Lecture 1
Overview

- course mechanics
- outline & topics
- what is a linear dynamical system?
- why study linear systems?
- some examples

Course mechanics

- all class info, lectures, homeworks, announcements on class web page:

  `www.stanford.edu/class/ee263`

course requirements:

- weekly homework
- takehome midterm exam (date TBD)
- takehome final exam (date TBD)

Prerequisites

- exposure to linear algebra ($e.g.$, Math 104)
- exposure to Laplace transform, differential equations

not needed, but might increase appreciation:

- control systems
- circuits & systems
- dynamics

Major topics & outline

- linear algebra & applications
- autonomous linear dynamical systems
- linear dynamical systems with inputs & outputs
- basic quadratic control & estimation

Linear dynamical system

continuous-time linear dynamical system (CT LDS) has the form

$$\frac{dx}{dt} = A(t)x(t) + B(t)u(t), \quad y(t) = C(t)x(t) + D(t)u(t)$$

where:

- $t \in \mathbf{R}$ denotes *time*
- $x(t) \in \mathbf{R}^n$ is the *state* (vector)
- $u(t) \in \mathbf{R}^m$ is the *input* or *control*
- $y(t) \in \mathbf{R}^p$ is the *output*

- $A(t) \in \mathbf{R}^{n\times n}$ is the *dynamics matrix*
- $B(t) \in \mathbf{R}^{n\times m}$ is the *input matrix*
- $C(t) \in \mathbf{R}^{p\times n}$ is the *output* or *sensor matrix*
- $D(t) \in \mathbf{R}^{p\times m}$ is the *feedthrough matrix*

for lighter appearance, equations are often written

$$\dot{x} = Ax + Bu, \qquad y = Cx + Du$$

- CT LDS is a first order vector *differential equation*
- also called *state equations*, or 'm-input, n-state, p-output' LDS

Some LDS terminology

- most linear systems encountered are *time-invariant*: A, B, C, D are constant, $i.e.$, don't depend on t
- when there is no input u (hence, no B or D) system is called *autonomous*
- very often there is no feedthrough, $i.e.$, $D = 0$
- when $u(t)$ and $y(t)$ are scalar, system is called *single-input, single-output* (SISO); when input & output signal dimensions are more than one, MIMO

Discrete-time linear dynamical system

discrete-time linear dynamical system (DT LDS) has the form

$$x(t+1) = A(t)x(t) + B(t)u(t), \qquad y(t) = C(t)x(t) + D(t)u(t)$$

where

- $t \in \mathbf{Z} = \{0, \pm 1, \pm 2, \ldots\}$
- (vector) signals x, u, y are *sequences*

DT LDS is a first-order vector *recursion*

Why study linear systems?

applications arise in **many** areas, $e.g.$

- automatic control systems
- signal processing
- communications
- economics, finance
- circuit analysis, simulation, design
- mechanical and civil engineering
- aeronautics
- navigation, guidance

Usefulness of LDS

- depends on availability of **computing power**, which is large & increasing exponentially
- used for
  - analysis & design
  - implementation, embedded in real-time systems
- like DSP, was a specialized topic & technology 30 years ago

Origins and history

- parts of LDS theory can be traced to 19th century
- builds on classical circuits & systems (1920s on) (transfer functions . . .) but with more emphasis on linear algebra
- first engineering application: aerospace, 1960s
- transitioned from specialized topic to ubiquitous in 1980s (just like digital signal processing, information theory, . . .)

Nonlinear dynamical systems

many dynamical systems are **nonlinear** (a fascinating topic) so why study **linear** systems?

- most techniques for nonlinear systems are based on linear methods
- methods for linear systems often work unreasonably well, in practice, for nonlinear systems
- if you don't understand linear dynamical systems you certainly can't understand nonlinear dynamical systems

Examples (ideas only, no details)

