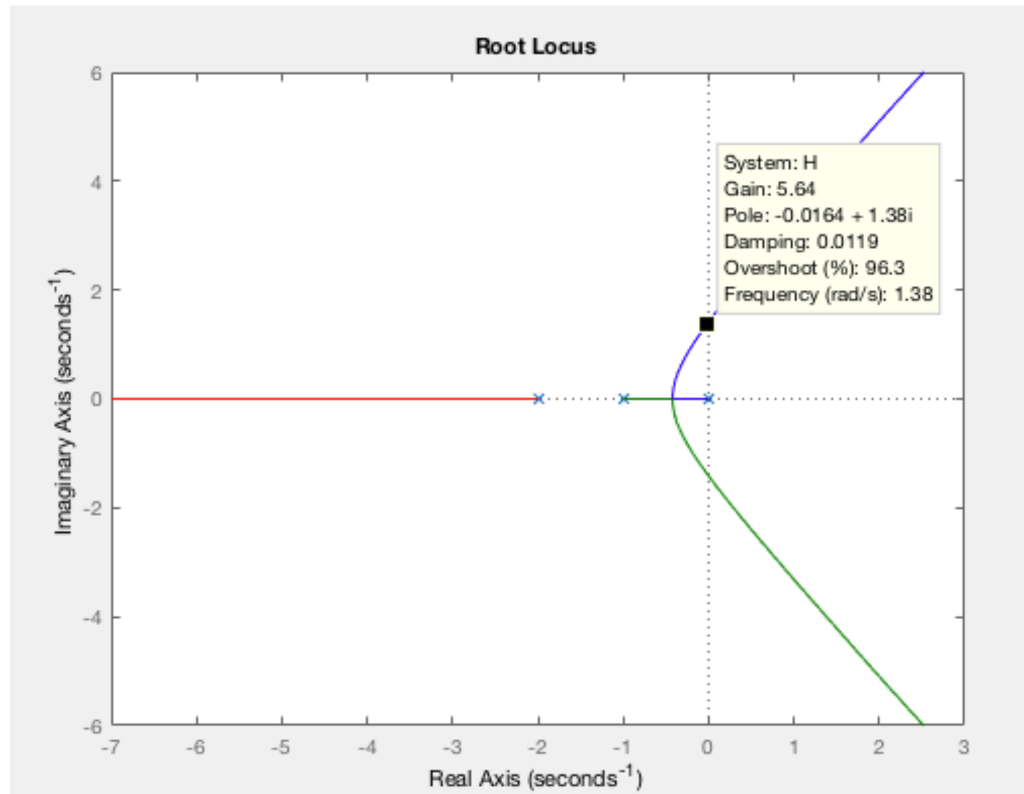
$$CL(s)=\frac{\dfrac{K}{s(s+1)(s+2)}}{1+\dfrac{K}{s(s+1)(s+2)}}=\frac{K}{\boxed{s(s+1)(s+2)+K}}$$

Factor K thus affects pole location

Thus we can change the location of the close loop poles by changing the gain K

If we plot the locations of the poles and zeros on the s-plane as gain K changes, we get what is known as the ”root locus” plot

From the plot we can see when they poles moves to the unstable RHS of of s-plane



Stable feedback system

Below we use a low gain of 3 and the system is stable

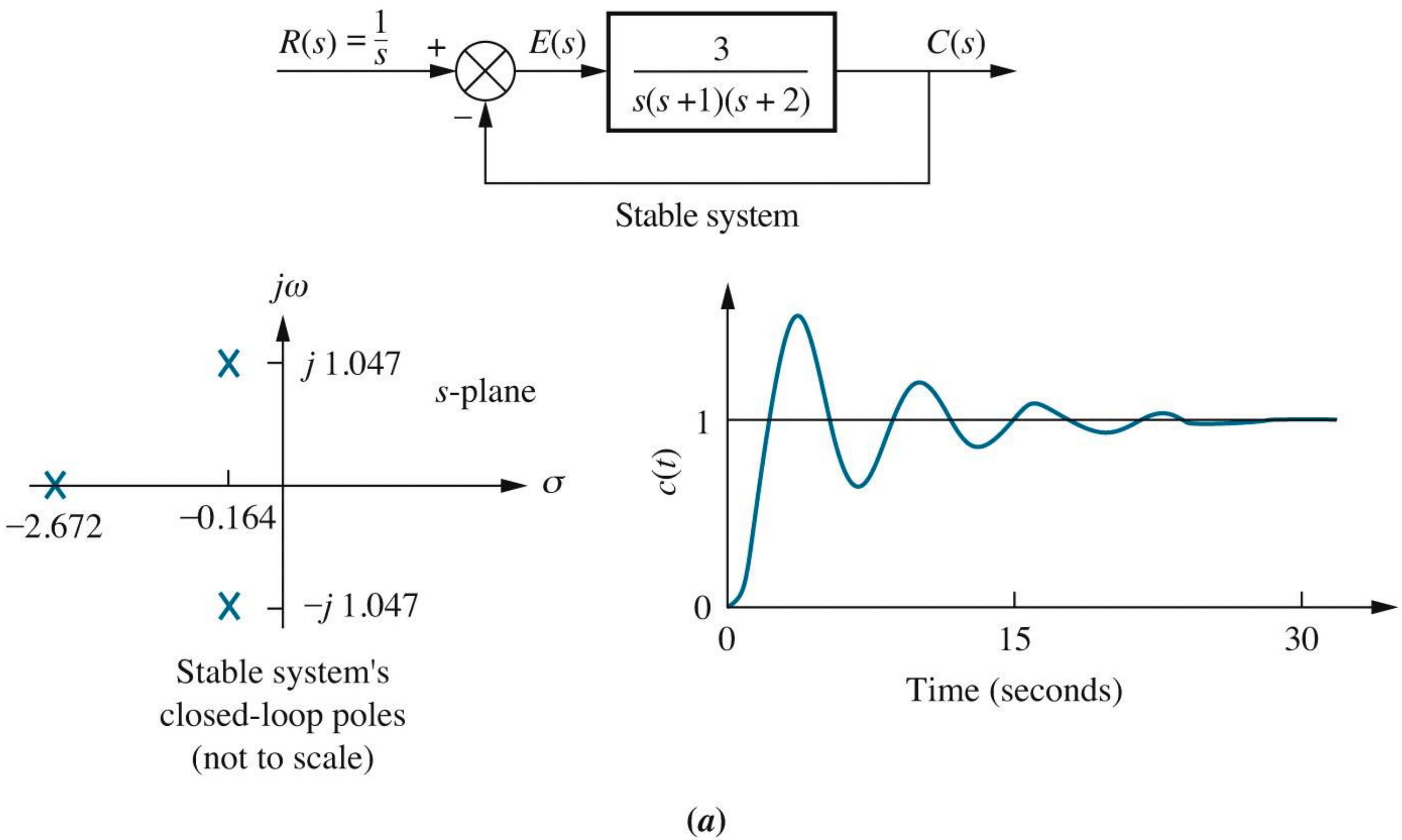


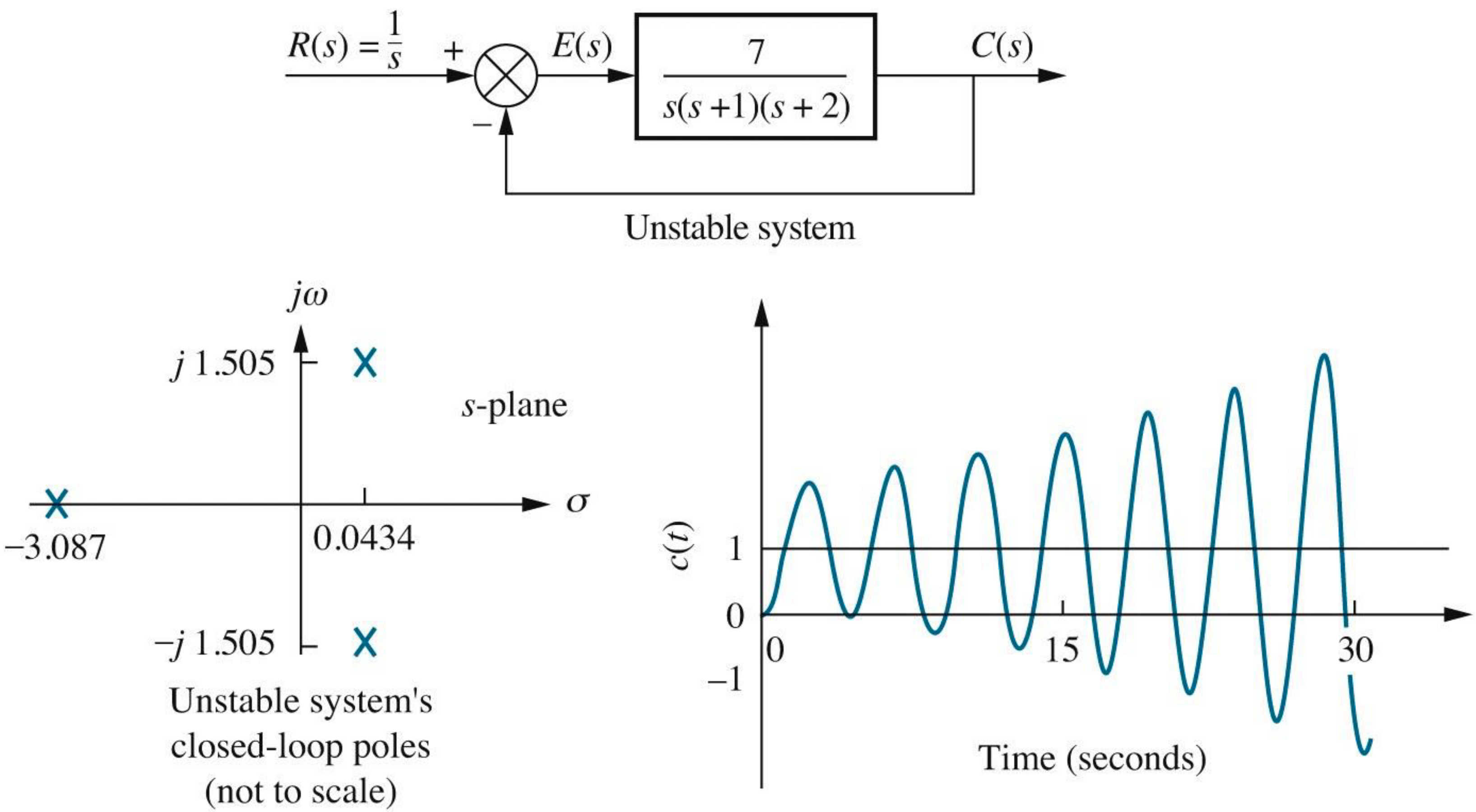
Figure 6.1a

© John Wiley & Sons, Inc. All rights reserved.

