Digital Control (SC42095)

Lecture 1, November 13, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Information about the course.
- Introduction to computer controlled systems.
- Sampling of continuous-time signals.

Digital Control (SC42095)

Lecturer: Tamás Keviczky (Mekelweg 2, C-3-310)

- Two lectures per week:
  - Monday 10:45 – 12:30, CT-CZ C, DUWO-CZ, 3mE-CZ C
  - Wednesday 13:45 – 15:30, EWI-Pi
- Office hour: Mondays 13:00 – 14:00 (ask for appointment)
- Last class: December 20, 2017
- Final quiz: January 10, 2018
  - Wednesday 13:45 – 15:30, 2C-Zaal 1
- Final grade based on project assignment + quiz
  no final exam!

Goals of the course

Main subject:
computer-controlled systems, digital implementation

- Analyze discrete-time systems (analysis).
- Design sampled-data controllers (synthesis).
- Identify implementation issues.
- Apply the design methodology to a simulated process
  (project assignment with Matlab and Simulink).

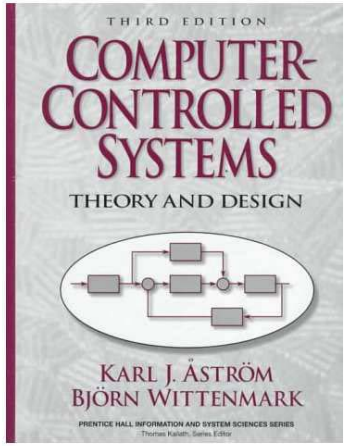
Project assignment

- Purpose: learn to apply the design and analysis techniques and the use of numerical tools
- Procedure:
  - download an assignment from Brightspace,
  - notify the lecturer for confirmation,
  - start with the continuous-time control task,
  - gradually solve the discrete-time problems,
  - write a short report (deadline 13:00, January 10, 2018)
- Your report will be graded and forms the basis of the final mark (80% of the total grade)

Contents of lectures

- Introduction to computer controlled systems
- Sampling of signals and systems (aliasing, discrete-time models, z-transform, zeros and poles)
- Discrete-time state-space and input-output models
- Analysis of discrete-time systems (stability, controllability)
- Design methods
  (state-space, digital PID, linear quadratic, repetitive)
- Observers, stochastic models
  (Luenberger observer, Kalman-filter)

Course material



Book:
Aström K.J. and Wittenmark B.:
Computer Controlled Systems
3rd ed., Prentice Hall, 1997.

Transparencies:
available through Brightspace

MATLAB/Simulink software
available through Brightspace

Supplementary textbook

Book:
L. Keviczky et. al:
Control Engineering
Széchenyi University Press, 2011.

Selected chapters on the subject of sampled data control and background material for project assignment (e.g. PID design).

Available for purchase at Leeghwater.

Prerequisites

- Mathematical analysis
- Linear algebra
- Dynamical control systems, state-space theory (SC42015)
- Working knowledge of Matlab and Simulink

Some important concepts to refresh

- Differential equations, Laplace transform.
- Transfer functions, state-space models.
- Poles and zeros, stability, root-locus, ...
- PID control, system type

Matlab and Simulink

- Elementary Matlab commands (`plot`, `load`, `save`, etc.)
- Control System Toolbox:
  - LTI class (`ss`, `tf`, `zpk`)
  - time-domain and frequency analysis (`step`, `bode`)
  - control design tools (`rlocus`, `place`)
- Simulink (implementing models, simulation methods)

Final Grade

- 80% for the project + 20% for the quiz (no resit!)
- Quiz (closed book, closed notes, no registration needed):
  - about 20 multiple choice questions in one hour
  - testing essential concepts and definitions
- Relevant material: contents of the lectures, handouts and selected parts of the Aström book (download quiz demands).

Important Note

- Some parts of the book will not be discussed in the lectures and are left for self-study (we will not cover all chapters).
- Some material will be discussed in more detail than in the slides using the blackboard, so make sure you attend the lectures.

Course information on the Web

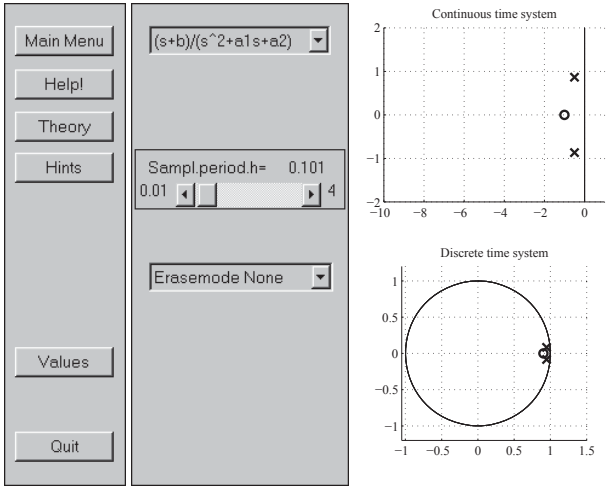
Brightspace

- Basic course information.
- Important dates, messages.
- Solutions manual, errata for the book.
- Exercises, mailing list, etc.
- Course software (for Matlab and Simulink):
  - Matlab scripts to generate figures in the book: (see how examples are implemented, etc.)
  - CCSDEMO: interactive tool

Exercises in Brightspace

- Multiple-choice questions.
- Several tests will be available.
- Graded, but no influence on the final mark.
- Should be useful as preparation for the quiz, but it is <u>not</u> a complete set of possible exam questions.

CCS – example

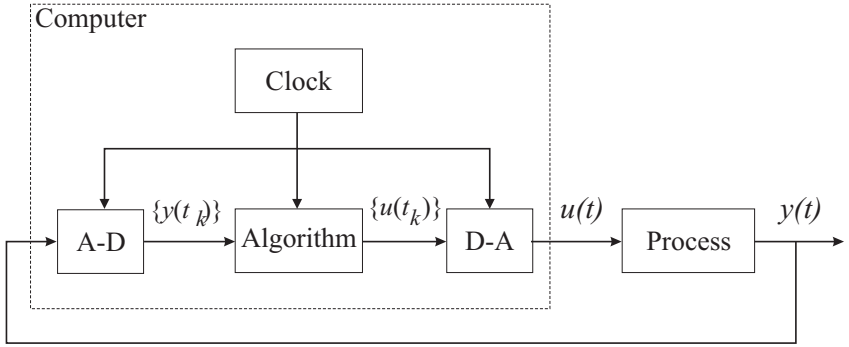


Sampling of

$$\frac{s+b}{s^2+a_1 s+a_2}$$

for different sampling periods

Computer-controlled systems



both continuous-time and sampled (discrete-time) signals:
(sampled-data systems) $\rightarrow$ may give some difficulties

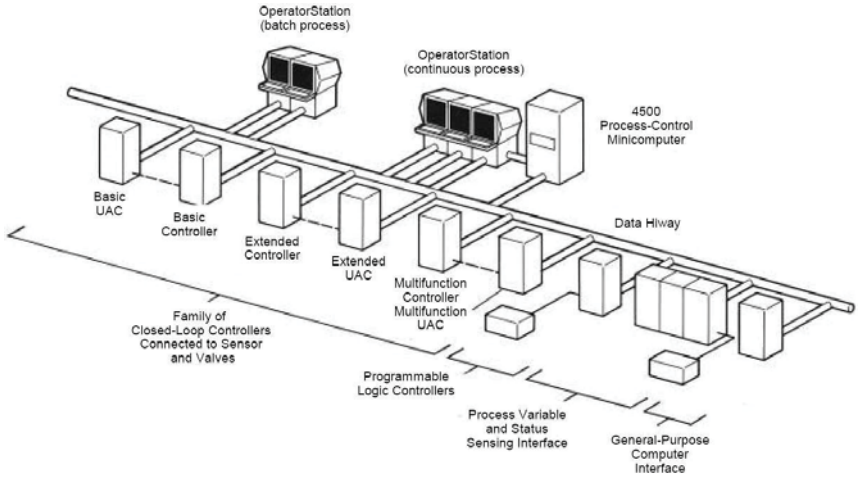
Computer-controlled systems

- Common approach: describe process at the sampling instants only, i.e., disregard intersample behavior (discrete-time systems).
- Some systems are inherently sampled, e.g.
  - due to measurement procedure, such as radars, economic systems
  - due to pulsed operation, such as thyristors, combustion engines, biological systems

Development of computer controlled systems

- Pioneering period $\approx$ 1955 (process industry: Texaco, very unreliable with CPU MTBF 50-100h, operator guide, set-point control).
- Direct digital control $\approx$ 1962 (able to control more than 100 variables, still often more expensive than analog control, MTBF 1000h).
- Minicomputers $\approx$ 1967 (16 bits, 124k memory, price still \$10.000 – \$100.000, MTBF 20000h).
- Microcomputers, general use of digital control $\approx$ 1972-80 (\$500, VLSI, standard single-loop controllers, PLCs, microcontrollers).
- Distributed control $\approx$ 1990 (plant-wide control, integration of control, optimization and planning).

Distributed control system (DCS) origins



Honeywell TDC 2000

Distributed control system (DCS) origins

- Honeywell TDC 2000: Development started in 1969, system announced in 1975.
- Several "firsts", and not just for process automation:
  - the first 16-bit microprocessor in a commercial product
  - a "local area network" before the term was known
  - CRT-based operator consoles, a precursor to the GUI
- Basic TDC 2000 controller is still running many refineries worldwide (e.g., Shell Pernis).

Development of theory

- Sampling Theorem (1949)
- Difference Equations (1948)
- Numerical Analysis
- Transform Methods (1947)
- State-space Theory (1958)
- Optimal Control (1957)
- Systems Theory (1969)
- System Identification (1971)
- Adaptive Control (1973)
- Automatic Tuning (1995)

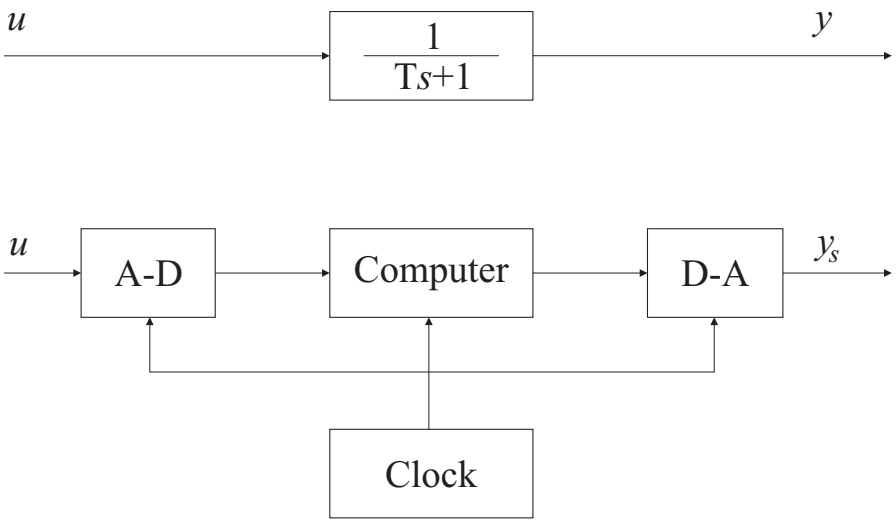
Advantages of computer control

- cheap hardware, no drifting, no ageing
- extra control performance (example later on)
- include models, nonlinearities
- signal processing, adaptation ('learning')
- complex calculations, logical functions, sequencing, flexibility
- integrated (plant-wide) control, user interface, data storage

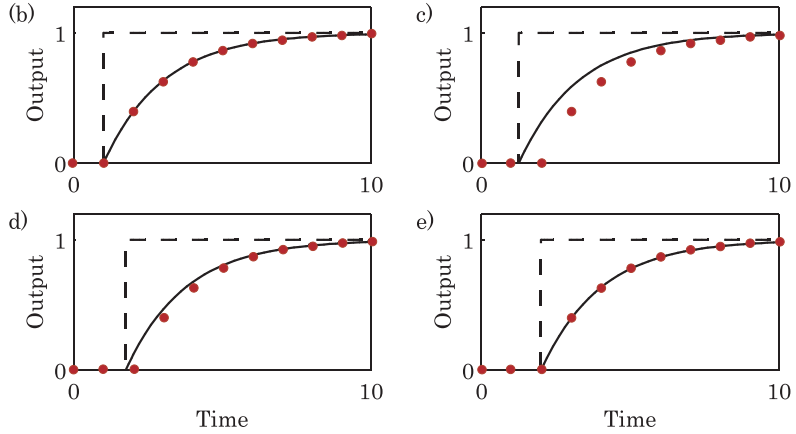
Limitations of computer control

- limited performance of computers (speed)
  - real time aspects
  - large-scale systems, networks
- robustness, safety (faults)
  - maintenance (e.g., cars)
  - safety-critical applications (e.g., aircraft, reactors)
- sampling $\rightarrow$ new phenomena (extra caution)
  - time-dependence
  - influence of sampling period
  - sampling of high-frequency signals (noise)

Time-varying behavior



Time-varying behavior – cont'd



dashed line: input step (u), solid line: response of continuous filter (y), dotted line: response of digital filter (y_s)

Influence of sampling period

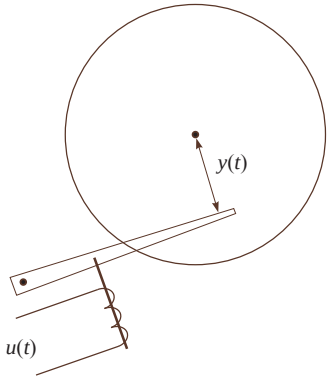
A naive approach:

1. Design continuous-time controller.
2. Discretize this controller.
3. Implement on a computer.

Influence of sampling period – cont'd

Example: control of a disk drive arm (double integrator):

$$G(s) = \frac{Y(s)}{U(s)} = \frac{k}{Js^2}$$



1. Design continuous-time controller

$$U(s) = \frac{bK}{a}U_c(s) - K\frac{s+b}{s+a}Y(s) \qquad (1)$$

with

$$\begin{aligned} a &= 2\omega_0 \\ b &= \omega_0/2 \\ K &= 2J\omega_0^2/k \end{aligned}$$

a closed loop characteristic polynomial is obtained

$$P(s) = s^3 + 2\omega_0 s^2 + 2\omega_0^2 s + \omega_0^3$$

1. Design continuous-time controller – cont'd

From (1) after elementary manipulations:

$$U(s) = K\left(\frac{b}{a}U_c(s) - Y(s) + \frac{a-b}{s+a}Y(s)\right)$$

$$\boxed{\begin{aligned} u(t) &= K\left(\frac{b}{a}u_c(t) - y(t) + x(t)\right) \\ \frac{dx(t)}{dt} &= -ax(t) + (a-b)y(t) \end{aligned}}$$

2. Discretize continuous-time controller

$$\begin{aligned} u(t) &= K\left(\frac{b}{a}u_c(t) - y(t) + x(t)\right) \\ \frac{dx(t)}{dt} &= -ax(t) + (a-b)y(t) \end{aligned}$$

Approximate the derivative:

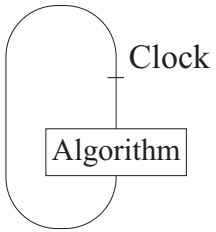
$$\frac{x(t+h) - x(t)}{h} = -ax(t) + (a-b)y(t)$$

Discrete-time controller:

$$\boxed{\begin{aligned} u(t_k) &= K\left(\frac{b}{a}u_c(t_k) - y(t_k) + x(t_k)\right) \\ x(t_k+h) &= x(t_k) + h\Big[(a-b)y(t_k) - ax(t_k)\Big] \end{aligned}}$$

3. Implement discrete-time controller

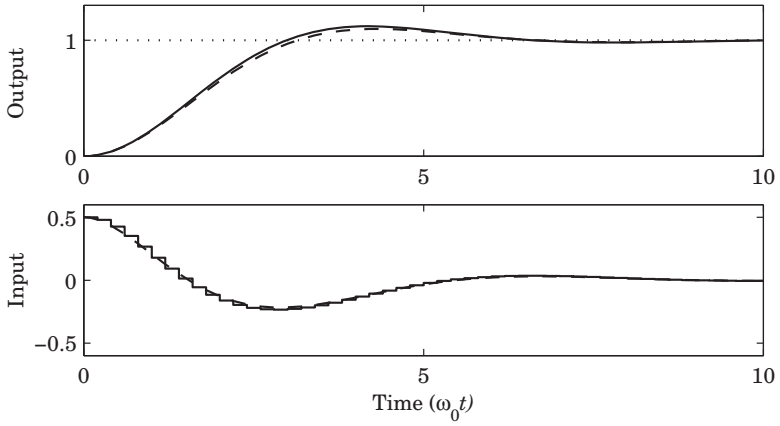
$$\begin{aligned} u(t_k) &= K\left(\frac{b}{a}u_c(t_k) - y(t_k) + x(t_k)\right) \\ x(t_k+h) &= x(t_k) + h\Big[(a-b)y(t_k) - ax(t_k)\Big] \end{aligned}$$




```
y:= adin                    {read process value}
u:= K*(b/a*uc-y+x)
dout(u)                     {output control signal}
x:= x+h*((a-b)*y - a*x)
```

Sampling period $h = 0.2/\omega_0$

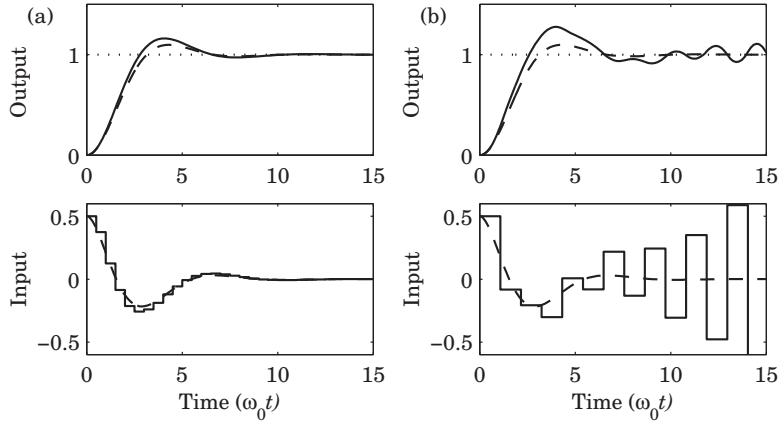
Performance Comparison



solid line: digital controller, dashed line: analog controller

Longer sampling period

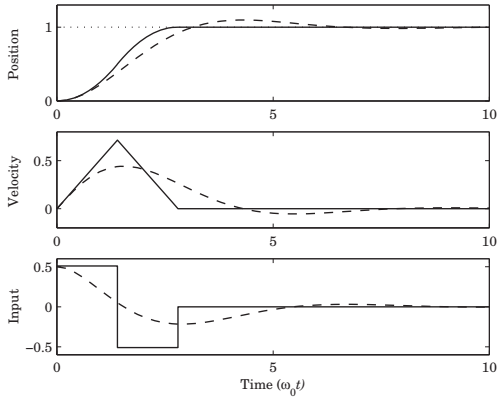
a) $h = 0.5/\omega_0$, b) $h = 1.08/\omega_0$



solid line: digital, dashed line: analog

Digital controllers can do better

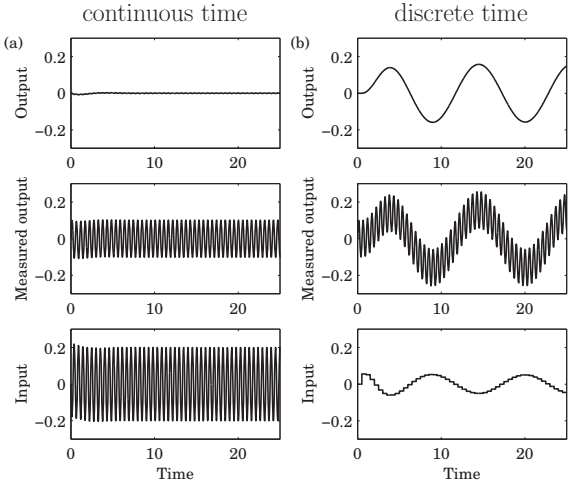
$u(t_k) = t_0 u_c(t_k) + t_1 u_c(t_{k-1}) - s_0 y(t_k) - s_1 y(t_{k-1}) - r_1 u(t_{k-1})$
$h = 1.4/\omega_0$ (!)



solid line: digital, dashed line: analog

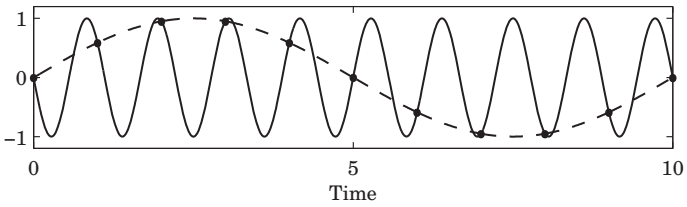
Sampling of measurement noise

Double integrator, $h = 0.5$, measurement noise $0.1\sin(12t)$



Sampling creates signals with new frequencies!

Aliasing



Sampling of two or more harmonic signals with *different* frequencies may result in *the same* discrete time signal. This is called aliasing.

$$\omega_{sampled} = |\omega \pm n\omega_s|, \quad n \text{ integer}$$

with

ω ... frequency of continuous-time signal

$\omega_s = 2\pi/h$ ... sampling frequency

Summary

- Advantages and drawbacks of computer control.
- Short sampling interval may be OK, but we can do better by using discrete-time design.
- Alias frequencies

$$\omega_{sampled} = |\omega \pm n\omega_s|$$

Digital Control (SC42095)

Lecture 2, November 15, 2017

Tamás Keviczky

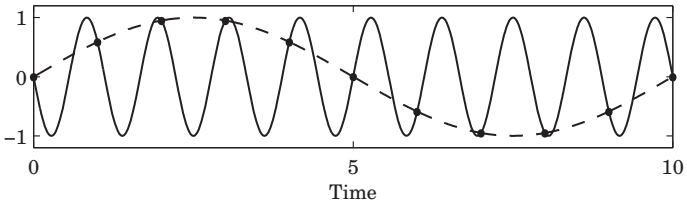
Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Shannon's sampling theorem, frequency folding.
- Sampling of continuous-time systems.
- State-space models.
- Input-output models.
- Shift-operator calculus.
- Pulse-transfer function.
- Poles and zeros.

Aliasing (recap)



Sampling of two or more harmonic signals with *different* frequencies may result in *the same* discrete time signal. This is called aliasing.

$$\omega_{sampled} = |\omega \pm n\omega_s|, \quad n \text{ integer}$$

with

ω ... frequency of continuous-time signal

$\omega_s = 2\pi/h$ ... sampling frequency

Aliasing – cont'd

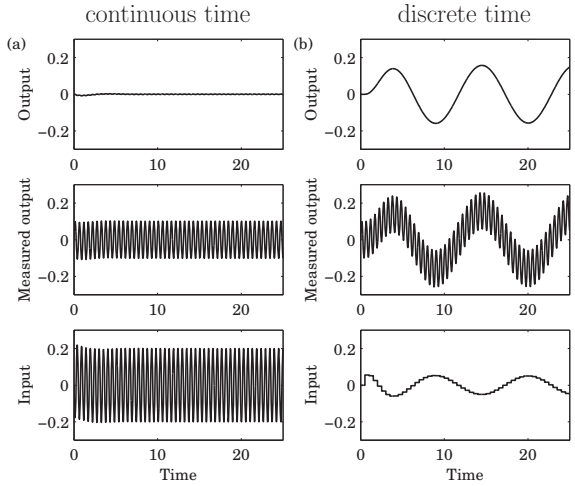
Proof:

$$e^{j2\pi(f+n/h)t} = e^{j2\pi ft} \cdot e^{j2\pi nt/h} = e^{j2\pi ft}$$

as nt/h is an integer for $t \in \{0, h, 2h, 3h, \ldots\}$.

Sampling of measurement noise

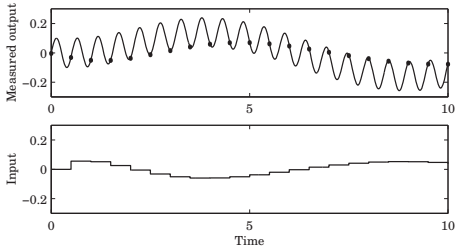
Double integrator, $h = 0.5$, measurement noise $0.1\sin(12t)$



Sampling creates signals with new frequencies!

Measurement noise - cont'd

Zoom in:



$$\omega_{sampled} = |\omega \pm n\omega_s|$$

$\omega = 12$ rad/s

$\omega_s = 2\pi/h = 4\pi = 12.57$ rad/s

$\omega_{sampled} = 0.57$ rad/s $\longrightarrow$ period of 11 s

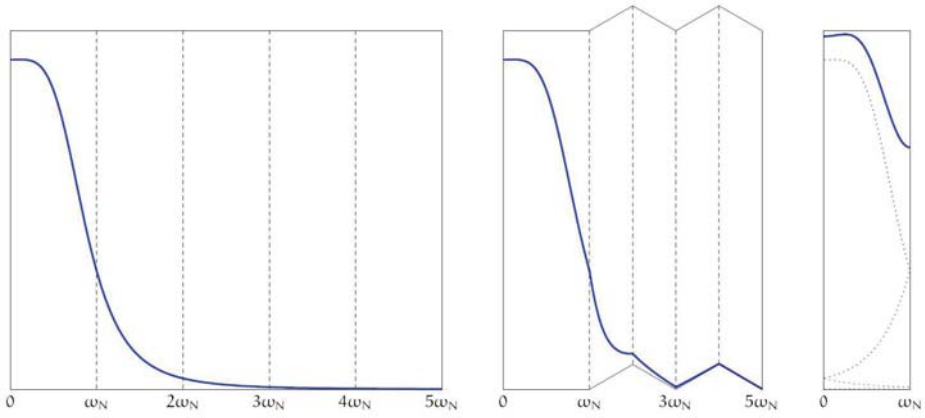
The sampling period is too long with respect to the noise.
It is important to filter before sampling!

Shannon's sampling theorem

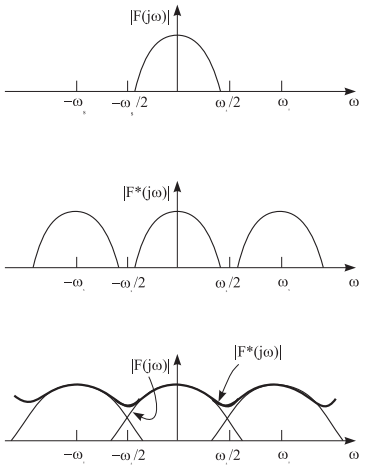
A continuous-time signal which contains no frequency components greater than ω_0 is uniquely determined by its values in equidistant points if the sampling frequency ω_s is higher than $2\omega_0$.

$\omega_N = \omega_s/2$ is called the *Nyquist frequency*.

Frequency Folding



Frequency Folding



$$|F^*(j\omega)| = \frac{1}{h}\sum_{k\in\mathbb{Z}} |F(j\omega + j\omega_s k)|$$

$$\omega_s = \frac{2\pi}{h}$$

Anti-aliasing filters (filter before sampling!)

Quiz question: aliasing

Q: The phenomenon that sampling ($\omega_s = 2\pi/h$) of two or more harmonic signals with different frequencies may result in the same discrete time signal is called aliasing.

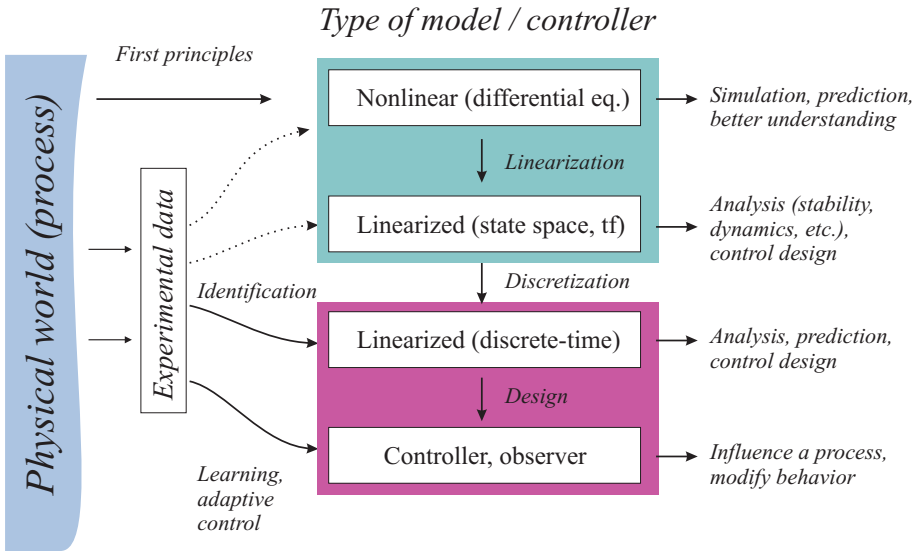
Assume that you are going to sample a system that has frequency content up to ω_{max}. Should you sample with sample frequency

1. $\omega_s \geq 2\omega_{max}$

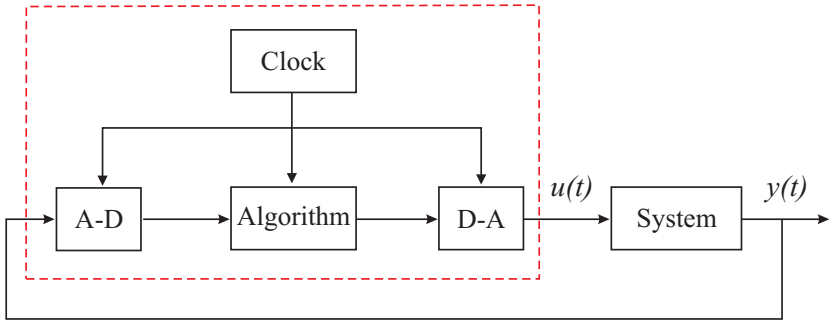
   or

2. $\omega_s \geq \frac{1}{2}\omega_{max}$?