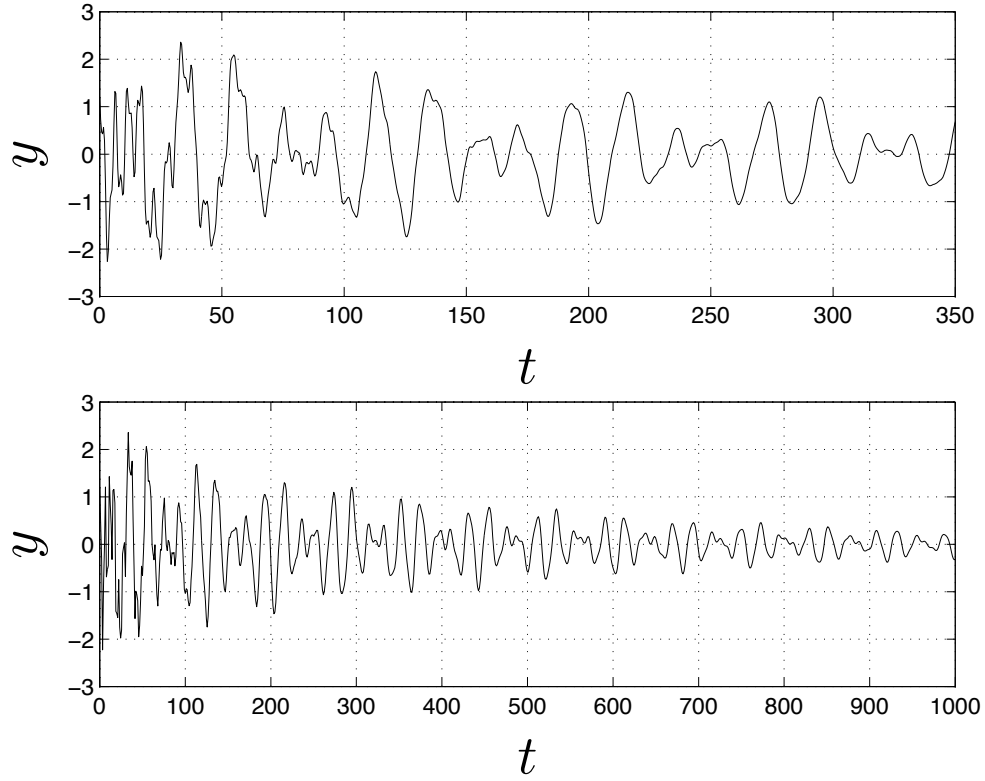
- let's consider a specific system

$$\dot{x} = Ax, \qquad y = Cx$$

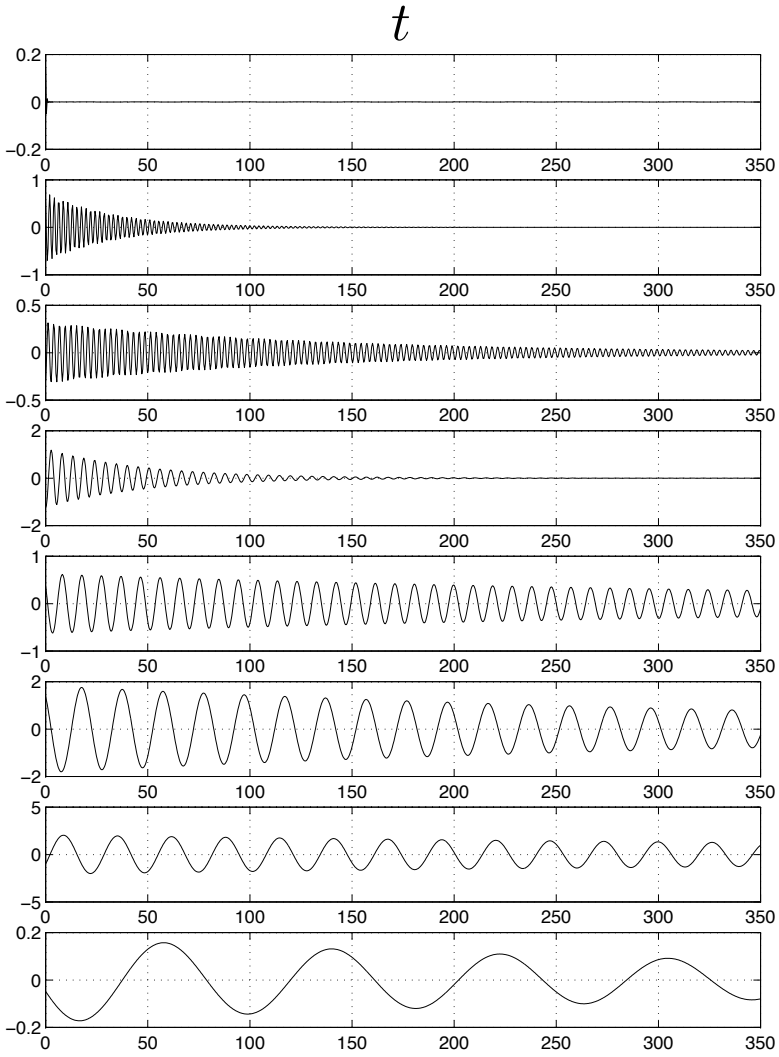
  with $x(t) \in \mathbf{R}^{16}$, $y(t) \in \mathbf{R}$ (a '16-state single-output system')