Unstable feedback system

However as the gain goes above the gain margin of the system will go unstable

Below we use a higher gain of 7 and the system is stable



(b)

Figure 6.1b

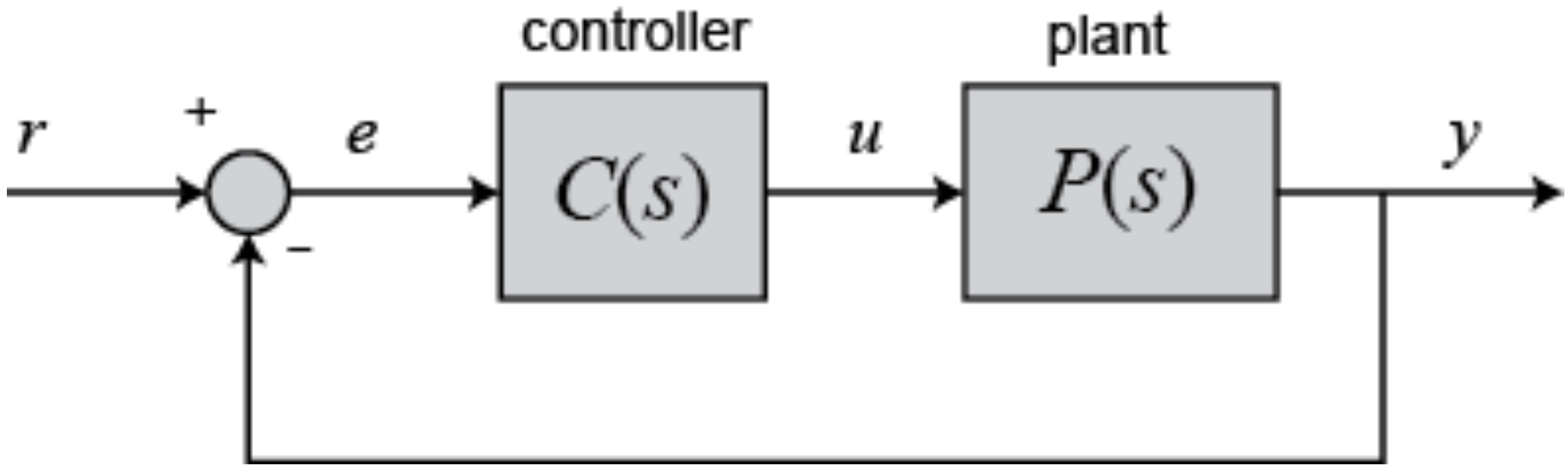
© John Wiley & Sons, Inc. All rights reserved.

ROCO218: Control Engineering
Dr Ian Howard

Lecture 9

PID control

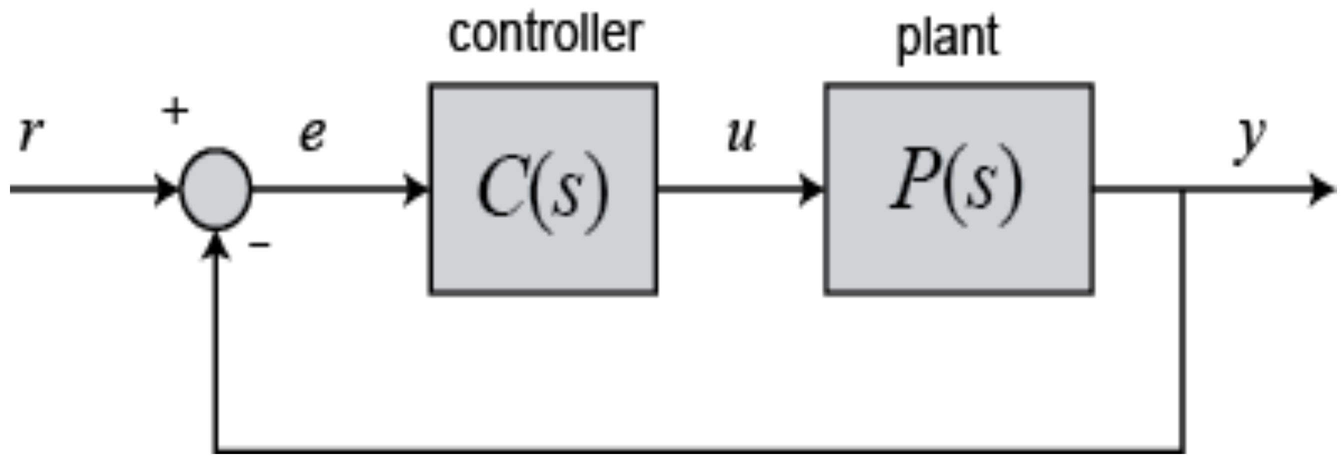
Simple feedback controller



- Plant characteristics are often fixed
- We need to design a controller to achieve required plant performance
- Have to be careful because feedback system can go unstable!
- So what should the controller be?

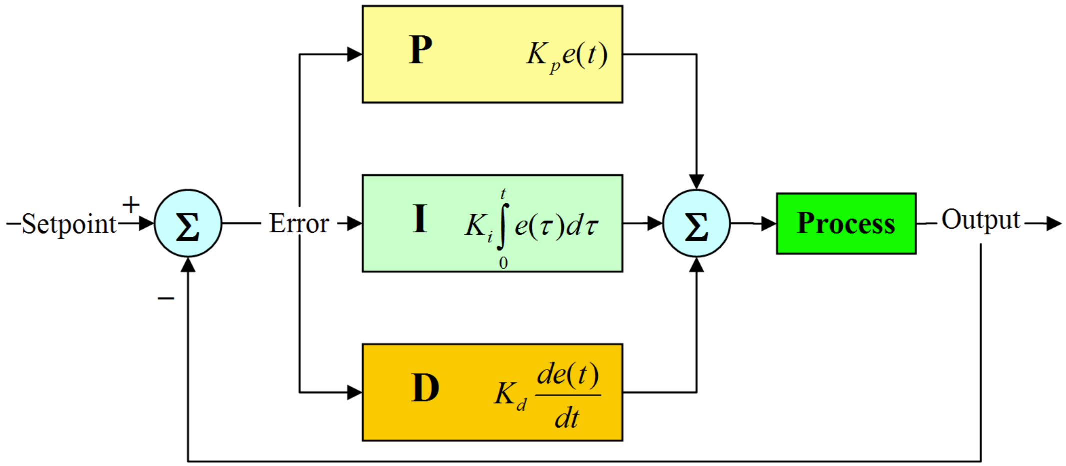
Adding a PID controller

- Place a PID controller C(s) in series with the plant P(s) and make use of negative feedback



- A well-designed control system will ensure
  - We reach a target value quickly
  - Without overshoot
  - With low steady state error.

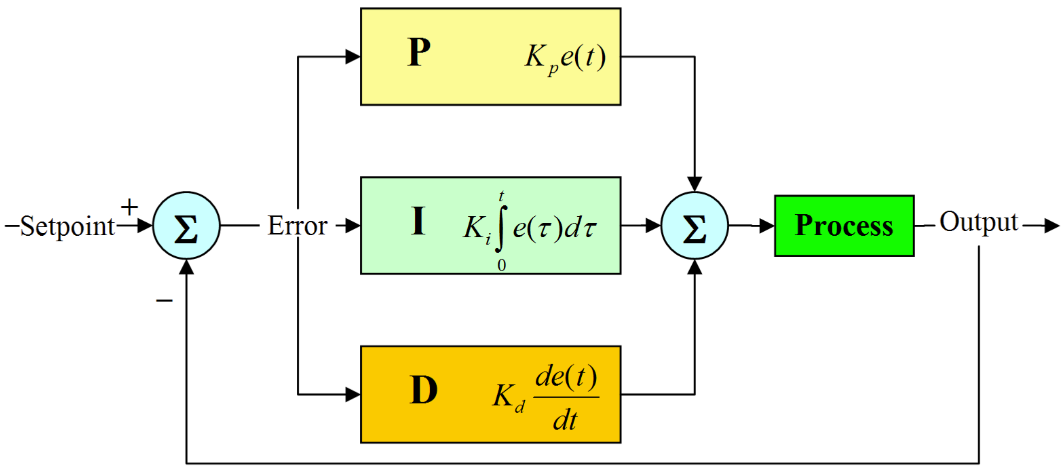
PID parallel pathways

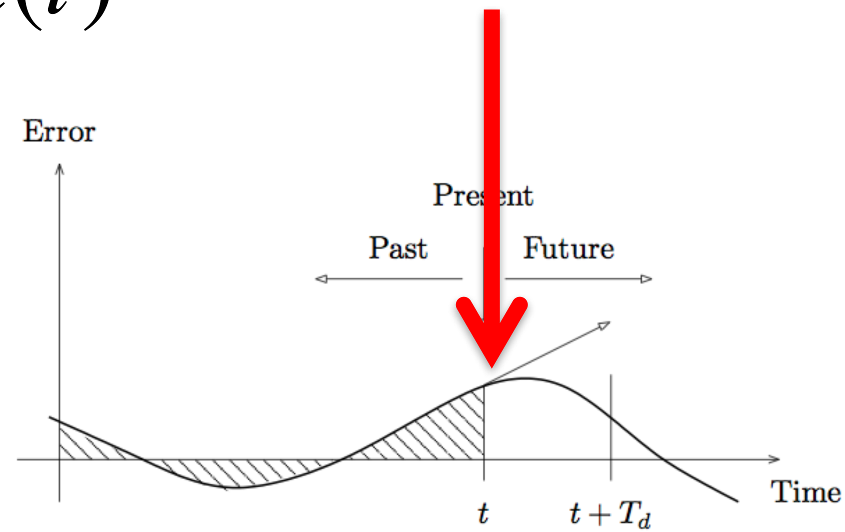


- PID is a mathematical algorithm that the controller uses to compensate for load fluctuations and changes in set point.
- These operations are executed on the error between the set point and the actual value.
- They determine how the plant will react to change
- A PID controller consists of 3 parallel elements that are located in the forward path in a feedback controller scheme located before the plant
- The plant to be controller received the sum of these three processed signals as its input
- The 3 PID elements have different effects

Proportional term

$$term = K_p e(t)$$



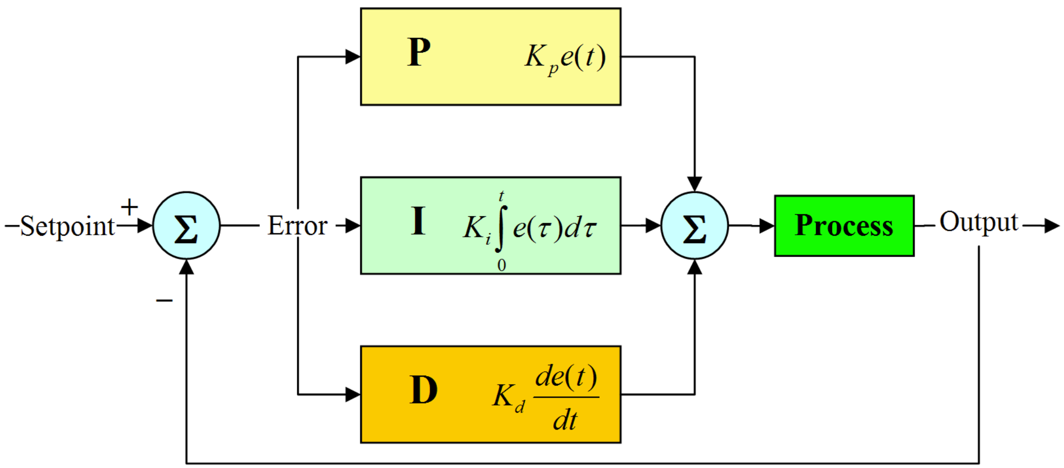


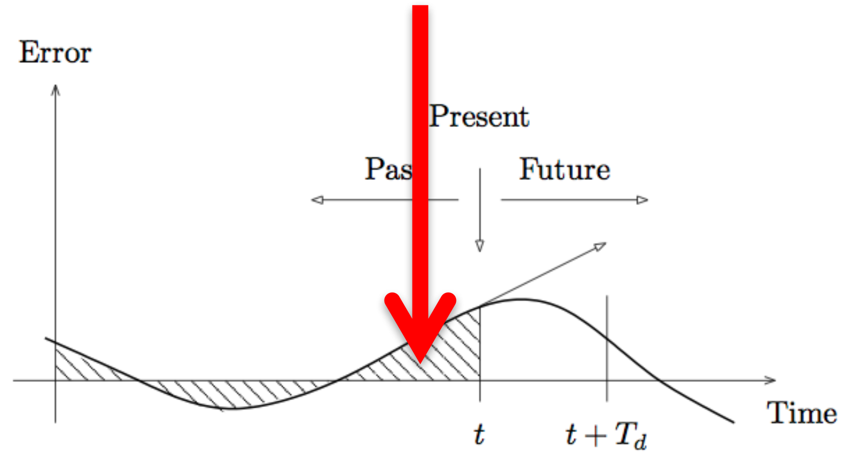
The proportional term directly relates to the error value at the present moment in time

