7. Discrete-time Systems

- Discretization of continuous-time linear systems
- Example: forces applied to a mass
- Example: computing the state
- Example: geometric growth
- Example: 2d oscillations
- Eigenvalues and eigenvectors
- Modal decomposition
- Eigenvalues of the discretization
- Solution of discrete-time LDS
- Convolution and impulse response
- Block Toeplitz matrices
- Example: controlling a hovercraft
- Comments on controlling a hovercraft

The Key Points of This Section

- if we have a *continuous-time* linear dynamical system

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

  we can discretize it to make a *discrete-time* LDS in the form

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

  which gives the behavior at sample times

- discrete-time systems are just like continuous-time ones;
  - they have a modal decomposition
  - stability is determined by eigenvalues of A; but we need $|\lambda| < 1$ not $\mathrm{Re}(\lambda) < 0$
  - they map u to y via convolution

- for control and estimation, we can form least-squares problems using the *block Toeplitz matrix* which maps u to y

Discretization

we have the continuous-time linear dynamical system (LDS)

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

we will *sample* $x(t)$

- sample times $0, h, 2h, 3h, \ldots$
  $h > 0$ is the *sampling interval*
- k'th sample is $x_d(k) = x(kh)$, and $y_d(k) = y(kh)$

we have

$$\begin{aligned} x_d(k+1) &= x(kh+h) \\ &= e^{Ah}x(kh) + \int_{kh}^{kh+h} e^{A(kh+h-\tau)}Bu(\tau)\,d\tau \\ &= e^{Ah}x_d(k) + \int_0^h e^{A(h-s)}Bu(kh+s)\,ds \end{aligned}$$

discretization

suppose input signal u is *constant between sampling times;* i.e., on intervals $[kh, (k+1)h)$

$$u(t) = u_d(k) \qquad \text{for } kh \le t < (k+1)h$$

then

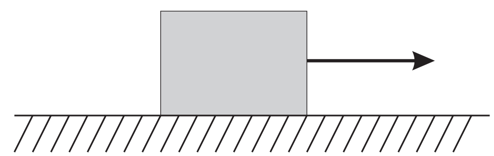
$$\begin{aligned} x_d(k+1) &= e^{Ah} x_d(k) + \int_0^h e^{A(h-s)} B u(kh+s)\, ds \\ &= e^{Ah} x_d(k) + \left(\int_0^h e^{As} B\, ds \right) u_d(k) \end{aligned}$$

so we have the *discretized system*

$$\boxed{\begin{aligned} x_d(k+1) &= A_d x_d(k) + B_d u_d(k) \qquad\qquad x_d(0) = x_0 \\ y_d(k) &= C_d x_d(k) + D_d u_d(k) \end{aligned}}$$

$$A_d = e^{Ah} \qquad B_d = \int_0^h e^{As} B\, ds \qquad C_d = C \qquad D_d = D$$

example: force on mass



Newton's law gives continuous-time LDS

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

let's compute the discretization

$$\begin{aligned} A_d &= e^{Ah} \\ &= I + Ah + \frac{1}{2}A^2h^2 + \cdots \\ &= I + Ah \\ &= \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} \end{aligned}$$

example: force on mass

also

$$\begin{aligned} B_d &= \int_0^h e^{As} B \, ds \\ &= \int_0^h \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} ds \\ &= \int_0^h \begin{bmatrix} s \\ 1 \end{bmatrix} ds = \begin{bmatrix} \frac{1}{2}h^2 \\ h \end{bmatrix} \end{aligned}$$

so the discretization is

$$\begin{aligned} x_d(k+1) &= \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} x_d(k) + \begin{bmatrix} \frac{1}{2}h^2 \\ h \end{bmatrix} u_d(k) \\ y_d(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_d(k) \end{aligned}$$

computing the discretization

we want to compute

$$A_d = e^{Ah} \qquad B_d = \int_0^h e^{As} B \, ds$$

useful fact: $\dfrac{d}{dt} e^{At} = A e^{At}$

$$\begin{aligned} B_d &= \int_0^h e^{As} B \, ds = A^{-1} \int_0^h A^{-1} e^{As} B \, ds \\ &= A^{-1} \Big[e^{At} B \Big]_0^h = A^{-1}(e^{Ah} - I)B \end{aligned}$$

so, if A is invertible

$$\boxed{B_d = A^{-1}(e^{Ah} - I)B}$$

example: computing the state

what is $x_d(4)$?