- model of a lightly damped mechanical system, but it doesn't matter

typical output:



- output waveform is very complicated; looks almost random and unpredictable
- we'll see that such a solution can be decomposed into much simpler (modal) components



(idea probably familiar from ‘poles’)

Input design

add two inputs, two outputs to system:

$$\dot{x} = Ax + Bu, \quad y = Cx, \quad x(0) = 0$$

where $B \in \mathbf{R}^{16\times 2}$, $C \in \mathbf{R}^{2\times 16}$ (same A as before)

problem: find appropriate $u : \mathbf{R}_+ \to \mathbf{R}^2$ so that $y(t) \to y_{\rm des} = (1, -2)$

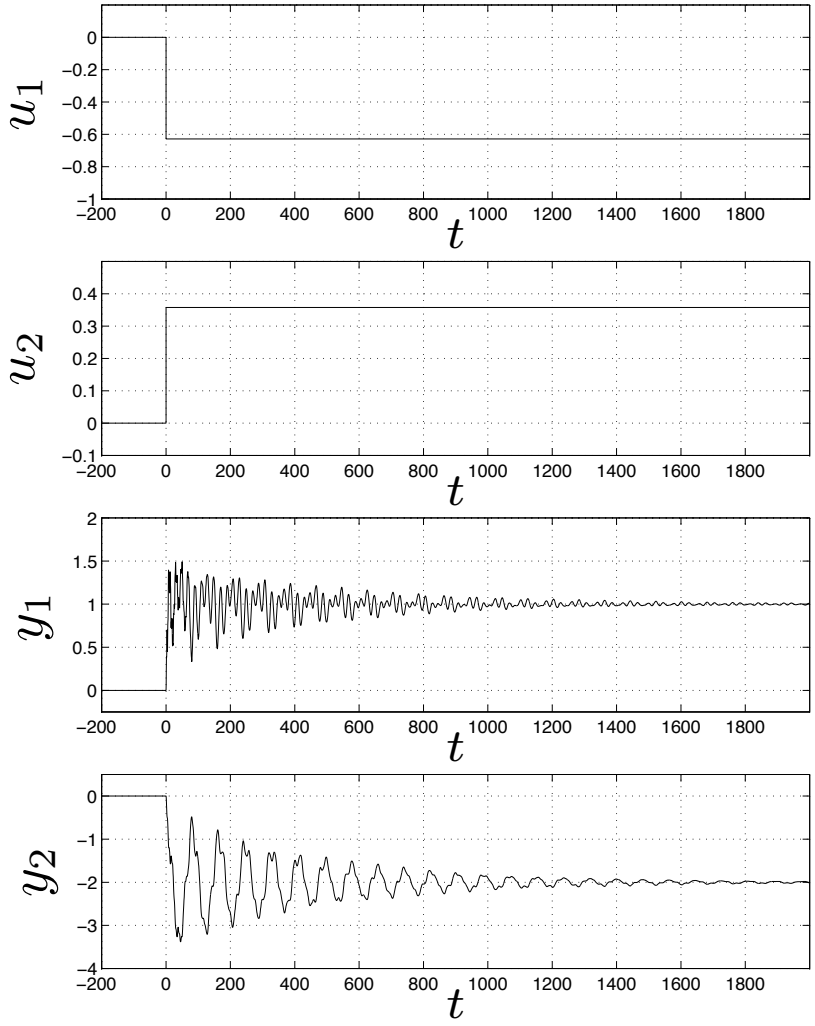
simple approach: consider static conditions (u, x, y constant):

$$\dot{x} = 0 = Ax + Bu_{\rm static}, \quad y = y_{\rm des} = Cx$$

solve for u to get:

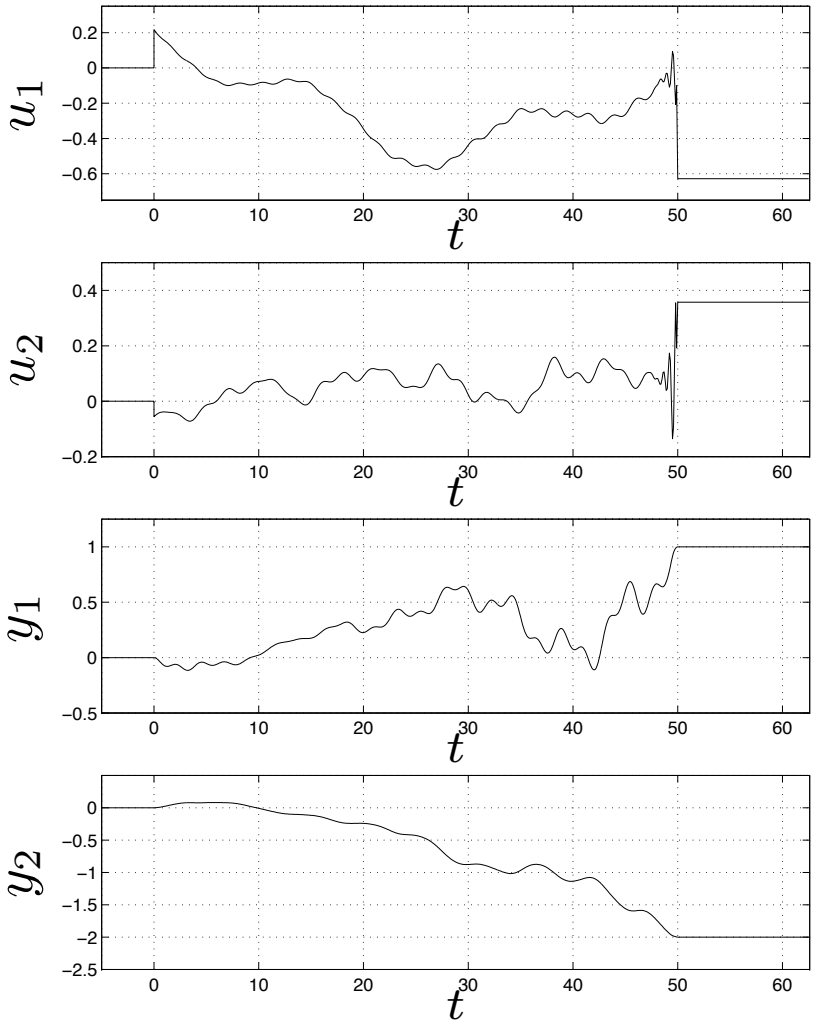
$$u_{\rm static} = \left(-CA^{-1}B\right)^{-1} y_{\rm des} = \begin{bmatrix} -0.63 \\ 0.36 \end{bmatrix}$$

let's apply $u = u_{\rm static}$ and just wait for things to settle:



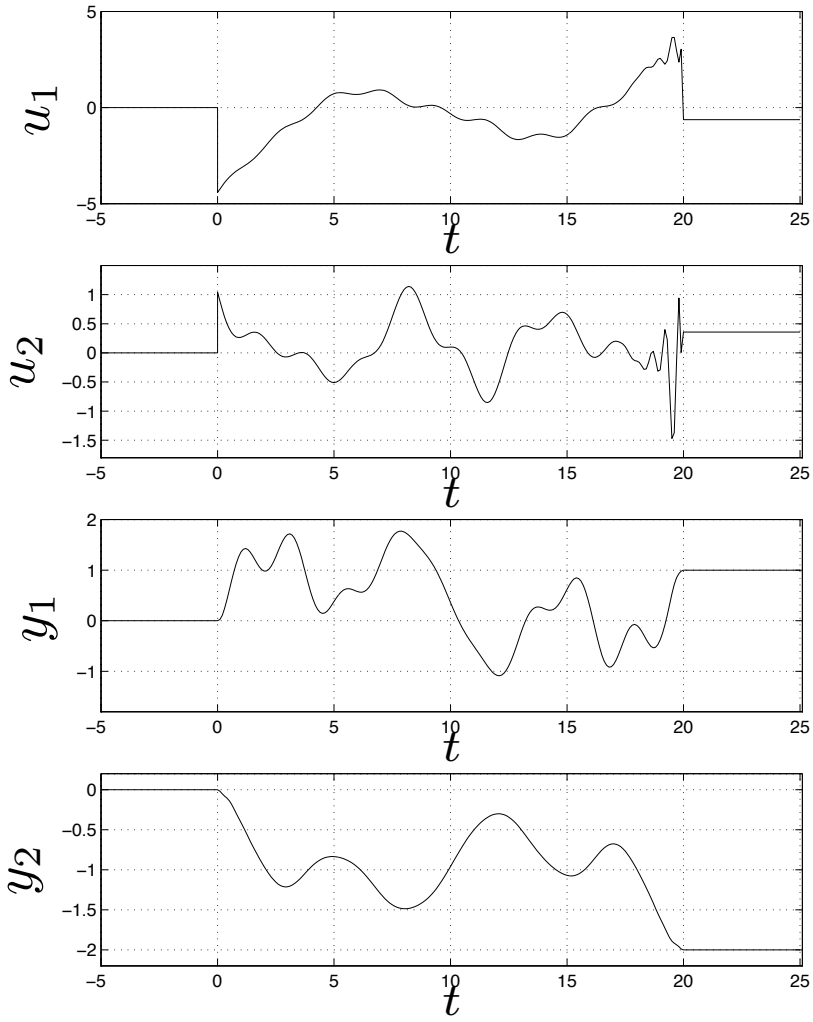
. . . takes about 1500 sec for $y(t)$ to converge to $y_{\rm des}$