- The proportional gain is given as K_p
- Proportional gain can improve rise time
- If K_p too high the system can become unstable

Integral term

$$term = K_i \int e(t)\,dt$$



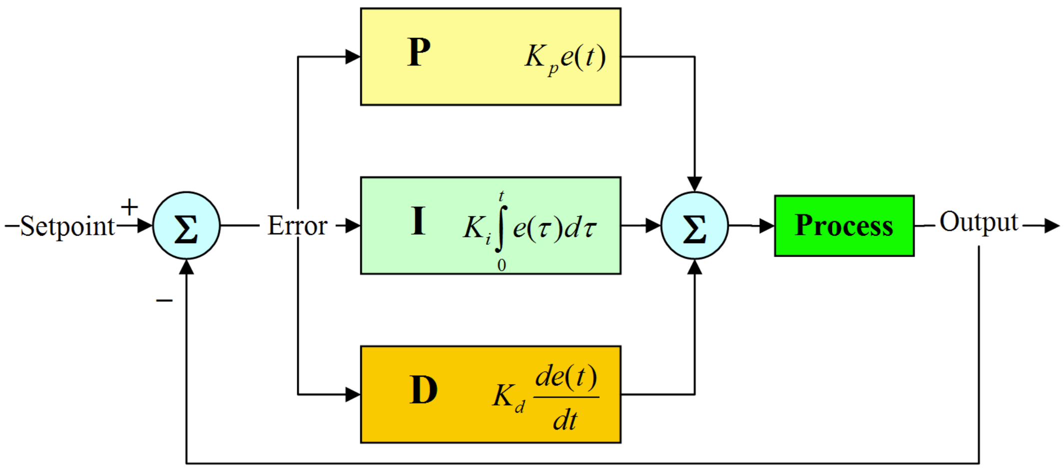


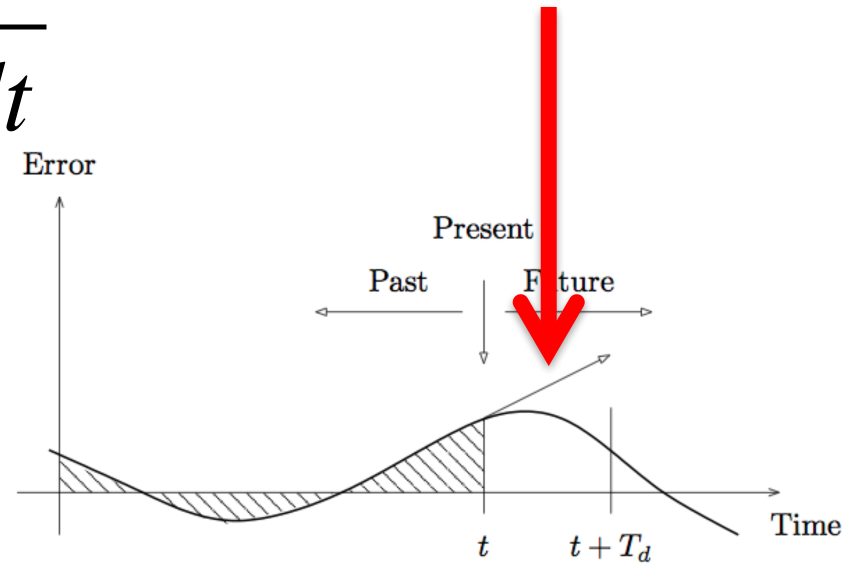
The integral term relates to a past values of the error up to the present point in time
It is proportional to both the magnitude of the past error and its duration

- The integrator gain is given as K_i
- Integral term eliminates residual steady-state error

Derivative term

$$term = K_d \frac{de}{dt}$$

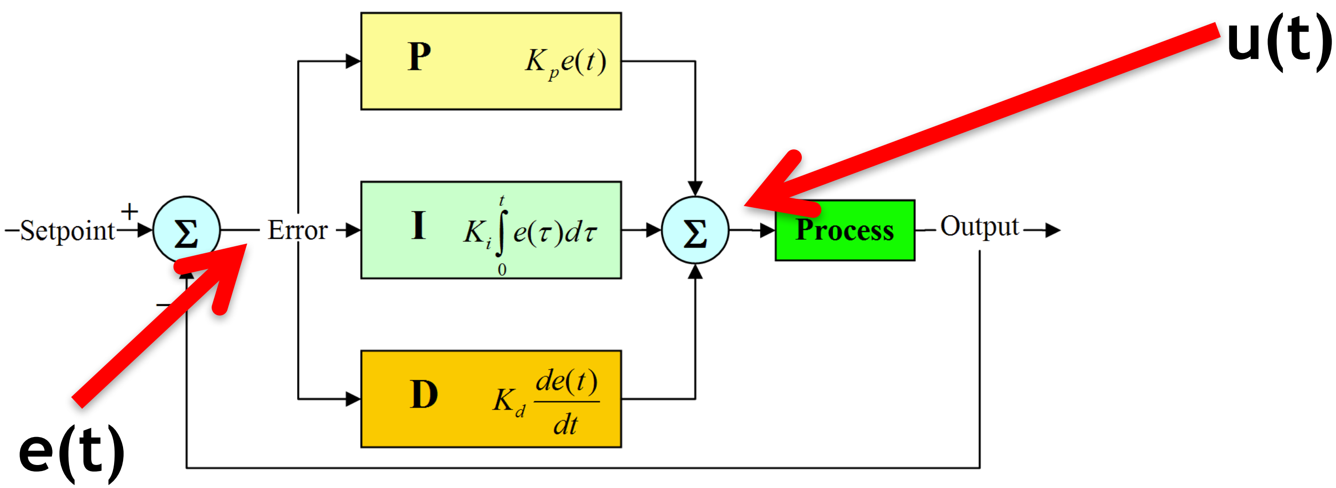




The derivative term that is proportional to the slope of the error over time and relates to a prediction of what the error will be like in the future

- The differentiator gain is given as K_d
- Derivative term improves settling time and stability of the system

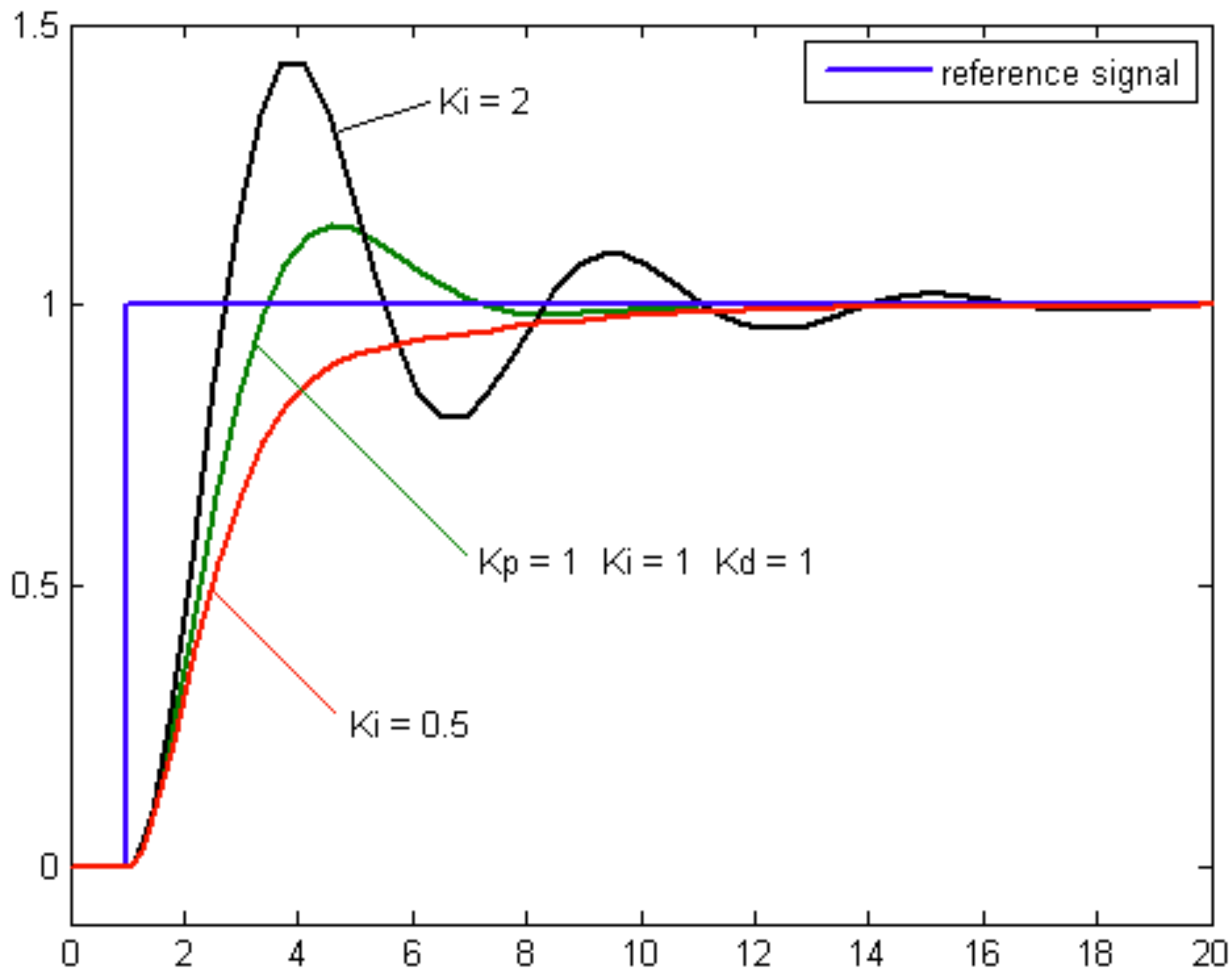
PID differential equation



- Relationship between input error e(t) and output control signal u(t) is thus captured by the differential equation:

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de}{dt}$$

Changing PID controller characteristics



- Tuning PID parameters strongly affects controller performance

Transfer function of PID controller

- Differential equation that described PID controller

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de}{dt}$$

- Taking Laplace transformations and rearranging gives the transfer function for the PID controller:

$$U(s) = K_p E(s) + K_i \frac{E(s)}{s} + K_d s E(s)$$

$$\Rightarrow U(s) = E(s)\left(K_p + \frac{K_i}{s} + K_d s \right)$$

$$\Rightarrow \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s$$

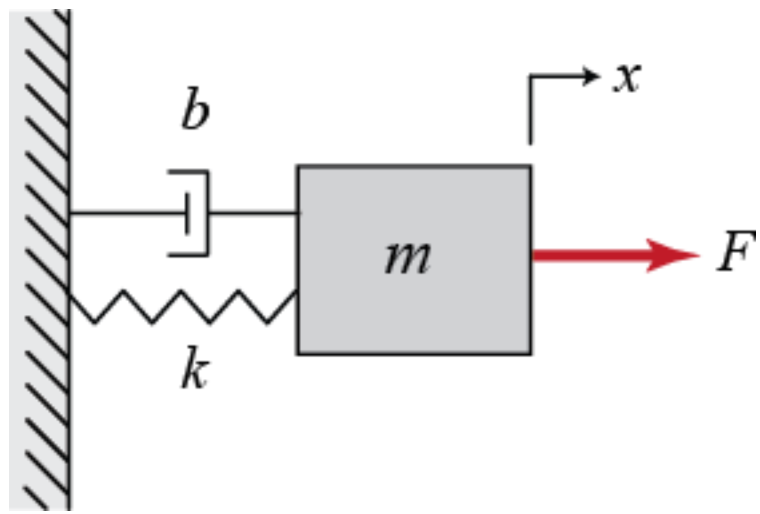
ROCO218: Control Engineering
Dr Ian Howard

Lecture 9

Simple PID control example

Analyze a simple mechanical system

- Consider the mass-spring-damper system shown below:



- Describe using differential equation of motion
- Balancing the forces gives:

$$F = m\frac{d^2x}{dt^2} + b\frac{dx}{dt} + kx$$

Analyze a simple mechanical system

Given the differential equation of motion

$$F = m\frac{d^2x}{dt^2} + b\frac{dx}{dt} + kx$$

The Laplace transform gives

$$F(s) = ms^2X(s) + bsX(s) + kX(s) \quad = X(s)\left(ms^2 + bs + k\right)$$

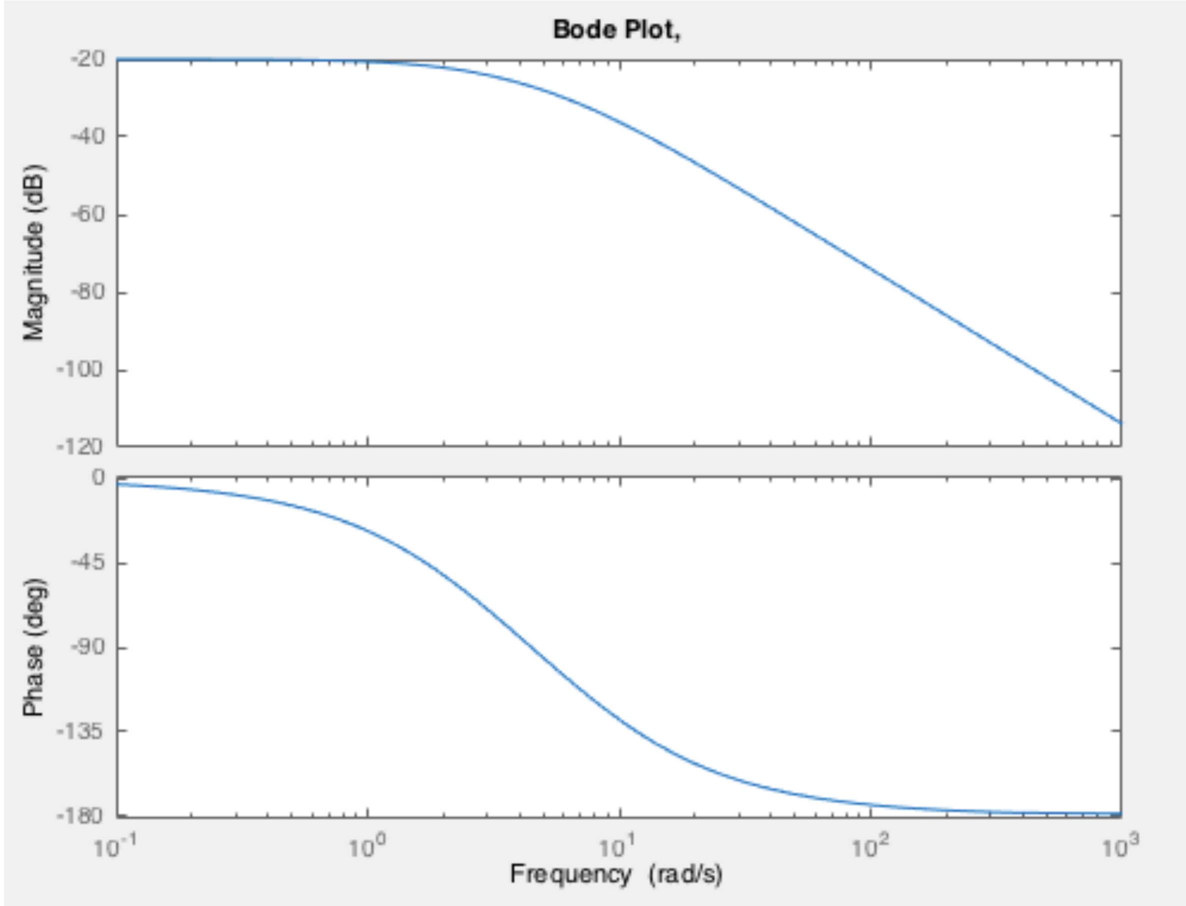
$$\frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

Use these parameters
M = 0.5 kg
b = 5 N s/m
k = 10 N/m
F = 1 N

$$P(s) = \frac{1}{0.5s^2 + 5s + 10}$$

Bode plot of a transfer function

- Lets use Matlab to generate a Bode plot of the transfer function of the mechanical system
- You should get a plot that looks like this:



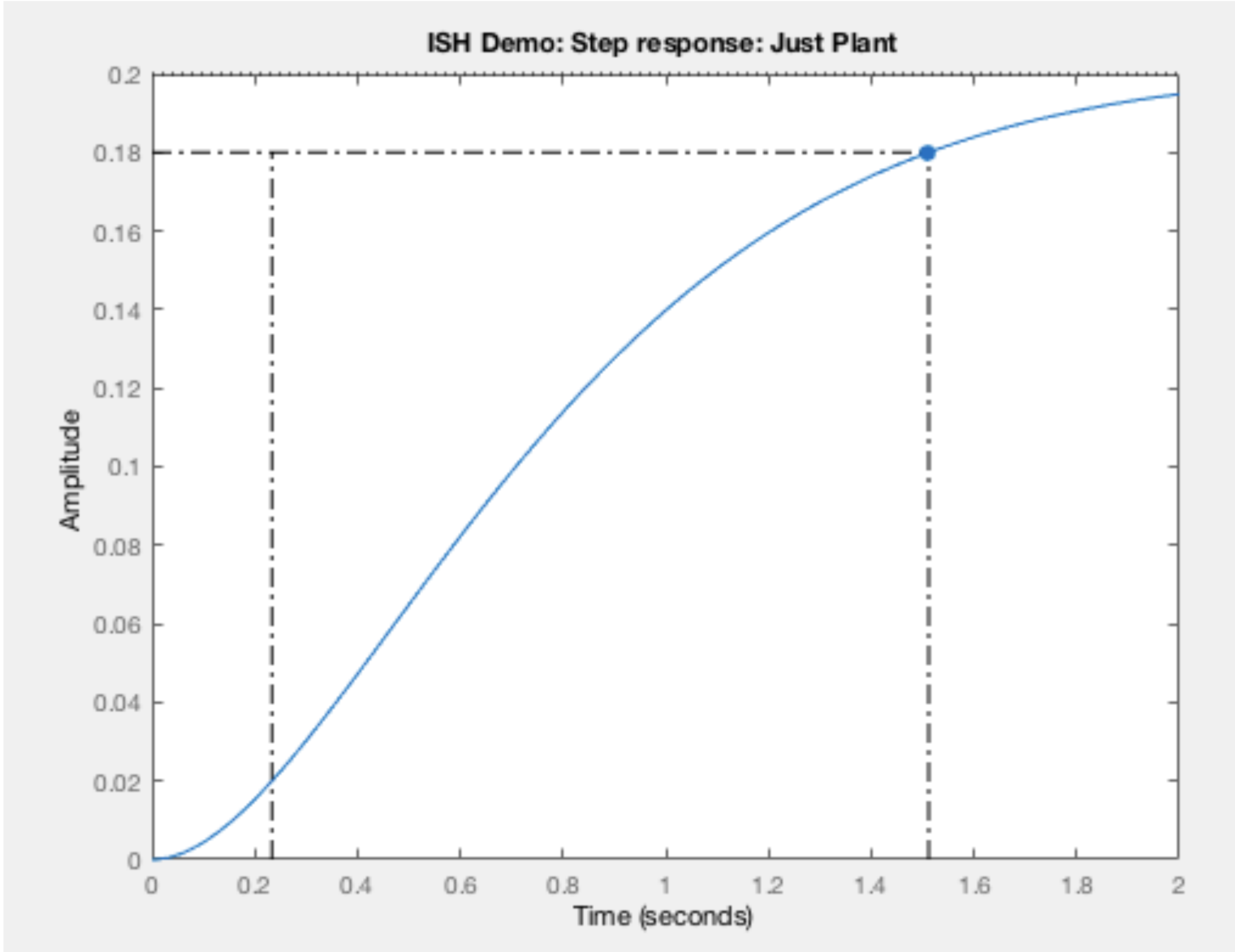
- Find the -3dB point
    - Where response goes down -3dB from max
- Estimate rise time
    - Time needed for value to go from 10% to 90%

Generate step response

- Lets use the Matlab step function to plot the step response of the system.

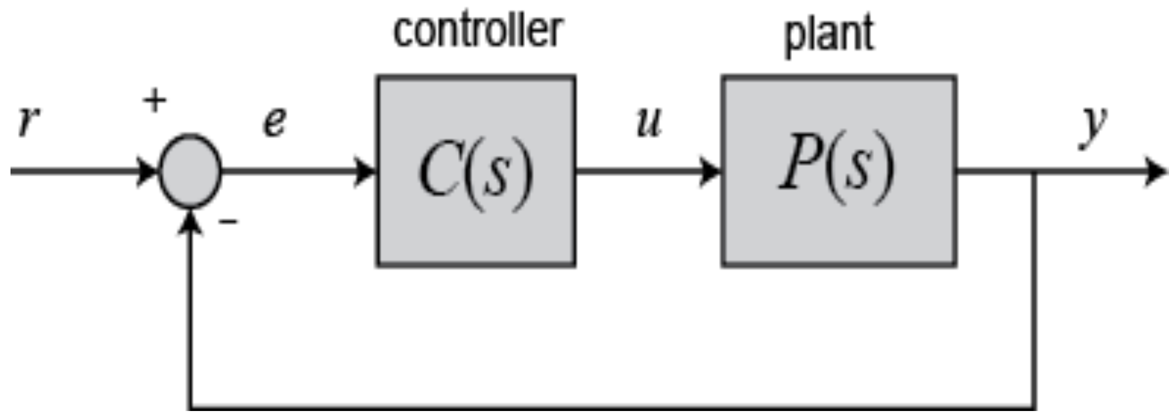
- Look at the Matlab help for the function step
- Pass it a time vector that gives time values between 0 and 2 seconds in steps of 0.01 seconds.

