Chapter 8: Discretization of Continuous-time Systems

July 24, 2015

Outline

Discretization

Use of discretization

Many systems in the real world are continuous-time systems: chemical reactions, rocket trajectories, power plants, ice cap melting... Computers, however, are mainly digital. If we want to **simulate** the continuous system with a digital device, we need a method to convert the continuous model into a discrete one. This conversion is called "discretization". Discretization also comes in handy when a continuous **filter with useful properties** has been designed and a discrete filter with the same properties is required. In addition, discretization is also used in **control**. For instance, in order to design a digital controller it is necessary to have a discrete model of the plant, which is typically given in the form of differential equations.

Discretization

Problem statement

While converting, some information of the continuous model may be lost due to the different nature of the systems. It is important that the loss of information is minimized. Each discretization method has its own qualities and they will all lead to different discrete representations of the same continuous system.

Discretization methods discussed in this chapter

- Numerical Integration
- Impulse Invariant Method
- Zero-pole Equivalent
- Hold Equivalents

Outline

Outline

Numerical Integration

General approach

For a given continuous-time integrator

$$H(s) = \frac{U(s)}{E(s)} = \frac{1}{s} \quad \Leftrightarrow \quad \dot{u}(t) = e(t) \quad \Leftrightarrow \quad u(t) = \int_0^t e(\tau)\mathrm{d}\tau$$

its output at $t = kT_s$ can be written as follows:

$$u(kT_s) = \int_0^{(k-1)T_s} e(\tau)\mathrm{d}\tau + \int_{(k-1)T_s}^{kT_s} e(\tau)\mathrm{d}\tau$$

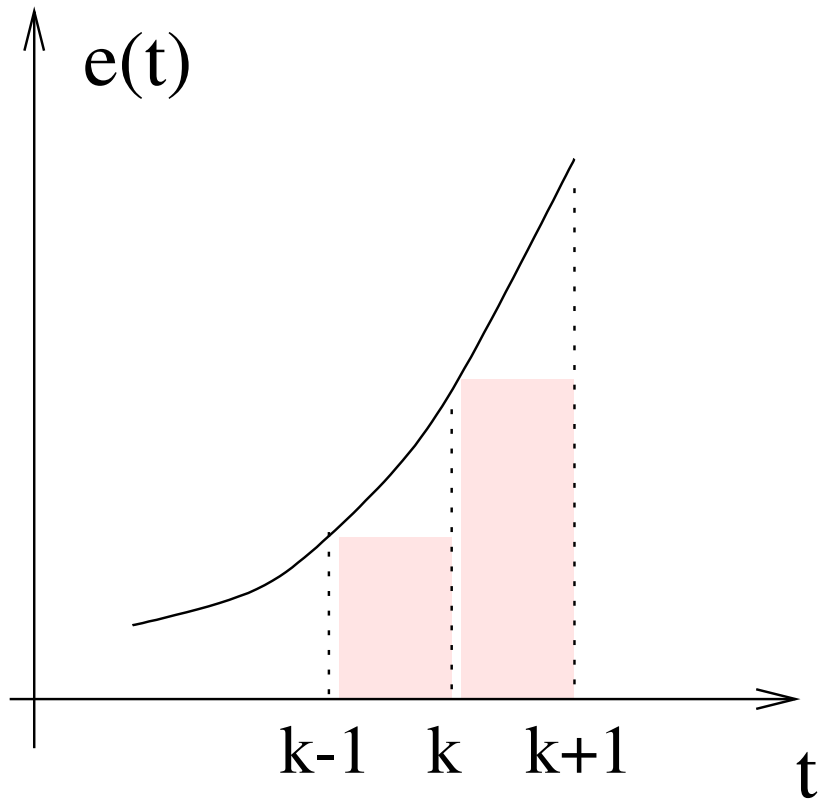
$$u(kT_s) = u((k-1)T_s) + \begin{cases} \text{area of } e(\tau) \\ \text{over } (k-1)T_s \leq \tau < kT_s \end{cases} \qquad (8.1)$$

where T_s is the sampling time.

Forward rectangular rule (=Forward Euler)

General approach

The area is approximated by the rectangle looking **forward** from $(k-1)$ toward k with an amplitude equal to the value of the function at $(k-1)$.



Forward rectangular rule

Mathematical approach

From equation (8.1) we have that

$$u(kT_s) = u\big((k-1)T_s\big) + T_s e\big((k-1)T_s\big)$$

By taking the $\mathcal{Z}$-transform we obtain the discrete equivalent of H(s),

$$U(z) = z^{-1}U(z) + T_s z^{-1}E(z) \quad \Leftrightarrow$$
$$(1-z^{-1})U(z) = T_s z^{-1}E(z) \quad \Leftrightarrow \quad \frac{z-1}{T_s}U(z) = E(z)$$

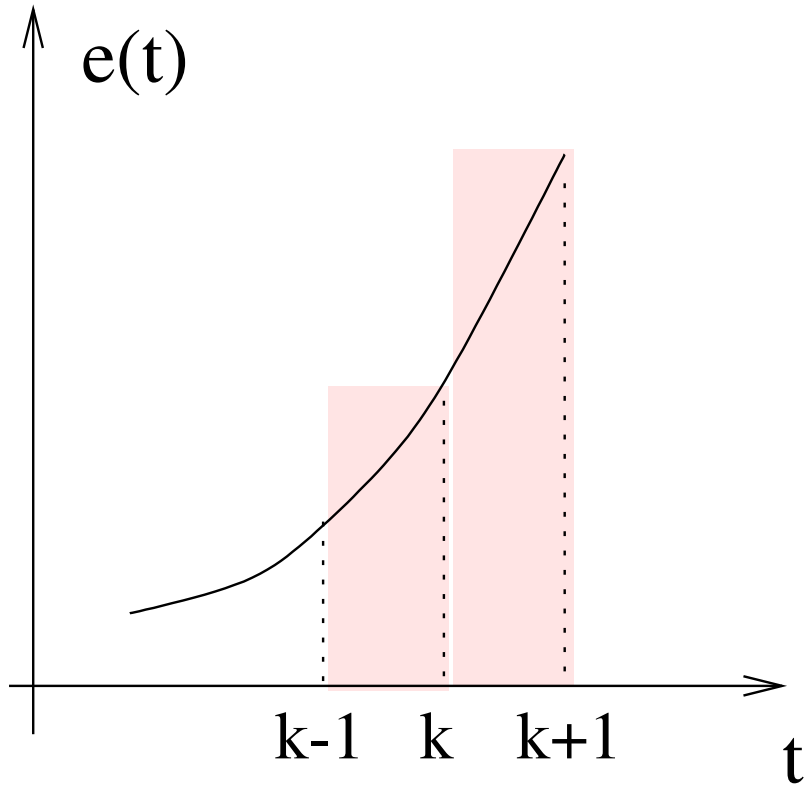
This means that we need to apply the following substitution in order to discretize a given continuous-time transfer function:

$$s \leftarrow \frac{z-1}{T_s} \quad (8.2)$$

Backward rectangular rule (=Backward Euler)

General approach

The area is approximated by the rectangle looking **backward** from k toward $(k-1)$ with an amplitude equal to the value of the function at k.



Backward rectangular rule

Mathematical approach

From equation (8.1) we have that

$$u(kT_s) = u\big((k-1)T_s\big) + T_s e\big(kT_s\big)$$

By taking the $\mathcal{Z}$-transform we obtain the discrete equivalent of H(s),

$$U(z) = z^{-1}U(z) + T_s E(z) \quad \Leftrightarrow$$
$$(1 - z^{-1})U(z) = T_s E(z) \quad \Leftrightarrow \quad \frac{z-1}{zT_s}U(z) = E(z)$$