The big picture

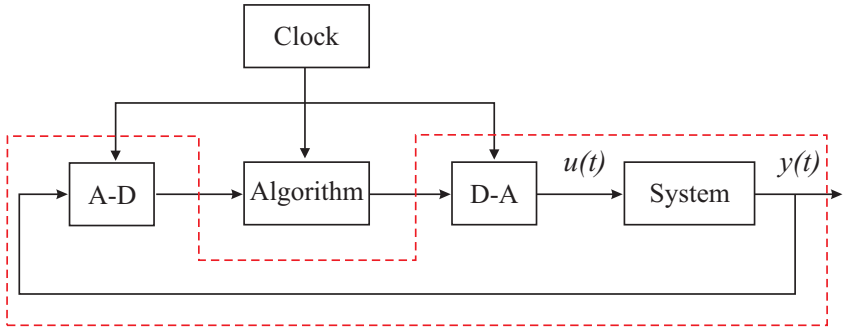


Computer-controlled system: approach 1



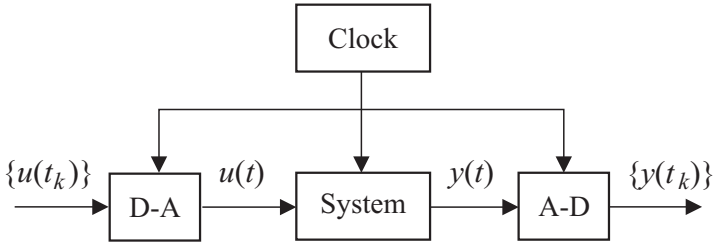
Design a continuous-time controller, make sure that the computer implementation resembles the continuous-time controller as well as possible.

Computer-controlled system: approach 2



Describe the system from the computer's viewpoint and design directly a discrete-time controller.

System from the computer's viewpoint



Zero-order hold sampling of systems

System description:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

- Let the input be piecewise constant (ZOH).
- Look at the behavior at sampling points only.
- Use linearity and calculate step responses.

Solving the system equation

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t)$$

$$\begin{aligned} x(t) &= e^{A(t-t_k)}x(t_k) + \int_{t_k}^{t} e^{A(t-s')}Bu(s')ds' \\ &= e^{A(t-t_k)}x(t_k) + \int_{t_k}^{t} e^{A(t-s')}ds'Bu(t_k) \\ &= e^{A(t-t_k)}x(t_k) + \int_{0}^{t-t_k} e^{As}dsBu(t_k) \\ &= \Phi(t,t_k)x(t_k) + \Gamma(t,t_k)u(t_k) \end{aligned}$$

General discrete-time model

$$x(t_{k+1}) = \Phi(t_{k+1}, t_k)x(t_k) + \Gamma(t_{k+1}, t_k)u(t_k)$$
$$y(t_k) = Cx(t_k) + Du(t_k)$$

with

$$\Phi(t_{k+1}, t_k) = e^{A(t_{k+1}-t_k)} \quad , \quad \Gamma(t_{k+1}, t_k) = \int_0^{t_{k+1}-t_k} e^{As} ds B$$

linear difference equation

Periodic sampling

$$\Phi(t_{k+1}, t_k) = e^{A(t_{k+1}-t_k)} \quad , \quad \Gamma(t_{k+1}, t_k) = \int_0^{t_{k+1}-t_k} e^{As} ds B$$

$t_k = k \cdot h \quad \rightarrow \quad t_{k+1} - t_k = h$ (sampling period)

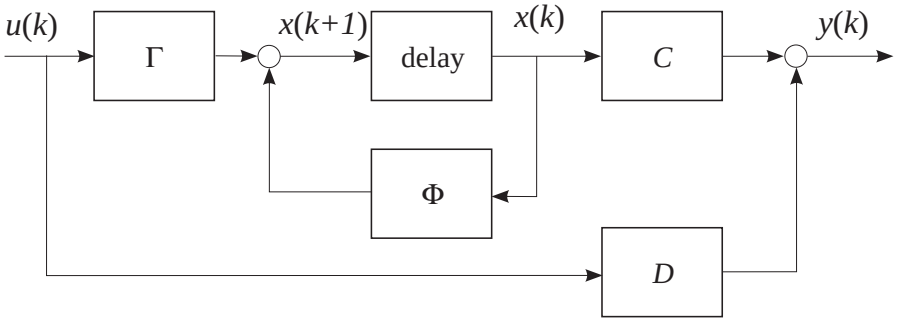
$$x(kh + h) = \Phi x(kh) + \Gamma u(kh)$$
$$y(kh) = Cx(kh) + Du(kh)$$

with

$$\Phi = e^{Ah} \quad , \quad \Gamma = \int_0^h e^{As} ds B$$

Discrete-time state-space system

$$x(k+1) = \Phi x(k) + \Gamma u(k)$$
$$y(k) = Cx(k) + Du(k)$$



Example: first-order system

$$\frac{dx(t)}{dt} = ax(t) + bu(t)$$

System matrices (parameters):

$$\Phi = e^{ah}$$
$$\Gamma = \int_0^h e^{as} ds \cdot b = \frac{b}{a}\,(e^{ah} - 1)$$

Discrete-time system:

$$x(kh + h) = e^{ah}x(kh) + \frac{b}{a}(e^{ah} - 1)u(kh)$$

Computing Φ and Γ

- Series expansion of the matrix exponential.
- The Laplace transform – the Laplace transform of $\exp(At)$ is $(sI - A)^{-1}$.
- Cayley-Hamilton's theorem (computation of matrix functions, see Appendix B).
- Numerical calculation in MATLAB or other SW.
- Symbolic computer algebra, using programs such as Maple and Mathematica, or Symbolic Toolbox in MATLAB.

Series expansion

$$\Phi = e^{Ah} \qquad \Gamma = \int_0^h e^{As} ds B$$

$$\Phi = e^{Ah} = I + Ah + \frac{A^2h^2}{2!} + \frac{A^3h^3}{3!} + \cdots$$

$$\Psi = \int_0^h e^{As} ds = Ih + \frac{Ah^2}{2!} + \frac{A^2h^3}{3!} + \cdots$$

$$\Gamma = \Psi B$$

Example: sampling of double integrator

$$\frac{dx}{dt} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 \end{pmatrix} x$$

$$\Phi = e^{Ah} = I + Ah + A^2h^2/2 + \cdots = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & h \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}$$

Example: sampling of double integrator

$$\frac{dx}{dt} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 \end{pmatrix} x$$

$$\Gamma = (Ih + Ah^2/2 + A^2h^3/3! + \cdots)B$$

$$= \begin{pmatrix} h & 0 \\ 0 & h \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 & \frac{h^2}{2} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{h^2}{2} \\ h \end{pmatrix}$$

Matrix exponential

$$\Phi = e^{Ah}$$
$$\Gamma = \int_0^h e^{As} ds B$$

Differentiate

$$\frac{d\Phi(t)}{dt} = A\Phi(t) = \Phi(t)A$$
$$\frac{d\Gamma(t)}{dt} = \Phi(t)B$$

$$\frac{d}{dt}\begin{pmatrix} \Phi(t) & \Gamma(t) \\ 0 & I \end{pmatrix} = \begin{pmatrix} \Phi(t) & \Gamma(t) \\ 0 & I \end{pmatrix}\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}$$

Matrix exponential

$$\frac{d}{dt}\begin{pmatrix} \Phi(t) & \Gamma(t) \\ 0 & I \end{pmatrix} = \begin{pmatrix} \Phi(t) & \Gamma(t) \\ 0 & I \end{pmatrix}\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}$$

$\Phi(h)$ and $\Gamma(h)$ can be obtained from the block matrix

$$\begin{pmatrix} \Phi(h) & \Gamma(h) \\ 0 & I \end{pmatrix} = \exp\left\{\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} h\right\}$$

Laplace transform

$$\mathcal{L}^{-1}\left\{(sI - A)^{-1}\right\} = e^{At}$$

Proof from $h(t) = \mathcal{L}^{-1}\{H(s)\}$, where

$$H(s) = C(sI - A)^{-1}B + D$$
$$h(t) = Ce^{At}B + D$$

Then use

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-s')}Bu(s')ds'$$

with $x(0) = 0$ and $u(t) = \delta(t)$.

Laplace transform

- To get $\Phi = e^{Ah}$ use the inverse Laplace transform of $(sI - A)^{-1}$ and substitute $t = h$.
- The function e^{At} can be used also to calculate $\Gamma = \int_0^h e^{As}Bds$.

Laplace transform example

DC motor model:

$$H(s) = \frac{1}{s(s+1)} \quad \Rightarrow \quad \frac{dx(t)}{dt} = \begin{pmatrix} -1 & 0 \\ 1 & 0 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u(t)$$

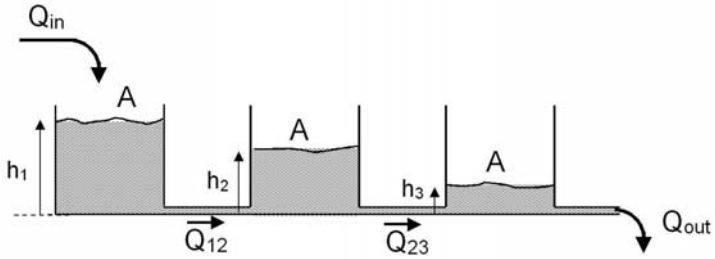
$$y(t) = \begin{pmatrix} 0 & 1 \end{pmatrix} x(t)$$

$$(sI - A)^{-1} = \begin{pmatrix} s+1 & 0 \\ -1 & s \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{s+1} & 0 \\ \frac{1}{s(s+1)} & \frac{1}{s} \end{pmatrix}$$

$$\Phi = e^{Ah} = \begin{pmatrix} e^{-h} & 0 \\ 1 - e^{-h} & 1 \end{pmatrix} \quad \Rightarrow \quad \Gamma = \int_0^h \begin{pmatrix} e^{-s'} \\ 1 - e^{-s'} \end{pmatrix} ds' = \begin{pmatrix} 1 - e^{-h} \\ h - 1 + e^{-h} \end{pmatrix}$$

Discretization in MATLAB

Water system:



From linearized balance equations:

$$\dot{h}_1 = \frac{1}{A} Q_{in} - \frac{k}{A}(h_1 - h_2)$$
$$\dot{h}_2 = \frac{k}{A}(h_1 - h_2) - \frac{k}{A}(h_2 - h_3)$$
$$\dot{h}_3 = \frac{k}{A}(h_2 - h_3) - \frac{k}{A} h_3$$

Discretization in MATLAB

$$A = \begin{pmatrix} -1 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{pmatrix} \quad B = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$ Discretize using c2d command.

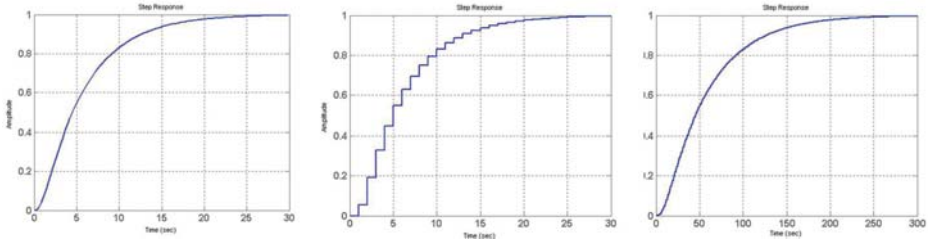
With $h = 1$:

$$\Phi = \begin{pmatrix} 0.5235 & 0.3070 & 0.1138 \\ 0.3070 & 0.3304 & 0.1931 \\ 0.1138 & 0.1931 & 0.2165 \end{pmatrix} \quad \Gamma = \begin{pmatrix} 0.7018 \\ 0.2253 \\ 0.0557 \end{pmatrix}$$

With $h = 0.1$:

$$\Phi = \begin{pmatrix} 0.9092 & 0.0864 & 0.0042 \\ 0.0864 & 0.8271 & 0.0821 \\ 0.0042 & 0.0821 & 0.8228 \end{pmatrix} \quad \Gamma = \begin{pmatrix} 0.0953 \\ 0.0045 \\ 0.0001 \end{pmatrix}$$

Open-loop step responses of water system



continuous $h = 1$ sec $h = 0.1$ sec

Sampling systems with a time delay

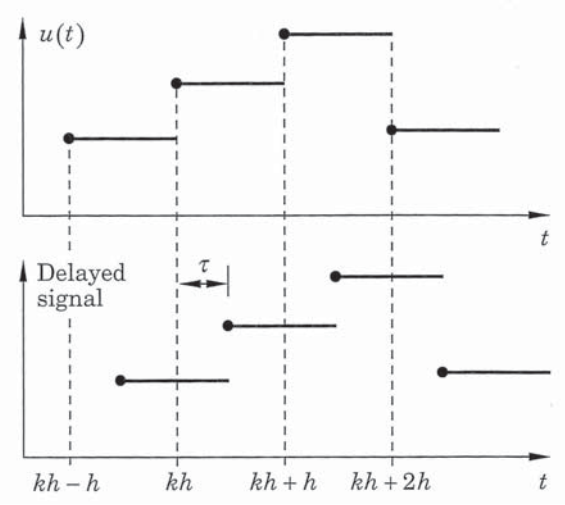
$$\frac{dx(t)}{dt} = Ax(t) + Bu(t-\tau)$$

$$x(kh+h) = e^{Ah}x(kh) + \int_{kh}^{kh+h} e^{A(kh+h-s')}Bu(s'-\tau)ds'$$

$$\int_{kh}^{kh+h} e^{A(kh+h-s')}Bu(s'-\tau)ds' = \int_{kh}^{kh+\tau} e^{A(kh+h-s')}Bds'u(kh-h) +$$

$$+ \int_{kh+\tau}^{kh+h} e^{A(kh+h-s')}Bds'u(kh) = \Gamma_1 u(kh-h) + \Gamma_0 u(kh)$$

Sampling systems with a time delay



Sampling systems with a time delay

$$x(kh+h) = \Phi x(kh) + \Gamma_0 u(kh) + \Gamma_1 u(kh-h)$$

State-space model:

$$\begin{pmatrix} x(kh+h) \\ u(kh) \end{pmatrix} = \begin{pmatrix} \Phi & \Gamma_1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x(kh) \\ u(kh-h) \end{pmatrix} + \begin{pmatrix} \Gamma_0 \\ I \end{pmatrix} u(kh)$$

r additional states introduced (r = number of inputs)

Double integrator with time delay

$$\Phi = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}$$

$$\Gamma_1 = \begin{pmatrix} 1 & h-\tau \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{\tau^2}{2} \\ \tau \end{pmatrix} = \begin{pmatrix} \tau\left(h - \frac{\tau}{2}\right) \\ \tau \end{pmatrix}$$

$$\Gamma_0 = \begin{pmatrix} \frac{(h-\tau)^2}{2} \\ h-\tau \end{pmatrix}$$

Longer time delays

In continuous time, time delay can be approximated by infinite series of first-order systems:

$$e^{-\tau s} = \lim_{n\to\infty} \left(\frac{1}{1+\frac{\tau}{n}s}\right)^n$$

In discrete time we know exactly what the order of the approximation is:

If $\tau = dh + \tau'$ with $0 < \tau' < h$,
then additional $(d+1)\cdot r$ states are needed.

Longer time delays

$$x(k+1) = \Phi x(k) + \Gamma_0 u(k-d) + \Gamma_1 u(k-d-1)$$

In state-space form:

$$\begin{pmatrix} x(k+1) \\ u(k-d) \\ \vdots \\ u(k-1) \\ u(k) \end{pmatrix} = \begin{pmatrix} \Phi & \Gamma_1 & \Gamma_0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x(k) \\ u(k-d-1) \\ \vdots \\ u(k-2) \\ u(k-1) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ I \end{pmatrix} u(k)$$

Inverse of sampling

Is it always possible to find a continuous-time system that corresponds to a discrete-time system? Two problems:
existence, uniqueness.

Existence (example):

$$x(kh+h) = \Phi x(kh) + \Gamma u(kh)$$

$$e^{ah} = \Phi \quad \longrightarrow \quad a = \frac{1}{h}\ln\Phi$$

$$\frac{b}{a}(e^{ah}-1) = \Gamma \quad \longrightarrow \quad b = \frac{1}{h}\ln\Phi \cdot \frac{\Gamma}{\Phi - 1}$$

Continuous-time system with real coefficients obtained only for positive Φ.

Solution of discrete-time system equations

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned}$$

Iterate the equations:

$$\begin{aligned} x(k_0+1) &= \Phi x(k_0) + \Gamma u(k_0) \\ x(k_0+2) &= \Phi x(k_0+1) + \Gamma u(k_0+1) \\ &= \Phi^2 x(k_0) + \Phi\Gamma u(k_0) + \Gamma u(k_0+1) \\ &\vdots \\ x(k) &= \Phi^{k-k_0} x(k_0) + \Phi^{k-k_0-1}\Gamma u(k_0) + \cdots + \Gamma u(k-1) \end{aligned}$$

Solution of discrete-time system equations

$$x(k) = \Phi^{k-k_0}x(k_0) + \Phi^{k-k_0-1}\Gamma u(k_0) + \cdots + \Gamma u(k-1)$$

$$= \Phi^{k-k_0}x(k_0) + \sum_{j=k_0}^{k-1} \Phi^{k-j-1}\Gamma u(j)$$

$$y(k) = C\Phi^{k-k_0}x(k_0) + \sum_{j=k_0}^{k-1} C\Phi^{k-j-1}\Gamma u(j) + Du(k)$$

Initial value + influence of input signal

Input-output models

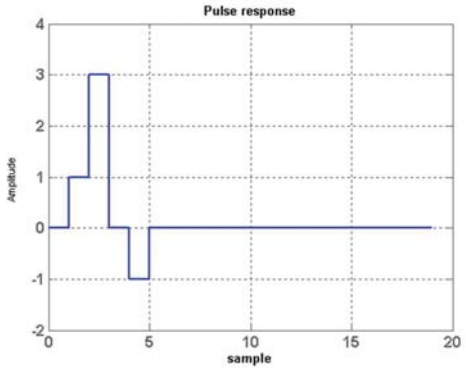
$$y(k) = C\Phi^{k-k_0}x(k_0) + \sum_{j=k_0}^{k-1} C\Phi^{k-j-1}\Gamma u(j) + Du(k)$$

$$= C\Phi^{k-k_0}x(k_0) + \sum_{j=k_0}^{k} h(k-j)u(j)$$

Pulse-response (weighting) function:

$$h(k) = \begin{cases} 0 & k < 0 \\ D & k = 0 \\ C\Phi^{k-1}\Gamma & k > 0 \end{cases}$$

Discrete-time convolution example



Pulse response:

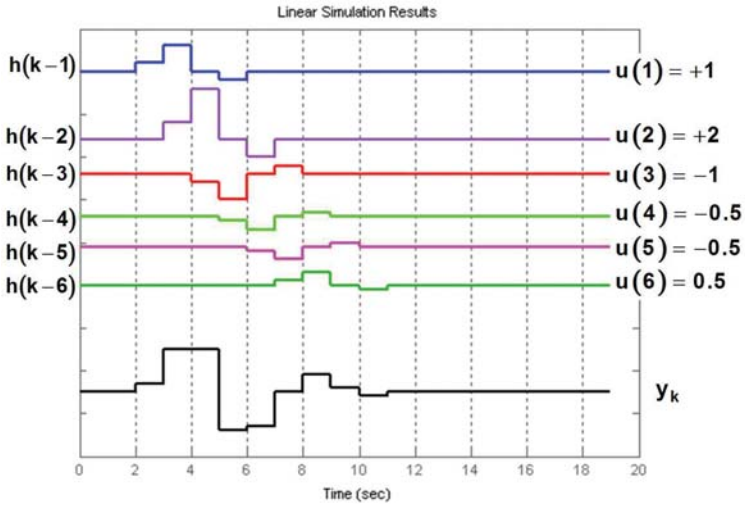
$$h(k) = \begin{cases} 0 \\ 1 \\ 3 \\ 0 \\ -1 \\ 0 \end{cases}$$

Input:

$$u(k) = \begin{cases} 1 \\ 2 \\ -1 \\ -0.5 \\ -0.5 \\ 0.5 \end{cases}$$

Discrete-time convolution example

Discrete convolution is just a summation: $y(k) = \sum_{j=1}^{k} h(k-j)u(j)$



Change of coordinates

Introduce new state vector $z(k) = Tx(k)$

$$\begin{aligned} z(k+1) = Tx(k+1) &= T\Phi x(k) + T\Gamma u(k) \\ &= T\Phi T^{-1} z(k) + T\Gamma u(k) \\ &= \tilde{\Phi} z(k) + \tilde{\Gamma} u(k) \end{aligned}$$

and

$$\begin{aligned} y(k) &= Cx(k) + Du(k) = CT^{-1} z(k) + Du(k) \\ &= \tilde{C} z(k) + \tilde{D} u(k) \end{aligned}$$

Change of coordinates

- State space representation depends on the coordinate system
- Invariants:
  - Characteristic equation $\det[\lambda I - \Phi]$
  - Input-output representation
- Simple representations:
  - Diagonal form
  - Jordan form

Diagonal form

Assume that Φ has distinct eigenvalues λ_i, then

$$T\Phi T^{-1} = \begin{pmatrix} \lambda_1 & & 0 \\ & \cdots & \\ 0 & & \lambda_n \end{pmatrix}$$

First-order decoupled difference equations

$$\begin{aligned} z_1(k+1) &= \lambda_1 z_1(k) + \beta_1 u(k) \\ &\vdots \\ z_n(k+1) &= \lambda_n z_n(k) + \beta_n u(k) \end{aligned}$$

Diagonal form – solution

Each mode has the solution

$$z_i(k) = \lambda_i^k z_i(0) + \sum_{j=0}^{k-1} \lambda_i^{k-j-1} \beta_i u(j)$$

Notice the importance of λ_i

Shift operator

Forward shift operator

$$\begin{aligned} qf(k) &= f(k+1) \\ q^2 f(k) &= f(k+2), \text{ etc.} \end{aligned}$$

Backward shift operator

$$\begin{aligned} q^{-1} f(k) &= f(k-1) \\ q^{-2} f(k) &= f(k-2), \text{ etc.} \end{aligned}$$

Compare with the differential operator $p = \frac{d}{dt}$.

Representing difference equations

$$y(k+n_a) + a_1 y(k+n_a-1) + \cdots + a_{n_a} y(k) = b_0 u(k+n_b) + \cdots + b_{n_b} u(k)$$

where $n_a \geq n_b$. Using the shift operator gives

$$(q^{n_a} + a_1 q^{n_a-1} + \cdots + a_{n_a}) y(k) = (b_0 q^{n_b} + \cdots + b_{n_b}) u(k)$$

introduce polynomials:

$$A(q)y(k) = B(q)u(k)$$

or

$$y(k) = \frac{B(q)}{A(q)} u(k)$$

Representing difference equations

Reciprocal polynomial:

$$A^*(q) = 1 + a_1 q + \cdots + a_{n_a} q^{n_a} = q^{n_a} A(q^{-1})$$

by reversing the order of the coefficients.

$$A^*(q^{-1}) y(k) = B^*(q^{-1}) u(k-d), \quad d = n_a - n_b$$

Example:

$$\begin{aligned} \left(q^2 + a_1 q + a_2\right) y(k) &= \left(b_0 q + b_1\right) u(k) \\ y(k+2) + a_1 y(k+1) + a_2 y(k) &= b_0 u(k+1) + b_1 u(k) \\ y(k) + a_1 y(k-1) + a_2 y(k-2) &= b_0 u(k-1) + b_1 u(k-2) \\ \left(1 + a_1 q^{-1} + a_2 q^{-2}\right) y(k) &= \left(b_0 q^{-1} + b_1 q^{-2}\right) u(k) \\ &= q^{-1} \left(b_0 + b_1 q^{-1}\right) u(k) \end{aligned}$$

so $y(k) = -a_1 y(k-1) - a_2 y(k-2) + b_0 u(k-1) + b_1 u(k-2)$, one time step delay.

Pulse-transfer function operator

$$x(k+1) = qx(k) = \Phi x(k) + \Gamma u(k)$$

$$\begin{aligned} (qI - \Phi) x(k) &= \Gamma u(k) \\ x(k) &= (qI - \Phi)^{-1} \Gamma u(k) \end{aligned}$$

$$\begin{aligned} y(k) &= Cx(k) + Du(k) = \left(C(qI - \Phi)^{-1}\Gamma + D\right) u(k) \\ y(k) &= H(q) u(k) \end{aligned}$$

Pulse-transfer operator: $H(q) = C(qI - \Phi)^{-1}\Gamma + D$

SISO systems

$$H(q) = C(qI - \Phi)^{-1}\Gamma + D = \frac{B(q)}{A(q)}$$

If no common factors

$$\begin{aligned} \deg A &= n \\ A(q) &= \det[qI - \Phi] \end{aligned}$$

and

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

where a_i are the coefficients of the characteristic polynomial of Φ (usually $b_0 = 0$ ... causality)

Poles, zeros and system order

$$H(q) = C(qI - \Phi)^{-1}\Gamma + D = \frac{B(q)}{A(q)}$$

Poles: $A(q) = 0$

Zeros: $B(q) = 0$

System order: $\deg A(q)$

$$H^*(q^{-1}) = C(I - q^{-1}\Phi)^{-1}q^{-1}\Gamma + D$$

Example - double integrator

$$H(q) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} q-1 & -1 \\ 0 & q-1 \end{pmatrix}^{-1} \begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix} = \frac{1}{2}\frac{q+1}{(q-1)^2}$$

zero at -1, 2 poles at $+1$

$$H^*(q^{-1}) = \frac{1}{2}q^{-1}\frac{1+q^{-1}}{(1-q^{-1})^2}$$

Example - double integrator with a delay

$h = 1$ and $\tau = 0.5$ gives

$$\Phi = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \Gamma_1 = \begin{pmatrix} 0.375 \\ 0.5 \end{pmatrix} \quad \Gamma_0 = \begin{pmatrix} 0.125 \\ 0.5 \end{pmatrix}$$

$$\begin{aligned} H(q) &= C(qI - \Phi)^{-1}(\Gamma_0 + \Gamma_1 q^{-1}) \\ &= \begin{pmatrix} 1 & 0 \end{pmatrix} \frac{\begin{pmatrix} q-1 & 1 \\ 0 & q-1 \end{pmatrix}}{(q-1)^2} \begin{pmatrix} 0.125 + 0.375q^{-1} \\ 0.5 + 0.5q^{-1} \end{pmatrix} \end{aligned}$$

Example - double integrator with a delay

$$H(q) = \begin{pmatrix} 1 & 0 \end{pmatrix} \frac{\begin{pmatrix} q-1 & 1 \\ 0 & q-1 \end{pmatrix}}{(q-1)^2} \begin{pmatrix} 0.125 + 0.375q^{-1} \\ 0.5 + 0.5q^{-1} \end{pmatrix}$$

$$= \frac{0.125(q^2+6q+1)}{q(q^2-2q+1)} = \frac{0.125(q^{-1}+6q^{-2}+q^{-3})}{1-2q^{-1}+q^{-2}}$$

order: 3, poles: 0, 1 and 1, zeros: $-3 \pm \sqrt{8}$

How to get $H(q)$ from $G(s)$?

Use Table 2.1 in book

$G(s)$	$H(q)$
$\frac{1}{s}$	$\frac{h}{q-1}$
$\frac{1}{s^2}$	$\frac{h^2(q+1)}{2(q-1)^2}$
e^{-sh}	q^{-1}
$\frac{a}{s+a}$	$\frac{1-\exp(-ah)}{q-\exp(-ah)}$

zero-order hold included !

Summary

- Aliasing, filter before sampling.
- Sampling of continuous-time systems.
- State-space and input-output models.
- Shift-operator calculus.
- Pulse-transfer function, poles and zeros.

Digital Control (SC42095)

Lecture 3, November 20, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Poles and zeros
- Selection of sampling interval
- z–transform, relation to shift operator
- Computation of $H(z)$ from $G(s)$
- Obtaining a process model
- Linearization

Interpretation of poles and zeros

Poles:

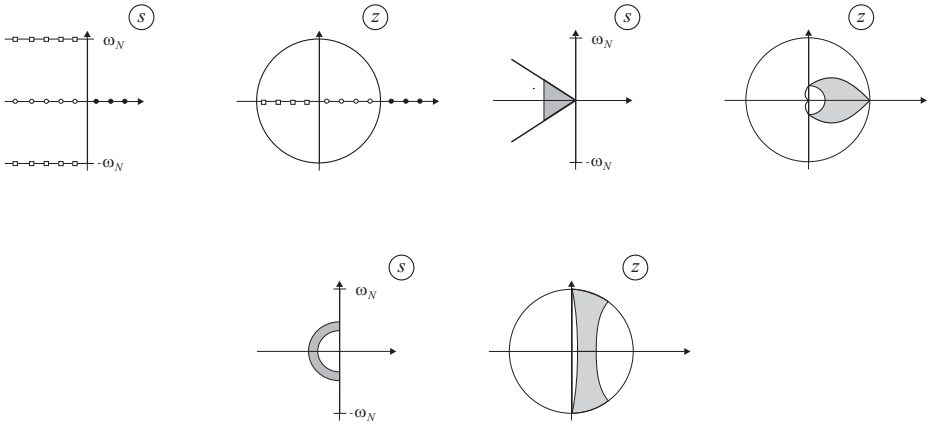
- A pole is an eigenvalue of Φ
- A pole $z = a$ corresponds to a free mode $y(k) = a^k$

Zeros:

- A zero $z = a$ implies that the transmission of the input $u(k) = a^k$ is blocked by the system
- A zero is related to how inputs and outputs are coupled to the states

Transformation of poles

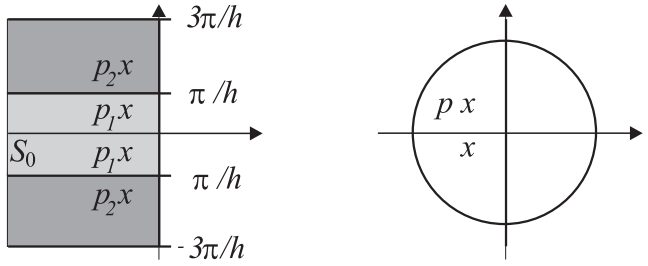
$\lambda_i(\Phi) = e^{\lambda_i(A)h}$ (property of matrix functions)



Another evidence of alias problem

$$z = e^{sh}$$

Several points in the s-plane are mapped into the same point in the z-plane.