- What is the rise time here?



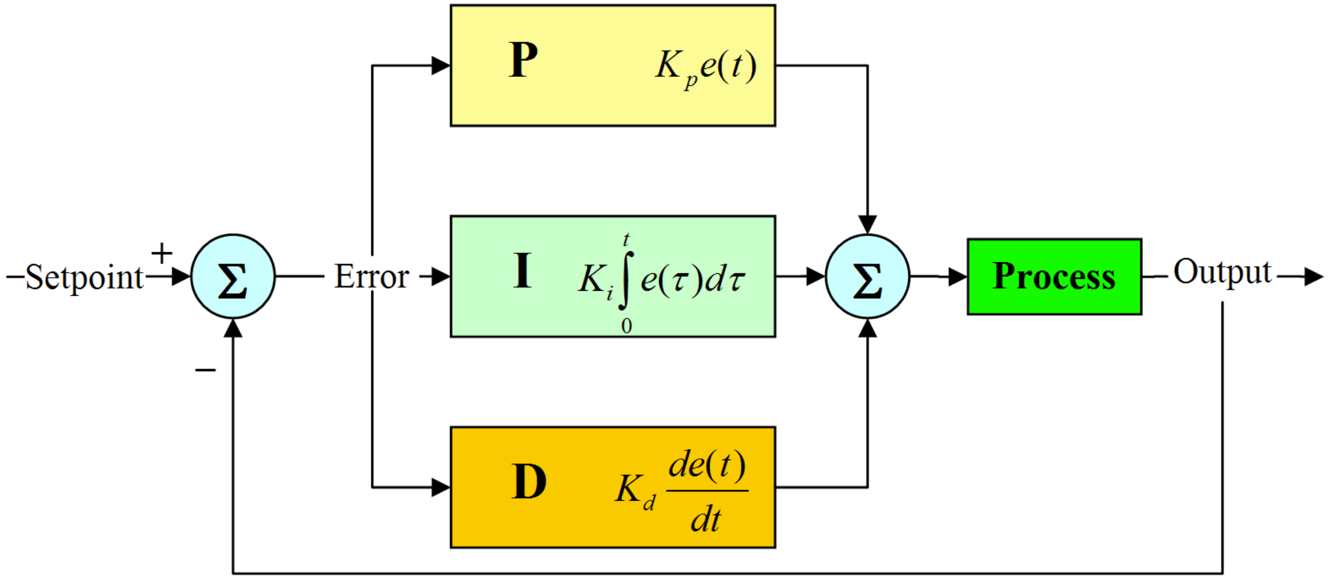
Adding a PID controller

- Lets now place a PID controller C(s) in series with the plant P(s) and make use of negative feedback



- A well-designed control system will ensure
  - We reach a target value quickly
  - Without overshoot
  - With low steady state error.

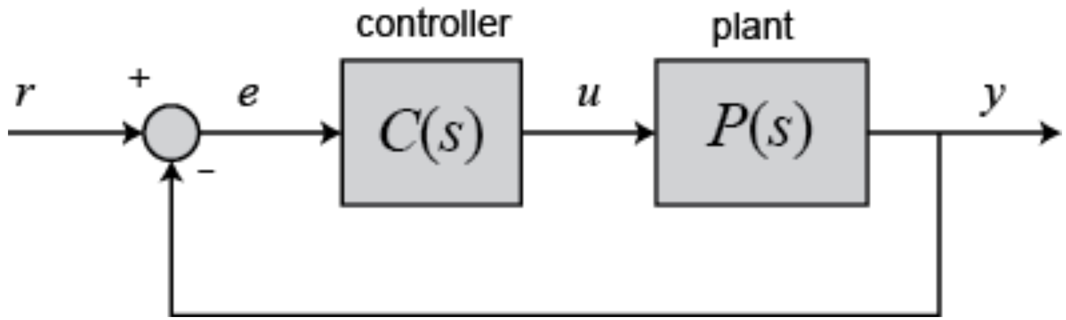
PID differential equation



- Relationship between input error e(t) and output control signal u(t) captured by the differential equation:

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de}{dt}$$

Example: Overall open-loop transfer function



Controller gives

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

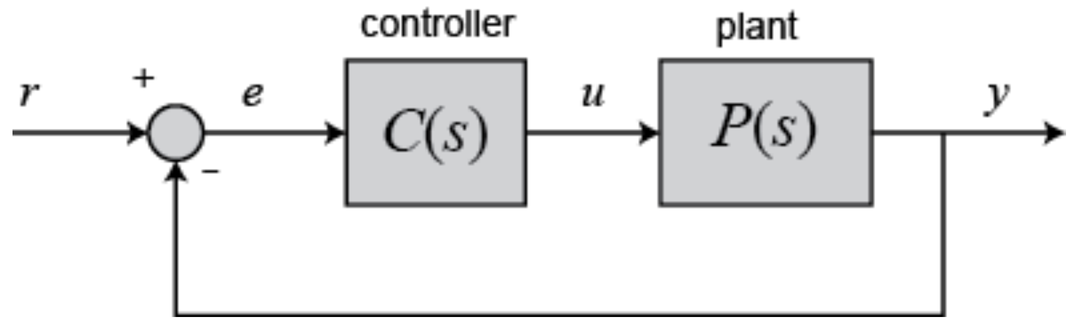
Plant gives

$$P(s) = \frac{1}{0.5s^2 + 5s + 10}$$

Product of these series elements gives

$$C(s)P(s) = \frac{K_p + \frac{K_i}{s} + K_d s}{0.5s^2 + 5s + 10} = \frac{K_p s + K_i + K_d s^2}{0.5s^3 + 5s^2 + 10s}$$

Overall closed-loop transfer function

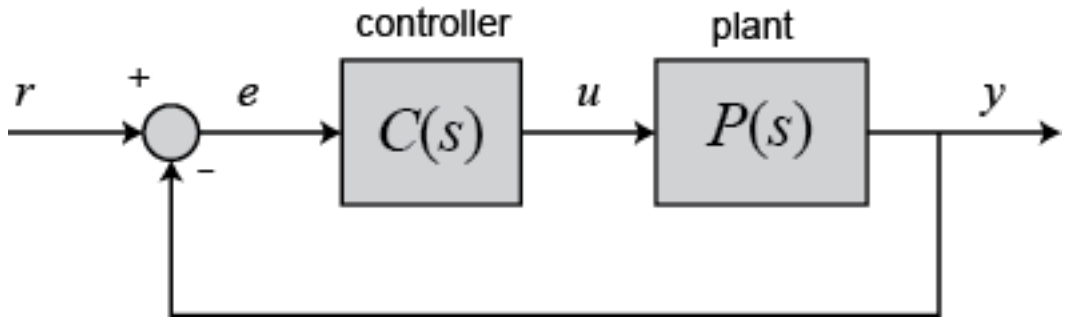


Given this feedback arrangement and serial controller and plant configuration

The close loop transfer function is

$$\frac{Y(s)}{R(s)} = \frac{C(s)P(s)}{1+C(s)P(s)}$$

Example: Overall closed-loop transfer function



So given $$C(s)P(s) = \frac{K_p s + K_i + K_d s^2}{0.5s^3 + 5s^2 + 10s}$$

The close loop transfer function is

$$\boxed{\frac{Y(s)}{R(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)}}$$

$$\frac{Y(s)}{R(s)} = \frac{\dfrac{K_p s + K_i + K_d s^2}{0.5s^3 + 5s^2 + 10s}}{1 + \dfrac{K_p s + K_i + K_d s^2}{0.5s^3 + 5s^2 + 10s}} = \frac{K_d s^2 + K_p s + K_i}{0.5s^3 + (K_d + 5)s^2 + (K_p + 10)s + K_i}$$

ROCO218: Control Engineering
Dr Ian Howard

Lecture 9

Simulating PID in Matlab

Building plant transfer function in Matlab

- Matlab can perform symbolic manipulation
  - (similar to Mathematica)

- The tf function generates Matlab format transfer functions
- You can use it to specify a complex transfer function
- Or even a simple simple ones
- Then use mathematical manipulation on the variable to generate other transfer function relationships
- You can of course always do the maths by hand if you like
- But Matlab is less likely to make a mistake than a human!
- You can then use Matlab functions like Bode and Nyquist to directly analyze the transfer functions

Building plant transfer function in Matlab

Define plant parameter variables

```
% parameters
F = 1;
M = 0.5;
b = 5;
k = 10;
```

Build the plant transfer function

```
% use Matlab transfer function to setup s element
s = tf('s');

% build plant transfer function
P = 1/(M * s^2 + b *s + k);
```

Displaying content of P gives

```
P =

          1
  ------------------
  0.5 s^2 + 5 s + 10

Continuous-time transfer function.
```

Building plant transfer function in Matlab

Define example PID parameter variables

```
% full unitiy gain  proportional integral diffenetial gain
Kp = 1;
Ki = 1;
Kd = 1;
```

Build the PID controller transfer function

```
% build PID transfer function
PIDC = Kp + Ki/s + Kd*s;
```

Displaying content of PIDC gives

```
PIDC =

  s^2 + s + 1
  -----------
       s
```

Calculating open-loop transfer function

Given plant and PID transfer functions

```
% build plant transfer function
P = 1/(M * s^2 + b *s + k);

% build PID transfer function
PIDC = Kp + Ki/s + Kd*s;
```

```
% full unitiy gain  proportional integral diffenetial gain
Kp = 1;
Ki = 1;
Kd = 1;
```

Open-loop transfer function is their product

```
% overall open loop transfer funtion
OL = PIDC * P;
```

Displaying content of OL gives

```
OL =

       s^2 + s + 1
  ----------------------
  0.5 s^3 + 5 s^2 + 10 s

Continuous-time transfer function.
```

Calculating closed-loop transfer function

Given plant and PID transfer functions

```
% build plant transfer function
P = 1/(M * s^2 + b *s + k);

% build PID transfer function
PIDC = Kp + Ki/s + Kd*s;
```

```
% full unitiy gain  proportional integral diffenetial gain
Kp = 1;
Ki = 1;
Kd = 1;
```

Open-loop transfer function is their product

```
% overall open loop transfer funtion
OL = PIDC * P;
```

Closed-loop transfer function is given by

```
% compute closed loop function
FB = OL / (OL + 1);
```