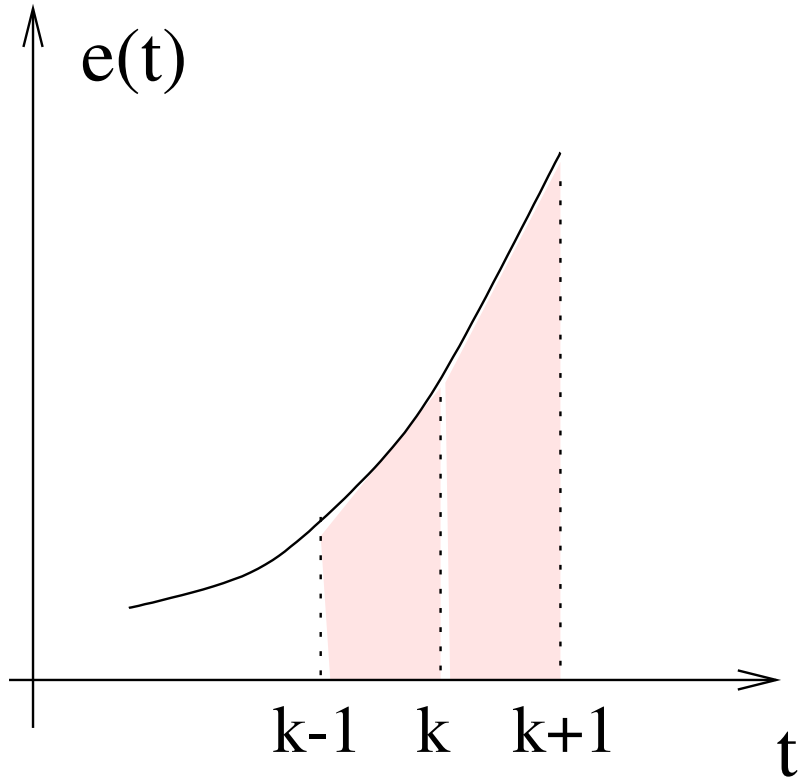
This means that we need to apply the following substitution in order to discretize a given continuous-time transfer function:

$$s \leftarrow \frac{z-1}{zT_s} \quad (8.3)$$

Bilinear rule (= trapezoidal or Tustin rule)

General approach

This method makes use of the area of the **trapezoid** formed by the average of the selected rectangles used in the forward and backward rectangular rule.Thus the amplitude equal to the value of the function at $(k-1)$ and the amplitude equal to the value of the function at (k) are connected by a line as shown in the illustration.



Bilinear transformation

Mathematical approach

From equation (8.1) we have that

$$u(kT_s) = u\big((k-1)T_s\big) + T_s \frac{e(kT_s)+e\Big((k-1)T_s\Big)}{2}$$

By taking the $\mathcal{Z}$-transform we obtain the discrete equivalent of H(s),

$$U(z) = z^{-1}U(z) + \frac{T_s}{2}\big(E(z) + z^{-1}E(z)\big) \quad \Leftrightarrow$$
$$(1-z^{-1})U(z) = \frac{T_s}{2}(1+z^{-1})E(z) \quad \Leftrightarrow \quad \frac{2}{T_s}\frac{z-1}{z+1}U(z) = E(z)$$

This means that we need to apply the following substitution in order to discretize a given continuous-time transfer function:

$$s \leftarrow \frac{2}{T_s}\frac{z-1}{z+1} \quad (8.4)$$

Bilinear rule

Example

Given:

$$H(s) = \frac{s+1}{0.1s+1}$$

We now apply substitution (8.4):

$$H(z) = \frac{(2+T)(T-2)z^{-1}}{(0.2+T)+(T-0.2)z^{-1}}$$

Using T=0.25s, this results in:

$$H(z) = \frac{5(z-0.7778)}{z+0.1111}$$

Discretization of state-space models

General approach applied on Forward Euler

Given the following continuous-time model in state-space form:

$$\dot{x} = Ax + Bu \qquad sX = AX + BU$$
$$y = Cx + Du \qquad Y = CX + DU$$

If we use the Forward Euler method, we have that s is replaced by $\frac{z-1}{T_s}$, so we can find the discrete-time equivalent as follows:

$$\frac{z-1}{T_s}X = AX + BU$$
$$Y = CX + DU$$

Discretization of state-space models

General approach applied on Forward Euler

Which leads to the following

$$zX = (I - AT_s)X + BT_sU$$
$$x(k+1) = (I - AT_s)x(k) + (BT_s)u(k)$$
$$x(k+1) = A_d x(k) + B_d u(k)$$

The output equation $Y = CX + DU$ remains.

Discretization of state-space models

State-space models

A similar calculation can be done for the backward rectangular rule and the bilinear transformation resulting in the following table:

	Euler	backward rect.	bilinear transf.
A_d	$I + AT_s$	$(I - AT_s)^{-1}$	$(I - \frac{AT_s}{2})^{-1}(I + \frac{AT_s}{2})$
B_d	BT_s	$(I - AT_s)^{-1}BT_s$	$(I - \frac{AT_s}{2})^{-1}BT_s$
C_d	C	$C(I - AT_s)^{-1}$	$C(I - \frac{AT_s}{2})^{-1}$
D_d	D	$D + C(I - AT_s)^{-1}BT_s$	$D + C(I - \frac{AT_s}{2})^{-1}\frac{BT_s}{2}$

Stability of the numerical integration methods

Stability

As already mentioned, a discrete system is stable when its poles lie within the unit circle of the z-plane and a continuous system is stable when its poles have a negative real part in the s-plane. Subsequently the $(s = j\omega)$-axis is the boundary between poles of stable and unstable continuous systems.

Each of the discretization methods can be considered as a map from the s-plane to the z-plane. It is interesting to know how the $j\omega$-axis is mapped by every rule and where the stable part of the s-plane appears in the z-plane. This can be realized by solving formulas (8.2-8.3) to z and replacing s by $j\omega$.

Stability of the numerical integration methods

Boudaries of the stable regions

Expressions of z in terms of s:

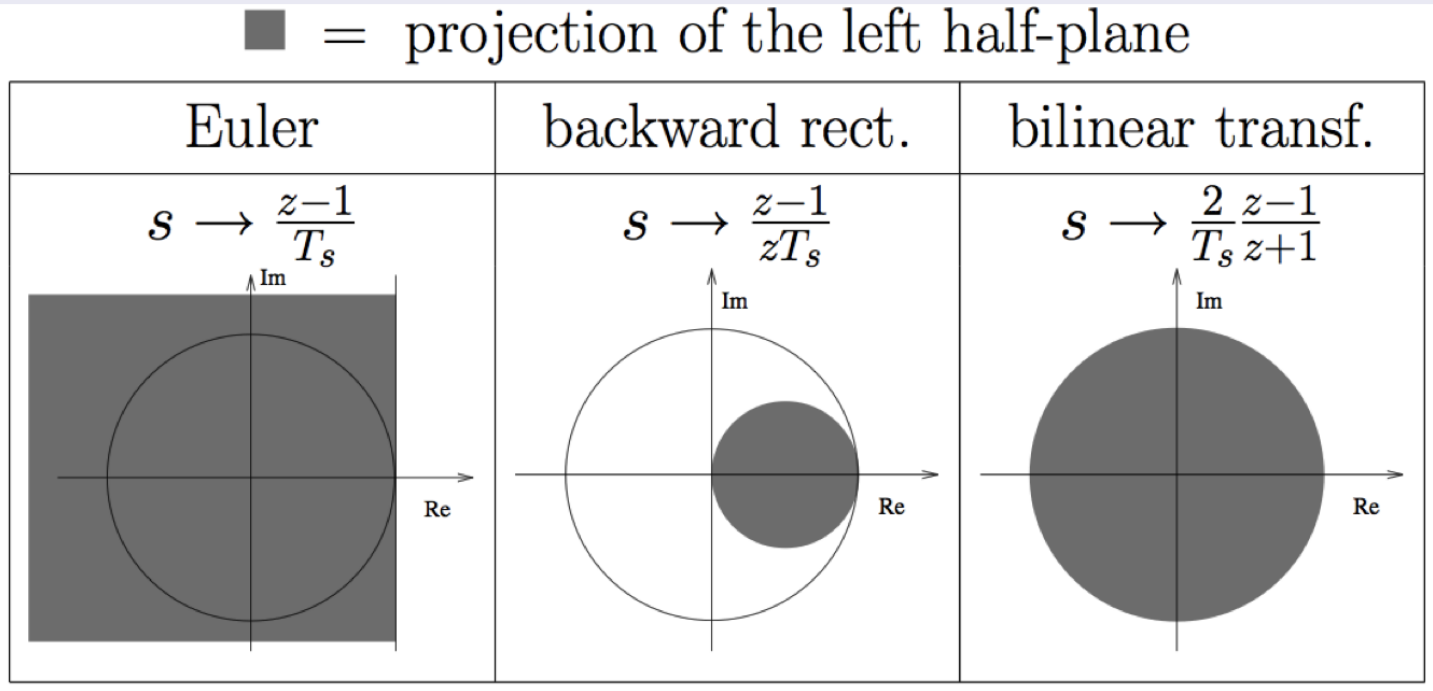
- $z = 1 + T_s s$
- $z = \frac{1}{1 - T_s s}$
- $z = \frac{1 + T_s \frac{s}{2}}{1 - T_s \frac{s}{2}}$

By substituting s by $j\omega$ the boundaries of the regions in the z-plane, which originate from the stable portion of the s-plane, are obtained.

