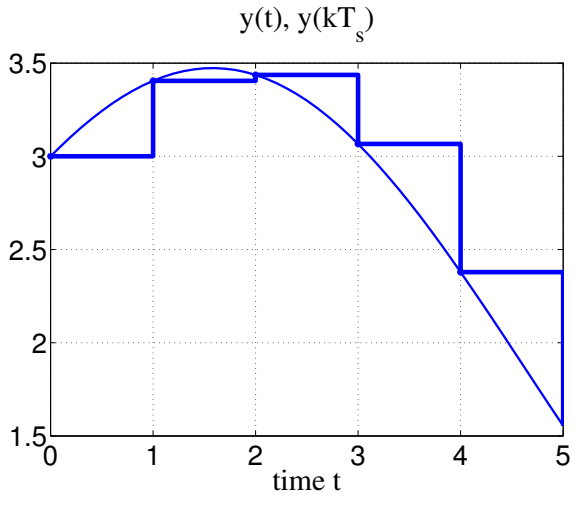
Modele matematice in timp discret. Sisteme de reglare automata in timp discret. Tranzformarea Z. Functii de transfer si modele de stare discrete. Analiza stabilitatii sistemelor in timp discret. Modele matematice in timp discret ale proceselor si regulatoarelor continue (discretizare). Regulatoare numerice

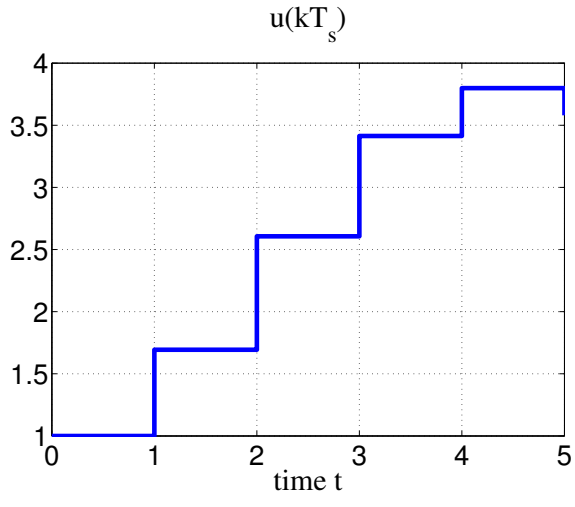
Special thanks to:
A. Bemporad, Automatic Control 1, Lecture Notes, University of Trento, Italy, 2011, http://cse.lab.imtlucca.it/~bemporad/automatic_control_course.html
P. Raica, Systems Theory, Lecture Notes, Technical University of Cluj-Napoca, Cluj-Napoca, 2012, https://cs.utcluj.ro/files/educatie/licenta/2021-2022/22_CALCen_ST.pdf

Discrete-time models

Introduction



Sampling of a continuous signal



Discrete-time signal

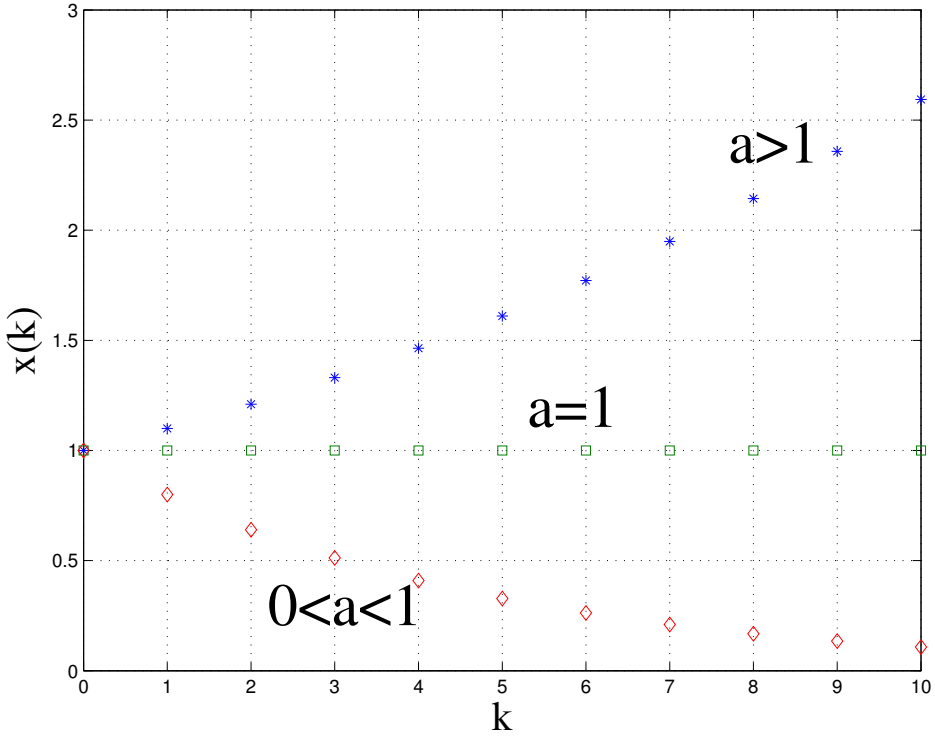
- Discrete-time models describe relationships between *sampled* variables $x(kT_s)$, $u(kT_s)$, $y(kT_s)$, $k = 0, 1, \ldots$
- The value $x(kT_s)$ is kept constant during the *sampling interval* $[kT_s, (k+1)T_s)$
- A discrete-time signal can either represent the *sampling* of a *continuous-time* signal, or be an intrinsically discrete signal
- Discrete-time signals are at the basis of *digital controllers* (as well as of digital filters in signal processing)

Difference equation

- Consider the first order *difference equation* (autonomous system)

$$\begin{cases} x(k+1) &= ax(k) \\ x(0) &= x_0 \end{cases}$$

- The solution is $x(k) = a^k x_0$



Difference equation

- First-order difference equation with input (non-autonomous system)

$$\begin{cases} x(k+1) &= ax(k)+bu(k) \\ x(0) &= x_0 \end{cases}$$

- The solution has the form

$$x(k) = \underbrace{a^k x_0}_{\text{natural response}} + \underbrace{\sum_{i=0}^{k-1} a^i b u(k-1-i)}_{\text{forced response}}$$

- The *natural response* depends on the initial condition $x(0)$, the *forced response* on the input signal $u(k)$

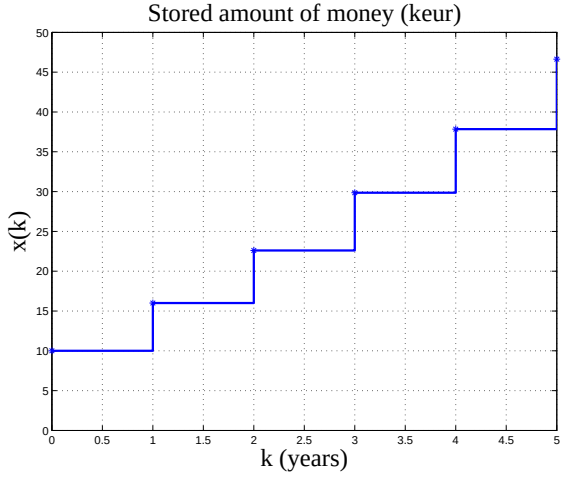
Example - Wealth of a bank account

- k: year counter
- ρ: interest rate
- $x(k)$: wealth at the beginning of year k
- $u(k)$: money saved at the end of year k
- x_0: initial wealth in bank account

Discrete-time model:

$$\begin{cases} x(k+1) &= (1+\rho)x(k)+u(k) \\ x(0) &= x_0 \end{cases}$$

x_0	10 k€
$u(k)$	5 k€
ρ	10 %



$$x(k) = (1.1)^k \cdot 10 + \frac{1-(1.1)^k}{1-1.1}5 = 60(1.1)^k - 50$$

Linear discrete-time system

- Consider the set of n first-order linear difference equations forced by the input $u(k) \in \mathbb{R}$

$$\begin{cases} x_1(k+1) &= a_{11}x_1(k) + \ldots + a_{1n}x_n(k) &+ b_1u(k) \\ x_2(k+1) &= a_{21}x_1(k) + \ldots + a_{2n}x_n(k) &+ b_2u(k) \\ \vdots & \vdots & \vdots \\ x_n(k+1) &= a_{n1}x_1(k) + \ldots + a_{nn}x_n(k) &+ b_nu(k) \\ x_1(0) = x_{10}, & \ldots \quad x_n(0) = x_{n0} \end{cases}$$

- In compact matrix form:

$$\begin{cases} x(k+1) &= Ax(k) + Bu(k) \\ x(0) &= x_0 \end{cases}$$

where $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$.

Linear discrete-time system

- The solution is

$$x(k) = \underbrace{A^k x_0}_{\text{natural response}} + \underbrace{\sum_{i=0}^{k-1} A^i B u(k-1-i)}_{\text{forced response}}$$

- If matrix A is diagonalizable, $A = T\Lambda T^{-1}$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \Rightarrow A^k = T \begin{bmatrix} \lambda_1^k & 0 & \dots & 0 \\ 0 & \lambda_2^k & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n^k \end{bmatrix} T^{-1}$$

where $T = [v_1 \dots v_n]$ collects n independent eigenvectors.

Modal response

- Assume input $u(k) = 0,\ \forall k \geq 0$
- Assume A is diagonalizable, $A = T\Lambda T^{-1}$
- The state trajectory (natural response) is

$$x(k) = A^k x_0 = T\Lambda^k T^{-1} x_0 = \sum_{i=1}^{n} \alpha_i \lambda_i^k v_i$$

where

- λ_i = eigenvalues of A
- v_i = eigenvectors of A
- α_i = coefficients that depend on the initial condition $x(0)$

$$\alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = T^{-1}x(0),\ T = [v_1 \dots v_n]$$

- The system modes depend on the eigenvalues of A, as in continuous-time

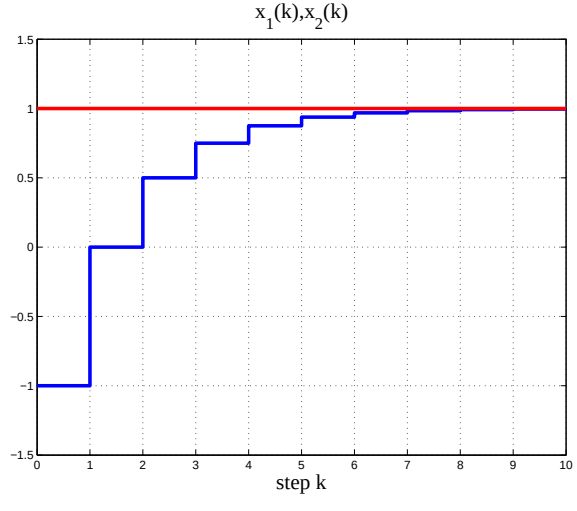
Example

- Consider the linear discrete-time system

$$\begin{cases} x_1(k+1) = \frac{1}{2}x_1(k) + \frac{1}{2}x_2(k) \\ x_2(k+1) = x_2(k) + u(k) \\ x_1(0) = -1 \\ x_2(0) = 1 \end{cases} \quad \Rightarrow \quad \begin{cases} x(k+1) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \\ x(0) = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \end{cases}$$

- Eigenvalues of A: $\lambda_1 = \frac{1}{2}$, $\lambda_2 = 1$
- Solution:

$$\begin{aligned} x(k) &= \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 1 \end{bmatrix}^k \begin{bmatrix} -1 \\ 1 \end{bmatrix} + \sum_{i=0}^{k-1} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 1 \end{bmatrix}^i \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k-1-i) \\ &= \begin{bmatrix} \frac{1}{2^k} & 1-\frac{1}{2^k} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} + \sum_{i=0}^{k-1} \begin{bmatrix} 1-\frac{1}{2^i} \\ 1 \end{bmatrix} u(k-1-i) \\ &= \underbrace{\begin{bmatrix} 1-\left(\frac{1}{2}\right)^{k-1} \\ 1 \end{bmatrix}}_{\text{natural response}} + \underbrace{\sum_{i=0}^{k-1} \begin{bmatrix} 1-\frac{1}{2^i} \\ 1 \end{bmatrix} u(k-1-i)}_{\text{forced response}} \end{aligned}$$



simulation for $u(k) \equiv 0$

n^{th}-order difference equation

- Consider the *n^{th}-order difference equation* forced by u

$$a_n y(k-n) + a_{n-1} y(k-n+1) + \cdots + a_1 y(k-1) + y(k) = b_n u(k-n) + \cdots + b_1 u(k-1) + b_0 u(k)$$

- Equivalent linear discrete-time system in *canonical state matrix form*

$$\begin{cases} x(k+1) = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \dots & -a_1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} (b_n - b_0 a_n) & \dots & (b_1 - b_0 a_1) \end{bmatrix} x(k) + b_0 u(k) \end{cases}$$

- There are infinitely many state-space realizations

MATLAB
`tf2ss`

- n^{th}-order difference equations are very useful for digital filters, digital controllers, and to reconstruct models from data (*system identification*)

Discrete-time linear system

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ y(k) &= Cx(k)+Du(k) \\ x(0) &= x_0 \end{cases}$$

- Given the initial condition $x(0)$ and the input sequence $u(k)$, $k \in \mathbb{N}$, it is possible to predict the entire sequence of states $x(k)$ and outputs $y(k)$, $\forall k \in \mathbb{N}$
- The state $x(0)$ summarizes all the past history of the system
- The dimension n of the state $x(k) \in \mathbb{R}^n$ is called the *order* of the system
- The system is called *proper* (or *strictly causal*) if $D = 0$
- General multivariable case:

$$\begin{aligned} x(k) &\in \mathbb{R}^n \\ u(k) &\in \mathbb{R}^m \\ y(k) &\in \mathbb{R}^p \end{aligned} \qquad \begin{aligned} A &\in \mathbb{R}^{n\times n} \\ B &\in \mathbb{R}^{n\times m} \\ C &\in \mathbb{R}^{p\times n} \\ D &\in \mathbb{R}^{p\times m} \end{aligned}$$

Example - Student dynamics

- Problem Statement:
  - 3-years undergraduate course
  - percentages of students promoted, repeaters, and dropouts are roughly constant
  - direct enrollment in 2nd and 3rd academic year is not allowed
  - students cannot enrol for more than 3 years
- Notation:

k	Year
$x_i(k)$	Number of students enrolled in year i at year k, $i = 1, 2, 3$
$u(k)$	Number of freshmen at year k
$y(k)$	Number of graduates at year k
α_i	promotion rate during year i, $0 \leq \alpha_i \leq 1$
β_i	failure rate during year i, $0 \leq \beta_i \leq 1$
γ_i	dropout rate during year i, $\gamma_i = 1 - \alpha_i - \beta_i \geq 0$

- 3^{rd}-order linear discrete-time system:

$$\begin{cases} x_1(k+1) &= \beta_1 x_1(k) + u(k) \\ x_2(k+1) &= \alpha_1 x_1(k) + \beta_2 x_2(k) \\ x_3(k+1) &= \alpha_2 x_2(k) + \beta_3 x_3(k) \\ y(k) &= \alpha_3 x_3(k) \end{cases}$$

Example - Student dynamics

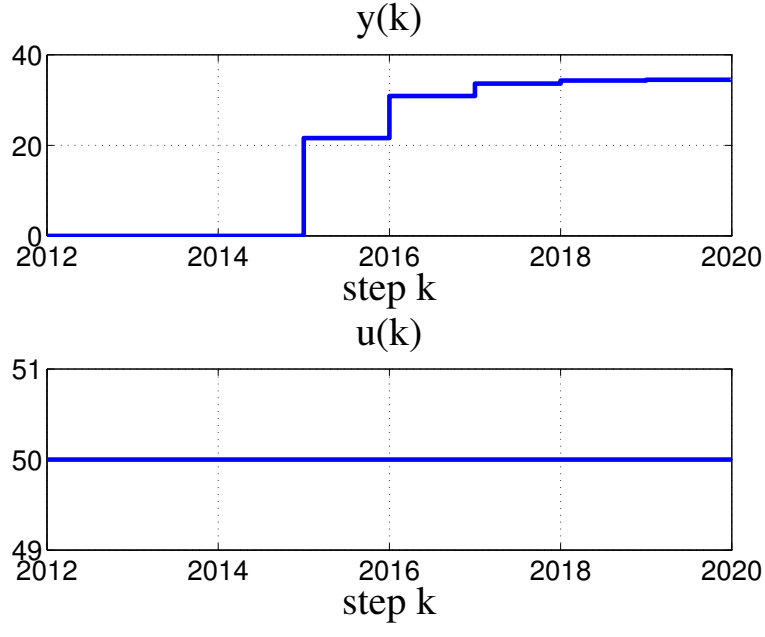
- In matrix form

$$\begin{cases} x(k+1) &= \begin{bmatrix} \beta_1 & 0 & 0 \\ \alpha_1 & \beta_2 & 0 \\ 0 & \alpha_2 & \beta_3 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 0 & \alpha_3 \end{bmatrix} x(k) \end{cases}$$

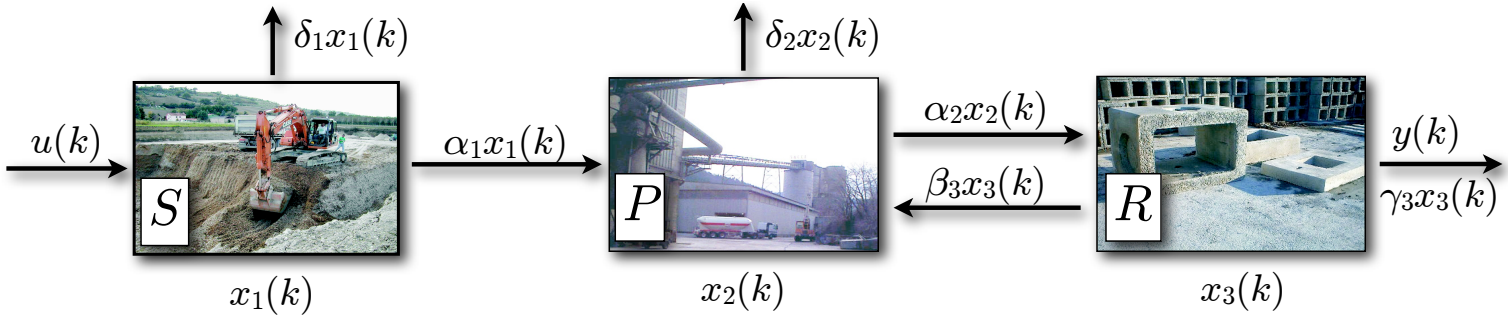
- Simulation

$\alpha_1 = .60$	$\beta_1 = .20$
$\alpha_2 = .80$	$\beta_2 = .15$
$\alpha_3 = .90$	$\beta_3 = .08$

$u(k) \equiv 50,\ k = 2012, \ldots$



Example - Supply chain



- Problem Statement:
  - S purchases the quantity $u(k)$ of raw material at each month k
  - a fraction δ_1 of raw material is discarded, a fraction α_1 is shipped to producer P
  - a fraction α_2 of product is sold by P to retailer R, a fraction δ_2 is discarded
  - retailer R returns a fraction β_3 of defective products every month, and sells a fraction γ_3 to customers
- Mathematical model:

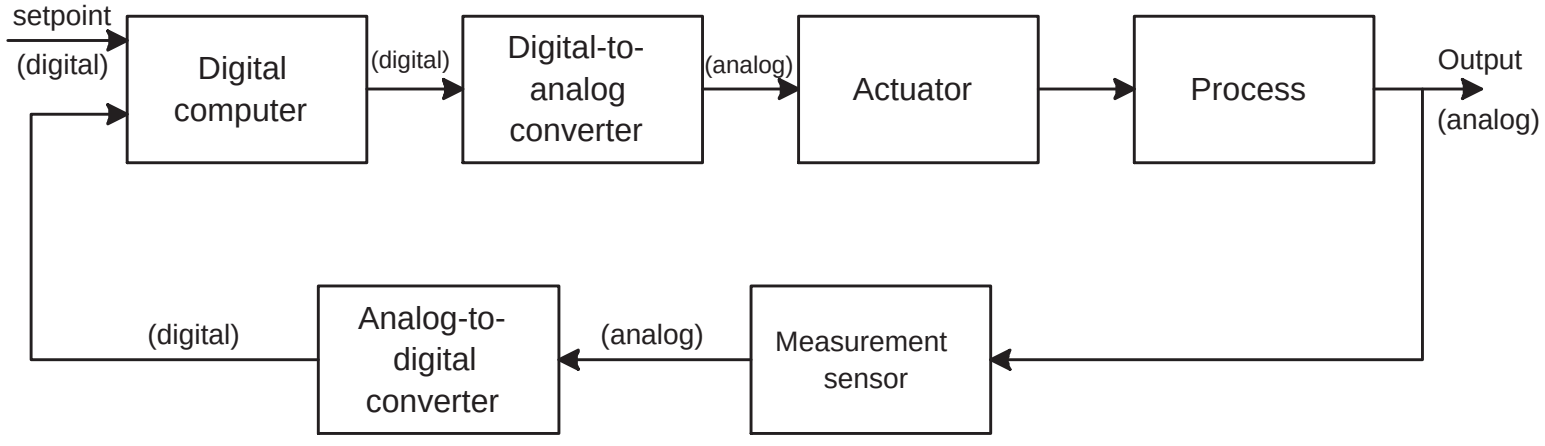
$$\begin{cases} x_1(k+1) &= (1-\alpha_1-\delta_1)x_1(k)+u(k) \\ x_2(k+1) &= \alpha_1 x_1(k)+(1-\alpha_2-\delta_2)x_2(k)+\beta_3 x_3(k) \\ x_3(k+1) &= \alpha_2 x_2(k)+(1-\beta_3-\gamma_3)x_3(k) \\ y(k) &= \gamma_3 x_3(k) \end{cases}$$

k	month counter
$x_1(k)$	raw material in S
$x_2(k)$	products in P
$x_3(k)$	products in R
$y(k)$	products sold to customers

Digital control systems

Introduction

Digital control system



- The measurement data are converted from analog form to digital form by means of the analog-to-digital converter (A/D)
- The output of the digital controller is then converted to analog form by the digital-to-analog converter (D/A).