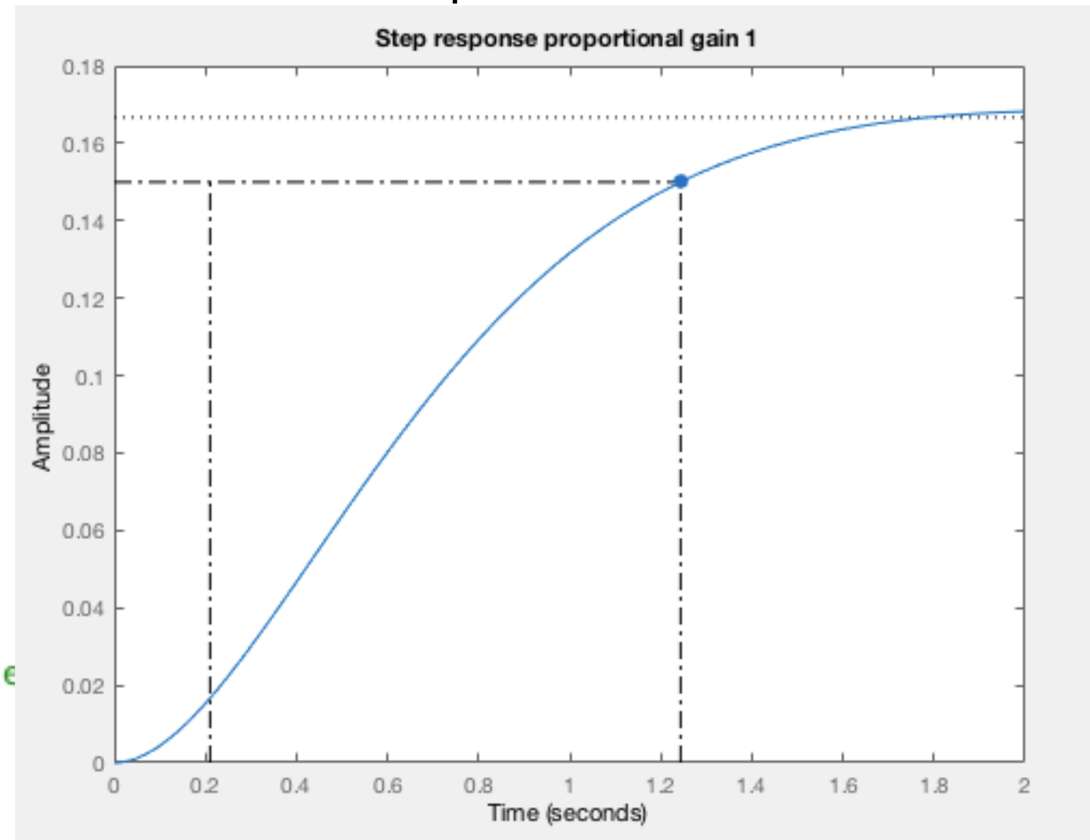
Displaying content of FB gives

```
FB =

        0.5 s^5 + 5.5 s^4 + 15.5 s^3 + 15 s^2 + 10 s
  -----------------------------------------------------------
  0.25 s^6 + 5.5 s^5 + 40.5 s^4 + 115.5 s^3 + 115 s^2 + 10 s
```

Feedback control with only proportional gain

```
% with only proportional gain feedback
Kp = 1;
Ki = 0;
Kd = 0;

% build PID transfer function
PIDC = Kp + Ki/s + Kd*s;

% overall open loop trabsfer funtion
OL = PIDC * P;

% compute closed loop function
FB = OL / (OL + 1);


% use Matlab step function to generate ste
figure
hold on
t = 0:0.01:2;
step(FB, t);
title('Step response proportional gain 1');
```

$K_p = 1$

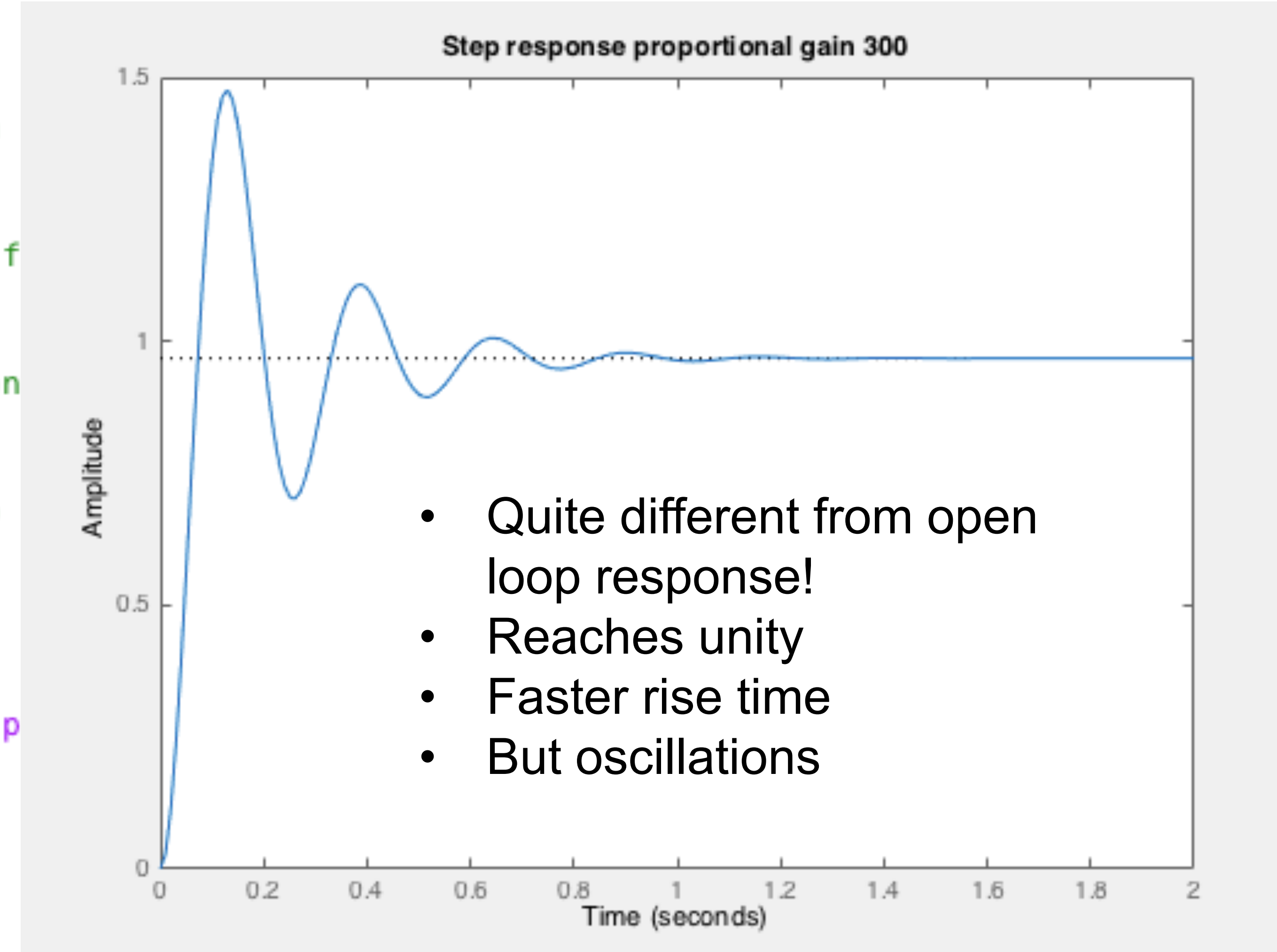


Not much has changed from open loop response!

Feedback control with only proportional gain

$K_p = 300$

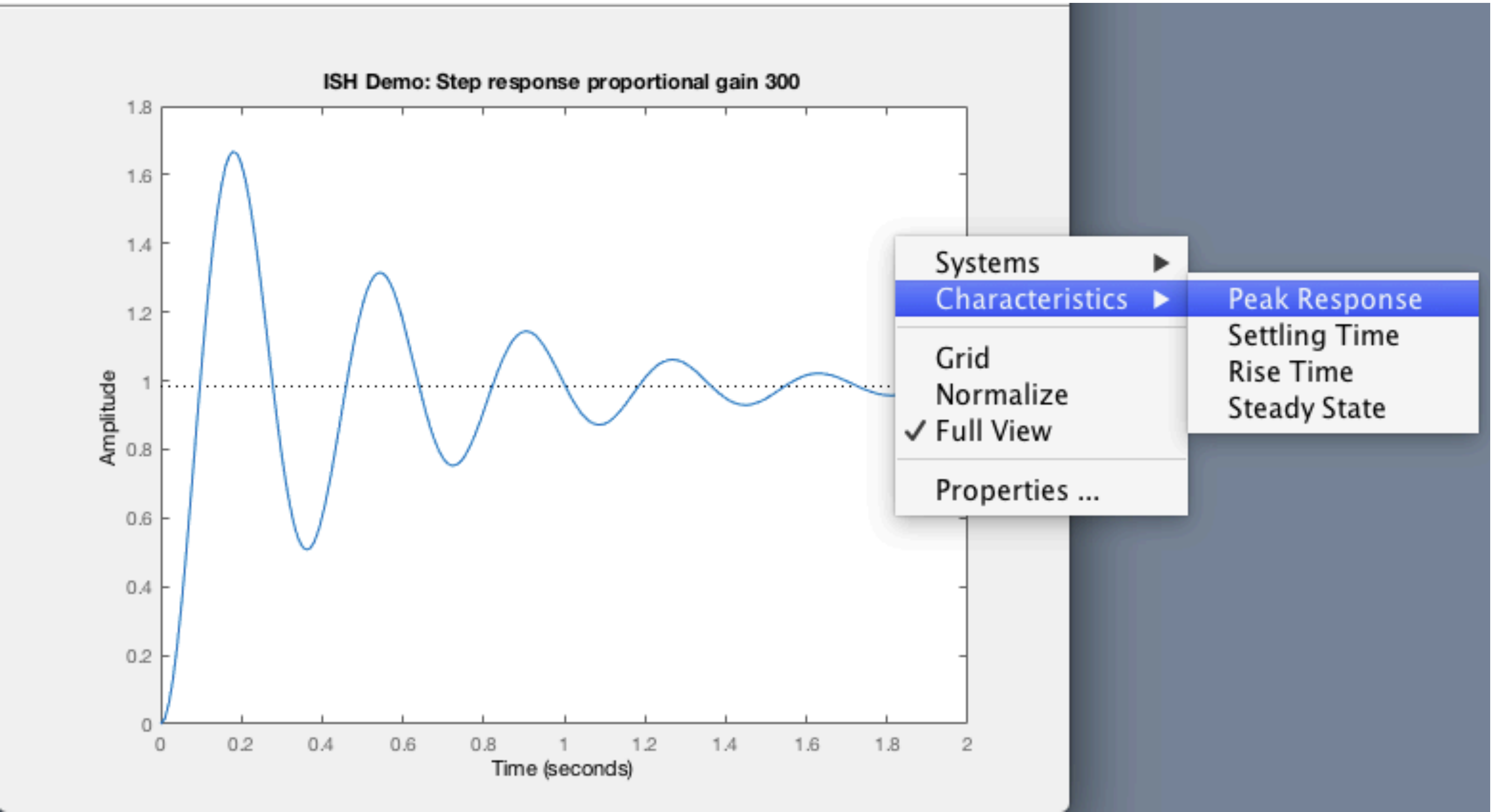
```
% with only proportional gain
Kp = 300;
Ki = 0;
Kd = 0;

% build PID transfer function
PIDC = Kp + Ki/s + Kd*s;

% overall open loop transfer f
OL = PIDC * P;

% compute closed loop function
FB = OL / (OL + 1);

% use Matlab step function to
figure
hold on
t = 0:0.01:2;
step(FB, t);
title(sprintf('Step response p
```



- Quite different from open loop response!
- Reaches unity
- Faster rise time
- But oscillations