Stability of the numerical integration methods

Graphical representation

Stable s-plane poles map onto the shaded regions in the z-plane. The unit circle is shown for reference.



Stability of the numerical integration methods

Mapping of the left-hand-s-plane

- Forward Euler: a stable continuous-time system may become unstable after discretization;
- Backward Euler: a stable continuous-time system will stay stable after discretization, but the number of degrees of freedom is restricted;
- Bilinear transformation: the entire left-hand-plane is mapped into the unit circle.

Bilinear rule with prewarping

Distortion

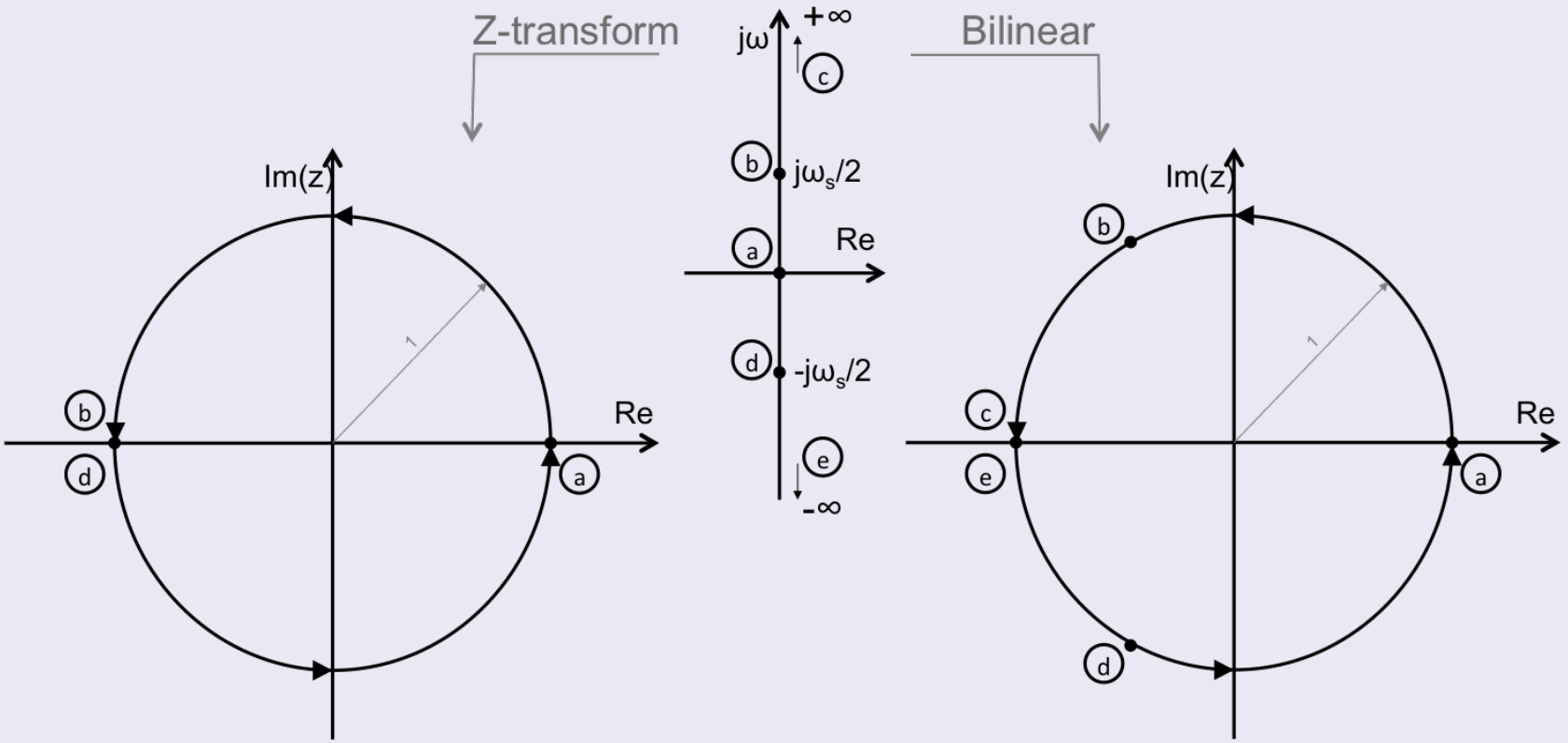
Looking at the graphical representation of the stability on the previous slide, you can see that the bilinear rule maps the stable region of the s-plane into the stable region of the z-plane. The entire $j\omega$-axis is compressed into the 2π-length of the unit circle, causing a frequency distortion.

Origin of prewarping

On the next slide you can see that when we compare the $\mathcal{Z}$-transform with the bilinear transformation, points b and d in the s-plane are mapped onto different points in the z-plane. $(-1 + 0j)$ can only be reached by the bilinear rule when ω goes to infinity.

Bilinear rule with prewarping

Origin of prewarping



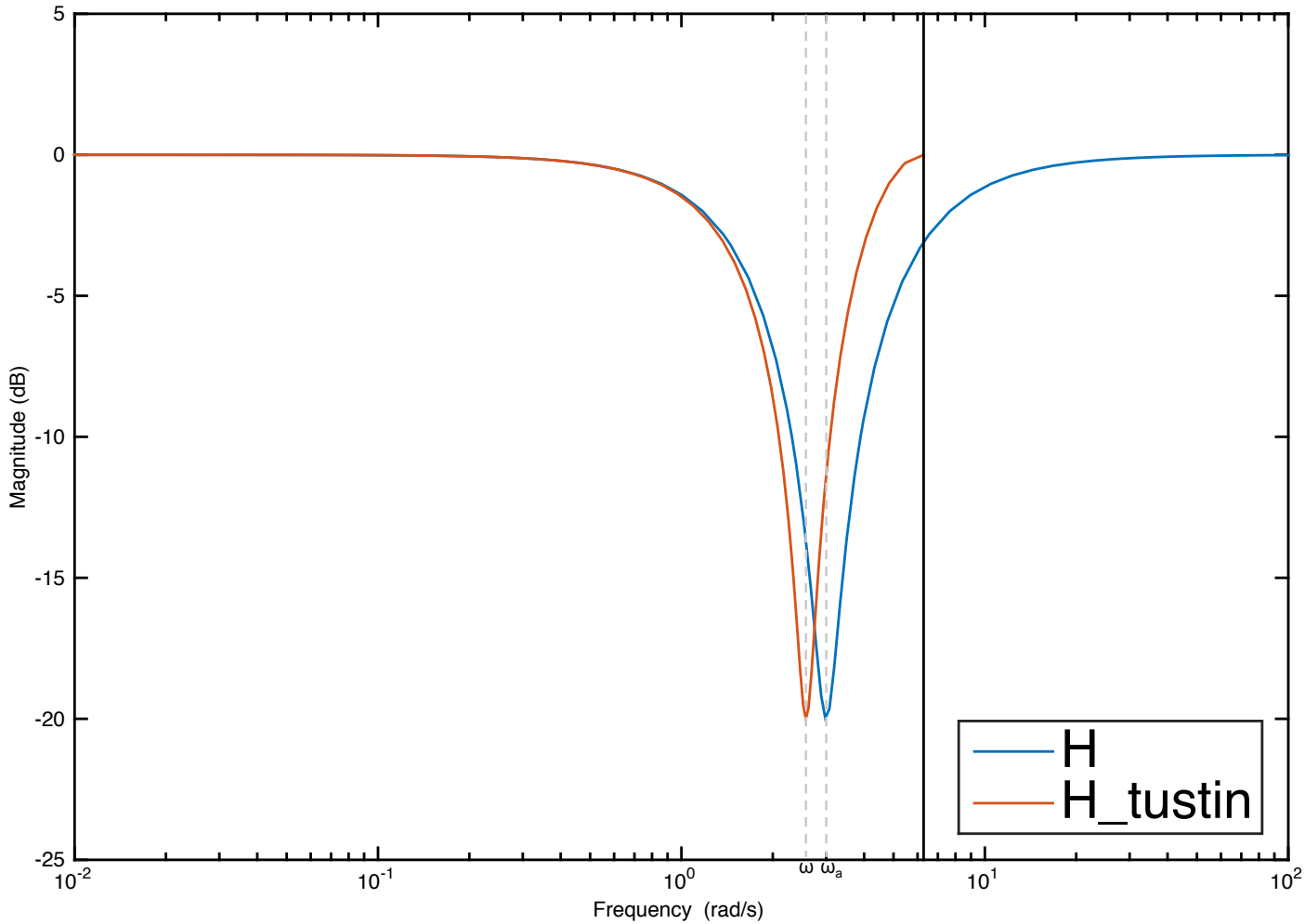
Bilinear rule with prewarping

Distortion: frequency domain

The bilinear rule causes every feature that is visible in the frequency response of the continuous-time filter to also be visible in the discrete-time filter, but at a different frequency. However, at low frequencies, the same behaviour of the function is guaranteed. This is illustrated by the figure on the next slide.