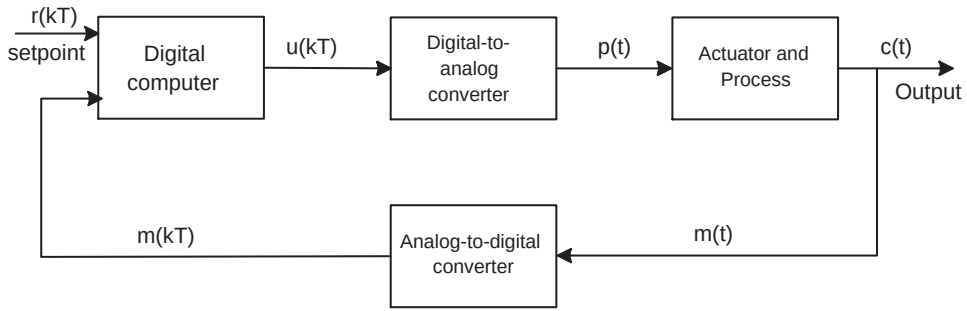
Introduction

The advantages of using digital control:

- Improved measurement sensitivity; the use of digitally coded signals
- Reduced sensitivity to signal noise
- The capability to easily reconfigure the control algorithm in software
- Wide application of digital devices and communications
- Low-energy signals required by digital sensors and devices

- Computers used in control systems are interconnected to the actuator and the process by means of signal converters.
- We shall assume that all the numbers enter or leave the computer at the same fixed period T, called the **sampling period**.

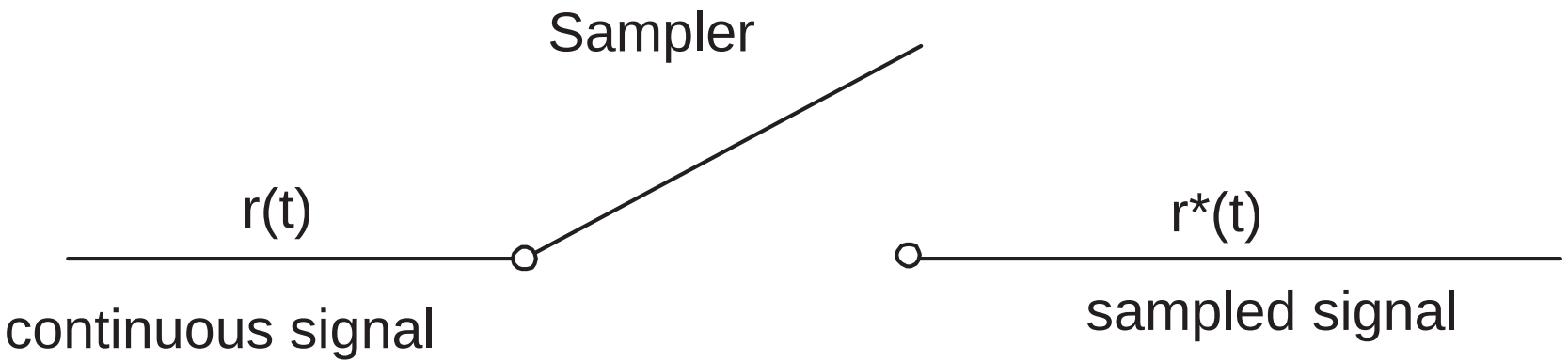
Sampled-data systems



- $r(kT)$, $m(kT)$ and $u(kT)$ are discrete signals.
- $m(t)$, $c(t)$ are continuous signals.

A **sampler** is basically a switch that closes every T seconds for one instant of time.

Sampled-data systems



The input is $r(t)$ and the output $r^*(t)$ where nT is the current sample time and the current value of $r^*(t)$ is $r(nT)$.

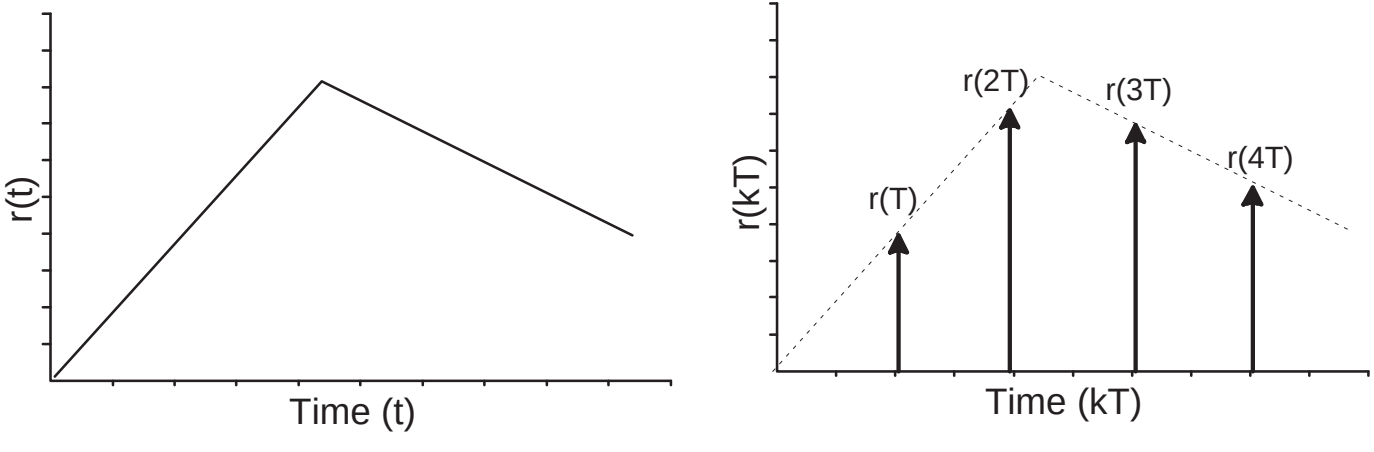
$$r^*(t) = r(nT) \cdot \delta(t - nT)$$

where δ is the impulse function.

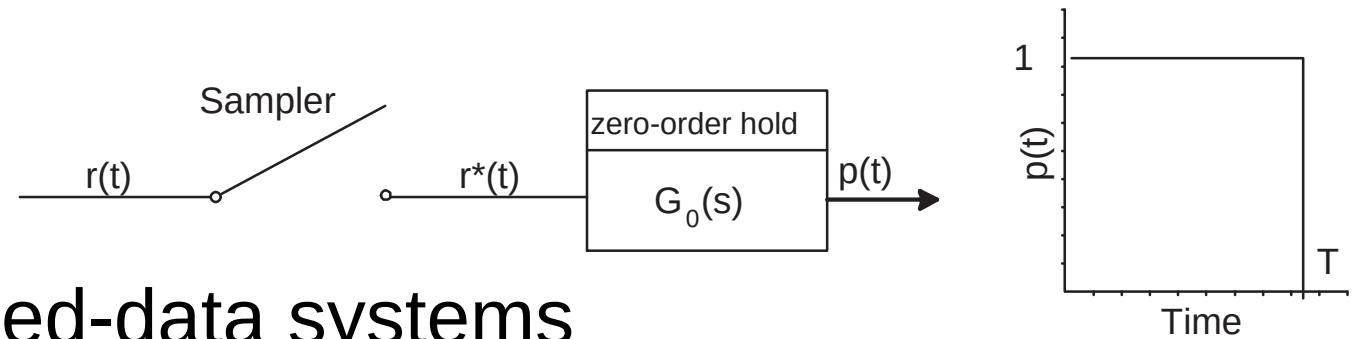
$r^*(t)$ a string of impulses starting at $t = 0$ spaced at T seconds and of amplitude $r^*(kT)$.

Sampled-data systems

$$r^*(t) = \sum_{k=0}^{\infty} r(kT)\delta(t - kT)$$



- A digital-to-analog converter: a device that converts the sampled signal $r^*(t)$ to a continuous signal $p(t)$. It can be represented by a **zero-order hold (ZOH)** circuit.
- (Figure) A sampler and zero-order circuit (left); the response of a ZOH to an impulse input r(kT), which equals 1 when $k = 0$ and equals zero when $k \neq 0$ (right):

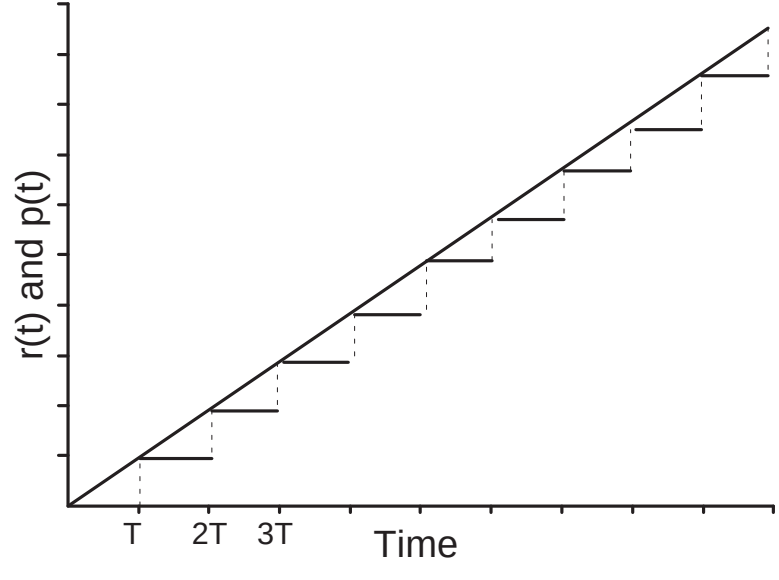


Sampled-data systems

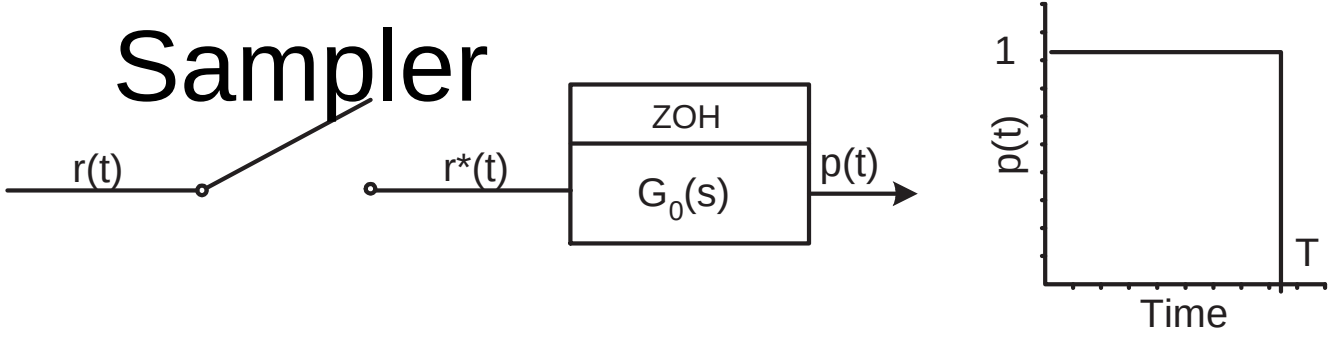
ZOH - takes the value r(kT) and holds it constantly for $kT \leq t \leq (k+1)T$.

Sampled-data systems

- A sampler and ZOH can accurately follow the input signal if T is small compared to the transient changes in the signal.
- (Figure) The response of a sampler and zero-order hold for a ramp input $r(t) = t$.



Sampled-data systems



(input: $u(t) = \delta(t)$, output: $y(t) = 1(t) - 1(t - T)$, $1(t) =$ unit step)

The transfer function of a ZOH:

$$G_0(s) = \frac{Y(s)}{U(s)} = \frac{1}{s} - \frac{1}{s}e^{-sT} = \frac{1 - e^{-sT}}{s}$$

Z transform

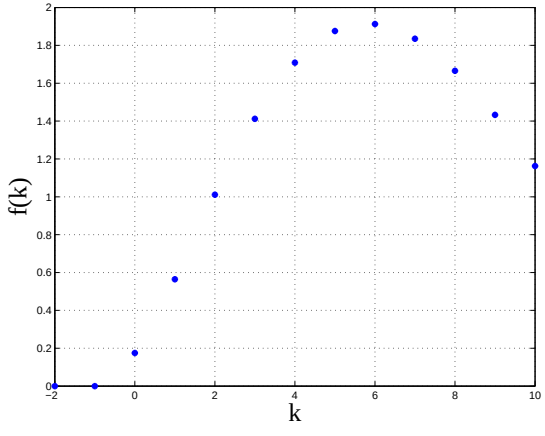
Z-transform

Consider a function $f(k), f : \mathbb{Z} \to \mathbb{R}, f(k) = 0$ for all $k < 0$

Definition

The unilateral *Z-transform* of $f(k)$ is the function of the complex variable $z \in \mathbb{C}$ defined by

$$F(z) = \sum_{k=0}^{\infty} f(k) z^{-k}$$



Witold Hurewicz

(1904-1956)

Once $F(z)$ is computed using the series, it's extended to all $z \in \mathbb{C}$ for which $F(z)$ makes sense

Z-transforms convert difference equations into algebraic equations. It can be considered as a discrete equivalent of the Laplace transform.

Examples of Z-transforms

- *Discrete impulse*

$$f(k) = \delta(k) \triangleq \begin{cases} 0 & \text{if } k \neq 0 \\ 1 & \text{if } k = 0 \end{cases} \quad \Rightarrow \quad \mathcal{Z}[\delta] = F(z) = 1$$

- *Discrete step*

$$f(k) = \mathbb{1}(k) \triangleq \begin{cases} 0 & \text{if } k < 0 \\ 1 & \text{if } k \geq 0 \end{cases} \quad \Rightarrow \quad \mathcal{Z}[\mathbb{1}] = F(z) = \frac{z}{z-1}$$

- *Geometric sequence*

$$f(k) = a^k \, \mathbb{1}(k) \quad \Rightarrow \quad \mathcal{Z}[f] = F(z) = \frac{z}{z-a}$$

Properties of Z-transforms

- *Linearity*

$$\mathcal{Z}[k_1 f_1(k) + k_2 f_2(k)] = k_1 \mathcal{Z}[f_1(k)] + k_2 \mathcal{Z}[f_2(k)]$$

Example: $f(k) = 3\delta(k) - \frac{5}{2^k}\,\mathbb{1}(t) \Rightarrow \mathcal{Z}[f] = 3 - \frac{5z}{z-\frac{1}{2}}$

- *Forward shift*[1]

$$\mathcal{Z}[f(k+1)\,\mathbb{1}(k)] = z\mathcal{Z}[f] - zf(0)$$

Example: $f(k) = a^{k+1}\,\mathbb{1}(k) \Rightarrow \mathcal{Z}[f] = z\frac{z}{z-a} - z = \frac{az}{z-a}$

[1] z is also called *forward shift operator*

Properties of Z-transforms

- *Backward shift* or *unit delay* [2]

$$\mathcal{Z}[f(k-1)\,\mathbb{1}(k)] = z^{-1}\mathcal{Z}[f]$$

Example: $f(k) = \mathbb{1}(k-1) \Rightarrow \mathcal{Z}[f] = \frac{z}{z(z-1)}$

- *Multiplication by k*

$$\mathcal{Z}[kf(k)] = -z\frac{d}{dz}\mathcal{Z}[f]$$

Example: $f(k) = k\,\mathbb{1}(k) \Rightarrow \mathcal{Z}[f] = \frac{z}{(z-1)^2}$

[2] z^{-1} is also called *backward shift operator*

Initial and final value theorems

Initial value theorem

$$f(0) = \lim_{z \to \infty} F(z)$$

Example: $f(k) = \mathbb{1}(k) - k\,\mathbb{1}(k) \Rightarrow F(z) = \frac{z}{z-1} - \frac{z}{(z-1)^2}$

$f(0) = \lim_{z \to \infty} F(z) = 1$

Final value theorem

$$\lim_{k \to +\infty} f(k) = \lim_{z \to 1} (z-1) F(z)$$

Example: $f(k) = \mathbb{1}(k) + (-0.7)^k\,\mathbb{1}(t) \Rightarrow F(z) = \frac{z}{z-1} + \frac{z}{z+0.7}$

