EE3304: Digital Control Systems (Part 2)

Ben M. Chen
Associate Professor

Department of Electrical & Computer Engineering
The National University of Singapore

Phone: 6874-2289 Office: Block E4-6-7
Email: bmchen@nus.edu.sg
http://vlab.ee.nus.edu.sg/~bmchen

Course outline — Part 2

- Introduction to control systems; control system examples; basic principle of feedback control; brief review of control development history.
- Digital control systems design: Time domain specifications; Dynamic response to unit step and ramp functions. Stability of discrete time systems; Digital PID design; Pole placement design.
- Digital Control system design: Frequency domain specifications; Gain and phase margins; Compensator design with bilinear transformations.
- Digital control system design through state space approach: State space description of discrete systems; State feedback design via pole placement; State estimator design; Controller design with state estimator.

Textbook — Primary selection

- GF Franklin, JD Powell and ML Workman, *Digital Control of Dynamic Systems*, 3rd Edition, Addison Wesley, 1998.

Homework assignments

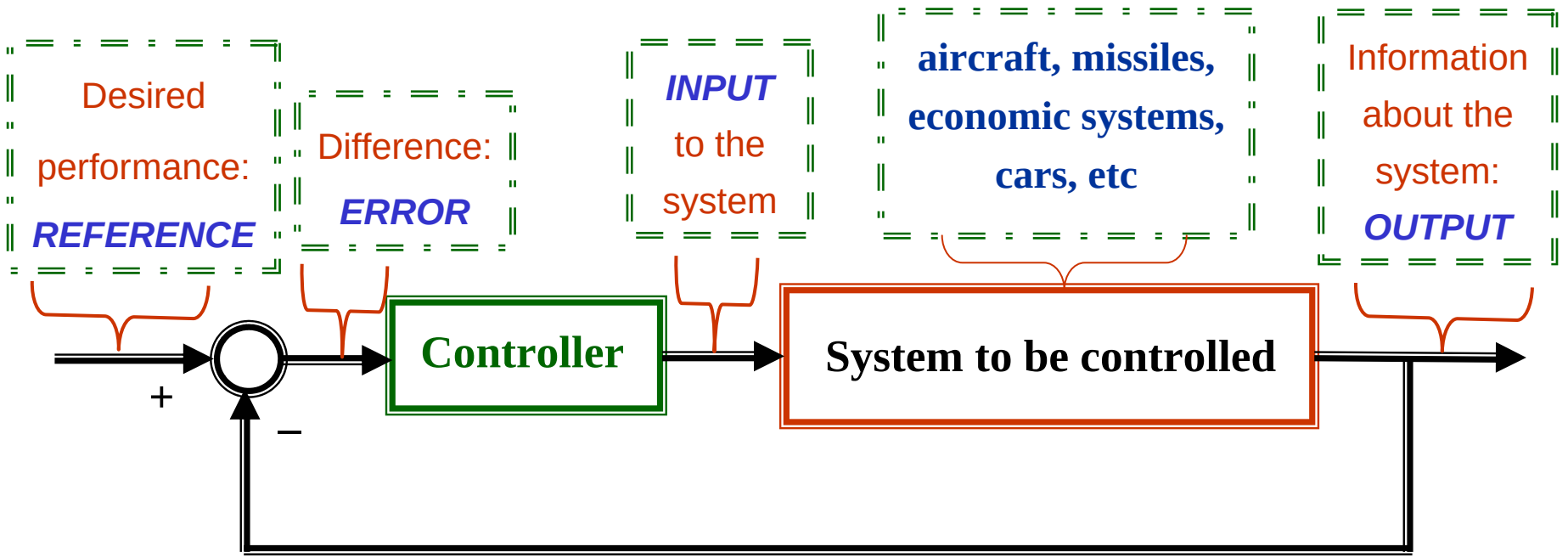
- There will be 3 homework assignments for this second part. The homework assignments will be graded and credited as 10% of the final grade.

Additional note on supporting software

- Students are expected to be familiar with the computational software tool MATLAB and its package SIMULINK.

1. Introduction & Revision

1.1. What is a control system?

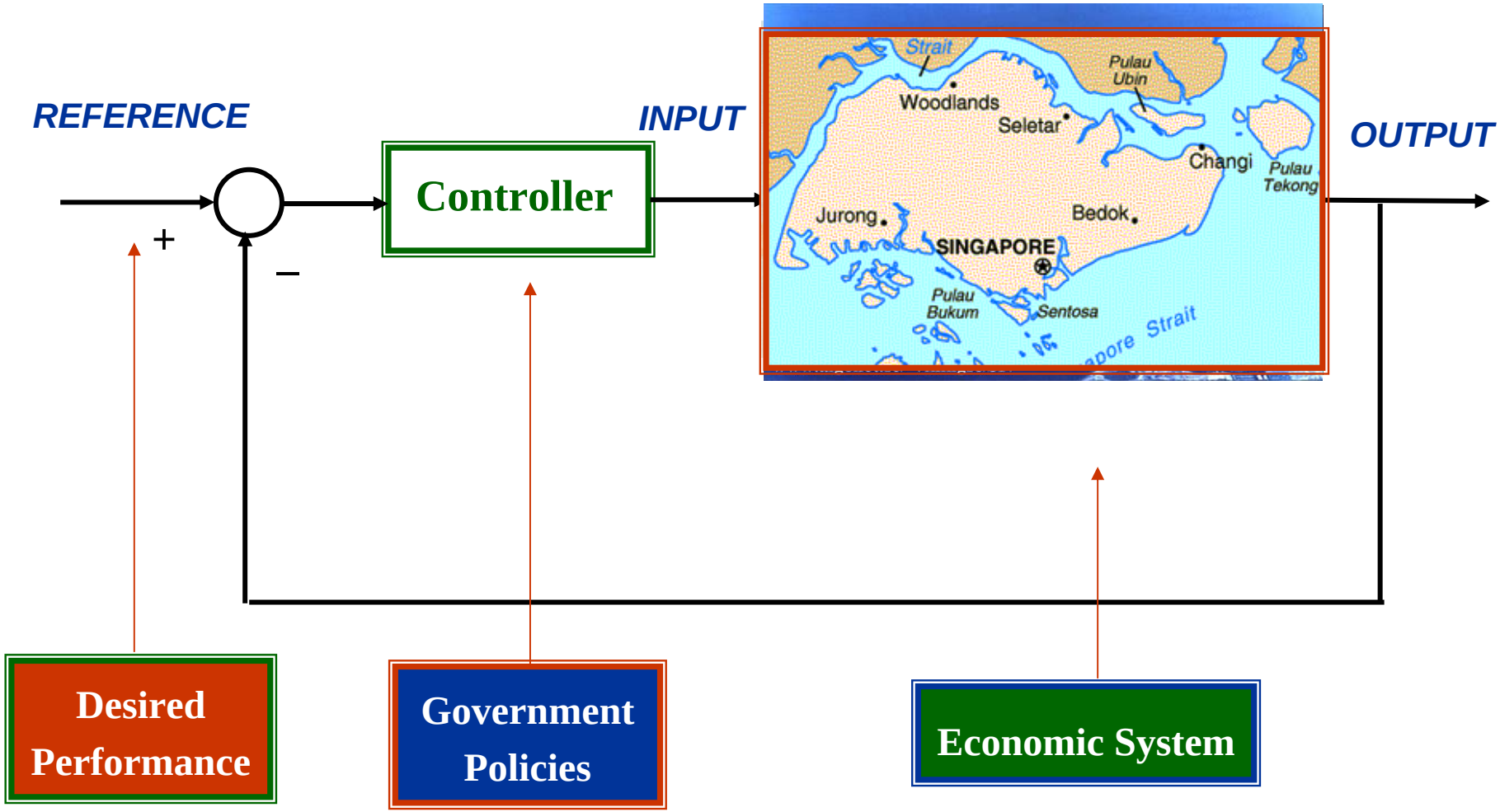


Objective: To make the system *OUTPUT* and the desired *REFERENCE* as close as possible, i.e., to make the *ERROR* as small as possible.

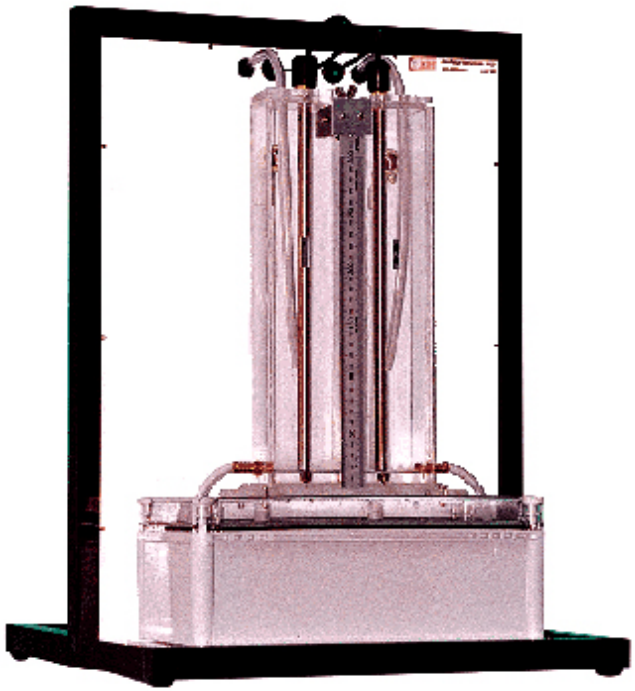
Key Issues: 1) How to describe the system to be controlled? (Modelling)

2) How to design the controller? (Control)

1.2. Some control systems examples:



1.3. A live demonstration on control of a coupled-tank system through Internet based virtual laboratory in ECE Dept, NUS



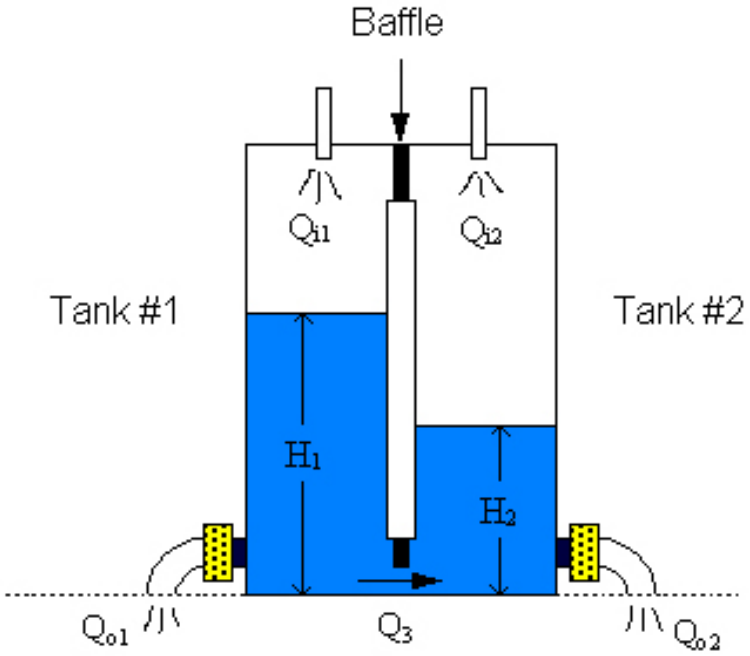
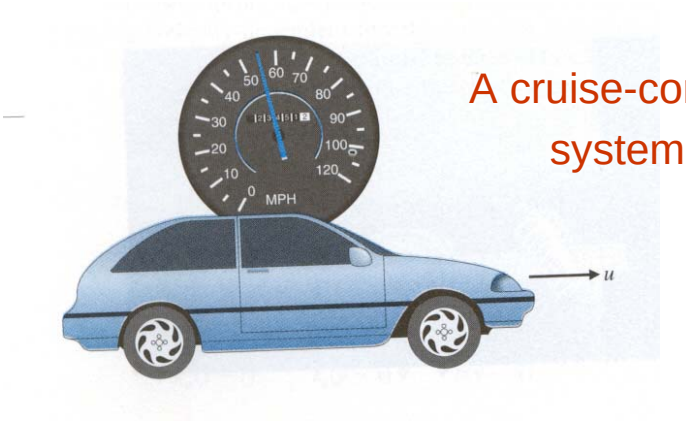


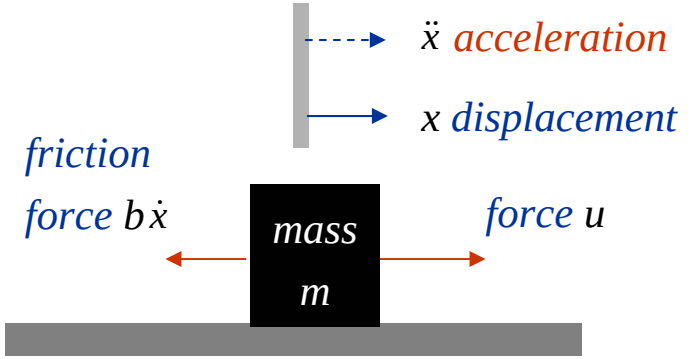
Figure 1 Diagram of coupled-tank control apparatus

The objective is to control the flow levels of two coupled tanks. It is a reduced-scale model of some commonly used chemical plants.

1.4. Modeling of a physical system — A simple mechanical system



A cruise-control system



By the well-known Newton's Law of motion: $f = m\,a$, where f is the total force applied to an object with a mass m and a is the acceleration, we have

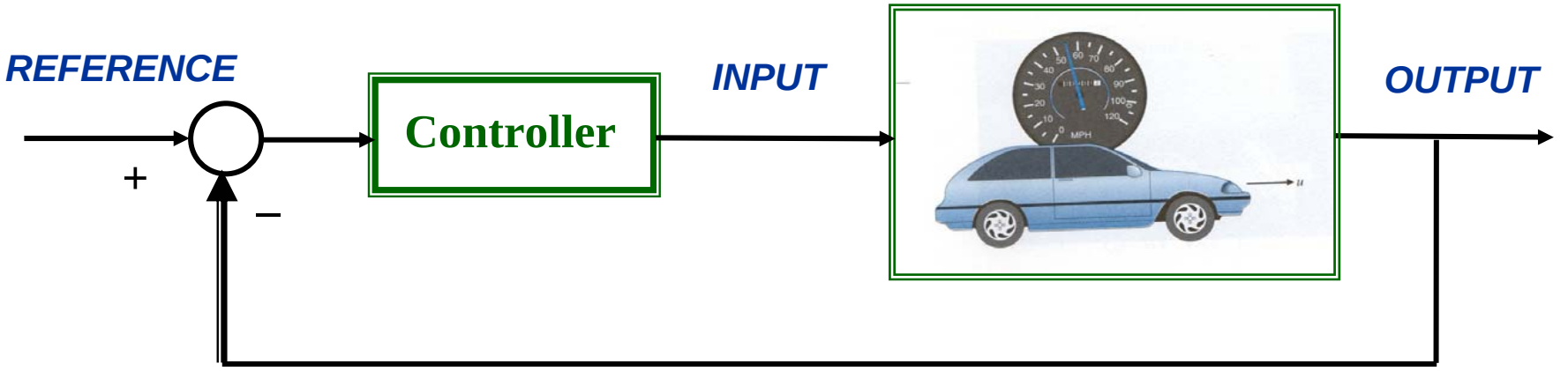
$$u - b\dot{x} = m\ddot{x} \quad \Leftrightarrow \quad \ddot{x} + \frac{b}{m}\dot{x} = \frac{u}{m}$$

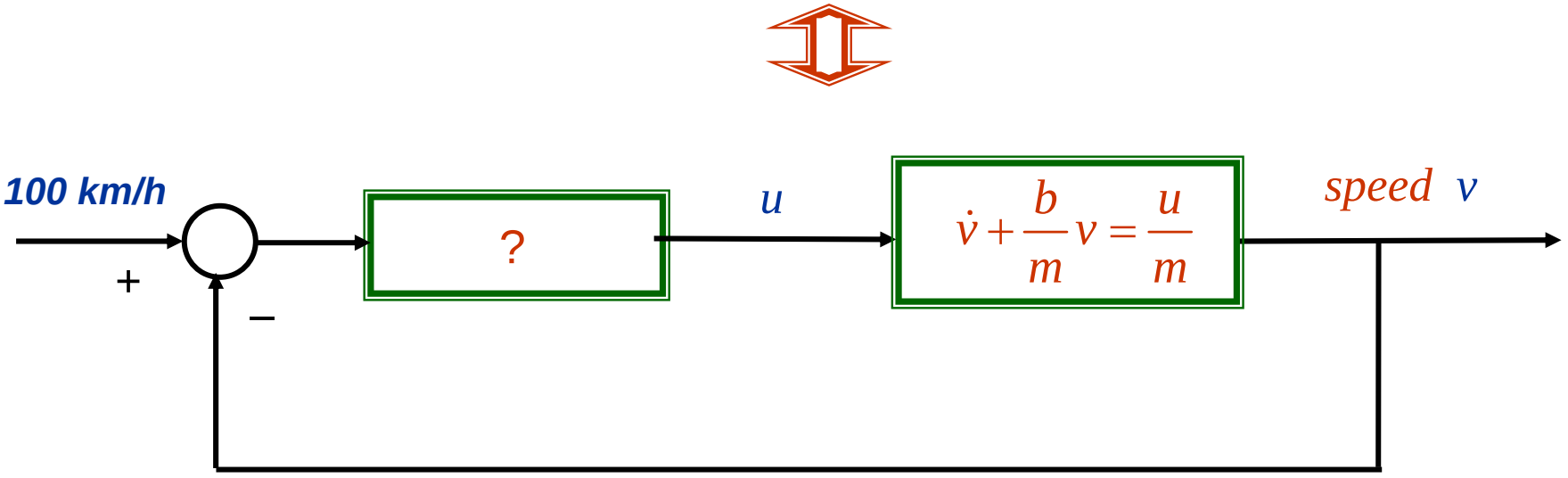
This a 2nd order *Ordinary Differential Equation* with respect to displacement x. It can be written as a 1st order *ODE* with respect to speed $v = \dot{x}$:

$$\dot{v} + \frac{b}{m}v = \frac{u}{m}$$

← model of the cruise control system, u is input force, v is output.

1.5. A cruise-control system:





1.6. Re-express ODE models using Laplace transform

Recall that the mechanical system in the cruise-control problem can be represented by an ODE:

$$m\dot{v} + bv = u$$

Taking Laplace transform on both sides of the equation, we obtain

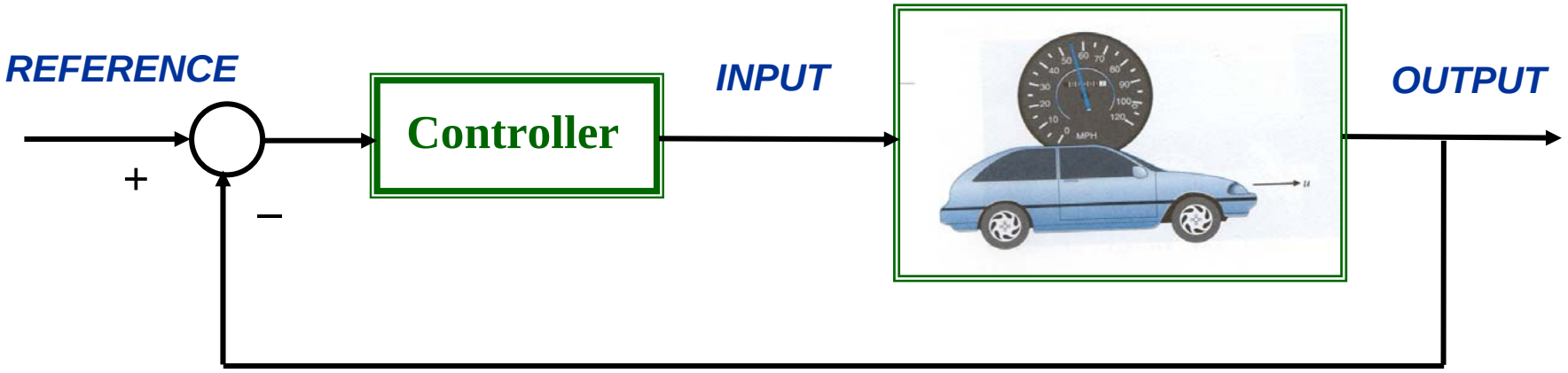
$$L\{m\,\dot{v} + bv\} = L\{u\} \quad \Rightarrow \quad L\{m\dot{v}\} + L\{bv\} = L\{u\}$$

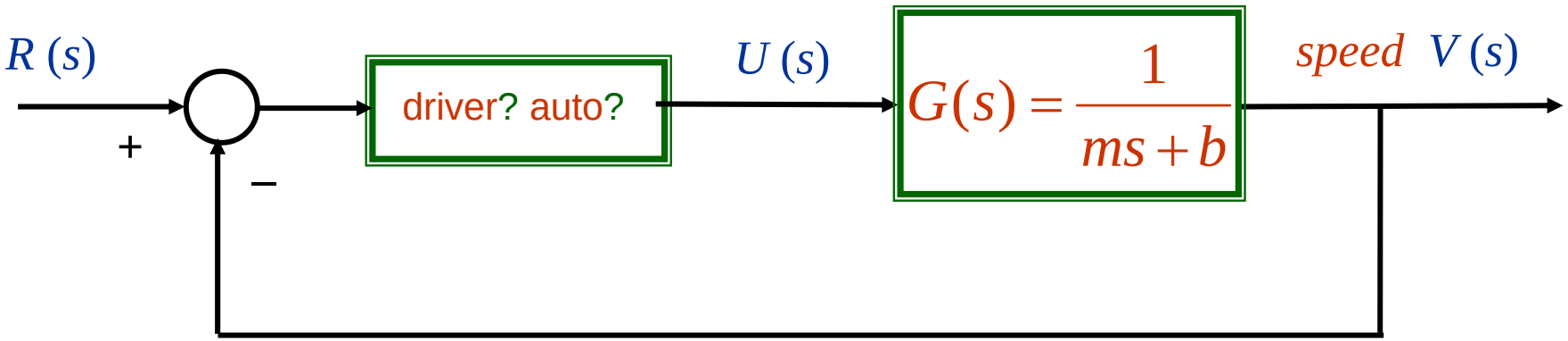
$$\Rightarrow \quad msL\{v\} + bL\{v\} = L\{u\} \quad \Rightarrow \quad msV(s) + bV(s) = U(s)$$

$$\Rightarrow \quad (ms + b)V(s) = U(s) \quad \Rightarrow \quad \frac{V(s)}{U(s)} = \frac{1}{ms + b} = G(s)$$

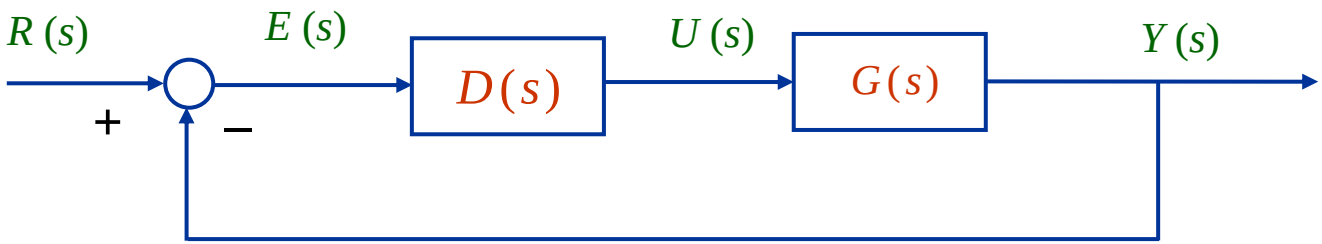
This is called the transfer function of the system model

1.7. A cruise-control system in frequency domain:

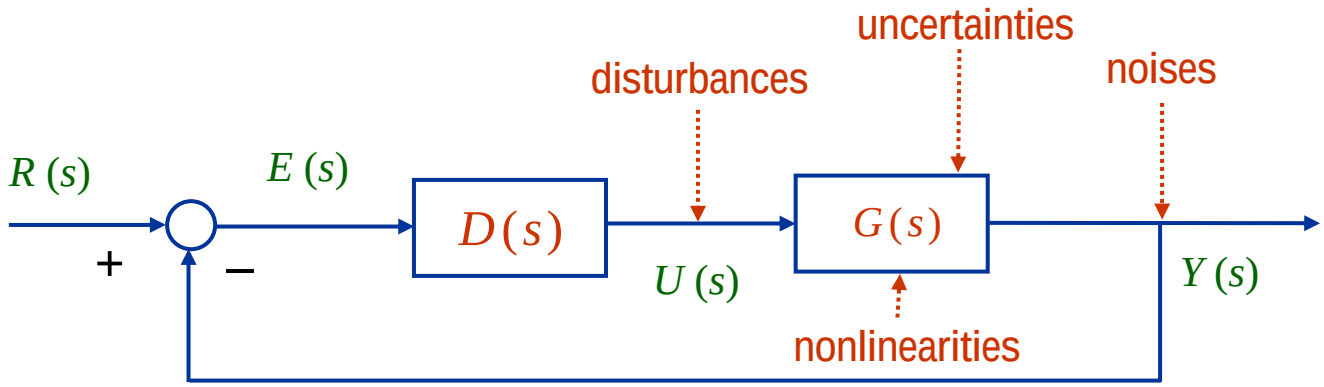




In general, a feedback control system can be represented by the block diagram below:



Given a system represented by $G(s)$ and a reference $R(s)$, the objective of control system design is to find a control law (or controller) $D(s)$ such that the resulting output $Y(s)$ is as close to reference $R(s)$ as possible, or the error $E(s) = R(s) - Y(s)$ is as small as possible. However, many other factors of life have to be carefully considered when dealing with real-life problems. These factors include:



1.8. Brief view of control techniques:

There are tons of research published in the literature on how to design control laws for various purposes. These can be roughly classified as the following:

- Classical control: Proportional-integral-derivative (PID) control, developed in 1940s and used for control of industrial processes. Examples: chemical plants, commercial aeroplanes.

- Optimal control: Linear quadratic regulator control, Kalman filter, H_2 control, developed in 1960s to achieve certain optimal performance and boomed by *NASA Apollo Project*.

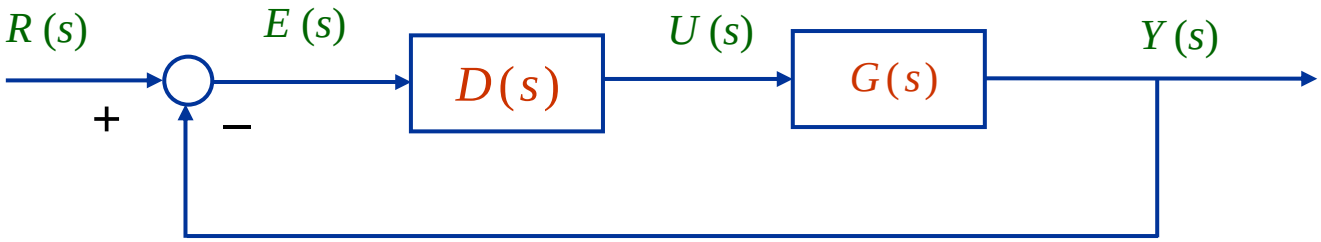
- Robust control: H_∞ control, developed in 1980s & 90s to handle systems with uncertainties and disturbances and with high performances. Example: military systems.

- Nonlinear control: Currently hot research topics, developed to handle nonlinear systems with high performances. Examples: military systems such as aircraft, missiles.

- Intelligent control: Knowledge-based control, adaptive control, neural and fuzzy control, etc., researched heavily in 1990s, developed to handle systems with unknown models. Examples: economic systems, social systems, human systems.

1.9. Classical feedback control in continuous setting

Let us examine the following block diagram of control system:



Recall that the objective of control system design is trying to match the output $Y(s)$ to the reference $R(s)$. Thus, it is important to find the relationship between them. Recall that

$$G(s) = \frac{Y(s)}{U(s)} \quad \Rightarrow \quad Y(s) = G(s)U(s)$$

Similarly, we have $U(s) = D(s)E(s)$, and $E(s) = R(s) - Y(s)$. Thus,

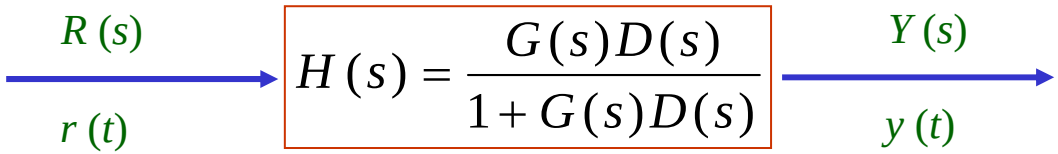
$$Y(s) = G(s)U(s) = G(s)D(s)E(s) = G(s)D(s)\left[R(s) - Y(s)\right]$$

$$Y(s) = G(s)D(s)R(s) - G(s)D(s)Y(s) \;\Rightarrow\; \left[1 + G(s)D(s)\right]Y(s) = G(s)D(s)R(s)$$

$$\Rightarrow \quad H(s) = \frac{Y(s)}{R(s)} = \frac{G(s)D(s)}{1 + G(s)D(s)}$$

⟸ Closed-loop transfer function from R to Y.

Thus, the block diagram of the control system can be simplified as,



The problem becomes how to choose an appropriate $D(s)$ such that $H(s)$ will have desired properties. Furthermore, we have

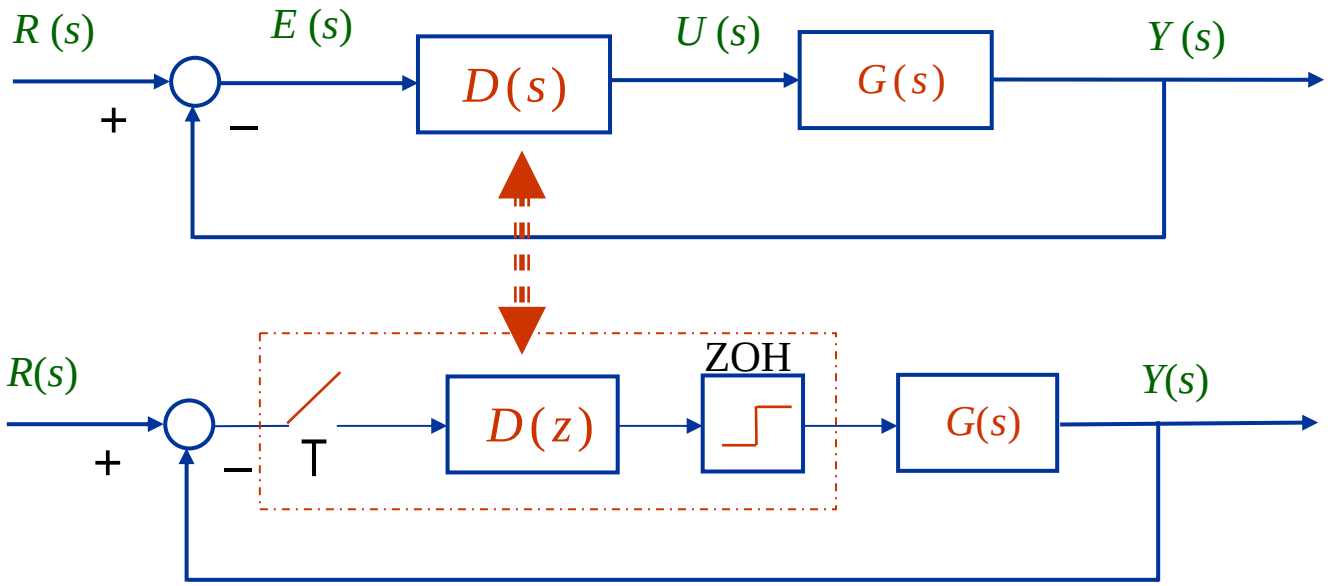
$$E(s) = R(s) - Y(s) = R(s) - H(s)R(s) = \left[1 - \frac{G(s)D(s)}{1+G(s)D(s)}\right]R(s) = \frac{R(s)}{1+G(s)D(s)}$$

The problem is equivalent to finding an appropriate control law $D(s)$ such that the resulting error function $e(t)$ goes to zero as quick and as smooth as possible, which is the same as saying that the output $y(t)$ is tracking the given reference $r(t)$ as quick and as smooth as possible. In our class, we will mainly focus on the cases when $r(t)$ is either a step function or a ramp function.