When the actual frequency of ω is input to the discrete-time filter designed by use of the bilinear transform, it is desired to know at what frequency, ω_a, for the continuous-time filter that this ω is mapped to.

Bilinear rule with prewarping



Bilinear rule with prewarping

ω_a with input ω

We now look for a relation between ω_a and ω, using the 's-domain'-'z-domain' relation $z = e^{sT}$ and the substitution $s = j\omega$:

$$\begin{aligned}
\left(\frac{2}{T_s}\frac{z-1}{z+1}\right)_{z=e^{j\omega T_s}} &= \frac{2}{T_s}\frac{e^{j\omega T_s}-1}{e^{j\omega T_s}+1} \\
&= \frac{2}{T_s}\frac{e^{j\omega T_s/2}\left(e^{j\omega T_s/2}-e^{-j\omega T_s/2}\right)}{e^{j\omega T_s/2}\left(e^{j\omega T_s/2}+e^{-j\omega T_s/2}\right)} \\
&= j\frac{2}{T_s}\frac{sin(\omega T_s/2)}{cos(\omega T_s/2)} \\
&= j\frac{2}{T_s}tan\left(\frac{\omega T_s}{2}\right) = j\omega_a
\end{aligned}$$

Bilinear rule with prewarping

Frequency warping

The discrete-time system has the same behavior at frequency ω as the continuous-time system at frequency $\omega_a = \frac{2}{T_s} tan(\frac{\omega T_s}{2})$.

Specifically, the gain and phase shift that the discrete-time filter has at frequency ω is the same gain and phase shift that the continuous-time filter has at frequency $\omega_a = \frac{2}{T_s} tan(\frac{\omega T_s}{2})$.

This effect of the non-linear relation between ω and ω_a is called **frequency warping**.

Bilinear rule with prewarping

Frequency prewarping

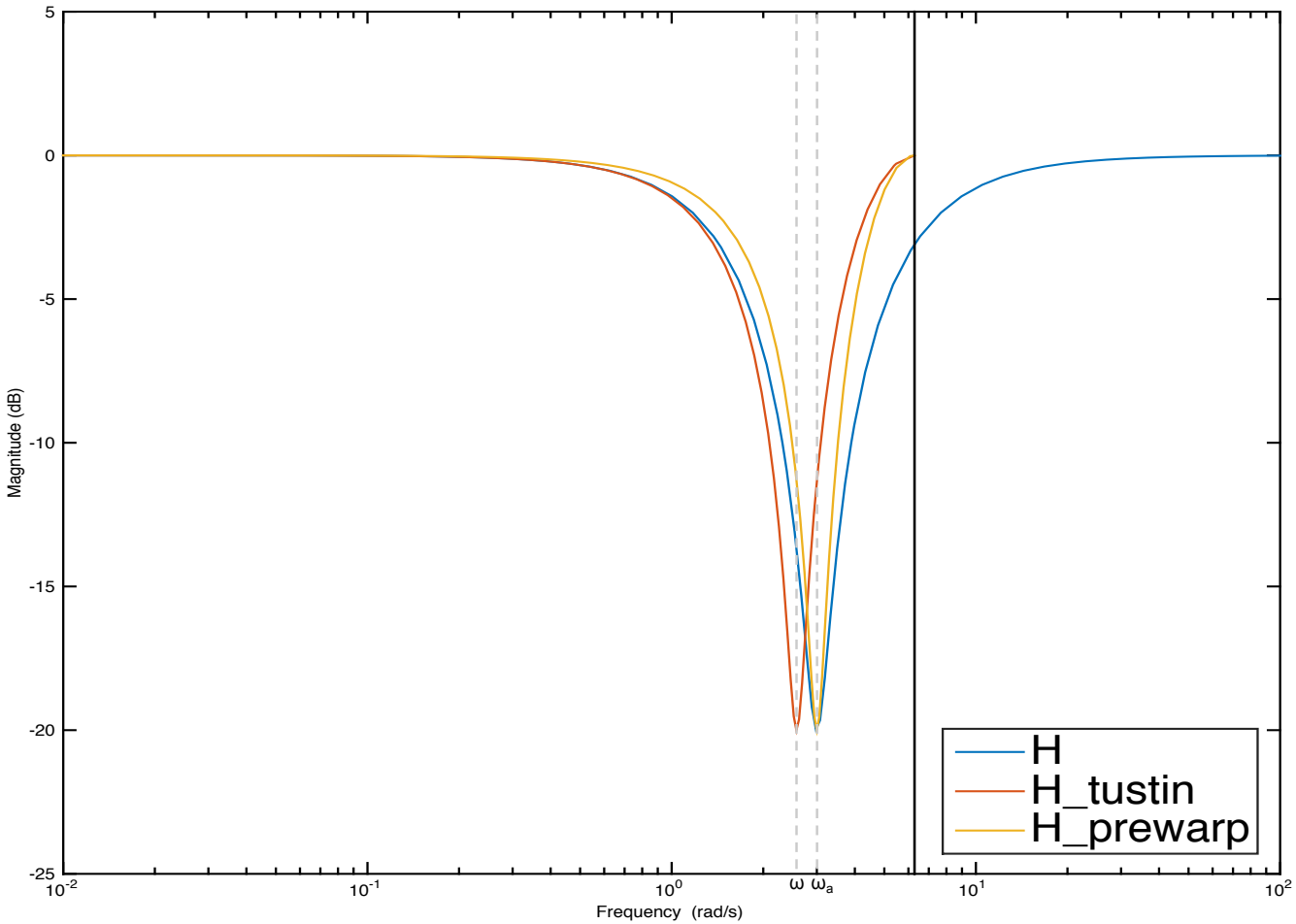
In certain situations, however, we really want the characteristics to be conserved during the discretization.

By setting $\omega_a = \frac{2}{T} tan(\frac{\omega T}{2})$ for every frequency specification that the designer has control over, the frequency warping will be compensated. This is called **frequency prewarping**.

The digital filter can be made to match the frequency response of the continuous filter at frequency ω_0 if the following transformation is substituted into the continuous filter transfer function:

$$s \leftarrow \frac{\omega_0}{tan\left(\frac{\omega_0 T_s}{2}\right)} \frac{z-1}{z+1} \quad (8.4)$$

Bilinear rule with prewarping



Outline

Impulse-invariant method

Practical rule

This method converts a continuous-time system into a discrete one by matching the impulse response using these transformations:

$H(s)$	$H(z),\ a = e^{bT_s}$
$\frac{c}{s-b}$	$\frac{T_s c}{1-az^{-1}}$
$\frac{c}{(s-b)^2}$	$T_s^2 \frac{caz^{-1}}{(1-az^{-1})^2}$
$\frac{c}{(s-b)^3}$	$\frac{T_s^3 caz^{-1}(1+az^{-1})}{2(1-az^{-1})^3}$
$\frac{c}{(s-b)^4}$	$\frac{T_s^4 caz^{-1}(1+4az^{-1}+a^2z^{-2})}{6(1-az^{-1})^4}$

Outline

Zero-pole equivalent

General approach

The map $z = e^{sT}$ is applied to the poles as well as to the zeros of the continuous system. The following rules must be followed:

1. If $s = -a$ is a pole of H(s), then $z = e^{-aT}$;
2. All finite zeros $s = -b$ are mapped by $z = e^{-bT}$;
3. Zeros at ∞ are mapped to $z = -1$;
4. The DC gain is set such that $\lim_{s\to 0} G(s) = \lim_{z\to 1} G_d(z)$.