using very clever input waveforms (EE263) we can do much better, $e.g.$



. . . here y converges *exactly* in 50 sec

in fact by using larger inputs we do still better, $e.g.$

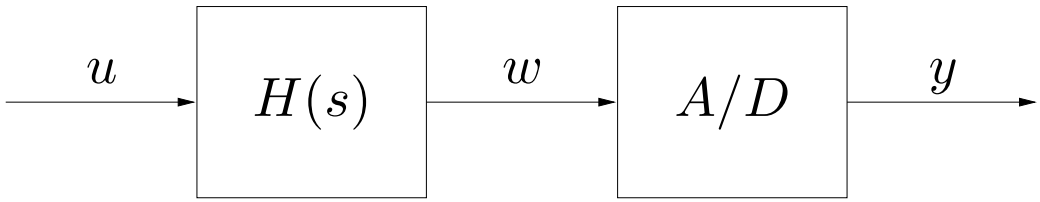


. . . here we have (exact) convergence in 20 sec

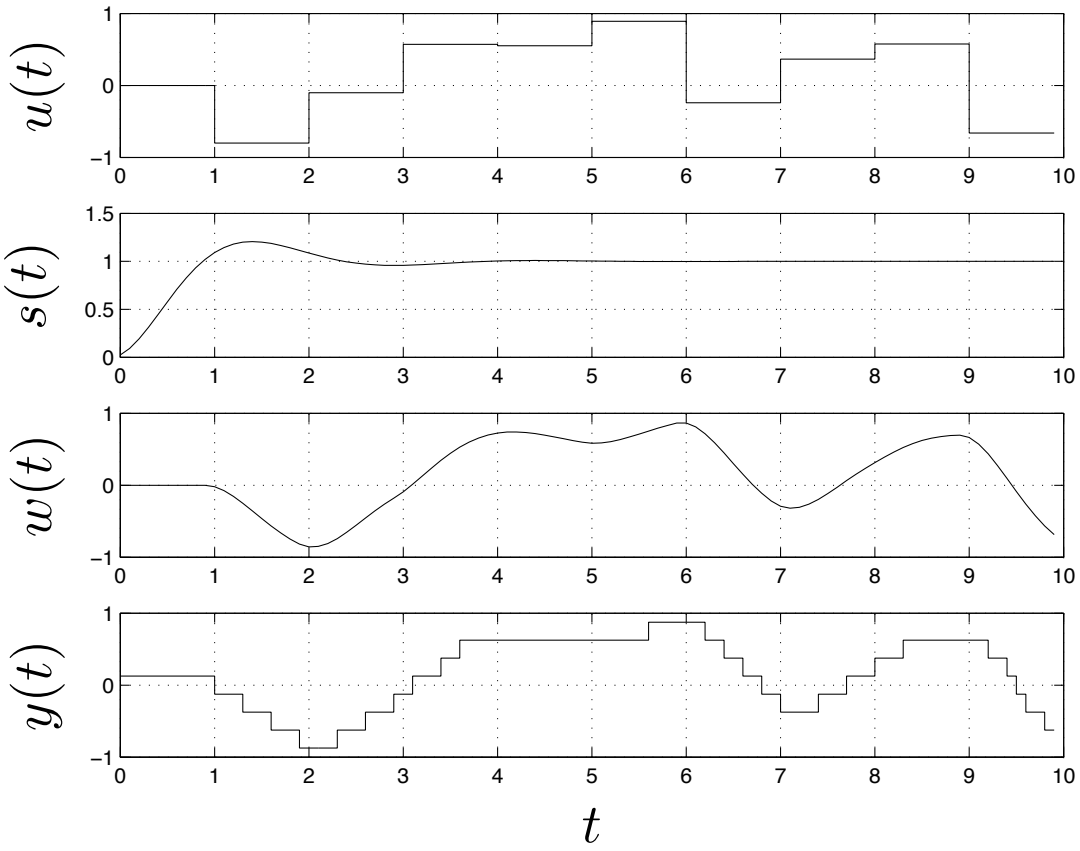
in this course we'll study

- how to synthesize or design such inputs
- the tradeoff between size of u and convergence time

Estimation / filtering



- signal u is piecewise constant (period $1\,\mathrm{sec}$)