$f(+\infty) = \lim_{z \to 1} (z-1) F(z) = 1$

The output of an ideal sampler $r^*(t) =$ series of impulses with values $r(kT)$:

$$r^*(t) = \sum_{k=0}^{\infty} r(kT)\delta(t - kT), \quad t > 0$$

$$L[r^*(t)] = \sum_{k=0}^{\infty} r(kT)e^{-ksT}, \qquad z = e^{sT}$$

$\Rightarrow$ The z-transform:

$$Z[r(t)] = Z[r^*(t)] = \sum_{k=0}^{\infty} r(kT)z^{-k}$$

Example. The z-transform of a unit step $u(t) = 1$

$$U(z) = Z[u(t)] = \sum_{k=0}^{\infty} u(kT)z^{-k} = \sum_{k=0}^{\infty} z^{-k}, \quad u(kT) = 1 \Rightarrow$$

$$U(z) = \frac{1}{1 - z^{-1}} = \frac{z}{z - 1}$$

Obs. The infinite geometric series may be written:
$(1 - bx)^{-1} = 1 + bx + (bx)^2 + ...$ *if* $(bx)^2 < 1$

Z transform table

x(t)	X(s)	X(z)
$\delta(t) = \begin{cases} 1, t = 0 \\ 0, t = kT, k \neq 0 \end{cases}$	1	1
$\delta(t - kT) = \begin{cases} 1, t = kT \\ 0, t \neq kT, \end{cases}$	e^{-kTs}	z^{-k}
$u(t) = 1$	$\frac{1}{s}$	$\frac{z}{z-1}$
t	$\frac{1}{s^2}$	$\frac{Tz}{(z-1)^2}$
e^{-at}	$\frac{1}{s+a}$	$\frac{z}{z-e^{-aT}}$
$1 - e^{-at}$	$\frac{1}{s(s+a)}$	$\frac{(1-e^{-aT})z}{(z-1)(z-e^{-aT})}$
$sin\omega t$	$\frac{\omega}{s^2+\omega^2}$	$\frac{zsin\omega T}{z^2-2zcos\omega T+1}$

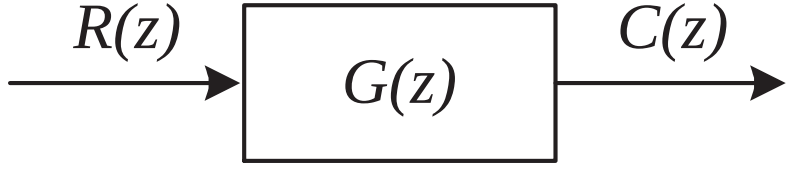
Properties of Z transform

$x(t)$	$X(z)$
$kx(t)$	$kX(z)$
$x_1(t) + x_2(t)$	$X_1(z) + X_2(z)$
$x(t+T)$	$zX(z) - zx(0)$
$tx(t)$	$-Tz\frac{dX(z)}{dz}$
$e^{-at}x(t)$	$X(ze^{aT})$
x(0), initial value	$lim_{z\to\infty}X(z)$ if the limit exists
x(∞), final value	$lim_{z\to 1}(z-1)X(z)$ if the limit exists and the system is stable

Discrete transfer functions and state-space models

Discrete transfer function

The transfer function - block diagram

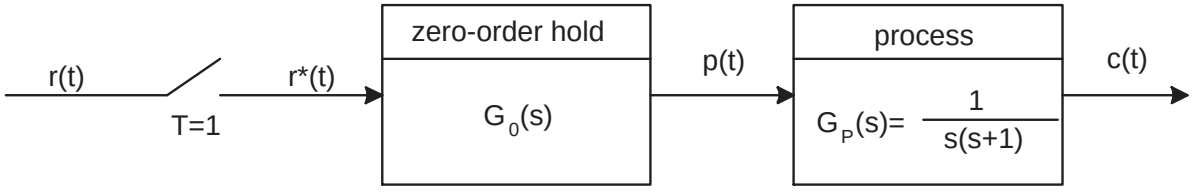


If $C(z)$ is the z-transform of the output signal and the input is $R(z) \Rightarrow$ the transfer function

$$\frac{C(z)}{R(z)} = G(z)$$

The transfer function of the open-loop system, with $T = 1$.

Example



Transfer function in s-domain:

$$G(s) = \frac{C(s)}{R^*(s)} = G_0(s)G_P(s) = \frac{1 - e^{-sT}}{s} \cdot \frac{1}{s(s+1)} = \frac{1 - e^{-sT}}{s^2(s+1)}$$

Partial fraction expansion:

$$G(s) = (1 - e^{-sT}) \left(\frac{1}{s^2} - \frac{1}{s} + \frac{1}{s+1} \right)$$

Example

$$G(z) = Z[G(s)] = (1 - z^{-1})Z\left[\frac{1}{s^2} - \frac{1}{s} + \frac{1}{s+1}\right]$$

Using the entries from the z-transform table:

$$G(z) = (1 - z^{-1})\left(\frac{Tz}{(z-1)^2} - \frac{z}{z-1} + \frac{z}{z - e^{-aT}}\right)$$

$$G(z) = \frac{(ze^{-T} - z + Tz) + (1 - e^{-T} - Te^{-T})}{(z-1)(z - e^{-T})}$$

For $T = 1$:

$$G(z) = \frac{ze^{-1} + 1 - 2e^{-1}}{(z-1)(z - e^{-1})} = \frac{0.3678z + 0.2644}{z^2 - 1.3678z + 0.3678}$$

Impulse response: $R(z) = 1 \Rightarrow$ the output $C(z) = G(z) \cdot 1$ may be obtained by dividing the denominator into the numerator as:

$$C(z) = 0.3678z + 0.2644 : z^2 - 1.3678z + 0.3678$$

$$C(z) = 0.3678z^{-1} + 0.7675z^{-2} + 0.9145z^{-3} + ...$$

$\Rightarrow$ the response at the sampling instants:

Example (system response)

$$C(z) = \sum_{k=0}^{\infty} c(kT)z^{-k}$$

In this case we have obtained $c(kT)$: c(0)=0, c(T)=0.3678, c(2T)=0.7675, c(3T)=0.9145...

$c(kT) =$ the values of $c(t)$ at $t = kT$

Discrete-time transfer function

- Let's apply the Z-transform to discrete-time linear systems

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ y(k) &= Cx(k)+Du(k) \end{cases}$$
$$x(0)=x_0$$

- Define $X(z)=\mathcal{Z}[x(k)]$, $U(z)=\mathcal{Z}[u(k)]$, $Y(z)=\mathcal{Z}[y(k)]$

- Apply linearity and forward shift rules

$$\begin{aligned} zX(z)-zx_0 &= AX(z)+BU(z) \\ Y(z) &= CX(z)+DU(z) \end{aligned}$$

Discrete-time transfer function

$$\begin{aligned} X(z) &= z(zI-A)^{-1}x_0 + (zI-A)^{-1}BU(z) \\ Y(z) &= \underbrace{zC(zI-A)^{-1}x_0}_{\text{Z-transform of natural response}} + \underbrace{(C(zI-A)^{-1}B+D)U(z)}_{\text{Z-transform of forced response}} \end{aligned}$$

Definition:

The transfer function of a discrete-time linear system (A,B,C,D) is the ratio

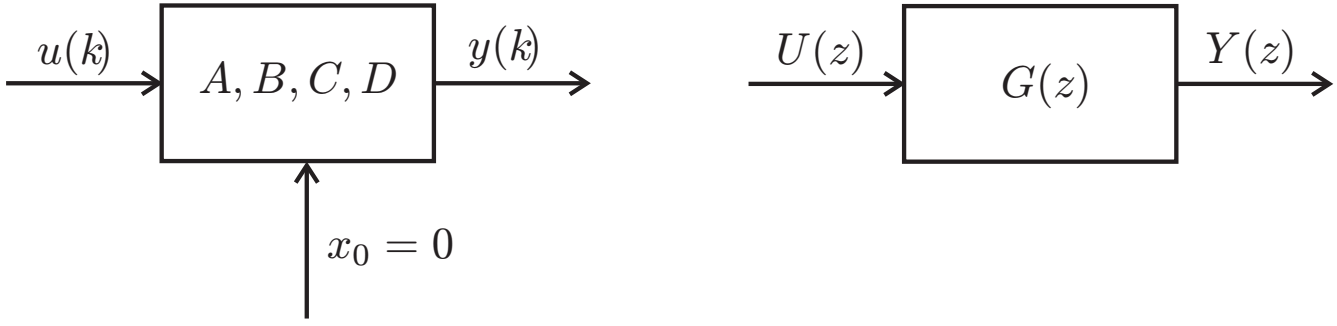
$$G(z) = C(zI-A)^{-1}B + D$$

between the Z-transform $Y(z)$ of the output and the Z-transform $U(z)$ of the input signals *for the initial state* $x_0 = 0$

MATLAB

```
»sys=ss(A,B,C,D,Ts);
»G=tf(sys)
```

Discrete-time transfer function



Example: The linear system

$$\begin{cases} x(k+1) &= \begin{bmatrix} 0.5 & 1 \\ 0 & -0.5 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 1 & -1 \end{bmatrix} x(k) \end{cases}$$

with sampling time $T_s = 0.1$ s has the transfer function

$$G(z) = \frac{-z + 1.5}{z^2 - 0.25}$$

Note: Even for discrete-time systems, the transfer function does not depend on the input $u(k)$. It's only a property of the linear system

MATLAB

```
»sys=ss([0.5 1;
  0 -0.5],[0;1],[1 -1],0,0.1);
»G=tf(sys)

Transfer function:
-z + 1.5
----------
s^2 - 0.25
```

Difference equations

- Consider the n^{th}-order difference equation forced by u

$$\begin{aligned} & a_n y(k-n) + a_{n-1} y(k-n+1) + \cdots + a_1 y(k-1) + y(k) \\ & = b_n u(k-n) + \cdots + b_1 u(k-1) \end{aligned}$$

- For zero initial conditions we get the transfer function

$$\begin{aligned} G(z) &= \frac{b_n z^{-n} + b_{n-1} z^{-n+1} + \cdots + b_1 z^{-1}}{a_n z^{-n} + a_{n-1} z^{-n+1} + \cdots + a_1 z^{-1} + 1} \\ &= \frac{b_1 z^{n-1} + \cdots + b_{n-1} z + b_n}{z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n} \end{aligned}$$

Difference equations

- **Example:** $3y(k-2)+2y(k-1)+y(k)=2u(k-1)$

$$G(z)=\frac{2z^{-1}}{3z^{-2}+2z^{-1}+1}=\frac{2z}{z^2+2z+3}$$

- **Note:** The same transfer function $G(z)$ is obtained from the equivalent matrix form

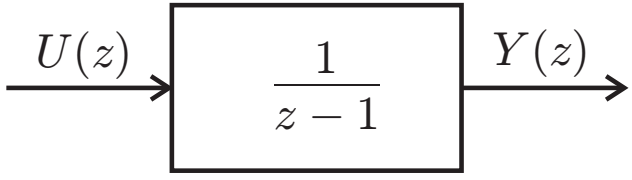
$$\begin{cases} x(k+1) &= \begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix} x(k)+\begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 2 \end{bmatrix} x(k) \end{cases}$$

$$\Rightarrow G(z)=\begin{bmatrix} 0 & 2 \end{bmatrix}\left(z\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}-\begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix}\right)^{-1}\begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Some common transfer functions

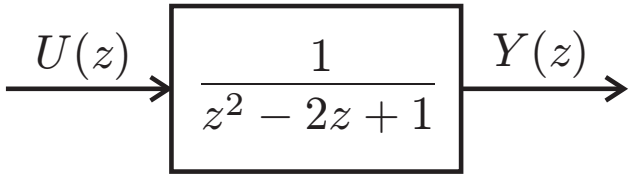
- *Integrator*

$$\begin{cases} x(k+1) &= x(k)+u(k) \\ y(k) &= x(k) \end{cases}$$



- *Double integrator*

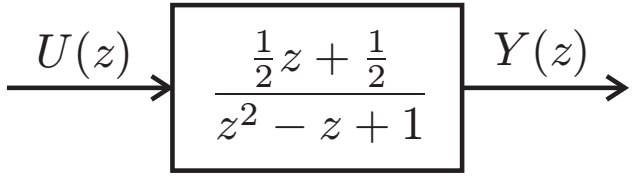
$$\begin{cases} x_1(k+1) &= x_1(k)+x_2(k) \\ x_2(k+1) &= x_2(k)+u(k) \\ y(k) &= x_1(k) \end{cases}$$

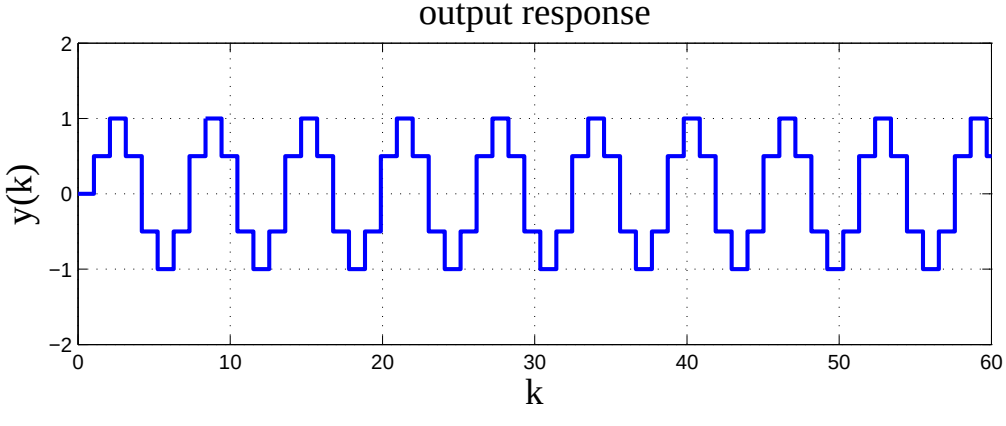


Some common transfer functions

- *Oscillator*

$$\begin{cases} x_1(k+1) &= x_1(k) - x_2(k) + u(k) \\ x_2(k+1) &= x_1(k) \\ y(k) &= \frac{1}{2}x_1(k) + \frac{1}{2}x_2(k) \end{cases}$$



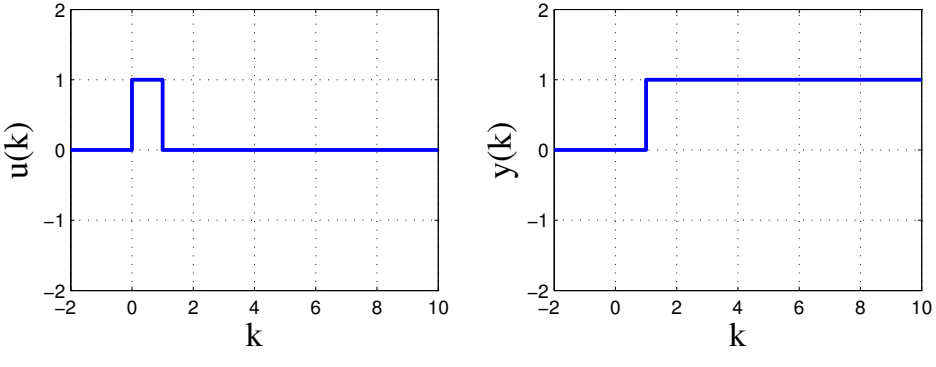


Impulse response

- Consider the impulsive input $u(k) = \delta(k)$, $U(z) = 1$. The corresponding output $y(k)$ is called *impulse response*
- The Z-transform of $y(k)$ is $Y(z) = G(z) \cdot 1 = G(z)$
- Therefore the impulse response coincides with the *inverse Z-transform* $g(k)$ of the transfer function $G(z)$

Example (integrator:)

$$\begin{aligned} u(k) &= \delta(k) \\ y(k) &= \mathcal{Z}^{-1}\left[\frac{1}{z-1}\right] = \mathbb{1}(k-1) \end{aligned}$$



Poles, eigenvalues, modes

- Linear discrete-time system

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ y(k) &= Cx(k)+Du(k) \end{cases} \qquad G(z) = C(zI-A)^{-1}B+D \triangleq \frac{N_G(z)}{D_G(z)}$$
$$x(0)=0$$

- Use the adjugate matrix to represent the inverse of $zI-A$

$$C(zI-A)^{-1}B+D = C\frac{C\,\mathrm{Adj}(zI-A)B}{\det(zI-A)}+D$$

- The denominator $D_G(z) = \det(zI-A)$!

The poles of $G(z)$ coincide with the eigenvalues of A

- Well, as in continuous-time, not always ...

Steady-state solution and DC gain

- Let A asymptotically stable ($|\lambda_i| < 1$). Natural response vanishes asymptotically
- Assume constant $u(k) \equiv u_r$, $\forall k \in \mathbb{N}$. What is the asymptotic value $x_r = lim_{k\to\infty} x(k)$?

  Impose $x_r(k+1) = x_r(k) = Ax_r + Bu_r$ and get $x_r = (I-A)^{-1}Bu_r$

  The corresponding *steady-state* output $y_r = Cx_r + Du_r$ is

$$y_r = \underbrace{(C(I-A)^{-1}B + D)}_{\text{DC gain}} u_r$$

- Cf. final value theorem:

$$\begin{aligned} y_r &= \lim_{k\to+\infty} y(k) = \lim_{z\to 1}(z-1)Y(z) \\ &= \lim_{z\to 1}(z-1)G(z)U(z) = \lim_{z\to 1}(z-1)G(z)\frac{u_r z}{z-1} \\ &= G(1)u_r = (C(I-A)^{-1}B + D)u_r \end{aligned}$$

- $G(1)$ is called the *DC gain* of the system

Example - Student dynamics

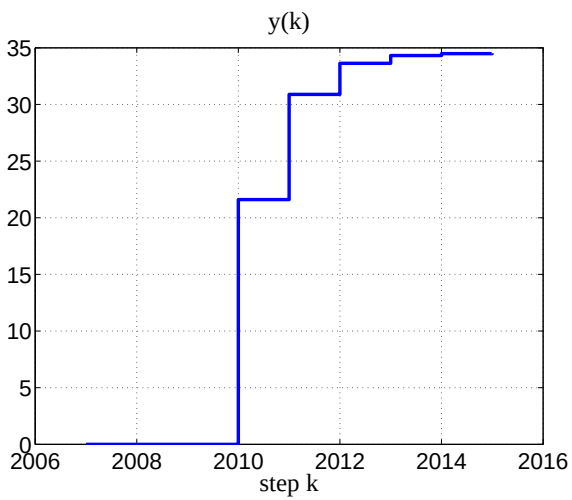
- Recall student dynamics in 3-years undergraduate course

$$\begin{cases} x(k+1) &= \begin{bmatrix} .2 & 0 & 0 \\ .6 & .15 & 0 \\ 0 & .8 & .08 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 0 & .9 \end{bmatrix} x(k) \end{cases}$$

- DC gain:

$$\begin{bmatrix} 0 & 0 & .9 \end{bmatrix} \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} .2 & 0 & 0 \\ .6 & .15 & 0 \\ 0 & .8 & .08 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \approx 0.69$$

- Transfer function: $G(z) = \dfrac{0.432}{z^3 - 0.43z^2 + 0.058z - 0.0024}$, $G(1) \approx 0.69$



MATLAB

```
»A=[b1 0 0; a1 b2 0; 0 a2 b3];
»B=[1;0;0];
»C=[0 0 a3];
»D=[0];
»sys=ss(A,B,C,D,1);
»dcgain(sys)

ans =

    0.6905
```

- For $u(k) \equiv 50$ students enrolled steadily, $y(k) \to 0.69 \cdot 50 \approx 34.5$ graduate