1.10. Design a digital controller — from continuous to digital

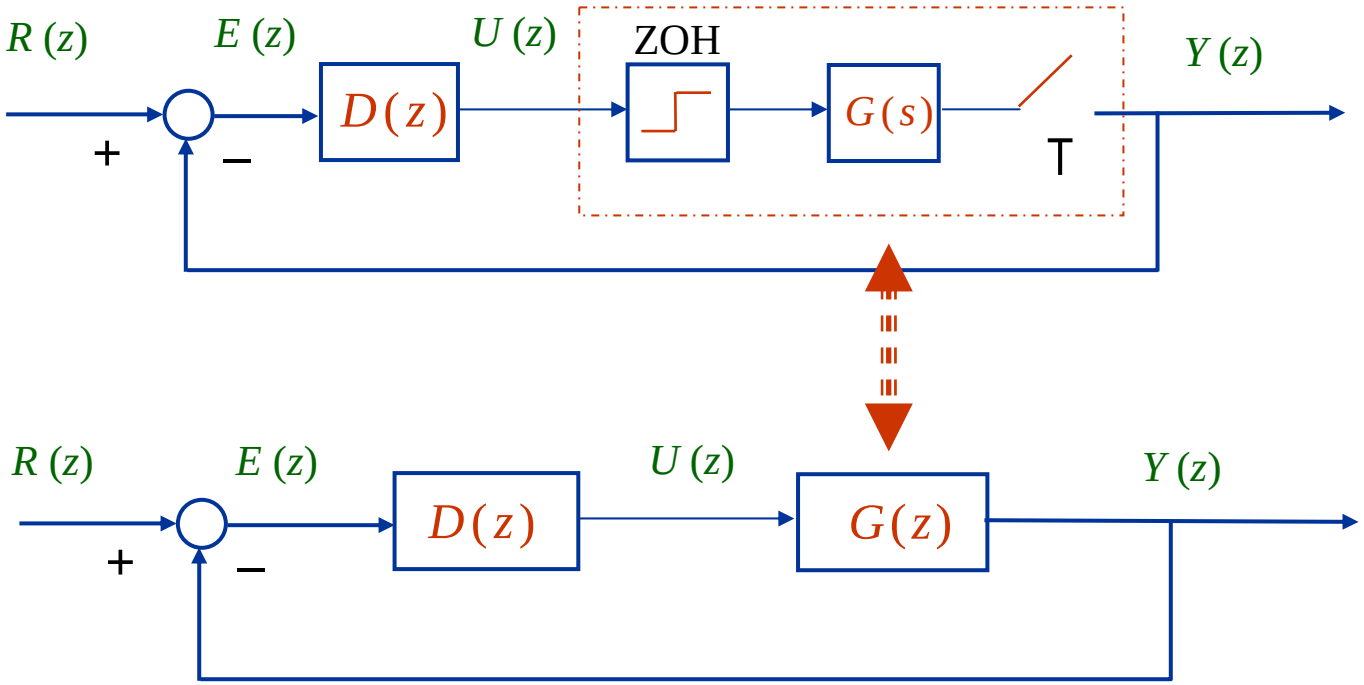
There are two ways to design a digital controller or a discrete-time control system:

- Follow whatever we have learnt in EE2010 or EE2131 to design a continuous-time controller and then discretize it using ZOH or bilinear transformation or any discretization technique to obtain an equivalent digital controller.



The above design works very well if sampling period T is sufficiently small.

- Alternatively, one could discretize the plant first to obtain a sampled-data system or discrete-time system and then apply digital control system design techniques to design a digital controller:



It should be shown in Part 1: $G(z) = (1 - z^{-1})\mathbf{Z}\left\{\frac{G(s)}{s}\right\}$ (Show this as an exercise !)

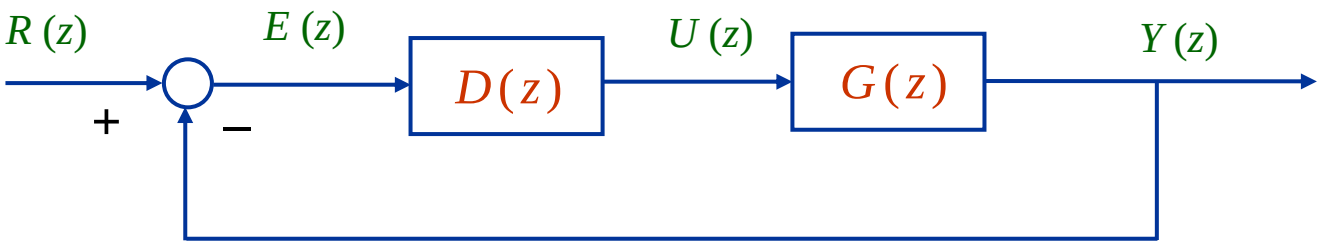
1.11. Why digital control?

- Control systems in continuous-time setting in the past are usually implemented using analogue devices such resistors, capacitors, inductors and operational amplifiers together with some necessary mechanical components. These devices are neither economical nor durable. The advances in computer technologies make the implementation of control systems in discrete-time setting (i.e., digital controllers) much more efficiently and economically possible.

- Most of control systems nowadays are implemented using either computers such as PCs or Digital Signal Processes (DSP), which are specially designed to carry out computations related to control algorithm realizations. The advantages of digital controllers using PC or DSP are obvious — it is fast, reliable, reusable and can be modified thru simple recoding whenever needed.

1.12. Feedback control in discrete setting

Let us examine the following block diagram of control system:



Note that

$$G(z) = \frac{Y(z)}{U(z)} \quad \Rightarrow \quad Y(z) = G(z)U(z)$$

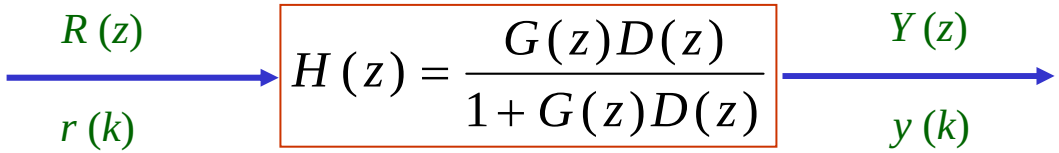
Similarly, we have $U(z) = D(z)E(z)$, and $E(z) = R(z) - Y(z)$. Thus,

$$Y(z) = G(z)U(z) = G(z)D(z)E(z) = G(z)D(z)\left[R(z) - Y(z)\right]$$

$$Y(z) = G(z)D(z)R(z) - G(z)D(z)Y(z) \;\Rightarrow\; \left[1 + G(z)D(z)\right]Y(z) = G(z)D(z)R(z)$$

$$\Rightarrow \quad \boxed{H(z) = \frac{Y(z)}{R(z)} = \frac{G(z)D(z)}{1 + G(z)D(z)}} \quad \Leftarrow \text{Closed-loop transfer function from } R \text{ to } Y.$$

Thus, the block diagram of the control system can be simplified as,



The problem becomes how to choose an appropriate $D(z)$ such that $H(z)$ will yield desired properties. As in the continuous-time case, we have

$$E(z) = R(z) - Y(z) = R(z) - H(z)R(z) = \left[1 - \frac{G(z)D(z)}{1+G(z)D(z)}\right]R(z) = \frac{R(z)}{1+G(z)D(z)}$$

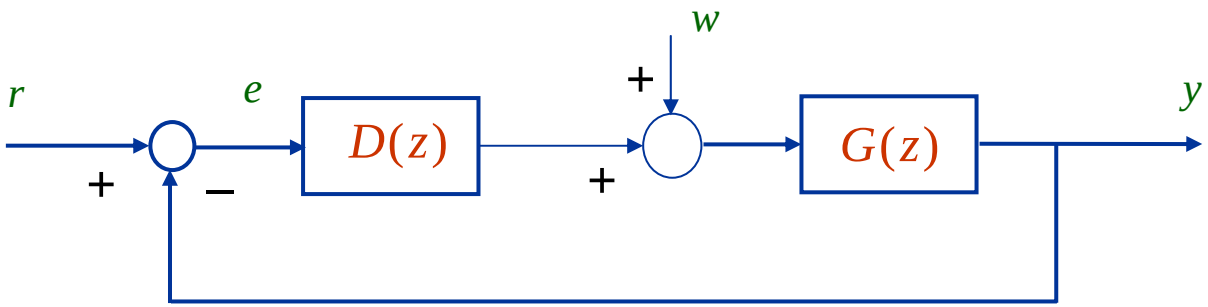
The problem is equivalent to finding an appropriate control law $D(z)$ such that the resulting error function $e(k)$ goes to zero as quick and as smooth as possible, which is the same as saying that the output $y(k)$ is tracking the given reference $r(k)$ as quick and as smooth as possible. In our class, we will mainly focus on the cases when $r(k)$ is either a step function or a ramp function.

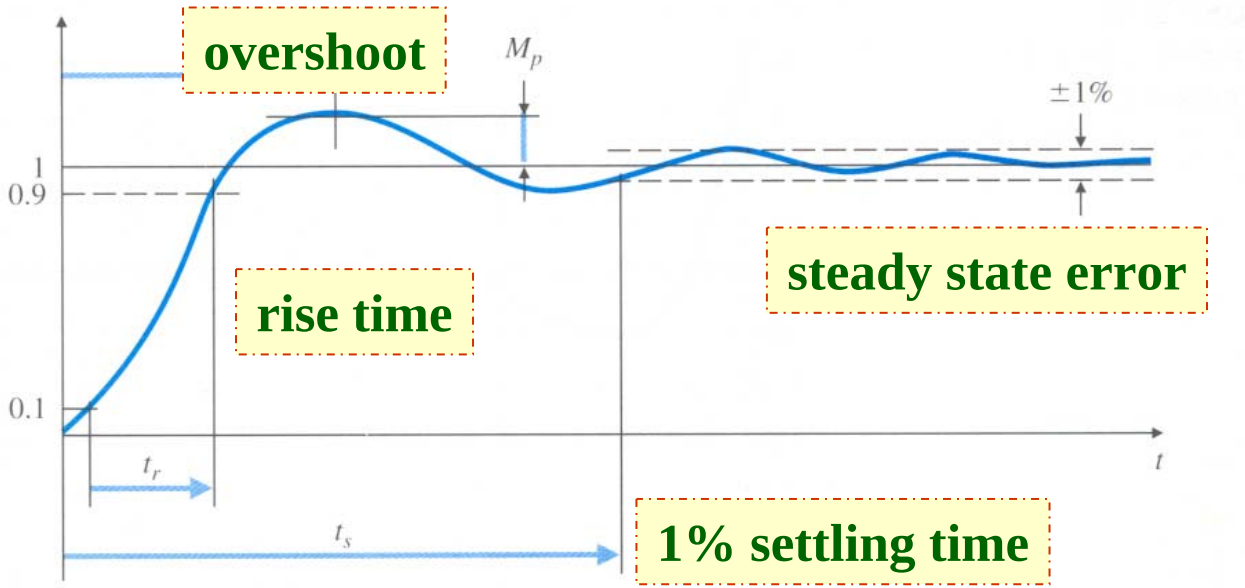
2. Design Specifications

2.1. Time domain design specifications:

- steady state accuracy
- stability
- rise time
- overshoot
- settling time
- disturbance rejection
- others

Recall the unit feedback control system with disturbance w,





2.2. Steady state accuracy — Continuous systems

Case 1: If $r(t)$ is a unit step, $R(s) = \frac{1}{s}$, by the final value theorem of Laplace Transform

$$e(\infty) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} s \cdot \frac{1}{s} \cdot \frac{1}{1+G(s)D(s)} = \frac{1}{1+G(0)D(0)} = \frac{1}{1+K_p}$$

proportional error constant

Obviously, we need the proportional error constant to be infinity, which implies $G(s)D(s)$ has at least a factor of $1/s$ (TYPE I SYSTEM) in order to make $e(\infty)$ zero. If K_p is a finite scalar, the open-loop system $G(s)D(s)$ is said to be a TYPE 0 system.

Case 2: If $r(t)$ is a unit ramp, $R(s) = \frac{1}{s^2}$, by the final value theorem

$$e(\infty) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} s \cdot \frac{1}{s^2} \cdot \frac{1}{1+G(s)D(s)} = \frac{1}{\lim_{s\to 0} sG(s)D(s)} = \frac{1}{K_v}$$

velocity error constant

Similarly, we need the velocity error constant to be infinity, which implies $G(s)\ D(s)$ has at least a factor of $1/s^2$ (TYPE II SYSTEM) in order to make $e(\infty)$ zero.

2.3. Steady state accuracy — Discrete systems

Case 1: If $r(k)$ is a unit step, $R(z) = \dfrac{z}{z-1}$, by the final value theorem of z-Transform

$$e(\infty) = \lim_{z\to 1}(z-1)E(z) = \lim_{z\to 1}(z-1)\cdot\frac{z}{z-1}\cdot\frac{1}{1+G(z)D(z)} = \frac{1}{1+G(1)D(1)} \stackrel{\Delta}{=} \frac{1}{1+K_p}$$

Similarly, as in the continuous-time case we need the *proportional error constant* $K_p = \infty$, i.e., $G(z)D(z)$ has at least a factor of $1/(z-1)$ (TYPE I SYSTEM) in order to make $e(\infty)$ zero. If K_p is a finite scalar, the open-loop system $G(z)D(z)$ is said to be a TYPE 0 system.

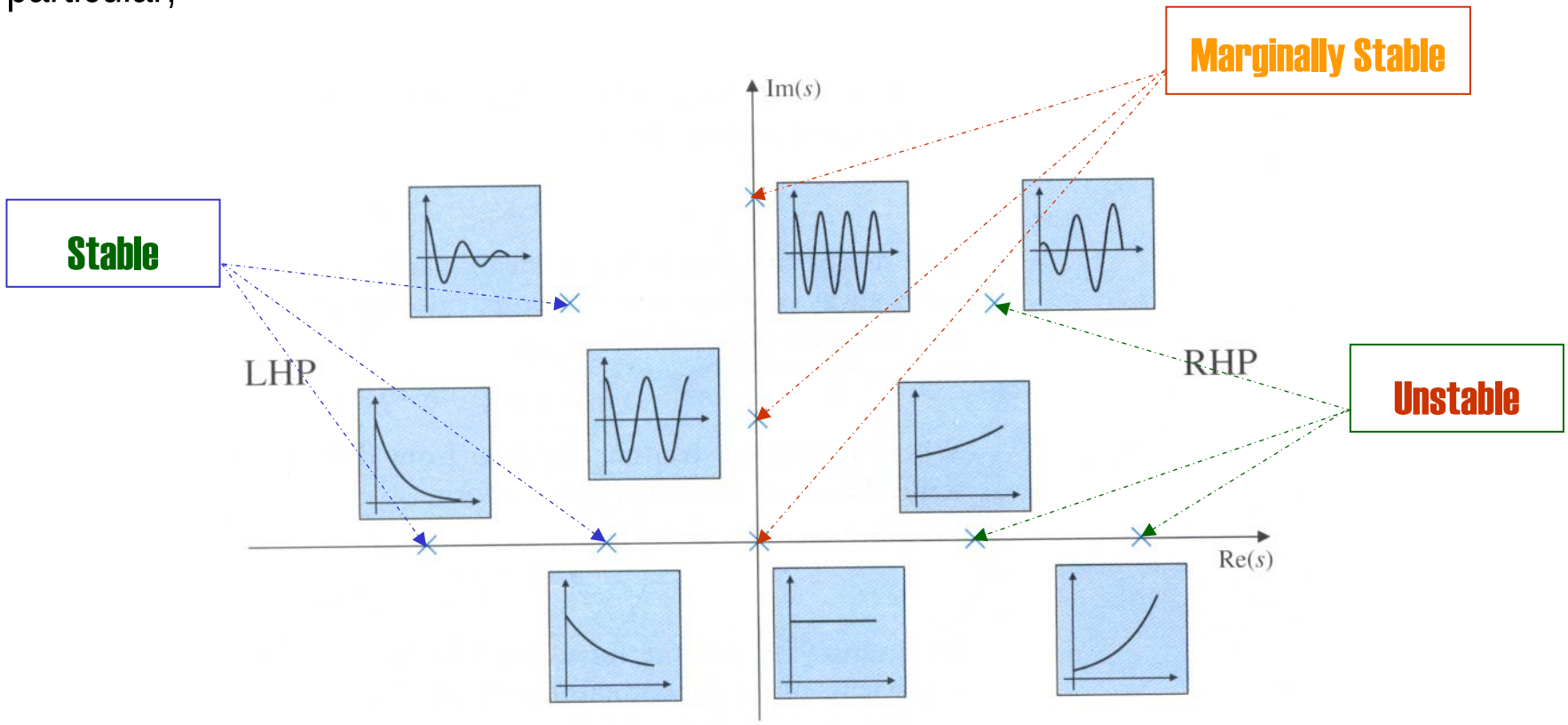
Case 2: If $r(k)$ is a unit ramp, $R(z) = \dfrac{Tz}{(z-1)^2}$, by the final value theorem

$$e(\infty) = \lim_{z\to 1}(z-1)\cdot\frac{Tz}{(z-1)^2}\cdot\frac{1}{1+G(z)D(z)} = \frac{T}{\lim_{z\to 1}(z-1)G(z)D(z)} = \frac{1}{K_v}$$

Similarly, we need the *velocity error constant* $K_v = \infty$, which implies $G(z)D(z)$ has at least a factor of $1/(z-1)^2$ (TYPE II SYSTEM) in order to make $e(\infty)$ zero.

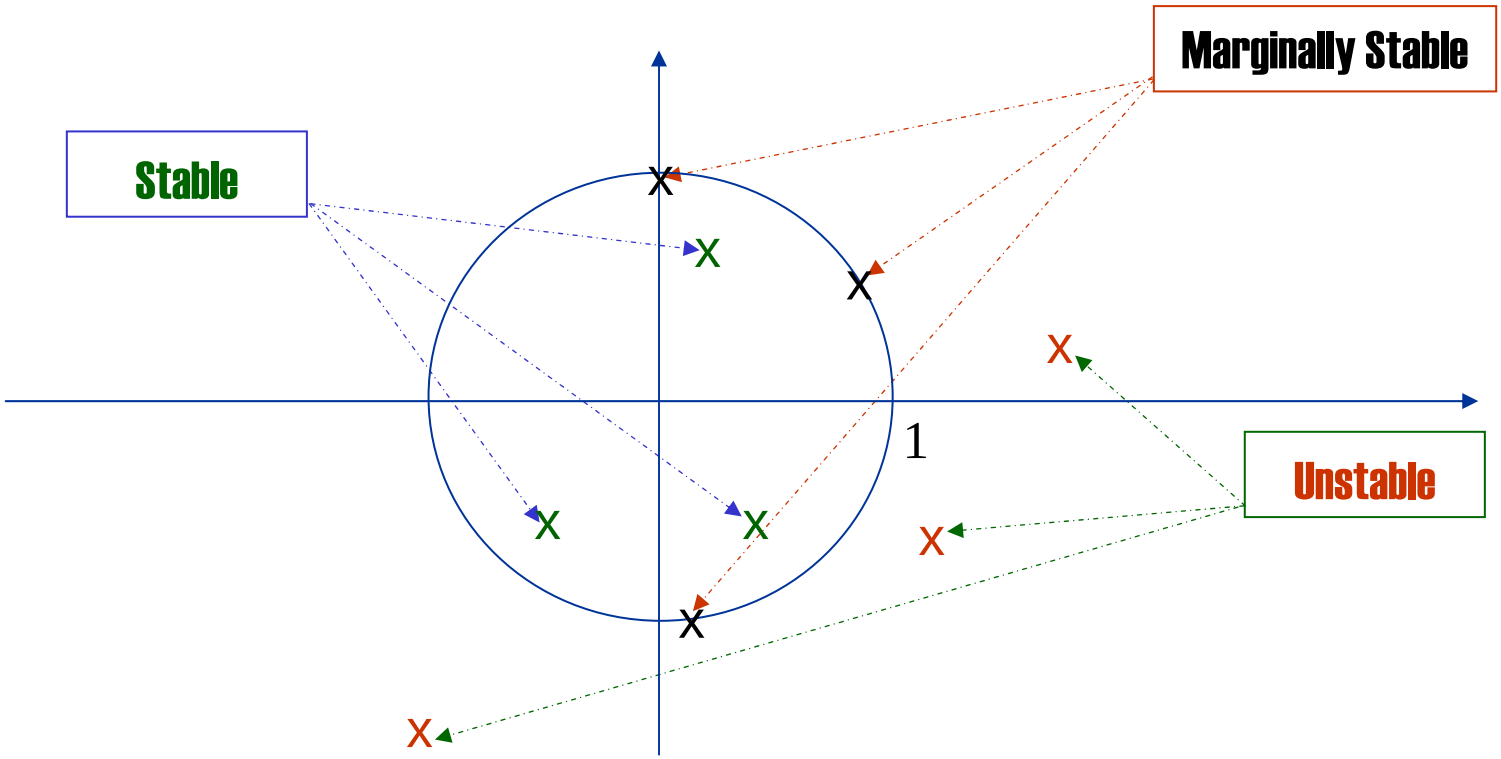
2.4. Stability — Continuous systems

A continuous-time system is said to be stable if its denominator of the system has no roots or poles with a positive real part. It is unstable if it has poles or roots with a positive real part. In particular,



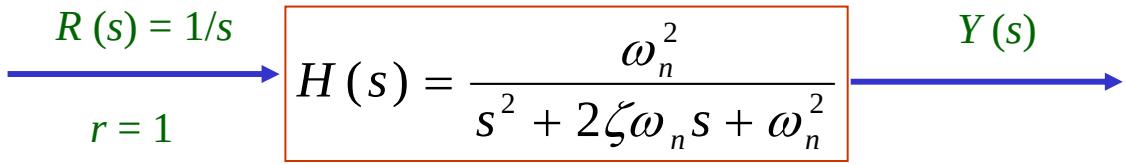
2.5. Stability — Discrete systems

A discrete-time system is said to be stable if its denominator of the system has no roots or poles outside unit circle. It is unstable if it has poles or roots outside unit circle. In particular,

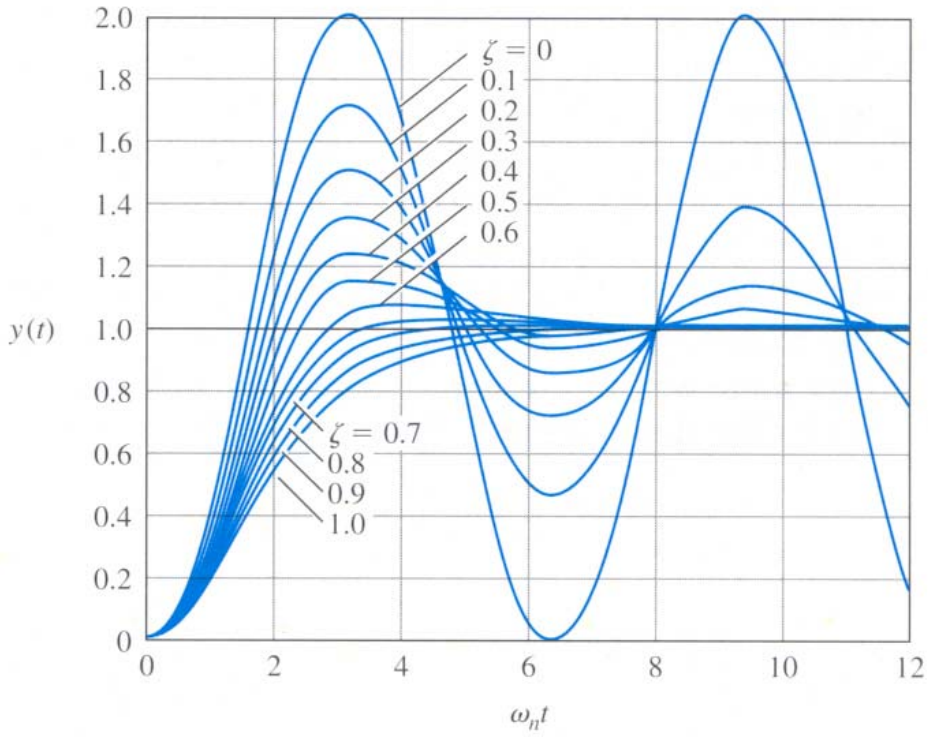


2.6. Behavior of continuous 2nd order systems with unit step input

Consider the following block diagram with a standard 2nd order system,

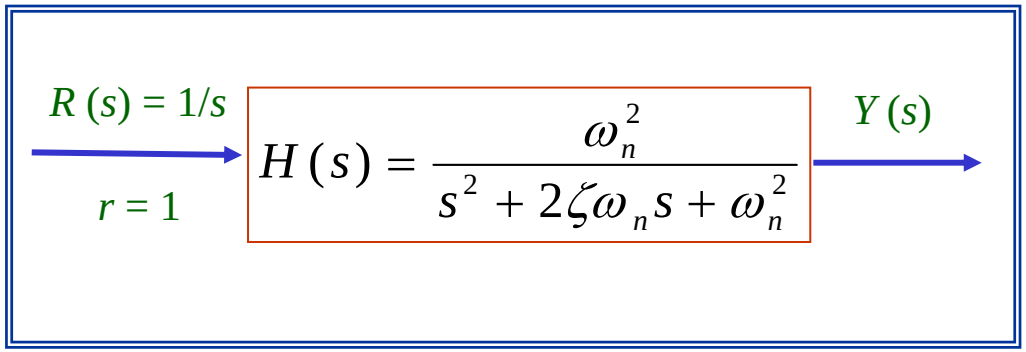


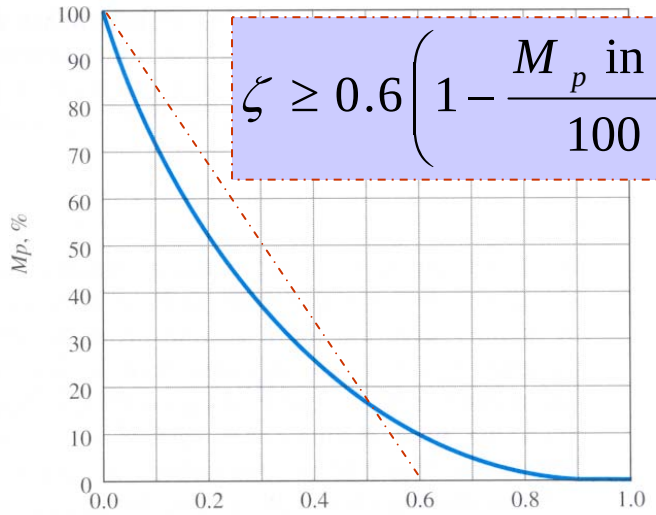
The behavior of the system is as follows:



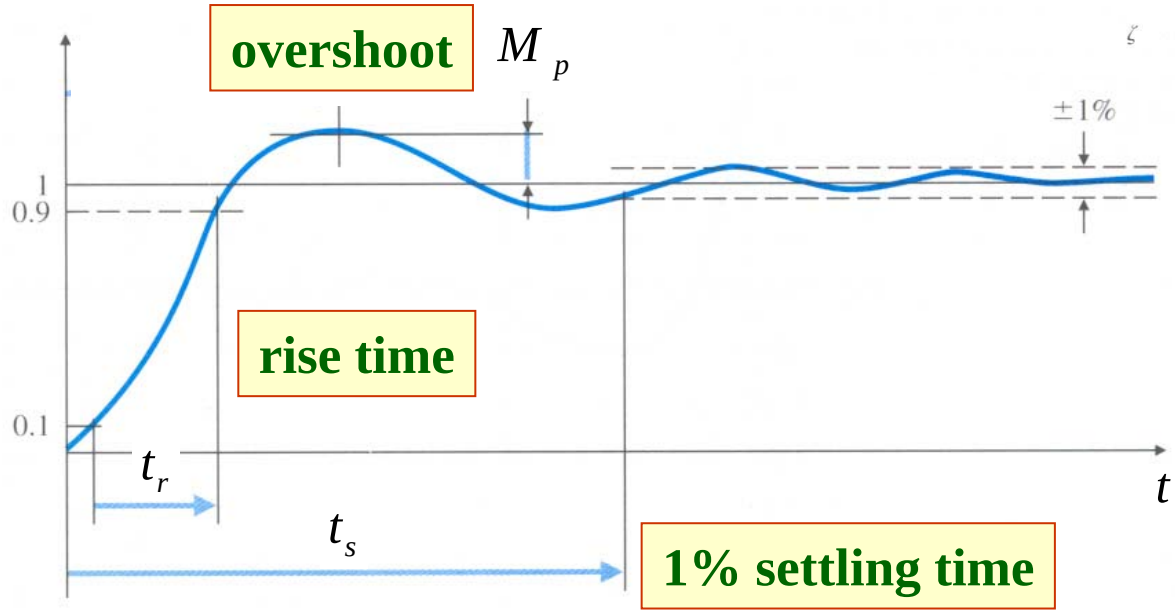
The behavior of the system is fully characterized by ζ, which is called the *damping ratio*, and ω_n, which is called the *natural frequency*.