- filtered by 2nd-order system $H(s)$, step response $s(t)$
- A/D runs at 10Hz, with 3-bit quantizer



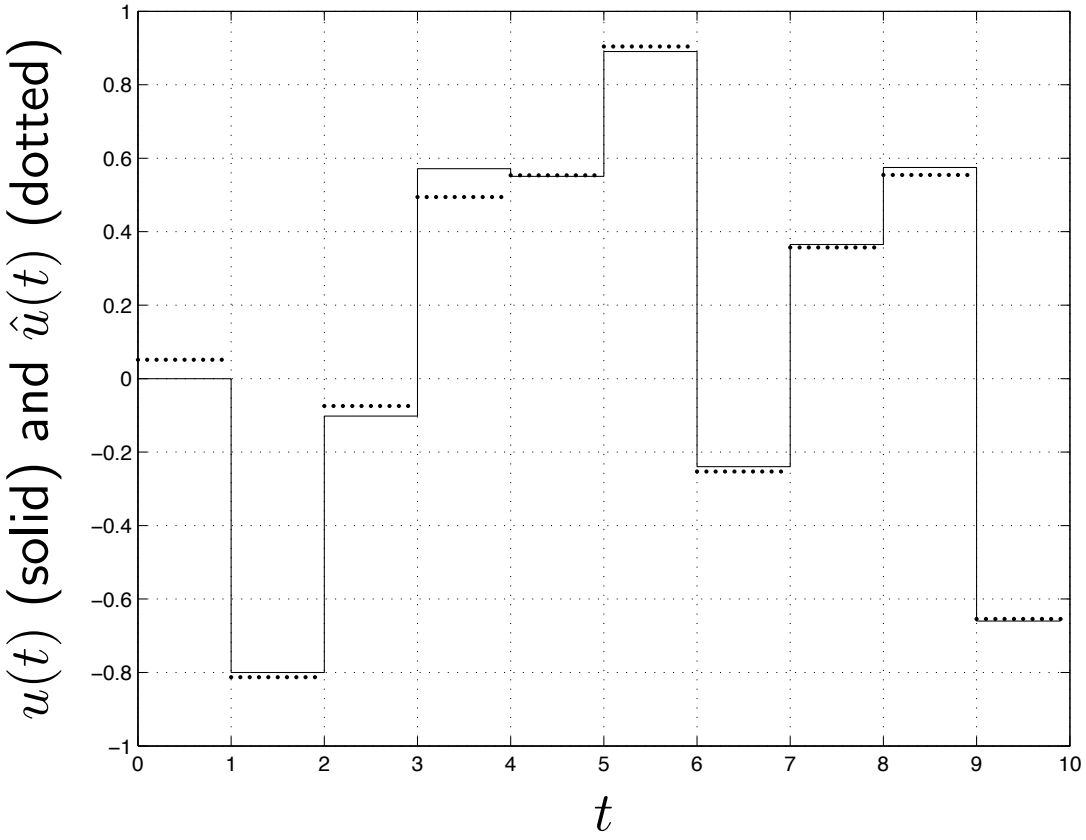
problem: estimate original signal u, given quantized, filtered signal y

simple approach:

- ignore quantization
- design equalizer $G(s)$ for $H(s)$ $(i.e.,\ GH \approx 1)$
- approximate u as $G(s)y$

. . . yields terrible results

formulate as *estimation problem* (EE263) . . .



RMS error 0.03, well **below** quantization error (!)