Sampling of a second order system

$$\frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$$

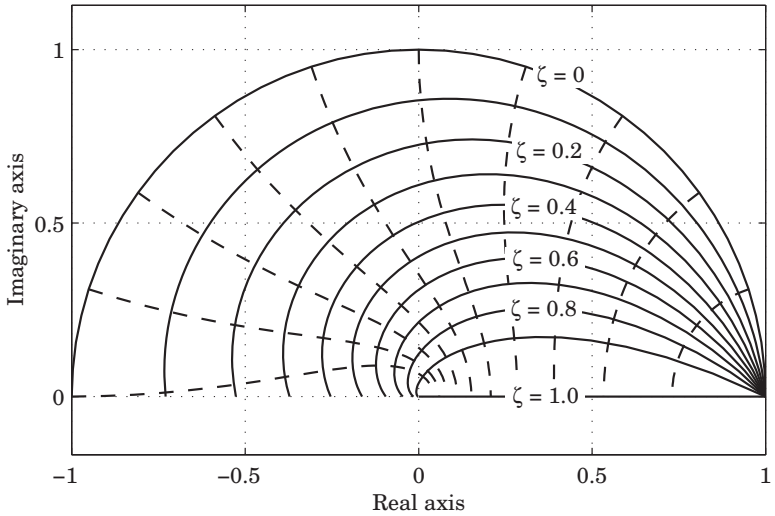
Poles of the discrete-time system are given by

$$z^2 + a_1 z + a_2 = 0$$

where

$$a_1 = -2e^{-\zeta\omega_0 h}\cos\left(\sqrt{1-\zeta^2}\omega_0 h\right)$$
$$a_2 = -2e^{-\zeta\omega_0 h}$$

Sampling of a second-order system



Transformation of zeros

No simple formula for the mapping of zeros
For short sampling periods ($h \to 0$)

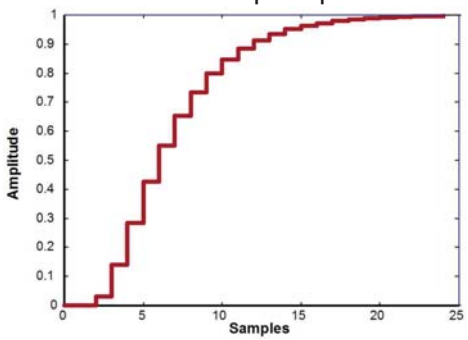
$$z_i \approx e^{s_i h}$$

plus $r = d - 1$ zeros introduced by sampling

d	Z_d
1	1
2	$z + 1$
3	$z^2 + 4z + 1$
4	$z^3 + 11z^2 + 11z + 1$
5	$z^4 + 26z^3 + 66z^2 + 26z + 1$

System with unstable inverse

Step response

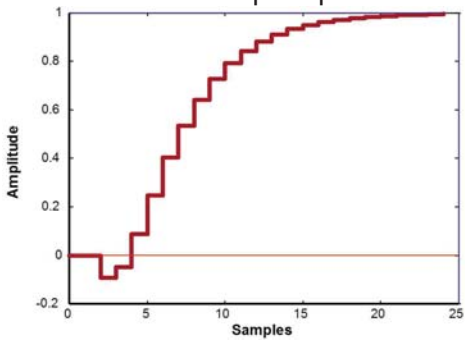


$$H(z) = K\frac{z+2}{\left(z-\frac{1}{4}\right)\left(z-\frac{1}{2}\right)\left(z-\frac{3}{4}\right)}$$

Zero outside unit circle
(unstable inverse)

Other system with unstable inverse

Step response

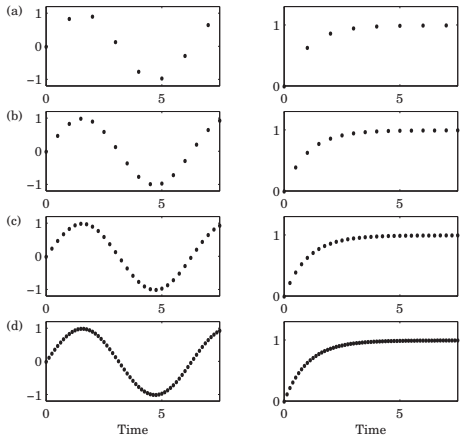


$$H(z) = K\frac{z-2}{\left(z-\frac{1}{4}\right)\left(z-\frac{1}{2}\right)\left(z-\frac{3}{4}\right)}$$

Zero outside unit circle
(unstable inverse)

Selection of sampling period

Number of samples per rise time: $N_r = \frac{T_r}{h} \approx 4 - 10$ is reasonable



a) $N_r = 1$

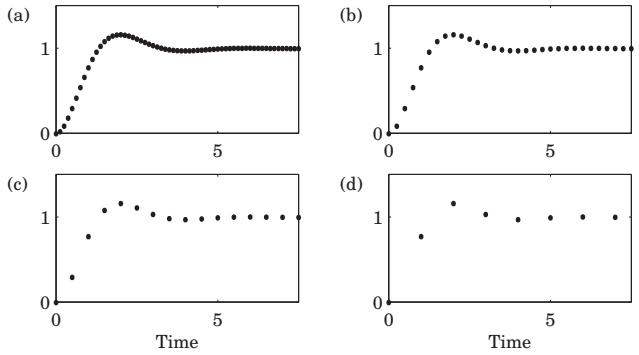
b) $N_r = 2$

c) $N_r = 4$

d) $N_r = 8$

Second order system

$N_r = \frac{T_r}{h} \approx 4 - 10$ corresponds to $\omega_0 h \approx 0.2 - 0.6$



a) $h = 0.125$ ($\omega_0 h = 0.23$), b) $h = 0.250$ ($\omega_0 h = 0.46$),
c) $h = 0.500$ ($\omega_0 h = 0.92$), d) $h = 1.000$ ($\omega_0 h = 1.83$)

z-transform

- Continuous systems: Laplace transform,
  discrete systems: z-transform

Consider discrete-time signal: $\{f(kh) \mid k = 0, 1, \ldots\}$

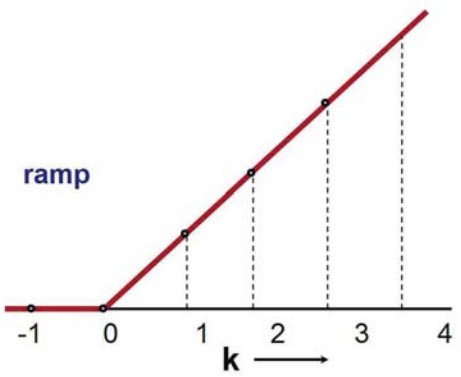
The z-transform of $f(kh)$ is defined as

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

where z is a complex variable.

Example: a ramp signal

Ramp: $y(kh) = kh$ for $k \geq 0$



$$\begin{aligned} Y(z) &= \sum_{k=0}^{\infty} f(kh) z^{-k} \\ &= 0 + hz^{-1} + 2hz^{-2} + \cdots \\ &= h(z^{-1} + 2z^{-2} + \cdots) \\ &= \frac{hz}{(z-1)^2} \end{aligned}$$

Properties of z-transform

1. Linearity:

$$Z\{\alpha f + \beta g\} = \alpha Z f + \beta Z g$$

2. Time-shift:

$$Z\{q^{-n} f\} = z^{-n} F$$

$$Z\{q^{n} f\} = z^{n} (F - F_1)$$

where

$$F_1(z) = \sum_{j=0}^{n-1} f(jh) z^{-j}$$

Properties of z-transform (cont'd)

3. Initial value theorem:

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

4. Final value theorem:

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

(if $(1 - z^{-1})F(z)$ has no poles on or outside the unit circle)

5. Convolution:

$$Z(f * g) = Z\left\{ \sum_{n=0}^{k} f(n) g(k-n) \right\} = (Zf)\,(Zg)$$

Solving difference equations

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned}$$

Take the z-transform of both sides

$$\sum_{k=0}^{\infty} z^{-k} x(k+1) = \sum_{k=0}^{\infty} \Phi z^{-k} x(k) + \sum_{k=0}^{\infty} \Gamma z^{-k} u(k)$$

$$z\left(\sum_{k=0}^{\infty} z^{-k} x(k) - x(0)\right) = \sum_{k=0}^{\infty} \Phi z^{-k} x(k) + \sum_{k=0}^{\infty} \Gamma z^{-k} u(k)$$

Solving difference equations

$$\begin{aligned} z\Big(X(z) - x(0)\Big) &= \Phi X(z) + \Gamma U(z) \\ X(z) &= (zI - \Phi)^{-1}(zx(0) + \Gamma U(z)) \end{aligned}$$

$$Y(z) = C(zI - \Phi)^{-1} zx(0) + (C(zI - \Phi)^{-1}\Gamma + D)U(z)$$

Pulse-transfer function (the same as with q):

$$H(z) = C(zI - \Phi)^{-1}\Gamma + D$$

Why both z and q ?

- Both can be used to manipulate difference equations, sometimes, the same notation used for both.
- Formal difference: q is a shift operator, z is a complex variable.
- Also a system-theoretic reason: example

$$y(k+1) + ay(k) = u(k+1) + au(k) \qquad (1)$$

$$H(z) = \frac{z+a}{z+a} = 1 \quad (z \text{ is a complex variable})$$

Solution of (1): $y(k) = (-a)^k y(0) - a^k u(0) + u(k)$

Calculation of $H(z)$ from $G(s)$

Method 1: State-space model, calculate Φ and Γ, find $H(z)$
1. Matrices Φ and Γ

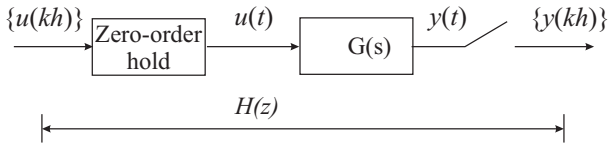
$$\Phi = e^{Ah} \quad , \quad \Gamma = \int_0^h e^{As} ds B$$

2. Convert $\{\Phi, \Gamma, C, D\}$ to pulse transfer function:

$$H(z) = C(zI - \Phi)^{-1}\Gamma + D$$

Calculation of $H(z)$ from $G(s)$

Method 2:



1. Determine the step response of the system: $Y(s) = G(s)/s$.
2. Determine the corresponding z-transform of the step response using the table: $\tilde{Y}(z) = Z(L^{-1}Y)$.
3. Divide by the z-transform of the step function:

$$H(z) = \frac{z-1}{z}\tilde{Y}(z)$$

Example: double integrator

$$1) \quad Y(s) = G(s) \cdot \frac{1}{s} = \frac{1}{s^3}$$

$$2) \quad Y(z) = \frac{h^2}{2}\frac{z(z+1)}{(z-1)^3} \qquad \text{(Table 2.3)}$$

$$3) \quad H(z) = \frac{Y(z)}{U(z)} = \frac{h^2}{2}\frac{z(z+1)}{(z-1)^3} \cdot \frac{z-1}{z} = \frac{h^2}{2}\frac{(z+1)}{(z-1)^2}$$

Compare with the use of state-space representation.

z-transform table

$f(kh)$	L_f	Z_f
$\delta(k)$ (pulse)	-	1
1 $k \geq 0$ (step)	$\frac{1}{s}$	$\frac{z}{z-1}$
kh	$\frac{1}{s^2}$	$\frac{hz}{(z-1)^2}$
$\frac{1}{2}(kh)^2$	$\frac{1}{s^3}$	$\frac{h^2 z(z+1)}{2(z-1)^3}$
$e^{-kh/T}$	$\frac{T}{1+sT}$	$\frac{z}{z-e^{-h/T}}$

Use the table correctly! Z_f in the table *does not* give the zero-order-hold sampling of a system with the transfer function L_f.

Modified z-transform

$$\tilde{F}(z,m) = \sum_{k=0}^{\infty} z^{-k} f(kh - h + mh), \quad 0 \leq m \leq 1$$

Can be used to determine intersample behavior

Controller's computational delay less than h

Using Matlab to change system representation

$$G(s) = \frac{b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_1 s + a_0}$$

$$\Updownarrow$$

`ss2tf` and `tf2ss`

$$\Updownarrow$$

$$\frac{dx(t)}{dt} = \begin{pmatrix} -a_{n-1} & -a_{n-2} & \cdots & -a_1 & -a_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} u(t)$$

$$y(t) = \begin{pmatrix} b_{n-1} & b_{n-2} & \cdots & b_1 & b_0 \end{pmatrix} x(t)$$

How to obtain process model?

- physical (mechanistic) modeling
  1. first principles → differential equations (nonlinear)
  2. linearization around an operating point
  3. discretization
- system identification
  1. measure input–output data (around an operating point)
  2. define model structure (order)
  3. estimate model parameters from data (least squares)

Linearization of nonlinear models

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= g(x,u) \end{aligned}$$

Choose x_0, y_0 or u_0 such that $0 = f(x_0, u_0)$ and $y_0 = g(x_0, u_0)$ (equilibrium). Linearize the above model:

$$\dot{x} = \left.\frac{\partial f}{\partial x}\right|_{\substack{x=x_0 \\ u=u_0}} (x - x_0) \;+\; \left.\frac{\partial f}{\partial u}\right|_{\substack{x=x_0 \\ u=u_0}} (u - u_0)$$

$$y - y_0 = \left.\frac{\partial g}{\partial x}\right|_{\substack{x=x_0 \\ u=u_0}} (x - x_0) \;+\; \left.\frac{\partial g}{\partial u}\right|_{\substack{x=x_0 \\ u=u_0}} (u - u_0)$$

Linearization of nonlinear models

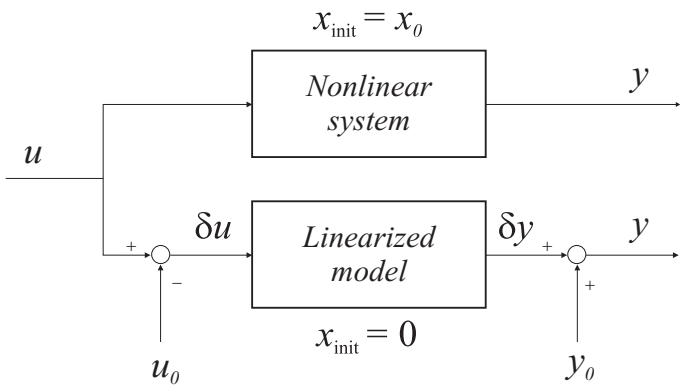
$$\dot{x} = \left.\frac{\partial f}{\partial x}\right|_{\substack{x=x_0 \\ u=u_0}} (x - x_0) \;+\; \left.\frac{\partial f}{\partial u}\right|_{\substack{x=x_0 \\ u=u_0}} (u - u_0)$$

$$y - y_0 = \left.\frac{\partial g}{\partial x}\right|_{\substack{x=x_0 \\ u=u_0}} (x - x_0) \;+\; \left.\frac{\partial g}{\partial u}\right|_{\substack{x=x_0 \\ u=u_0}} (u - u_0)$$

$$\begin{aligned} \dot{x} &= A\,\delta x \;+\; B\,\delta u \\ \delta y &= C\,\delta x \;+\; D\,\delta u \end{aligned}$$

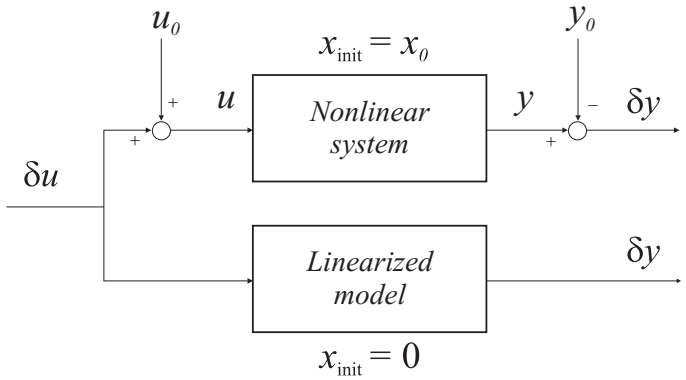
with $\delta x = x - x_0$, $\delta u = u - u_0$, $\delta y = y - y_0$, etc.

Comparison of models through simulation



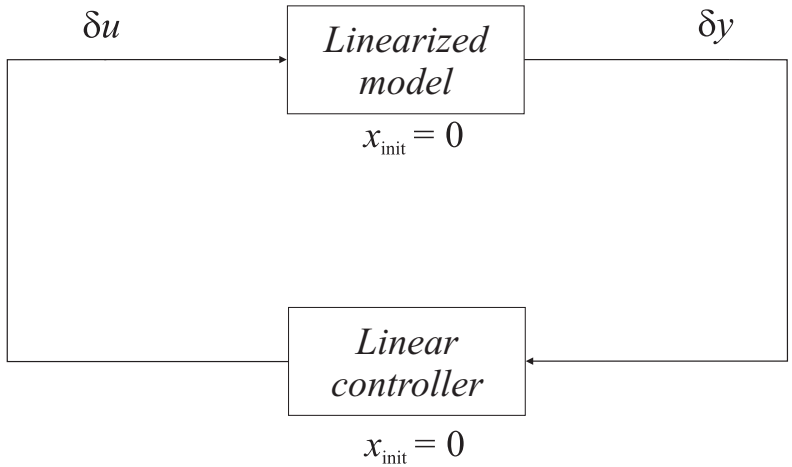
$\delta u = u - u_0$ $\qquad$ $y = \delta y + y_0$

Alternatively . . .

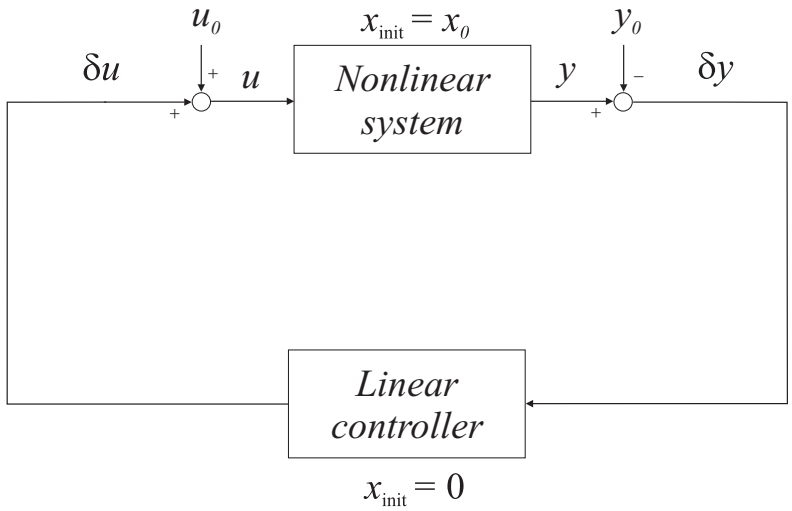


$u = \delta u + u_0$ $\qquad$ $\delta y = y - y_0$

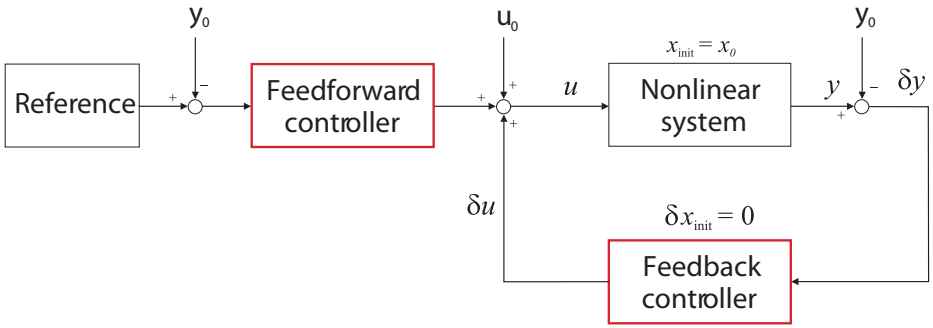
Simulation of a linear controller



Control by linear controller



Two-degree-of-freedom control



Linear controllers must use signals δu, δy, δx !

Summary

- Poles and zeros in discrete time
- Selection of sampling interval
- z–transform
- How to obtain $H(z)$ from $G(s)$
- Linearization of nonlinear models

Digital Control (SC42095)

Lecture 4, November 27, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Analysis of discrete-time systems
  - Stability (definitions, tests, Lyapunov method)
  - Controllability and reachability
  - Observability and detectability
  - Simulation

Stability: basic notions

Defined with respect to change in initial conditions.

The solution $x^0(k)$ of

$$x(k+1) = f(x(k), k)$$

is *stable* if for a given $\epsilon > 0$, there exists a $\delta(\epsilon, k_0) > 0$

such that all solutions $x(k)$ with $\|x(k_0) - x^0(k_0)\| < \delta$

are such that $\|x(k) - x^0(k)\| < \epsilon$ for $k \geq k_0$.

Definition - Asymptotic stability

The solution $x^0(k)$ is *asymptotically stable*

if it is stable and δ can be chosen such that

$\|x(k_0) - x^0(k_0)\| < \delta$ implies that

$\|x(k) - x^0(k)\| \to 0$ when $k \to \infty$.

Linear discrete-time systems

$$x(k+1) = \Phi x(k), \quad x(0) = a$$

Stability of *one solution* implies stability of all solutions, i.e., stability of the *system*.

A linear discrete-time system is asymptotically stable if and only if all eigenvalues of Φ are strictly inside the unit disc ($|\lambda_i| < 1$).

(Solutions $x(k) = \Phi^k x(0)$ are combinations of terms λ_i^k or $p_i(k)\lambda_i^k$.)

BIBO stability

A linear time-invariant system is BIBO stable if a bounded input gives a bounded output for all initial values.

Asymptotic stability is the strongest concept.

Asymptotically stable $\Rightarrow$ BIBO stable,

but

stable $\not\Rightarrow$ BIBO stable.

Stability tests

- Direct numerical or algebraic computation of $\lambda(\Phi)$ in MATLAB: `eig(Phi)`.
- Methods based on properties of the characteristic polynomial (Jury's criterion).

$$A(z) = a_0 z^n + a_1 z^{n-1} + \ldots + a_n = 0$$

- Root-locus method, Nyquist criterion, Bode plot.
- Lyapunov method.

Lyapunov theory

$$x(k+1) = f(x(k)), \quad f(0) = 0$$

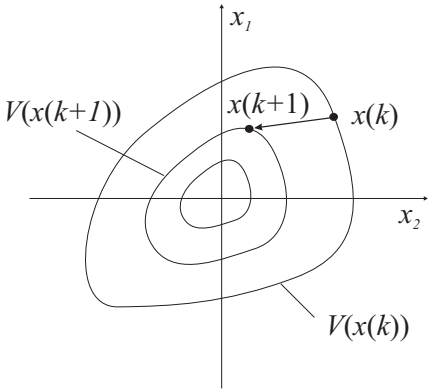
Monotonic convergence $\|x(k+1)\| < \|x(k)\|$ – too strong condition for stability. Another measure: Lyapunov function:

- $V(x)$ is continuous in x and $V(0) = 0$
- $V(x)$ is positive definite
- $\Delta V(x) = V(f(x)) - V(x)$ is negative definite

Existence of a Lyapunov function implies asymptotic stability for the solution $x = 0$

Geometric interpretation

$$x(k+1) = f(x(k)), \quad f(0) = 0$$



Linear system

$$x(k+1) = \Phi x(k), \qquad V(x) = x^T P x, \quad P > 0$$

$$\Delta V(k) = V(\Phi x) - V(x) = x^T \Phi^T P \Phi x - x^T P x$$

$$= x^T (\Phi^T P \Phi - P) x$$

V is a Lyapunov function iff there exists a $P > 0$ that satisfies the *Lyapunov equation*

$$\Phi^T P \Phi - P = -Q, \qquad Q > 0$$

Controllability and reachability

Controllability: Possible to find a control sequence such that the origin can be reached from any initial state in finite time.

Reachability: Possible to find a control sequence such that an arbitrary state can be reached from any initial state in finite time.

With continuous-time systems, controllability implies reachability.

Not for discrete-time systems...

Controllability and reachability (cont'd)

$$x(k+1) = \Phi x(k) + \Gamma u(k), \quad x(0) \text{ given}$$
$$y(k) = Cx(k)$$

At time n ($n =$ order of the system):

$$x(n) = \Phi^n x(0) + \Phi^{n-1}\Gamma u(0) + \ldots + \Gamma u(n-1)$$
$$x(n) = \Phi^n x(0) + W_c U$$

$$W_c = \begin{pmatrix} \Gamma & \Phi\Gamma & \ldots & \Phi^{n-1}\Gamma \end{pmatrix}, \quad U = \begin{pmatrix} u^T(n-1) & \ldots & u^T(0) \end{pmatrix}^T$$

System is reachable iff W_c has rank n
W_c is called the controllability matrix

Reachability and coordinate transformation

$$\begin{aligned}\tilde{W}_c &= \begin{pmatrix}\tilde{\Gamma} & \tilde{\Phi}\tilde{\Gamma} & \dots & \tilde{\Phi}^{n-1}\tilde{\Gamma}\end{pmatrix} \\ &= \begin{pmatrix}T\Gamma & T\Phi T^{-1}T\Gamma & \dots & T\Phi^{n-1}T^{-1}T\Gamma\end{pmatrix} \\ &= TW_c\end{aligned}$$

$\Rightarrow$ reachability is not influenced by change in coordinates

Controllable canonical form

If W_c is nonsingular and Φ has the characteristic polynomial:

$$\det[\lambda I - \Phi] = \lambda^n + a_1\lambda^{n-1} + \cdots + a_n = 0$$

then there exists a transformation $T = \tilde{W}_c W_c^{-1}$ such that

$$z(k+1) = \begin{pmatrix} -a_1 & -a_2 & \dots & -a_{n-1} & -a_n \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix} z(k) + \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} u(k)$$

$$y(k) = \begin{pmatrix} b_1 & \dots & b_n \end{pmatrix} z(k)$$

Observability and detectability

The system

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) \end{aligned}$$

is observable if there is a finite k such that the knowledge of $u(0), \dots, u(k-1)$ and $y(0), \dots, y(k-1)$ is sufficient to determine the initial state $x(0)$ of the system.

Observability and detectability (cont'd)

Assume $u(k) = 0$, iterate the system equation:

$$\begin{aligned} y(0) &= Cx(0) \\ y(1) &= Cx(1) = C\Phi x(0) \\ &\vdots \\ y(n-1) &= C\Phi^{n-1}x(0) \end{aligned}$$

$$\begin{pmatrix} y(0) \\ y(1) \\ \vdots \\ y(n-1) \end{pmatrix} = \begin{pmatrix} C \\ C\Phi \\ \vdots \\ C\Phi^{n-1} \end{pmatrix} x(0)$$

Observability and detectability (cont'd)

The system is observable iff the *observability matrix*

$$W_o = \begin{pmatrix} C \\ C\Phi \\ \vdots \\ C\Phi^{n-1} \end{pmatrix}$$

has rank n.

A system is detectable if the unobservable states decay to the origin (the corresponding eigenvalues are stable).

Observability - Example

$$x(k+1) = \begin{pmatrix} 1.1 & -0.3 \\ 1 & 0 \end{pmatrix} x(k)$$

$$y(k) = \begin{pmatrix} 1 & -0.5 \end{pmatrix} x(k)$$

Observability matrix

$$W_o = \begin{pmatrix} C \\ C\Phi \end{pmatrix} = \begin{pmatrix} 1 & -0.5 \\ 0.6 & -0.3 \end{pmatrix}$$

The rank of W_o is 1

The Kalman decomposition

It is possible introduce coordinates that lead to this partitioning

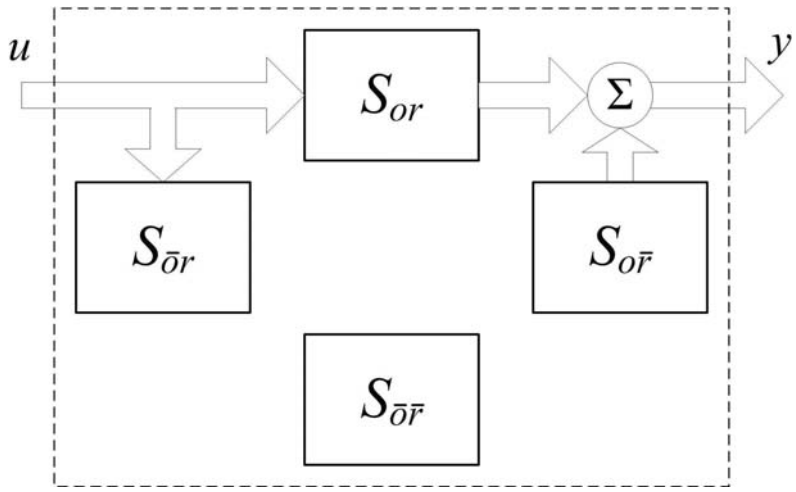
$$x(k+1) = \begin{pmatrix} \Phi_{11} & \Phi_{12} & 0 & 0 \\ 0 & \Phi_{22} & 0 & 0 \\ \Phi_{31} & \Phi_{32} & \Phi_{33} & \Phi_{34} \\ 0 & \Phi_{42} & 0 & \Phi_{44} \end{pmatrix} x(k) + \begin{pmatrix} \Gamma_1 \\ 0 \\ \Gamma_3 \\ 0 \end{pmatrix} u(k)$$

$$y(k) = \begin{pmatrix} C_1 & C_2 & 0 & 0 \end{pmatrix} x(k)$$

Pulse-transfer operator:

$$H(q) = C_1 \left(qI - \Phi_{11}\right)^{-1} \Gamma_1$$

The Kalman decomposition



Sampling $\rightarrow$ reachability and observability

- Φ and Γ depend on h.
- To get reachable discrete-time system it is necessary that the continuous-time system is controllable.
- May happen that reachability is lost for some h
- Sampled-data system may be unobservable even if the continuous-time system is observable (hidden oscillations).