2.7. Rise time, overshoot and settling time — Continuous systems





$$\zeta \geq 0.6\left(1 - \frac{M_p \text{ in } \%}{100}\right)$$



$$t_r \cong \frac{1.8}{\omega_n}$$

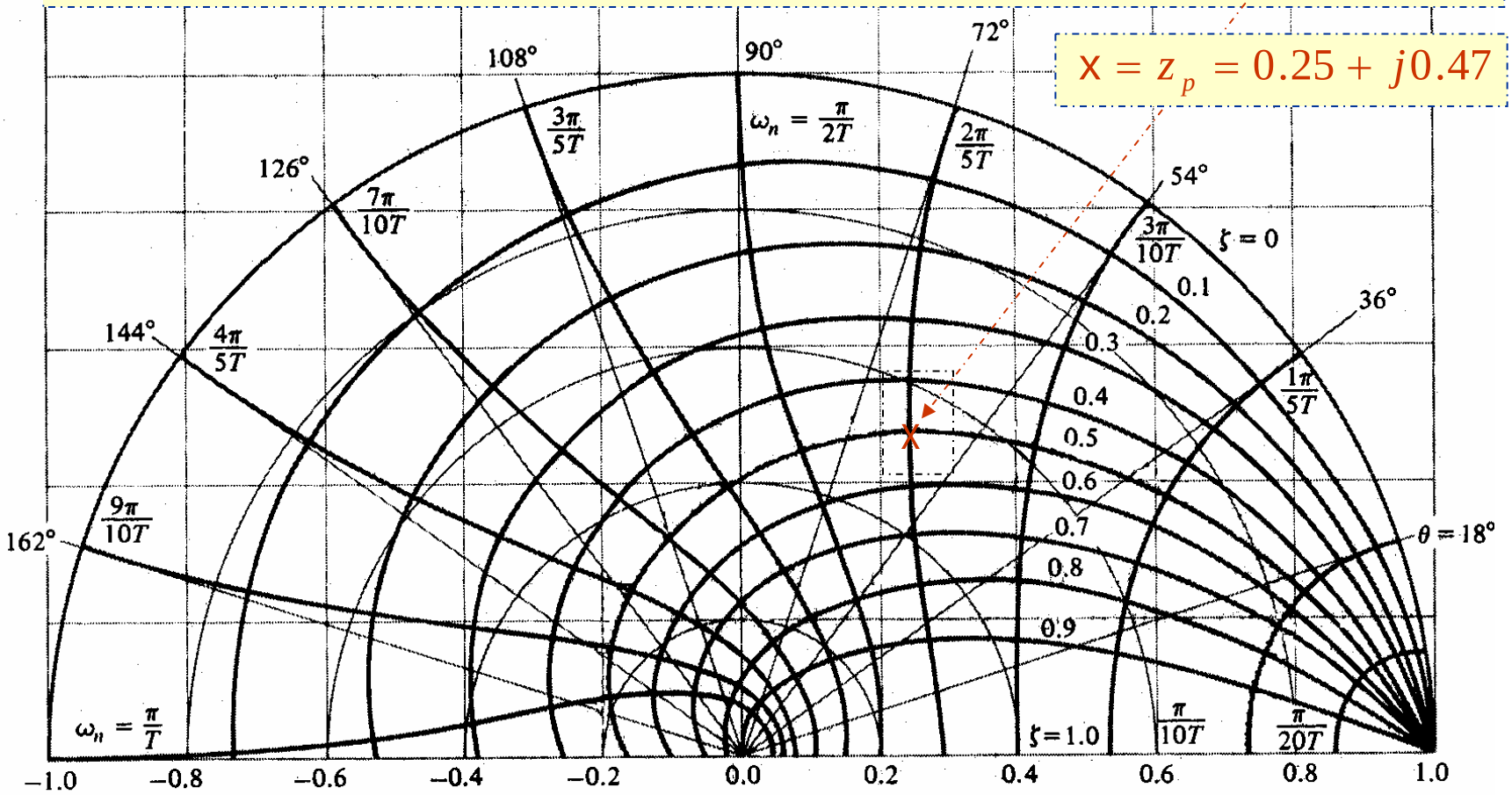
1% settling time

$$t_s \cong \frac{4.6}{\zeta\omega_n}$$

2.8. Rise time, overshoot and settling time — Discrete systems

Example : $M_p = 18\%,\ t_s = 7.3\,\text{sec} \Rightarrow \omega_n = 1.256,\ \zeta = 0.5$ with $T = 1$

$x = z_p = 0.25 + j0.47$



z = plane loci of roots of constant ζ and ω_n

$s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$

$z = e^{Ts}$

T = sampling period

$r(k) = 1 \rightarrow$

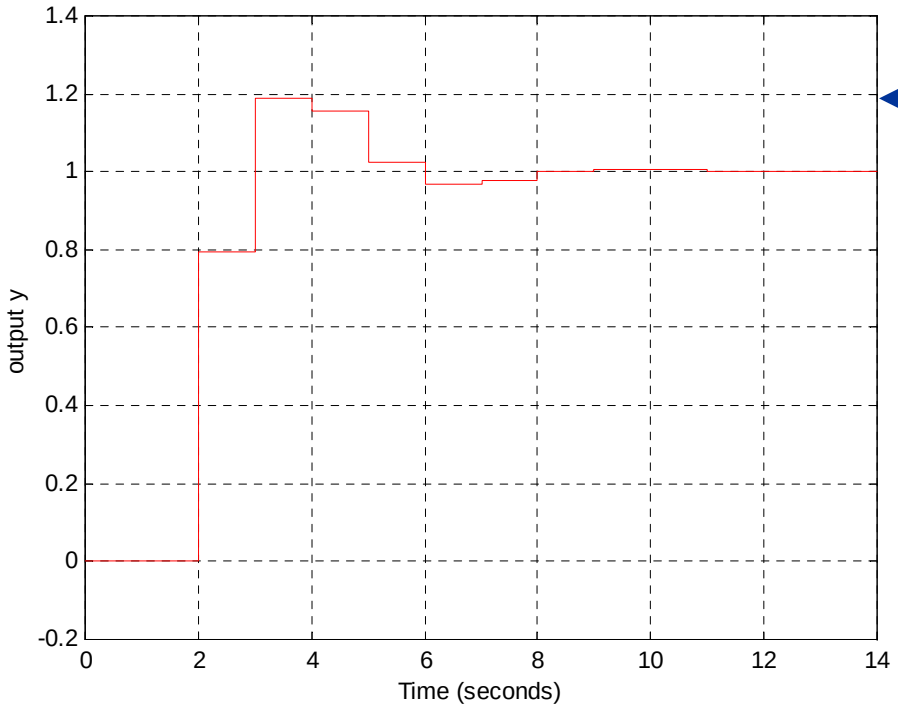
$$H(z) = \frac{(1 - z_p)(1 - \bar{z}_p)}{(z - z_p)(z - \bar{z}_p)}$$

$\rightarrow y(k)$

2.9. Rise time, overshoot and settling time — Discrete systems

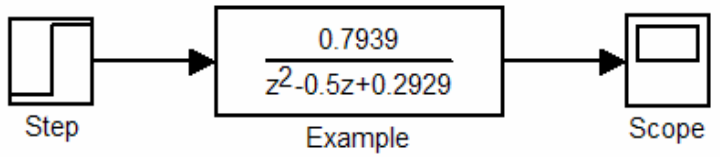
Let us verify the example given on the previous page using MATLAB SIMULATION to check whether the resulting discrete-time system indeed produces pre-specified overshoot and settling time or not. The resulting discrete-time system is given by

$$H(z)=\frac{(1-0.25-j0.48)(1-0.25+j0.48)}{(z-0.25-j0.48)(z-0.25+j0.48)}=\frac{0.7929}{z^2-0.5z+0.2929}$$



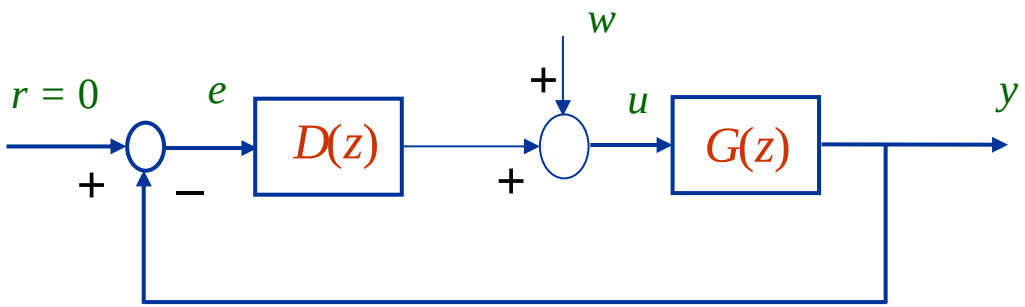
The system output to a step input

- Simulation block in SIMULINK



2.10. Disturbance rejection

Recall the block diagram below. We set the reference $r = 0$ for simplicity and for the study of the effects of the disturbance w (unwanted signal) on the system output.



Exercise: Show that the output and the disturbance are related as follows:

$$Y(z) = \frac{G(z)}{1 + D(z)G(z)} W(z)$$

where $Y(z)$ and $W(z)$ are respectively z-transforms of y and w.

Consider the case when w is a unit step function. We have

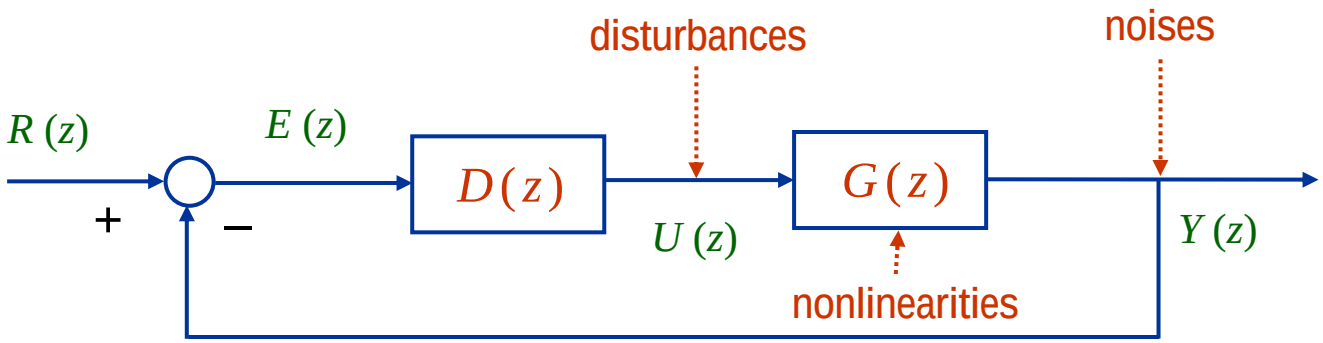
$$y(\infty) = \lim_{z \to 1} (z-1) Y(z)$$

$$= \lim_{z \to 1} (z-1) \cdot \frac{z}{z-1} \cdot \frac{G(z)}{1 + D(z)G(z)}$$

$$= \frac{G(1)}{1 + D(1)G(1)}$$

In order to make $y(\infty) = 0$, we need either $G(1) = 0$, which is impractical (why?), or $D(1) = \infty$, i.e., it has a factor $1/(z-1)$.

2.11. Other considerations



- Plant nonlinearities — nonlinear plants — nonlinear control
- Plant parameter uncertainties — robust control
- Control input nonlinearities (control saturations) — nonlinear control
- Sensor nonlinearities — nonlinear control
- Noise rejection — robust control, optimal control

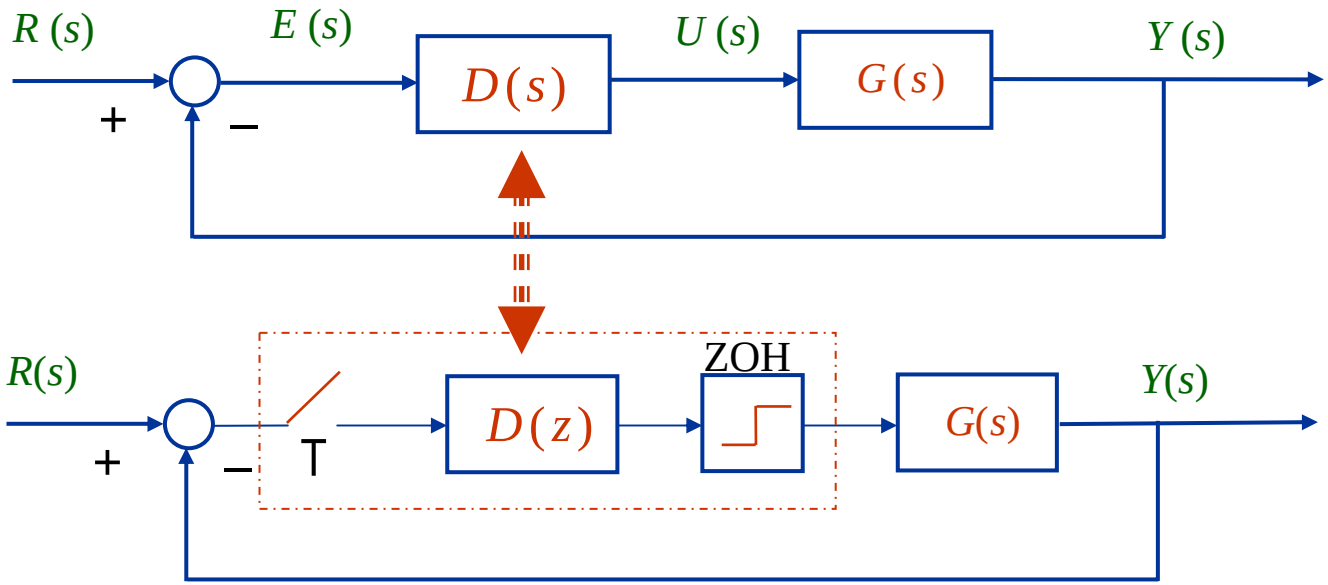
These issues are too complicated to be considered in this third year level course...

Come back for postgraduate studies if you are interested in these topics.

3. Time Domain Design

3.1. Digital control system design using emulation

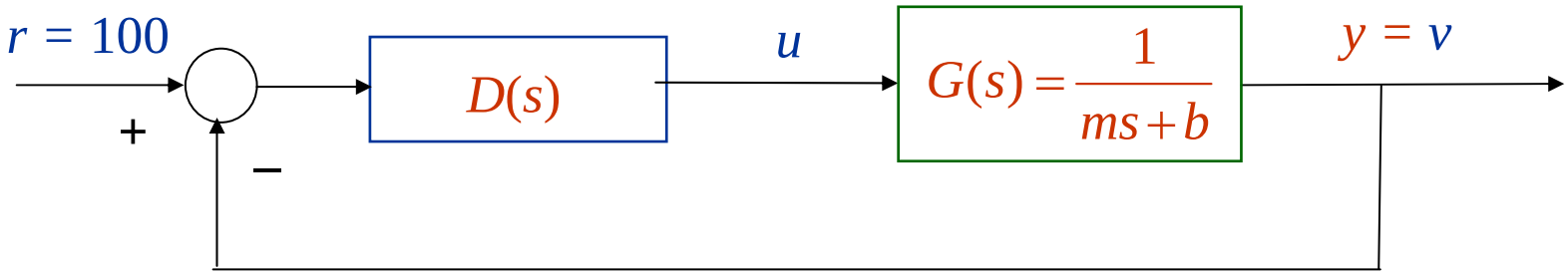
In this approach, we design a continuous-time controller that meets all design specifications and then discretize it using a bilinear transformation or other discretization technique to obtain an equivalent digital controller.



This method works if the sampling rate is 30 times faster than the system bandwidth. Further refinement is necessary for the case where the sampling rate is 6 times the bandwidth.

3.2. Design example:

Consider a car (BMW), which has a weight $m = 1000\ \text{kg}$. Assuming the average friction coefficient $b = 100$, design a cruise control system such that the car can reach $100\ \text{km/h}$ from $0\ \text{km/h}$ in $8\ \text{s}$ with an overshoot less 20%.

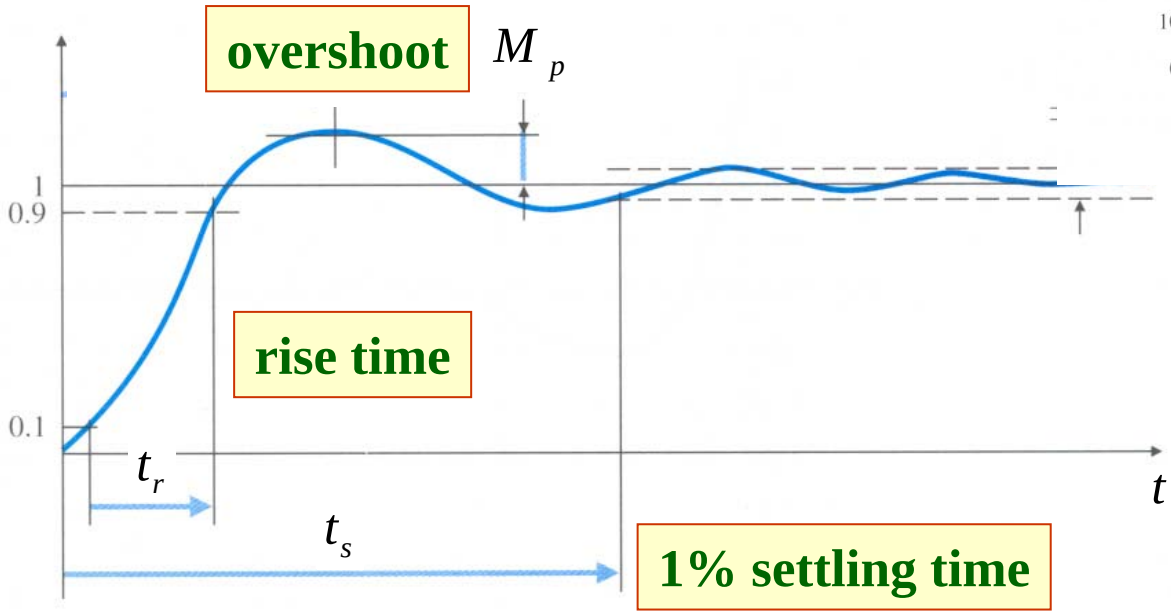


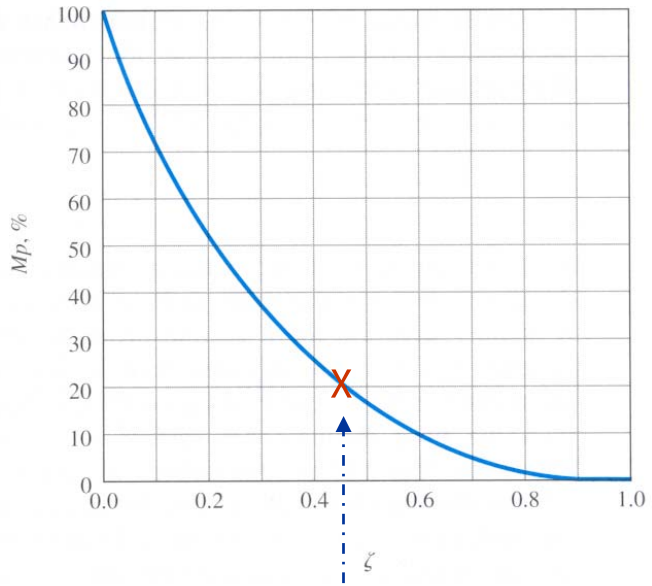
Let us try a PI controller, i.e., $D(s) = k_p + \dfrac{k_i}{s}$. The first component is proportional to the error and the second consists of an integration action. Following results derived earlier, we have

$$H(s) = \frac{Y(s)}{R(s)} = \frac{G(s)D(s)}{1 + G(s)D(s)} = \frac{0.001k_p s + 0.001k_i}{s^2 + (0.1 + 0.001k_p)s + 0.001k_i} \qquad (*)$$

3.3. Deriving ζ and ω_n from the design specifications

$R(s) = 1/s$ (r = 1) $\rightarrow$ $H(s) = \dfrac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$ $\rightarrow$ $Y(s)$





$\zeta \geq 0.46 \quad \Rightarrow \quad \zeta = 0.7$

$$t_s \cong \frac{4.6}{\zeta\omega_n} \quad \Rightarrow$$

$$\omega_n \cong \frac{4.6}{\zeta t_s} = \frac{4.6}{0.7 \times 8} = 0.82$$

$\Rightarrow$ desired closed-loop transfer function $H_{\text{desired}}(s) = \dfrac{0.67}{s^2 + 1.15s + 0.67}$

3.4. Calculating desired controller parameters

Recall from (*) the closed-loop transfer function of the cruise control system with the PI control law, i.e.,

$$H(s)=\frac{Y(s)}{R(s)}=\frac{G(s)D(s)}{1+G(s)D(s)}=\frac{0.001k_p s+0.001k_i}{s^2+(0.1+0.001k_p)s+0.001k_i}$$

and the desired transfer function that produces desired performance

$$H_{\text{desired}}(s)=\frac{0.67}{s^2+1.15s+0.67}$$

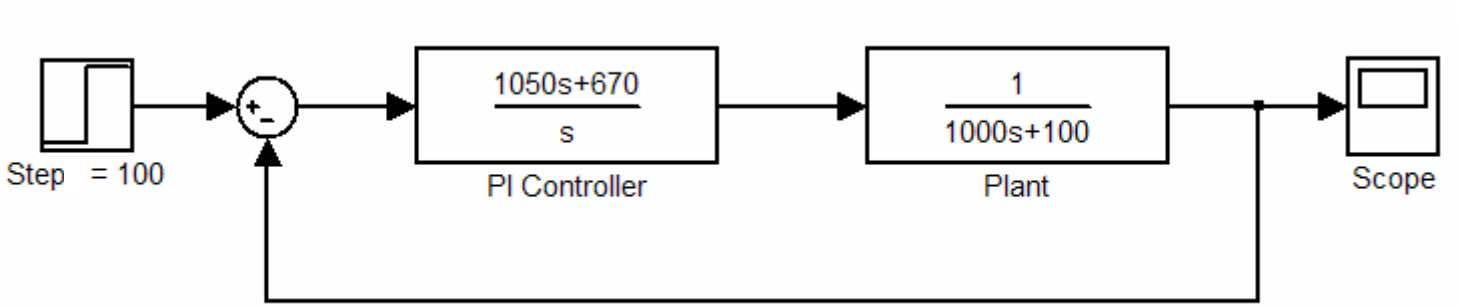
Comparing the coefficients on the denominators, we have

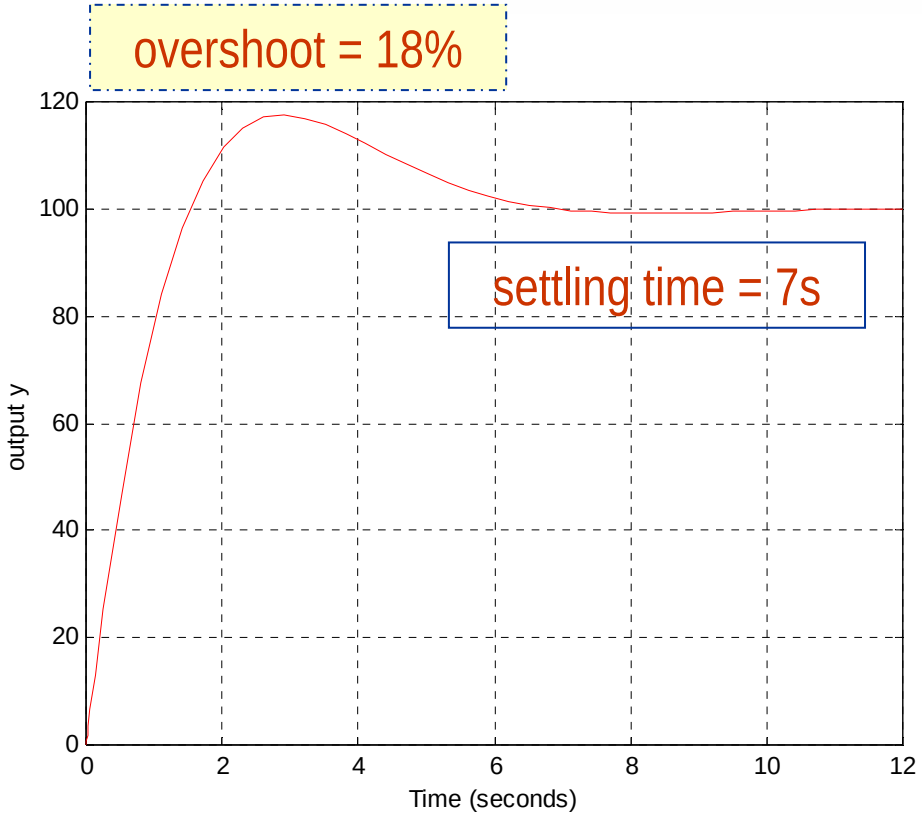
$$0.1+0.001k_p=1.15 \quad \Rightarrow \quad k_p=1050$$

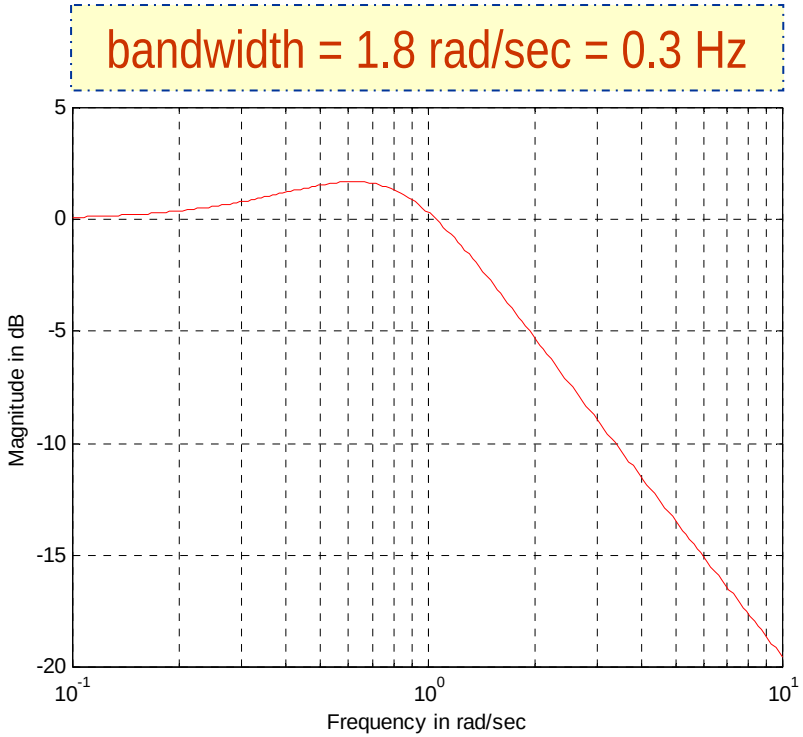
$$0.001k_i=0.67 \quad \Rightarrow \quad k_i=670$$

$$\text{and the resulting closed-loop transfer function } H(s)=\frac{1.05s+0.67}{s^2+1.15s+0.67}$$

3.5. Verification through SIMULINK







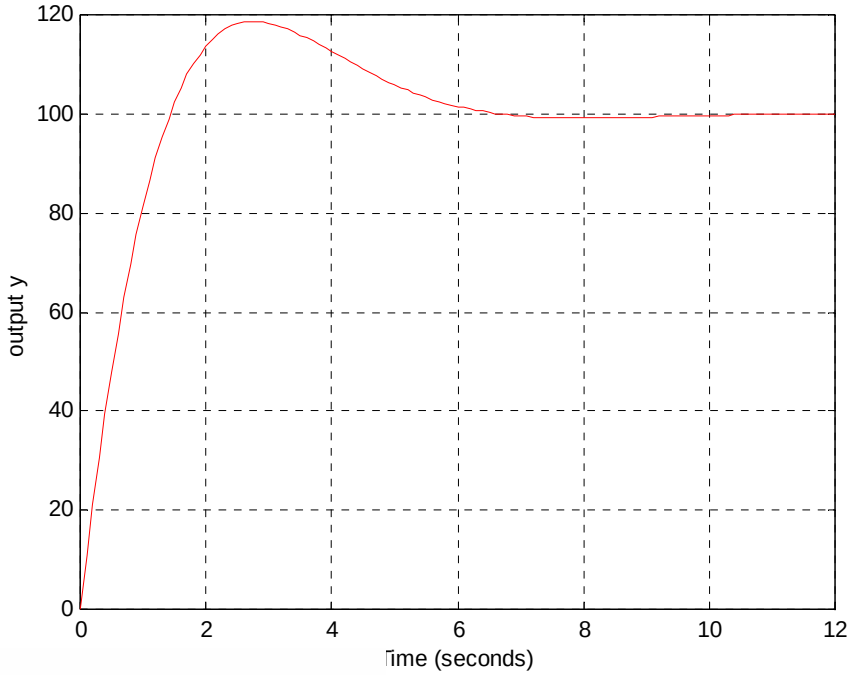
3.6. Digital controller with a sampling rate 30 times the bandwidth

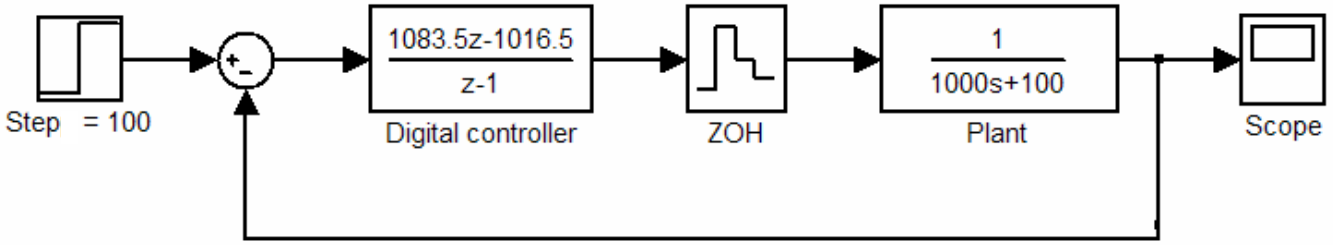
We first discretize the continuous-time PI control law with $T = 1/(30\times0.3) \approx 0.1$ seconds using a bilinear transformation method, i.e.,

$$D(z) = D(s)\Big|_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)} = \frac{1050\,s + 670}{s}\Bigg|_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)}$$

$$= \frac{1050\,\dfrac{2}{T}\left(\dfrac{z-1}{z+1}\right) + 670}{\dfrac{2}{T}\left(\dfrac{z-1}{z+1}\right)}$$

$$= \frac{1083.5z - 1016.5}{z-1}$$





Performance is about the same as the continuous-time case.