Linear algebra recalls: Change of coordinates

- Let $\{v_1, \ldots, v_n\}$ be a *basis* of $\mathbb{R}^n$ ($= n$ linearly independent vectors)
- The *canonical basis* of $\mathbb{R}^n$ is $e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, $e_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$, $\ldots$, $e_n = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$
- A vector $w \in \mathbb{R}^n$ can be expressed as a *linear combination* of the basis vectors, whose coefficients are the *coordinates* in the corresponding basis

$$w = \sum_{i=1}^{n} x_i e_i = \sum_{i=1}^{n} z_i v_i$$

- The relation between the coordinates $x = [x_1 \ \ldots \ x_n]'$ in the canonical basis and the coordinates $z = [z_1 \ \ldots \ z_n]'$ in the new basis is

$$x = Tz$$

where $T = \begin{bmatrix} v_1 & \ldots & v_n \end{bmatrix}$ ($=$ *coordinate transformation matrix*)

Algebraically equivalent systems

- Consider the linear system

$$\begin{cases} x(k+1) &= Ax(k)+Bu(k) \\ y(k) &= Cx(k)+Du(k) \end{cases}$$

$$x(0)=x_0$$

- Let T be invertible and define the change of coordinates $x=Tz$, $z=T^{-1}x$

$$\begin{cases} z(k+1) &= T^{-1}x(k+1)=T^{-1}\left(Ax(k)+Bu(k)\right)=T^{-1}ATz(k)+T^{-1}Bu(k) \\ y(k) &= CTz(k)+Du(k) \end{cases}$$

$$z_0=T^{-1}x_0$$

and hence

$$\begin{cases} z(k+1) &= \tilde{A}z(k)+\tilde{B}u(k) \\ y(k) &= \tilde{C}z(k)+\tilde{D}u(k) \end{cases}$$

$$z(0)=T^{-1}x_0$$

- The dynamical systems (A,B,C,D) and $(\tilde{A},\tilde{B},\tilde{C},\tilde{D})$ are called *algebraically equivalent*

Transfer function of algebraically equivalent systems

- Consider two algebraically equivalent systems (A, B, C, D) and $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$

$$\begin{array}{ll} \tilde{A} = T^{-1}AT & \tilde{C} = CT \\ \tilde{B} = T^{-1}B & \tilde{D} = D \end{array}$$

- (A, B, C, D) and $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$ have the same transfer functions:

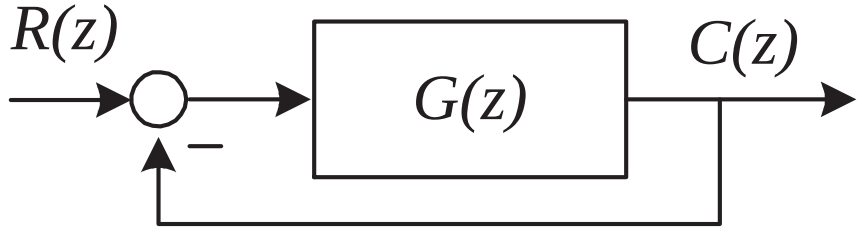
$$\begin{aligned} \tilde{G}(z) &= \tilde{C}(zI - \tilde{A})^{-1}\tilde{B} + \tilde{D} \\ &= CT(zT^{-1}IT - T^{-1}AT)^{-1}T^{-1}B + D \\ &= CTT^{-1}(zI - A)TT^{-1}B + D \\ &= C(zI - A)^{-1}B + D \\ &= G(z) \end{aligned}$$

- The same result holds for continuous-time linear systems

Stability analysis of discrete-time systems

Consider the sampled data system:

Closed-loop sampled data systems



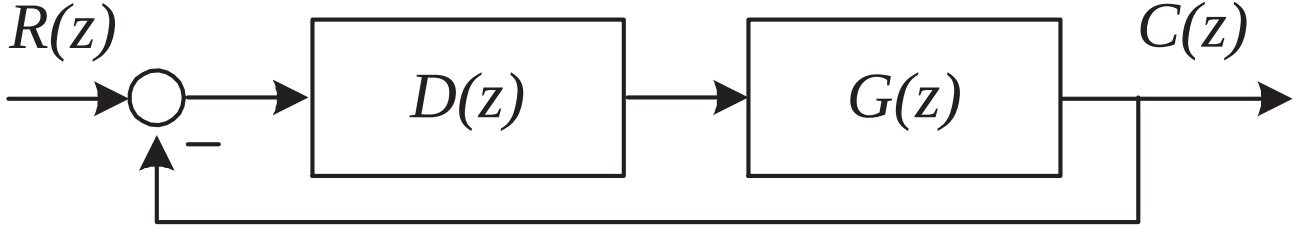
The closed-loop transfer function:

$$\frac{C(z)}{R(z)} = T(z) = \frac{G(z)}{1 + G(z)}$$

$G(z)$: is the z transform of $G_0(s)G_p(s)$, where $G_0(s) =$ ZOH and $G_p(s)$ - the plant transfer function.

Closed-loop sampled data systems

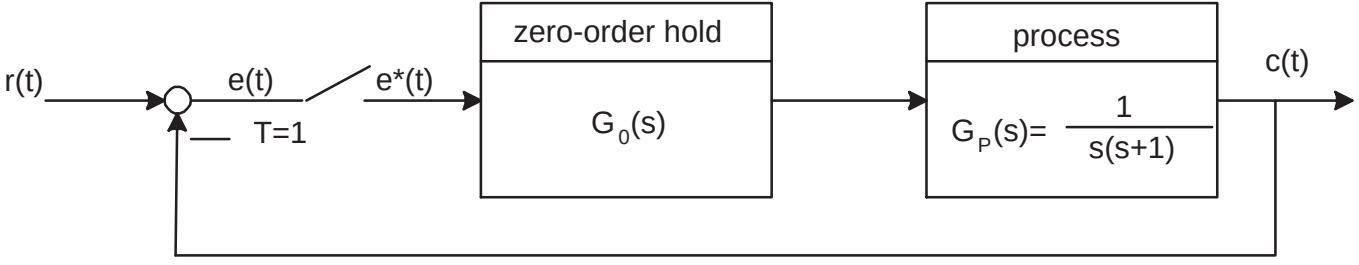
Negative feedback closed-loop system:



$$\frac{C(z)}{R(z)} = T(z) = \frac{G(z)D(z)}{1 + G(z)D(z)}$$

Example

Response



$$\frac{C(z)}{R(z)} = \frac{G(z)}{1+G(z)} = \frac{0.3678z + 0.2644}{z^2 - z + 0.6322}$$

Input - unit step:

$$R(z) = \frac{z}{z-1}$$

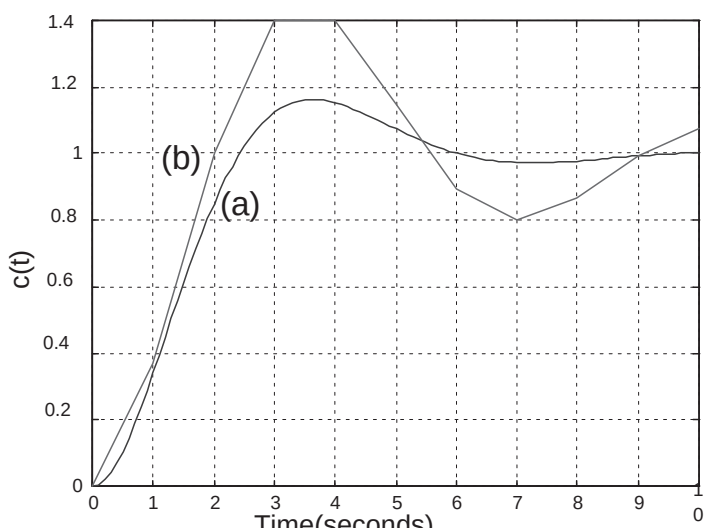
$$C(z) = \frac{z(0.3678z + 0.2644)}{(z-1)(z^2 - z + 0.6322)} = \frac{0.3678z^2 + 0.2644z}{z^3 - 2z^2 + 1.6322z - 0.6322}$$

Response

Dividing the numerator by the denominator:

$$C(z) = 0.3678z^{-1} + z^{-2} + 1.4z^{-3} + 1.4z^{-4} + 1.147z^{-5} + ...$$

The response (a) continuous, T=0; (b) discrete, T=1



The overshoot of the sampled data system is 40% in contrast to 17% for a continuous system. The settling time is twice as long as of the continuous system.

Stability analysis in z-plane

- A linear continuous feedback control system is stable if all the poles of the closed-loop transfer function $T(s)$ lie in the left half of the s-plane
- The z-plane is related to the z-plane by the transformation:

$$z = e^{sT} = e^{(\sigma + j\omega)T} \quad \text{or} \quad |z| = e^{\sigma T}, \quad \angle z = \omega T$$

- In the left-hand s-plane, the real part of s, $\sigma < 0$ and therefore the related magnitude of z varies between 0 and 1 ($0 < e^{\sigma T} < 1$). $\Rightarrow$ unit circle.
- *A sampled data system is stable if all the poles of the closed-loop transfer function $T(z)$ lie within the unit circle of the z-plane.*

Stability analysis in z-plane

Consider a feedback control system with the open-loop transfer function:

$$\begin{aligned} G(s) &= \frac{k}{s(s+1)} \Rightarrow G(z) = Z\{G_0(s)G(s)\} \\ &= Z\left\{\frac{1-e^{-sT}}{s}\frac{k}{s(s+1)}\right\} = (1-z^{-1})Z\left\{\frac{k}{s^2(s+1)}\right\} \end{aligned}$$

For $T = 1$:

$$G(z) = \frac{k(0.367z + 0.264)}{z^2 - 1.367z + 0.367}$$

Closed-loop transfer function: Example

$$T(z) = \frac{G(z)}{1 + G(z)} = \frac{k(0.367z + 0.264)}{z^2 + (0.367k - 1.367)z + 0.264k + 0.367}$$

The poles of $T(z)$ are the solutions of the characteristic eq.
$q(z) = 1 + G(z) = 0$

- For $k = 1$ we have: $q(z) = z^2 - z + 0.631$.

  The poles : $z_1 = 0.5 + 0.6173j$ and $z_2 = 0.5 - 0.6173j$. $\Rightarrow$ The system is stable.

- For $k = 10$ we have: $q(z) = z^2 + 2.31z + 3.01$.

  The poles : $z_1 = -1.1550 + 1.2946j$ and $z_2 = 1.1550 + 1.2946j$. $\Rightarrow$ The system is not stable.

Stability

- Consider the nonlinear system

$$\begin{cases} x(k+1) &= f(x(k), u_r) \\ y(k) &= g(x(k), u_r) \end{cases}$$

and let x_r an equilibrium state, $f(x_r, u_r) = x_r$

Definition

The equilibrium state x_r is *stable* if for each initial conditions $x(0)$ "close enough" to x_r, the corresponding trajectory $x(k)$ remains near x_r for all $k \in \mathbb{N}$ [a]

[a] Analytic definition: $\forall \epsilon > 0 \; \exists \delta > 0 : \|x(0) - x_r\| < \delta \Rightarrow \|x(k) - x_r\| < \epsilon, \; \forall k \in \mathbb{N}$

- The equilibrium point x_r is called *asymptotically stable* if it is stable and $x(k) \to x_r$ for $k \to \infty$
- Otherwise, the equilibrium point x_r is called *unstable*

Stability of first-order linear systems

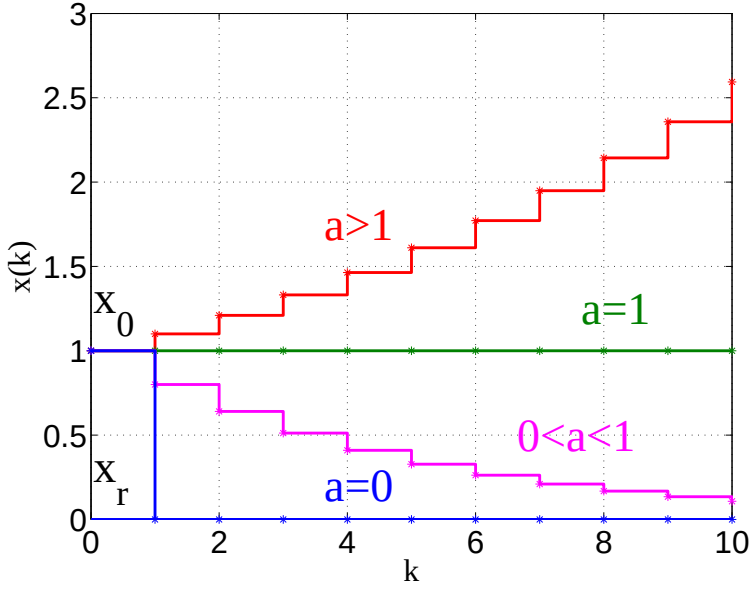
- Consider the first-order linear system

$$x(k+1) = ax(k) + bu(k)$$

- $x_r = 0$, $u_r = 0$ is an equilibrium pair
- For $u(k) \equiv 0$, $\forall k = 0, 1, \ldots$, the solution is

$$x(k) = a^k x_0$$

- The origin $x_r = 0$ is
  - unstable if $|a| > 1$
  - stable if $|a| \leq 1$
  - asymptotically stable if $|a| < 1$



Stability of discrete-time linear systems

Since the natural response of $x(k+1) = Ax(k) + Bu(k)$ is $x(k) = A^k x_0$, the stability properties depend only on A. We can therefore talk about *system stability* of a discrete-time linear system (A, B, C, D)

Theorem:

Let $\lambda_1, \ldots, \lambda_m$, $m \leq n$ be the eigenvalues of $A \in \mathbb{R}^{n \times n}$. The system $x(k+1) = Ax(k) + Bu(k)$ is

- asymptotically stable iff $|\lambda_i| < 1$, $\forall i = 1, \ldots, m$
- (marginally) stable if $|\lambda_i| \leq 1$, $\forall i = 1, \ldots, m$, and the eigenvalues with unit modulus have equal algebraic and geometric multiplicity [a]
- unstable if $\exists\ i$ such that $|\lambda_i| > 1$

[a] Algebraic multiplicity of λ_i = number of coincident roots λ_i of $\det(\lambda I - A)$. Geometric multiplicity of λ_i = number of linearly independent eigenvectors v_i, $Av_i = \lambda_i v_i$

The stability properties of a discrete-time linear system only depend on the **modulus** of the eigenvalues of matrix A

Stability of discrete-time linear systems

Proof:

- The natural response is $x(k) = A^k x_0$
- If matrix A is diagonalizable[1] , $A = T\Lambda T^{-1}$,

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \Rightarrow A^k = T \begin{bmatrix} \lambda_1^k & 0 & \dots & 0 \\ 0 & \lambda_2^k & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n^k \end{bmatrix} T^{-1}$$

- Take any eigenvalue $\lambda = \rho e^{j\theta}$:

$$|\lambda^k| = \rho^k |e^{jk\theta}| = \rho^k$$

- A is always diagonalizable if algebraic multiplicity - geometric multiplicity

□

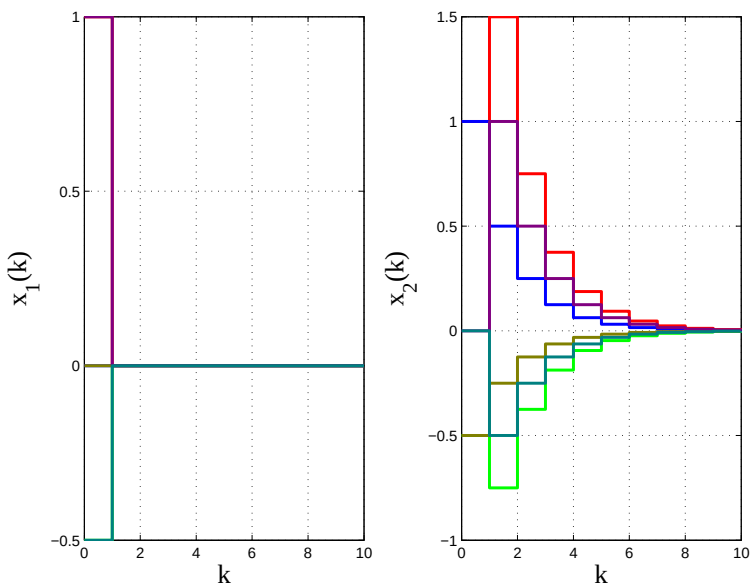
[1] If A is not diagonalizable, it can be transformed to Jordan form. In this case the natural response $x(t)$ contains modes $k^j\lambda^k$, $j = 0, 1, \dots$, alg. multiplicity = geom. multiplicity

Example 1

$$\begin{cases} x(k+1) = \begin{bmatrix} 0 & 0 \\ 1 & \frac{1}{2} \end{bmatrix} x(k) \\ x(0) = \begin{bmatrix} x_{10} \\ x_{20} \end{bmatrix} \end{cases} \Rightarrow \text{ eigenvalues of } A\text{: } \left\{0, \frac{1}{2}\right\}$$

solution:

$$\begin{cases} x_1(k) &= 0,\ k = 1, 2, \ldots \\ x_2(k) &= \left(\frac{1}{2}\right)^{k-1} x_{10} + \left(\frac{1}{2}\right)^{k} x_{20},\ k = 1, 2, \ldots \end{cases}$$



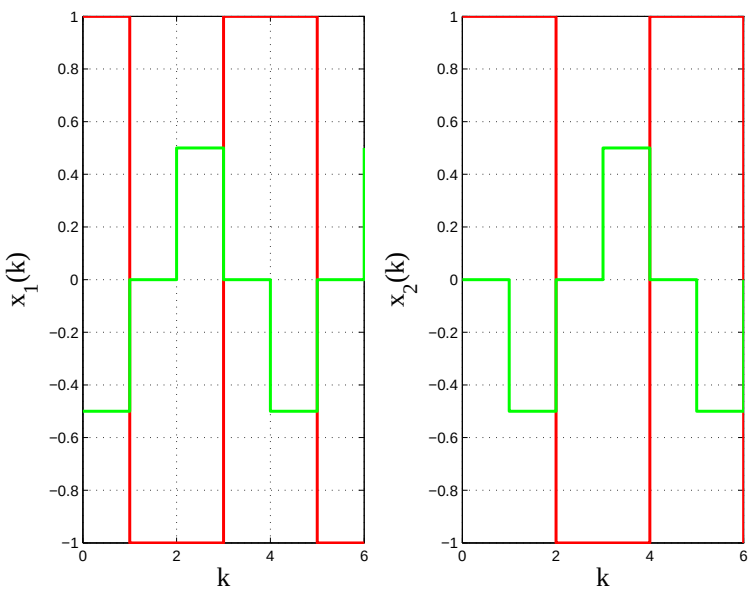
asymptotically stable

Example 2

$$\begin{cases} x(k+1) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} x(k) \\ x(0) = \begin{bmatrix} x_{10} \\ x_{20} \end{bmatrix} \end{cases} \Rightarrow \text{ eigenvalues of } A\text{: } \{+j, -j\}$$

solution:

$$\begin{cases} x_1(k) &= x_{10} \cos\frac{k\pi}{2} + x_{20} \sin\frac{k\pi}{2}, \ k = 0, 1, \ldots \\ x_2(k) &= x_{10} \sin\frac{k\pi}{2} + x_{20} \cos\frac{k\pi}{2}, \ k = 0, 1, \ldots, \end{cases}$$



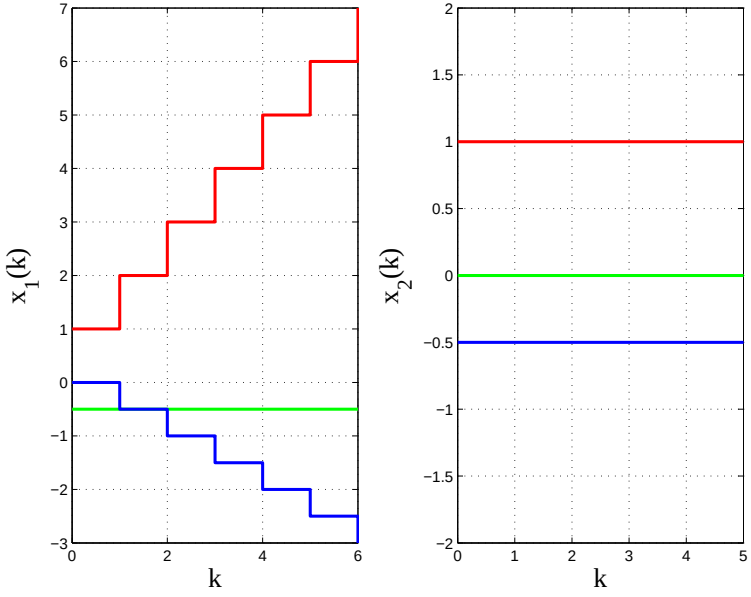
marginally stable

Example 3

$$\begin{cases} x(k+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(k) \\ x(0) = \begin{bmatrix} x_{10} \\ x_{20} \end{bmatrix} \end{cases} \Rightarrow \text{ eigenvalues of } A\text{: } \{1, 1\}$$

$$\begin{cases} x_1(k) = x_{10} + x_{20}k, \ k = 0, 1, \ldots \\ x_2(k) = x_{20}, \ k = 0, 1, \ldots \end{cases}$$

Note: A is not diagonalizable !