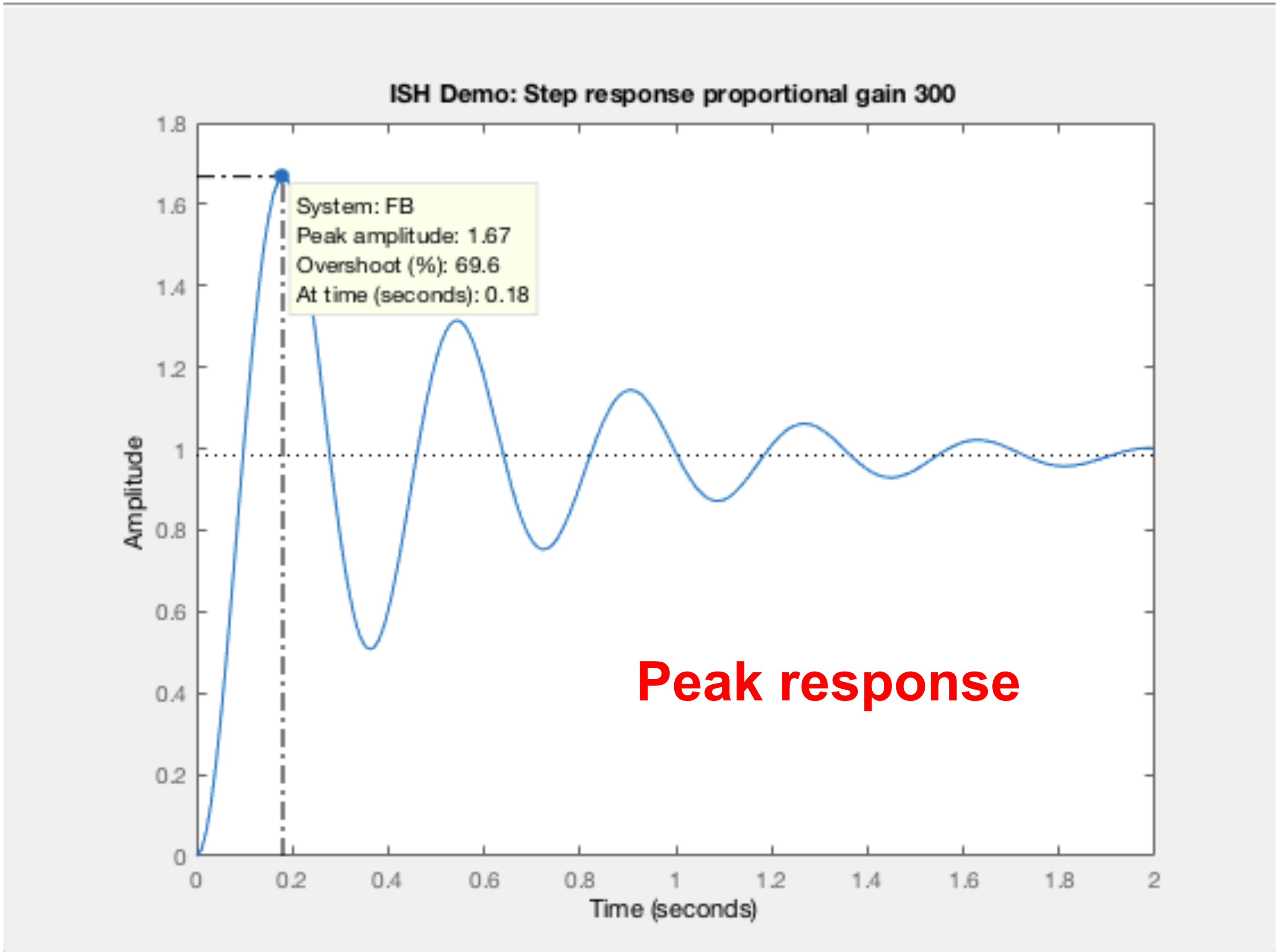
Use Matlab step function to show response characteristics

$K_p = 300$



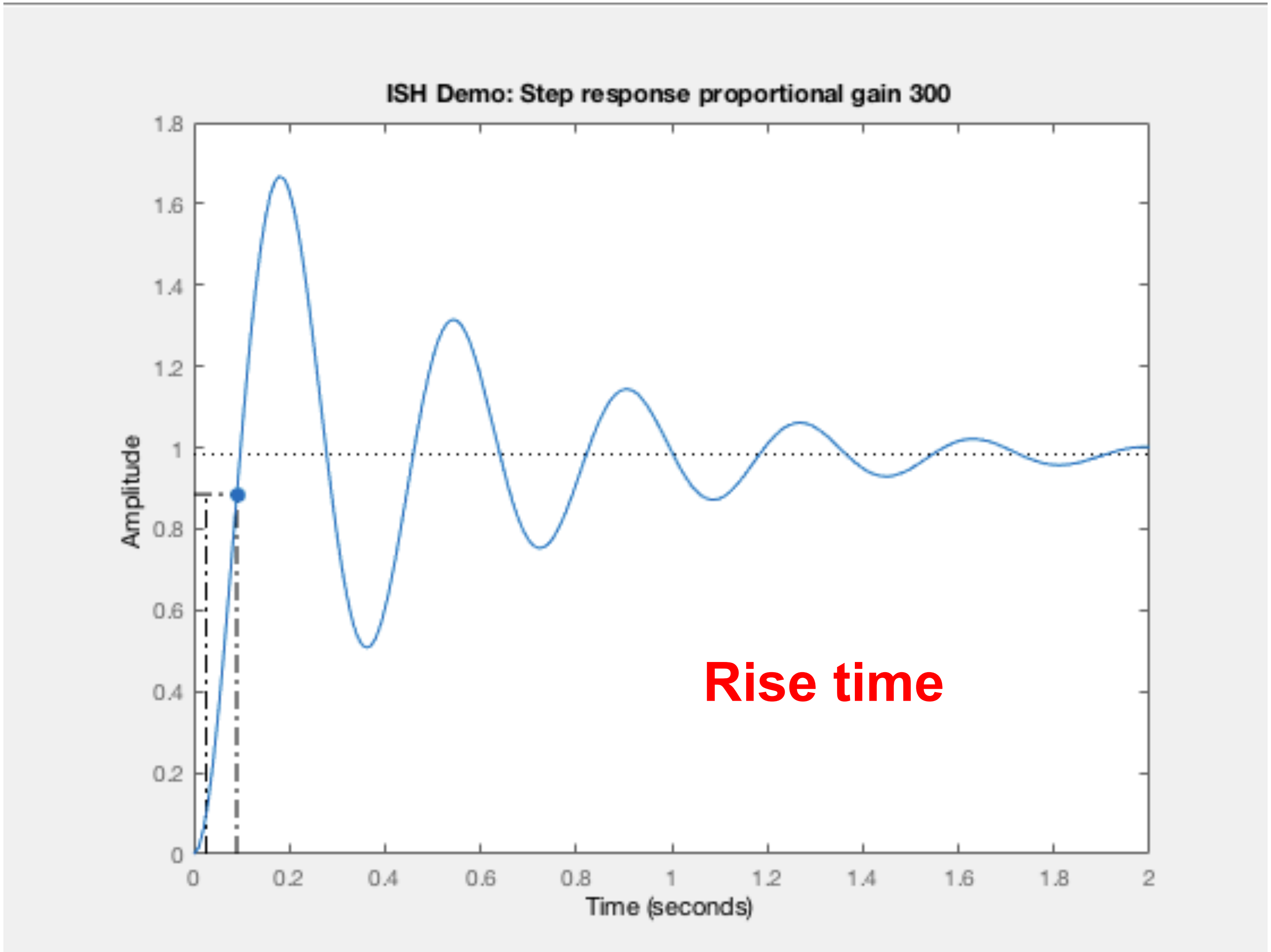
Feedback control with only proportional gain

$K_p = 300$



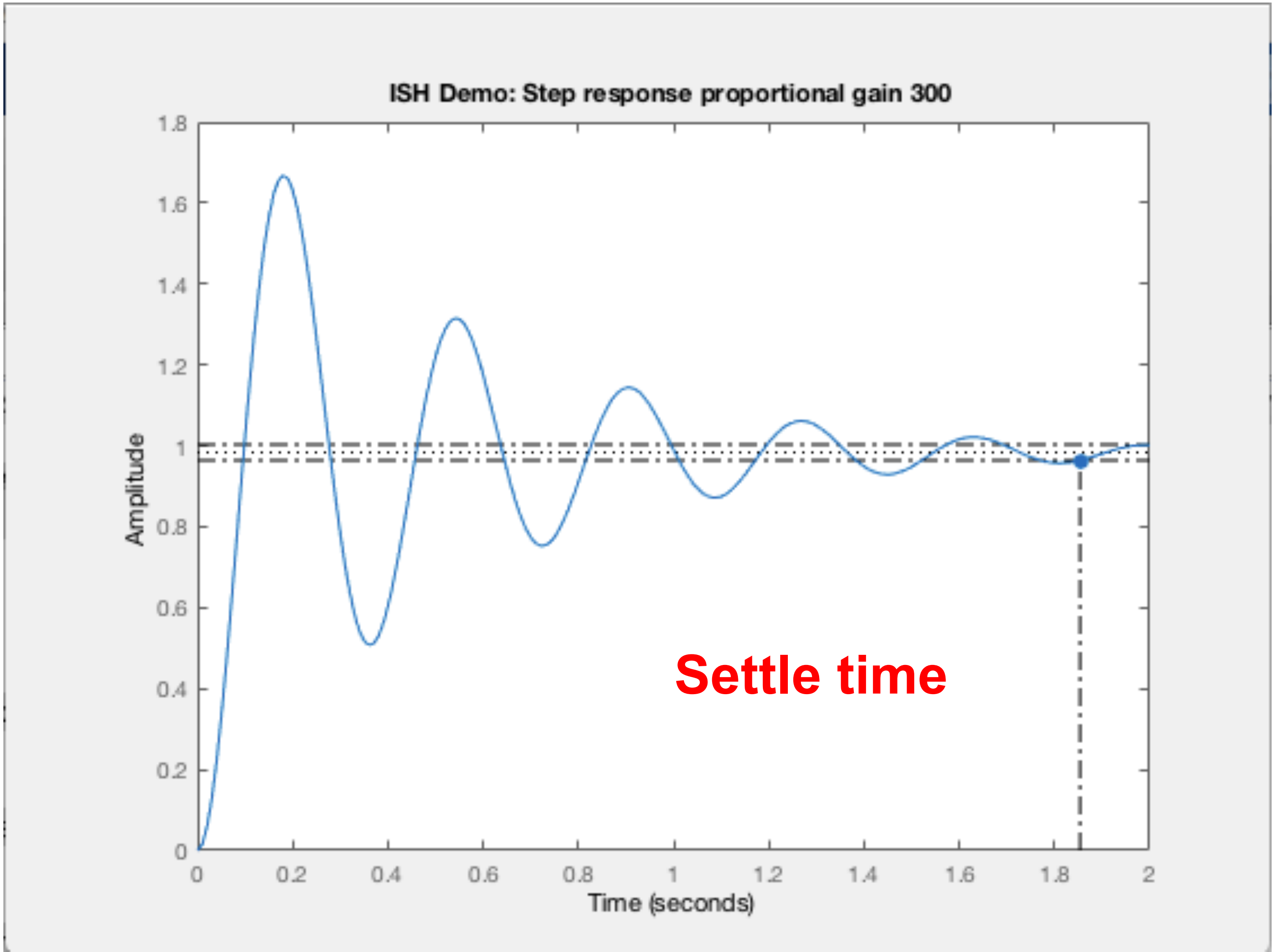
Feedback control with only proportional gain