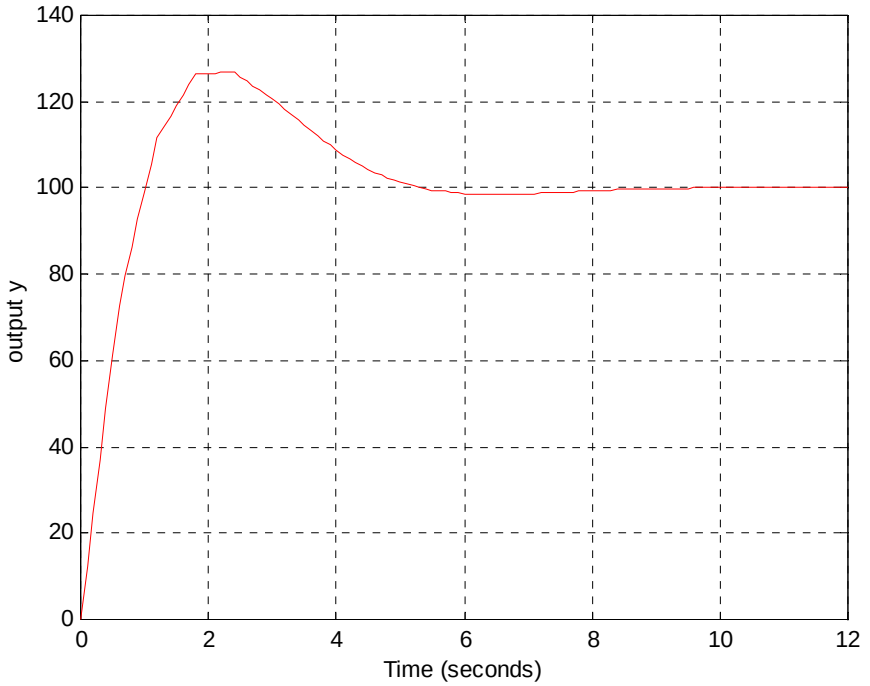
3.7. Digital controller with a sampling rate 6 times the bandwidth

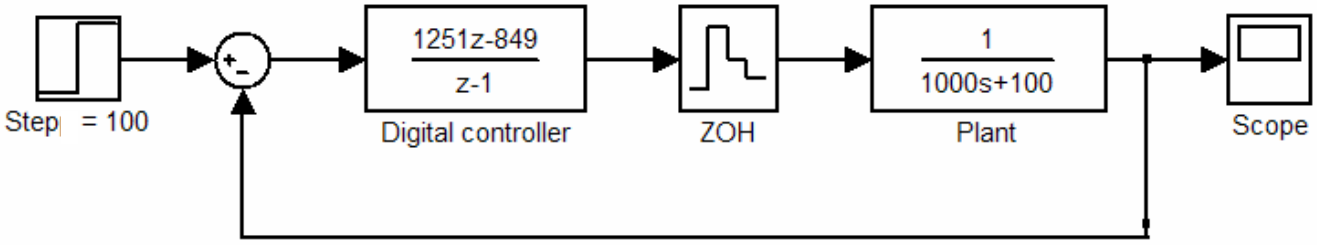
We now discretize the continuous-time PI control law with $T = 1/(6\times 0.3) \approx 0.6$ seconds using the bilinear transformation method, i.e.,

$$D(z) = D(s)\Big|_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)} = \frac{1050\,s + 670}{s}\Bigg|_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)}$$

$$= \frac{1050 + \dfrac{2}{0.6}\left(\dfrac{z-1}{z+1}\right) + 670}{\dfrac{2}{0.6}\left(\dfrac{z-1}{z+1}\right)}$$

$$= \frac{1251z - 849}{z-1}$$

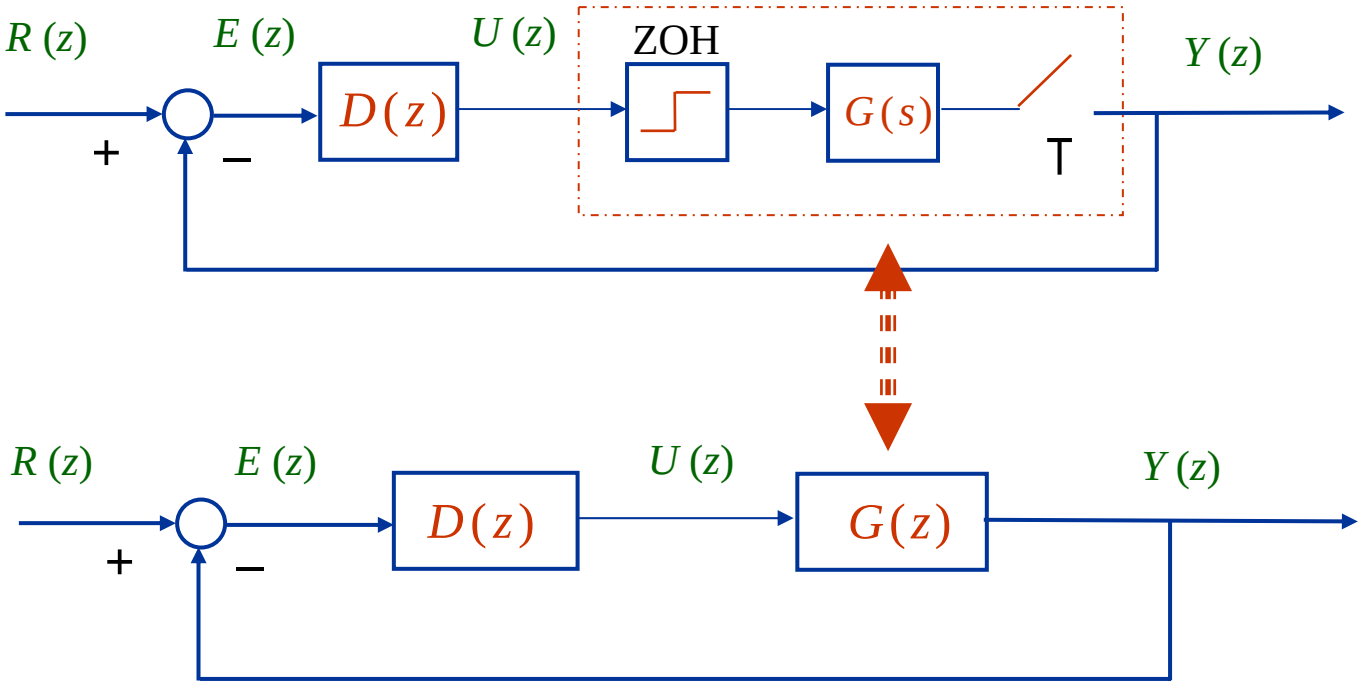




Performance is not as good as the continuous-time case.

3.8. Digital PID control system design via pole placement technique

In this approach, we discretize the continuous-time plant first or directly work on a discrete-time plant to design a digital controller using the well known PID framework.



where $G(z) = (1 - z^{-1})\mathbf{Z}\left\{\frac{G(s)}{s}\right\}$ and $D(z)$ is taken to be a PID controller.

3.9. PID Control

PID control is widely used in process control and most of industrial control systems. Unknown source reports that more than 90% of industrial processes are actually controlled by PID type of controllers. PID control consists of three essential components, namely, P (proportional control), I (integral control) and D (derivative control).

Proportional Control

A discrete implementation of proportional control is identical to continuous. The continuous is

$$u(t) = k_p e(t) \quad \Rightarrow \quad D(s) = k_p$$

The discrete is

$$u(k) = k_p e(k) \quad \Rightarrow \quad D(z) = k_p$$

where $e(t)$ or $e(k)$ is the error signal as given in the feedback block diagram.

Derivative Control

The continuous derivative control is

$$u(t) = k_d \dot{e}(t) \quad \Rightarrow \quad D(s) = k_d s$$

The discrete derivative control is

$$u(k) = k_d \frac{e(k) - e(k-1)}{T} \quad \Rightarrow \quad D(z) = k_d \frac{1 - z^{-1}}{T} = k_d \frac{z-1}{Tz}$$

Integral Control

The continuous integral control is $u(t) = k_i \int_{t_0}^{t} e(t)dt \quad \Rightarrow \quad D(s) = k_i \frac{1}{s}$

The discrete integral control is

$$u(k) = u(k-1) + k_i T e(k) \quad \Rightarrow \quad D(z) = \frac{k_i T}{1 - z^{-1}} = \frac{k_i T z}{z-1}$$

Digital PID Control (conventional version)

$$D(z) = k_P + k_I \frac{z}{z-1} + k_D \frac{z-1}{z} = \frac{(k_P + k_I + k_D)z^2 - (k_P + 2k_D)z + k_D}{z(z-1)}$$

Digital PI Control (conventional version)

Digital PI control consists of only P and I actions and is given by

$$D(z) = k_P + k_I \frac{z}{z-1} = \frac{(k_P + k_I)z - k_P}{z-1}$$

Digital PD Control (conventional version)

Digital PD control consists of only P and D actions and is given by

$$D(z) = k_P + k_D \frac{z-1}{z} = \frac{(k_P + k_D)z - k_D}{z}$$

Digital PID Control (via bilinear transformation)

$$D(z)=\left(k_p+\frac{k_i}{s}+k_d s\right)_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)}=k_p+\frac{k_i T(z+1)}{2(z-1)}+\frac{2k_d(z-1)}{T(z+1)}=\frac{\alpha_2 z^2+\alpha_1 z+\alpha_0}{(z-1)(z+1)}$$

where α_0, α_1 and α_2 are design parameters.

Digital PI Control (via bilinear transformation) – the same as the previous version

$$D(z)=\left(k_p+\frac{k_i}{s}\right)_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)}=k_p+\frac{k_i T(z+1)}{2(z-1)}=\frac{\alpha_1 z+\alpha_0}{z-1}$$

Digital PD Control (via bilinear transformation)

$$D(z)=\left(k_p+k_d s\right)_{s=\frac{2}{T}\left(\frac{z-1}{z+1}\right)}=k_p+\frac{2k_d(z-1)}{T(z+1)}=\frac{\alpha_1 z+\alpha_0}{z+1}$$

3.10. Design example:

Consider a car (BMW), which has a weight $m = 1000\ kg$. Assuming the average friction coefficient $b = 100$, design a cruise control system such that the car can reach $100\ km/h$ from $0\ km/h$ in $8\ s$ with an overshoot less 20%.

Assuming the sampling period $T = 0.6$ seconds, design a digital PI controller that achieve the above specifications.

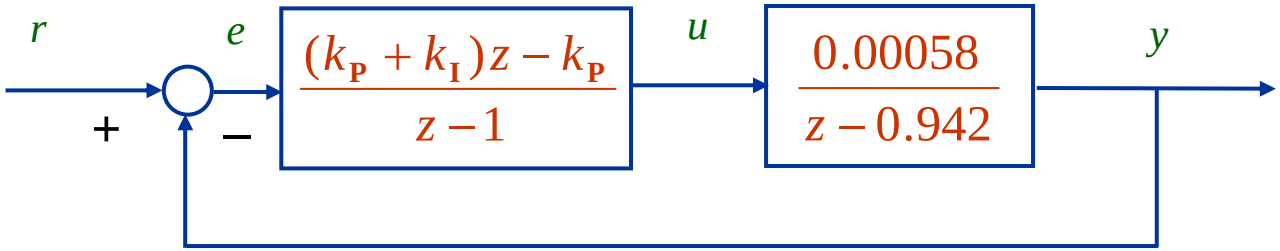
$$G(s) = \frac{1}{ms+b} = \frac{1}{1000s+100} \quad \Rightarrow$$

$$G(z) = (1-z^{-1})\mathbf{Z}\left\{\frac{G(s)}{s}\right\} = (1-z^{-1})\mathbf{Z}\left\{\frac{0.001}{s(s+0.1)}\right\} = \frac{z-1}{z}\mathbf{Z}\left\{\frac{0.01}{s} - \frac{0.01}{s+0.1}\right\}$$

$$= 0.01\cdot\frac{z-1}{z}\cdot\left[\frac{z}{z-1} - \frac{z}{z-e^{-0.1\times 0.6}}\right] = \boxed{\frac{0.00058}{z-0.942}}$$

discretized plant with T = 0.6

3.11. Discretized plant with digital PI controller:



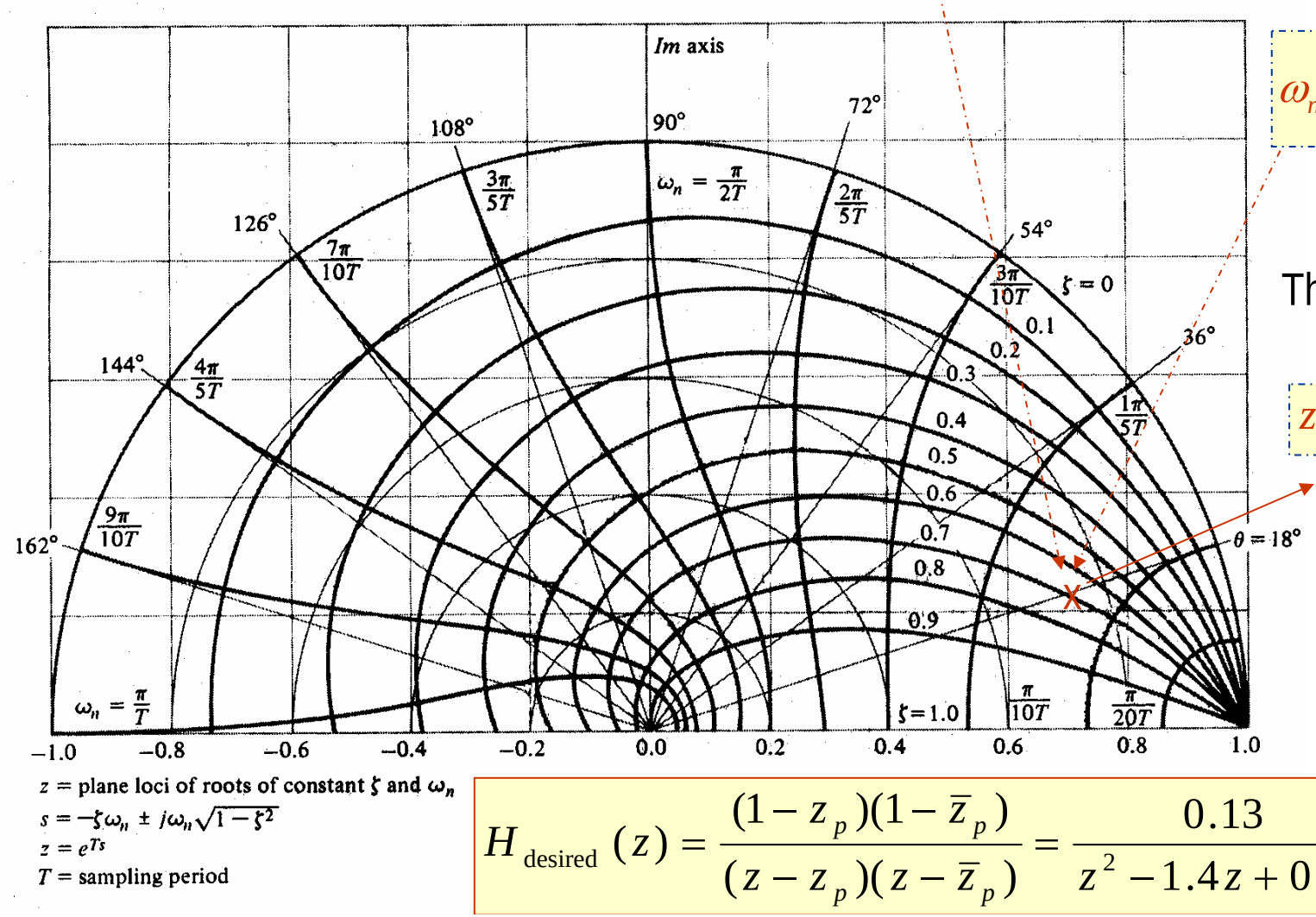
The resulting closed-loop transfer function from r to y is given by

$$H(z) = \frac{G(z)D(z)}{1+G(z)D(z)} = \frac{\dfrac{0.00058}{z-0.942}\cdot\dfrac{(k_P+k_I)z-k_P}{z-1}}{1+\dfrac{0.00058}{z-0.942}\cdot\dfrac{(k_P+k_I)z-k_P}{z-1}}$$

$$= \frac{0.00058\,(k_P+k_I)z - 0.00058\,k_P}{z^2 + [0.00058\,(k_P+k_I) - 1.942\,]z + (0.942 - 0.00058\,k_P)}$$

3.12. Desired closed-loop transfer function:

From the design in 3.3, we obtain the desired $\zeta = 0.7$ and $\omega_n = 0.82$ in continuous setting, which would achieve the design specifications. Using the following chart with $T = 0.6$



$$\omega_n = 0.82 \approx 0.16\frac{\pi}{T}$$

The desired poles:

$$z_p = 0.7 + j0.2$$

$$H_{\text{desired}}(z) = \frac{(1 - z_p)(1 - \bar{z}_p)}{(z - z_p)(z - \bar{z}_p)} = \frac{0.13}{z^2 - 1.4z + 0.53}$$

3.13. Determination of the PI controller parameters

Comparing the denominator of the actual closed-loop transfer function

$$H(z)=\frac{0.00058\,(k_{\mathrm{P}}+k_{\mathrm{I}})z-0.00058\,k_{\mathrm{P}}}{z^{2}+[0.00058\,(k_{\mathrm{P}}+k_{\mathrm{I}})-1.942\,]z+(0.942-0.00058\,k_{\mathrm{P}})}$$

with that of the desired one

$$H_{\text{desired}}(z)=\frac{0.13}{z^{2}-1.4z+0.53}$$

we obtain

$$\begin{cases}0.00058\,(k_{\mathrm{P}}+k_{\mathrm{I}})-1.942=-1.4\\ 0.942-0.00058\,k_{\mathrm{P}}=0.53\end{cases}\Rightarrow\quad\begin{matrix}k_{\mathrm{I}}=224\\ k_{\mathrm{P}}=710\end{matrix}$$

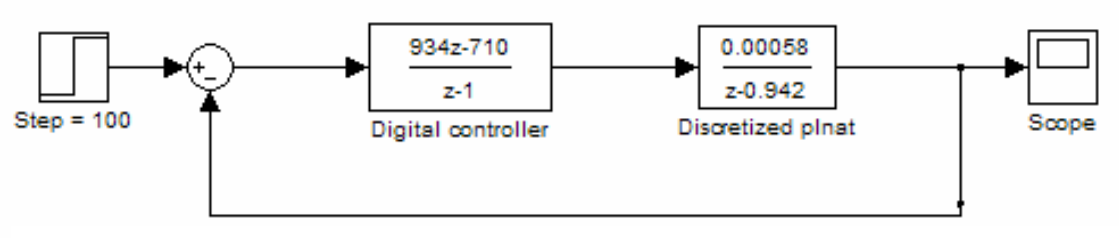
and a digital PI controller

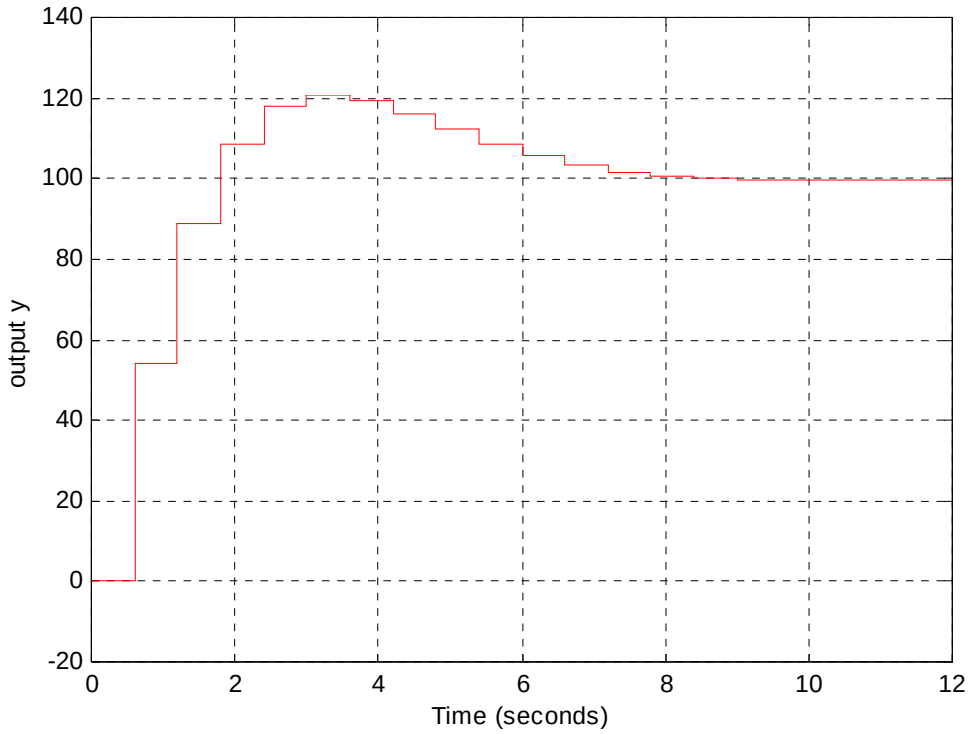
$$D(z)=k_{\mathrm{P}}+k_{\mathrm{I}}\frac{z}{z-1}=\frac{(k_{\mathrm{P}}+k_{\mathrm{I}})z-k_{\mathrm{P}}}{z-1}=\frac{934\,z-710}{z-1}$$

Note: we cannot do much with the numerators of these transfer functions. It does affect the overall performance.

3.14. Simulation of the digital controller with discretized plant

We simulate the digital controller with the discretized plant to see whether the specifications are fulfilled in the discrete-time setting:

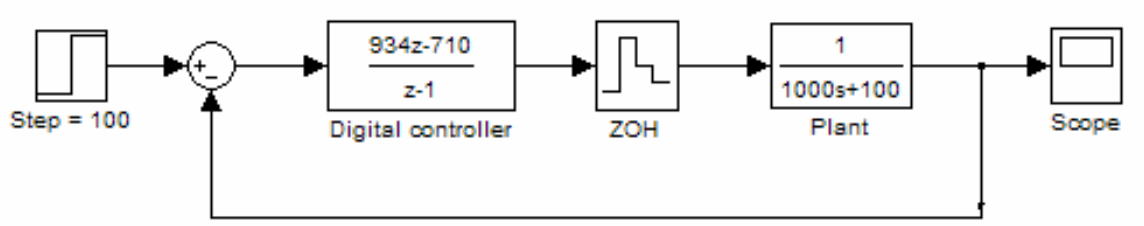


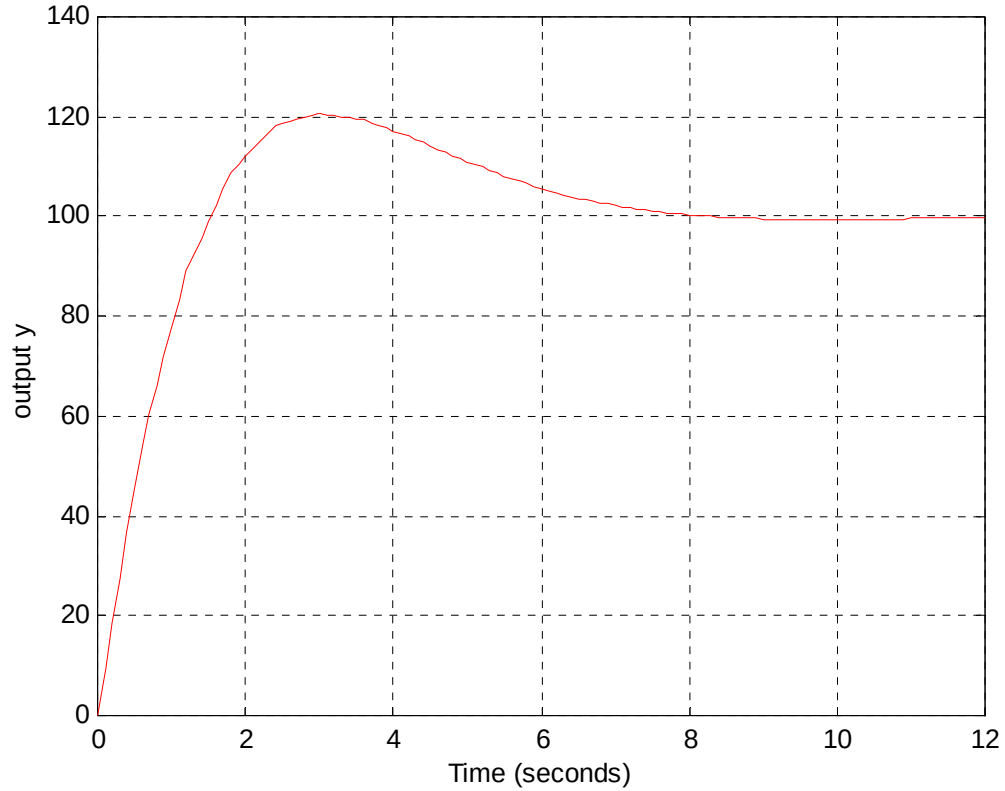


<u>Remark:</u> The overshoot is slightly larger than the design specification. But, the settling time meets the specification. The performance can be fine tuned by re-selecting the desired pole locations in z-plane.

3.15. Simulation of the digital controller with actual plant

We simulate the digital controller with the actual plant.

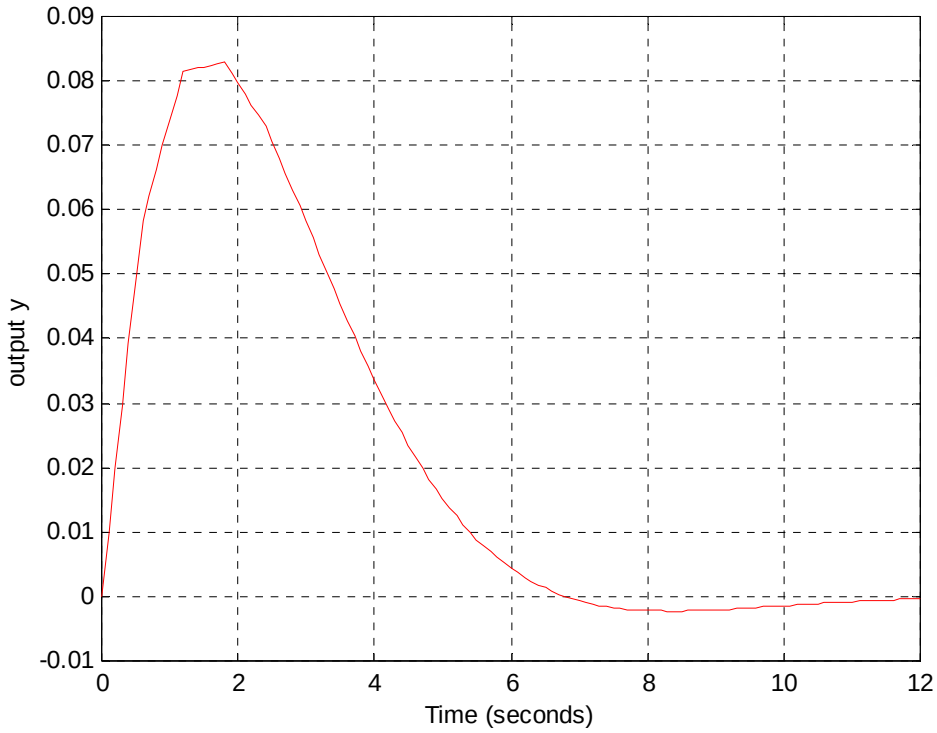


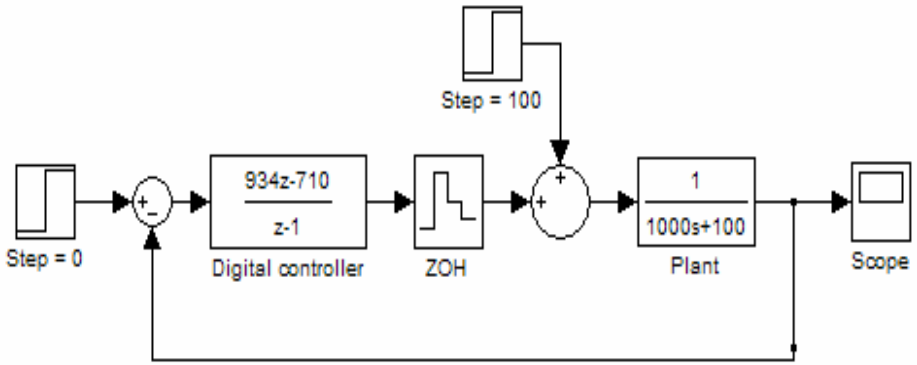


Remark: When the control law is implemented onto the actual continuous-time plant, the overshoot and the settling time are above the same as those obtained with the discretized system. All design specifications are met with a sampling period $T = 0.6$ seconds.

3.16. Disturbance rejection

From the analysis in 2.10, it was shown that a step disturbance can be rejected when the controller has an integral action. Since we are using a PI controller, step disturbance should be rejected in our design although we haven't explicitly considered such a property in our design. We verify this through simulation by injecting a step function into the plant input. To see the effect of the disturbance on the system output, we let the reference to be zero.





A step disturbance with magnitude = 100 nicely is rejected from the output ! How about a ramp disturbance? D.I.Y.

3.17. Remarks on higher order systems

When the given plant has a dynamic order higher than 1 and/or a general PID controller is used, the overall closed-loop transfer function from r to y will have an order larger than 2, e.g.,

$$H(z) = \frac{b_m z^m + b_{m-1} z^{m-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}, \qquad m \le n, \;\; n > 2$$

In this case, we should place the poles of the above transfer function by comparing it to the following desired transfer function

$$H_{\text{desired}}(z) = \frac{*}{(z-\alpha_1)\cdots(z-\alpha_{n-2})\cdot(z-z_p)(z-\bar{z}_p)}$$

i.e., by placing all the rest poles close to the origin, which is the fastest location in digital control. Eventually, dynamics associated with the poles close to the origin will die out very fast and the overall system is dominated by the pair left. This is left for students to practice in tutorial questions.

3.18. Deadbeat controller design

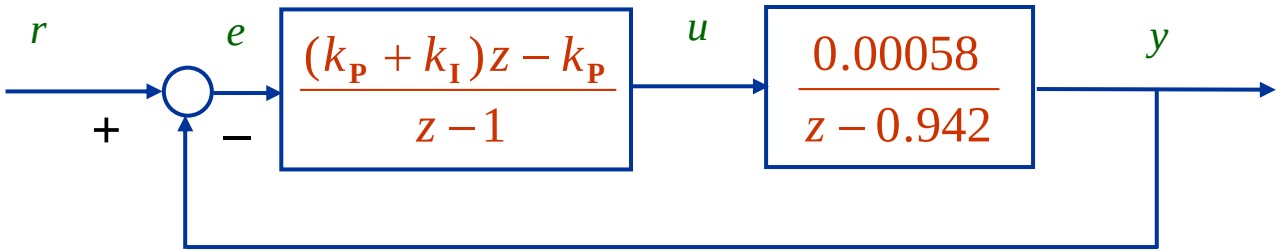
A unique feature of digital control is that we can design a control law such that the resulting system output is capable of following the reference input in a finite number of steps, i.e., in finite time interval, which can never be achieved in continuous-time setting. Such a control law is called deadbeat controller, which in fact places all the closed-loop system poles at the origin. Thus, the desired closed-loop transfer function under the deadbeat control is

$$H_{\text{desired}}(z) = \frac{1}{z^n}$$

The deadbeat control can only guarantee the system output to reach the target reference in n steps. The resulting overshoot could be huge and the control input could be very high (which is equivalent to that the energy required to achieve such a performance is large). As such, deadbeat control is generally impractical and rarely used in practical situations.