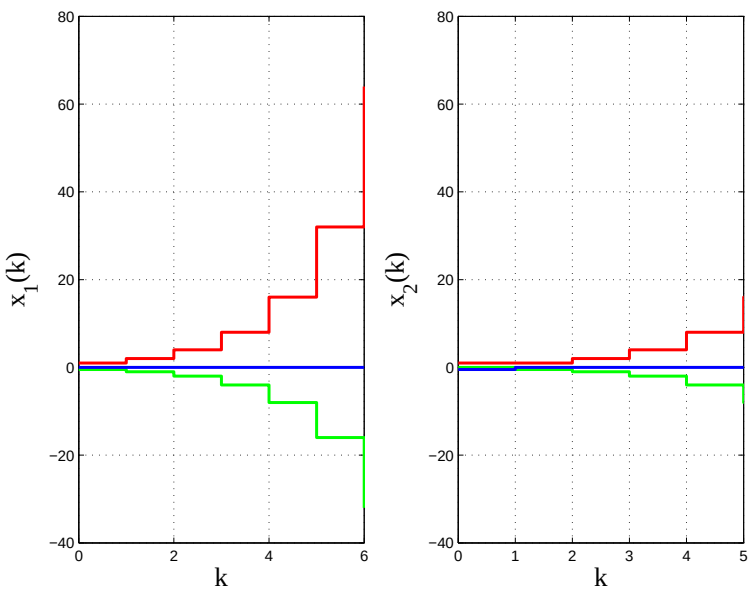
unstable

Example 4

$$\begin{cases} x(k+1) = \begin{bmatrix} 2 & 0 \\ 1 & 0 \end{bmatrix} x(k) \\ x(0) = \begin{bmatrix} x_{10} \\ x_{20} \end{bmatrix} \end{cases} \Rightarrow \text{ eigenvalues of } A\text{: } \{0, 2\}$$

solution:

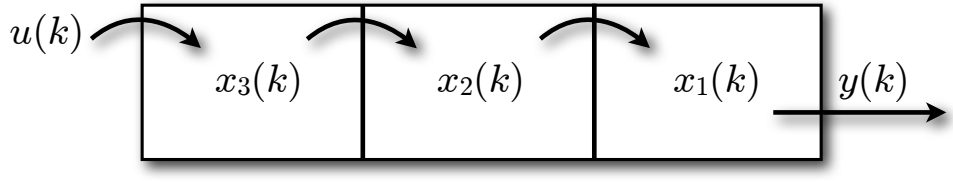
$$\begin{cases} x_1(k) &= 2^k x_{10},\ k = 0, 1, \ldots \\ x_2(k) &= 2^{k-1} x_{10},\ k = 1, 2, \ldots \end{cases}$$



unstable

Zero eigenvalues

- Modes λ_i=0 determine *finite-time* convergence to zero.
- This has no continuous-time counterpart, where instead all converging modes tend to zero in infinite time ($e^{\lambda_i t}$)
- Example: dynamics of a buffer



$$\begin{cases} x_1(k+1) = x_2(k) \\ x_2(k+1) = x_3(k) \\ x_3(k+1) = u(k) \\ y(k) = x_1(k) \end{cases} \Rightarrow \begin{cases} x(k+1) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(k) \end{cases}$$

- Natural response: $A^3 x(0) = 0$ for all $x(0) \in \mathbb{R}^3$
- For $u(k) \equiv 0$, the buffer deploys after at most 3 steps !

Summary of stability conditions for linear systems

system		*continuous-time*	*discrete-time*
asympt. stable	$\forall i = 1, \ldots, n$	$\Re(\lambda_i) < 0$	$\lvert\lambda_i\rvert < 1$
unstable	$\exists i$ such that	$\Re(\lambda_i) > 0$	$\lvert\lambda_i\rvert > 1$
stable	$\forall i, \ldots, n$	$\Re(\lambda_i) \leq 0$	$\lvert\lambda_i\rvert \leq 1$
	and $\forall \lambda_i$ such that algebraic = geometric mult.	$\Re(\lambda_i) = 0$	$\lvert\lambda_i\rvert = 1$

Discrete-time models of continuous-time processes and controllers (sampling)

Exact sampling

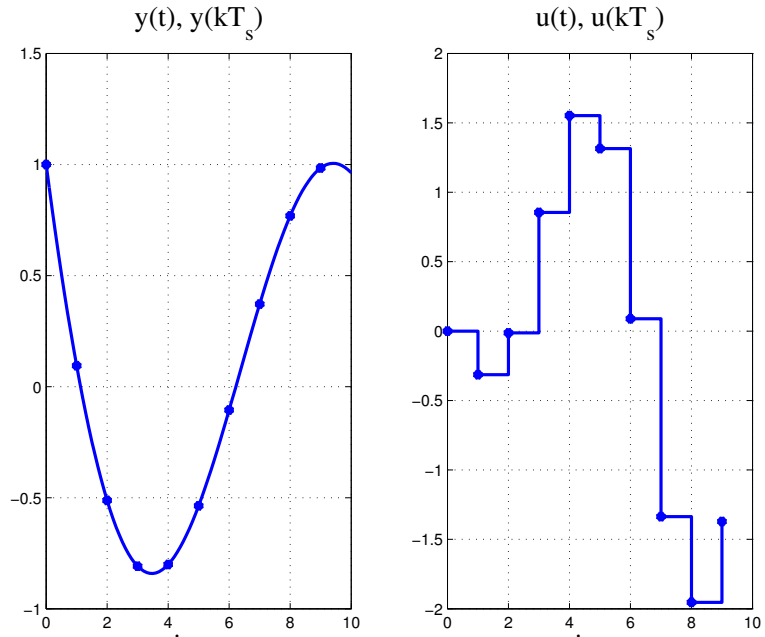
- Consider the continuos-time system

$$\begin{cases} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \\ x(0) &= x_0 \end{cases}$$

- We want to characterize the value of $x(t), y(t)$ at the time instants $t = 0, T_s, 2T_s, \ldots, kT_s, \ldots,$ **under the assumption that the input $u(t)$ is constant during each sampling interval** (*zero-order hold, ZOH*)

$$u(t) = \bar{u}(k),\ kT_s \leq t < (k+1)T_s$$

- $\bar{x}(k) \triangleq x(kT_s)$ and $\bar{y}(k) \triangleq y(kT_s)$ are the state and the output samples at the k^{th} sampling instant, respectively



Exact sampling

- Let us evaluate the response of the continuous-time system between time $t_0 = kT_s$ and $t = (k+1)T_s$ from the initial condition $x(t_0) = x(kT_s)$ using Lagrange formula:

$$x(t) = e^{A(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{A(t-\sigma)}Bu(\sigma)d\sigma = e^{A((k+1)T_s - kT_s)}x(kT_s) + \int_{kT_s}^{(k+1)T_s} e^{A((k+1)T_s-\sigma)}Bu(\sigma)d\sigma$$

- Since the input $u(t)$ is piecewise constant, $u(\sigma) \equiv \bar{u}(k),\ kT_s \le \sigma < (k+1)T_s$. By setting $\tau = \sigma - kT_s$ we get

$$x((k+1)T_s) = e^{AT_s}x(kT_s) + \left(\int_0^{T_s} e^{A(T_s-\tau)}d\tau\right) Bu(kT_s)$$

and hence

$$\bar{x}(k+1) = e^{AT_s}\bar{x}(k) + \left(\int_0^{T_s} e^{A(T_s-\tau)}d\tau\right) B\bar{u}(k)$$

which is a linear difference relation between $\bar{x}(k)$ and $\bar{u}(k)$!

Exact sampling

- The discrete-time system

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

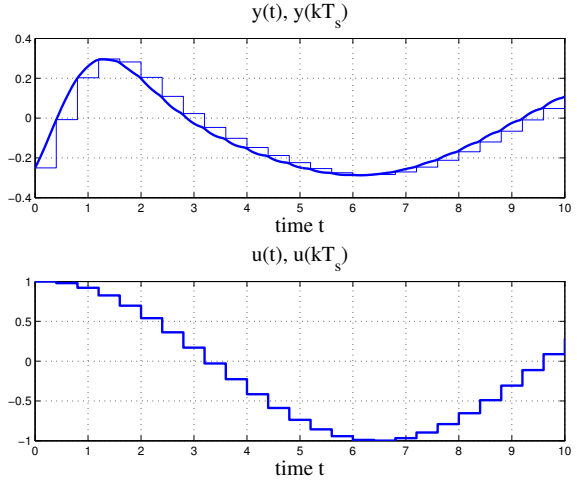
depends on the original continuous-time system through the relations

$$\bar{A} \triangleq e^{AT_s}, \quad \bar{B} \triangleq \left(\int_0^{T_s} e^{A(T_s-\tau)} d\tau \right) B, \quad \bar{C} \triangleq C, \quad \bar{D} \triangleq D$$

- If $u(t)$ is piecewise constant, $(\bar{A},\bar{B},\bar{C},\bar{D})$ provides the *exact* evolution of state and output samples at discrete times kT_s

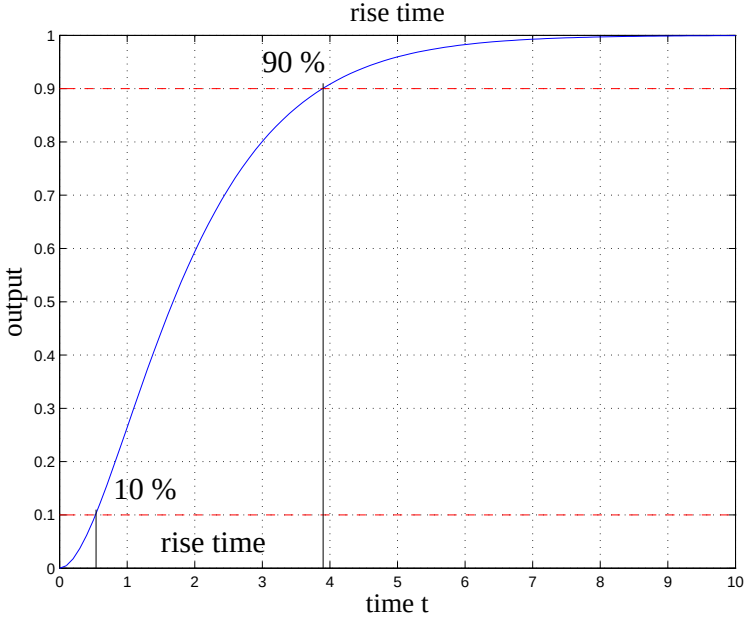
MATLAB

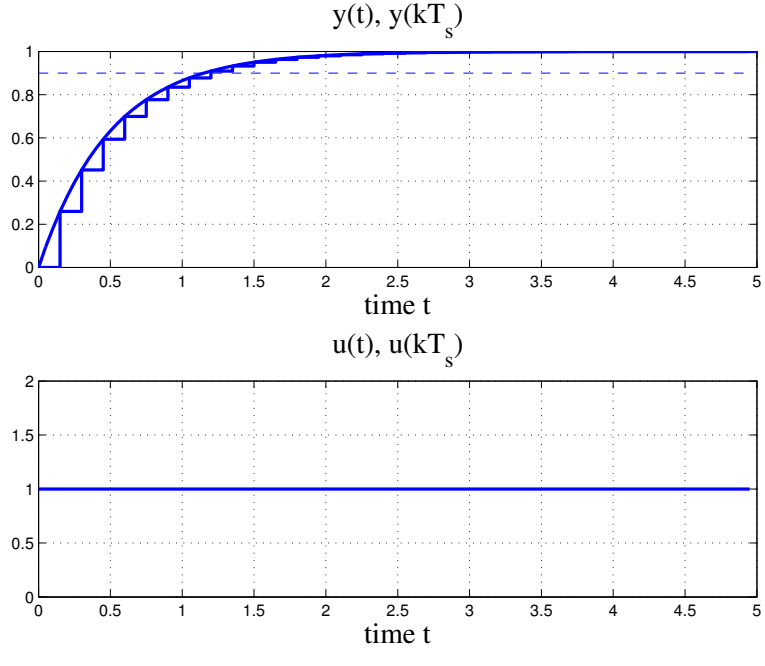
```
sys=ss(A,B,C,D);
sysd=c2d(sys,Ts);
[Ab,Bb,Cb,Db]=ssdata(sysd);
```



Exact sampling

Rule of thumb: $T_s \approx \frac{1}{10}$ of the *rise time* = time to move from 10% to 90% of the steady-state value, for input $u(t) \equiv 1, x(0) = 0$

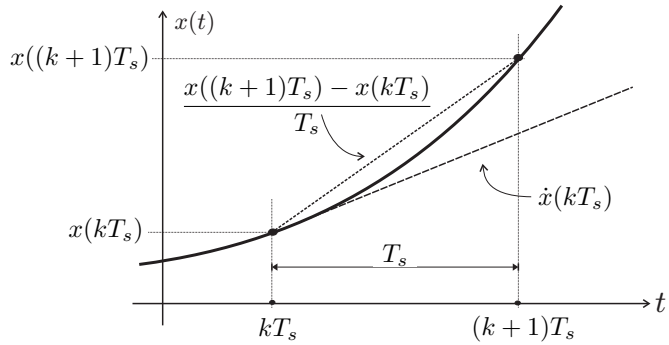




More on the choice of sampling time in the second part of the course ...

Approximate sampling - Euler's method

$$\dot{x}(kT_s) \approx \frac{x((k+1)T_s) - x(kT_s)}{T_s}$$



Leonhard Paul Euler
(1707-1783)

- For nonlinear systems $\dot{x}(t) = f(x(t), u(t))$:

$$\bar{x}(k+1) = \bar{x}(k) + T_s f(\bar{x}(k), \bar{u}(k))$$

- For linear systems $\dot{x}(t) = Ax(t) + Bu(t)$:

$$x((k+1)T_s) = (I + T_s A)x(kT_s) + T_s Bu(kT_s)$$

$$\bar{A} \triangleq I + AT_s, \quad \bar{B} \triangleq T_s B, \quad \bar{C} \triangleq C, \quad \bar{D} \triangleq D$$

- Note that $e^{T_s A} = I + T_s A + \ldots + \frac{T_s^n A^n}{n!} + \ldots$
  Therefore when T_s is small Euler's method and exact sampling are similar

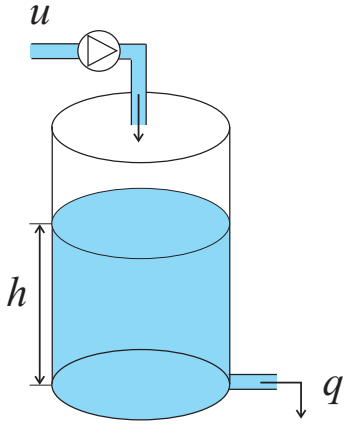
Example - Hydraulic system

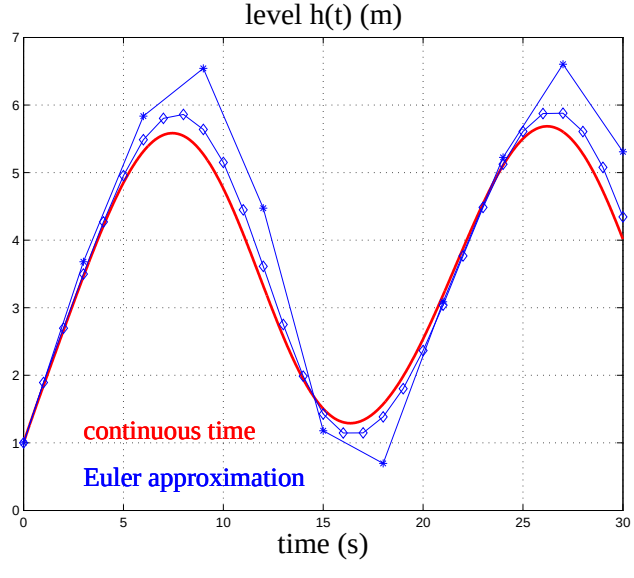
Continuous time:

$$\begin{cases} \frac{d}{dt}h(t) &= -\frac{a\sqrt{2g}}{A}\sqrt{h(t)} + \frac{1}{A}u(t) \\ q(t) &= a\sqrt{2g}\sqrt{h(t)} \end{cases}$$

Discrete time:

$$\begin{cases} \bar{h}(k+1) &= \bar{h}(k) - \frac{T_s a\sqrt{2g}}{A}\sqrt{\bar{h}(k)} + \frac{T_s}{A}\bar{u}(k) \\ \bar{q}(k) &= a\sqrt{2g}\sqrt{\bar{h}(k)} \end{cases}$$





Digitizing analog t.f. (ver. 1)

- One approach in digitizing an analog transfer function is transformation into z-domain, as described in the previous subsections. Recap.
- Given a tf in s-domain: $G(s)$. The z-transform is obtained as:

$$G(z) = Z\{G_0(s)G(s)\} = Z\left\{\frac{1-e^{-sT}}{s}G(s)\right\}$$

$$G(z) = (1-z^{-1})Z\left\{\frac{G(s)}{s}\right\}$$

where $G_0(s)$ - ZOH transfer function and T - sampling time.