Zero-pole equivalent

Example

Given:

$$H(s) = \frac{s+1}{0.1s+1}$$

Pole at $s = -10$ and zero at $s = -1$.
New discrete transfer function with equivalent poles and zeros:

$$H(z) = K\frac{z-e^{-T}}{z-e^{-10T}}$$

K is chosen so that $\mid H(z) \mid_{z=1} = \mid H(s) \mid_{s=0} \rightarrow K = 4.150$
Using $T = 0.25$, this results in:

$$H(z) = 4.150\frac{z-0.7788}{z-0.0821}$$

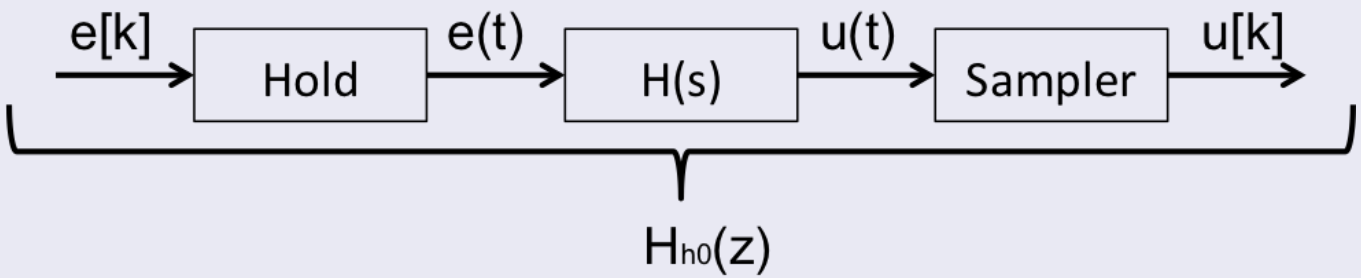
Outline

Hold equivalent

General approach

This method uses a discrete system consisting of 3 subsystems, each with its own purpose.

1. Hold: approximating $e(t)$ from the samples $e[k]$
2. H(s): putting the $e(t)$ through the given transfer function H(s) of the continuous system, resulting in u(t)
3. Sampler: sampling u(t)



There are many techniques for holding a sequence of samples.

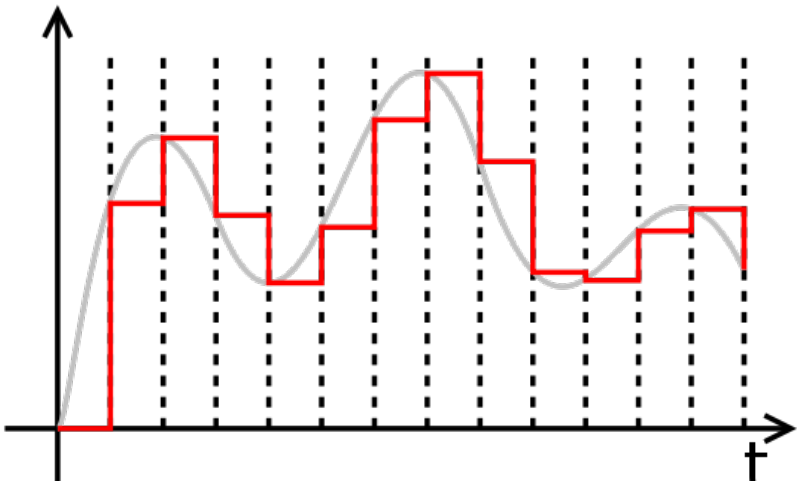
Zero-order hold equivalent (ZOH)

Practical rule

The zero-order hold equivalent transfer function $H_{zoh}(z)$ can be found by computing the following:

$$H_{zoh}(z) = (1 - z^{-1})\mathcal{Z}\{\frac{H(s)}{s}\}$$

This is obtained by holding e(t) constant at e(k) over the interval kT_s to $(k+1)T_s$ and $\mathcal{Z}$-transforming it.



Zero-order hold equivalent

Example

$$H(s) = \frac{0.1}{s+0.1}$$

Step 1: multiply by 1/s and perform partial fraction expansion

$$\frac{H(s)}{s} = \frac{0.1}{s*(s+0.1)} = \frac{1}{s} - \frac{1}{s+0.1}$$

Step 2: perform z-transformation

$$\mathcal{Z}\{\frac{H(s)}{s}\} = \frac{1}{1-z^{-1}} - \frac{1}{1-e^{-0.1T}*z^{-1}}$$

Step3: simplify and multiply by $(1 - z^1)$

$$H_{zoh}(z) = \frac{1-e^{-0.1T}}{z-e^{0.1T}}$$

Zero-order hold equivalent

Step Invariant Transformation

This rule is also called the step-invariant method because it matches the step response of the continuous and the discrete system. Next transformations may be useful:

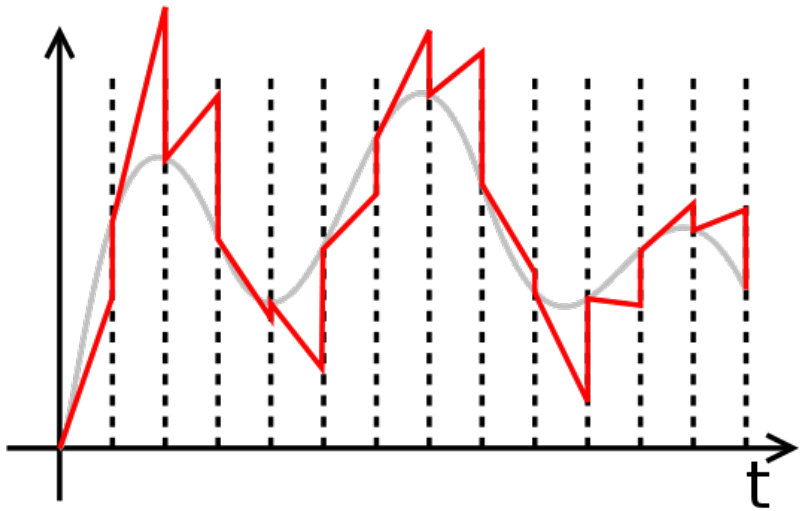
$f(t)$	$F(s)$	$f(k)$	$F(z)$
$u(t)$, unit step	$\frac{1}{s}$	$u(k)$, unit step	$\frac{z}{z-1}$
$tu(t)$	$\frac{1}{s^2}$	$kTu(k)$	$\frac{Tz}{(z-1)^2}$
$e^{-at}u(t)$	$\frac{1}{s+a}$	$(e^{-aT})^k u(k)$	$\frac{z}{z-e^{-aT}}$
$te^{-at}u(t)$	$\frac{1}{(s+a)^2}$	$kT(e^{-aT})^k u(k)$	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
$\sin(\omega t)u(t)$	$\frac{\omega}{s^2+\omega^2}$	$\sin(k\omega T)u(k)$	$\frac{z\sin\omega T}{z^2-2z\cos\omega T+1}$
$\cos(\omega t)u(t)$	$\frac{s}{s^2+\omega^2}$	$\cos(k\omega T)u(k)$	$\frac{z(z-\cos\omega T)}{z^2-2z\cos\omega T+1}$

First-order hold equivalent

Practical rule

The non-causal first-order hold equivalent transfer function $H_{foh}(z)$ can be found by computing the following:

$$H_{foh}(z) = \frac{(z-1)^2}{Tz}\mathcal{Z}\{\frac{H(s)}{s^2}\}$$



First-order hold equivalent

Example

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

Step 1: multiply by $1/s^2$ and perform partial fraction expansion

$$H(s) = \frac{1}{s^4}$$

Step 2: perform z-transformation

$$\mathcal{Z}\{\frac{H(s)}{s^2}\} = \frac{T^3}{6}\frac{(z^2+4z+1)z}{(z-1)^4}$$

Step 3: simplify and multiply by $\frac{(z-1)^2}{T*z}$

$$H_{foh}(z) = \frac{T^2}{6}\frac{z^2+4z+1}{(z-1)^2}$$

Outline

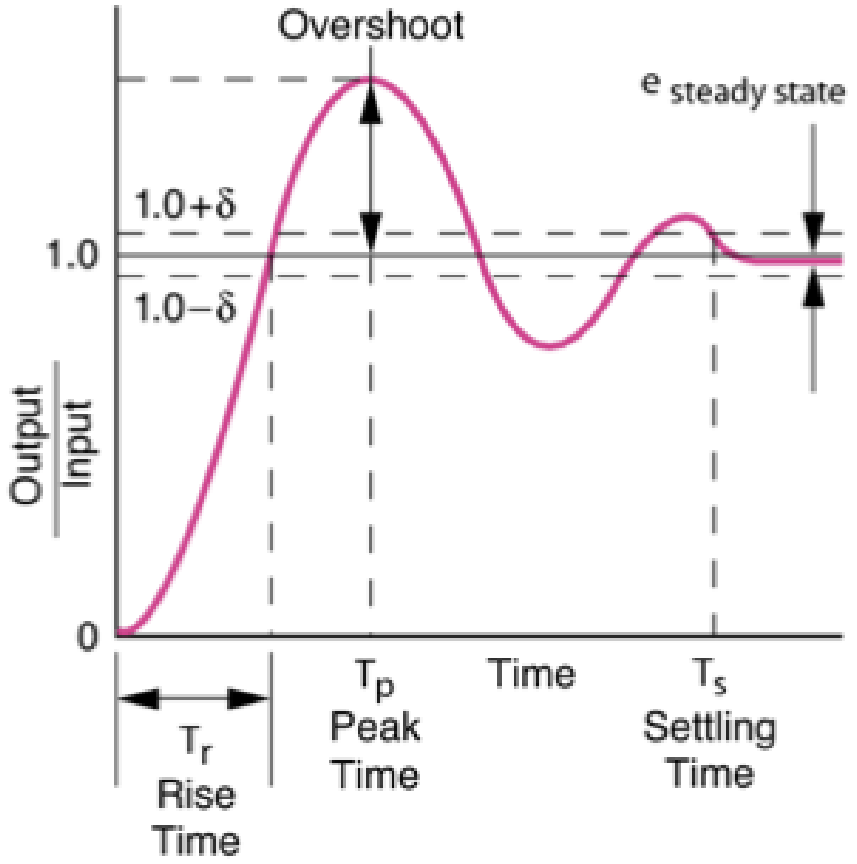
Sampling time T_s based on the time response

Rise time

Time needed to reach the steady state for the first time.

Practical rule

A good rule of thumb is $T_s = \frac{T_{rise}}{10}$.



Sampling time T_s based on the frequency response

Practical rule

A good rule of thumb is $T_s = \frac{1}{2{,}2 Bandwidth}$.

Signal vs System

It is very important to understand that although the previous rule of thumb for a system seems the same as the rule of thumb for a signal, they differ! The rule of thumb for a signal uses the bandwidth of the signal and the rule of thumb for a system uses the bandwidth of the system, which can be found in the bodeplot or using MATLAB.

Outline

Outline

MATLAB

MATLAB Commands for calculations

MATLABs Control System Toolbox offers extensive support for discretization and re-sampling of linear systems:

- "c2d(system,sampling time,method)" used for discretization
- "d2c(system,sampling time,method)" used for transforming a discrete-time system to a continuous-time system (not used in the examples).

These commands can also have extra options. It is necessary to specify the options in an additional command: "c2dOptions('OptionName',OptionValue)".

MATLAB

Available options

- 'Method'
- 'PrewarpFrequency'
- 'FractDelayApproxOrder'

Available methods

- zero-order hold equivalent: "zoh"
- first-order hold equivalent: "foh"
- impulse invariant rule: "impuls"
- zero-pole matching equivalent: "matched"
- bilinear: "tustin"

MATLAB

MATLAB Commands for visualisation

- MATLAB can draw the bode plots of the continuous system $\rightarrow$ command: "bode(system)";
- MATLAB can carry out a graphic showing the impulse response of the continuous system and the discretized system $\rightarrow$ command: "impulse(H,'b',Hd,'r')" in which 'b' and 'r' stand for blue and red. By using colours, the step responses of both systems can easily be distinguished;
- MATLAB can carry out a graphic showing the step response of the continuous system and the discretized system $\rightarrow$ command: "step(H,'b',Hd,'r')".

Outline

Exercise 1 with MATLAB

Exercise 1

We will now discuss 4 methods applied on the same continuous system: $H(s) = \frac{s+1}{s^2+s+1}$

- The sampling time can be determined by the rule of thumb previously explained: $T_s = 2.2\text{Bandwidth}$;
- MATLAB can calculate the bandwidth of a given continuous system, using the command: "bandwidth(system)".

This results in a sampling time of 0.25033 seconds for the given system

Zero-order hold equivalent

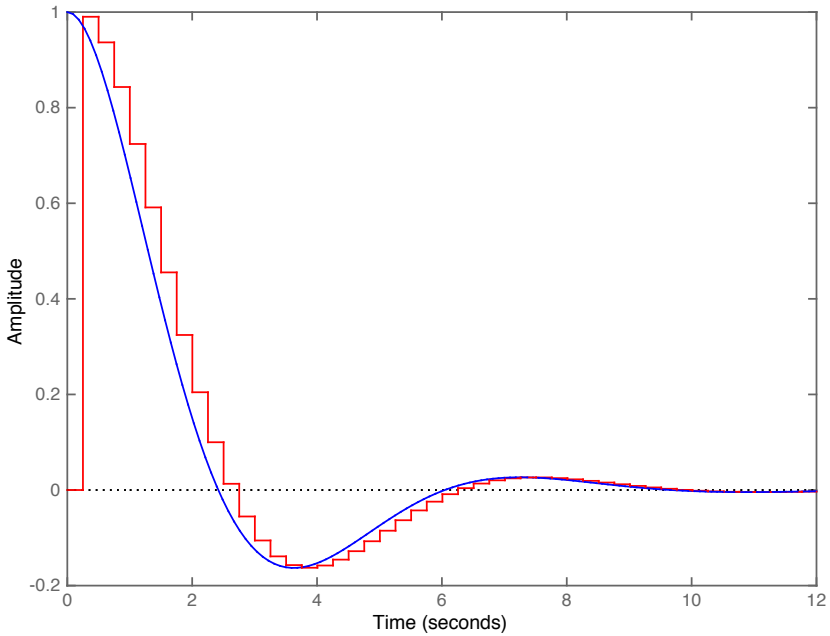
Example

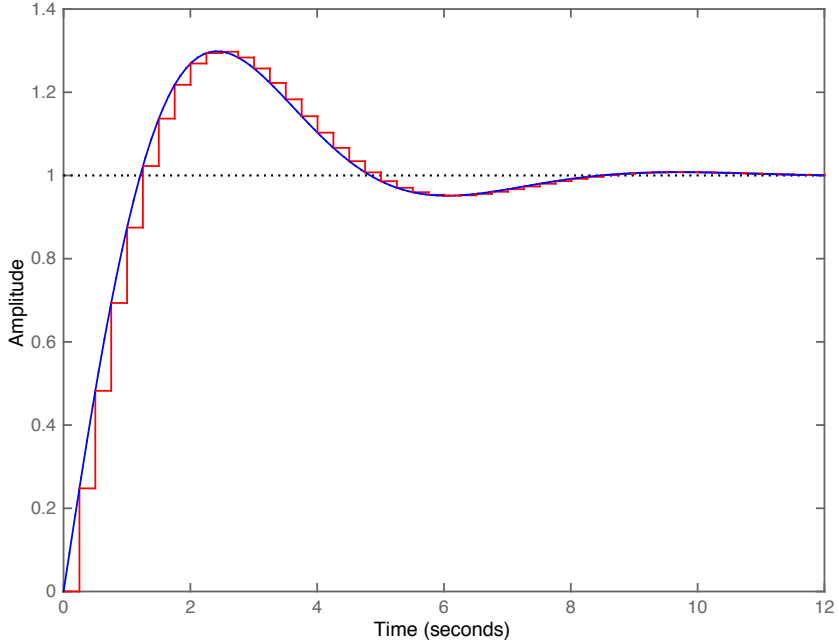
Given:
$H(s) = \frac{s+1}{s^2+s+1}$
Sampling time $= 0.25033$sec

MATLAB commands:
H = tf([1 1],[1 1 1]) we define the system H by its transfer function
Hd = c2d(H,0.25033,'zoh') we calculate the discrete-time model

Result: $H_{zoh}(z) = \frac{0.2479z-0.1927}{z^2-1.723z+0.7785}$

Zero-order hold equivalent (impulse and step response)