Hidden oscillations

Inter-sample ripple

Two cases:

1. Oscillations seen in continuous-time output of an open-loop / closed-loop system.
   Loss of observability purely by sampling.

2. Oscillations seen in the control input, but not in the sampled output. Loss of observability by feedback, cancellation of poorly damped zeros by the controller.

Hidden oscillation in open-loop system

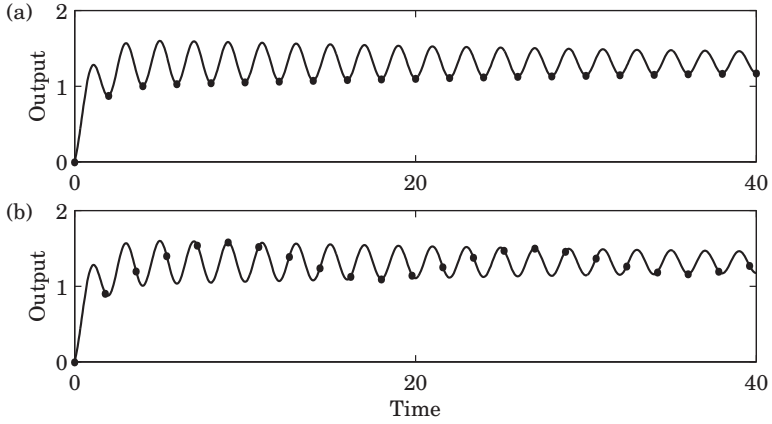
$$G(s) = \frac{1}{s+1} + \frac{\pi}{(s+0.02)^2 + \pi^2}$$

Sample with $h = 2, (\omega_s = \pi)$

$$H(z) = \frac{1-a}{z-a} + \frac{0.0125}{z-\alpha}$$

where $a = e^{-2}$ and $\alpha = e^{-0.04}$

Hidden oscillation in open-loop system



a) $h = 2$, b) $h = 1.8$

Controller-induced hidden oscillation

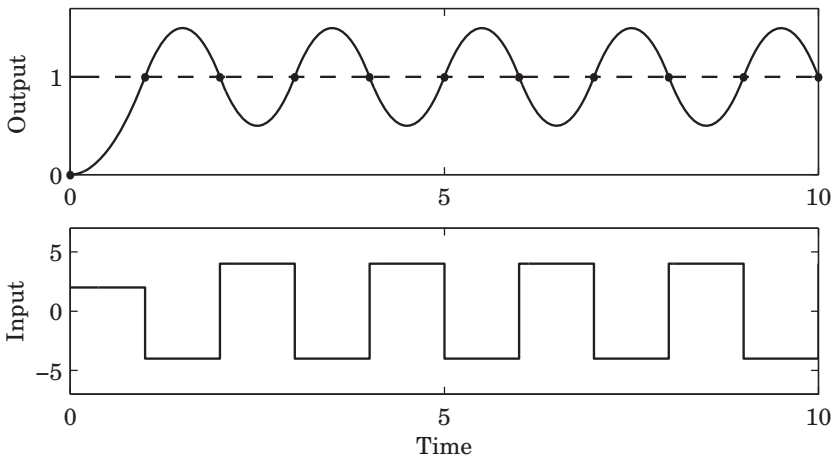
Consider a double integrator with the controller:

$$u(k) = \frac{2q}{q+1}u_c(k) - \frac{2(2q-1)}{q+1}y(k)$$

Closed-loop system

$$\begin{aligned} y(k) &= \frac{q(q+1)}{(q+1)(q^2-2q+1-(-2q+1))}u_c(k) \\ &= \frac{q(q+1)}{q^2(q+1)}u_c(k) = u_c(k-1) \end{aligned}$$

Controller-induced hidden oscillation



How to detect hidden oscillations?

- Modified z-transform or solve system equation for times between sampling points
- Simulation. Check continuous-time output and control signal

Importance of simulation

- Simpler, cheaper, safer to experiment with a model.
- If the system does not exist.
- Investigate influence of parameter variations.

Simulation can never replace a field test.

Beware of numerical aspects, choice of integration method!

Summary

- Stability
- Controllability and reachability
- Observability and detectability
- Simulation

Digital Control (SC42095)

Lecture 5, November 29, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

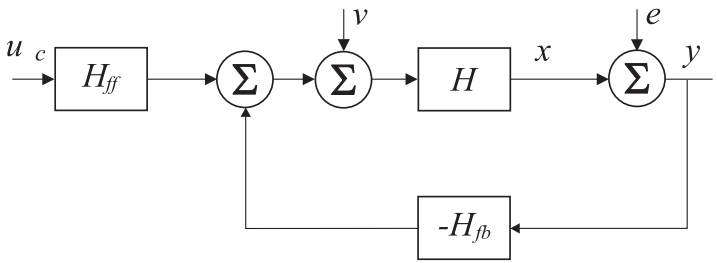
Lecture outline

- State-feedback control
- State estimation – observers
- Output-feedback control

Taxonomy of Controllers

- Presence of feedback: feedforward, feedback, 2-DOF
- Type of feedback: output, state
- Presence of dynamics: static, dynamic
- Dependence on time: fixed, adaptive
- Use of models: model-free, model-based

Control design



- Attenuation of load disturbances (regulation problem).
- Effect of measurement noise.
- Reference tracking (servo problem).
- Uncertain dynamics / process parameters (robust control). Section 3.3 in CCS book on sensitivity and robustness.

State feedback: problem formulation

- *Model*: $x(k+1) = \Phi x(k) + \Gamma u(k)$
- *Admissible controls*: linear controllers

$$u(k) = -Lx(k)$$

- *Disturbances*: widely spread pulses ($x(0) = x_0$)
- *Criterion*: $x(k) \to 0$ reasonably quickly with reasonable inputs u (choose closed-loop poles)

State feedback: problem formulation

- *Design parameters*: closed-loop poles, sampling interval
- *Evaluation*: compare $x(k)$ and $u(k)$ with specifications (trade-off between control magnitude and speed of response)

More complicated (and realistic) problems later ...

Example - double integrator

$$x(k+1) = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} x(k) + \begin{pmatrix} h^2/2 \\ h \end{pmatrix} u(k)$$

Linear state-feedback controller

$$u(k) = -Lx(k) = -l_1 x_1(k) - l_2 x_2(k)$$

Closed-loop system becomes

$$x(k+1) = (\Phi - \Gamma L)x(k) = \begin{pmatrix} 1 - l_1 h^2/2 & h - l_2 h^2/2 \\ -l_1 h & 1 - l_2 h \end{pmatrix} x(k)$$

Double integrator - cont'd

Desired characteristic equation: $z^2 + p_1 z + p_2 = 0$
Closed-loop characteristic equation:

$$z^2 + \left(\frac{l_1 h^2}{2} + l_2 h - 2\right) z + \left(\frac{l_1 h^2}{2} - l_2 h + 1\right) = 0$$

Linear equations for l_1 and l_2:

$$\begin{aligned} \tfrac{l_1 h^2}{2} + l_2 h - 2 &= p_1 \\ \tfrac{l_1 h^2}{2} - l_2 h + 1 &= p_2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} l_1 &= \tfrac{1}{h^2}(1 + p_1 + p_2) \\ l_2 &= \tfrac{1}{2h}(3 + p_1 - p_2) \end{aligned}$$

Unique solution (in this example), L depends on h

Solution in the general SISO case

Transform $x(k+1) = \Phi x(k) + \Gamma u(k)$ into the reachable canonical form: $z(k+1) = \tilde{\Phi} z(k) + \tilde{\Gamma} u(k)$ with:

$$\tilde{\Phi} = \begin{pmatrix} -a_1 & -a_2 & \dots & -a_{n-1} & -a_n \\ 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix} \quad \tilde{\Gamma} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Coefficients of the characteristic polynomial of $\tilde{\Phi} - \tilde{\Gamma}\tilde{L}$:

$$\begin{pmatrix} -a_1 - \tilde{l}_1 & -a_2 - \tilde{l}_2 & \dots & -a_n - \tilde{l}_n \end{pmatrix} \quad \Rightarrow \quad \tilde{L}$$

Transform back: $u = -\tilde{L}z = -\tilde{L}Tx = -Lx$

How to find the transformation matrix T

Recall the reachability matrix for a transformed system:

$$\begin{aligned} \tilde{W}_c &= \begin{pmatrix} \tilde{\Gamma} & \tilde{\Phi}\tilde{\Gamma} & \dots & \tilde{\Phi}^{n-1}\tilde{\Gamma} \end{pmatrix} \\ &= \begin{pmatrix} T\Gamma & T\Phi T^{-1}T\Gamma & \dots & T\Phi^{n-1}T^{-1}T\Gamma \end{pmatrix} \\ &= TW_c \end{aligned}$$

Can be solved for T if the system is reachable: $T = \tilde{W}_c W_c^{-1}$

Ackermann's formula

After some derivations (book on page 127) ...

$$L = \begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix} W_c^{-1} P(\Phi)$$

where $P(\Phi)$ is the desired characteristic polynomial (in Φ!)

in Matlab:

- `L = acker(Phi,Gamma,Po)` (SISO, numerical problems)
- `L = place(Phi,Gamma,Po)` (MISO, more robust)

Example I - double integrator

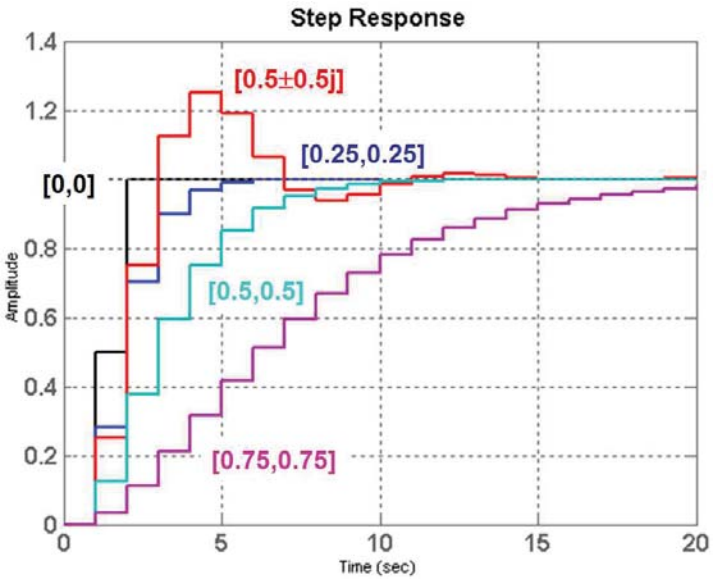
Desired characteristic polynomial: $P(z) = z^2 + p_1 z + p_2$
Ackermann's formula: $L = \begin{pmatrix} 0 & 1 \end{pmatrix} W_c^{-1} P(\Phi)$

$$W_c = \begin{pmatrix} \Gamma & \Phi\Gamma \end{pmatrix} = \begin{pmatrix} h^2/2 & 3h^2/2 \\ h & h \end{pmatrix}, \quad W_c^{-1} = \begin{pmatrix} -1/h^2 & 1.5/h \\ 1/h^2 & -0.5/h \end{pmatrix}$$

$$P(\Phi) = \Phi^2 + p_1\Phi + p_2 I = \begin{pmatrix} 1 + p_1 + p_2 & 2h + p_1 h \\ 0 & 1 + p_1 + p_2 \end{pmatrix}$$

$$\begin{aligned} L = \begin{pmatrix} 0 & 1 \end{pmatrix} W_c^{-1} P(\Phi) &= \begin{pmatrix} 1/h^2 & -0.5/h \end{pmatrix} P(\Phi) \\ &= \begin{pmatrix} \dfrac{1 + p_1 + p_2}{h^2} & \dfrac{3 + p_1 - p_2}{2h} \end{pmatrix} \end{aligned}$$

Example I - double integrator



Example II – an unreachable system

$$x(k+1) = \begin{pmatrix} 0.5 & 1 \\ 0 & 0.3 \end{pmatrix} x(k) + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u(k)$$

not reachable because $\det W_c = \det \begin{pmatrix} 1 & 0.5 \\ 0 & 0 \end{pmatrix} = 0$

Control law $u(k) = -l_1 x_1(k) - l_2 x_2(k)$ gives:

$$\det(zI - \Phi + \Gamma L) = 0$$

$$(z - 0.5 + l_1)(z - 0.3) = 0$$

How to place the poles?

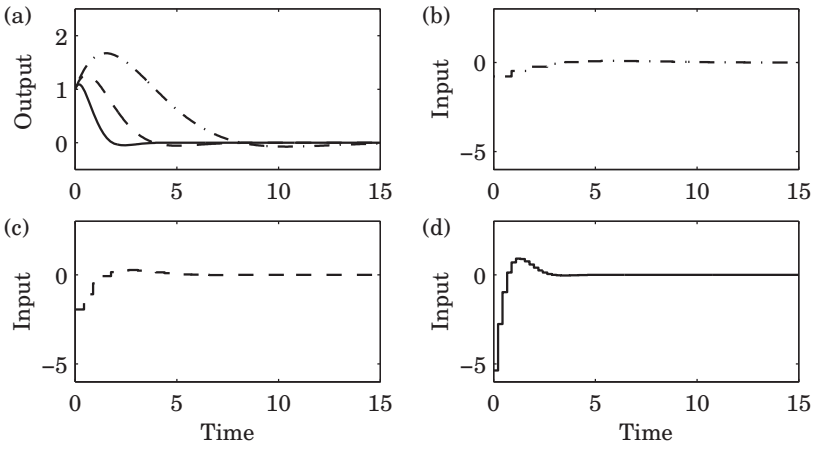
Using the continuous-time counterpart (2nd order)

$$s^2 + 2\zeta\omega s + \omega^2$$

gives $z^2 + p_1 z + p_2$ with

$$p_1 = -2e^{-\zeta\omega h} \cos\left(\omega h\sqrt{1-\zeta^2}\right)$$
$$p_2 = e^{-2\zeta\omega h}$$

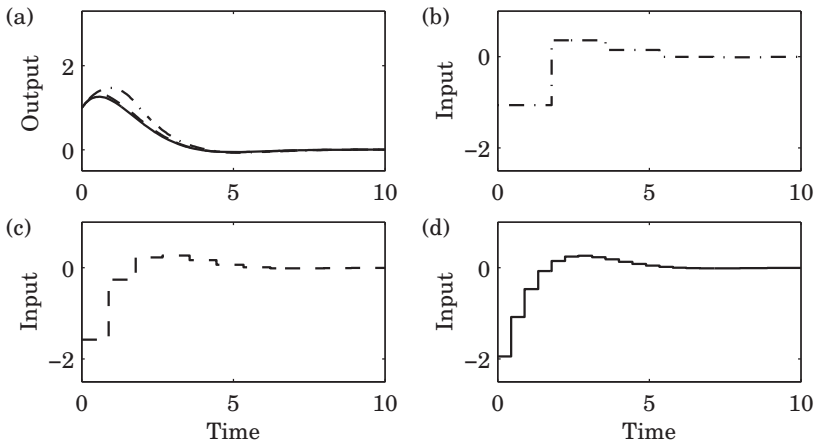
Response for different ω



$x(0) = [1 \quad 1]^T$ b) $\omega = 0.5$, c) $\omega = 1$, d) $\omega = 2$

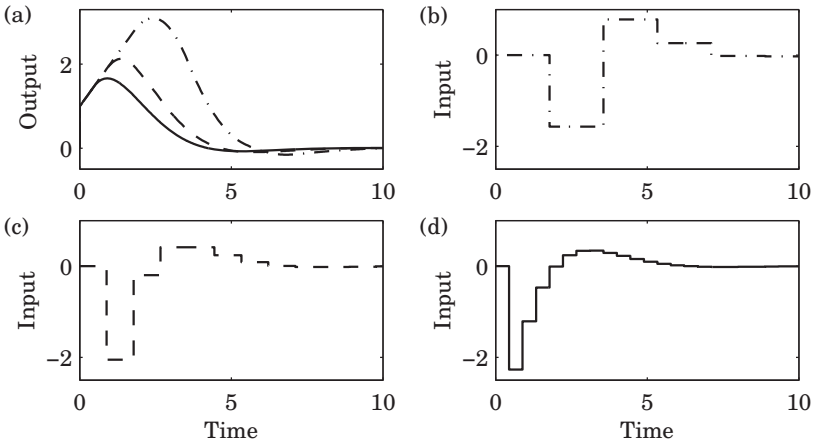
Response for different sampling times

Change of the number of samples within $2\pi/\omega_0$



$x(0) = [1 \quad 1]^T$ b) $N = 5$, c) $N = 10$, d) $N = 20$

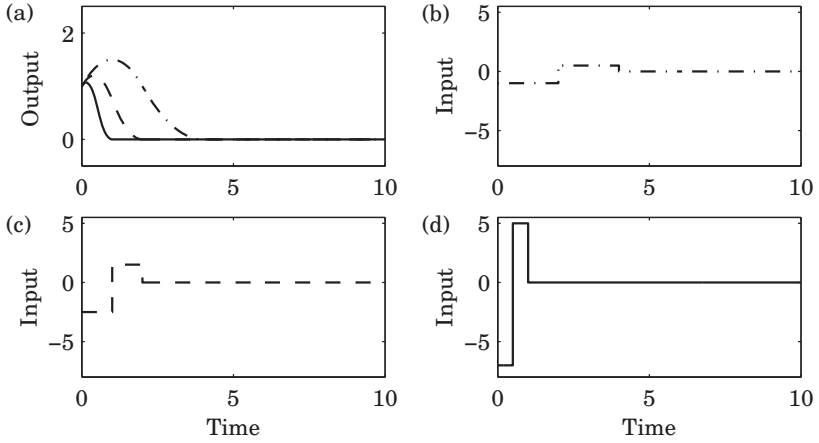
Another initial condition



$x^T(0^+) = [1 \quad 1]$ b) $N = 5$, c) $N = 10$, d) $N = 20$

Deadbeat control

Choose $P(z) = z^n$ (the remaining design parameter is h)



b) $h = 2$, c) $h = 1$, d) $h = 0.5$

State estimation: Observers

$$x(k+1) = \Phi x(k) + \Gamma u(k)$$
$$y(k) = Cx(k)$$

Input and output available, reconstruct the state:

- Direct calculation (requires differentiation)
- Luenberger observer (model-based)
- Kalman filter (optimal in presence of noise)

the terms "observer", "estimator", "filter" are in this context used synonymously

Direct calculation of state variables

Starting from initial state $x(k-n+1)$ and applying the control sequence $U(k-1) = (u(k-n+1) \;\; u(k-n+2) \;\; \cdots \;\; u(k-1))^T$, the resulting output sequence $Y(k) = (y(k-n+1) \;\; y(k-n+2) \;\; \cdots \;\; y(k))^T$ can be expressed as

$$Y(k) = W_o x(k-n+1) + W_u U(k-1)$$

$$W_o = \begin{pmatrix} C \\ C\Phi \\ C\Phi^2 \\ \vdots \\ C\Phi^{n-1} \end{pmatrix} \qquad W_u = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ C\Gamma & 0 & \cdots & 0 \\ C\Phi\Gamma & C\Gamma & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C\Phi^{n-2}\Gamma & C\Phi^{n-3}\Gamma & \cdots & C\Gamma \end{pmatrix}$$

Direct calculation of state variables

$$Y(k) = W_o x(k-n+1) + W_u U(k-1)$$

If the states are observable then

$$x(k-n+1) = W_o^{-1} Y(k) - W_o^{-1} W_u U(k-1)$$

and propagating the past states leads to

$$\begin{aligned} x(k) &= \Phi^{n-1} x(k-n+1) + \Phi^{n-2}\Gamma u(k-n+1) + \cdots + \Gamma u(k-1) \\ x(k) &= A_y Y(k) + B_u U(k-1) \end{aligned}$$

where

$$A_y = \Phi^{n-1} W_o^{-1} \qquad B_u = \begin{pmatrix} \Phi^{n-2}\Gamma & \Phi^{n-3}\Gamma & \cdots & \Gamma \end{pmatrix} - \Phi^{n-1} W_o^{-1} W_u$$

Direct calculation of state variables – example

$$\Phi = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}, \quad \Gamma = \begin{pmatrix} \frac{h^2}{2} \\ h \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

Observer

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

requires differentiation

State reconstruction based on a model

Consider the process:

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= C x(k) \end{aligned}$$

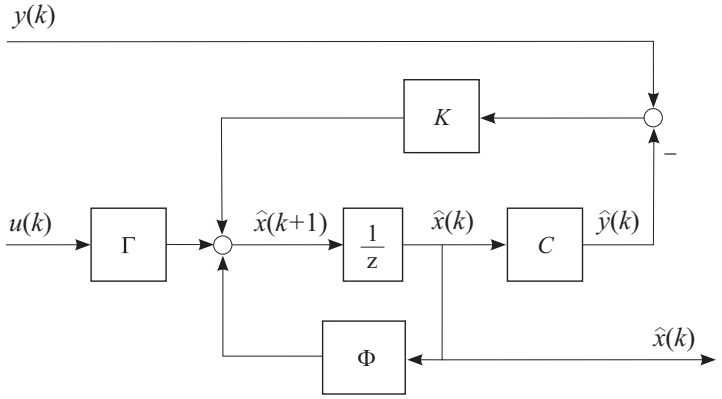
and its model:

$$\begin{aligned} \hat{x}(k+1) &= \Phi \hat{x}(k) + \Gamma u(k) \\ \hat{y}(k) &= C \hat{x}(k) \end{aligned}$$

Introduce "feedback" from measured $y(k)$:

$$\hat{x}(k+1) = \Phi \hat{x}(k) + \Gamma u(k) + K[y(k) - C\hat{x}(k)]$$

Observer



$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + K\Big[\, y(k) - \hat{y}(k) \,\Big]$$
$$\hat{y}(k) = C\hat{x}(k)$$

Design of observer gain

$$x(k+1) = \Phi x(k) + \Gamma u(k)$$
$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + K[y(k) - C\hat{x}(k)]$$

Estimation error $e = x - \hat{x}$

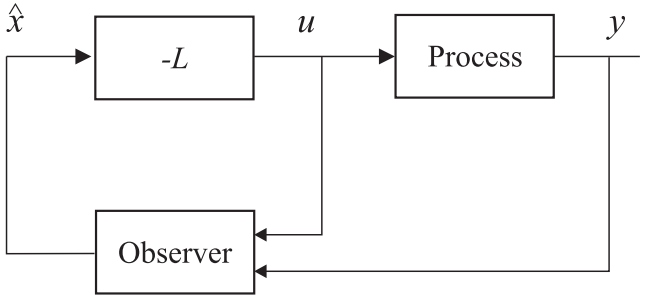
$$e(k+1) = \Phi e(k) - KCe(k) = [\Phi - KC]e(k)$$

- Choose K such that $e(k)$ goes to zero
- Any eigenvalues possible, provided W_o has full rank
- Dual problem to state-feedback design (transpose)

Various types of observers

- Deadbeat observer, see deadbeat controller.
- Observer without delay (estimates $\hat{x}(k \mid k)$ instead of $\hat{x}(k \mid k-1)$).
- Reduced-order observer, MIMO.
- Kalman filter (stochastic disturbances).

Output feedback (observer + state feedback)



$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + K\Big(y(k) - C\hat{x}(k)\Big)$$
$$u(k) = -L\hat{x}(k)$$

Output feedback controller

$$\begin{aligned}\hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma u(k) + K(y(k) - C\hat{x}(k)) \\ u(k) &= -L\hat{x}(k)\end{aligned}$$

$$\boxed{\begin{aligned}\hat{x}(k+1) &= (\Phi - KC - \Gamma L)\hat{x}(k) + Ky(k) \\ u(k) &= -L\hat{x}(k)\end{aligned}}$$

Transfer function of the output-feedback controller:

$$G_r(z) = \frac{U(z)}{Y(z)} = -L(zI - \Phi + \Gamma L + KC)^{-1}K$$

Poles of the closed-loop system

$$\begin{aligned}x(k+1) &= \Phi x(k) + \Gamma u(k) \\ e(k+1) &= x(k+1) - \hat{x}(k+1) = (\Phi - KC)e(k) \\ u(k) &= -L(x(k) - e(k))\end{aligned}$$

Eliminate $u(k)$:

$$\begin{pmatrix} x(k+1) \\ e(k+1) \end{pmatrix} = \begin{pmatrix} \Phi - \Gamma L & \Gamma L \\ 0 & \Phi - KC \end{pmatrix} \begin{pmatrix} x(k) \\ e(k) \end{pmatrix}$$

Process poles: $A_r(z) = \det(zI - \Phi + \Gamma L)$
Observer poles: $A_o(z) = \det(zI - \Phi + KC)$

Separation principle

Summary

- State-feedback control
- Ackermann's formula
- Choosing closed-loop poles
- Deadbeat control
- Observers and output-feedback control

Digital Control (SC42095)

Lecture 6, December 4, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Compensation for disturbances
- Servo problem (tracking)
- Actuator saturation
- Input–output design

More general disturbances

System:

$$\frac{dx}{dt} = Ax + Bu + v \qquad y = Cx$$

Describe disturbance as a dynamic system:

$$\frac{dw}{dt} = A_w w, \quad v = C_w w$$

Combine the two into an augmented system:

$$\frac{d}{dt}\begin{pmatrix} x \\ w \end{pmatrix} = \begin{pmatrix} A & C_w \\ 0 & A_w \end{pmatrix}\begin{pmatrix} x \\ w \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} u$$

$$y = \begin{pmatrix} C & 0 \end{pmatrix}\begin{pmatrix} x & w \end{pmatrix}^T$$

More general disturbances – cont'd

$$\begin{pmatrix} x(k+1) \\ w(k+1) \end{pmatrix} = \begin{pmatrix} \Phi & \Phi_{xw} \\ 0 & \Phi_w \end{pmatrix}\begin{pmatrix} x(k) \\ w(k) \end{pmatrix} + \begin{pmatrix} \Gamma \\ 0 \end{pmatrix} u(k)$$

State feedback:

$$u(k) = -Lx(k) - L_w w(k)$$

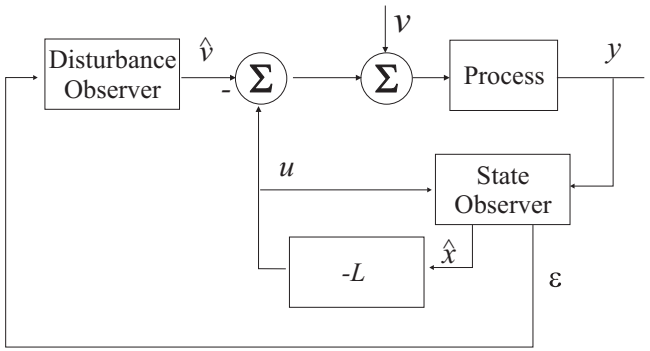
Closed-loop system

$$x(k+1) = (\Phi - \Gamma L)x(k) + \underbrace{(\Phi_{xw} - \Gamma L_w)}_{\approx 0?} w(k)$$

$$w(k+1) = \Phi_w w(k)$$

w uncontrollable from u, not directly measurable $\longrightarrow$ use observer

Disturbance observer



$$\begin{aligned}\hat{x}(k+1) &= \Phi\hat{x}(k) + \Phi_{xw}\hat{w}(k) + \Gamma u(k) + K\varepsilon(k)\\ \hat{w}(k+1) &= \Phi_w\hat{w}(k) + K_w\varepsilon(k)\\ \hat{v}(k) &= C_w\hat{w}(k), \quad \text{with} \quad \varepsilon(k) = y(k) - C\hat{x}(k)\end{aligned}$$

Example – load disturbance at process input

$$w(k+1) = w(k) \qquad v(k) = w(k)$$

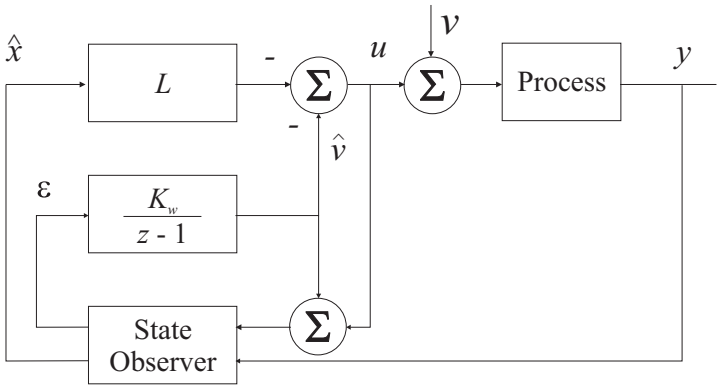
$$u(k) = -L\hat{x}(k) - \hat{w}(k) = -L\hat{x}(k) - \hat{v}(k)$$

$$\begin{aligned}\hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma(\hat{v}(k) + u(k)) + K(y(k) - C\hat{x}(k))\\ \hat{v}(k+1) &= \hat{v}(k) + K_w(y(k) - C\hat{x}(k))\end{aligned}$$

Here, the disturbance observer is an integrator.