Digitizing analog t.f. (ver. 2)

- The variable z can be approximated by various substitutions for s in the transfer function of the continuous system.
- Compare: multiplying by s = differentiation; multiplying by z^{-1} = delay of one sampling interval.
- Obs. The time derivative at a moment k can be approximated by:

$$\frac{de}{dt}\,|_k = \frac{e_k - e_{k-1}}{T}$$

In z-domain, where $e_k \rightarrow E(z)$:

$$\frac{E(z) - E(z) \cdot z^{-1}}{T} = E(z)\frac{1 - z^{-1}}{T}$$

Digitizing analog t.f. (ver. 2)

- Multiplying $E(z)$ by $\frac{1-z^{-1}}{T}$ is equivalent to multiplying $E(s)$ by s.
- $\Rightarrow$ Simple substitution:

$$s = \frac{1 - z^{-1}}{T}$$

- A more exact substitution for s (Tustin transformation) is:

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

- This more complicated substitution allows close approximation of the behavior of the analog controller using a longer sampling interval (T).

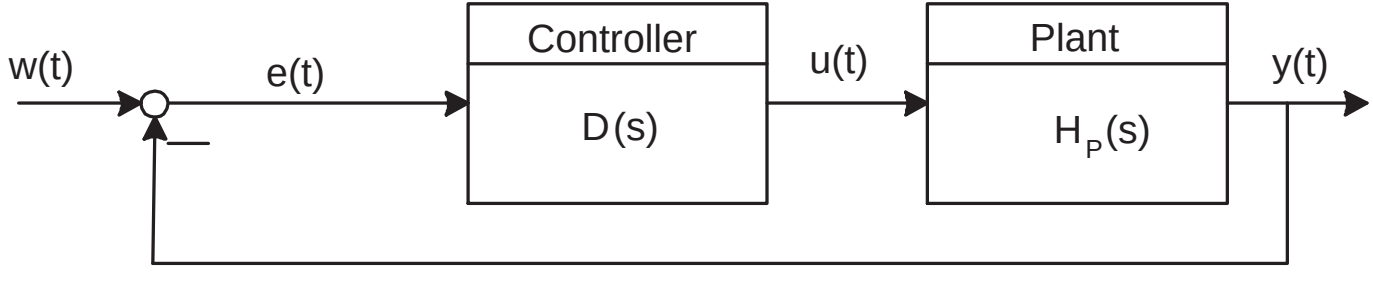
Example. A continuous process having the transfer function:

Example

$$H_P(s) = \frac{Y(s)}{U(s)} = \frac{10}{s^3 + 7s^2 + 6s}$$

is to be controlled in the closed-loop with unity negative feedback, using a forward path lead compensator of transfer function:

$$D(s) = \frac{1.5(s+1)}{s+3}$$



The controller is to be implemented in digital form. We shall investigate the performance of implementations having a sampling period of 0.1s and being converted into the z-domain by:

Example

a) Simple substitution for z

b) Tustin method

c) The z-transform method

a) Substitution:

$$s = \frac{1 - z^{-1}}{T} \text{ with T=0.1: } s = 10(1 - z^{-1})$$

$$D(z) = \frac{1.5[10(1 - z^{-1}) + 1]}{10(1 - z^{-1}) + 3} = \frac{1.2692 - 1.1538z^{-1}}{1 - 0.7692z^{-1}}$$

Example

b) Tustin substitution: $s = 2(1 - z^{-1})/(T(1 + z^{-1}))$ the transfer function is:

$$D(z) = \frac{1.5\left[\frac{2(1-z^{-1})}{T(1+z^{-1})} + 1\right]}{\left[\frac{2(1-z^{-1})}{T(1+z^{-1})} + 3\right]} = \frac{1.3696 - 1.2391z^{-1}}{1 - 0.7391z^{-1}}$$

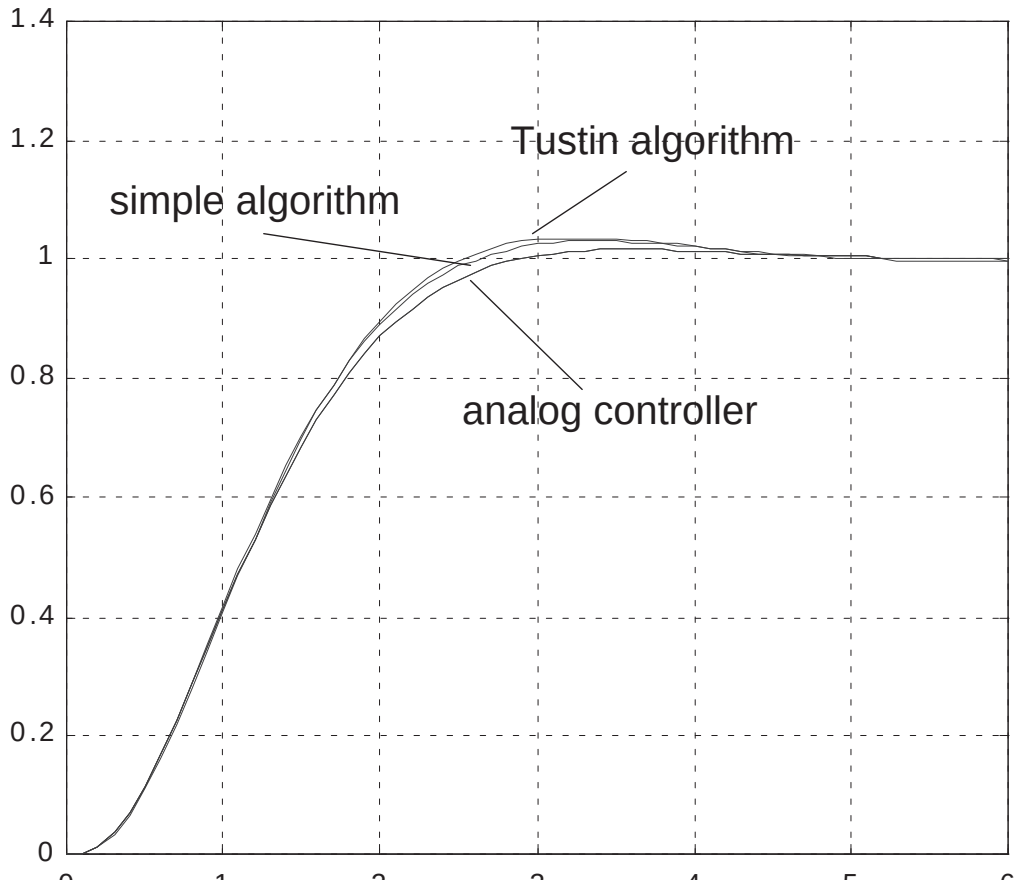
c) The z-transform method:

$$D(z) = Z\left(\frac{1 - e^{-sT}}{s}G(s)\right) = (1-z^{-1})Z\left(\frac{1.5(s+1)}{s(s+3)}\right)$$

$$D(z) = \frac{1.50 - 1.3704z^{-1}}{1 - 0.7408z^{-1}}$$

Step response of the closed-loop system in all three cases:

Example

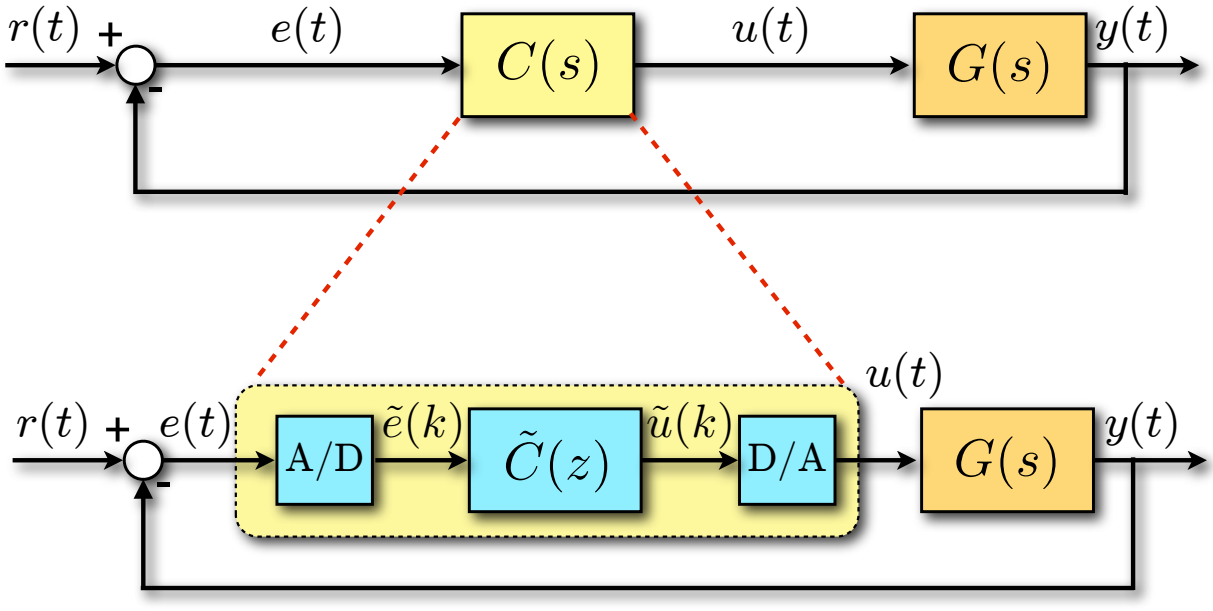


Example

Conversion type	*simple*	*Tustin*
overshoot %	3.6	2.9
peak time, sec	3.2	3.35

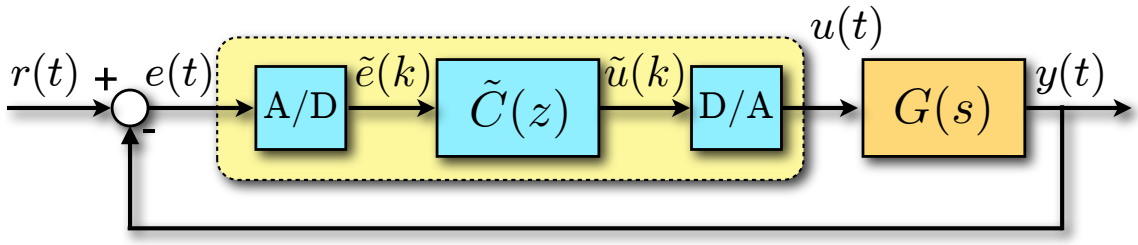
For comparison, the analog controller would give an overshoot of 1.6% at 3.5 seconds, so all the digital implementations produce some degree of performance degradation.

Time-discretization of continuous-time controllers

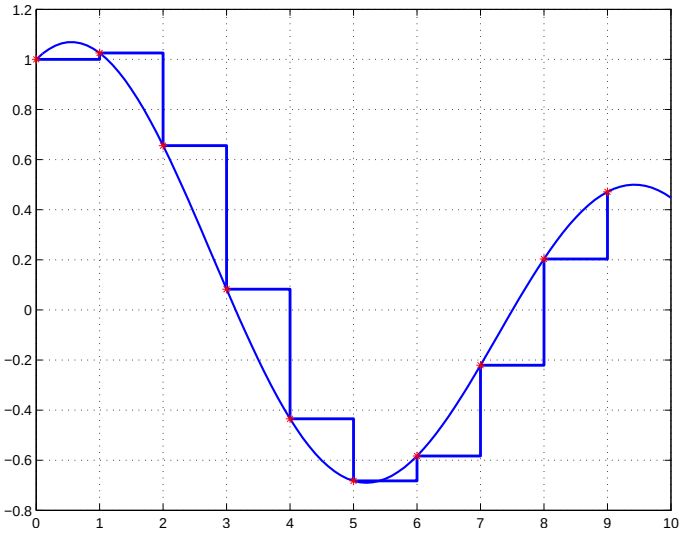


- We have designed an analog controller $C(s)$ in the *continuous-time* domain (by loop shaping, pole-placement, etc.)
- We want to implement $C(s)$ in digital form (for example, in a microcontroller). We need to convert the design to *discrete-time*

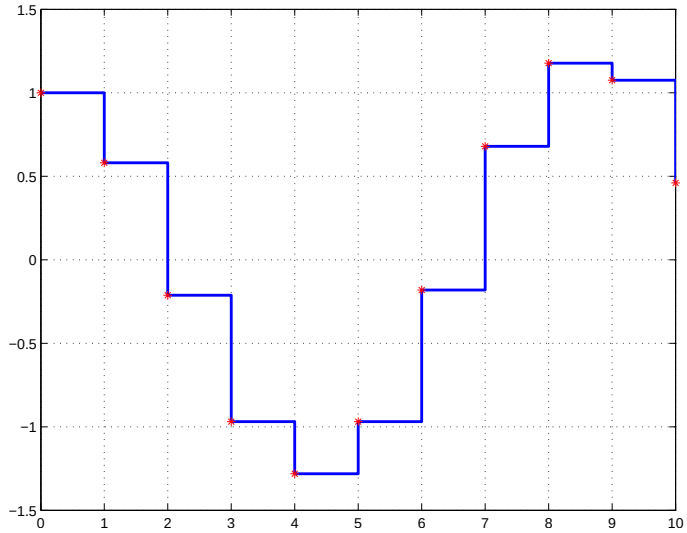
Relations between continuous/discrete-time signals



- Let the digital controller operate with sampling time T

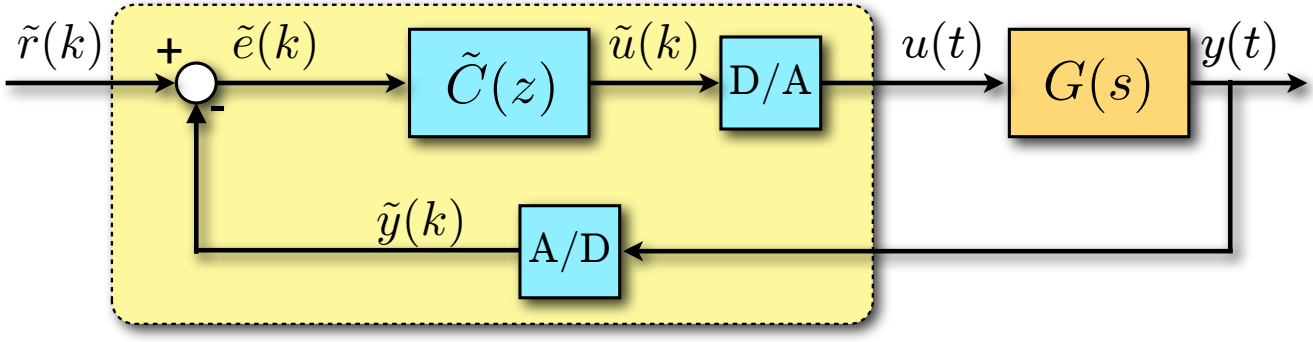


tracking error $\tilde{e}(k) \triangleq e(kT)$
for $k \in \{0, 1, \ldots\}$



control input $u(t) = \tilde{u}(k) \triangleq u(kT)$
for $t \in [kT, (k+1)T)$

Time-discretization of continuous-time controllers



- Alternative scheme: more often the reference signal $r(t)$ is already given in digital form by its samples $\tilde{r}(k)$, $\tilde{r}(k) = r(kT)$, $k = 0, 1, \ldots$, T=sampling time

Problem

How to synthesize a *discrete-time* control law $\tilde{C}(z)$ that in closed-loop behaves as the given *continuous-time* controller $C(s)$?

Time-discretization

- Consider the continuous-time controller $C(s)$

$$u(t) = C(s)e(t) = \frac{N(s)}{D(s)}e(t) = \frac{b_{n-1}s^{n-1} + \ldots + b_0}{s^n + a_{n-1}s^{n-1} + \ldots + a_0}e(t)$$

- We can look at the control law as at a differential equation linking $u(t)$ to $e(t)$

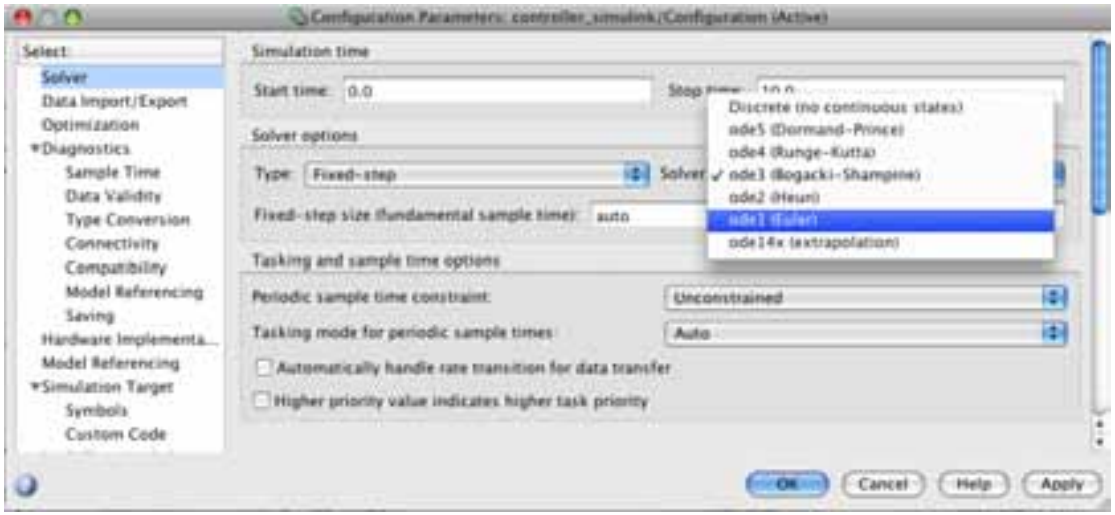
$$\frac{d^n}{dt^n}u(t) + a_{n-1}\frac{d^{n-1}}{dt^{n-1}}u(t) + \ldots + a_0 u(t) = b_{n-1}\frac{d^{n-1}}{dt^{n-1}}e(t) + \ldots + b_0 e(t)$$

- A *time-discretization* of $u(t) = C(s)e(t)$ is nothing else than a *numerical approximation* (with constant *integration-step* T)

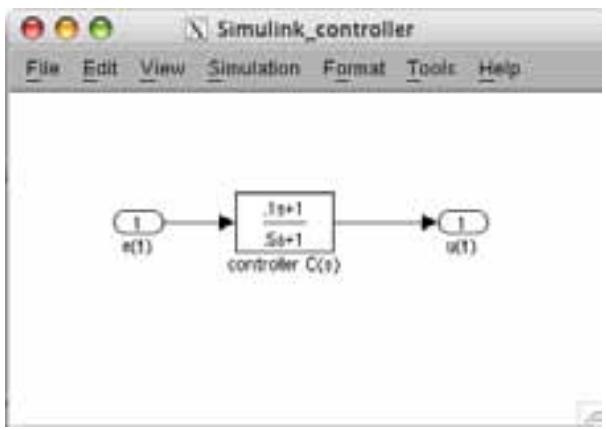
$$\tilde{u}(k) = \tilde{C}(z)\tilde{e}(k)$$

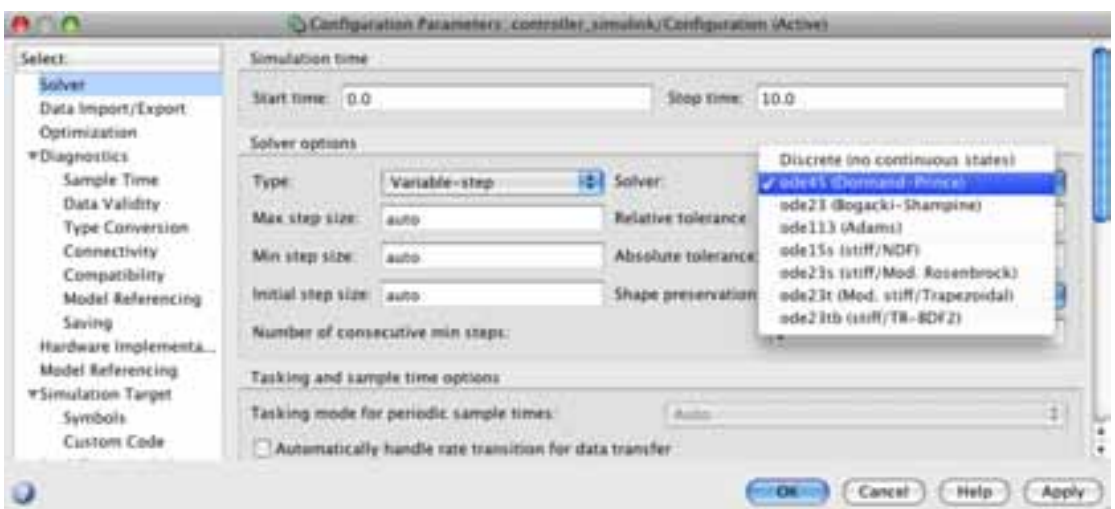
  linking the input samples $\tilde{u}(k) \triangleq u(kT)$ to the error samples $\tilde{e}(k) \triangleq e(kT)$
- There are many numerical methods to integrate a differential equation (constant step, variable step, n-th order approximations, etc.)