First-order hold equivalent

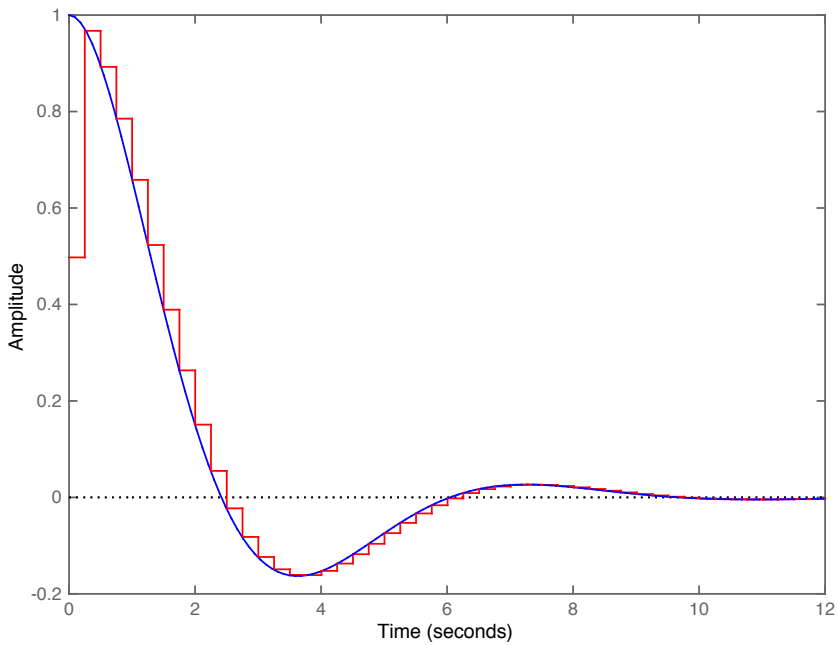
Example

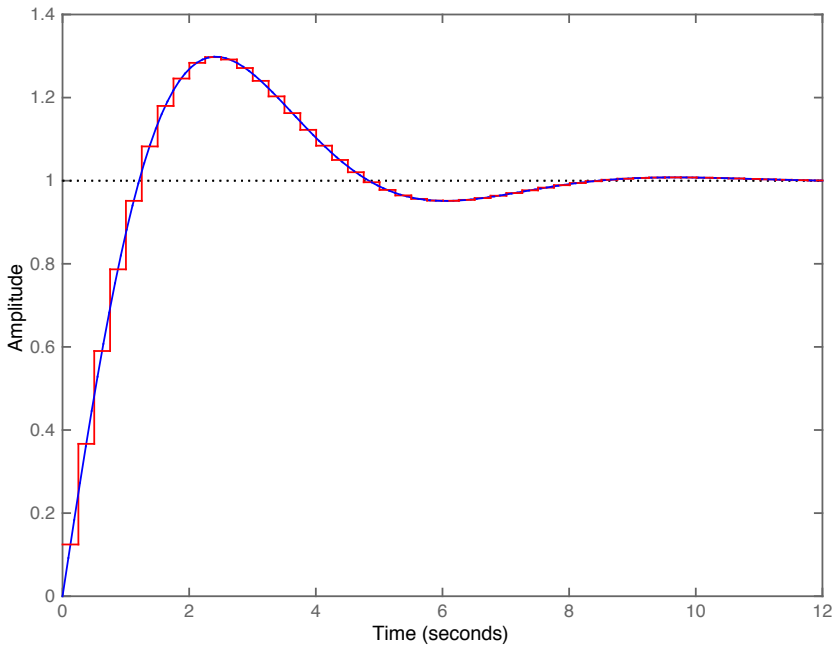
Given:
$H(s) = \frac{s+1}{s^2+s+1}$
Sampling time $= 0.25033$sec

MATLAB commands:
H = tf([1 1],[1 1 1]) we define the system H by its transfer function
Hd = c2d(H,0.25033,'foh') we calculate the discrete-time model

Result: $H_{foh}(z) = \frac{0.1245z^2+0.02752z-0.09691}{z^2-1.723z+0.7785}$

First-order hold equivalent (impulse and step response)





Impulse invariant rule

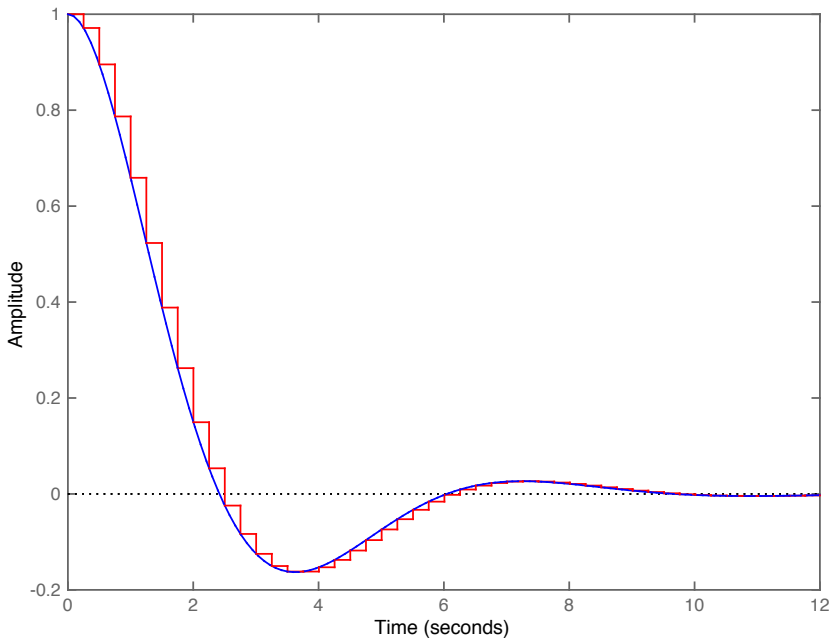
Example

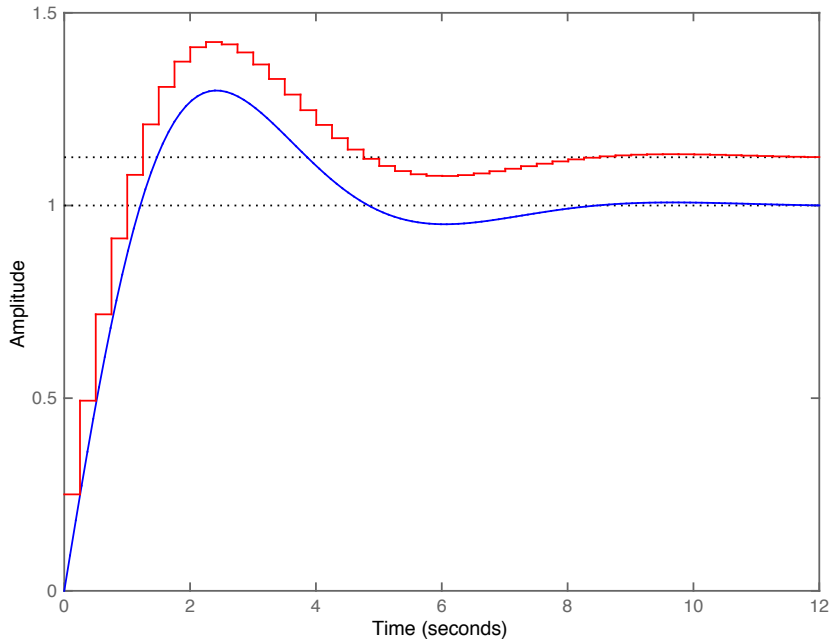
Given:
$H(s) = \frac{s+1}{s^2+s+1}$
Sampling time $= 0.25033$sec

MATLAB commands:
H = tf([1 1],[1 1 1]) we define the system H by its transfer function
Hd = c2d(H,0.25033,'impuls') we calculate the discrete-time model

Result: $H_{impulse}(z) = \frac{0.2503z^2 - 0.1883z}{z^2 - 1.723z + 0.7785}$

Impuls invariant rule (impulse and step response)