$$\begin{aligned} x_d(4) &= A_d x_d(3) + B_d u_d(3) \\ &= A_d\big(A_d x_d(2) + B_d u_d(2)\big) + B_d u_d(3) \\ &= A_d^2 x_d(2) + A_d B_d u_d(2) + B_d u_d(3) \\ &\vdots \\ &= A_d^4 x_d(0) + \begin{bmatrix} A_d^3 B_d & A_d^2 B_d & A_d B_d & B_d \end{bmatrix} \begin{bmatrix} u_d(0) \\ u_d(1) \\ u_d(2) \\ u_d(3) \end{bmatrix} \end{aligned}$$

if $h = 1$, then

$$A_d^k = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^k = \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \qquad B_d = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$$

so $A_d^k B_d = \begin{bmatrix} \frac{1}{2} + k \\ 1 \end{bmatrix}$, and

$$\begin{bmatrix} A_d^3 B_d & A_d^2 B_d & A_d B_d & B_d \end{bmatrix} = \begin{bmatrix} \frac{7}{2} & \frac{5}{2} & \frac{3}{2} & \frac{1}{2} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

discrete-time systems

a discrete-time LDS has the form

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

an *autonomous* discrete-time LDS is

$$x(t+1) = Ax(t)$$

we use either t or k as the time variable

example: geometric growth

when $x(t)$ is a scalar,

$$x(t+1) = ax(t)$$

the solution is

$$x(t) = a^t x(0)$$

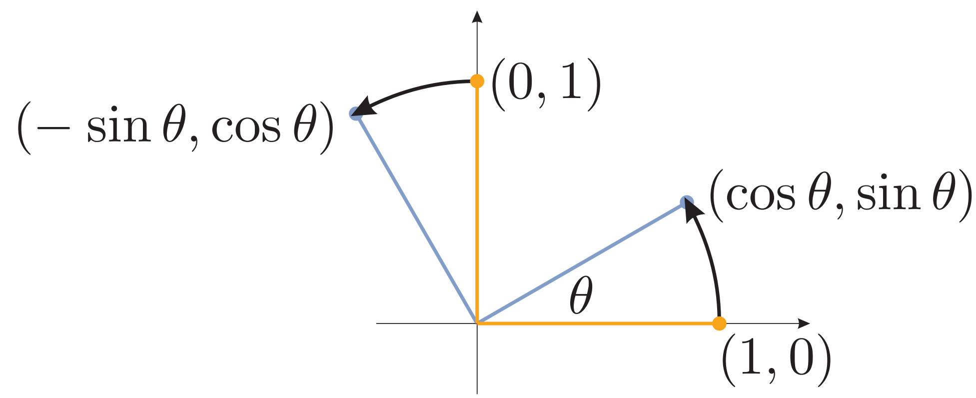
- discrete-time LDS is stable if $|a| < 1$, unstable if $|a| > 1$
- compare with continuous-time LDS: $\dot{x}(t) = \alpha x(t)$ has solution $x(t) = e^{\alpha t} x(0)$
  cts-time LDS is stable if $\alpha < 0$, unstable if $\alpha > 0$
- discretization of $\dot{x}(t) = \alpha x(t)$ is

  $$x_d(k+1) = e^{\alpha h} x_d(k)$$

  $\alpha < 0$ implies $|e^{\alpha h}| < 1$ so stability is preserved by discretization for any $h > 0$

example: 2d oscillations

$$x(t+1) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} x(t)$$



so

- $x(t)$ = rotation by $t\theta$ of $x(0)$
- eigenvalues of A are $e^{j\theta}$ and $e^{-j\theta}$
- A is orthogonal