Numerical integration in Simulink



Fixed-step integration panel





Variable-step integration panel

Example of time-discretization

- Consider the continuous-time controller

$$u(t) = \frac{1}{s+2} e(t) \quad \Rightarrow \quad \frac{d}{dt} u(t) + 2u(t) = e(t)$$

- Use *Euler method* to approximate the derivative

$$\frac{d}{dt} u(t) \approx \frac{u((k+1)T) - u(kT)}{T} = \frac{1}{T}(\tilde{u}(k+1) - \tilde{u}(k))$$

- Recall that z represents the unit-shift operator $z\tilde{u}(k) = \tilde{u}(k+1)$

$$\tilde{u}(k) = \frac{1}{\left(\frac{z-1}{T}\right) + 2} \tilde{e}(k) \triangleq \tilde{C}(z)\tilde{e}(k)$$

- Note that formally this is equivalent to replace $s = \left(\frac{z-1}{T}\right)$ in $C(s)$ to get $\tilde{C}(z)$

Finite-difference approximation of the controller

- Consider a state-space realization of $C(s)$

$$\begin{cases} \frac{dx_c}{dt} &= A_c x_c + B_c e \\ u &= C_c x_c + D_c e \end{cases}$$

$$C(s) = C_c(sI - A_c)^{-1}B_c + D_c$$

- Integrate between time kT and $(k+1)T$

$$x_c((k+1)T) - x_c(kT) = \int_{kT}^{(k+1)T} \frac{dx_c(\tau)}{dt} d\tau$$

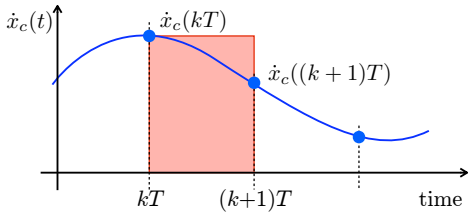
gives

$$x_c((k+1)T) - x_c(kT) = A_c \int_{kT}^{(k+1)T} x_c(\tau)d\tau + B_c \int_{kT}^{(k+1)T} e(\tau)d\tau$$

- Unfortunately, in general both $x(\tau)$ and $e(\tau)$ are not constant between consecutive sampling instants, so we can't use *exact* discretization (i.e., the exponential matrix $e^{A_c T}$)

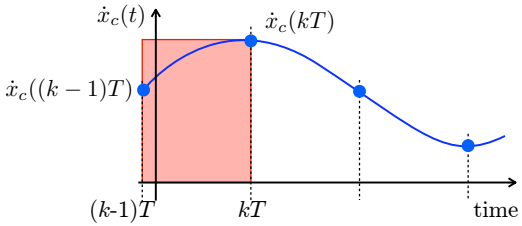
Approximation of the integral of $\dot{x}_c(\tau)$

- *(forward) Euler method*



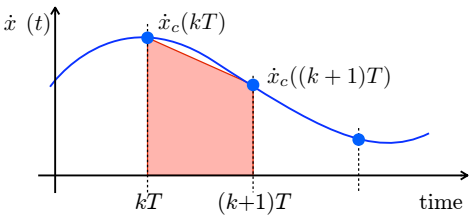
$$\underbrace{x_c((k+1)T) - x_c(kT)}_{(z-1)x_c(kT)} = \int_{kT}^{(k+1)T} \dot{x}_c(\tau)d\tau \approx T\underbrace{\dot{x}_c(kT)}_{Tsx_c(kT)} \quad \rightarrow \quad s = \frac{z-1}{T}$$

- *backward Euler method*



$$\underbrace{x_c(kT) - x_c((k-1)T)}_{(1-z^{-1})x_c(kT)} = \int_{(k-1)T}^{kT} \dot{x}_c(\tau)d\tau \approx \underbrace{T\dot{x}_c(kT)}_{Tsx_c(kT)} \quad \rightarrow \quad s = \frac{1-z^{-1}}{T}$$

- *Trapezoidal rule*



$$\underbrace{x_c((k+1)T) - x_c(kT)}_{(z-1)x_c(kT)} = \int_{kT}^{(k+1)T} \dot{x}_c(\tau)d\tau \approx \underbrace{\frac{T[\dot{x}_c((k+1)T) + \dot{x}_c(kT)]}{2}}_{\frac{T}{2}(z+1)sx_c(kT)}$$

$$\rightarrow \quad s = \frac{2(z-1)}{T(z+1)}$$

Finite-difference approximation of the controller

$$(z-1)x_c = T(A_c x_c + B_c e) \longrightarrow \left[\left(\frac{z-1}{T}\right) I - A_c\right] x_c = B_c e$$

forward Euler method

$$\tilde{C}(z) = \frac{U(z)}{E(z)} = C_c \left[\left(\frac{z-1}{T}\right) I - A_c\right]^{-1} B_c + D_c = C\left(\frac{z-1}{T}\right)$$

$$(1-z^{-1})x_c = T(A_c x_c + B_c e) \longrightarrow \left[\left(\frac{z-1}{zT}\right) I - A_c\right] x_c = B_c e$$

Backward Euler method

$$\tilde{C}(z) = \frac{U(z)}{E(z)} = C_c \left[\left(\frac{1-z^{-1}}{T}\right) I - A_c\right]^{-1} B_c + D_c = C\left(\frac{1-z^{-1}}{T}\right)$$

$$\tilde{C}(z) = C_c \left[\left(\frac{2(z-1)}{T(z+1)}\right) I - A_c\right]^{-1} B_c + D_c = C\left(\frac{2(z-1)}{T(z+1)}\right)$$

Tustin's method

Finite-difference approximation of the controller

- Formally, we just replace s in $C(s)$ with the corresponding function of z:
  - forward Euler method

$$s = \frac{z-1}{T} \quad \longrightarrow \quad \dot{x}_c(kT) \approx \frac{x_c((k+1)T) - x_c(kT)}{T}$$

  - backward Euler method

$$s = \frac{1-z^{-1}}{T} \quad \longrightarrow \quad \dot{x}_c(kT) \approx \frac{x_c(kT) - x_c((k-1)T)}{T}$$

  - Trapezoidal rule (*Tustin's method*)

$$s = \frac{2(z-1)}{T(z+1)} \quad \text{(bilinear transformation)}$$

- Note that all three methods preserve the DC gain: $z = 1 \rightarrow s = 0$
  (no approximation error exists in constant steady-state !)
- Compare to the *exact discretization method*: $\tilde{A}_c = e^{TA_c}$, $\tilde{B}_c = \int_0^T e^{tA_c} dt B_c$, $\tilde{C}_c = C_c$, $\tilde{D}_c = D_c$

Relations between poles in s and z

- What is the relation between the poles s_i of $C(s)$ and the poles z_i of $\tilde{C}(z)$?
  - forward Euler differences:

$$\frac{z_i - 1}{T} = s_i \quad \Rightarrow z_i = 1 + Ts_i$$

  - backward Euler differences:

$$\frac{z_i - 1}{Tz_i} = s_i \quad \Rightarrow z_i = \frac{1}{1 - s_i T}$$

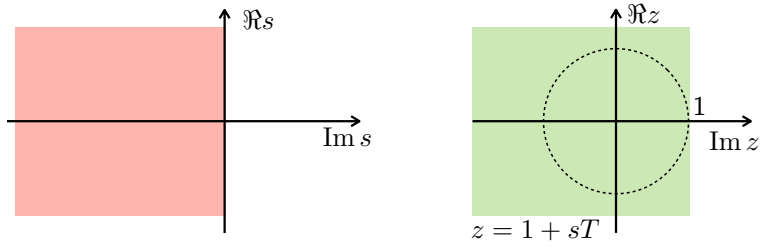
  - Tustin's method:

$$\frac{2(z_i - 1)}{T(z_i + 1)} = s_i \quad \Rightarrow z_i = \frac{1 + s_i T/2}{1 - s_i T/2}$$

  - Exact method: $z_i = e^{s_i T}$
- Note that the three (approximate) integration methods approximate the function $z = e^{sT}$ by a rational function

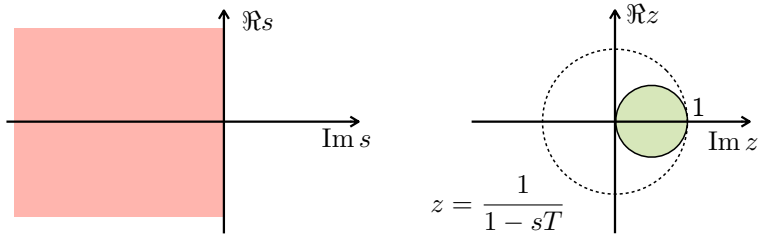
Relations between poles in s and z

forward Euler differences



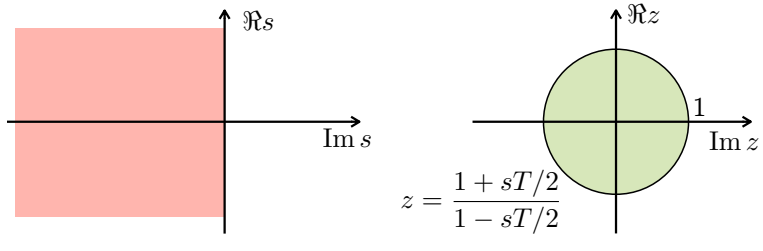
stable poles in s may be mapped to unstable poles in z

backward Euler differences



stable poles in s are also stable poles in z, marginally stable poles may become asymptotically stable

Tustin's rule



stable poles in s are also stable poles in z

Time-discretization in MATLAB

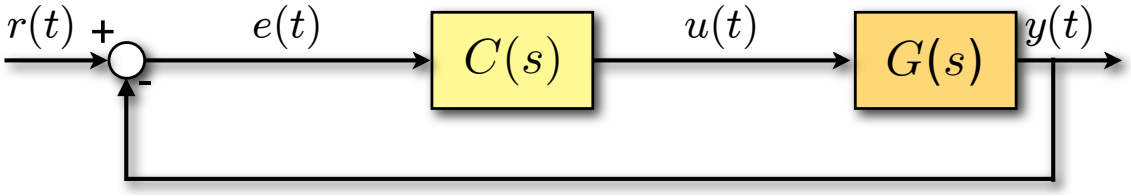
MATLAB

```
C2D Conversion of continuous-time models to discrete time.

    SYSD = C2D(SYSC,TS,METHOD) converts the continuous-time LTI
    model SYSC to a discrete-time model SYSD with sample time TS.
    The string METHOD selects the discretization method among the
    following:
       'zoh'        Zero-order hold on the inputs.
       'foh'        Linear interpolation of inputs (triangle appx.)
       'tustin'     Bilinear (Tustin) approximation.
       'prewarp'    Tustin approximation with frequency prewarping.
                    The critical frequency Wc is specified as fourth
                    input by C2D(SYSC,TS,'prewarp',Wc).
       'matched'    Matched pole-zero method (for SISO systems only).
    The default is 'zoh' when METHOD is omitted.
```

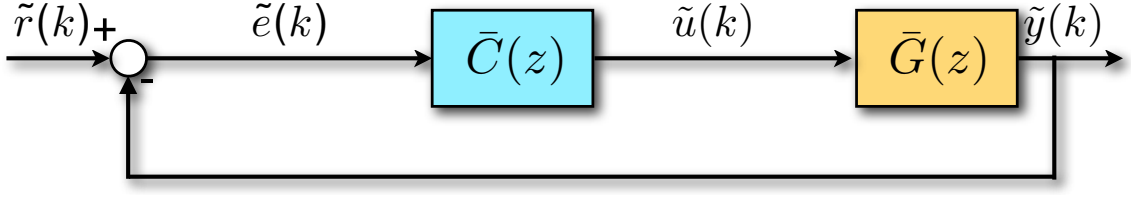
Sampled-data systems

- continuous-time control system (Laplace transform/frequency analysis)



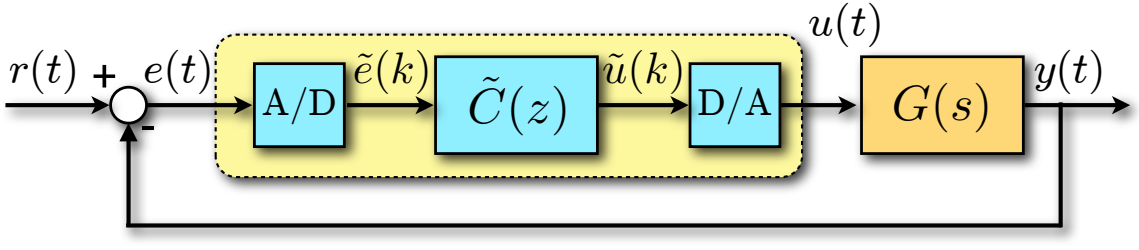
$t =$ continuous time

- discrete-time control system (Z-transform)



$k =$ discrete-time step counter

- *sampled-data system*



continuous-time and discrete-time signals

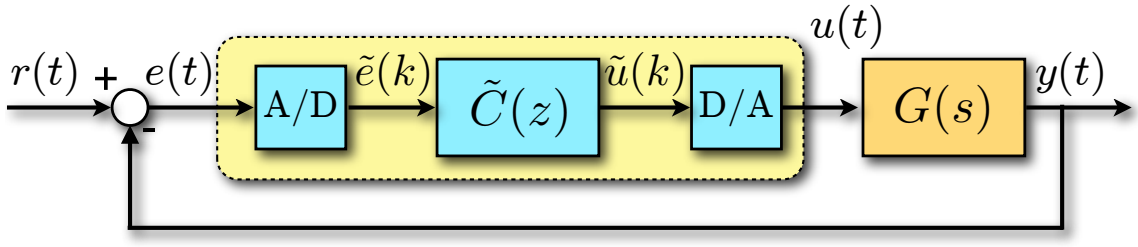
Why analyzing sampled-data systems

- The model $G(s)$ of the *process* to be controlled is (very often) given in continuous time
  - differential equations
  - actuator signals (electrical voltages to motors, etc.) vary continuously in time
  - output variables (temperature, pressure, position, etc.) vary continuously in time

- On the other hand, the *controller* is (almost always) implemented in digital form, $\tilde{C}(z)$:
  - cheaper to implement (computer code)
  - easier to reconfigure
  - can exploit time-sharing (multiple controllers on the same hardware)
  - much more versatile (arbitrary nonlinear control laws)

- Hence the need to analyze sampled-data systems, namely a continuous process in closed loop with a digital controller

Ways to analyze sampled-data systems



1. Convert the system $G(s)$ to its discrete-time equivalent $\bar{G}(z)$ (use for instance exact sampling), ignoring its inter-sampling behavior (controller's point of view or "*stroboscopic model*") $\Longrightarrow$ discrete-time analysis
2. Model the digital controller in continuous time

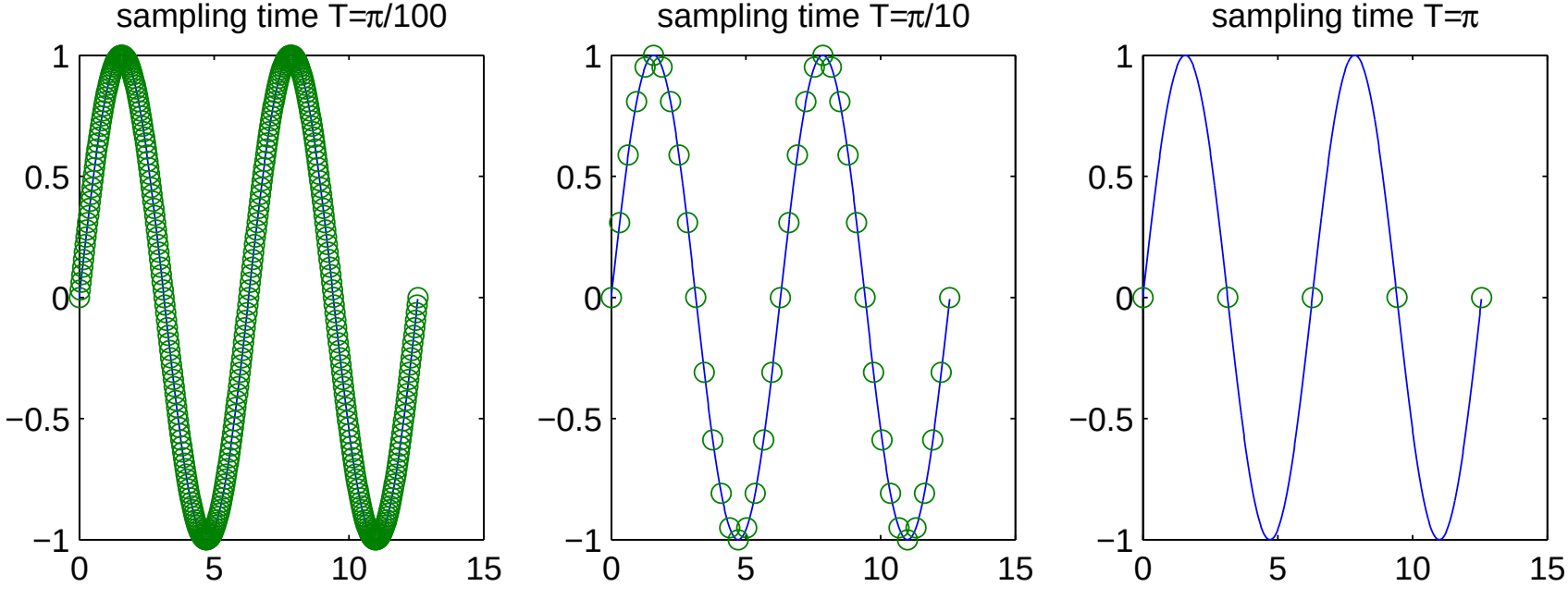
$$U(s) \simeq \frac{1-e^{-sT}}{sT}\tilde{C}(e^{sT})E(s)$$ (this can be shown ...)