Integrator in the controller

$$\begin{aligned}\hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma(\hat{v}(k) + u(k)) + K(y(k) - C\hat{x}(k))\\ \hat{v}(k+1) &= \hat{v}(k) + K_w(y(k) - C\hat{x}(k))\end{aligned}$$



Integrator – another approach

- Include integrator in an outer loop
- Analogous to PID control
- Extended system:

$$\begin{pmatrix} x(k+1) \\ x_i(k+1) \end{pmatrix} = \begin{pmatrix} \Phi & 0 \\ C & I \end{pmatrix} \begin{pmatrix} x(k) \\ x_i(k) \end{pmatrix} + \begin{pmatrix} \Gamma \\ 0 \end{pmatrix} u(k)$$

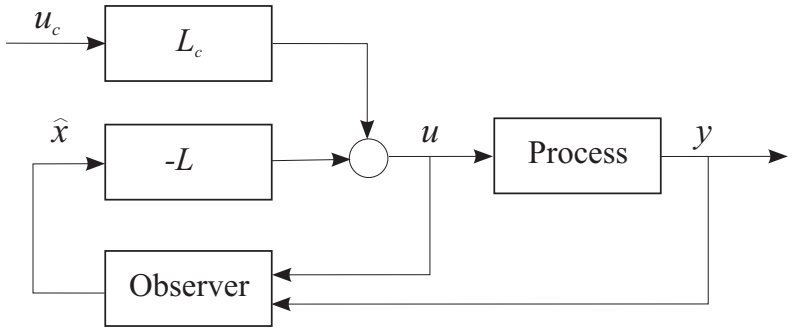
- State feedback:

$$u(k) = -\begin{pmatrix} L & L_i \end{pmatrix} \begin{pmatrix} x(k) \\ x_i(k) \end{pmatrix}$$

Servo case: simple approach

Goal: respond to a reference signal in a specified way.

Replace $u(k) = -L\hat{x}(k)$ by: $u(k) = -L\hat{x}(k) + L_c u_c(k)$



Servo case: simple approach (cont'd)

Closed-loop system:

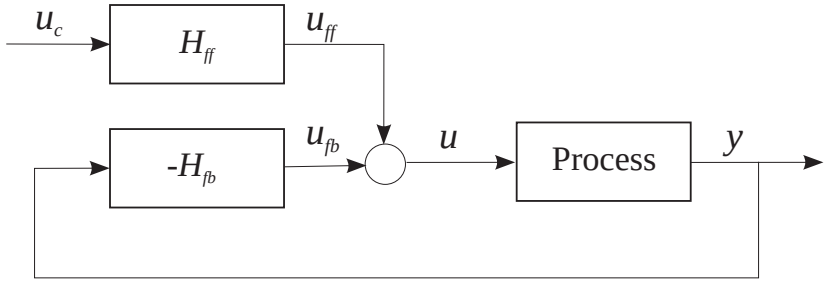
$$\begin{aligned} x(k+1) &= (\Phi - \Gamma L)x(k) + \Gamma L e(k) + \Gamma L_c u_c(k) \\ e(k+1) &= (\Phi - KC)e(k) \\ y(k) &= Cx(k) \end{aligned}$$

Pulse-transfer function from u_c to y:

$$H_{cl}(z) = C(zI - \Phi + \Gamma L)^{-1}\Gamma L_c = L_c \frac{B(z)}{A_m(z)}$$

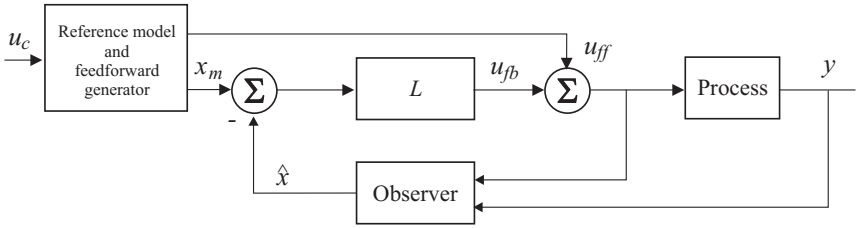
L_c for desired DC gain: $L_c = \dfrac{A_m(1)}{B(1)} \cdot H_{cl}(1)$

Servo case: two degrees of freedom



- H_{fb} is designed to obtain closed-loop system that is insensitive to disturbances, measurement noise, process uncertainties.
- H_{ff} is designed to obtain desired servo properties.

Servo case: model and feedforward



Controller

$$u(k) = \underbrace{L(x_m(k) - \hat{x}(k))}_{u_{fb}(k)} + u_{ff}(k)$$

Feedforward signal

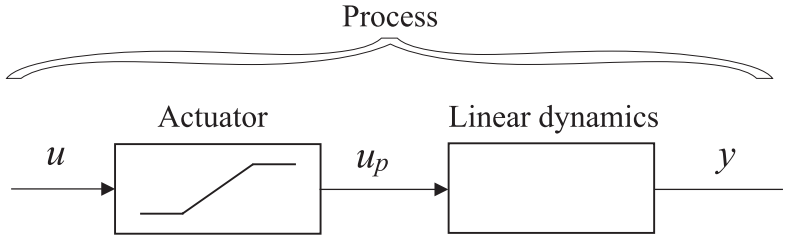
$$u_{ff}(k) = \frac{H_m(q)}{H(q)} u_c(k)$$

Reference trajectories for states

$$x_m(k+1) = \Phi_m x_m(k) + \Gamma_m u_c(k)$$
$$y_m(k) = C_m x_m(k)$$

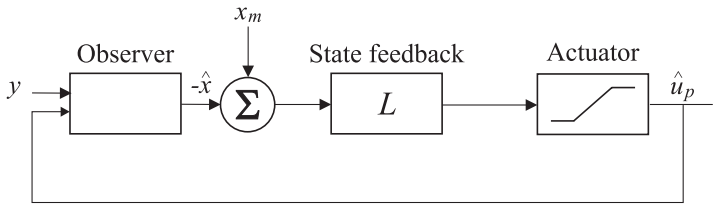
the same state coordinates as the process

Nonlinear actuators



Feedback loop broken if control saturated
Is the controller stable?
Controller states may wind-up

Tracking scheme

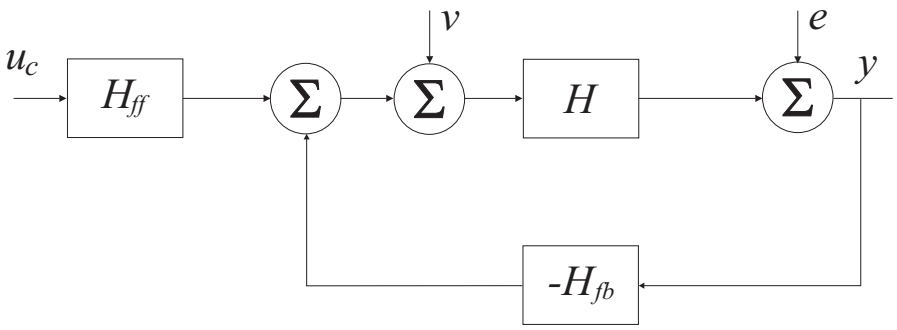


Measure or estimate the actual input u_p

$$\hat{x}(k+1) = (\Phi - KC)\hat{x}(k) + Ky(k) + \Gamma\hat{u}_p(k)$$
$$\hat{u}_p(k) = \mathsf{sat}(u(k))$$

A similar mechanism can be used for any type of controller (PID later)

Input/output design



Design H_{fb} and H_{ff} directly as rational transfer functions

Input/output design: formulation

- Process (SISO, including hold circuit, actuator, sensor, antialiasing filter):

$$H(z) = \frac{B(z)}{A(z)}$$

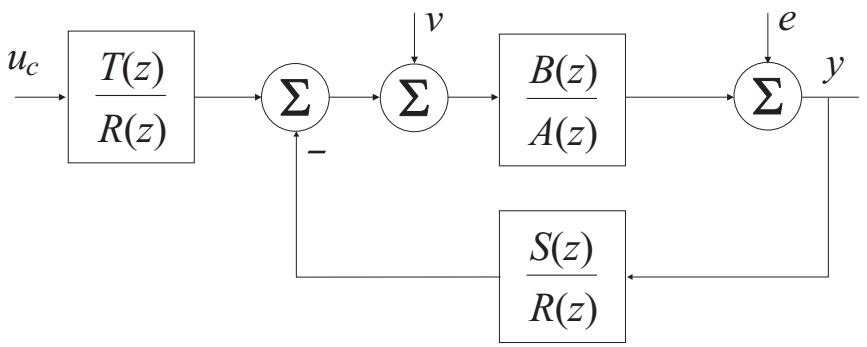
- RST Controller:

$$R(q)u(k) = T(q)u_c(k) - S(q)y(k)$$

Causality implies: $\deg R \geq \deg T, \quad \deg R \geq \deg S$

$$H_{fb}(z) = S(z)/R(z), \quad H_{ff}(z) = T(z)/R(z)$$

RST controller



Closed loop

$$\begin{aligned} A(q)y(k) &= B(q)u(k) \\ R(q)u(k) &= T(q)u_c(k) - S(q)y(k) \end{aligned}$$

Closed loop system:

$$y = \frac{BT}{AR + BS}u_c$$

Desired input-output relation:

$$\frac{BT}{AR + BS} = \frac{BT}{A_{cl}} = \frac{BT}{A_c A_o} = \frac{t_o B}{A_c}$$

I/O pole-placement design

1. Find $R(z)$ and $S(z)$ ($\deg S(z) \leq \deg R(z)$) satisfying

$$A(z)R(z) + B(z)S(z) = A_{cl}(z)$$

2. Factor $A_{cl}(z)$ as $A_{cl}(z) = A_c(z)A_o(z)$ with $\deg A_o(z) \leq \deg R(z)$ and choose

$$T(z) = t_o A_o(z)$$

where $t_o = A_c(1)/B(1)$ is chosen for desired static gain.

Diophantine equation

$$A(z)X(z) + B(z)Y(z) = C(z)$$

(Diophantus $\approx$ A.D. 300, also called Bezout identity)

- One equation, two unknowns.
- When does the Diophantine equation have a (unique) solution?
- Analogy: algebraic example.

Simple algebraic example

Assume x, y integers and

$$3x + 2y = 5$$

Some solutions are

$x:$	-5	-3	-1	1	3	5	7
$y:$	10	7	4	1	-2	-5	-8

General solution:

$$\begin{aligned} x &= x_0 + 2n \\ y &= y_0 - 3n \end{aligned} \qquad n \text{ integer}$$

Unique solution if $0 \leq x < 2$ or $0 \leq y < 3$

Solution of the Diophantine equation

$$A(z)X(z) + B(z)Y(z) = C(z)$$

- Solution exists if and only if the greatest common factor of A and B is also a factor in C.
- Many solutions. If X_0 and Y_0 is a solution then for arbitrary Q

$$\begin{aligned} X &= X_0 + QB \\ Y &= Y_0 - QA \end{aligned}$$

  is also a solution
- Uniqueness if $\deg X < \deg B$ or $\deg Y < \deg A$

Euclid's algorithm, Sylvester matrix

Summary

- Compensate disturbances – integrator (see also PID)
- Servo control (reference following)
- Actuator saturation
- Input–output design (transfer functions)

Digital Control (SC42095)

Lecture 7, December 6, 2017

Tamás Keviczky

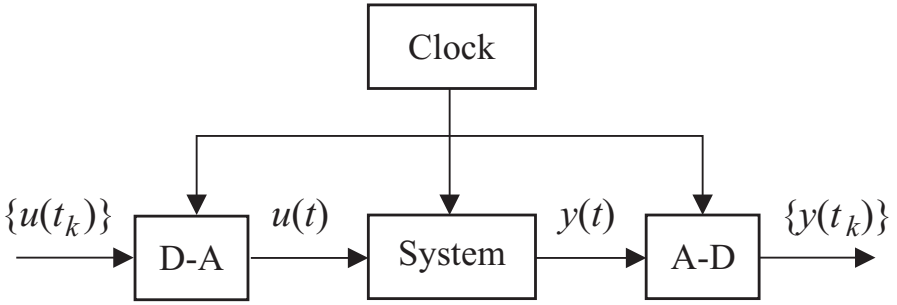
Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

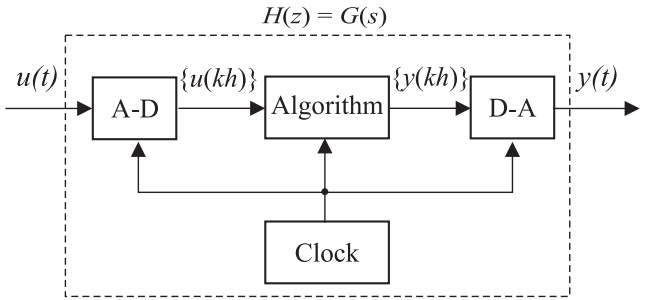
Lecture outline

- Redesign of continuous-time controllers
- First-order holds
- PID control
- Implementation issues

So far: system from computer's viewpoint



Computer implementation of analog controllers



$G(s)$ is designed by using continuous-time techniques

Want to get

$$\text{A/D} + \text{Algorithm} + \text{D/A} \approx G(s)$$

Approximation of derivatives

Forward difference (Euler method)

$$px(t) = \frac{dx(t)}{dt} \approx \frac{x(t+h) - x(t)}{h} = \frac{q-1}{h}x(t)$$

Backward difference

$$px(t) = \frac{dx(t)}{dt} \approx \frac{x(t) - x(t-h)}{h} = \frac{q-1}{qh}x(t)$$

Trapezoidal method (Tustin, bilinear)

$$\frac{dx(t)}{dt} \approx \frac{\dot{x}(t+h) + \dot{x}(t)}{2} \quad \Rightarrow \quad px(t) = \frac{2}{h} \cdot \frac{q-1}{q+1}x(t)$$

Approximation of transfer functions

$$H(z) = G(s)$$

with the substitutions:

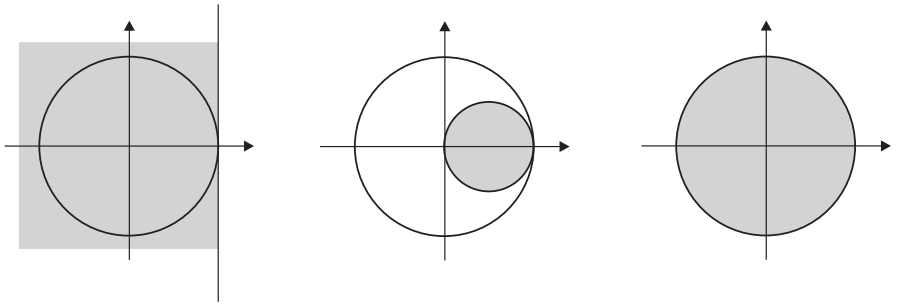
$s = \frac{z-1}{h}$ (forward difference or Euler method)

$s = \frac{z-1}{zh}$ (backward difference)

$s = \frac{2}{h}\frac{z-1}{z+1}$ (Tustin or bilinear approximation)

Note: The frequency scale gets distorted with these approximations (warping effect)!
See page 295 of CCS book about frequency prewarping.

Stability of the approximations



Forward differences | Backward differences | Tustin

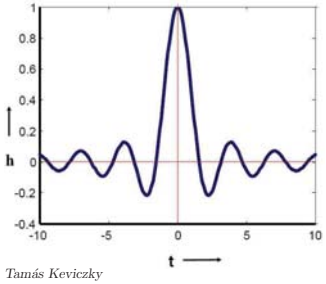
∀ unstable CT → unstable DT | ∀ stable CT → stable DT | ∀ unstable CT → unstable DT

∃ stable CT → unstable DT | ∃ unstable CT → stable DT | ∀ stable CT → stable DT

Signal reconstruction

Shannon's sampling theorem:
If a signal $f(t)$ has frequency content $\omega < \omega_N$, where the ω_N Nyquist frequency is half of the sampling frequency ω_s, then it is uniquely determined by its sample points.

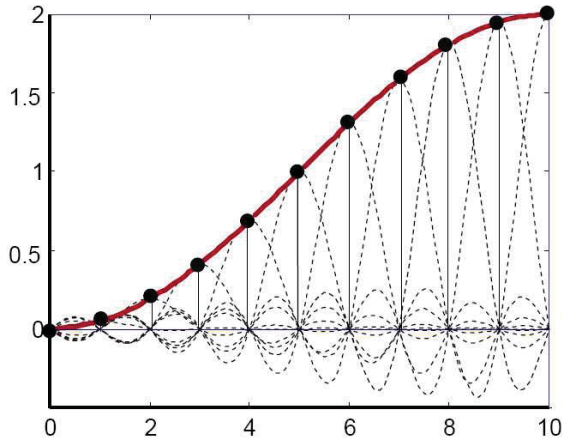
$$f(t) = \sum_{-\infty}^{\infty} f(k)\frac{\sin(\omega_s(t-kh)/2)}{\omega_s(t-kh)/2}$$



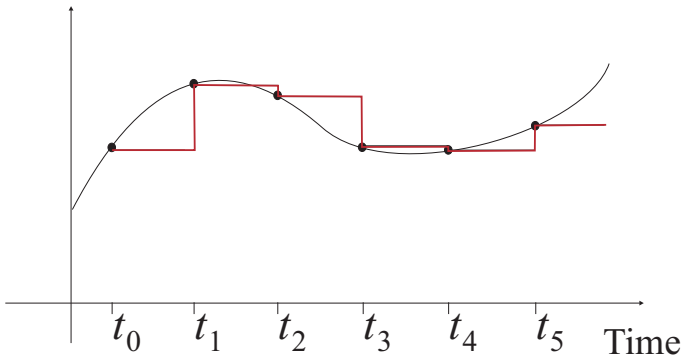
Shannon reconstruction with

$h(t) = \frac{\sin(\omega_s t/2)}{\omega_s t/2}$ is not causal!

Shannon reconstruction (exact)



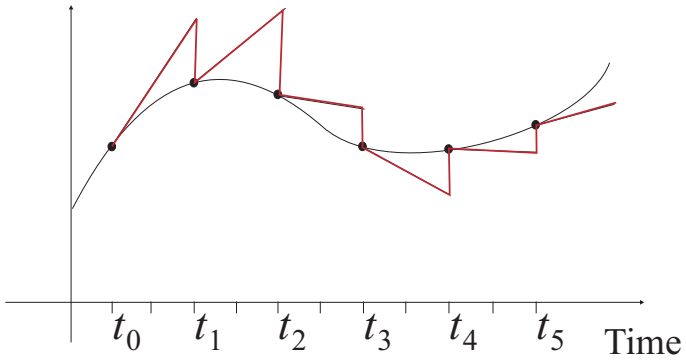
Zero-order hold



$$u(t) = u(t_k), \quad t_k \le t < t_{k+1}$$

$$\frac{1 - e^{-hs}}{s}$$

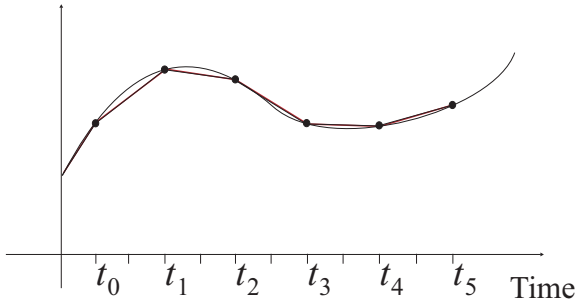
First-order hold



$$u(t) = u(t_k) + \frac{t - t_k}{t_k - t_{k-1}}\Big(u(t_k) - u(t_{k-1})\Big), \quad t_k \le t < t_{k+1}$$

$$\frac{s+1}{s^2}(1 - 2e^{-hs} + e^{-2hs})$$

Predictive first order hold



$$u(t) = u(t_k) + \frac{t - t_k}{t_k - t_{k-1}}\Big(u(t_{k+1}) - u(t_k)\Big), \quad t_k \le t < t_{k+1}$$

Input linear between samples, rather than constant (different sampling formulas).

PID control

The "textbook" version

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

Common modifications:

- Filter on the derivative

$$sT_d \approx \frac{sT_d}{1 + sT_d/N}$$

- Derivative of $-y$ not of $e = u_c - y$
- Only fraction of u_c in the P-term (reduce overshoot)

More realistic PID controller

$$U(s) = K\left(bU_c(s) - Y(s) + \frac{1}{sT_i}(U_c(s) - Y(s)) - \frac{sT_d}{1 + sT_d/N}Y(s)\right)$$

Other modifications:

- high-frequency roll-off (extra filter)
- nonlinearities, e.g., $Ke|e|$
- anti-windup (later)

Discrete-time PID

P-term: $P(k) = K(bu_c(k) - y(k))$

I-term: $I(k+1) = I(k) + \frac{Kh}{T_i}e(k)$

D-term: $D(k) = \frac{T_d}{T_d+Nh}D(k-1) - \frac{KT_dN}{T_d+Nh}(y(k) - y(k-1))$

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

special case of RST: $R(q)u(k) = T(q)u_c(k) - S(q)y(k)$

table for $R(q)$, $S(q)$ and $T(q)$ (page 309)

Discrete-time PID (cont'd)

Simple position form:

$$u(k) = K\left(1 + \frac{h}{T_i}\frac{1}{q-1} + \frac{T_d}{h}\frac{q-1}{q}\right)e(k)$$

$$\Downarrow$$

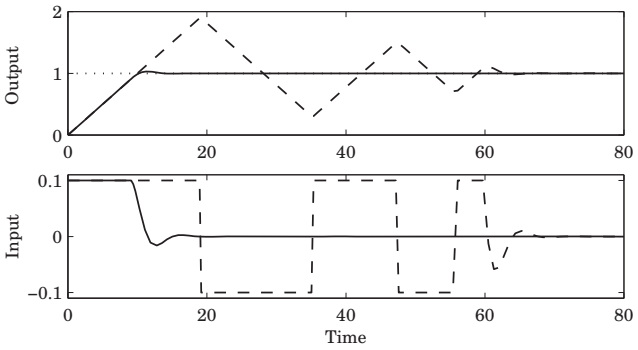
$$\Delta u(k) = u(k) - u(k-1)$$

$$\Downarrow$$

Simple velocity form:

$$\Delta u(k) = K\left(\frac{q-1}{q} + \frac{h}{T_i}\frac{1}{q-1} + \frac{T_d}{h}\frac{q-1}{q}\right)e(k)$$

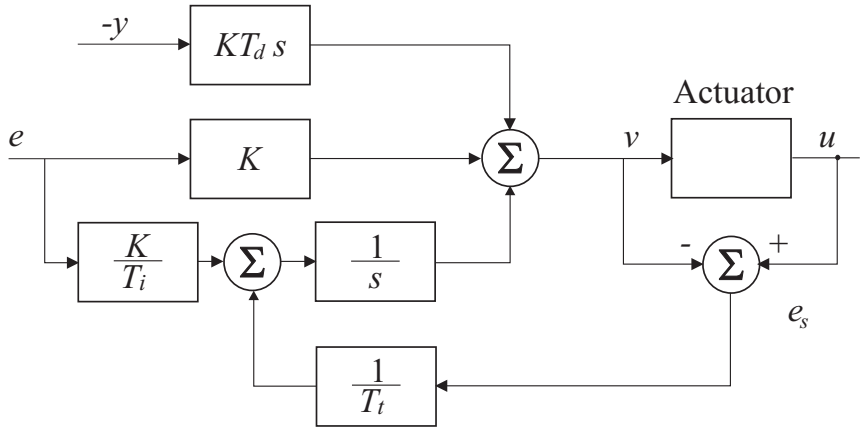
Integrator windup (due to actuator saturation)



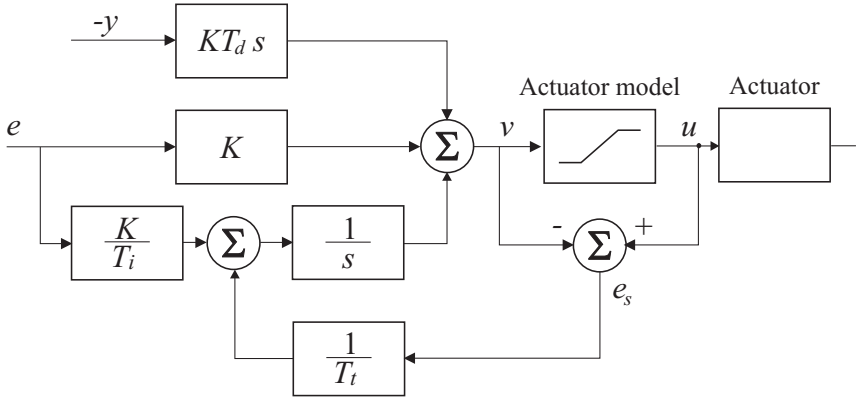
Solution: reset the integrator

- stop integrating (when actuator saturates)
- tracking schemes

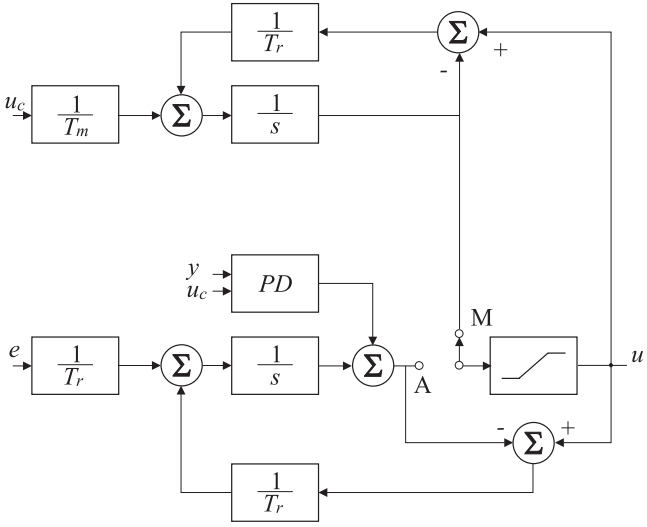
Tracking scheme for anti-windup



Tracking scheme for anti-windup



Bumpless transfer manual–automatic



Bumpless parameter change

Use:

$$x_I = \int^t \frac{K}{T_i} e(s)ds$$

instead of

$$x_I = \frac{K}{T_i} \int^t e(s)ds$$

Tuning of PID controllers

K, T_i, T_d and other parameters: N, T_t, u_{low}, u_{high}, h, b

- $N = 10$ typically
- T_t equal to T_i or 0.1–0.5 times T_i
- u_{low} and u_{high} close to true saturation values
- Sampling period h

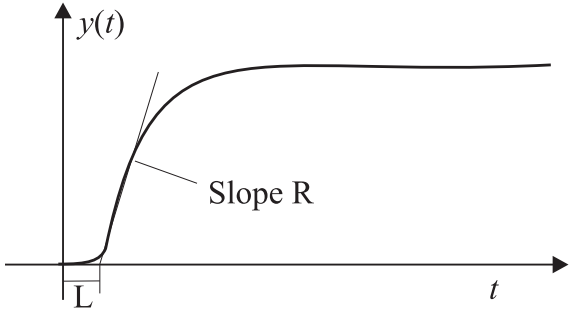
$$\text{For PI} : \frac{h}{T_i} \approx 0.1 - 0.3$$

$$\text{For PID} : \frac{hN}{T_d} \approx 0.2 - 0.6$$

- $b < 1$ to decrease overshoot after setpoint changes

Ziegler–Nichols tuning rules

1. Step-response method



$a = RL$, $a \longrightarrow K$, T_i, T_d (page 315)

Ziegler–Nichols tuning rules – cont'd

2. Ultimate-sensitivity method

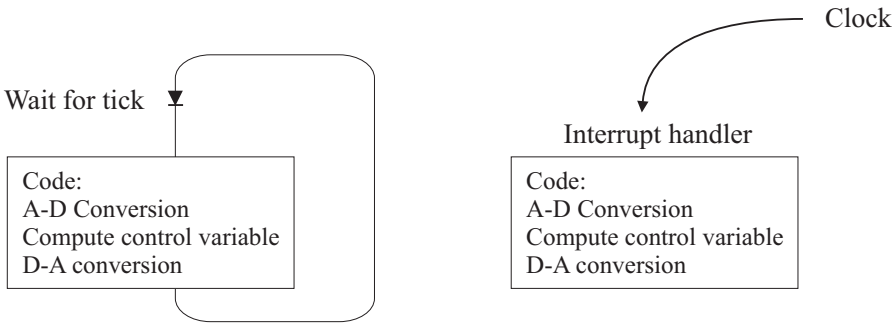
Use a P controller and increase the gain $\Rightarrow K_u$, T_u

K_u, $T_u \longrightarrow K$, T_i, T_d (page 316)

Use with care (poor damping, $\zeta \approx 0.2$)

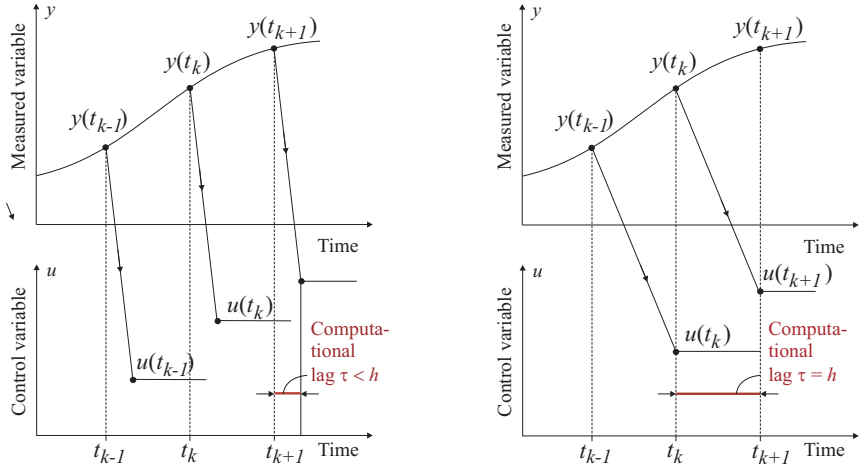
Many other tuning methods (including pole placement)

Implementation



Real-time operating systems, multi-tasking

Computational delay



Keep it constant, include in the process model.