3.19. Design example:

We now design a deadbeat PI controller for the cruise control system. Recall



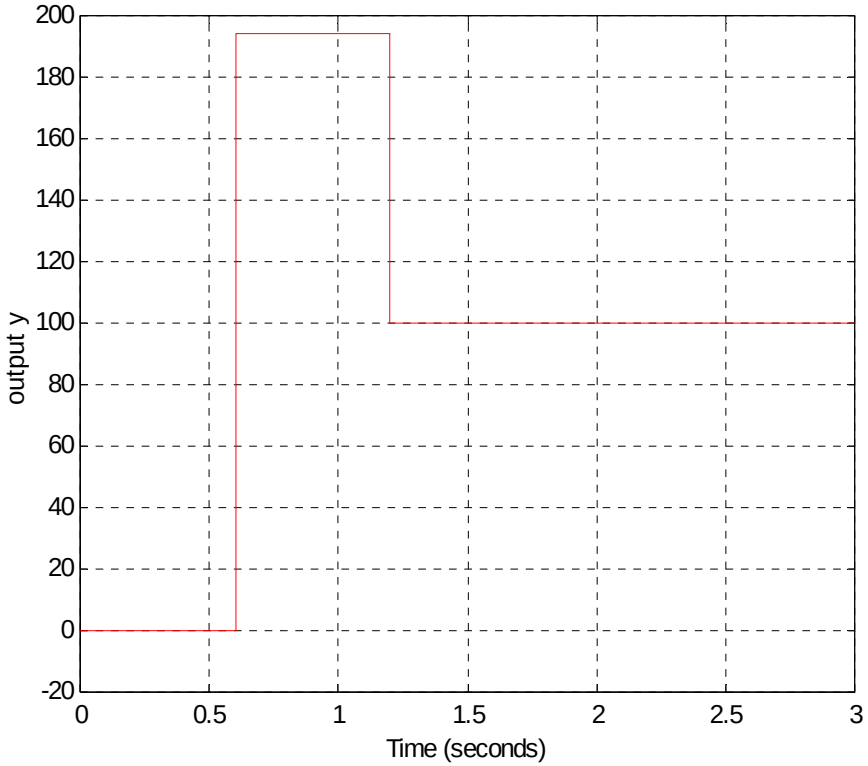
The resulting closed-loop transfer function from r to y is given by

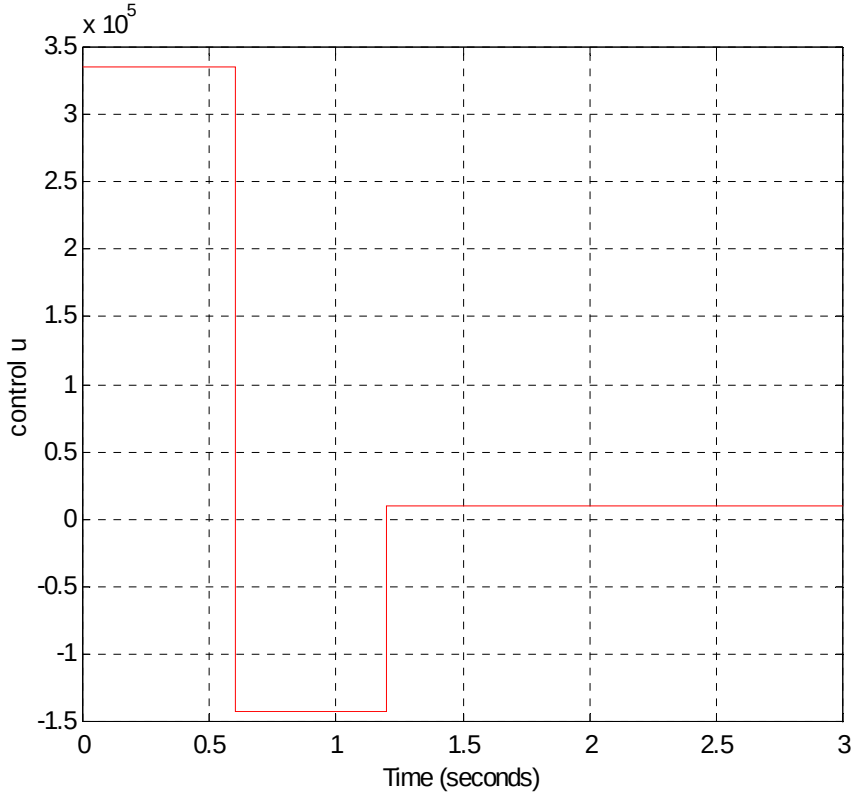
$$H(z) = \frac{0.00058\,(k_P + k_I)z - 0.00058\,k_P}{z^2 + [0.00058\,(k_P + k_I) - 1.942\,]z + (0.942 - 0.00058\,k_P)} \quad \succ \quad \frac{*}{z^2}$$

$$\Rightarrow \quad k_P = 1624.137931, \;\; k_I = 1724.137931$$

$$\Rightarrow \quad D(z) = \frac{(k_P + k_I)z - k_P}{z-1} = \frac{3348.275862\,z - 1624.137931}{z-1}$$

3.20. Simulation results





The system out settles down to the target reference in 2 steps, i.e., 2 x 0.6 = 1.2 seconds (very fast). But, it has about 100% overshoot and huge control input. Such a controller cannot be implemented in real life, period ! However, there could be applications (such as military applications) that deadbeat control might be desirable.

Homework assignment 1 (hand in your solution next week):

A typical hard disk drive actuator can be modeled quite accurately as a double integrator:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{6 \times 10^7}{s^2}$$

where y is the displacement of the read/ write head in micrometer and u is actuator input voltage in volts. The sampling rate used in a typical hard disk drive servo system is about 10 kHz, which is equivalent to a sampling period T = 1/10000 = 0.0001 seconds. It is required to design an appropriate control law such that the resulting closed-loop system has an overshoot less than 25% and a settling time less than 8 milliseconds due to a step reference of 1 micrometer.

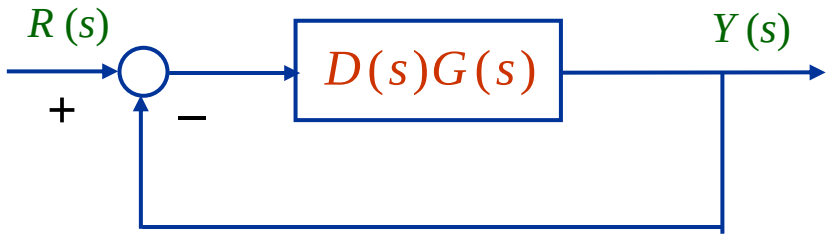
- Design a digital PD or PI or PID controller that meets the above design specifications using the emulation method.
- Design a digital PD or PI or PID controller that meets the design specifications using the pole placement technique.

Show all the detailed design procedures and simulation results. Use MATLAB and SIMULINK whenever is possible.

4. Frequency Domain Design

4.1. Bode diagrams for continuous systems —A revisit

Bode diagram is the frequency responses (both magnitude response and phase response with respect to frequency). Consider a standard unit feedback system:



$$H(s) = \frac{Y(s)}{R(s)} = \frac{G(s)D(s)}{1 + G(s)D(s)}$$

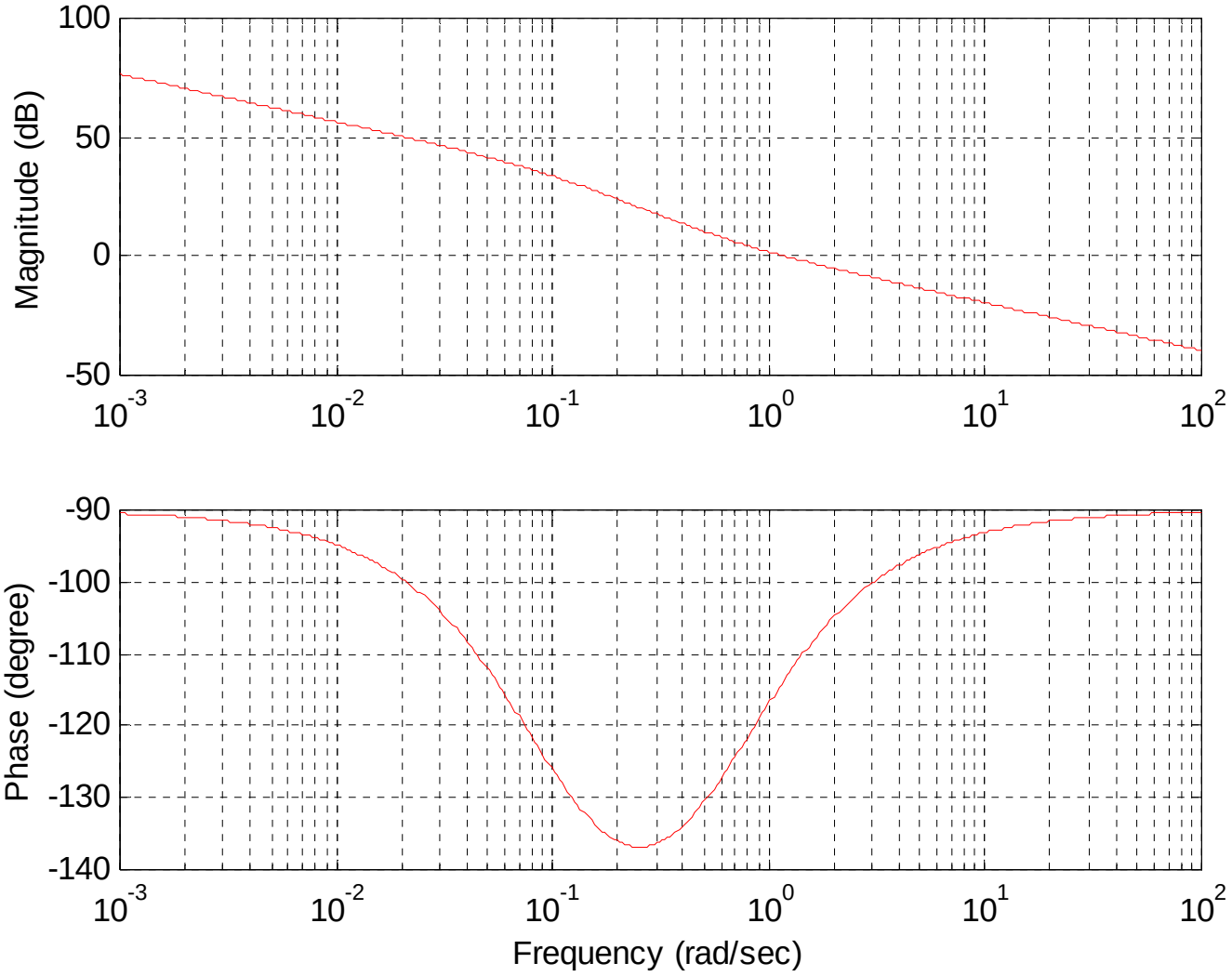
which has a closed-loop transfer function $H(s)$. Interestingly, if the open-loop transfer function $D(s)G(s)$ is not unstable, then its frequency responses (or Bode diagram) can be easily used to determine the stability of the closed-loop system $H(s)$. As such, it is often to plot the Bode diagram for the open-loop system $D(s)G(s)$. Magnitude & phase responses are defined as:

$$\text{Magnitude Response} = \left| D(j\omega)G(j\omega) \right| \qquad -\infty < \omega < \infty$$

$$\text{Phase Response} = \angle D(j\omega)G(j\omega)$$

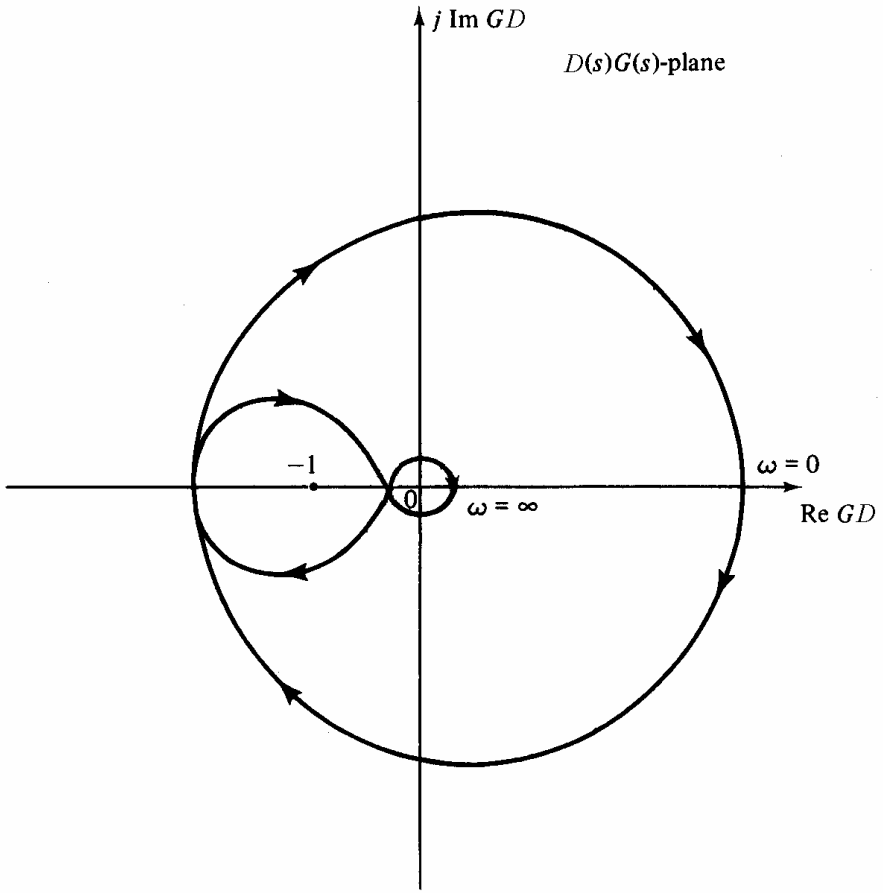
4.2. Example: Bode diagrams of the open-loop system in 3.4 & 3.5

$$D(s)G(s) = \frac{1050\,s + 670}{s} \cdot \frac{1}{1000\,s + 100} = \frac{1050\,s + 670}{1000\,s^2 + 100\,s}$$



4.3. Nyquist stability theory in continuous systems — A revisit

Nyquist plot is to draw the frequency response of the open-loop system in a single complex plane instead of separating the magnitude and phase responses into two individual diagrams as in the Bode plot.



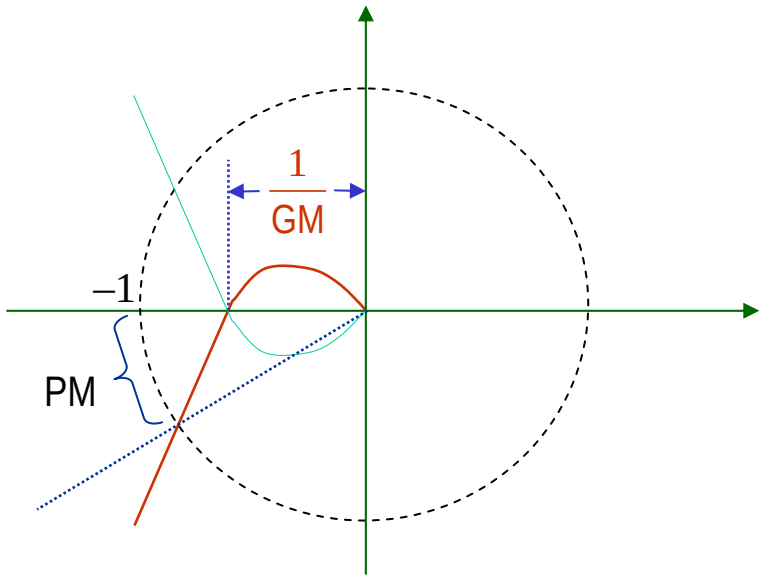
Nyquist Stability Criterion

Let N be the number of clockwise encirclements of the point -1 in the Nyquist plot, P be the numbers of unstable poles of the open-loop system $D(s)G(s)$. Then, the number of unstable pole of the closed-loop system $H(s)$, say Z, is given by $Z = P + N$.

If $D(s)G(s)$ is stable, i.e., $P = 0$, then N has to be zero in order to guarantee the stability of the closed-loop system $H(s)$.

4.4. Gain and phase margins in the Nyquist plot

Assuming the open-loop system is stable, the gain margin and phase margin can be found from the Nyquist plot by zooming in the region in the neighbourhood of the origin.



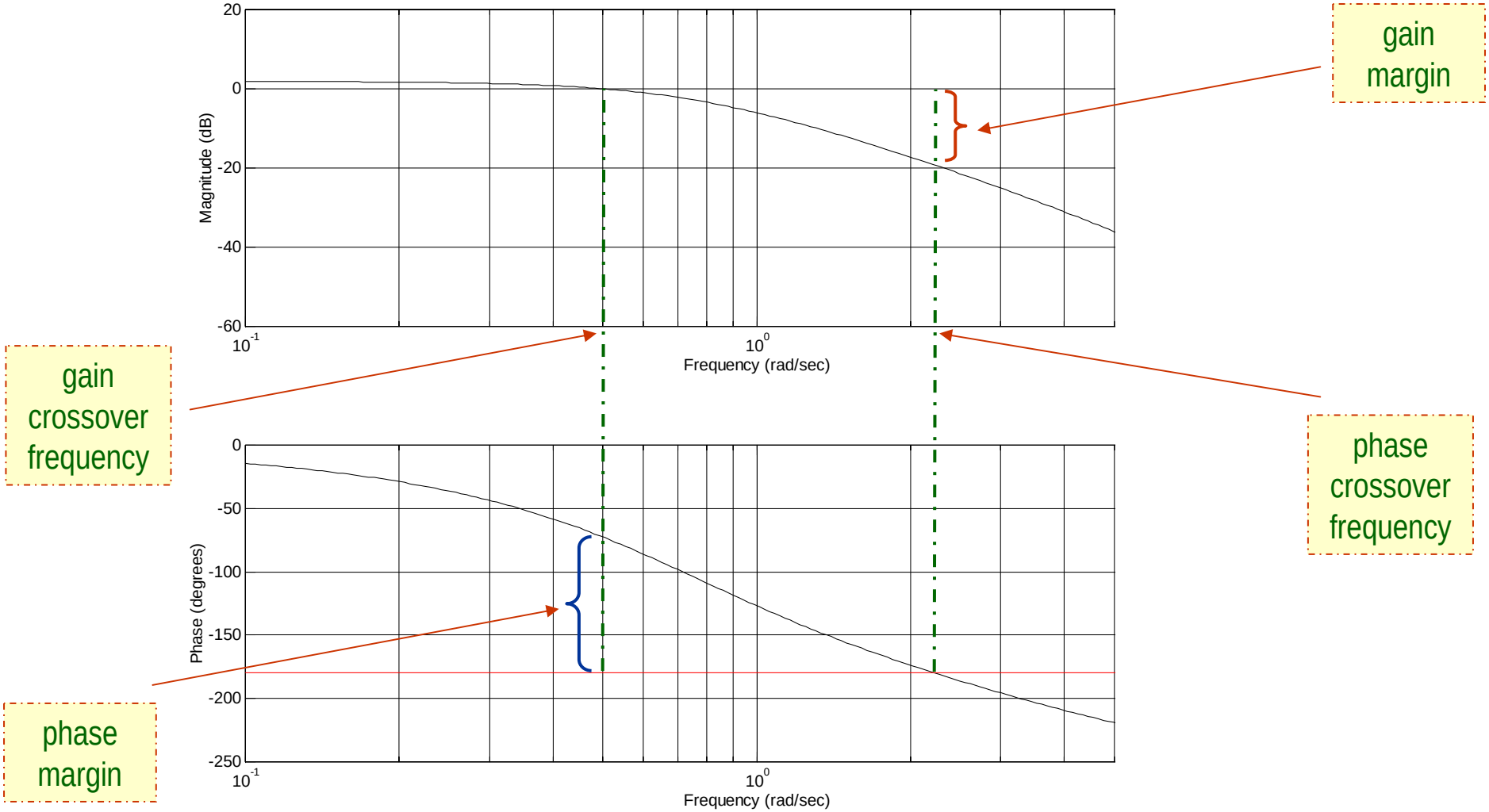
Remark: Gain margin is the maximum additional gain that can be applied to the closed-loop system such that it will still remain stable. Similarly, phase margin is the maximum phase that the closed-loop system can tolerate such that it will still remain stable.

Mathematically,

$$\text{GM} = \frac{1}{\left|D(j\omega_p)G(j\omega_p)\right|}, \quad \text{where } \omega_p \text{ is such that } \angle D(j\omega_p)G(j\omega_p) = 180^\circ$$

$$\text{PM} = \angle D(j\omega_g)G(j\omega_g) + 180^\circ, \quad \text{where } \omega_g \text{ is such that } \left|D(j\omega_g)G(j\omega_g)\right| = 1$$

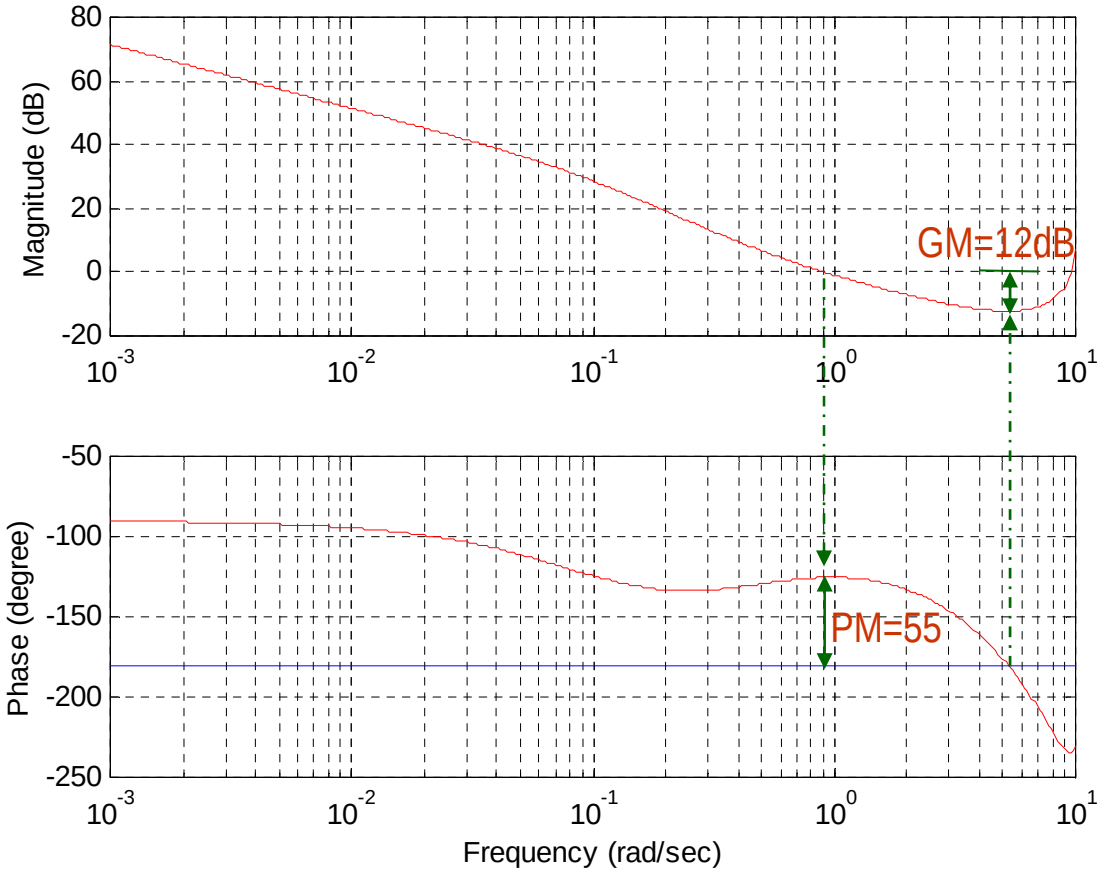
4.5. Gain and phase margins in the Bode diagram



4.6. Bode diagram for discrete systems

Bode diagram for discrete systems consists of both the magnitude and phase responses of a discrete system, which are defined as follows:

$$\text{Magnitude} = \left| D(z)G(z) \right|_{z=e^{j\omega T}} \qquad \text{Phase} = \angle D(z)G(z)_{z=e^{j\omega T}} \qquad -\pi \le \omega T \le \pi$$



Example: Let us draw Bode diagram for the open-loop system in 3.14, where

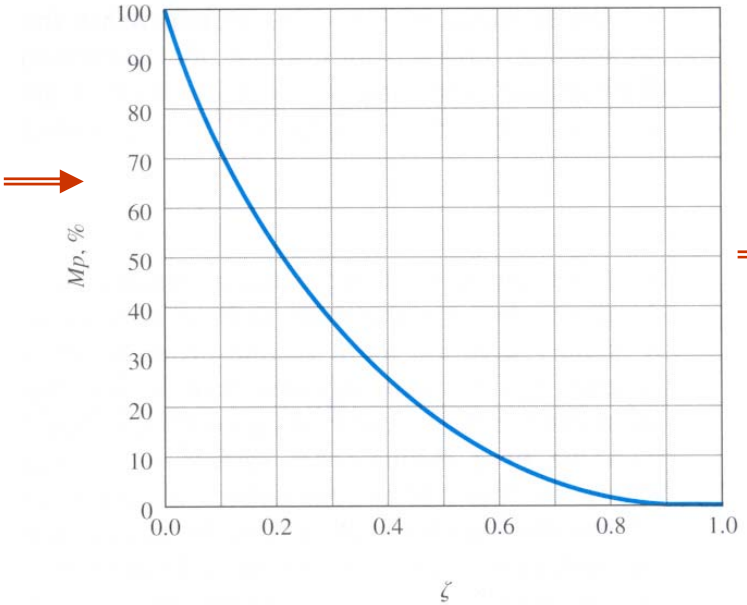
$$D(z)G(z) = \frac{0.5417\,z - 0.4118}{z^2 - 1.942\,z + 0.942}$$

Note: Both magnitude and phase responses repeat after $2\pi/T$.

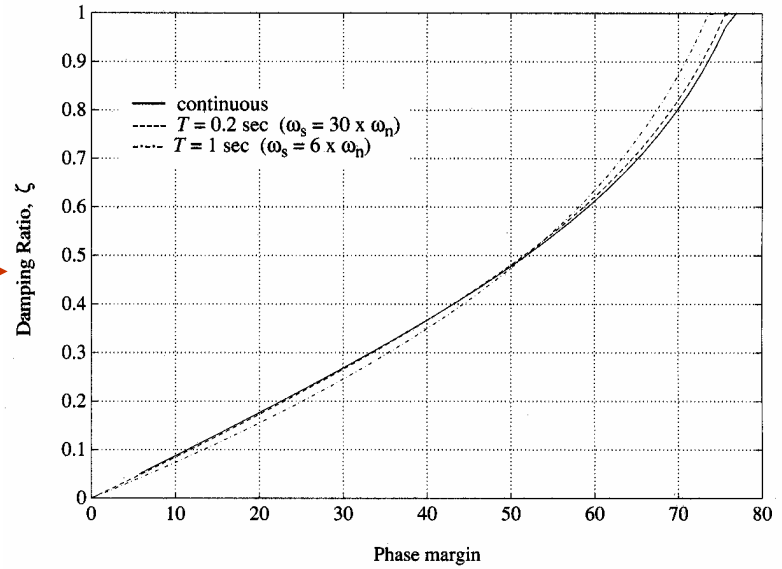
4.7. Frequency domain design specifications

The time domain design specifications can be translated into the requirement on phase margin and some other frequency domain properties. In particular, we have the following relationships for typical second order systems:

overshoot ⟹



⟹

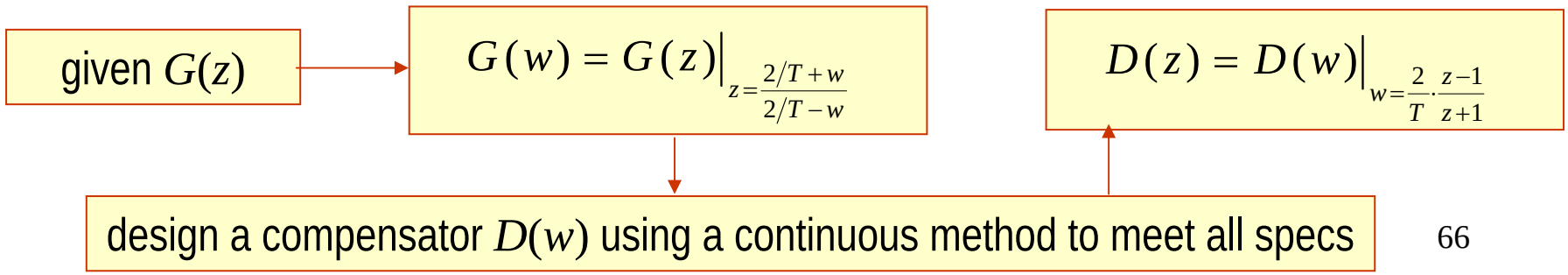


Damping ratio of a second-order system versus phase margin

settling time ⟹ $\omega_n \cong \dfrac{4.6}{\zeta t_s}$ ⟹ open - loop gain crossover frequency $\omega_g \cong \omega_n$

4.8. Frequency domain design methods

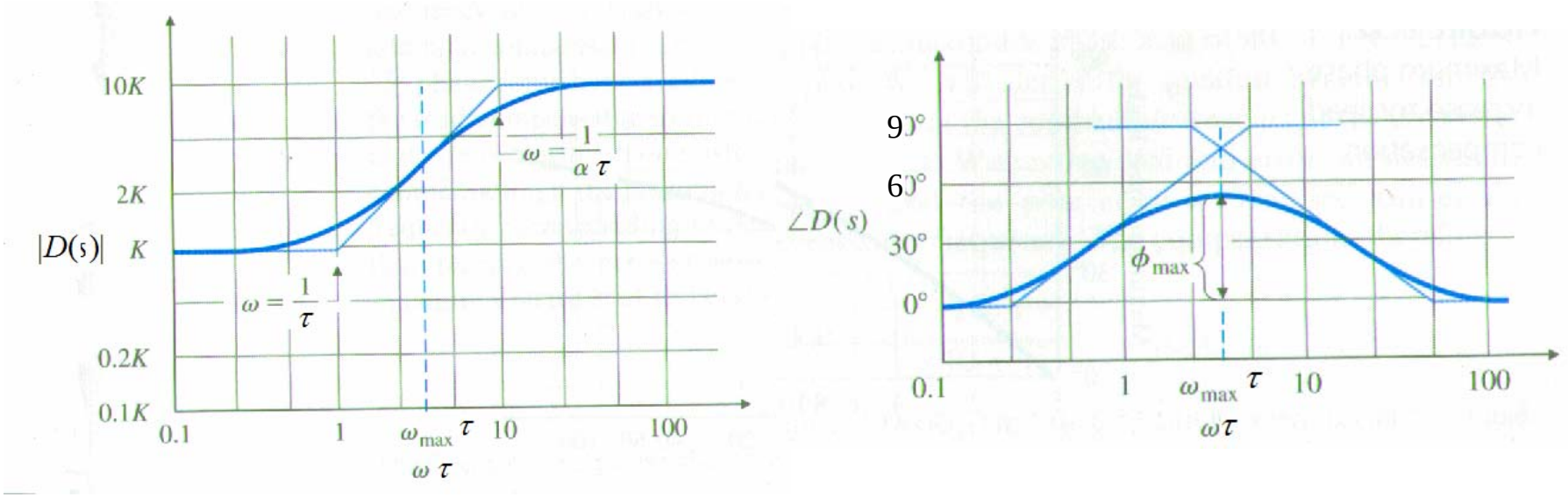
There are three commonly used frequency domain design methods, namely Lead Compensator, which is an approximation of PD control, Lag Compensator, which is an approximation of PI control, and lastly the PID Compensator. The frequency domain design methods for continuous systems and discrete systems are basically the same. For discrete systems, the idea is to use a so-called w-transformation (actually it is a bilinear transformation) to transform the discrete system into a w-domain system, which is pretty much the same as a continuous system. Everything we have learnt from the compensation design for continuous systems can be directly applied to yield a necessary compensator in w-domain, which can be transformed back to a discrete version through an inverse w-transformation (= an inverse bilinear transformation).