(process' point of view) $\Longrightarrow$ continuous-time analysis

3. Use numerical simulation (e.g., Simulink) (only provides an answer for a certain finite set of initial states)

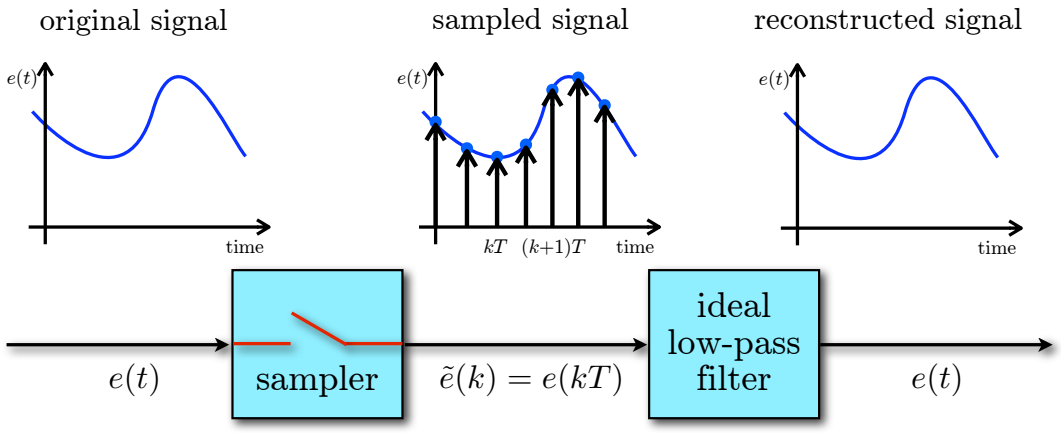
Choosing the sampling time

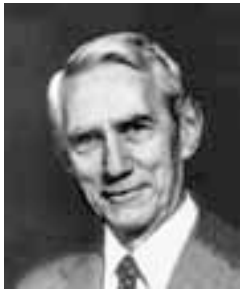
- How to choose the sampling time T ?
- Example: sampling of $\sin(t)$



- How can we say that a sampling time is “good” ?

Nyquist-Shannon sampling theorem





Claude Elwood Shannon

(1916–2001)

Sampling theorem

Let $e(t)$ be a signal and $E(j\omega) = \int_{-\infty}^{+\infty} e(\tau)e^{-j\omega\tau}d\tau$ its *Fourier transform* $\mathscr{F}[e]$.

Let $E(j\omega) = 0$ for $|\omega| \geq \omega_{\max}$. For all T such that $\omega_N \triangleq \frac{\pi}{T} > \omega_{\max}$

$$e(t) = \sum_{k=-\infty}^{+\infty} e(kT)\frac{\sin(\omega_N(t-kT))}{\omega_N(t-kT)}$$

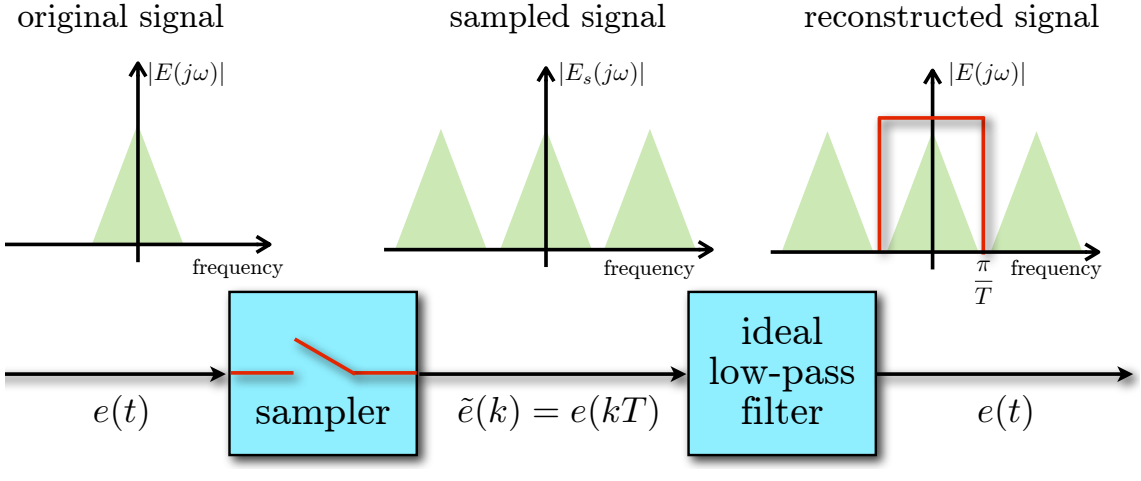
ω_N is called the *Nyquist frequency*, equal to half the sampling frequency $\omega_s = \frac{2\pi}{T}$

Shannon's reconstruction

- We can look at sampling as at the modulation of a *Dirac comb* $\sum_{k=-\infty}^{\infty} \delta(t-kT)$
- Let's go to Fourier transforms

$$\underbrace{\mathscr{F}\left[\sum_{k=-\infty}^{\infty} e(kT)\delta(t-kT)\right]}_{\text{modulated Dirac comb}} = \sum_{k=-\infty}^{\infty} e(kT)e^{-jkT\omega} = \frac{1}{T}\sum_{k=-\infty}^{\infty} E(j(\omega+k\omega_s)) \triangleq \frac{1}{T}E_s(j\omega)$$

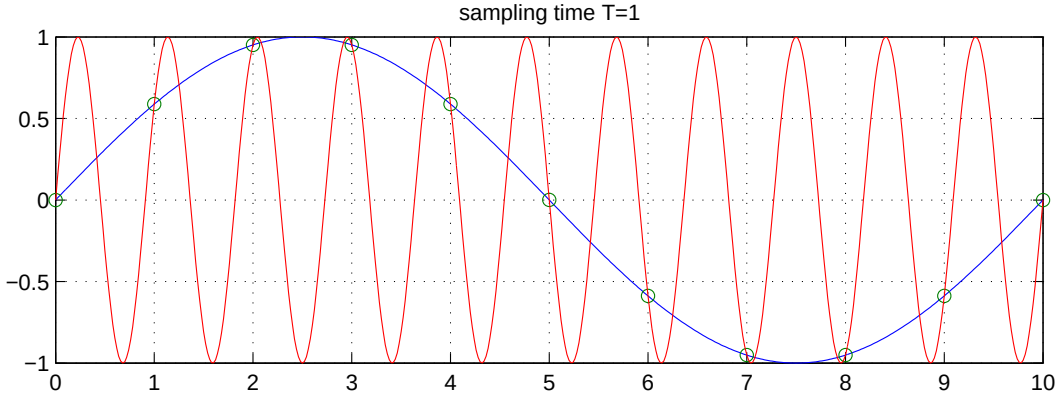
- $E_s(j\omega)$ is the periodic repetition of $E(j\omega)$ with period ω_s
- The ideal low-pass filter can only reconstruct $e(t)$ if $E(j\omega) = 0$ for $|\omega| \geq \omega_{\max}$



Aliasing

$$E_s(j\omega) = \frac{1}{T} \sum_{k=-\infty}^{+\infty} E(j(\omega + k\omega_s))$$

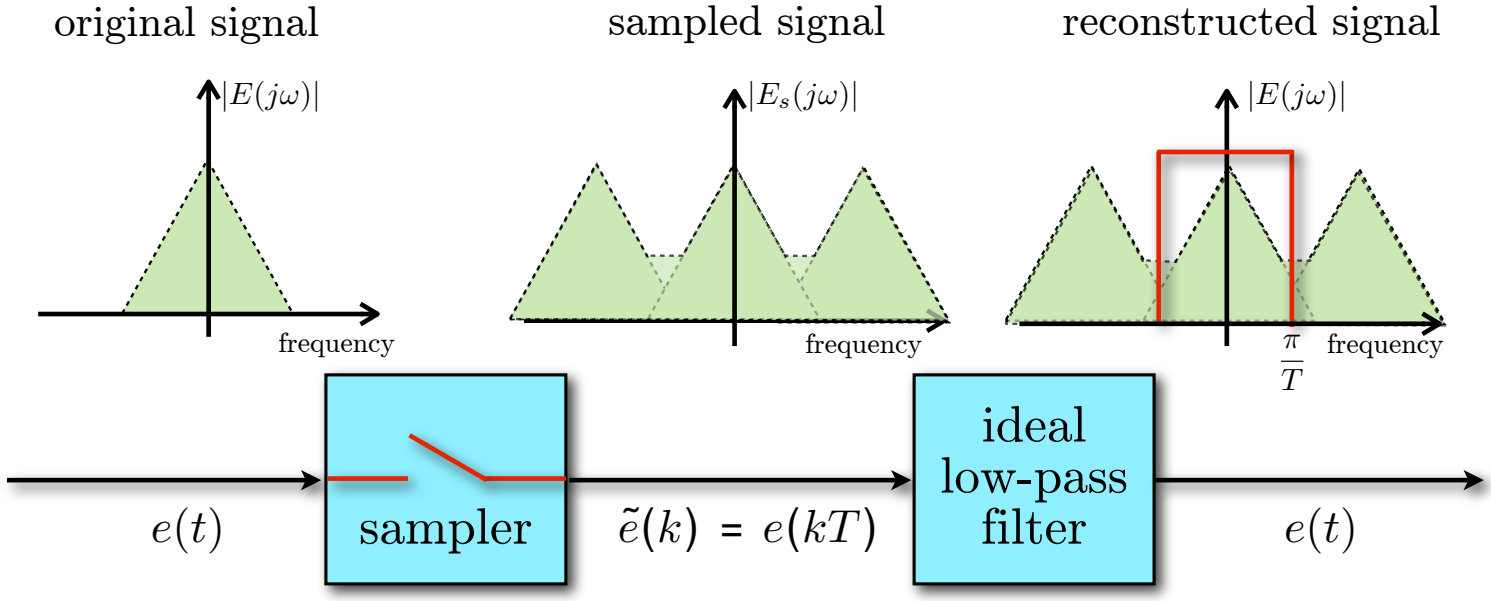
- The value $E_s(j\omega_1)$ at a certain frequency ω_1 not only depends on $E(j\omega_1)$, but also on all the (infinite) contributions $E(j(\omega_1 \pm k\omega_s))$
- The frequency $\omega_1 \pm k\omega_s$ is called an *alias* of ω_1
- Example: consider the signals $\sin(\omega_1 t)$ and $\sin(\omega_2 t)$ and let the sampling time $T = 1$ s



$$\omega_1 = \frac{2\pi}{10}$$
$$\omega_2 = \frac{2\pi}{T} + \frac{2\pi}{10} = \frac{22\pi}{10}$$

Aliasing

- What if $e(t)$ has spectral contributions $E(j\omega) \neq 0$ for $\omega > \omega_N = \frac{\pi}{T}$?



- *Aliasing* is the phenomenon for which frequencies $\omega > \omega_N$ make contributions in the frequency range $[-\omega_N, \omega_N]$
- Under aliasing conditions it is impossible to reconstruct the original signal
- An *anti-aliasing filter* is a low-pass filter that removes from $e(t)$ its high-frequency contents before sampling it. It partially mitigates the problem

Selecting the sampling time (period)

- In *signal processing* one is interested in making the difference between the original and the reconstructed signal as small as possible (=high fidelity)
- In *control systems* one is interested that the closed-loop system behaves according to specs, not much in carefully reconstructing $e(t) = y(t) - r(t)$!
- For control purposes, the sampling time is mainly related to the closed-loop bandwidth / settling-time
- The sampling time used in control is typically larger than the one used signal processing
- That's why in control applications we are often ok with micro-controllers and don't need DSPs (digital signal processors)

Selecting the sampling time

- Let ω_c be the desired bandwidth of the closed-loop system
- To avoid aliasing effects (and satisfy the sampling theorem) we must set

$$\frac{\pi}{T} > \omega_c$$

- A good choice is

$$5\omega_c \leq \frac{2\pi}{T} \leq 100\omega_c$$

  so that the (non-ideal) anti-aliasing filter has cutoff frequency ω_f in between ω_c and ω_N
- By recalling the approximate relation $t_s\omega_c \approx \frac{5}{\zeta}$, where t_s is the settling time (1%) and ζ is the damping factor, we get

$$\frac{t_s}{100} \leq T \leq \frac{t_s}{5}$$

- A related good choice is also to let $T = \frac{t_r}{10}$, where t_r is the rise time of the open-loop system

Numerical errors

- Recall the relation between poles s_i and z_i

$$z_i = e^{s_i T}$$

- For $T \to 0$, we have $z_i \to 1$ whatever the poles s_i are !
- This can make troubles when working in finite precision (both in control design and in controller implementation)
- Consider the following example: we have two poles $s_1 = -1$ and $s_2 = -10$
  - Sample with $T = 1\ ms$ and get $z_1 = e^{-0.001} \approx 0.9990$, $z_2 = e^{-0.01} \approx 0.9900$. If we truncate after two digits, we get $z_1 = z_2$, and then $s_1 = \frac{1}{T} \ln z_1 = \frac{1}{T} \ln z_2 = s_2 \approx -10.05$
  - Sample with $T = 100\ ms$ and truncate, we get $z_1 = 0.90$, $z_2 = 0.36$, and then $s_1 \approx -1.05$, $s_2 \approx -10.21$

Final remarks on choice of sampling time

- Make the sampling time T small enough to reproduce the open-loop time response enough precisely ($T = \frac{t_r}{10}$), and to avoid aliasing effects (Nyquist frequency $\frac{\pi}{T}$ larger than closed-loop bandwidth)
- Make the sampling time T small enough to react enough readily to disturbances affecting the system
- Make the sampling time large enough to avoid numerical issues
- Make the sampling time large enough to avoid fast and expensive control hardware

Digital controllers

- A digital control system arrangement for one of the above transfer functions in z.
- First step: convert the transfer function into a discrete time (difference) equation from which an algorithm or pseudo-code may be developed.
- $U(z)$ = controller output, $E(z)$ = controller input = error signal

$$D(z) = \frac{U(z)}{E(z)} = \frac{1.2692 - 1.1538z^{-1}}{1 - 0.7692z^{-1}}$$

$$(1 - 0.7692z^{-1})U(z) = (1.2692 - 1.1538z^{-1})E(z)$$

Dividing by z^{-1} = delay by T:

$$u(k) - 0.7692u(k-1) = 1.2692e(k) - 1.1538e(k-1)$$

or

$$u(k) = 0.7692u(k-1) + 1.2692e(k) - 1.1538e(k-1)$$

The outline algorithm:

Initialize: set $u(k-1) = 0$, $e(k-1) = 0$ or to their actual values if they are available
Loop: Reset sampling interval timer
Input e ($=e(k)$)
Calculate $u(k)$
Output $u(k)$
Set $u(k-1) = u(k)$
Set $e(k-1) = e(k)$
Wait for end of sampling interval
End Loop

Digital PID control

Any of the time-discretization we have seen so far can be used to convert a PID design in digital form. The following is most used:

- proportional part

$$P(t) = K_p(br(t) - y(t))$$ static relation, no need to approximate it !

- integral part

$$I(t) = \frac{K_p}{T_i} \int_0^t e(\tau)\, d\tau$$

is approximated by forward Euler as

$$I((k+1)T) = I(kT) + \frac{K_p T}{T_i} e(kT)$$

Digital PID

- derivative part:

$$\frac{T_d}{N}\frac{dD(t)}{dt} + D(t) = -K_p T_d \frac{dy(t)}{dt}$$

  is approximated by backward Euler as

$$D(kT) = \frac{T_d}{T_d + NT} D((k-1)T) - \frac{K_p T_d N}{T_d + NT}\left(y(kT) - y((k-1)T)\right)$$

  Note that the discrete pole $z = \frac{T_d}{T_d + NT}$ is inside the unit circle

- The complete control signal is

$$u(kT) = P(kT) + I(kT) + D(kT)$$

This type of approximation allows one to calculate P, I, and D actions separately

Digital PID in incremental form

- The digital PID form we described provides the full signal $u(kT)$ (*positional algorithm*)
- An alternative form is the so called *incremental form* (*velocity algorithm*), where the PID controller computes instead the input increments

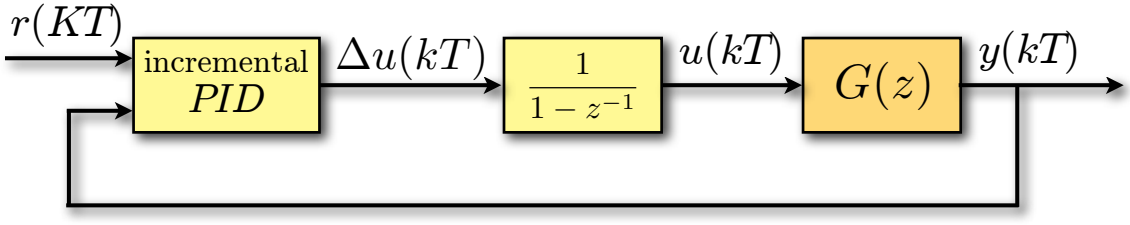
$$\Delta u(kT) = u(kT) - u((k-1)T) = \Delta P(kT) + \Delta I(kT) + \Delta D(kT)$$

where

$$\begin{aligned}
\Delta P(kT) &= K_p(br(kT) - br((k-1)T) - y(kT) + y((k-1)T) \\
\Delta I(kT) &= \alpha_1(r(kT) - y(kT)) + \alpha_2(r((k-1)T) - y((k-1)T)) \\
\Delta D(kT) &= \beta_1 \Delta D((k-1)T) - \beta_2(y(kT) - 2y((k-1)T) + y((k-2)T))
\end{aligned}$$

- Advantage: increased numerical precision in the presence of finite word-length (signal *quantization*)
- This form cannot be used for P and PD controllers

Digital PID in incremental form



- An integrator is needed to reconstruct the input signal $u(kT)$ from the incremental PID form

$$u(kT) = u((k-1)T) + \Delta u(kT) \quad \longrightarrow \quad u(kT) = \frac{1}{1-z^{-1}}\Delta u(kT)$$

Example. Implementation of Digital PID Controllers - another approach. The s-domain transfer function:

$$\frac{U(s)}{X(s)} = G_C(s) = k_1 + \frac{k_2}{s} + k_3 s$$

A digital implementation of this controller: using a discrete time approximation for the derivative and integration. For the time derivative use the *backward difference rule*

$$u(kT) = \frac{dx}{dt}|_{t=kT} = \frac{1}{T}\left[x(kT) - x((k-1)T)\right]$$

The z-transform of the equation is:

$$U(z) = \frac{1 - z^{-1}}{T} X(z) = \frac{z-1}{Tz} X(z)$$

The integration of x(t) can be represented by the *forward rectangular integration* at $t = kT$ as:

$$u(kT) = u((k-1)T) + Tx(kT)$$

where $u(kT)$ is the output of the integrator at t=kT.
The z-transform of the equation is:

$$U(z) = z^{-1}U(z) + TX(z)$$

and the transfer function is then:

$$\frac{U(z)}{X(z)} = \frac{Tz}{z-1}$$

The z-domain transfer function of the PID controller is:

$$G_C(z) = k_1 + \frac{k_2 Tz}{z-1} + k_3 \frac{z-1}{Tz}$$

The complete difference equation algorithm (where $x(kT) = x(k)$):

$$u(k) = k_1 x(k) + k_2[u(k-1) + Tx(k)] + \frac{k_3}{T}[x(k) - x(k-1)]$$

$$u(k) = [k_1 + k_2 T + \frac{k_3}{T}]x(k) + k_3 Tx(k-1) + k_2 u(k-1)$$

We can obtain a PI or a PD controller by setting an appropriate gain equal to zero (k_2 or k_3).

Thank you very much for your attention!