Numerical aspects

- Word-length, computer, A/D and D/A converters
- Fixed or floating point computations (IEEE standard)

  example: $(100 \quad 1 \quad 100)(100 \quad 1 \quad -100)^T$
- Use higher precision for internal calculations
- Influence of noise and quantization
- Choice of realization (different sensitivity, see on extra slides)

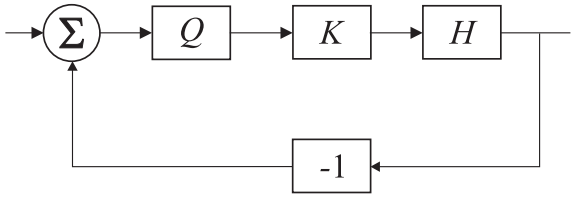
Effects of roundoff and quantization

- Nonlinear phenomena

$$Q(a+b) \neq Q(a) + Q(b)$$

- Limit cycles and/or bias
- Analysis tools
  - Nonlinear analysis
  - Describing function approximation
  - Model quantization as stochastic processes

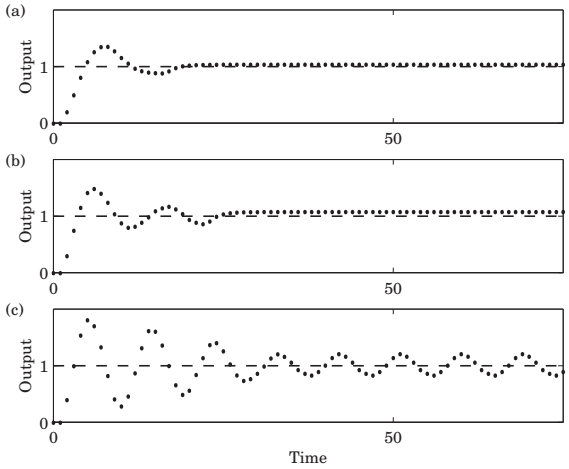
Effect of roundoff



$$H(z) = \frac{0.25}{(z-1)(z-0.5)}$$

Without quantization: asymptotically stable for $K < 2$

With quantization



a) $K = 0.8$, b) $K = 1.2$, c) $K = 1.6$

Summary

- Redesign of continuous-time controllers
- First-order holds
- PID control
- Implementation issues

Realization of Digital Controllers

Controller

$$y_k = H\left(q^{-1}\right) = \frac{b_0 + b_1 q^{-1} + \cdots + b_m q^{-m}}{1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_n\, q^{-n}} u_k$$

- Direct form
- Companion form
- Series form
- Parallel form
- δ-operator form

TU Delft

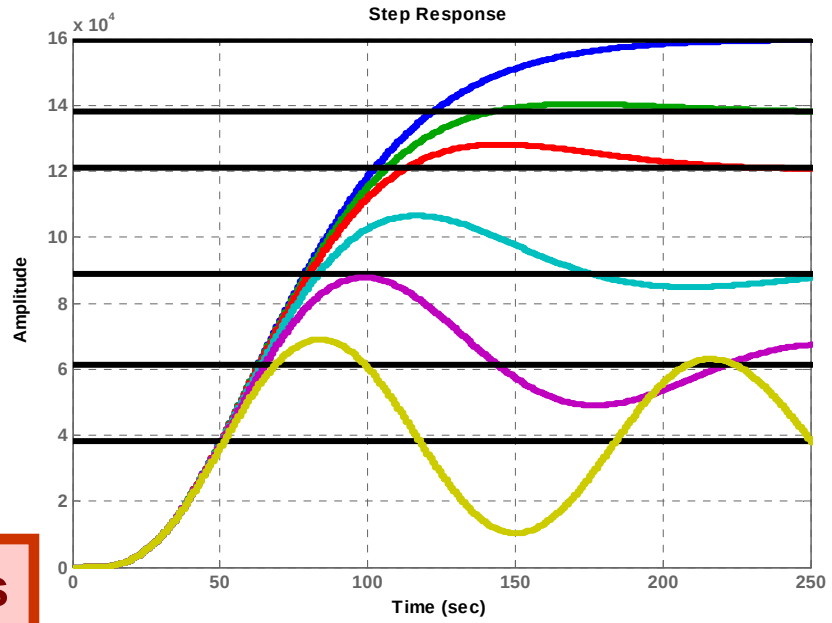
Direct form

$$y_k = \sum_{i=0}^{m} b_i u_{k-i} - \sum_{i=1}^{n} a_i y_{k-i}$$

$$y_k = u_{k-4} + 3.8y_{k-1} - 9.025y_{k-2} + 3.4561y_{k-3} - 0.8145062y_{k-4}$$

$\varepsilon = 10^{-6}$ deviation in a parameter

$$\frac{y_k}{u_k} = \frac{1}{(q-0.95)^4}$$



1× ε

2 × ε

5 × ε

10 × ε

20 × ε

Sensitive for parameter errors

TU Delft

Companion form

$$x_{k+1} = \begin{bmatrix} -a_1 & -a_2 & \cdots & -a_n \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} x_k$$

Equally sensitive for parameter errors

TU Delft

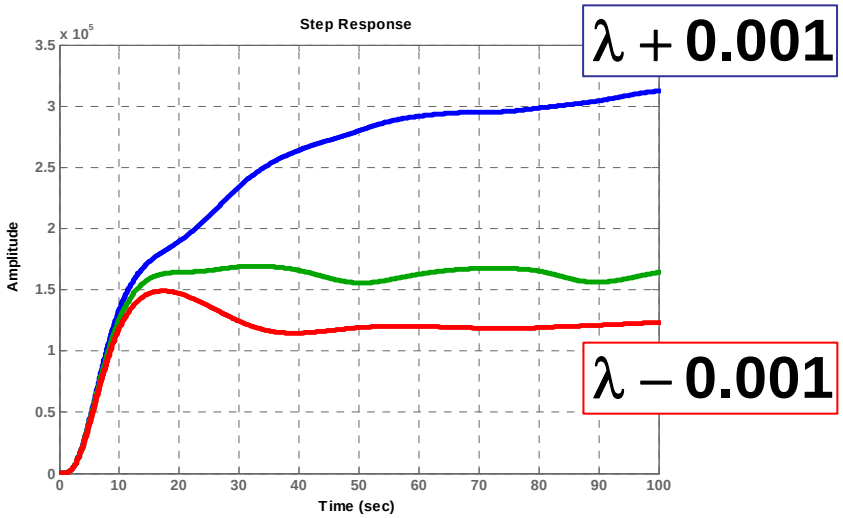
Series form

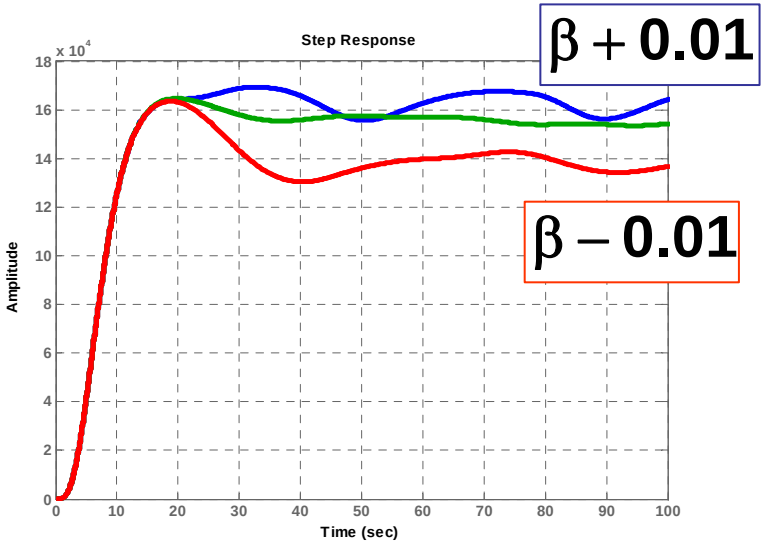
$$H(q) = \frac{num}{(q-\lambda)^4} \quad \Rightarrow \quad x_{k+1} = \begin{bmatrix} \lambda & 1 & 0 & 0 \\ 0 & \lambda & 1 & 0 \\ 0 & 0 & \lambda & 1 \\ 0 & 0 & 0 & \lambda \end{bmatrix} x_k + \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x_k$$

JORDAN

$$\frac{y_k}{u_k} = \frac{\beta_1}{q-\lambda} + \frac{\beta_2}{(q-\lambda)^2} + \frac{\beta_3}{(q-\lambda)^3} + \frac{\beta_4}{(q-\lambda)^4}$$





$\lambda = 0.95$

$\beta = 1$

$$\frac{y_k}{u_k} = \frac{\beta_1}{q-\lambda} + \frac{\beta_2}{(q-\lambda)^2} + \frac{\beta_3}{(q-\lambda)^3} + \frac{\beta_4}{(q-\lambda)^4}$$

Sensitivity reasonable

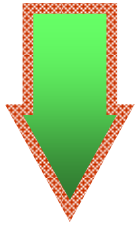
TU Delft

Parallel form

$$x_{k+1} = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & \lambda_4 \end{bmatrix} x_k + \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix} u_k$$

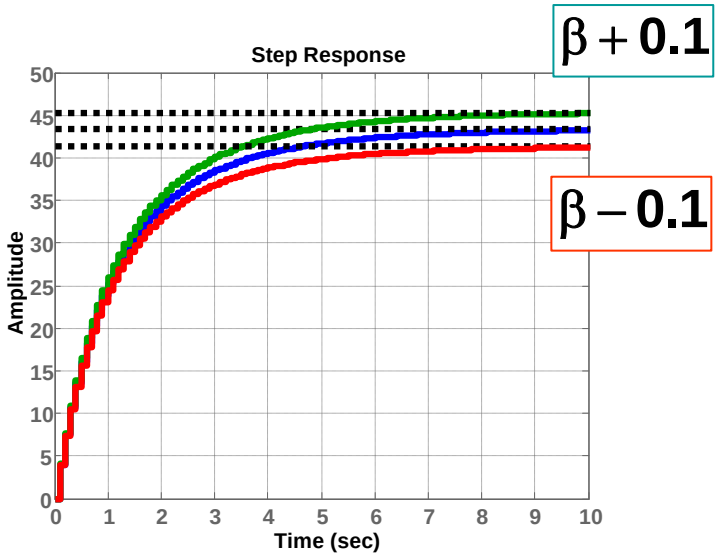
DIAGONAL

$$y_k = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} x_k$$

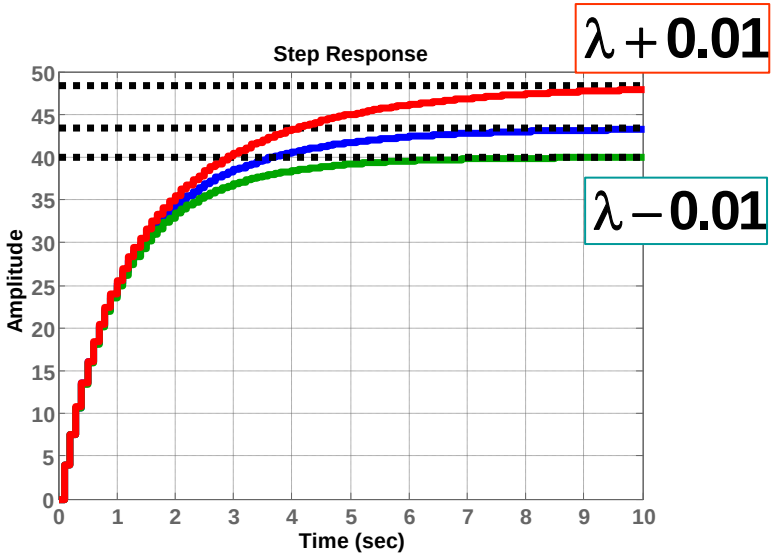


$$\frac{y_k}{u_k} = \frac{\beta_1}{q - \lambda_1} + \frac{\beta_2}{q - \lambda_2} + \frac{\beta_3}{q - \lambda_3} + \frac{\beta_4}{q - \lambda_4}$$

TU Delft



$\beta = 1$



$\lambda = 0.95$

$$\frac{y_k}{u_k} = \frac{\beta_1}{q-\lambda_1} + \frac{\beta_2}{q-\lambda_2} + \frac{\beta_3}{q-\lambda_3} + \frac{\beta_4}{q-\lambda_4}$$

Sensitivity good

TU Delft

δ-operator

$$\delta = \frac{q-1}{h} \quad \Rightarrow \quad \delta f_k = \frac{f_{k+1} - f_k}{h}$$

$$x_{k+1} = Fx_k + Gu_k$$

$$\Downarrow$$

$$x_{k+1} - x_k = (F - I)x_k + Gu_k$$

$$\Downarrow$$

$$\frac{x_{k+1} - x_k}{h} = \frac{F - I}{h} x_k + \frac{G}{h} u_k$$

$$\Downarrow$$

$$\delta x_k = \overline{F}\, x_k + \overline{G}\, u_k$$

TU Delft

Sensitivity

Characteristic equation:

$$(q - 0.95)^4 = q^4 - 3.8\,q^3 + 5.415\,q^2 - 3.4295\,q + 0.8145 = 0$$

Poles: 0.95 — **Stable**

$$q^4 - 3.8\,q^3 + 5.415\,q^2 - 3.4295\,q + 0.8146 = 0$$

Poles: $1.02 \pm j\,0.07$, $0.88 \pm j\,0.07$ — **UNSTABLE**

$$q^4 - 3.8\,q^3 + 5.415\,q^2 - 3.4295\,q + 0.8144 = 0$$

Poles: 1.05, $0.95 \pm j\,0.1$, 0.85 — **UNSTABLE**

TU Delft

Digital Control (SC42095)

Lecture 8, December 11, 2017

Tamás Keviczky

Delft Center for Systems and Control
Faculty of Mechanical Engineering
Delft University of Technology
The Netherlands

e-mail: t.keviczky@tudelft.nl

Lecture outline

- Internal model principle
- Repetitive control
- Disturbance models

Generating polynomial

Assume reference signal or disturbance $d(t)$ satisfies the differential equation:

$$\frac{d^{n_d}}{dt^{n_d}}d(t) + \gamma_{n_d-1}\frac{d^{n_d-1}}{dt^{n_d-1}}d(t) + \cdots + \gamma_1\frac{d}{dt}d(t) + \gamma_0 d(t) = 0$$

$$\underbrace{\left(s^{n_d} + \gamma_{n_d-1}s^{n_d-1} + \cdots + \gamma_0\right)}_{\Gamma_d(s)} D(s) = f(0,s)$$

- $\Gamma_d(s)$ is called disturbance generating polynomial
- $f(0,s)$ is a polynomial in s (due to initial conditions)

Disturbance generating polynomial examples

$$d(t) = d_0 \text{ constant} \quad \rightarrow \quad \Gamma_d(s) = s$$

$$d(t) = \sin(\omega t) \quad \rightarrow \quad \Gamma_d(s) = s^2 + \omega^2$$

$$d(t) = e^{at} \quad \rightarrow \quad \Gamma_d(s) = s - a$$

$$d(t) = d_0 + d_1 e^{at} \quad \rightarrow \quad \Gamma_d(s) = s(s-a)$$

Internal model principle (IMP)

Assume a standard one-degree-of-freedom control architecture.

If $d_i(t), d_o(t), r(t)$ has $\Gamma_d(s)$ as their generating polynomial, then the controller

$$C(s) = \frac{P(s)}{\Gamma_d(s)\bar{L}(s)}$$

can asymptotically reject the effect of input-, output-disturbance, and track the reference.

Note: Only the generating polynomial is needed, not the magnitude of the disturbance / reference.

Why does it work?

Recall: For step reference tracking, step disturbance rejection, we needed an integrator in the controller to get zero steady state error.

Describe plant as $G(s) = \frac{B(s)}{A(s)}$, and assume that $\Gamma_d(s)$ is not a factor of $B(s)$.

Closed-loop transfer functions:

$$S = \frac{\Gamma_d\bar{L}A}{\Gamma_d\bar{L}A + PB}, \quad S_i = \frac{\Gamma_d\bar{L}B}{\Gamma_d\bar{L}A + PB}$$

$$T = \frac{PB}{\Gamma_d\bar{L}A + PB}$$

Why does it work? (cont'd)

Suppose $\bar{L}, P$ chosen such that the closed-loop characteristic equation

$$A_{cl}(s) = \Gamma_d(s)\bar{L}(s)A(s) + P(s)B(s)$$

has roots with negative real parts.

E.g., by pole-placement, input-output design (using Diophantine equation or Sylvester matrix).

Why does it work? (cont'd)

Response of system to output disturbance $d_o(t)$ with generating polynomial $\Gamma_d(s)$:

$$Y(s) = S(s)D_o(s) = S(s)\frac{f(0,s)}{\Gamma_d(s)} = \frac{\bar{L}(s)A(s)}{A_{cl}(s)}$$

Notice that cancelation of $\Gamma_d(s)$ occurs and $y(t \to \infty) = 0$ since A_{cl} is stable.
For input disturbance $d_i(t)$:

$$Y(s) = S_i(s)D_i(s) = \frac{\bar{L}(s)B(s)}{A_{cl}(s)}$$

For reference $r(t)$ $(e(t) = r(t) - y(t))$:

$$E(s) = (1 - T(s))\frac{f(0,s)}{\Gamma_d(s)} = S(s)\frac{f(0,s)}{\Gamma_d(s)}$$

IMP in state-space

Create a disturbance exo-system

$$\begin{aligned} \dot{x}_d &= A_d x_d \\ d &= C_d x_d \end{aligned}$$

Estimate $d(t)$ using an observer (see in Lecture 6).

The controller will have the eigenvalues of A_d as poles.

$$\begin{aligned} D(s) &= C_d(sI - A_d)^{-1}x_d(0) \\ \Gamma_d(s) &= \det(sI - A_d) \end{aligned}$$

IMP in discrete-time

$d(k)$ satisfies:

$$\begin{aligned} d(k) + \gamma_1 d(k-1) + \cdots + \gamma_N d(k-N) &= 0 \\ \underbrace{\left(1 + \gamma_1 q^{-1} + \cdots + \gamma_N q^{-N}\right)}_{\Gamma_d(q^{-1})} d(k) &= 0 \end{aligned}$$

The rest of the story is the same as in continuous-time...

Repetitive control

Goal is to eliminate the effect of periodic disturbance,
or to track periodic reference input.
(Need to "learn" the disturbance → special IMP controller.)

Repetitive control

Consider N-periodic disturbances: $d(k-N) = d(k)$

This means that $d(k)$ satisfies $q^{-N}d(k) = d(k)$

$$\left(1 - q^{-N}\right) d(k) = 0 \quad \rightarrow \quad \Gamma_d(q^{-1}) = 1 - q^{-N}$$

Let's try to use IMP in discrete-time:

$$C(q^{-1}) = \frac{P(q^{-1})}{\Gamma_d(q^{-1})\bar{L}(q^{-1})}$$

Problem: Disturbance generating polynomial is of very high order, difficult to assign poles.

Prototype controller

Assume $G(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})}$ is stable, minimum phase.

Choose

$$C(q^{-1}) = k_r \frac{A q^{\delta - N}}{B(1 - q^{-N})}$$

which leads to $e(k) = (1 - k_r) e(k - N)$ and it

1. inverts the plant
2. has $(1 - q^{-N})$ in denominators because of IMP
3. delays the control by one cycle (q^{-N})

Remarks

- $A(q^{-1})$ should be stabilized first.
- $B(q^{-1})$ unstable factors shouldn't be canceled.
- Further modification necessary (gain mod., zero phase comp.).
- Robustness problems using $\Gamma_d = (1 - q^{-N})$, which implies controller with very high gain at *all* harmonics of the disturbance frequency.

  Solution: limit bandwidth by modifying $1 - q^{-N}$ to

  $$1 - Q(q, q^{-1}) q^{-N}$$

  where Q is a unity gain zero phase filter, e.g.:

  $$Q(q, q^{-1}) = 0.1q^2 + 0.15q + 0.5 + 0.15q^{-1} + 0.1q^{-2}$$

  It smoothes out the generating polynomial and reduces gain at high order harmonics.

Example

Consider a single integrator

$$G(q^{-1}) = \frac{h q^{-1}}{1 - q^{-1}}$$

thus $A = 1 - q^{-1}$, $B = h$, $\delta = 1$. The controller

$$C(q^{-1}) = \frac{k_r}{h} \frac{q^{-(N-1)}(1 - q^{-1})}{1 - q^{-N}}$$

leads to

$$u(k) = u(k - N) - \frac{k_r}{h} \left(e(k + 1 - N) - e(k + 2 - N) \right)$$

which shows that the control action is updated with the error in the previous cycle (the control is "learned").

Repetitive control in state-space

Disturbance generating exo-system:

$$\begin{pmatrix} x_{d1}(k+1) \\ \vdots \\ x_{dN}(k+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} x_{d1}(k) \\ \vdots \\ x_{dN}(k) \end{pmatrix}$$

$$d(k) = x_{d1}(k)$$

Repetitive control in state-space (cont'd)

We then proceed with the disturbance-estimate feedback approach to IMP:

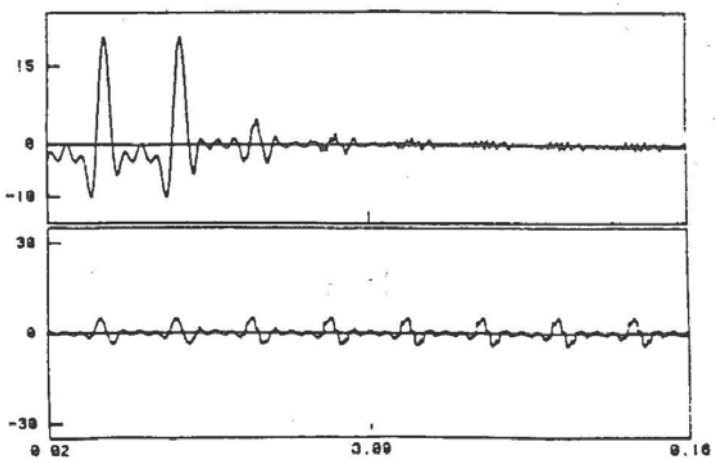
$$u(k) = -Kx(k) - \hat{d}(k)$$

$$\begin{pmatrix} \hat{x}(k+1) \\ \hat{x}_d(k+1) \end{pmatrix} = \overbrace{\begin{pmatrix} A & BC_d \\ 0 & A_d \end{pmatrix}}^{\tilde{A}} \begin{pmatrix} \hat{x}(k) \\ \hat{x}_d(k) \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} u(k) - L\left(y(k) - C\hat{x}(k)\right)$$

$$\hat{d}(k) = C_d \hat{x}_d(k)$$

Choose L such that $\tilde{A} - LC$ is stable.

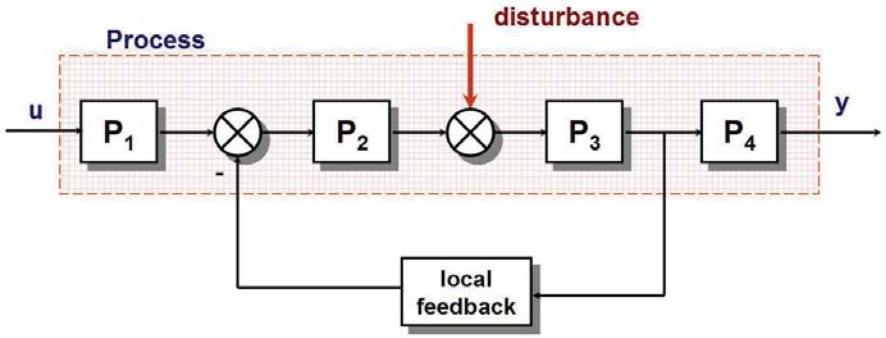
Repetitive control result example



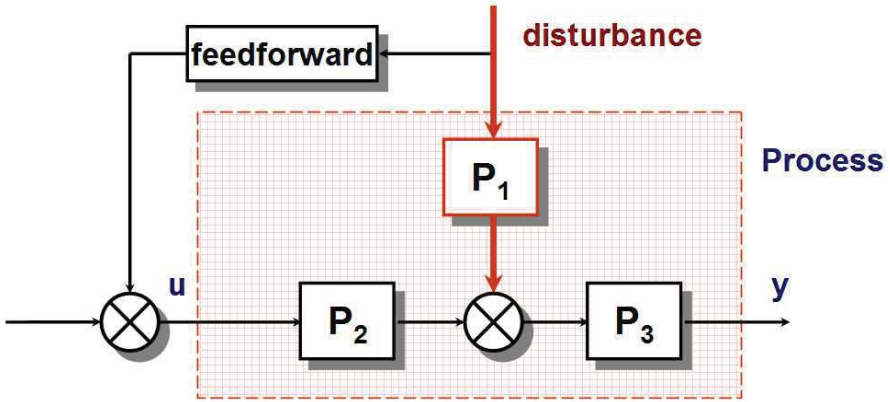
Reduction of effects of disturbances

- Reduction at the source
- Reduction by local feedback
- Reduction by feedforward
- Reduction by prediction

Disturbance compensation with local feedback



Disturbance compensation with feedforward



Disturbance compensation using prediction

Signal $y(k)$ is assumed to be generated by:

$$\begin{aligned} x(k+1) &= \Phi x(k) + v(k) \\ y(k) &= Cx(k) \end{aligned}$$

where $v(k)$ is assumed to be zero except at isolated points.

$$x(k-n+1) = W_o^{-1}\left(y(k-n+1) \quad \cdots \quad y(k-1) \quad y(k)\right)^T$$

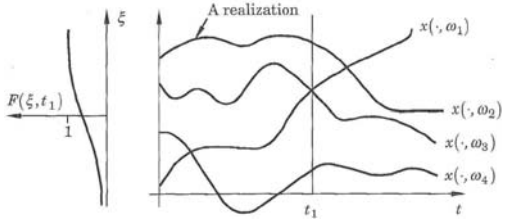
$$\hat{x}(k+m|k) = \Phi^{m+n-1} W_o^{-1}\left(y(k-n+1) \quad \cdots \quad y(k-1) \quad y(k)\right)^T$$

This gives the predictor as a polynomial of degree $n-1$

$$\hat{y}(k+m|k) = P^*(q^{-1})y(k)$$

Stochastic processes and disturbance models

Stochastic (random) process: $\{x(t,\omega), t \in T, \omega \in W\}$
$\sim$ indexed family of random variables.



Finite-dimensional distribution function:

$$F(\xi_1, \ldots, \xi_n; t_1, \ldots, t_n) = P\left\{x(t_1) \leq \xi_1, \ldots, x(t_n) \leq \xi_n\right\}$$

If all finite-dimensional distributions are shift-invariant:
stationary stochastic process

Stochastic processes and disturbance models

Mean-value function (constant for stationary processes):

$$m(t) = \mathrm{E}\left(x(t)\right) = \int_{-\infty}^{\infty} \xi dF(\xi; t)$$

Covariance function (function of only $s-t$ for stationary processes):

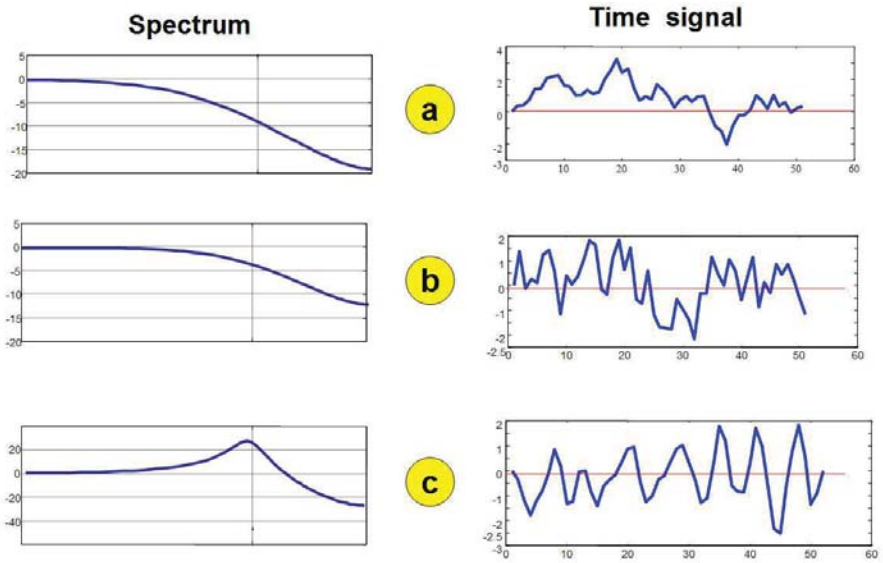
$$\begin{aligned} r_{xx}(s,t) &= \mathrm{cov}\left(x(s), x(t)\right) = \mathrm{E}\left((x(s)-m(s))(x(t)-m(t))^T\right) \\ &= \int\int (\xi_1 - m(s))(\xi_2 - m(t))^T dF(\xi_1, \xi_2; s, t) \end{aligned}$$

Process variance is $r_x(0)$ (how large fluctuations are).