4.9. Lead compensation

Lead compensation has a transfer function of $D(s) = K\dfrac{\tau s + 1}{\alpha\tau s + 1}, \quad \alpha < 1$. Clearly for small α, it is an approximation of PD control.

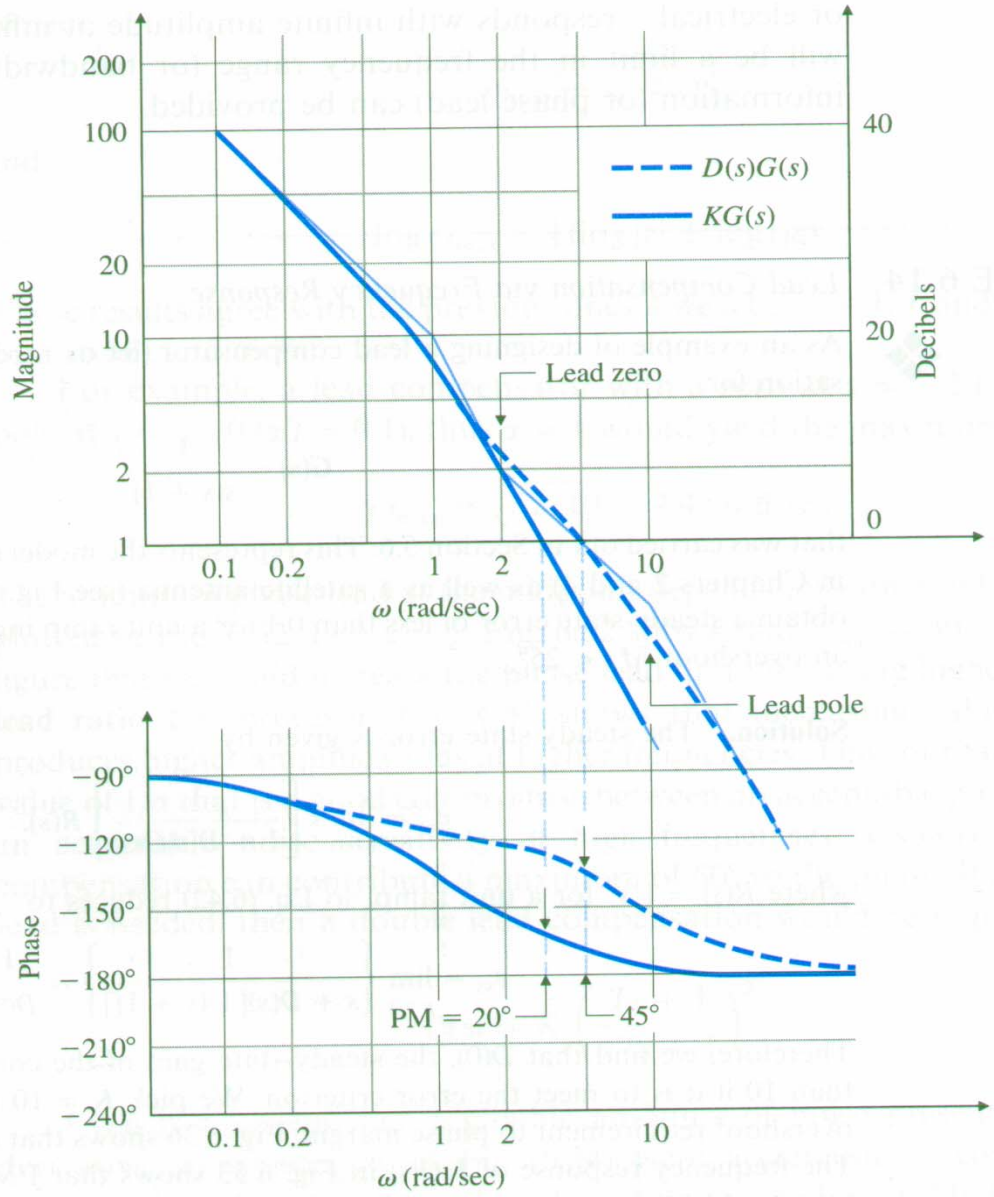
The frequency response of a typical lead compensator is shown in the figure below:



$$\alpha = \frac{1 - \sin\phi_{\max}}{1 + \sin\phi_{\max}} \quad \text{and} \quad \tau = \frac{1}{\omega_{\max}\sqrt{\alpha}}$$

Here parameters K, $\phi_{\max}$ and $\omega_{\max}$ are to be determined from design specifications.

4.10. Uncompensated and lead-compensated systems



The Kay Idea

The key idea using the lead compensation is enlarge the gain crossover frequency and add additional phase. Thus, if the desired gain crossover frequency is larger than that of $KG(s)$, the lead compensation should be used, period.

4.11. Lead Compensation Design Procedure

Step 1: Determine open-loop gain K to satisfy requirements on steady state error.

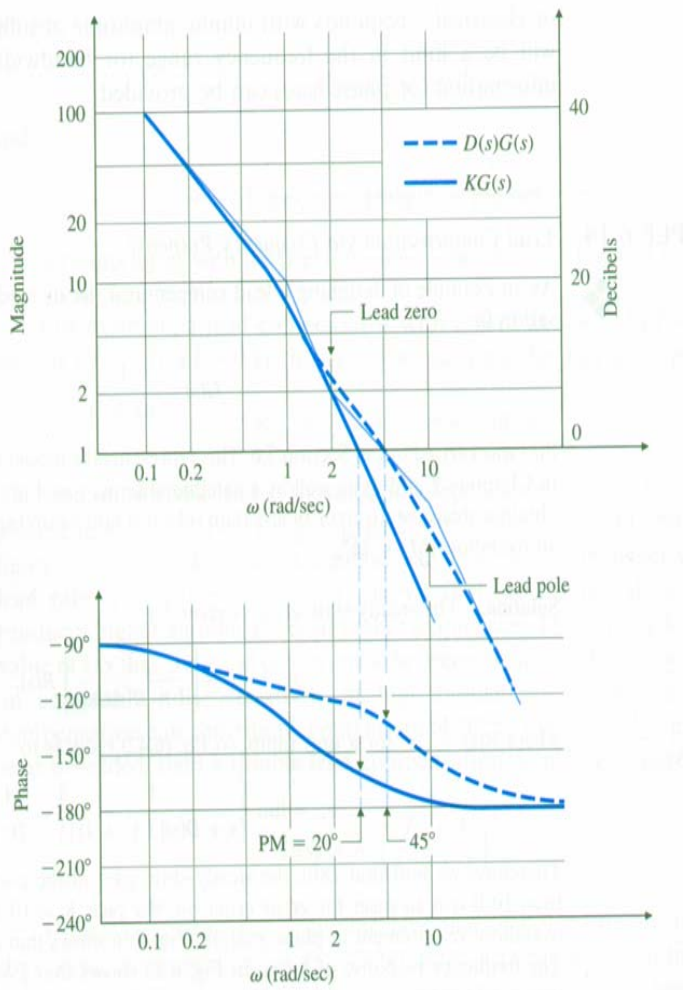
Step 2: Find new open-loop cross over freq. from desired $\omega_n = \omega_{max}$, the point the phase lead is added.

Step 3: Evaluate the PM of $KG(w)$ at the desired cross over frequency ω_n.

Step 4: Allow for some extra PM (5° to 12°), and determine the needed phase lead ϕ_{max}.

Step 5: Compute $$\alpha = \frac{1-\sin\phi_{max}}{1+\sin\phi_{max}}, \quad \tau = \frac{1}{\omega_{max}\sqrt{\alpha}}$$

Step 6: Verify the design using MATLAB. Redo if necessary.



4.12. Design example

An electric motor can be modelled with a sampling period $T = 0.1s$ as follows:

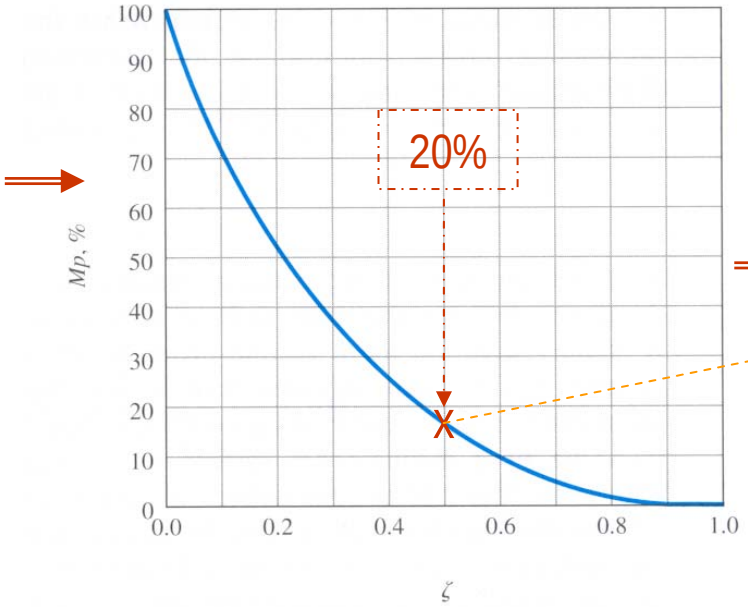
$$G(z) = \frac{0.5z + 0.5}{(z - 0.5)(z - 0.9)}$$

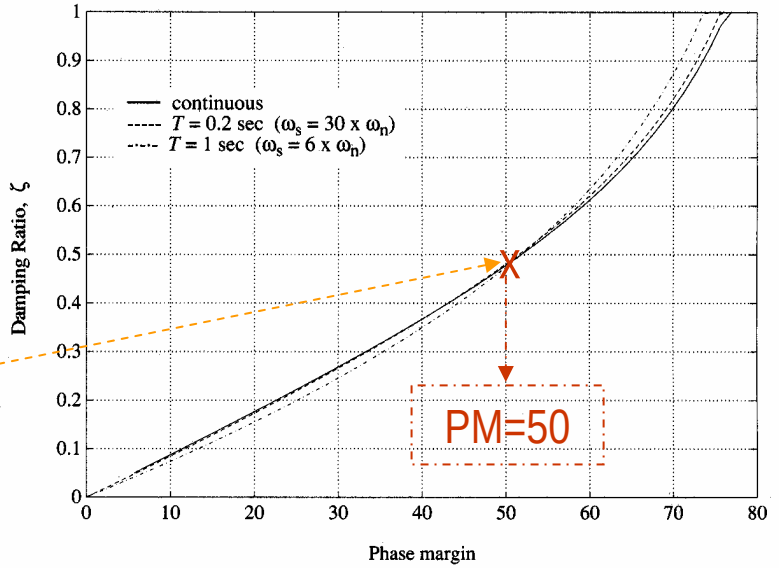
Design a digital control system with a lead compensator such that the resulting system output tracks a step reference with a settling time less than $1s$ with an overshoot less 20% and steady state error less than 12.5% .

Converting this to w-plane (see 4.8), we have

$$G(w) = G(z)\Big|_{z=\frac{2/T+w}{2/T-w}} = \frac{0.5\dfrac{20+w}{20-w} + 0.5}{\left(\dfrac{20+w}{20-w} - 0.5\right)\left(\dfrac{20+w}{20-w} - 0.9\right)} = \frac{20(20-w)}{(1.5w+10)(1.9w+2)}$$

overshoot





Damping ratio of a second-order system versus phase margin

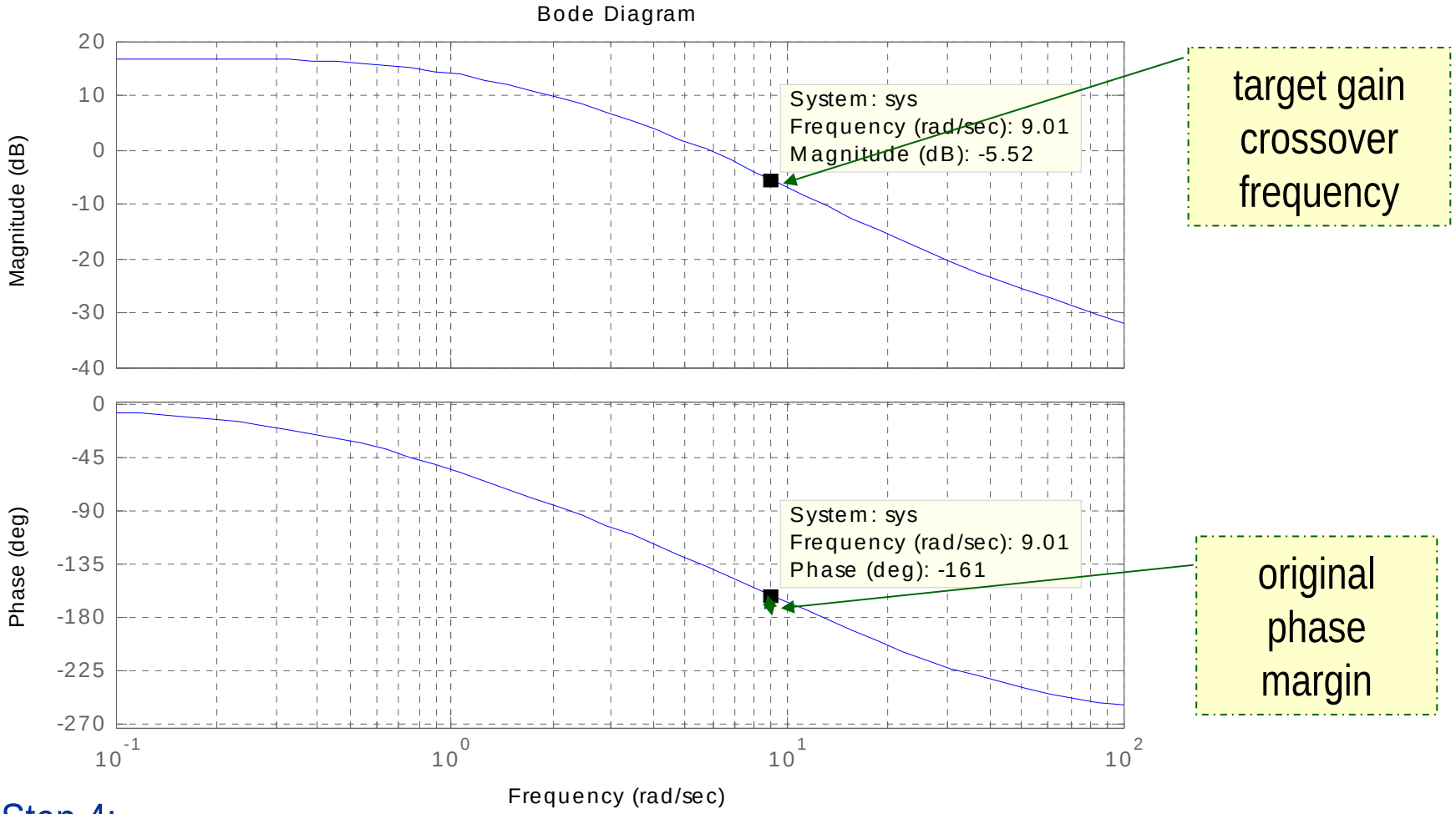
settling time $\Longrightarrow$ $\omega_n \cong \dfrac{4.6}{\zeta t_s}$ $\Longrightarrow$ open-loop gain crossover frequency $\omega_g \cong \omega_n$

1 second $\Longrightarrow$ $\omega_n \cong \dfrac{4.6}{0.5 \times 1} = 9.2$ $\Longrightarrow$ $\omega_g \cong \omega_n = 9$ $\Longleftarrow$ Step 2

Steady state error (see 2.3) — Step 1: Determination of the static gain K

$$e(\infty) = \frac{1}{1+K_p} = 0.125 \quad \Rightarrow \quad K_p = D(1)G(1) = K \cdot 1 \cdot \frac{0.5+0.5}{(1-0.5)(1-0.9)} = 7 \quad \Rightarrow \quad K = 0.35$$

Step 3:



Step 4:

$$\phi_{\max} = \text{Desired PM} - \text{Original PM} + \text{Allowance} = 50 - (180 - 161) + 4 = 35$$

Step 5:

$$\alpha = \frac{1-\sin\phi_{\max}}{1+\sin\phi_{\max}} = \frac{1-\sin 35^\circ}{1+\sin 35^\circ} = 0.27 \quad , \quad \tau = \frac{1}{\omega_{\max}\sqrt{\alpha}} = 0.214$$

Finally, we obtain the lead compensator in w-domain:

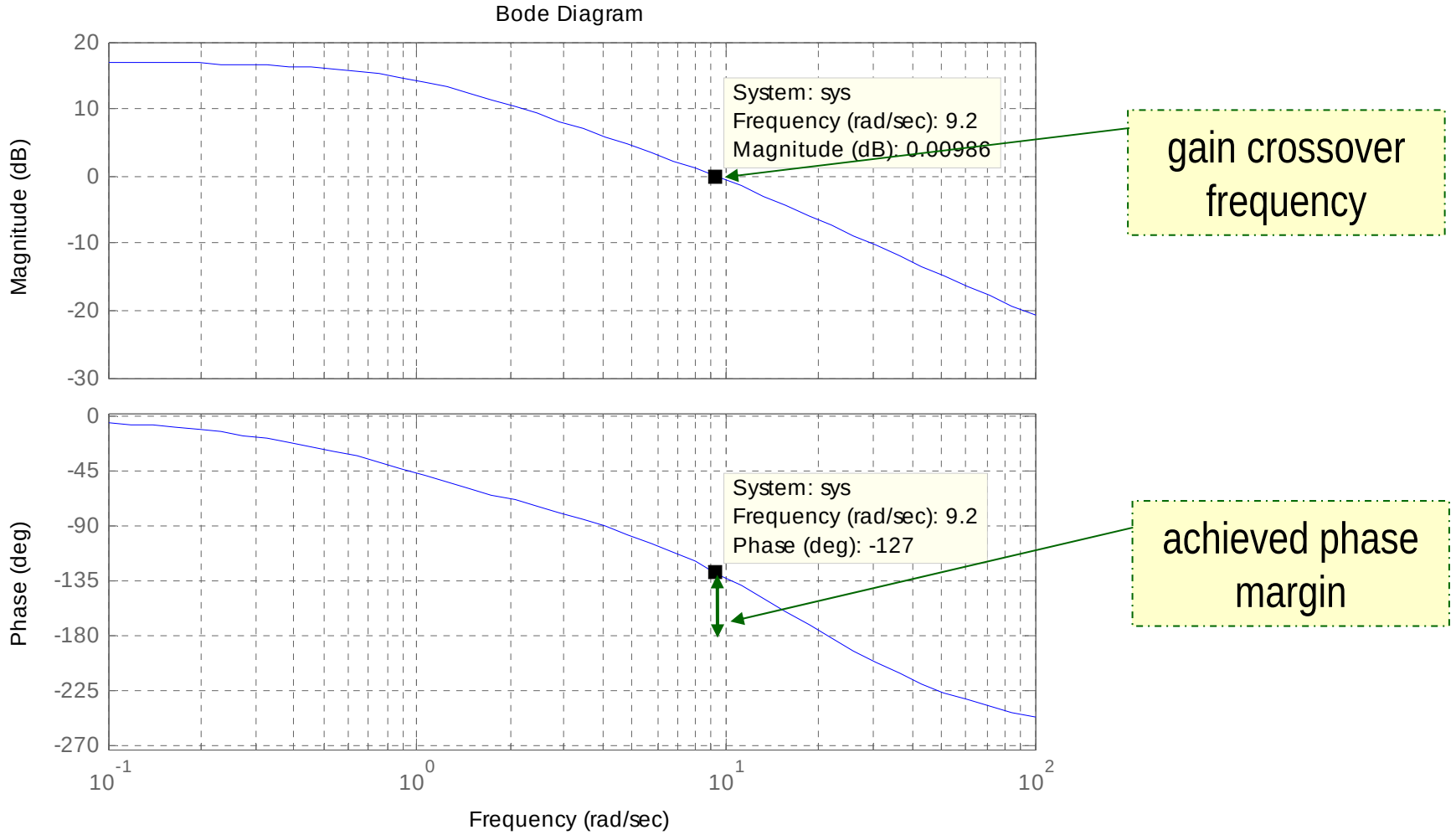
$$D(w) = K\,\frac{\tau w + 1}{\alpha\tau w + 1} = 0.35\left(\frac{0.214\,w + 1}{0.058\,w + 1}\right)$$

which can be converted back to z-domain using the inverse bilinear transformation (see 4.8)

$$D(z) = D(w)\Big|_{w=\frac{2}{T}\cdot\frac{z-1}{z+1}} = 0.35\,\frac{0.214\,\dfrac{2}{0.1}\cdot\dfrac{z-1}{z+1}+1}{0.058\,\dfrac{2}{0.1}\cdot\dfrac{z-1}{z+1}+1} = \frac{1.848\,z - 1.148}{2.16\,z - 0.16}$$

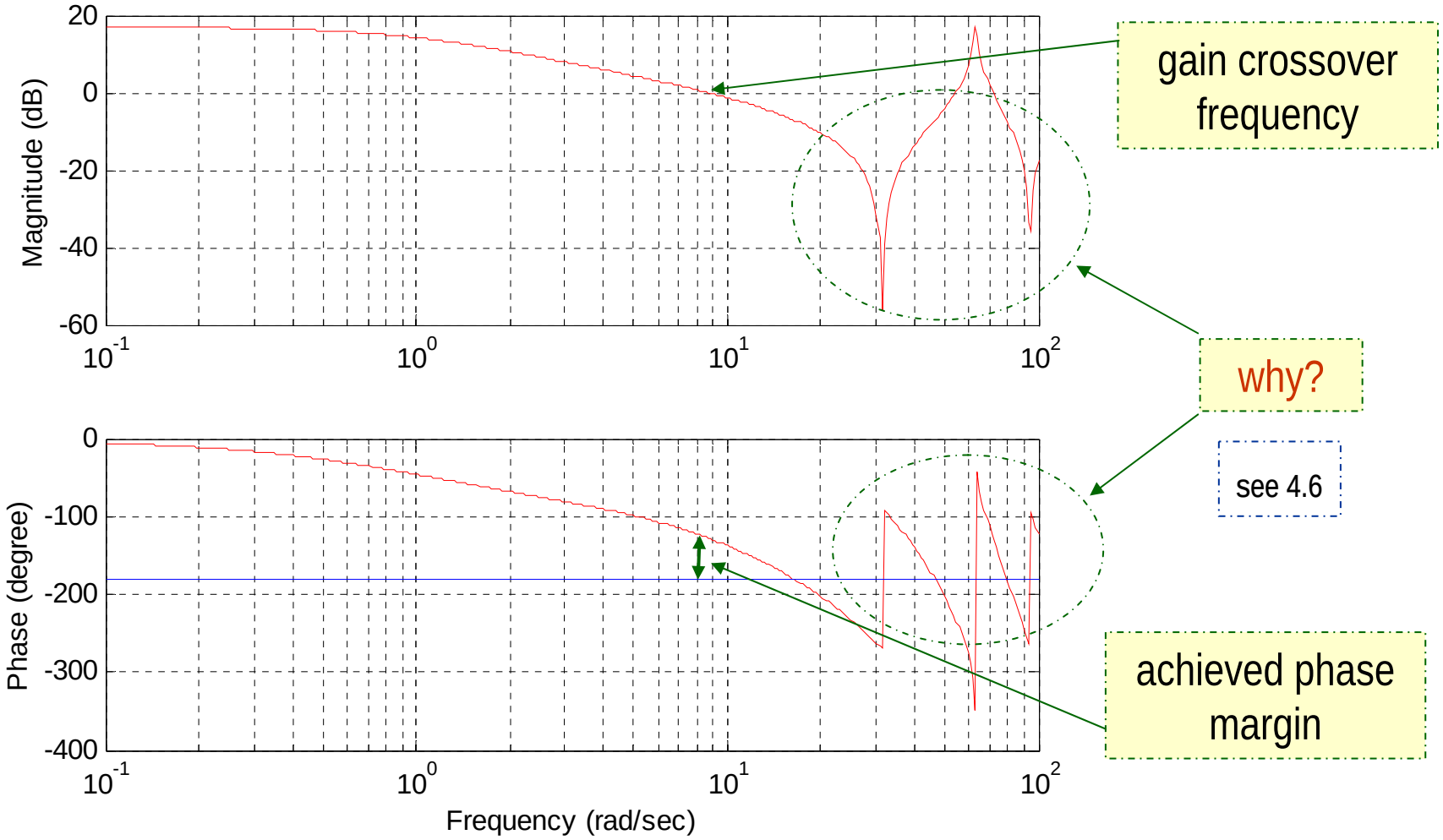
The above the digital lead compensator. However, nothing is certain without verification. We need to first verify our design in frequency domain. More importantly, it should also meet the design specifications in time domain.

4.13. Verification in w-domain (pseudo s-domain)



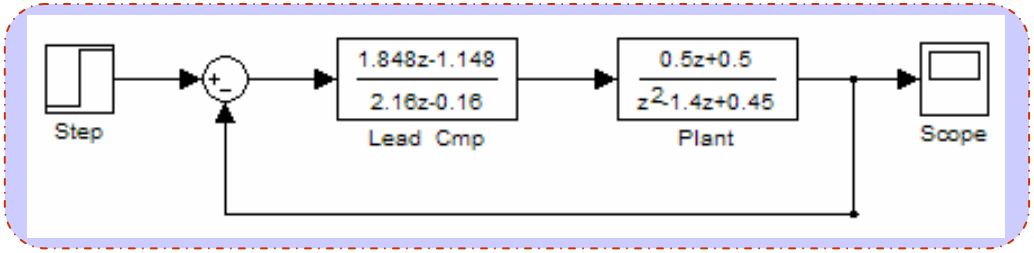
The resulting gain crossover frequency is 9.2 rad/sec and PM=180-127 = 53°. Perfect !

4.14. Verification in z-domain

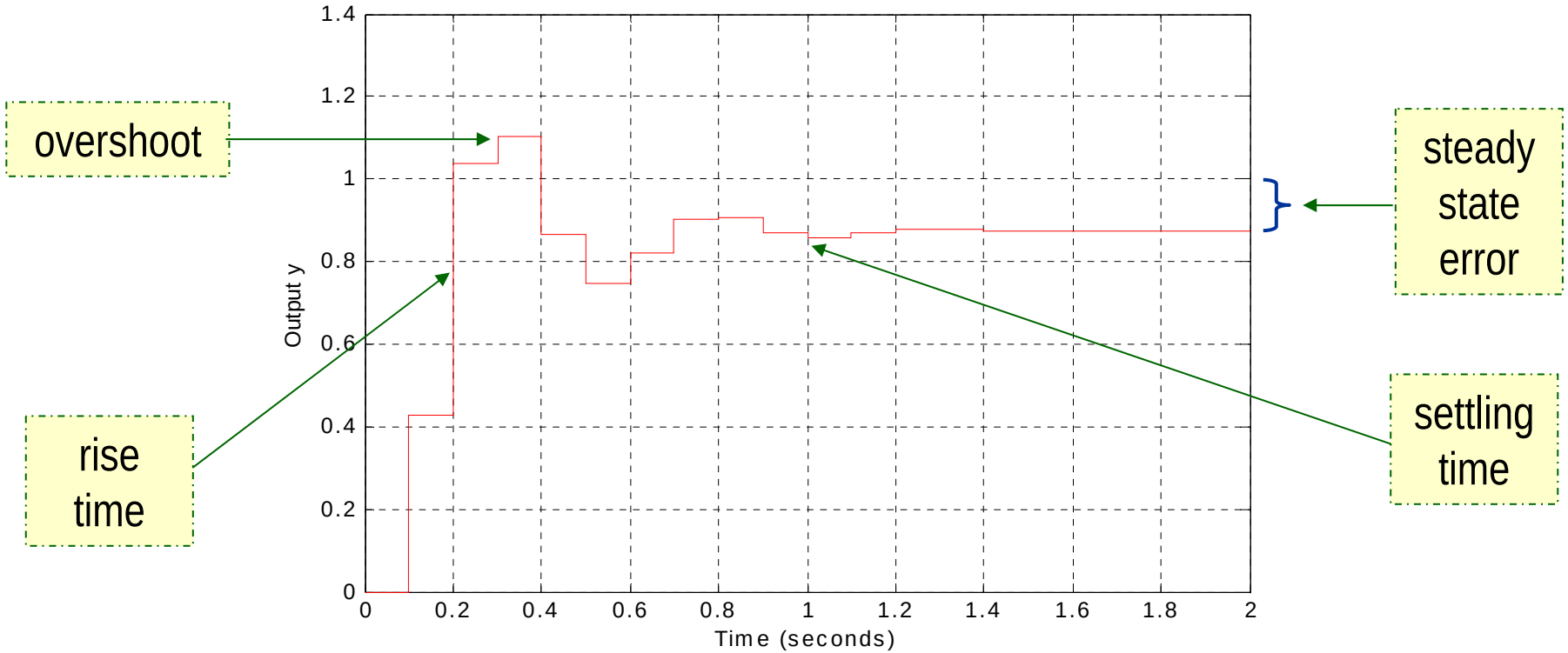


The resulting gain crossover frequency is 8.5 rad/sec and PM=50°. Perfect !

4.15. Verification in time domain



Simulink block

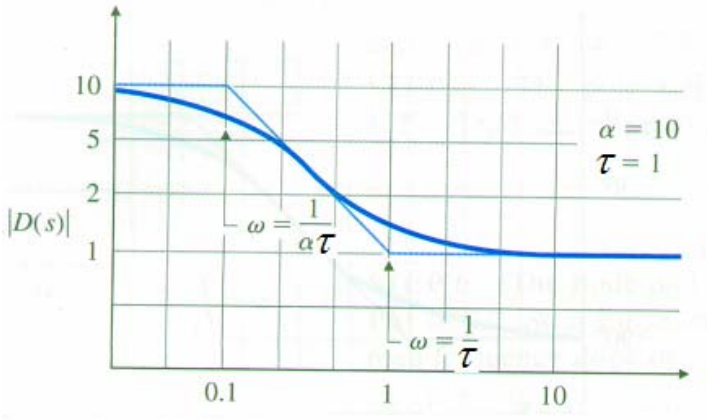


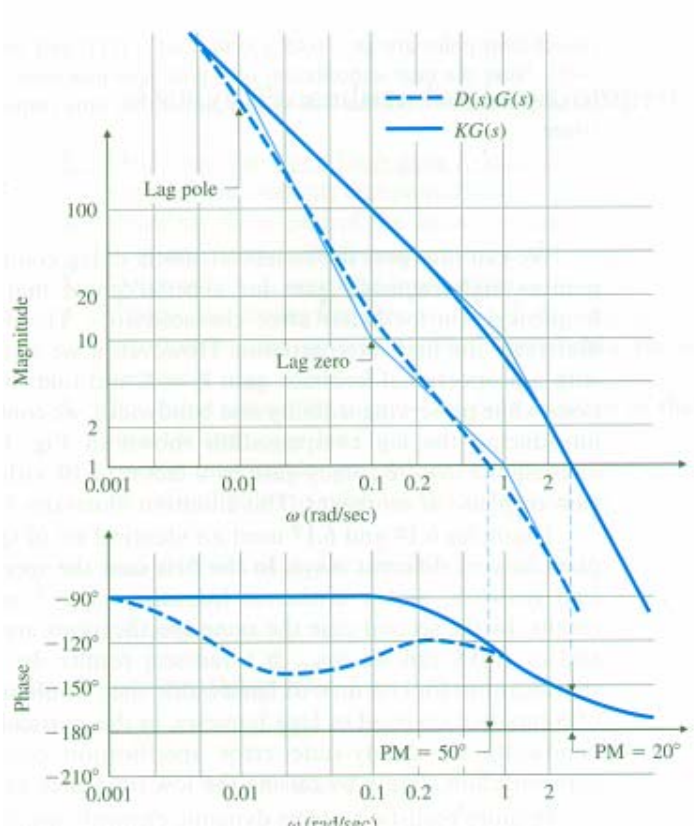
The steady state error = 12.5%, settling time = 1s and overshoot = 26 % (almost there) !