Zero-pole equivalent

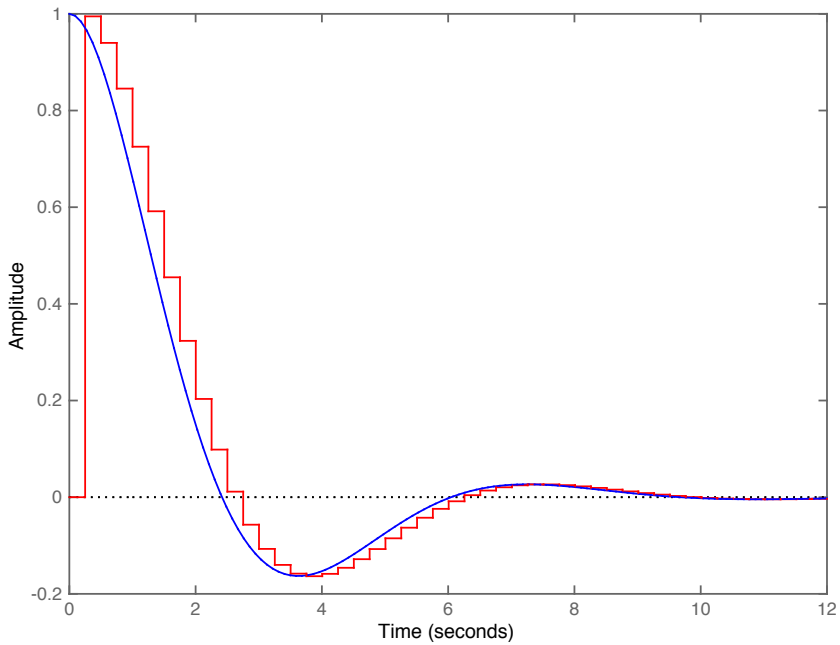
Example

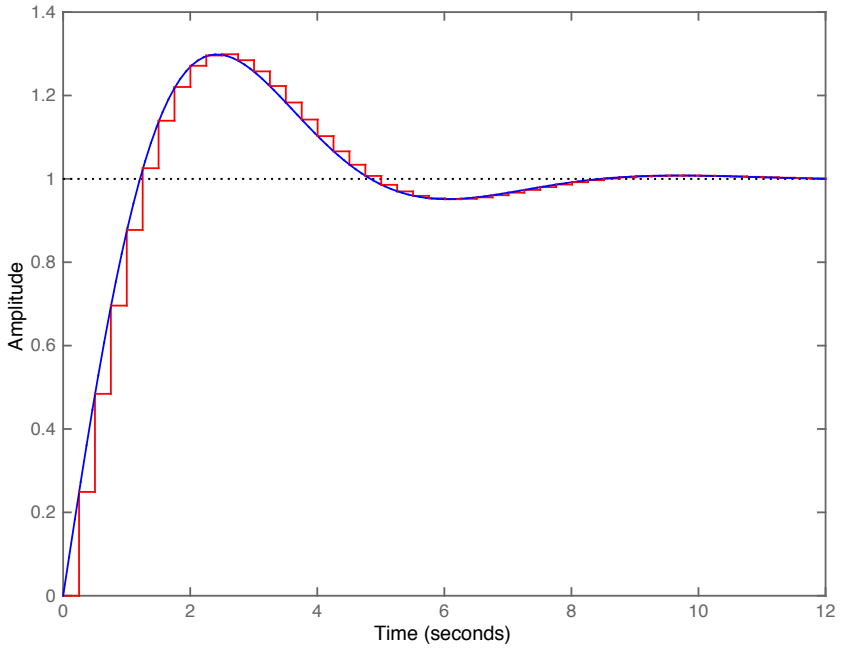
Given:
$H(s) = \frac{s+1}{s^2+s+1}$
Sampling time $= 0.25033$sec

MATLAB commands:
H = tf([1 1],[1 1 1]) we define the system H by its transfer function
Hd = c2d(H,0.25033,'matched') we calculate the discrete-time model

Result: $H_{matched}(z) = \frac{0.249z-0.1939}{z^2-1.723z+0.7785}$

Zero-pole equivalent (impulse and step response)





Exercise 2 with MATLAB

Exercise 2

We will now apply the bilinear rule with and without prewarping on the same continuous system: $H(s) = \frac{s^2+0.5s+9}{s^2+5s+1}$

Criteria:

1. The sampling time is chosen to be 0.5 seconds;
2. The discrete system must have the same behaviour as the continuous one at 3rad/s.

The last criteria applies to this specific exercise. It is possible that the parity of the magnitude, of the discrete and the continuous system, at a different frequency (e.g. the peak frequency) is required.

Tustin rule

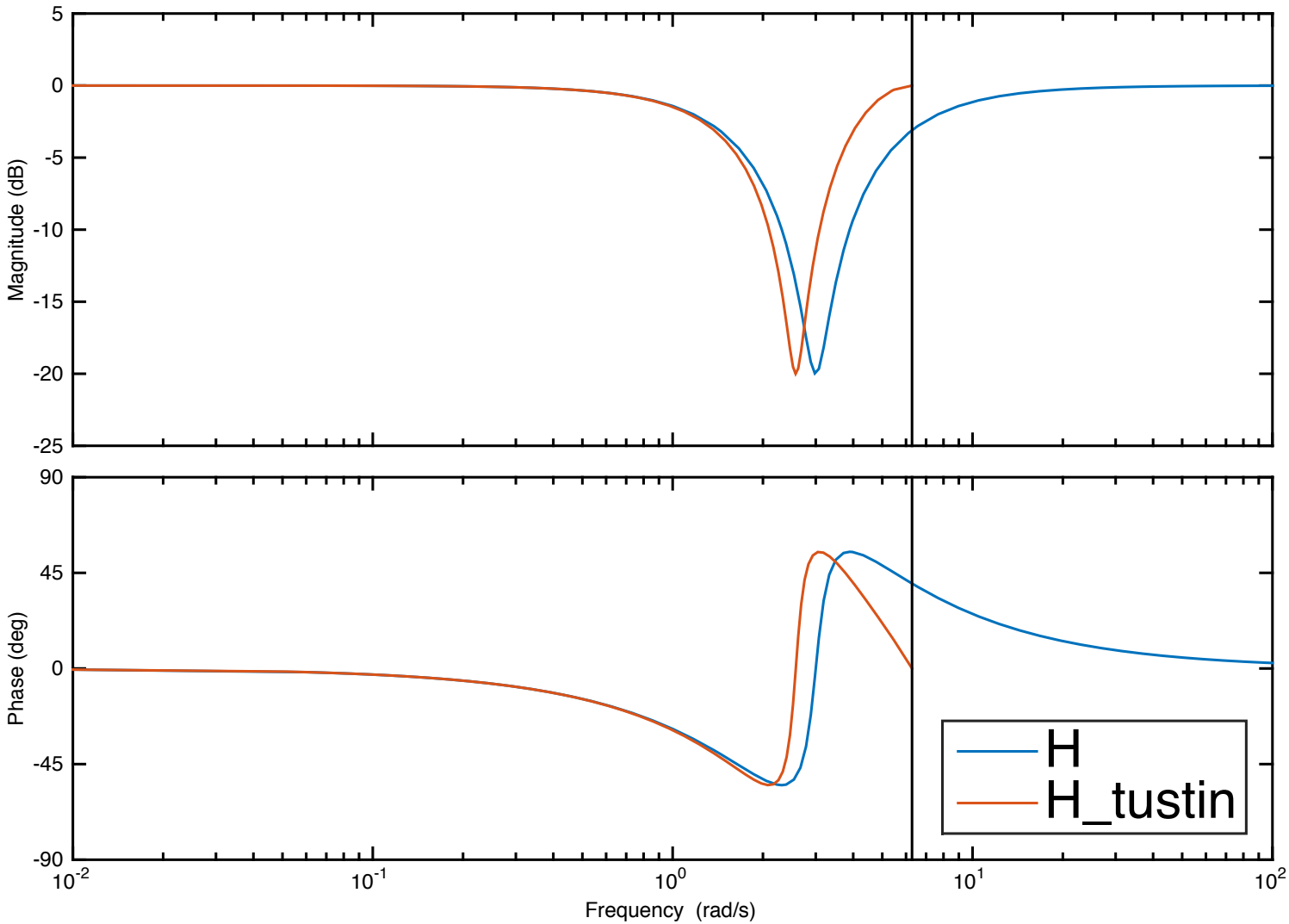
Example

Given:
$H(s) = \frac{s^2+0.5s+9}{s^2+5s+1}$
Sampling time = 0.5sec

MATLAB commands:
H = tf([1,0.5,9],[1,5,9]) definition of the system H
Hdt = c2d(H,0.5,'tustin') we calculate the discrete-time model

Result: $H_{Tustin}(z) = \frac{0.6z^2-0.3111z+0.5111}{z^2-0.3111z+0.1111}$

Tustin rule (bode plot)



Tustin rule with prewarping

Example

Given:
$H(s) = \frac{s^2+0.5s+9}{s^2+5s+1}$
Sampling time = 0.5sec

MATLAB commands:
H = tf([1,0.5,9],[1,5,9]) definition of the system H
damp(H) results in 3.0Hz
discopts = c2dOptions('Method','tustin','PrewarpFrequency',3.0)
at the prewarpfrequency, the discrete-time model will have the same behavior as the continuous-time one
Hdtp = c2d(H,0.5,discopts) we calculate the discrete-time model

Result: $H_{TustinPrewarp}(z) = \frac{0.5915z^2-0.07726z+0.5007}{z^2-0.07726z+0.09215}$

Tustin rule with prewarping (bode plot)