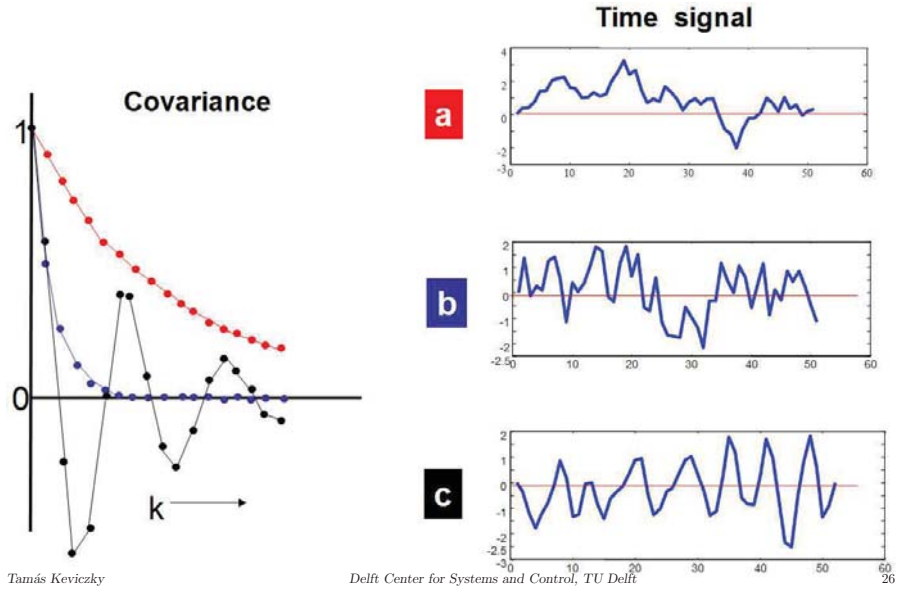
Spectral density:

$$\phi_x(\omega) = \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} r_x(k) e^{-jk\omega}$$

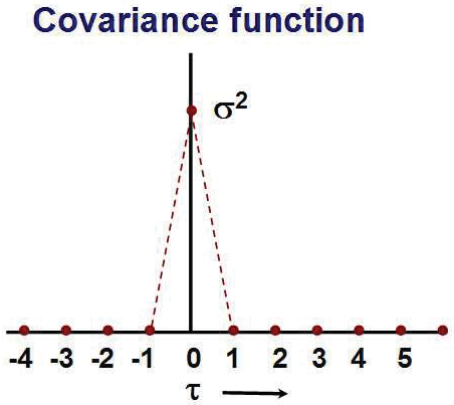
Interpretation of spectrum



Interpretation of covariance

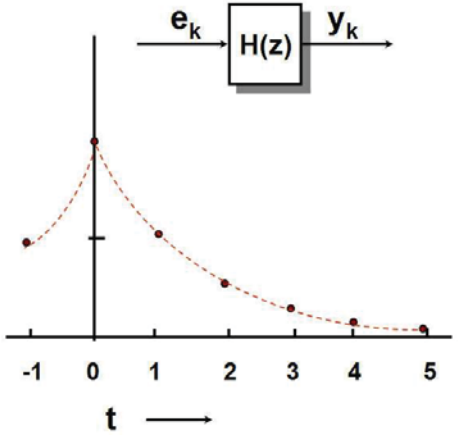


Discrete-time white noise



$$\phi(\omega) = \frac{\sigma^2}{2\pi}$$

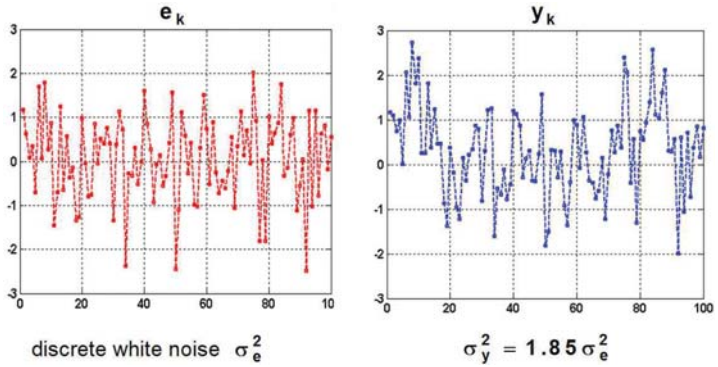
Stochastic disturbance model



$$\begin{aligned} H(z) &= \frac{1}{z-a} \\ &\Downarrow \\ y(k+1) &= ay(k) + e(k) \\ &\Downarrow \\ \sigma_y^2 &= \frac{1}{1-a^2}\sigma_e^2 \\ &\Downarrow \\ r_y(\tau) &= \frac{1}{1-a^2}a^{|\tau|} \end{aligned}$$

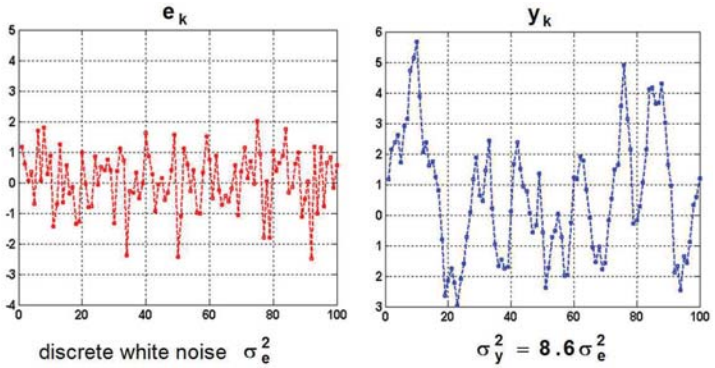
Calculation of variances - Example 1

$$H(q) = \frac{q - 0.5}{q - 0.9} \quad \text{with } \sigma_e^2 = 1$$



Calculation of variances - Example 2

$$H(q) = \frac{1}{q^2 - 1.3q + 0.4} \quad \text{with } \sigma_e^2 = 1$$



ARMA processes

Moving Average:

$$y(k) = e(k) + b_1 e(k-1) + \cdots + b_n e(k-n)$$

Auto Regressive:

$$y(k) + a_1 y(k-1) + \cdots + a_n y(k-n) = e(k)$$

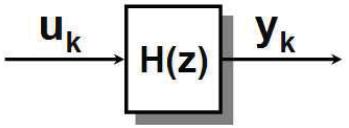
ARMA:

$$y(k)+a_1 y(k-1)+\cdots+a_n y(k-n) = e(k)+b_1 e(k-1)+\cdots+b_n e(k-n)$$

ARMAX:

$$\begin{aligned} y(k) + a_1 y(k-1) + \cdots + a_n y(k-n) &= b_0 u(k-d) + \cdots + b_m u(k-d-m) \\ &+ e(k) + c_1 e(k-1) + \cdots + c_n e(k-n) \end{aligned}$$

Spectral densities of filtered signals



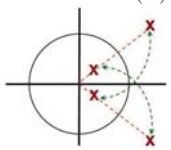
$$\begin{aligned} \phi_{yu}(\omega) &= H(e^{j\omega})\phi_u(\omega) \\ \phi_y(\omega) &= H(e^{j\omega})\phi_u(\omega)H^T(e^{-j\omega}) \end{aligned}$$

Spectral factorization

Given a spectral density $\phi(\omega)$, what is the linear system that gives this as an output, when driven by white noise?

$$F(z) = \frac{1}{2\pi} H(z) H^T(z^{-1})$$

The poles and zeros of $F(z)$ come in pairs:



$$z_i z_j = 1$$
$$p_i p_j = 1$$

$$H(z) = K \frac{\prod_i (z - z_i)}{\prod_i (z - p_i)} = \frac{B(z)}{A(z)}, \quad |z_i| < 1, |p_i| < 1$$

All stationary random processes can be thought of as:
generated by stable linear systems driven by white noise
(special ARMA processes)

Digital Control SC42095

Linear quadratic control

Tamás Keviczky
Delft Center for Systems and Control
Delft University of Technology

Lecture outline

1. Linear quadratic control
   - deterministic case
   - completing the squares
   - dynamic programming
   - example
   - stochastic case
2. Kalman filtering
3. Linear quadratic Gaussian control

Note: These slides are partly inspired by the slides for this course developed at the Department of Automatic Control, Lund Institute of Technology (see http://www.control.lth.se/~kursdr)

Lecture outline (continued)

- Previous control methods: PID, pole placement
  $\rightarrow$ focus on SISO
- Now, more general: MIMO + process & measurement noise
  $\rightarrow$ optimization-based approach
- First we consider case with *full state information*
  = Linear Quadratic (LQ) control

Next lecture:

- Estimating state from measurements of noisy output
  = Kalman filtering
- Finally, combination based on *separation theorem*
  = Linear Quadratic Gaussian (LQG) control

1. Linear quadratic control

1.1 LQ control: Deterministic case

- *Discrete-time* LTI state space model:

$$x(k+1) = \Phi x(k) + \Gamma u(k)$$
$$y(k) = Cx(k)$$

- We start at $k = 0$ with $x(0) = x_0$ and look N steps into the future, and determine optimal control sequence $u(0), \ldots, u(N-1)$
- Performance criterion ("loss function") J over period $[0, N]$

  Inspired by "power" of the state: $\int_0^T \|x(t)\|^2 dt$

  or in discrete time: $\sum_{k=0}^{N} x^{\mathsf{T}}(k)x(k)$

1.1 LQ control: Deterministic case (continued)

- Components of state might have different units
  $\rightarrow$ weighting
  $$J = \sum_{k=0}^{N} x^{\mathsf{T}}(k) Q_1 x(k)$$
  Control signal u can be penalized in similar way
- So in general:
  $$J = \sum_{k=0}^{N-1} \left(x^{\mathsf{T}}(k) Q_1 x(k) + 2x^{\mathsf{T}}(k) Q_{12} u(k) + u^{\mathsf{T}}(k) Q_2 u(k) \right) + x^{\mathsf{T}}(N) Q_0 x(N)$$
  with Q-matrices symmetric and positive semi-definite
  i.e., $v^{\mathsf{T}} Q v \geqslant 0$ for all v

  Note: Separate penalty term for final state

Deterministic LQ: Problem formulation

- Minimize
  $$J = \sum_{k=0}^{N-1} \left(x^{\mathsf{T}}(k) Q_1 x(k) + 2x^{\mathsf{T}}(k) Q_{12} u(k) + u^{\mathsf{T}}(k) Q_2 u(k) \right) + x^{\mathsf{T}}(N) Q_0 x(N)$$
  subject to $x(k+1) = \Phi x(k) + \Gamma u(k)$ and $x(0) = x_0$

  Note: For simplicity of notation we sometimes drop argument k for x and u in the sequel
- Solution approach based on quadratic optimization and dynamic programming

1.2 Completing the squares

- Scalar case: Find u that minimizes $ax^2 + 2bxu + cu^2$ with $c > 0$
- Use derivative or completing the squares:
  $$\begin{aligned} ax^2 + 2bxu + cu^2 &= ax^2 + c\left(2\frac{b}{c}xu + u^2\right) \\ &= ax^2 + c\left(u^2 + 2\frac{b}{c}xu + \frac{b^2}{c^2}x^2 - \frac{b^2}{c^2}x^2\right) \\ &= \left(a - \frac{b^2}{c}\right)x^2 + c\left(u + \frac{b}{c}x\right)^2 \end{aligned}$$
- First term independent of u, second term always nonnegative
  So minimum is reached for $u = -\frac{b}{c}x$, and minimum value is
  $\left(a - \frac{b^2}{c}\right)x^2$

1.2 Completing the squares (continued)

- Matrix case: Find u that minimizes $x^{\mathsf{T}} Q_x x + 2x^{\mathsf{T}} Q_{xu} u + u^{\mathsf{T}} Q_u u$ with Q_u positive definite
- Let L be such that $Q_u L = Q_{xu}^{\mathsf{T}}$. Then
  $$\begin{aligned} &x^{\mathsf{T}} Q_x x + 2x^{\mathsf{T}} Q_{xu} u + u^{\mathsf{T}} Q_u u = \\ &\qquad x^{\mathsf{T}}(Q_x - L^{\mathsf{T}} Q_u L)x + (u + Lx)^{\mathsf{T}} Q_u (u + Lx) \end{aligned}$$
  is minimized for $u = -Lx$
  and minimum value is $x^{\mathsf{T}}(Q_x - L^{\mathsf{T}} Q_u L)x$
- If Q_u is positive definite, then $L = Q_u^{-1} Q_{xu}^{\mathsf{T}}$

1.3 LQ control via least-squares

- The LQ control problem can be formulated (and solved) as a (large) least-squares problem.
- Note that $X = (x(0), \ldots, x(N))$ is a *linear function* of $x(0)$ and $U = (u(0), \ldots, u(N-1))$:

$$\underbrace{\begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} \Gamma & 0 & \cdots & 0 \\ \Phi\Gamma & \Gamma & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \Phi^{N-1}\Gamma & \Phi^{N-2}\Gamma & \cdots & \Gamma \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(N-1) \end{bmatrix}}_{U} + \underbrace{\begin{bmatrix} \Phi \\ \Phi^2 \\ \vdots \\ \Phi^N \end{bmatrix}}_{H} x(0)$$

1.3 LQ control via least-squares (continued)

- Thus (assuming $Q_{12} = 0$ for simplicity) we can express the LQ cost function as

$$J(U) = \left\| \mathbf{diag}\left(Q_1^{1/2}, \ldots, Q_1^{1/2}, Q_0^{1/2}\right)(GU + Hx(0)) \right\|^2 + \left\| \mathbf{diag}\left(Q_2^{1/2}, \ldots, Q_2^{1/2}\right) U \right\|^2$$

- This is just a (big) least-squares problem!
- This solution method requires forming and solving a least-squares problem with size that grows with N!

1.4 Dynamic programming

- **Principle of optimality**: From any point on optimal trajectory, remaining trajectory is also optimal

  $\rightarrow$ allows to determine best control law over period $[t_2, t_3]$ independent of how state at t_2 was reached

- Useful idea, has many applications beyond LQ control, e.g.
  - optimal flow control in communication networks
  - optimization in finance

LQ control: Dynamic programming solution

- Gives an efficient, recursive method to solve LQ control least-squares problem
- Define the value function J_k as

$$J_k(z) = \min_{u(k),\ldots,u(N-1)} \sum_{l=k}^{N-1} \left(x^{\mathsf{T}} Q_1 x + 2x^{\mathsf{T}} Q_{12} u + u^{\mathsf{T}} Q_2 u \right) + x^{\mathsf{T}}(N) Q_0 x(N)$$

  subject to $x(k) = z$ and $x(l+1) = \Phi x(l) + \Gamma u(l)$.
- We will write x and u instead of $x(l)$ and $u(l)$ for simplicity.
- $J_k(z)$ gives the minimum LQ cost-to-go, starting from state z at time k.
- $J_0(z)$ is the minimum LQ cost (from state $x(0)$ at time 0).

LQ control: Dynamic programming solution (continued)

We will find that

- J_k is quadratic, i.e., $J_k(z) = z^\mathsf{T} S(k) z$, where $S(k) = S(k)^\mathsf{T} \geq 0$
- $S(k)$ can be found recursively, working backwards from $k = N$
- the LQ optimal control u is easily expressed in terms of $S(k)$

LQ control: Dynamic programming solution (continued)

- Cost-to-go with no time left is just the final state cost:

$$J_N(z) = z^\mathsf{T} Q_0 z$$

  thus we have $S(N) = Q_0$.
- Now suppose we know J_{k+1}, what is the optimal choice for $u(k)$?
- Choice of $u(k)$ affects
  - current cost incurred (through $u(k)^\mathsf{T} Q_2 u(k)$)
  - where we end up, i.e., the min-cost-to-go from $x(k+1)$

Dynamic programming principle

$$J_k(x) = x^\mathsf{T} Q_1 x + \min_u \left(u^\mathsf{T} Q_2 u + J_{k+1}(\Phi x + \Gamma u) \right)$$

- Called DP, Bellman, or Hamilton-Jacobi equation
- Gives J_k recursively, in terms of J_{k+1}
- Any minimizing u gives optimal control action
- For N-step problem:
  - start from end at time $k = N$
  - now we can determine best control law for last step independent of how state at time $N-1$ was reached
  - iterate backward in time to initial time $k = 0$

Dynamic programming solution

- Define $S(k)$ as

$$J_k(x) = x^\mathsf{T} S(k) x$$

- Principle of optimality with $t_1 = k$, $t_2 = k+1$, $t_3 = N$ gives

$$\begin{aligned} x^\mathsf{T} S(k) x &= \min_u \; x^\mathsf{T} Q_1 x + 2 x^\mathsf{T} Q_{12} u + u^\mathsf{T} Q_2 u + x^\mathsf{T}(k+1) S(k+1) x(k+1) \\ &= \min_u \; x^\mathsf{T} Q_1 x + 2 x^\mathsf{T} Q_{12} u + u^\mathsf{T} Q_2 u + \\ &\qquad (\Phi x + \Gamma u)^\mathsf{T} S(k+1) (\Phi x + \Gamma u) \end{aligned}$$

Dynamic programming solution (continued)

- So

$$x^\mathsf{T}S(k)x = \min_u x^\mathsf{T}\underbrace{\left(Q_1+\Phi^\mathsf{T}S(k+1)\Phi\right)}_{Q_x}x + 2x^\mathsf{T}\underbrace{\left(Q_{12}+\Phi^\mathsf{T}S(k+1)\Gamma\right)}_{Q_{xu}}u + u^\mathsf{T}\underbrace{\left(Q_2+\Gamma^\mathsf{T}S(k+1)\Gamma\right)}_{Q_u}u$$

- Completing-of-squares solution then gives:

$$u(k) = -L(k)x(k)$$
$$L(k) = Q_u^{-1}Q_{xu}^\mathsf{T} = (Q_2+\Gamma^\mathsf{T}S(k+1)\Gamma)^{-1}\left(Q_{12}+\Phi^\mathsf{T}S(k+1)\Gamma\right)^\mathsf{T}$$
$$S(k) = Q_x - L^\mathsf{T}Q_uL = Q_1+\Phi^\mathsf{T}S(k+1)\Phi - L^\mathsf{T}(Q_2+\Gamma^\mathsf{T}S(k+1)\Gamma)L$$

$\rightarrow$ Discrete-time Riccati recursion

Dynamic programming solution (continued)

- Discrete-time Riccati recursion:

$$S(k) = Q_1+\Phi^\mathsf{T}S(k+1)\Phi - \left(Q_{12}+\Phi^\mathsf{T}S(k+1)\Gamma\right)\left(Q_2+\Gamma^\mathsf{T}S(k+1)\Gamma\right)^{-1}\left(Q_{12}^\mathsf{T}+\Gamma^\mathsf{T}S(k+1)\Phi\right)$$

- $S(N)$ determined via

$$x^\mathsf{T}(N)S(N)x(N) := x^\mathsf{T}(N)Q_0x(N)$$

So $S(N) = Q_0$

- Minimum of loss function over period $[0,N]$ is given by

$$\min J = x^\mathsf{T}(0)S(0)x(0)$$

Dynamic programming solution (continued)

- Solution of LQ problem is time-varying controller

$$u(k) = -L(k)x(k)$$

- Feedback matrix $L(k)$ does not depend on x and can be pre-computed for $k = N, N-1, \ldots, 0$ and stored on computer
- In practice only stationary controller is used, i.e., constant controller obtained when Riccati recursion is iterated until constant $S(k) = \bar{S}$ is obtained:

$$\bar{S} = Q_1+\Phi^\mathsf{T}\bar{S}\Phi - \left(Q_{12}+\Phi^\mathsf{T}\bar{S}\Gamma\right)^\mathsf{T}(Q_2+\Gamma^\mathsf{T}\bar{S}\Gamma)^{-1}\left(Q_{12}^\mathsf{T}+\Gamma^\mathsf{T}\bar{S}\Phi\right)$$

This is called the discrete-time algebraic Riccati equation (ARE). The stationary controller is then:

$$L = \left(Q_2+\Gamma^\mathsf{T}\bar{S}\Gamma\right)^{-1}\left(Q_{12}+\Phi^\mathsf{T}\bar{S}\Gamma\right)^\mathsf{T}$$

Note: $\bar{S}$ does not depend on final state weight matrix Q_0

Properties of the LQ controller

- Consider LTI system and loss function with positive definite Q-matrices.
- If steady-state solution $\bar{S}$ exists and is positive definite, then steady-state control strategy

$$u(k) = -Lx(k)$$

gives asymptotically stable closed-loop system

$$x(k+1) = (\Phi-\Gamma L)x(k)$$

Proof: See book Åström & Wittenmark, pp. 423–424
Based on using $x^\mathsf{T}(k)\bar{S}x(k)$ as Lyapunov function

Properties of the LQ controller (continued)

- Discrete-time LQ controller has finite gain margin as opposed to continuous-time LQ controller! (Just like every digital controller.)
- LQ control is readily extended to handle time-varying systems $\Phi(k), \Gamma(k)$, and time-varying cost matrices $Q_1(k), Q_{12}(k), Q_2(k)$. The DP solution idea is the same, but there need not be a steady-state solution.
- Tracking problems can also be considered by replacing the state $x(k)$ and control $u(k)$ terms in the cost function with $x(k) - \bar{x}(k)$ and $u(k) - \bar{u}(k)$. The $\bar{x}(k)$ and $\bar{u}(k)$ are given state and input trajectories.

How to find weighting matrices?

- In principle derived from "natural" loss function
  But not always possible/applicable in practice
  $\rightarrow$ designer chooses/tunes weighting matrices
- So what is the difference from pole placement or direct search over $L(k)$?
  In theory: none
  In practice: LQ preferred since easy to use
- Sometimes one selects $Q_{12} = 0$ and Q_1, Q_2 diagonal with as diagonal entries the inverse value of square of allowed deviations in states and control signals:

$$(Q_{1/2})_{ii} = \frac{1}{(\text{allowed deviation in state/control input } i)^2}$$

How to solve LQ problem in matlab?

- Command `[K,S,E] = dlqr(A,B,Q,R,N)`
- Calculates optimal gain matrix `K` such that state-feedback law `u[n] = -Kx[n]` minimizes cost function

```
J = Sum {x'Qx + u'Ru + 2*x'Nu}
```

  subject to state dynamics

```
x[n+1] = Ax[n] + Bu[n]
```

- Also returns steady-state solution `S` of Riccati equation and closed-loop eigenvalues `E=eig(A-B*K)`

1.5 Example

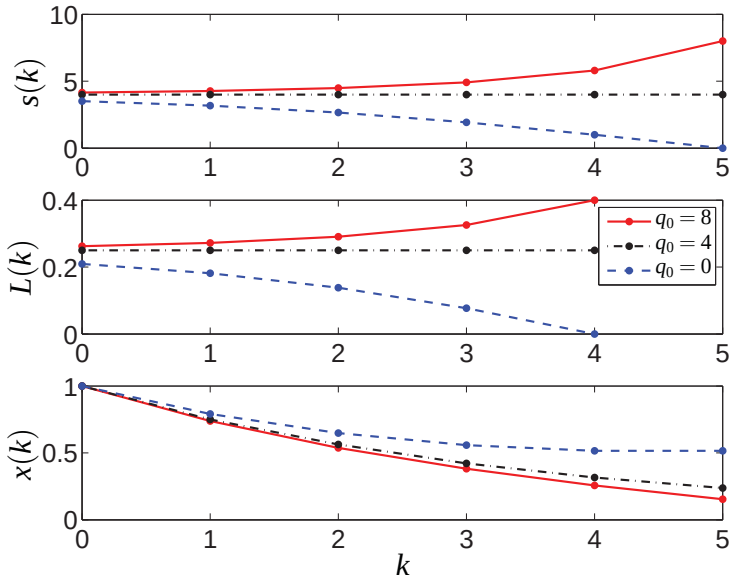
- Discrete-time system $x(k+1) = x(k) + u(k)$
- Loss function $\sum_{k=0}^{N-1} \left(x^2(k) + 12u^2(k) \right) + q_0 x^2(N)$ with $N = 5$
- Riccati equation:

$$s(k) = s(k+1) + 1 - \frac{s^2(k+1)}{s(k+1)+12} \qquad \text{with } s(N) = q_0$$

- Controller: $u(k) = -l(k)x(k) = -\frac{s(k+1)}{s(k+1)+12}x(k)$
- Stationary solution given by:

$$\bar{s} = \bar{s} + 1 - \frac{\bar{s}^2}{\bar{s}+12} \quad \rightarrow \quad \bar{s}^2 - \bar{s} - 12 = 0 \quad \rightarrow \quad \bar{s} = 4$$

1.5 Example (continued)



1.6 LQ control: Stochastic case

- Minimize

$$J = \mathsf{E}\left[\sum_{k=0}^{N-1}\left(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u\right) + x^{\mathsf{T}}(N)Q_0x(N)\right]$$

subject to $x(k+1) = \Phi x(k) + \Gamma u(k) + v(k)$ and $x(0) = x_0$
with v Gaussian zero-mean white noise process with

$$\mathsf{E}\left[v(k)v^{\mathsf{T}}(k)\right] = R_1$$

and $x(0)$ has Gaussian distribution with

$$\mathsf{E}\left[x(0)\right] = m_0, \quad \mathsf{cov}(x(0)) = \mathsf{E}\left[(x(0)-m_0)^{\mathsf{T}}(x(0)-m_0)\right] = R_0$$

- Solution: also based on dynamic programming and Riccati equation

1.6 LQ control: Stochastic case (continued)

- Define $S(k)$ by Riccati equation and

$$V_k(x(k)) = \min_{u(k),\ldots,u(N-1)} \mathsf{E}\left[\sum_{k}^{N-1}\left(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u\right) + x^{\mathsf{T}}(N)Q_0x(N)\right]$$

- Then with $x = x(N)$

$$V_N(x) = \mathsf{E}\left[x^{\mathsf{T}}Q_0x\right] = \mathsf{E}\left[x^{\mathsf{T}}S(N)x\right]$$

and with $x = x(N-1)$ and $u = u(N-1)$

$$V_{N-1}(x) = \min_u \mathsf{E}\left[(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u) + \underbrace{x^{\mathsf{T}}(N)Q_0x(N)}_{V_N(x(N))}\right]$$

1.6 LQ control: Stochastic case (continued)

- Using principle of optimality:

$$\begin{aligned}
&V_{N-1}(x)\\
&= \min_u \mathsf{E}\left[(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u) + V_N(x(N))\right]\\
&= \min_u \mathsf{E}\left[(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u) + (\Phi x + \Gamma u + v)^{\mathsf{T}}S(N)(\Phi x + \Gamma u + v)\right]\\
&= \min_u \mathsf{E}\left[(x^{\mathsf{T}}Q_1x + 2x^{\mathsf{T}}Q_{12}u + u^{\mathsf{T}}Q_2u) + (\Phi x + \Gamma u)^{\mathsf{T}}S(N)(\Phi x + \Gamma u)\right]\\
&\qquad + \mathsf{E}\left[v^{\mathsf{T}}S(N)v\right]
\end{aligned}$$

as v is independent of x and u

- Part to be minimized is independent of v, so

$$V_{N-1}(x) = \mathsf{E}\left[x^{\mathsf{T}}S(N-1)x\right] + \mathsf{E}\left[v^{\mathsf{T}}S(N)v\right]$$

1.6 LQ control: Stochastic case (continued)

- Results in

$$V_0(x) = \mathsf{E}\left[x^\mathsf{T} S(0) x\right] + \sum_{k=1}^{N-1} \mathsf{E}\left[v^\mathsf{T}(k) S(k+1) v(k)\right]$$

- Solution is then again given by $u(k) = -L(k)x(k)$ with $L(k)$ as defined in deterministic case and $S(k)$ given by Riccati equation

- Minimum cost equal to

$$V_0(x_0) = m_0^\mathsf{T} S(0) m_0 + \mathrm{tr}\left[S(0)R_0\right] + \sum_{k=1}^{N-1} \mathrm{tr}\left[S(k+1)R_1\right]$$

Proof: See book Åström and Wittenmark, pp. 418–419

Summary

- LQ control
  - minimize loss function over horizon of N steps
  - deterministic + stochastic case
  - full state information $\rightarrow$ state feedback controller
  - solution based on quadratic optimization and dynamic programming

Digital Control SC42095

Kalman filtering and LQG control

Tamás Keviczky
Delft Center for Systems and Control
Delft University of Technology

Lecture outline

1. Review
   - LQ control
   - completing the squares
   - dynamic programming
2. Kalman filtering
3. Linear quadratic Gaussian control

Note: These slides are partly inspired by the slides for this course developed at the Department of Automatic Control, Lund Institute of Technology (see http://www.control.lth.se/~kursdr)

Lecture outline (continued)

- Linear Quadratic (LQ) control $\rightarrow$ assumes *full state information*
- Estimating state from measurements of output = Kalman filtering
- Combination of LQ and state estimation = Linear Quadratic Gaussian (LQ) control based on *separation theorem*

1. Review

1.1 LQ control

- Minimize

$$J=\sum_{k=0}^{N-1}\left(x^{\mathsf{T}}(k)Q_1x(k)+2x^{\mathsf{T}}(k)Q_{12}u(k)+u^{\mathsf{T}}(k)Q_2u(k)\right)+x^{\mathsf{T}}(N)Q_0x(N)$$

  subject to $x(k+1)=\Phi x(k)+\Gamma u(k)$ and $x(0)=x_0$

- Solution approach based on quadratic optimization problem and dynamic programming
- Results in state feedback controller $u=-L(k)x(k)$ with $L(k)$ determined by solution $S(k)$ of Riccati recursion

1.2 Completing the squares

- Find u that minimizes $x^\mathsf{T} Q_x x + 2x^\mathsf{T} Q_{xu} u + u^\mathsf{T} Q_u u$ with Q_u positive definite
- Let L be such that $Q_u L = Q_{xu}^\mathsf{T}$. Then

$$x^\mathsf{T} Q_x x + 2x^\mathsf{T} Q_{xu} u + u^\mathsf{T} Q_u u =$$
$$x^\mathsf{T}(Q_x - L^\mathsf{T} Q_u L)x + (u + Lx)^\mathsf{T} Q_u (u + Lx)$$

  is minimized for $u = -Lx$

  and minimum value is $x^\mathsf{T}(Q_x - L^\mathsf{T} Q_u L)x$
- If Q_u is positive definite, then $L = Q_u^{-1} Q_{xu}^\mathsf{T}$

1.3 Dynamic programming

- **Principle of optimality**: From any point on optimal trajectory, remaining trajectory is also optimal

  $\rightarrow$ allows to determine best control law over period $[t_2, t_3]$ independent of how state at t_2 was reached

- For N-step problem:
  - start from end at time $k = N$
  - now we can determine best control law for last step independent of how state at time $N-1$ was reached
  - iterate backward in time to initial time $k = 0$

2. Kalman filtering

- LQ control requires full state information
- In practice: only output measured
  $\rightarrow$ how to estimate states from noisy measurements of output?
- Consider system

$$x(k+1) = \Phi x(k) + \Gamma u(k) + v(k)$$
$$y(k) = Cx(k) + e(k)$$

  with v, e Gaussian zero-mean white noise process with

$$\mathsf{E}\left[v(k)v^\mathsf{T}(k)\right] = R_1, \quad \mathsf{E}\left[e(k)e^\mathsf{T}(k)\right] = R_2, \quad \mathsf{E}\left[v(k)e^\mathsf{T}(k)\right] = R_{12}$$

  and $x(0)$ Gaussian distributed with

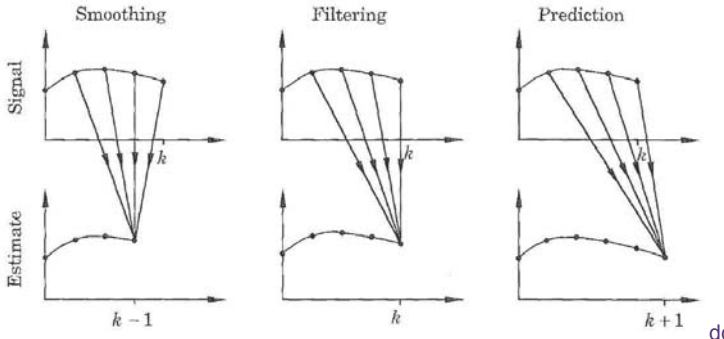
$$\mathsf{E}\left[x(0)\right] = m_0$$
$$\mathrm{cov}\left(x(0)\right) := \mathsf{E}\left[\left(x(0) - \mathsf{E}\left[x(0)\right]\right)\left(x(0) - \mathsf{E}\left[x(0)\right]\right)^\mathsf{T}\right] = R_0$$

2.1 Problem formulation

- Given the data

$$Y_k = \{y(i), u(i) \ : \ 0 \le i \le k\}$$

  find the "best" (to be defined) estimate $\hat{x}(k+m)$ of $x(k+m)$.
  ($m = 0$ filtering, $m > 0$ prediction, $m < 0$ smoothing)



Some history

Norbert Wiener: Filtering, prediction, and smoothing using integral equations. Spectral factorizations.