4.16. Lag compensation

Lag compensation has a transfer function of $D(s) = K\alpha \dfrac{\tau s + 1}{\alpha \tau s + 1}, \ \alpha > 1$. For large α, it is an approximation of PI control.

The frequency response of a typical lag compensator is shown in the figure below:

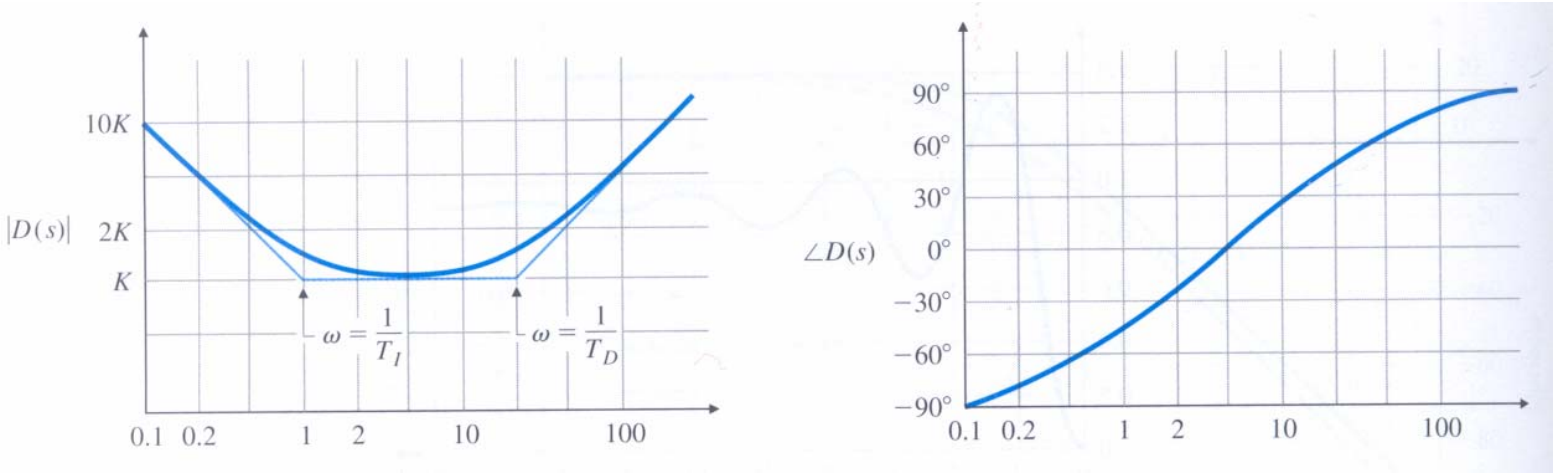




Kay Idea: Lag compensator reduce the gain crossover frequency to where the phase margin required.

4.17. PID Compensation

Note that the lead compensation is the approximation of PD control and the lag compensation approximates PI control. PID compensation is nothing more than the combination of lead compensation (PD) and lag compensation (PI), which has the following frequency responses:



Design procedures for the lag compensation and PID compensation can be found in almost any textbook on introduction to classical control including class notes for EE2131. Students who forget these design procedures are urged to dig out the old notes and textbooks and ...

Homework assignment 2 *(hand in your solution next week)**:***

Reconsider the hard disk drive servo system in the Homework Assignment 1, i.e.,

$$G(s) = \frac{Y(s)}{U(s)} = \frac{6 \times 10^7}{s^2}$$

where y is the displacement of the read/ write head in micrometer and u is actuator input voltage in volts. The sampling rate is again chosen to be 10 kHz, which gives a sampling period T = 1/10000 = 0.0001 seconds. It is required to design an appropriate compensator such that the resulting closed-loop system has an overshoot less than 25% and a settling time less than 8 milliseconds as well as a steady state error to be less 1% due to a step reference of 1 micrometer.

- Can you design a digital lead compensator that would achieve the above design specifications?

If your answer is yes, please give your solutions together with all detailed derivations and simulation results. If your answer is no, please give your reasons together with detailed justifications. Again, make use of MATLAB and SIMULINK whenever is possible.

5. State Space Approach

5.1. Why state space?

State variable technique is to convert transfer function system representations into some first order difference equations (in a matrix form), which many advanced matrix theories and computational tools can be applied to. Thus, control system design using the state variable technique can be done in a very *systematic* fashion and many well-developed commercial software tools such as MATLAB can be readily and easily utilized. Solutions to the control problem can be done in a straightforward manner.

There are two important equations associated with such a technique, the state equation and the output equation, which completely characterized the properties of a linear system. As we have seen in EE2008, any linear time-invariant system can be represented by a state space model with *four constant matrices* regardless its dynamical order. As such, most advanced control theories are and continue to be developed in the state space setting. The main idea of this section is to given a brief introduction to this technique.

5.2. State space representation of discrete systems

Consider the simplest discrete system (1st order) with a transfer function in the z-domain

$$D(z)=\frac{Y(z)}{U(z)}=\frac{b_0}{z+a_0} \quad \Longrightarrow \quad (z+a_0)\,Y(z)=b_0U(z)$$

$$\Longrightarrow \quad y(k+1)+a_0y(k)=b_0u(k)$$

Define a so-called state variable $x(k)=y(k)$

$$\Longrightarrow \quad x(k+1)=y(k+1)=-a_0y(k)+b_0u(k)=-a_0x(k)+b_0u(k)$$

$$\Rightarrow \quad \begin{cases} x(k+1)=-a_0\,x(k)+b_0u(k) \\ y(k)=x(k) \end{cases} \Longrightarrow \quad \begin{cases} \mathbf{x}(k+1)=\mathbf{A}\,\mathbf{x}(k)+\mathbf{B}\,u(k) \\ y(k)=\mathbf{C}\,\mathbf{x}(k) \end{cases}$$

It is trivial ! Nonetheless, the same idea can be used to convert more complicated transfer functions into a corresponding state space representation…

5.3. Let us consider a 2nd order discrete system with a transfer function in the z-domain

$$D(z)=\frac{Y(z)}{U(z)}=\frac{b_0}{z^2+a_1z+a_0} \quad \Longrightarrow \quad \left(z^2+a_1z+a_0\right)Y(z)=b_0U(z)$$

$$\Longrightarrow \quad y(k+2)+a_1y(k+1)+a_0y(k)=b_0u(k)$$

Define a set of so-called state variables,

$$x_1(k)=y(k)$$
$$x_2(k)=y(k+1)$$

$$\Longrightarrow$$

$$x_1(k+1)=y(k+1)=x_2(k)$$
$$x_2(k+1)=y(k+2)=-a_1y(k+1)-a_0y(k)+b_0u(k)$$
$$=-a_1x_2(k)-a_0x_1(k)+b_0u(k)$$

$$\Longrightarrow \quad \begin{cases} \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix}=\begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}+\begin{bmatrix} 0 \\ b_0 \end{bmatrix}u(k) \\ y(k)=x_1(k)=\begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \end{cases} \quad \Longrightarrow \quad \begin{cases} \mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k) \\ y(k)=\mathbf{C}\mathbf{x}(k) \end{cases}$$

5.4. Next, consider another type of 2nd order systems with a transfer function in the z-domain

$$D(z)=\frac{Y(z)}{U(z)}=\frac{b_1 z+b_0}{z^2+a_1 z+a_0} \quad \Longrightarrow \quad \left(z^2+a_1 z+a_0\right)Y(z)=\left(b_1 z+b_0\right)U(z)$$

$$\Longrightarrow \quad y(k+2)+a_1 y(k+1)+a_0 y(k)=b_1 u(k+1)+b_0 u(k)$$

Define $x_1(k)=y(k), \quad x_2(k)=y(k+1)-b_1 u(k)$

$$\Longrightarrow \quad \begin{aligned} x_1(k+1) &= y(k+1)=x_2(k)+b_1 u(k) \\ x_2(k+1) &= y(k+2)-b_1 u(k+1) \\ &= -a_1 y(k+1)-a_0 y(k)+b_1 u(k+1)+b_0 u(k)-b_1 u(k+1) \\ &= -a_1 x_2(k)-a_1 b_1 u(k)-a_0 x_1(k)+b_0 u(k) \\ &= -a_0 x_1(k)-a_1 x_2(k)+\left(b_0-a_1 b_1\right)u(k) \end{aligned}$$

$$\Longrightarrow \quad \begin{cases} \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix}=\begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}+\begin{bmatrix} b_1 \\ b_0-a_1 b_1 \end{bmatrix}u(k) \\ y(k)=x_1(k)=\begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \end{cases} \quad \Longrightarrow \quad \begin{cases} \mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k) \\ y(k)=\mathbf{C}\mathbf{x}(k) \end{cases}$$

5.5. We now consider a general system characterized by

$$D(z)=\frac{Y(z)}{U(z)}=\frac{b_m z^m+b_{m-1}z^{m-1}+\cdots+b_1 z+b_0}{z^n+a_{n-1}z^{n-1}+\cdots+a_1 z+a_0},\quad n\geq m$$

Using the partial fraction technique, we can rewrite this general system as

$$D(z)=\frac{Y(z)}{U(z)}=\underbrace{\frac{b_{1,1}z+b_{0,1}}{z^2+a_{1,1}z+a_{0,1}}}_{\begin{cases}\mathbf{x}_1(k+1)=\mathbf{A}_1\mathbf{x}_1(k)+\mathbf{B}_1u(k)\\ y_1(k)=\mathbf{C}_1\mathbf{x}_1(k)\end{cases}}+\cdots+\underbrace{\frac{b_{1,k}z+b_{0,k}}{z^2+a_{1,k}z+a_{0,k}}}_{\begin{cases}\mathbf{x}_k(k+1)=\mathbf{A}_k\mathbf{x}_k(k)+\mathbf{B}_ku(k)\\ y_k(k)=\mathbf{C}_k\mathbf{x}_k(k)\end{cases}}+\underbrace{\frac{b_0}{z+a_0}}_{\begin{cases}\mathbf{x}_0(k+1)=\mathbf{A}_0\mathbf{x}_0(k)+\mathbf{B}_0u(k)\\ y_0(k)=\mathbf{C}_0\mathbf{x}_0(k)\end{cases}}+d$$

The state space representation of the complicated system is then given by

$$\begin{cases}\mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k)\\ y(k)=\mathbf{C}\mathbf{x}(k)+\mathbf{D}u(k)\end{cases}$$

with

$$\mathbf{x}=\begin{pmatrix}\mathbf{x}_0\\ \mathbf{x}_1\\ \vdots\\ \mathbf{x}_k\end{pmatrix},\quad \mathbf{A}=\begin{bmatrix}\mathbf{A}_0 & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \mathbf{A}_1 & \cdots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_k\end{bmatrix},\quad \mathbf{B}=\begin{bmatrix}\mathbf{B}_0\\ \mathbf{B}_1\\ \vdots\\ \mathbf{B}_k\end{bmatrix},\quad \mathbf{D}=d$$

and $\mathbf{C}=\begin{bmatrix}\mathbf{C}_0 & \mathbf{C}_1 & \cdots & \mathbf{C}_k\end{bmatrix}$

Fortunately, function **`tf2ss`** in MATLAB can do this for us.

5.6. Example

Convert the discrete transfer function in 4.12 into a state space representation.

$$G(z) = \frac{0.5z + 0.5}{(z - 0.5)(z - 0.9)} = \frac{0.5z + 0.5}{z^2 - 1.4z + 0.45}$$

Recall the formula we have derived in 5.4, i.e.,

$$\begin{cases} \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_0 - a_1 b_1 \end{bmatrix} u(k) \\ y(k) = x_1(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \end{cases} \Longrightarrow \begin{cases} \mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}u(k) \\ y(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}u(k) \end{cases}$$

with

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.5 \\ 1.2 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D = 0.$$

Exercise: Verify that for this example,

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

You can verify this using MATLAB. But, show that the above identity is true in general.

5.7. Example

Now, convert the following discrete transfer function into a state space representation,

$$G(z)=\frac{0.5z^2+0.5}{z^2-1.4z+0.45}=\frac{0.5(z^2-1.4z+0.45)+0.7z+0.275}{z^2-1.4z+0.45}=0.5+\frac{0.7z+0.275}{z^2-1.4z+0.45}$$

Again, use the formula in 5.4, i.e.,

$$\begin{cases}\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix}=\begin{bmatrix}0 & 1\\-a_0 & -a_1\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}+\begin{bmatrix}b_1\\b_0-a_1b_1\end{bmatrix}u(k)\\ y(k)=x_1(k)=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}\end{cases}\Longrightarrow\begin{cases}\mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k)\\ y(k)=\mathbf{C}\mathbf{x}(k)+\mathbf{D}u(k)\end{cases}$$

with

$$\mathbf{A}=\begin{bmatrix}0 & 1\\-0.45 & 1.4\end{bmatrix},\quad \mathbf{B}=\begin{bmatrix}0.7\\1.255\end{bmatrix},\quad C=\begin{bmatrix}1 & 0\end{bmatrix},\quad D=0.5.$$

Exercise: Verify that using MATLAB that the eigenvalues of $\mathbf{A}$ are exactly the poles of $G(z)$ or the roots of the denominator of $G(z)$. Prove that it is true in general.

5.8. Digital control system design with state feedback

Once we obtain a state space model for the discrete system, the control system design in the state space setting is straightforward and systematic. We know by now any system can be represented as

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,u(k) \\ y(k) = \mathbf{C}\,\mathbf{x}(k) + \mathbf{D}\,u(k) \end{cases} \qquad (**)$$

Assume that the state variable, i.e., $\mathbf{x}(k)$, is measurable. We can design a state feedback controller that meets design specifications. The controller has the following structure:

$$u(k) = -\mathbf{F}\,\mathbf{x}(k) + J\,r(k)$$

Substituting this into (**), we have

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\left[-\mathbf{F}\mathbf{x}(k) + Jr(k)\right] = (\mathbf{A} - \mathbf{BF})\mathbf{x}(k) + \mathbf{B}Jr(k) \\ y(k) = \mathbf{C}\;\mathbf{x}(k) + \mathbf{D}\left[-\mathbf{F}\mathbf{x}(k) + Jr(k)\right] = (\mathbf{C} - \mathbf{DF})\mathbf{x}(k) + \mathbf{D}Jr(k) \end{cases} \qquad (3^*)$$

5.9. State feedback design via pole placement

The closed-loop system (3^*) under the the state feedback control again can be written as

$$\begin{cases} \mathbf{x}(k+1) = \overline{\mathbf{A}}\,\mathbf{x}(k) + \overline{\mathbf{B}}\,r(k) \\ y(k) = \overline{\mathbf{C}}\,\mathbf{x}(k) + \overline{\mathbf{D}}\,r(k) \end{cases} \quad \text{with} \quad \begin{aligned} \overline{\mathbf{A}} &= \mathbf{A} - \mathbf{BF}\,, \quad \overline{\mathbf{B}} = \mathbf{B}J \\ \overline{\mathbf{C}} &= \mathbf{C} - \mathbf{DF}\,, \quad \overline{\mathbf{D}} = \mathbf{D}J \end{aligned}$$

which has a new transfer function

$$\begin{aligned} H(z) = \overline{\mathbf{C}}(z\mathbf{I} - \overline{\mathbf{A}})^{-1}\overline{\mathbf{B}} + \overline{\mathbf{D}} &= (\mathbf{C} - \mathbf{DF})(z\mathbf{I} - \mathbf{A} + \mathbf{BF})^{-1}\mathbf{B}J + \mathbf{D}J \\ &= \left[(\mathbf{C} - \mathbf{DF})(z\mathbf{I} - \mathbf{A} + \mathbf{BF})^{-1}\mathbf{B} + \mathbf{D}\right]J \end{aligned}$$

The design procedure is very straightforward:

- Find an $\mathbf{F}$ is to such that the eigenvalues of $\mathbf{A} - \mathbf{BF}$ are in the locations, which would yield desired overshoot, settling time, etc. (how? Pole placement…)
- Find J such that the static gain or DC gain from r to y equal to unity.

5.10. Design example

Re-consider the example in 4.12. The motor modelled with a sampling period $T = 0.1s$ has a transfer function:

$$G(z) = \frac{0.5z + 0.5}{(z - 0.5)(z - 0.9)}$$

We are now to design a digital control system with a state feedback control law such that the resulting system output tracks a step reference with a settling time less than $1s$ with an overshoot less 5% and zero steady state error.

It was done in 5.6 that the state space description of the above system is given by

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}u(k) \\ y(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}u(k) \end{cases}$$

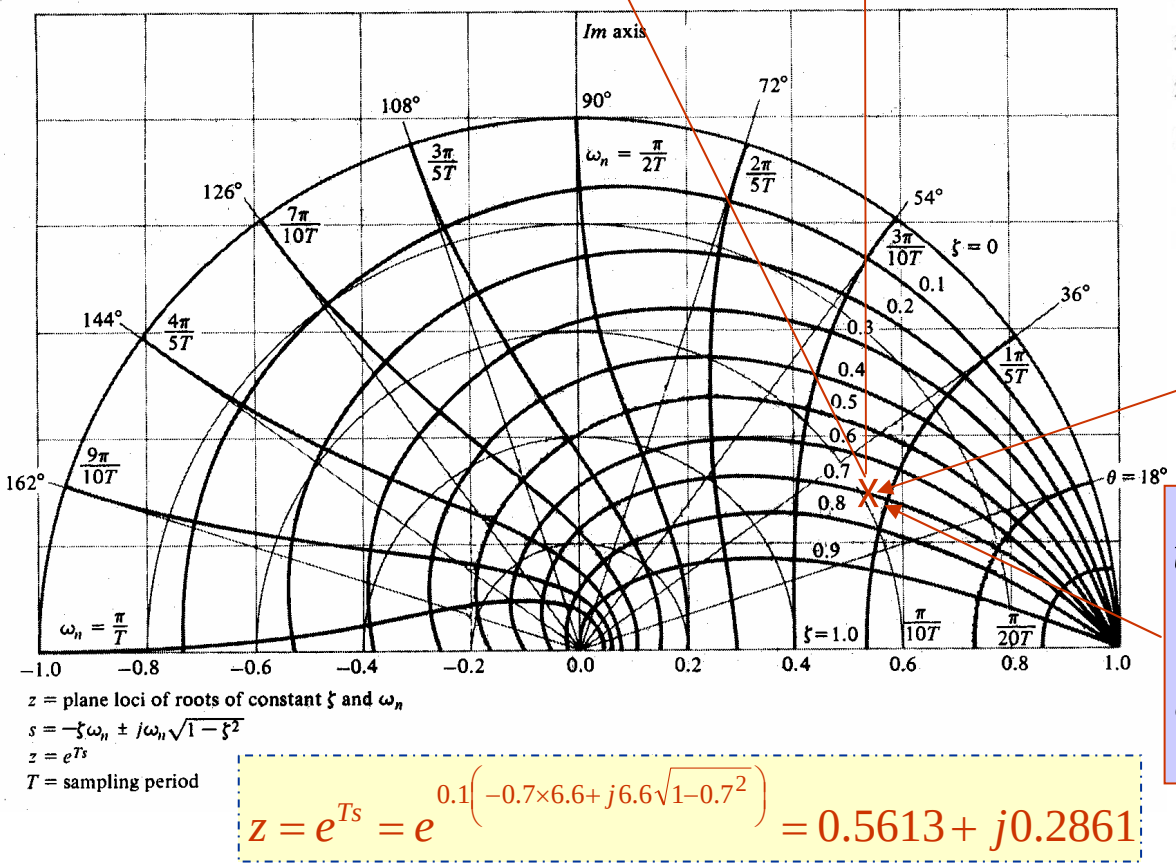
with

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.5 \\ 1.2 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D = 0.$$

Step 0: Translate the design specs into the requirements on desired pole locations:

$$\text{desired characteristic polynomial} = z^2 - 1.04z + 0.338$$

$$z_p^* = 0.52 - j0.26 \qquad z_p = 0.52 + j0.26$$

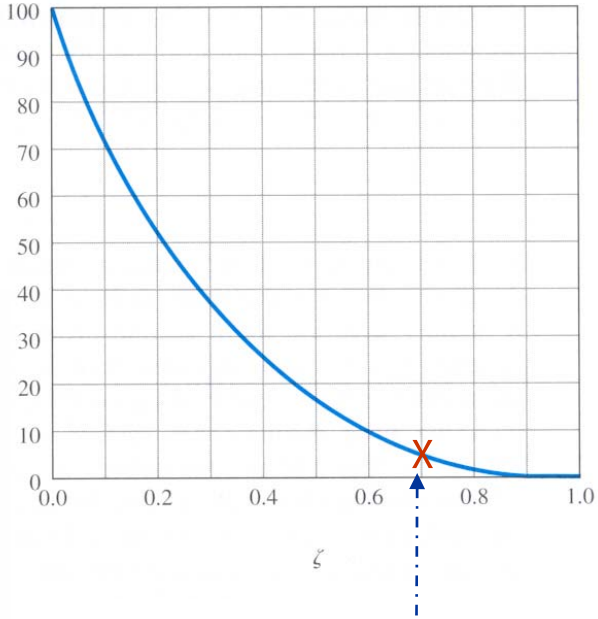


z = plane loci of roots of constant ζ and ω_n

$s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$

$z = e^{Ts}$

T = sampling period



$$\zeta \geq 0.7 \quad \Rightarrow \quad \zeta = 0.7$$

$$t_s \cong \frac{4.6}{\zeta\omega_n} = 1 \quad \Rightarrow$$

$$\omega_n \cong \frac{4.6}{\zeta t_s} = \frac{4.6}{0.7 \times 1} = 6.6 = \frac{\pi}{4.8T}$$

$$z = e^{Ts} = e^{0.1\left(-0.7\times 6.6 + j6.6\sqrt{1-0.7^2}\right)} = 0.5613 + j0.2861$$

Step 1: Find an **F** is to such that the eigenvalues of **A – BF** are in the locations. Let

$$\mathbf{F} = \begin{bmatrix} f_1 & f_2 \end{bmatrix} \quad \text{Note that } \mathbf{F} \text{ is a row vector with } n \text{ elements.}$$

$$\overline{\mathbf{A}} = \mathbf{A} - \mathbf{BF} = \begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix} - \begin{bmatrix} 0.5 \\ 1.2 \end{bmatrix} \begin{bmatrix} f_1 & f_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix} - \begin{bmatrix} 0.5 f_1 & 0.5 f_2 \\ 1.2 f_1 & 1.2 f_2 \end{bmatrix} = \begin{bmatrix} -0.5 f_1 & 1 - 0.5 f_2 \\ -1.2 f_1 - 0.45 & 1.4 - 1.2 f_2 \end{bmatrix}$$

$$\det(z\mathbf{I} - \overline{\mathbf{A}}) = \det \begin{bmatrix} z + 0.5 f_1 & -1 + 0.5 f_2 \\ 0.45 + 1.2 f_1 & z - 1.4 + 1.2 f_2 \end{bmatrix} = z^2 + (0.5 f_1 + 1.2 f_2 - 1.4) z + (0.45 + 0.5 f_1 - 0.225 f_2)$$

$$= \text{desired characteristic polynomial} = z^2 - 1.04 z + 0.338$$

$$\begin{cases} 0.5 f_1 + 1.2 f_2 - 1.4 = -1.04 \\ 0.45 + 0.5 f_1 - 0.225 f_2 = 0.338 \end{cases} \Longleftarrow \begin{cases} f_1 = -0.0749 \\ f_2 = 0.3312 \end{cases} \Longleftarrow \mathbf{F} = \begin{bmatrix} -0.0749 & 0.3312 \end{bmatrix}$$

In real life, the function **`place`** in MATLAB can do all the tedious calculations for us…

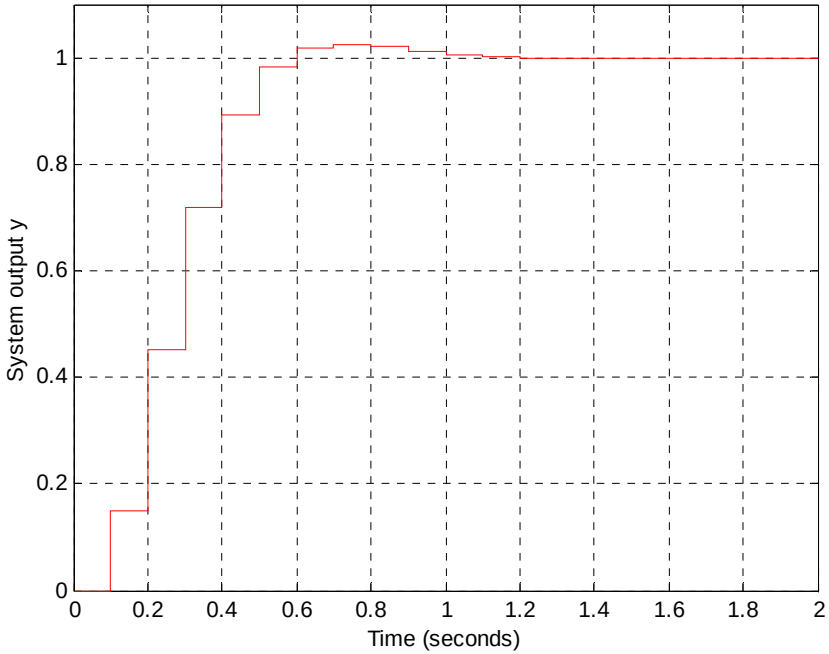
Step 2: Find J such that the DC gain from r to y equal to unity. Compute the DC gain for

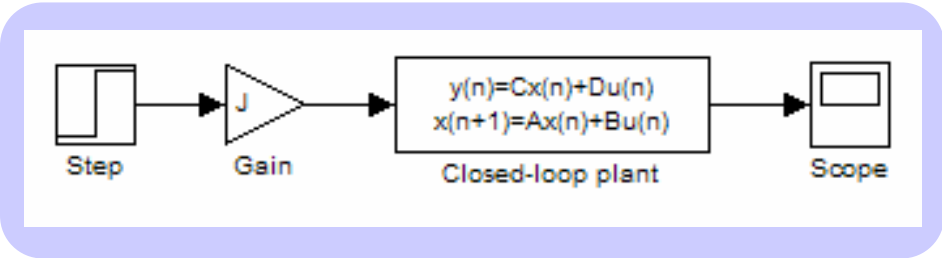
$$H(1) = \overline{\mathbf{C}}(\mathbf{I} - \overline{\mathbf{A}})^{-1}\overline{\mathbf{B}} + \overline{\mathbf{D}} = [(\mathbf{C} - \mathbf{DF})(\mathbf{I} - \mathbf{A} + \mathbf{BF})^{-1}\mathbf{B} + \mathbf{D}]J$$

$$= \left\{ \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0.9625 & -0.8344 \\ 0.3601 & -0.0025 \end{bmatrix}^{-1} \begin{bmatrix} 0.5 \\ 1.2 \end{bmatrix} \right\} J = 3.3557\,J = 1 \quad \Rightarrow \quad J = 0.298$$

We finally note that all computations in the above design can be easily done using MATLAB…

Let us now verify our design use MATLAB SIMULINK.





Overshoot = 2.5% and Settling time < 1 s

Perfect! This is the beauty of the state space design!

5.11. State estimator

Note that in the state feedback design, we have assume the state variable $\mathbf{x}$ of the plant being available for feedback (see 5.8). Unfortunately, this is usually not the case in most practical situations. In real life, usually, we can only access to the system output, i.e., y, instead. In order to implement the beauty of the state feedback control, we will have no choice but to estimate the state variables using only the system output and hope the implementation using such an estimation works, which turns out indeed the case. As such, implementing a state feedback control through a state estimator becomes the standard approach in modern control nowadays.

The first question one would ask is: Can we estimate the state variables using only system output? The answer is yes, fortunately, for most of systems and thus we will take it as doable for all situations considered in our class. The second question comes naturally: How can we estimate the state? The answer is again pretty simple — copy or duplicate of the original dynamical equation.

5.12. Estimator design

Let's be a copy cat for a while. Consider a system described by a state space representation:

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,u(k), \quad \mathbf{x}_0 = \mathbf{x}(0), \quad y(k) = \mathbf{C}\,\mathbf{x}(k) + \mathbf{D}\,u(k) \qquad (\$)$$

Of course, if the state $\mathbf{x}$ is unknown, usually, the initial condition $\mathbf{x}_0$ is unknown too. In order to estimate the state $\mathbf{x}$ (using a computer), it is very natural to follow its dynamics. Thus, let us try

$$\hat{\mathbf{x}}(k+1) = \mathbf{A}\,\hat{\mathbf{x}}(k) + \mathbf{B}\,u(k), \quad \hat{\mathbf{x}}_0 = \hat{\mathbf{x}}(0) \qquad (2\$)$$

Will this work? Let us see whether the estimated state will follow the true state. Define the estimation error as the difference between the true state and the estimated state, i.e.,

$$\mathbf{e}(k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k) \quad \Rightarrow$$

$$\mathbf{e}(k+1) = \mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,u(k) - \mathbf{A}\,\hat{\mathbf{x}}(k) - \mathbf{B}\,u(k) = \mathbf{A}\,\mathbf{e}(k)$$

Will this error $\mathbf{e}(k)$ go to zero as k progresses?