$K_p = 300$



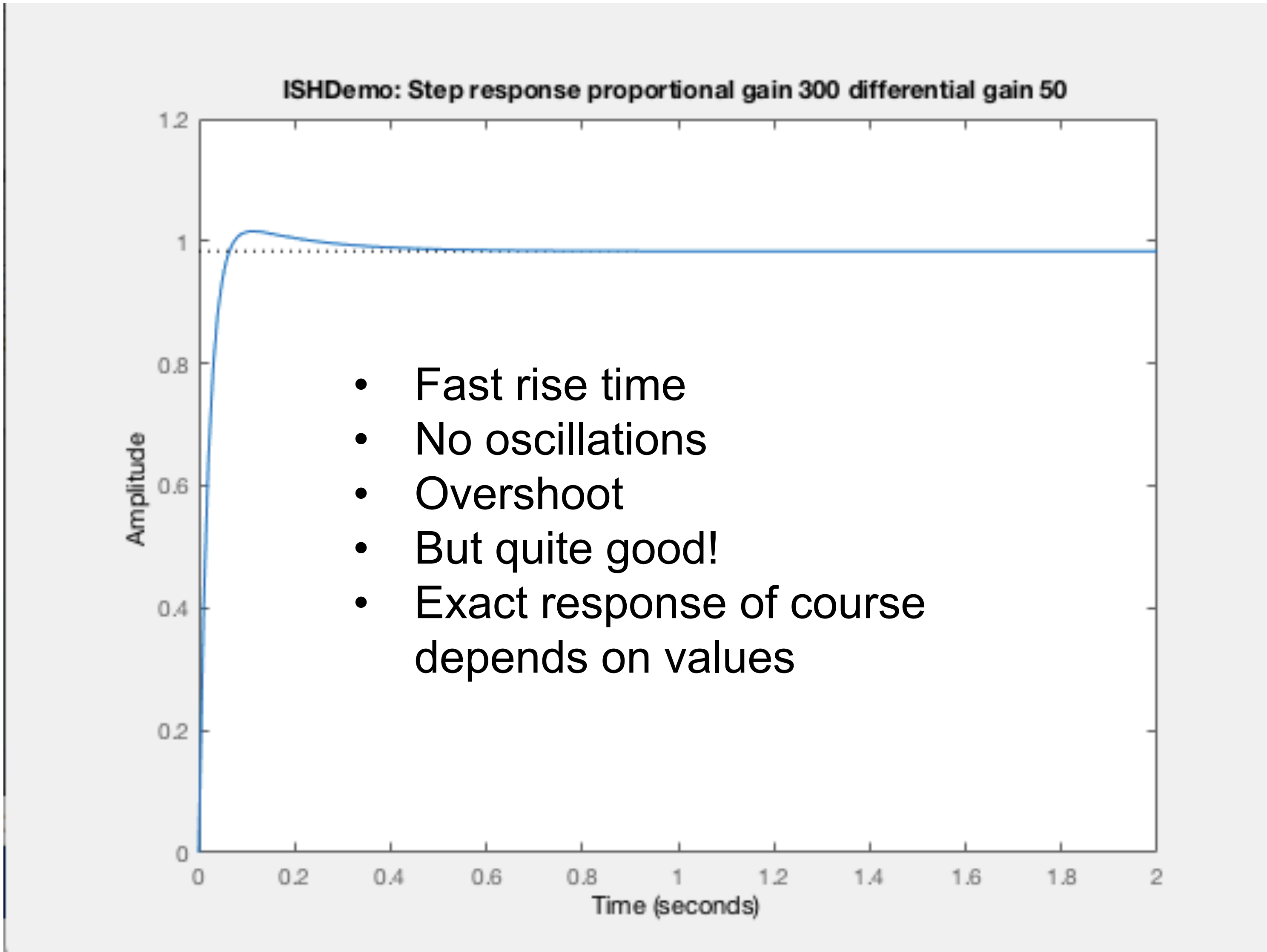
Feedback control with only proportional gain

$K_p = 300$



With proportional & differential gain

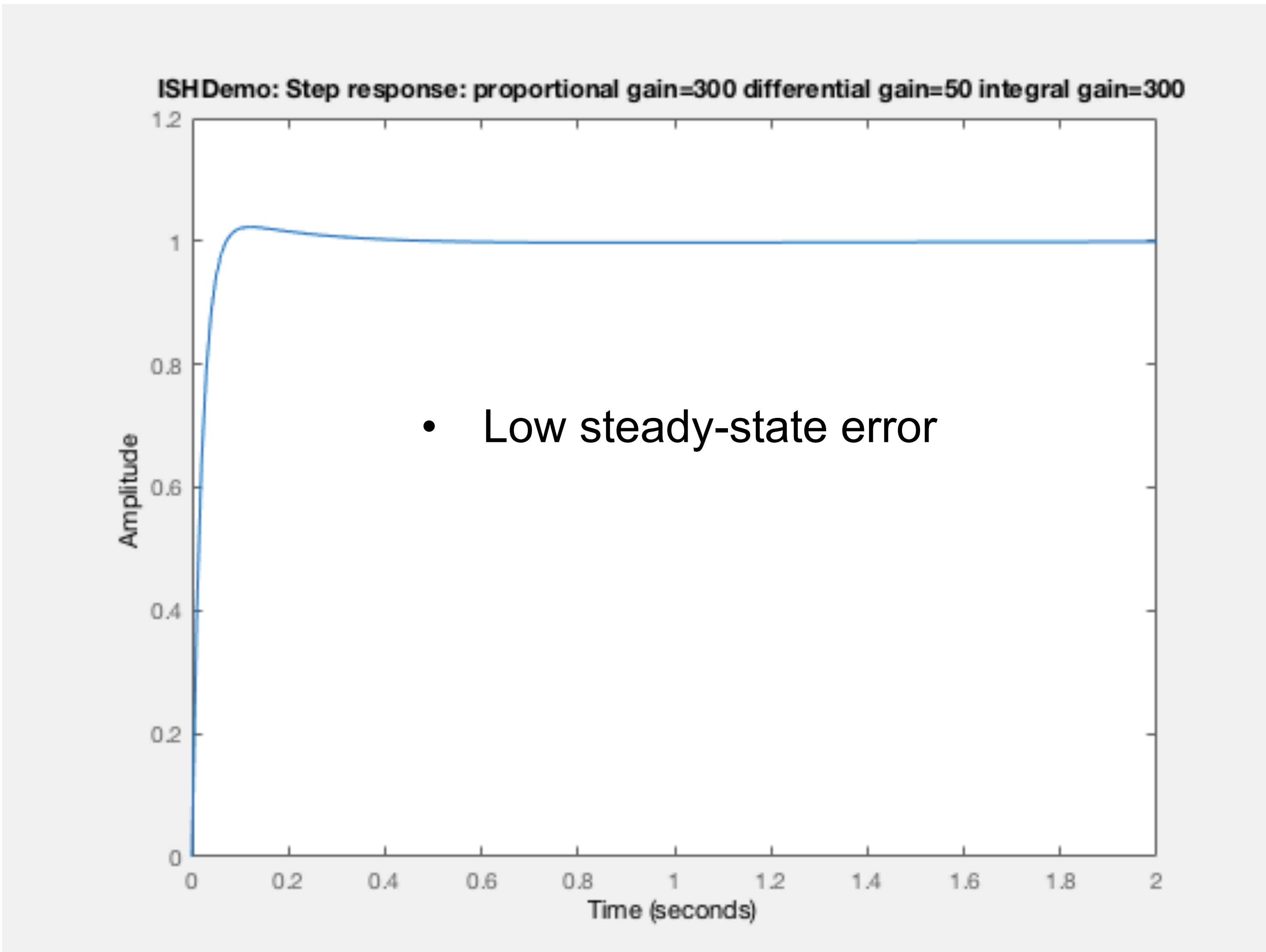
$K_p = 300$ and $K_d = 50$



- Fast rise time
- No oscillations
- Overshoot
- But quite good!
- Exact response of course depends on values

With full PID gains

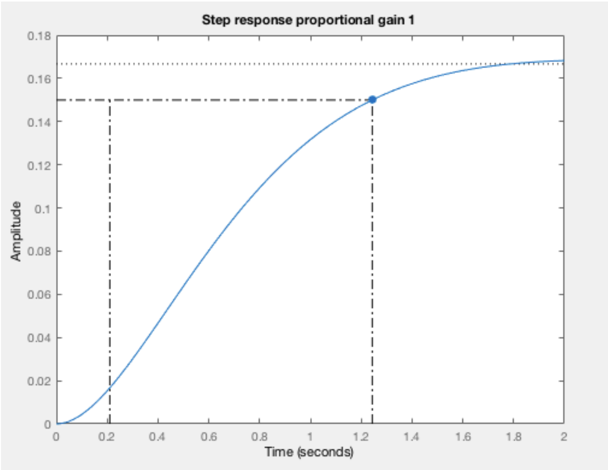
$K_p = 300, K_d = 50, K_i = 300$



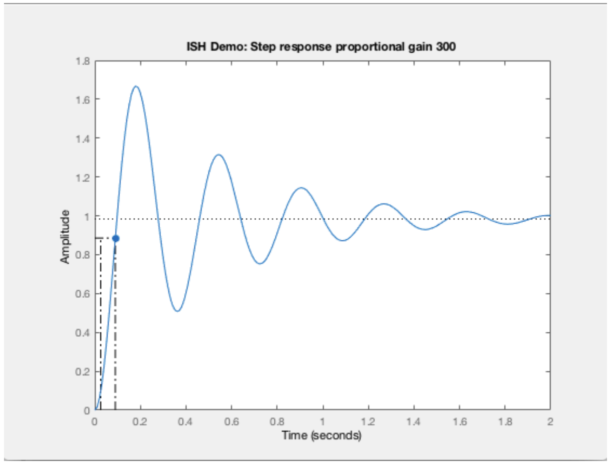
- Low steady-state error

Compare

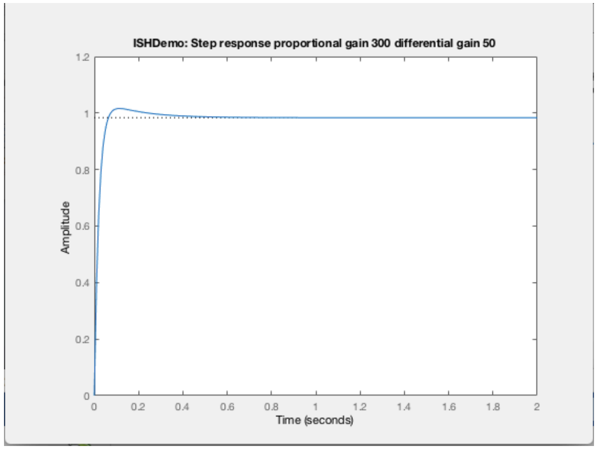
Open loop



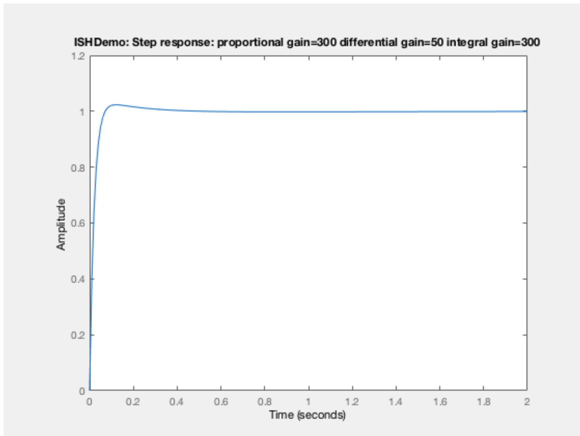
Just P



PD



PID



- But precise response depends on appropriate tuning of all feedback components