Rudolf E. Kalman: Filtering, prediction, and smoothing using state-space formulas. Riccati equations.

2.2 Kalman filter structure

- Goal is to estimate $x(k+1)$ by linear combination of previous inputs and outputs
- Estimator (cf. lecture on observers):

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k-1) + \Gamma u(k) + K(k)\big(y(k) - C\hat{x}(k|k-1)\big)$$

with $\hat{x}(k+1|k)$ estimate of state x at sample step $k+1$ using information available at step k

- Error dynamics $\tilde{x} = x - \hat{x}$ governed by

$$\begin{aligned}\tilde{x}(k+1) &= x(k+1) - \hat{x}(k+1|k)\\ &= \Phi x(k) + \Gamma u(k) + v(k) - \Phi\hat{x}(k|k-1) - \Gamma u(k) - K(k)\big(y(k) - C\hat{x}(k|k-1)\big)\\ &= \Phi x(k) + v(k) - \Phi\hat{x}(k|k-1) - K(k)\big(Cx(k) + e(k) - C\hat{x}(k|k-1)\big)\\ &= (\Phi - K(k)C)\tilde{x}(k) + v(k) - K(k)e(k)\end{aligned}$$

2.3 Determination of Kalman gain

- Error dynamics: $\tilde{x}(k+1) = (\Phi - K(k)C)\tilde{x}(k) + v(k) - K(k)e(k)$
- If $\hat{x}(0) = m_0$, then $\mathsf{E}\big[\tilde{x}(0)\big] = \mathsf{E}\big[x(0) - \hat{x}(0)\big] = \mathsf{E}\big[x(0)\big] - m_0 = 0$ and thus $\mathsf{E}\big[\tilde{x}(k)\big] = 0$ for all k
- How to choose $K(k)$? $\rightarrow$ minimize covariance of $\tilde{x}(k)$
- We have

$$\begin{aligned}\mathrm{cov}\,(\tilde{x}(k)) &= \mathsf{E}\left[\big(\tilde{x}(k) - \mathsf{E}\,[\tilde{x}(k)]\big)\,\big(\tilde{x}(k) - \mathsf{E}\,[\tilde{x}(k)]\big)^{\mathsf{T}}\right]\\ &= \mathsf{E}\left[\tilde{x}(k)\,\tilde{x}^{\mathsf{T}}(k)\right]\end{aligned}$$

- So if we define $P(k) = \mathsf{E}\left[\tilde{x}(k)\tilde{x}^{\mathsf{T}}(k)\right]$, then we have to determine Kalman gain such that $P(k)$ is minimized

2.3 Determination of Kalman gain (continued)

- Error dynamics: $\tilde{x}(k+1) = (\Phi - K(k)C)\tilde{x}(k) + v(k) - K(k)e(k)$
- We have

$$\begin{aligned}P(k+1) &= \mathsf{E}\left[\tilde{x}(k+1)\tilde{x}^{\mathsf{T}}(k+1)\right]\\ &= \mathsf{E}\left[\Big((\Phi - K(k)C)\tilde{x}(k) + v(k) - K(k)e(k)\Big)\Big((\Phi - K(k)C)\tilde{x}(k) + v(k) - K(k)e(k)\Big)^{\mathsf{T}}\right]\end{aligned}$$

- Since $\tilde{x}(k)$ is independent of $v(k)$ and $e(k)$, this results in

$$\begin{aligned}P(k+1) &= \mathsf{E}\Big[(\Phi - K(k)C)\tilde{x}(k)\tilde{x}^{\mathsf{T}}(k)(\Phi - K(k)C)^{\mathsf{T}}\\ &\qquad + v(k)v^{\mathsf{T}}(k) + K(k)e(k)e^{\mathsf{T}}(k)K^{\mathsf{T}}(k) - v(k)e^{\mathsf{T}}(k)K^{\mathsf{T}}(k) - K(k)e(k)v^{\mathsf{T}}(k)\Big]\\ &= (\Phi - K(k)C)\underbrace{\mathsf{E}\left[\tilde{x}(k)\tilde{x}^{\mathsf{T}}(k)\right]}_{P(k)}(\Phi - K(k)C)^{\mathsf{T}} + R_1 + K(k)R_2K^{\mathsf{T}}(k)\\ &\qquad - R_{12}K^{\mathsf{T}}(k) - K(k)R_{12}^{\mathsf{T}}\end{aligned}$$

2.3 Determination of Kalman gain (continued)

- So

$$\begin{aligned}P(k+1) &= (\Phi - K(k)C)P(k)(\Phi - K(k)C)^{\mathsf{T}} + R_1 + K(k)R_2K^{\mathsf{T}}(k) - R_{12}K^{\mathsf{T}}(k) - K(k)R_{12}^{\mathsf{T}} \\ &= K(k)\big(CP(k)C^{\mathsf{T}} + R_2\big)K^{\mathsf{T}}(k) - \big(\Phi P(k)C^{\mathsf{T}} + R_{12}\big)K^{\mathsf{T}}(k) - K(k)\big(CP^{\mathsf{T}}(k)\Phi^{\mathsf{T}} + R_{12}^{\mathsf{T}}\big) \\ &\quad + \big(\Phi P(k)\Phi^{\mathsf{T}} + R_1\big)\end{aligned}$$

- Minimize $P(k+1)$ with $K(k)$ as decision variable
- $P(k+1)$ is quadratic function of $K(k)$
- Use completing-of-squares solution with $u = K^{\mathsf{T}}(k)$ and $x = I$:

$$\underbrace{K(k)}_{u^{\mathsf{T}}}\underbrace{\big(CP(k)C^{\mathsf{T}} + R_2\big)}_{Q_u}\underbrace{K^{\mathsf{T}}(k)}_{u} + \underbrace{\big(-\Phi P(k)C^{\mathsf{T}} - R_{12}\big)}_{Q_{xu}}\underbrace{K^{\mathsf{T}}(k)}_{u}$$

$$+\underbrace{K(k)}_{u^{\mathsf{T}}}\underbrace{\big(-CP^{\mathsf{T}}(k)\Phi^{\mathsf{T}} - R_{12}^{\mathsf{T}}\big)}_{Q_{xu}^{\mathsf{T}}} + \underbrace{\big(\Phi P(k)\Phi^{\mathsf{T}} + R_1\big)}_{Q_x}$$

2.3 Determination of Kalman gain (continued)

- Results in:

$$\begin{aligned}K^{\mathsf{T}}(k) &= -L(k)x = -L(k) \\ L(k) &= Q_u^{-1}Q_{xu}^{\mathsf{T}} = \big(CP(k)C^{\mathsf{T}} + R_2\big)^{-1}\big(-\Phi P(k)C^{\mathsf{T}} - R_{12}\big)^{\mathsf{T}}\end{aligned}$$

- So

$$K(k) = -L^{\mathsf{T}}(k) = \big(\Phi P(k)C^{\mathsf{T}} + R_{12}\big)\big(CP(k)C^{\mathsf{T}} + R_2\big)^{-1}$$

- Furthermore,

$$\begin{aligned}P(k+1) &= Q_x - L^{\mathsf{T}}Q_uL \\ &= \Phi P(k)\Phi^{\mathsf{T}} + R_1 - K(k)\big(CP(k)C^{\mathsf{T}} + R_2\big)K^{\mathsf{T}}(k) \\ &= \Phi P(k)\Phi^{\mathsf{T}} + R_1 - \big(\Phi P(k)C^{\mathsf{T}} + R_{12}\big)\big(CP(k)C^{\mathsf{T}} + R_2\big)^{-1}\big(CP(k)\Phi^{\mathsf{T}} + R_{12}^{\mathsf{T}}\big)\end{aligned}$$

2.3 Determination of Kalman gain (continued)

- Initial value: $P(0) = \mathsf{E}\left[\tilde{x}(0)\tilde{x}^{\mathsf{T}}(0)\right]$

$$\begin{aligned}&= \mathsf{E}\left[\big(x(0) - \hat{x}(0)\big)\big(x(0) - \hat{x}(0)\big)^{\mathsf{T}}\right] \\ &= \mathsf{E}\left[\big(x(0) - m_0\big)\big(x(0) - m_0\big)^{\mathsf{T}}\right] \\ &= \mathsf{cov}\,(x(0)) = R_0\end{aligned}$$

- **Overall solution**:

$$\begin{aligned}\hat{x}(k+1|k) &= \Phi\hat{x}(k|k-1) + \Gamma u(k) + K(k)\big(y(k) - C\hat{x}(k|k-1)\big) \\ K(k) &= \big(\Phi P(k)C^{\mathsf{T}} + R_{12}\big)\big(CP(k)C^{\mathsf{T}} + R_2\big)^{-1} \\ P(k+1) &= \Phi P(k)\Phi^{\mathsf{T}} + R_1 - K(k)\big(CP(k)C^{\mathsf{T}} + R_2\big)K^{\mathsf{T}}(k) \\ P(0) &= R_0 \\ \hat{x}(0|-1) &= m_0\end{aligned}$$

2.4 Common equivalent implementation

- **Step 0.** (Initialization)

$$\begin{aligned}P(0|-1) &= P(0) = R_0 \\ \hat{x}(0|-1) &= m_0\end{aligned}$$

Covariance and mean of initial state.

2.4 Common equivalent implementation (continued)

- **Step 1.** (Corrector - use the most recent measurement)

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\left(y(k) - C\hat{x}(k|k-1)\right)$$
$$K(k) = P(k|k-1)C^{\mathsf{T}}\left(CP(k|k-1)C^{\mathsf{T}} + R_2\right)^{-1}$$
$$P(k|k) = P(k|k-1) - K(k)CP(k|k-1)$$

  Update estimate with $y(k)$, compute Kalman gain, update error covariance.

- **Step 2.** (One-step predictor)

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k) + \Gamma u(k)$$
$$P(k+1|k) = \Phi P(k|k)\Phi^{\mathsf{T}} + R_1$$

  Project the state and error covariance ahead.

Iterate Step 1 and 2, increase k.

2.5 Comments on Kalman filter solution

- From the Kalman gain $K(k)$ update equation, we see that a large R_2 (much measurement noise) leads to low influence of error on estimate.
- The $P(k)$ estimation error covariance is a measure of the uncertainty of the estimate. It is updated as

$$P(k+1) = \Phi P(k)\Phi^{\mathsf{T}} + R_1 - K(k)\left(CP(k)C^{\mathsf{T}} + R_2\right)K^{\mathsf{T}}(k)$$

- The first two terms on the right represent the natural evolution of the uncertainty, the last term shows how much uncertainty the Kalman filter removes.

2.5 Comments on Kalman filter solution (continued)

- The Kalman filter gives an unbiased estimate, i.e.,

$$\mathsf{E}\left[\hat{x}(k|k)\right] = \mathsf{E}\left[\hat{x}(k|k-1)\right] = \mathsf{E}\left[x(k)\right]$$

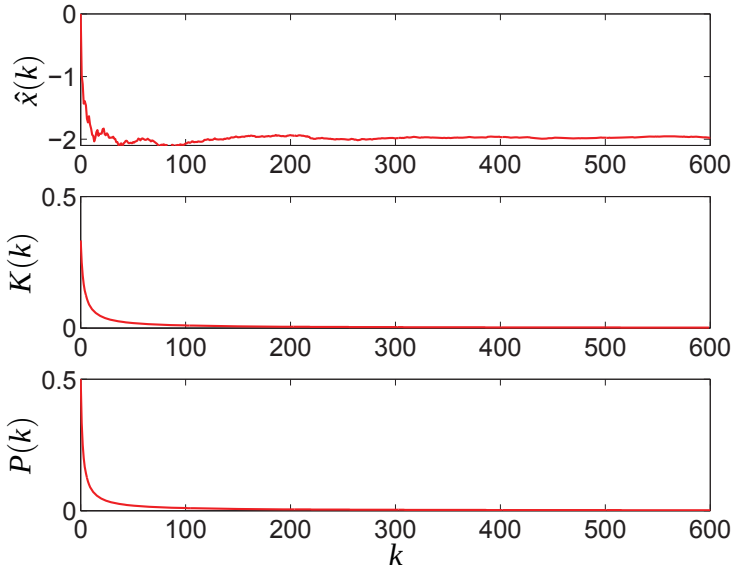
- If the noise is uncorrelated with $x(0)$, then the Kalman filter is optimal, i.e., no other *linear* filter gives a smaller variance of the estimation error.
  (For non-Gaussian assumptions, nonlinear filters, particle filters or moving-horizon estimators do a much better job.)

2.6 Example

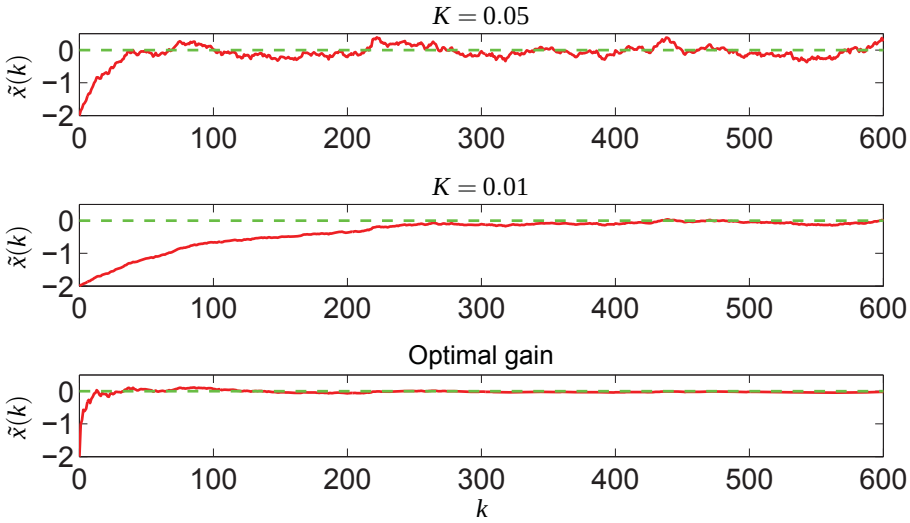
- Discrete-time system $x(k+1) = x(k)$
  $y(k) = x(k) + e(k)$
  $\rightarrow$ constant state, to be reconstructed from noisy measurements
- Measurement noise e has standard deviation σ (so $R_2 = \sigma^2$)
  $x(0)$ has covariance $R_0 = 0.5$
  No process noise v, so $R_1 = 0$ and $R_{12} = 0$
- Kalman filter is given by

$$\hat{x}(k+1|k) = \hat{x}(k|k-1) + K(k)\left(y(k) - \hat{x}(k|k-1)\right)$$
$$K(k) = \frac{P(k)}{P(k)+\sigma^2}$$
$$P(k+1) = P(k) - K(k)(P(k)+\sigma^2)K^{\mathsf{T}}(k) = \frac{\sigma^2 P(k)}{P(k)+\sigma^2}$$
$$P(0) = R_0 = 0.5 \qquad \hat{x}(0|-1) = m_0 = 0$$

2.6 Example (continued)



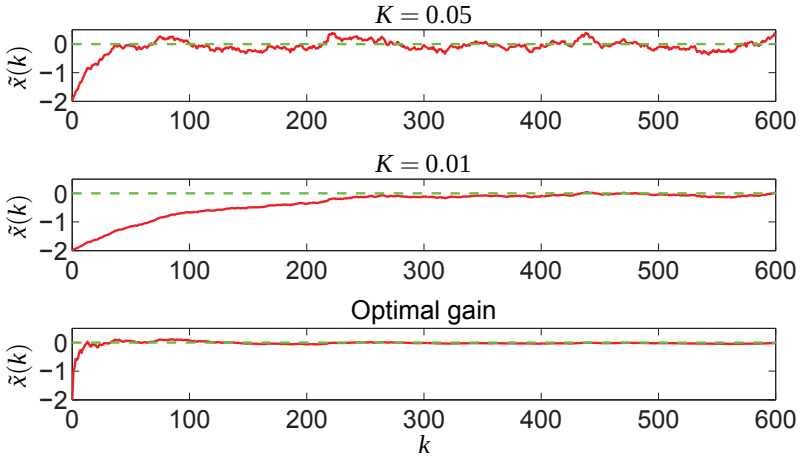
2.6 Example (continued)



Example – To think about

1. What is the problem with a large (small) K in the observer?
2. What does the Kalman filter do?
3. How would you change $P(0)$ in the Kalman filter to get a smoother (but slower) transient in $\tilde{x}(k)$?
4. In practice, R_1, R_2, and R_0 are often tuning parameters. What are their influence on the estimate?

Example – To think about (continued)



- Large fixed $K \rightarrow$ rapid initial convergence, but large steady-state variance
  Small fixed $K \rightarrow$ slower convergence, but better performance in steady state

Steady-state Kalman gain

- Recall:

$$P(k+1) = \Phi P(k)\Phi^\mathsf{T} + R_1 - \left(\Phi P(k) C^\mathsf{T} + R_{12}\right)\left(CP(k)C^\mathsf{T} + R_2\right)^{-1}\left(CP(k)\Phi^\mathsf{T} + R_{12}^\mathsf{T}\right)$$

- Steady-state solution given by

$$\bar{P} = \Phi \bar{P} \Phi^\mathsf{T} + R_1 - \left(\Phi \bar{P} C^\mathsf{T} + R_{12}\right)\left(C\bar{P}C^\mathsf{T} + R_2\right)^{-1}\left(C\bar{P}\Phi^\mathsf{T} + R_{12}^\mathsf{T}\right)$$

  $\rightarrow$ is also Riccati equation!

- Note: compare with steady-state LQ controller:

$$\bar{S} = \Phi^\mathsf{T} \bar{S} \Phi + Q_1 - \left(\Phi^\mathsf{T} \bar{S} \Gamma + Q_{12}\right)^\mathsf{T}\left(\Gamma^\mathsf{T} \bar{S} \Gamma + Q_2\right)^{-1}\left(\Gamma^\mathsf{T} \bar{S} \Phi + Q_{12}^\mathsf{T}\right)$$

Duality

- Equivalence between LQ control problem and Kalman filter state estimation problem

LQ	Kalman
k	$N-k$
Φ	Φ^T
Γ	C^T
Q_0	R_0
Q_1	R_1
Q_{12}	R_{12}
S	P
L	K^T

How to find Kalman filter using matlab?

- Command `[KEST,L,P] = kalman(SYS,QN,RN,NN)`
- Calculates (full) Kalman estimator `KEST` for system `SYS`

```
x[n+1] = Ax[n] + Bu[n] + Gw[n]          {State equation}
y[n]   = Cx[n] + Du[n] + Hw[n] + v[n]   {Measurements}
```

  with known inputs `u`, process noise `w`, measurement noise `v`, and noise covariances

```
E{ww'} = QN,      E{vv'} = RN,      E{wv'} = NN,
```

  Note: Construct `SYS` as `SYS=ss(A,[B G],C,[D H],-1)`

- Also returns steady-state estimator gain `L` and steady-state error covariance

```
P = E{(x - x[n|n-1])(x - x[n|n-1])'}  (Riccati solution)
```


3. Linear quadratic Gaussian control

- Given discrete-time LTI system

$$x(k+1) = \Phi x(k) + \Gamma u(k) + v(k)$$
$$y(k) = Cx(k) + e(k)$$

  with

$$\mathsf{E}\left[x(0)\right] = m_0, \quad \mathsf{cov}\left(x(0)\right) = R_0, \quad \mathsf{E}\left[\begin{bmatrix} v(k) \\ e(k) \end{bmatrix}\begin{bmatrix} v(k) \\ e(k) \end{bmatrix}^\mathsf{T}\right] = \begin{bmatrix} R_1 & R_{12} \\ R_{12}^\mathsf{T} & R_2 \end{bmatrix}$$

  find linear control law $y(0), y(1), \ldots, y(k-1) \mapsto u(k)$ that minimizes

$$\mathsf{E}\left[\sum_{k=0}^{N-1}\left(x^\mathsf{T} Q_1 x + 2x^\mathsf{T} Q_{12} u + u^\mathsf{T} Q_2 u\right) + x^\mathsf{T}(N) Q_0 x(N)\right]$$

- Solution: Separation principle

3.1 Separation principle

- Makes it possible to use control law

$$u(k) = -L\hat{x}(k|k-1)$$

(so $u(k) = -Lx(k) + L\tilde{x}(k)$) with closed-loop dynamics

$$x(k+1) = \Phi x(k) - \Gamma L x(k) + \Gamma L\tilde{x}(k) + v(k)$$

and to view term $\Gamma L\tilde{x}(k)$ as part of noise

$\rightarrow$ solve LQ problem and estimation problem separately

Proof of separation principle

- Solution of optimal observer design problem does not depend on input u
  So using state feedback does not influence optimality
  $\rightarrow$ Kalman filter still optimal
- Using $u(k) = -L(k)\hat{x}(k|k-1)$ results in closed-loop system

$$\hat{x}(k+1|k) = \big(\Phi - \Gamma L(k)\big)\hat{x}(k|k-1) + K(k)\underbrace{\big(y - C\hat{x}(k|k-1)\big)}_{w(k)}$$

- It can be shown that for optimal $K(k)$, $w(k)$ is white noise
- So dynamics become

$$\hat{x}(k+1|k) = (\Phi - \Gamma L(k))\hat{x}(k|k-1) + K(k)w(k)$$

Proof of separation principle (continued)

- For simplicity we assume $Q_0 = 0$
- For $u = -L\hat{x}$, control design problem in terms of $\hat{x}$ and $\tilde{x}$ becomes

$$\min_L \mathsf{E}\left[\sum_{k=0}^{N-1} x^\mathsf{T}Q_1x + 2x^\mathsf{T}Q_{12}u + u^\mathsf{T}Q_2u\right]$$

$$= \min_L \mathsf{E}\left[\sum_{k=0}^{N-1} (\hat{x}+\tilde{x})^\mathsf{T}Q_1(\hat{x}+\tilde{x}) + 2(\hat{x}+\tilde{x})^\mathsf{T}Q_{12}L\hat{x} + \hat{x}^\mathsf{T}L^\mathsf{T}Q_2L\hat{x}\right]$$

- Since it can be shown that $\mathsf{E}\left[\tilde{x}^\mathsf{T}Q\hat{x}\right] = 0$, we get

$$\min_L \mathsf{E}\left[\sum_{k=0}^{N-1} \hat{x}^\mathsf{T}Q_1\hat{x} + \tilde{x}^\mathsf{T}Q_1\tilde{x} + 2\hat{x}^\mathsf{T}Q_{12}L\hat{x} + \hat{x}^\mathsf{T}L^\mathsf{T}Q_2L\hat{x}\right]$$

- $\mathsf{E}\left[\tilde{x}^\mathsf{T}Q\tilde{x}\right]$ does not depend on L and is in fact minimal for Kalman gain K

Proof of separation principle (continued)

- Hence, we get

$$\min_L \mathsf{E}\left[\sum_{k=0}^{N-1} \hat{x}^\mathsf{T}Q_1\hat{x} + 2\hat{x}^\mathsf{T}Q_{12}L\hat{x} + \hat{x}^\mathsf{T}L^\mathsf{T}Q_2L\hat{x}\right]$$

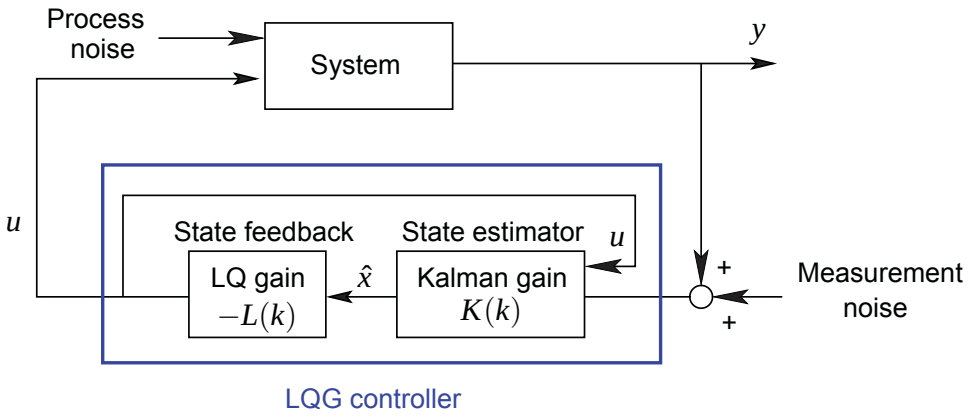
subject to

$$\hat{x}(k+1|k) = (\Phi - \Gamma L(k))\hat{x}(k|k-1) + K(k)w(k)$$

= stochastic LQ problem (but with $\hat{x}$ instead of x and with $u = -L\hat{x}$ already filled in)

$\rightarrow L(k)$ as computed before still optimal

3.2 LQG problem: Solution



LQG controller

Stationary LQG control

- Solution:

$$u(k) = -L\hat{x}(k|k-1)$$

with

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k-1) + \Gamma u(k) + K(y(k) - C\hat{x}(k|k-1)$$

- Closed-loop dynamics (with error state $\tilde{x}(k)$, see slide dc_kf_lqg.8):

$$x(k+1) = (\Phi - \Gamma L)x(k) + \Gamma L\tilde{x}(k) + v(k)$$
$$\tilde{x}(k+1) = (\Phi - KC)\tilde{x}(k) + v(k) - Ke(k)$$

or

$$\begin{bmatrix} x(k+1) \\ \tilde{x}(k+1) \end{bmatrix} = \begin{bmatrix} \Phi - \Gamma L & \Gamma L \\ 0 & \Phi - KC \end{bmatrix} \begin{bmatrix} x(k) \\ \tilde{x}(k) \end{bmatrix} + \begin{bmatrix} I \\ I \end{bmatrix} v(k) + \begin{bmatrix} 0 \\ -K \end{bmatrix} e(k)$$

$\rightarrow$ dynamics of closed-loop system determined by dynamics of LQ controller and of optimal filter

How to construct LQG controller using matlab?

- Command `RLQG = lqgreg(KEST,K)` produces LQG controller by connecting Kalman estimator `KEST` designed with `kalman` and state-feedback gain `K` designed with `dlqr`

Pros and cons of LQG

+ stabilizing
+ good robustness for SISO
+ works for MIMO
- robustness can be very bad for MIMO
- high-order controller
- how to choose weights?

Summary

- Kalman filtering
  - state estimator such that error covariance is minimized
- LQG control
  - separation principle $\rightarrow$ LQ + Kalman