5.13. Example

Recall the motor example (see 5.6), i.e.,

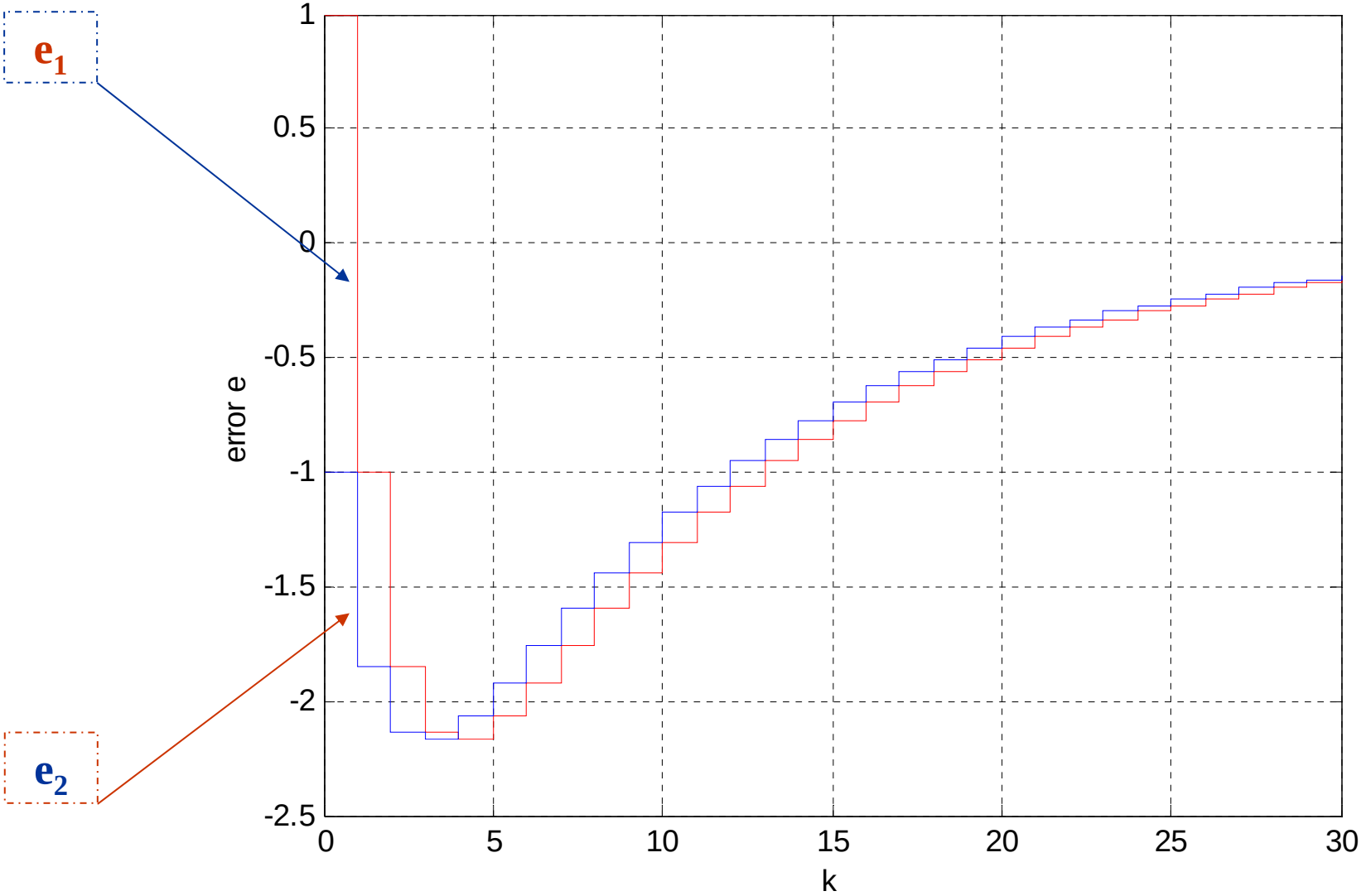
$$\mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\mathbf{x}(k)+\begin{bmatrix}0.5\\ 1.2\end{bmatrix}u(k),\quad \mathbf{x}_0=\begin{pmatrix}1\\ -1\end{pmatrix},\quad y(k)=\begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x}(k)$$

$$\Longrightarrow\quad \hat{\mathbf{x}}(k+1)=\mathbf{A}\hat{\mathbf{x}}(k)+\mathbf{B}u(k)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\hat{\mathbf{x}}(k)+\begin{bmatrix}0.5\\ 1.2\end{bmatrix}u(k),\quad \hat{\mathbf{x}}_0=\begin{pmatrix}0\\ 0\end{pmatrix}$$

$$\Longrightarrow\quad \mathbf{e}(k+1)=\mathbf{A}\,\mathbf{e}(k)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\mathbf{e}(k),\quad \mathbf{e}(0)=\mathbf{x}_0-\hat{\mathbf{x}}_0=\begin{pmatrix}1\\ -1\end{pmatrix}$$

$$\mathbf{e}(1)=\mathbf{A}\,\mathbf{e}(0)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\begin{pmatrix}1\\ -1\end{pmatrix}=\begin{pmatrix}1\\ -1.85\end{pmatrix},\quad \mathbf{e}(2)=\mathbf{A}\,\mathbf{e}(1)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\begin{pmatrix}1\\ -1.85\end{pmatrix}=\begin{pmatrix}-1.85\\ -2.14\end{pmatrix},\cdots$$

Computer simulation show on the next page clearly show for this example, the estimation error goes to zero as k is getting larger and larger. It works !


e1
e2
error e
k
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
5
10
15
20
25
30

5.14. Another example

Let us consider another 2nd order example, which is slightly modified from the previous one

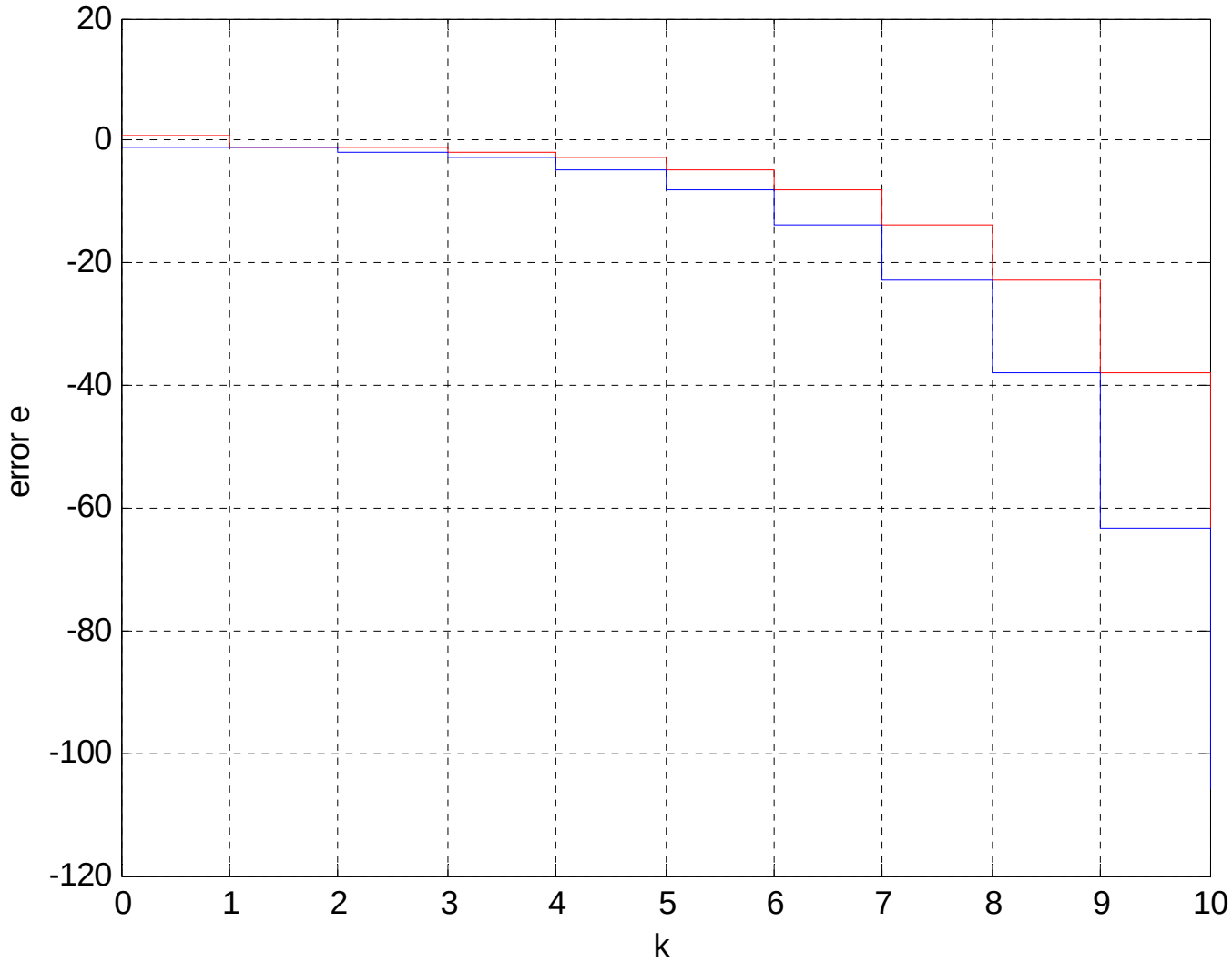
$$\mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k)=\begin{bmatrix}0 & 1\\ 0.45 & 1.4\end{bmatrix}\mathbf{x}(k)+\begin{bmatrix}0.5\\ 1.2\end{bmatrix}u(k),\quad \mathbf{x}_0=\begin{pmatrix}1\\ -1\end{pmatrix},\quad y(k)=\begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x}(k)$$

$$\Longrightarrow\quad \hat{\mathbf{x}}(k+1)=\mathbf{A}\hat{\mathbf{x}}(k)+\mathbf{B}u(k)=\begin{bmatrix}0 & 1\\ 0.45 & 1.4\end{bmatrix}\hat{\mathbf{x}}(k)+\begin{bmatrix}0.5\\ 1.2\end{bmatrix}u(k),\quad \hat{\mathbf{x}}_0=\begin{pmatrix}0\\ 0\end{pmatrix}$$

$$\Longrightarrow\quad \mathbf{e}(k+1)=\mathbf{A}\,\mathbf{e}(k)=\begin{bmatrix}0 & 1\\ 0.45 & 1.4\end{bmatrix}\mathbf{e}(k),\quad \mathbf{e}(0)=\mathbf{x}_0-\hat{\mathbf{x}}_0=\begin{pmatrix}1\\ -1\end{pmatrix}$$

$$\mathbf{e}(1)=\mathbf{A}\,\mathbf{e}(0)=\begin{bmatrix}0 & 1\\ 0.45 & 1.4\end{bmatrix}\begin{pmatrix}1\\ -1\end{pmatrix}=\begin{pmatrix}-1\\ -0.95\end{pmatrix},\quad \mathbf{e}(2)=\mathbf{A}\,\mathbf{e}(1)=\begin{bmatrix}0 & 1\\ 0.45 & 1.4\end{bmatrix}\begin{pmatrix}-1\\ -0.95\end{pmatrix}=\begin{pmatrix}-0.95\\ -1.78\end{pmatrix},\cdots$$

Computer simulation show on the next page clearly show for this example, the estimation error goes to infinity as k is getting larger and larger. It does not work !



WHAT IS WRONG HERE ?

5.15. What is going wrong?

The problem is pretty straightforward. The system in 5.13 is stable while the one in 5.14 is unstable. These can be seen by computing their poles or the eigenvalues of matrix $\mathbf{A}$.

$$\lambda\left\{\begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix}\right\} = 0.5,\ 0.9; \qquad \lambda\left\{\begin{bmatrix} 0 & 1 \\ 0.45 & 1.4 \end{bmatrix}\right\} = -0.2695,\ 1.6695$$

→ Unstable

To fix the problem, we will have to make use of information we can get from the system, i.e., the system output y. We modify the estimator dynamical equation in (2$) as follows:

$$\hat{\mathbf{x}}(k+1) = \mathbf{A}\,\hat{\mathbf{x}}(k) + \mathbf{B}\,u(k) + \mathbf{K}[y(k) - \hat{y}(k)], \quad \hat{y}(k) = \mathbf{C}\hat{\mathbf{x}}(k) + \mathbf{D}u(k), \quad \hat{\mathbf{x}}_0 = \hat{\mathbf{x}}(0)$$

Exercise: Show that the error dynamical equation for the system ($) in 5.12. with the above estimator is given by

$$\mathbf{e}(k+1) = (\mathbf{A} - \mathbf{KC})\,\mathbf{e}(k), \quad \text{where } \mathbf{e}(k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k)$$

We can then choose an appropriate estimator gain $\mathbf{K}$ to make $\mathbf{A} - \mathbf{KC}$ stable.

5.16. Deadbeat estimator

In principle, we can choose the estimator gain **K** such that the eigenvalues of **A – KC** can be placed anywhere we like so long as they are stable. If we place eigenvalues of **A – KC** all in the origin, then the resulting estimator is called an deadbeat estimator, which is capable of rendering the error to zero in n steps, where n is the order of the given system. We now proceed to design a deadbeat for the motor in 5.13, i.e.,

$$\mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{B}u(k)=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}\mathbf{x}(k)+\begin{bmatrix}0.5\\ 1.2\end{bmatrix}u(k),\quad \mathbf{x}_0=\begin{pmatrix}1\\ -1\end{pmatrix},\quad y(k)=\mathbf{C}x(k)=\begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x}(k)$$

Let

$$\mathbf{K}=\begin{bmatrix}k_1\\ k_2\end{bmatrix}$$

Note that K is a column vector with n elements.

$$\Longrightarrow \mathbf{A}-\mathbf{KC}=\begin{bmatrix}0 & 1\\ -0.45 & 1.4\end{bmatrix}-\begin{bmatrix}k_1\\ k_2\end{bmatrix}\begin{bmatrix}1 & 0\end{bmatrix}=\begin{bmatrix}-k_1 & 1\\ -k_2-0.45 & 1.4\end{bmatrix}$$

$$\det(z\mathbf{I}-\mathbf{A}+\mathbf{KC})=\det\begin{bmatrix}z+k_1 & -1\\ k_2+0.45 & z-1.4\end{bmatrix}$$

$$=z^2+(k_1-1.4)z+(0.45-1.4k_1+k_2)$$

$= z^2$ = deadbeat estimator

$$\Rightarrow \begin{cases} k_1 - 1.4 = 0 \\ 0.45 - 1.4k_1 + k_2 = 0 \end{cases} \Rightarrow \begin{cases} k_1 = 1.40 \\ k_2 = 1.51 \end{cases} \Rightarrow \mathbf{K} = \begin{bmatrix} 1.40 \\ 1.51 \end{bmatrix}$$

$$\mathbf{e}(1) = (\mathbf{A} - \mathbf{KC})\,\mathbf{e}(0) = \begin{bmatrix} -1.4 & 1.0 \\ -1.96 & 1.4 \end{bmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} -2.40 \\ -3.36 \end{pmatrix}$$

$$\mathbf{e}(2) = (\mathbf{A} - \mathbf{KC})\,\mathbf{e}(1) = \begin{bmatrix} -1.4 & 1.0 \\ -1.96 & 1.4 \end{bmatrix} \begin{pmatrix} -2.40 \\ -3.36 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

deadbeat

Exercise: The MATLAB function **F = `place(A,B,poles)`**, where poles is a vector containing the desired poles, is meant for finding state feedback gain **F**. Show that the same function can be adopted to find the estimator gain **K**, where

K = `place(A',C',poles)'`

Then, design a deadbeat estimator for the unstable system given in 5.14 and verify your result by computing the estimation error up to 3 steps.

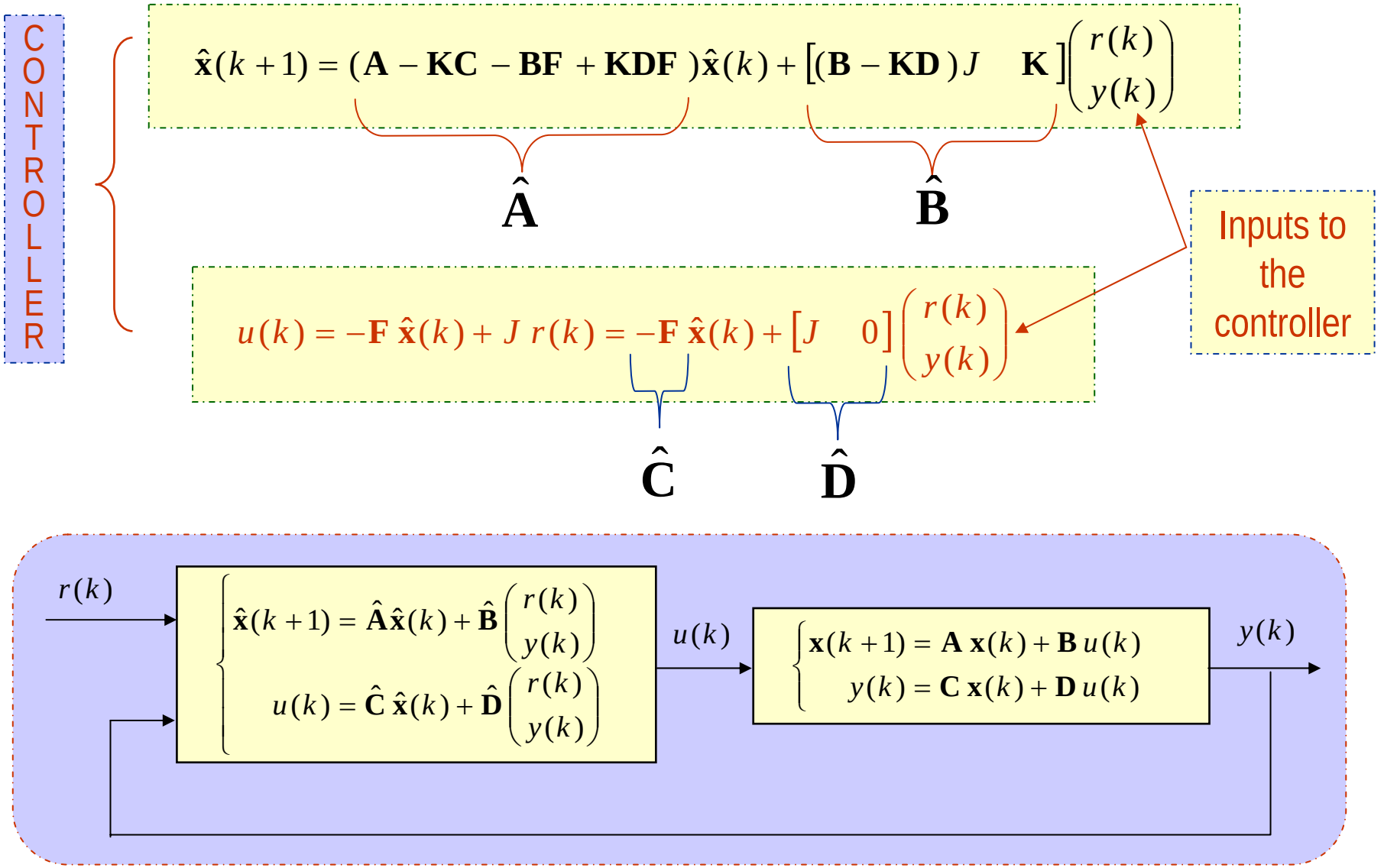
5.18. Control system design with state feedback and estimator

Instead of implementing the control law with a state (not available) feedback, we can realize it by replacing the true state with the estimated state thru an estimator, which only requires the information of the system output. Such a control system is practically feasible and has become the standard approach in modern control theory using state space approach. To summarize, we recall

$$u(k) = -\mathbf{F}\,\mathbf{x}(k) + J\,r(k) \qquad \Rightarrow \qquad u(k) = -\mathbf{F}\,\hat{\mathbf{x}}(k) + J\,r(k)$$

$$\begin{aligned}
\hat{\mathbf{x}}(k+1) &= \mathbf{A}\,\hat{\mathbf{x}}(k) + \mathbf{B}\,u(k) + \mathbf{K}[y(k) - \hat{y}(k)] \\
&= \mathbf{A}\,\hat{\mathbf{x}}(k) + \mathbf{B}\,u(k) + \mathbf{K}y(k) - \mathbf{K}[\mathbf{C}\hat{\mathbf{x}}(k) - \mathbf{D}u(k)] \\
&= (\mathbf{A} - \mathbf{KC})\hat{\mathbf{x}}(k) + (\mathbf{B} - \mathbf{KD})u(k) + \mathbf{K}y(k) \\
&= (\mathbf{A} - \mathbf{KC})\hat{\mathbf{x}}(k) + (\mathbf{B} - \mathbf{KD})[-\mathbf{F}\hat{\mathbf{x}}(k) + J\,r(k)] + \mathbf{K}y(k) \\
&= (\mathbf{A} - \mathbf{KC} - \mathbf{BF} + \mathbf{KDF})\hat{\mathbf{x}}(k) + (\mathbf{B} - \mathbf{KD})J\,r(k) + \mathbf{K}y(k) \\
&= (\mathbf{A} - \mathbf{KC} - \mathbf{BF} + \mathbf{KDF})\hat{\mathbf{x}}(k) + \begin{bmatrix}\mathbf{K} & (\mathbf{B} - \mathbf{KD})J\end{bmatrix}\begin{pmatrix} y(k) \\ r(k)\end{pmatrix}
\end{aligned}$$

5.19. Block diagram of the state space control system



5.20. Summary of control system design procedure using estimator

Step 1: Determine the desired closed-loop pole locations from the given design specs.

Step 2: Find the state feedback gain $\mathbf{F}$ (row vector) such that the eigenvalues of $\mathbf{A} - \mathbf{BF}$ coincides with the desired pole locations obtained in Step 1.

Step 3: Find the constant gain J such that $[(\mathbf{C} - \mathbf{DF})(\mathbf{I} - \mathbf{A} + \mathbf{BF})^{-1}\mathbf{B} + \mathbf{D}]J = 1$.

Step 4: Find the estimator gain $\mathbf{K}$ (column vector) such that the eigenvalues of $\mathbf{A} - \mathbf{KC}$ are in pre-specified locations (or place all at 0 otherwise to yield a deadbeat estimator).

Step 5: Compute $\hat{\mathbf{A}} = \mathbf{A} - \mathbf{KC} - \mathbf{BF} + \mathbf{KDF}$, $\hat{\mathbf{B}} = [(\mathbf{B} - \mathbf{KD})J \quad \mathbf{K}]$, $\hat{\mathbf{C}} = -\mathbf{F}$, $\hat{\mathbf{D}} = [J \quad 0]$.

Step 6: The digital controller in state space form:

$$\hat{\mathbf{x}}(k+1) = \hat{\mathbf{A}}\hat{\mathbf{x}}(k) + \hat{\mathbf{B}}\begin{pmatrix} r(k) \\ y(k) \end{pmatrix}, \quad u(k) = \hat{\mathbf{C}}\,\hat{\mathbf{x}}(k) + \hat{\mathbf{D}}\begin{pmatrix} r(k) \\ y(k) \end{pmatrix}$$

Remark: All computations in Steps 2 to 6 can be done in MATLAB.

5.21. Design example

Re-consider the example in 5.10. The motor modelled with a sampling period $T = 0.1s$ has a state space representation (see 5.10):

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}u(k) \\ y(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}u(k) \end{cases}$$ with $$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -0.45 & 1.4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.5 \\ 1.2 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D = 0.$$

We are now to design a digital control system with a deadbeat estimator feedback control law such that the resulting system output tracks a step reference with a settling time less than $1s$ with an overshoot less 5% and zero steady state error.

Steps 1 to 3 were done in 5.10, which gave $$\mathbf{F} = \begin{bmatrix} -0.0749 & 0.3312 \end{bmatrix}, \quad J = 0.298$$

Step 4 was done in 5.16, which gave a deadbeat estimator gain $$\mathbf{K} = \begin{bmatrix} 1.40 \\ 1.51 \end{bmatrix}$$

Steps 5 & 6: Compute the controller in the state space form:

$$\hat{\mathbf{A}} = \mathbf{A} - \mathbf{KC} - \mathbf{BF} + \mathbf{KDF} = \begin{bmatrix} -1.3625 & 0.8344 \\ -1.8701 & 1.0026 \end{bmatrix}$$

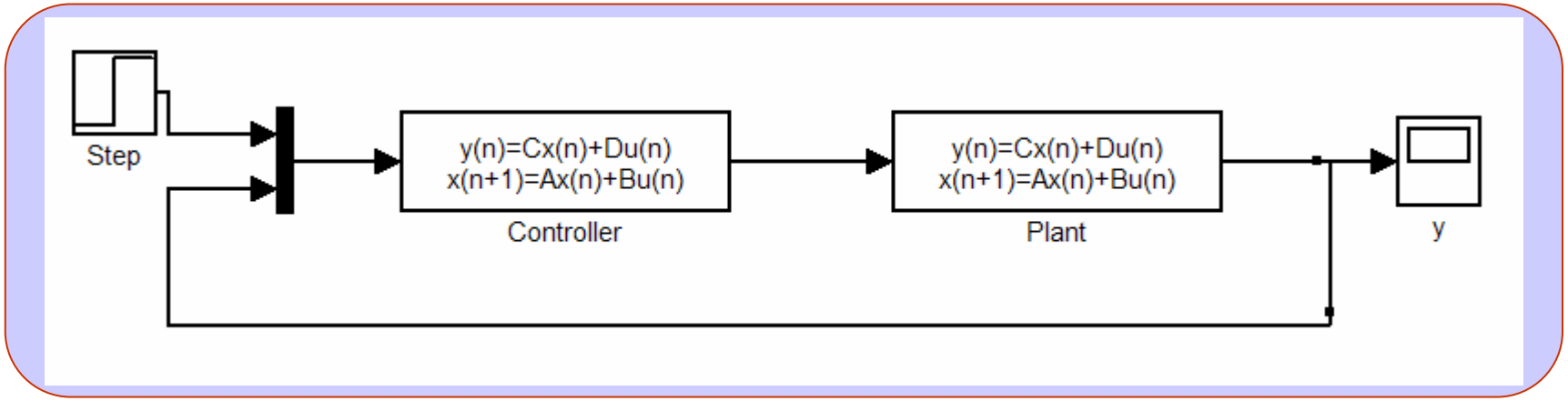
$$\hat{\mathbf{B}} = \begin{bmatrix} (\mathbf{B} - \mathbf{KD})J & \mathbf{K} \end{bmatrix} = \begin{bmatrix} 0.1490 & 1.40 \\ 0.3576 & 1.51 \end{bmatrix}$$

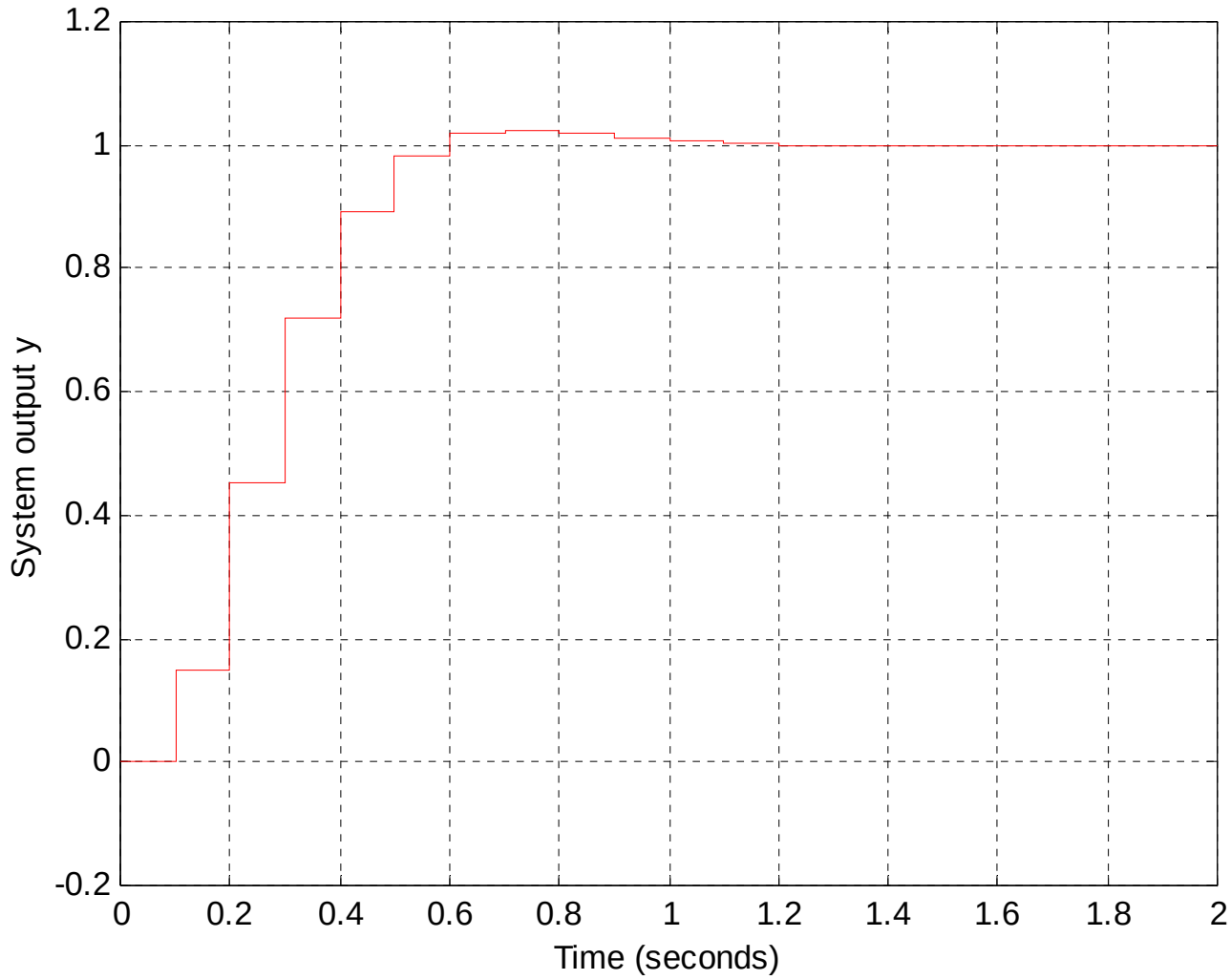
$$\hat{\mathbf{C}} = \begin{bmatrix} 0.0749 & -0.3312 \end{bmatrix}$$

$$\hat{\mathbf{D}} = \begin{bmatrix} J & 0 \end{bmatrix} = \begin{bmatrix} 0.2980 & 0 \end{bmatrix}$$

$$\begin{cases} \hat{\mathbf{x}}(k+1) = \hat{\mathbf{A}}\hat{\mathbf{x}}(k) + \hat{\mathbf{B}} \begin{pmatrix} r(k) \\ y(k) \end{pmatrix} \\ u(k) = \hat{\mathbf{C}}\,\hat{\mathbf{x}}(k) + \hat{\mathbf{D}} \begin{pmatrix} r(k) \\ y(k) \end{pmatrix} \end{cases}$$

Let us verify our design again using MATLAB SIMULINK…





Our design is perfect. This once again shows the beauty of the state space approach.

Homework assignment 3 *(hand in your solution to my office next week)**:***

Reconsider the hard disk drive servo system in the Homework Assignments 1 and 2, i.e.,

$$G(s) = \frac{Y(s)}{U(s)} = \frac{6 \times 10^7}{s^2}$$

where y is the displacement of the read/ write head in micrometer and u is actuator input voltage in volts. The sampling period is chosen to be T = 1/10000 = 0.0001 seconds. It is required to design an appropriate compensator such that the resulting closed-loop system has an overshoot less than 5% and a settling time less than 2 milliseconds due to a step reference of 1 micrometer.

Design a digital control system using the state space approach with a deadbeat estimator. Show all your design procedures together with simulation results. Again, make use of MATLAB and SIMULINK whenever is possible.

By now, you have designed the same system using four different approaches. Please comment the advantages and disadvantages of these methods. Which one would you recommend?

~~ THE ~ END ~~