Eigenvectors and Eigenvalues

if $Av = \lambda v$, then $x(0) = v$ gives

$$x(t) = A^t v = \lambda^t v$$

- if initial state is an eigenvector v, then state remains in $\operatorname{span}\{v\}$ always
- solution $x(t) = \lambda^t v$ is called a *mode* of the system
- $|\lambda| < 1$ means $\|x(t)\|$ shrinks as t increases; the mode is *stable*
- $|\lambda| > 1$ means $\|x(t)\|$ grows as t increases; the mode is *unstable*

eigenvectors and eigenvalues

if A has n linearly independent eigenvectors $v_1, v_2, \ldots, v_n$

$$Av_i = \lambda_i v_i$$

write this as

$$A \begin{bmatrix} v_1 & v_2 & \ldots & v_n \end{bmatrix} = \begin{bmatrix} v_1 & v_2 & \ldots & v_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}$$

matrix form

$$AT = T\Lambda \qquad \text{which gives } T^{-1}AT = \Lambda$$

we say T *diagonalizes* A

modal decomposition

as in continuous-time, this gives a *modal form* when A is diagonalizable

- let $\tilde{x} = T^{-1}x$, then
$$\begin{aligned}\tilde{x}(t+1) &= T^{-1}AT\tilde{x}(t)\\ &= \Lambda\tilde{x}(k)\end{aligned}$$
- this gives n independent systems $\tilde{x}_i(t+1) = \lambda_i\tilde{x}_i(t)$

stability

the autonomous system $x(t+1) = Ax(t)$ is called *stable* if

$$x(t) \to 0 \qquad \text{as } t \to \infty$$

if A is diagonalizable, modal decomposition implies

if every eigenvalue λ of A satisfies $\lvert\lambda\rvert < 1$, then $x(t+1) = Ax(t)$ is stable

fact: this is true for any A, even if A is not diagonalizable

diagonalization

- when A is diagonalizable, it's easy to compute A^t, since

$$A^t = (T\Lambda T^{-1})^t = T\Lambda^t T^{-1}$$

- also the matrix exponential

$$\begin{aligned} e^{Ah} &= I + Ah + \frac{1}{2}A^2h^2 + \dots \\ &= T(I + \Lambda h + \frac{1}{2}\Lambda^2h^2 + \dots)T^{-1} \\ &= Te^{\Lambda h}T^{-1} \end{aligned}$$

this gives the *eigenvalues of the discretization*:

λ is an eigenvalue of A if and only if $e^{\lambda h}$ is an eigenvalue of e^{Ah}

again, this is true even if A is not diagonalizable

Solution of Discrete-time LDS

solution of discrete-time LDS is just given by linear equations

$$x(t+1) = Ax(t) + Bu(t)$$

iterating this gives

$$x(t) = A^t x(0) + \begin{bmatrix} A^{t-1}B & A^{t-2}B & \dots & AB & B \end{bmatrix} \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(t-1) \end{bmatrix}$$

we can also write this as a sum

$$\boxed{x(t) = A^t x(0) + \sum_{\tau=0}^{t-1} A^{(t-1-\tau)} B u(\tau)}$$

convolution

in discrete-time, convolution takes the form

$$y(t) = \sum_{\tau=0}^{t} H(t-\tau)u(\tau)$$

- $H(t)$ is a matrix for each t
- the sequence $H(0), H(1), \ldots$ is called the *impulse response* of the system
- for state-space LDS, we have when $x(0) = 0$,

$$y(t) = \sum_{\tau=0}^{t-1} CA^{(t-1-\tau)}Bu(\tau) + Du(t)$$

so the impulse response is

$$H(t) = \begin{cases} D & \text{if } t = 0 \\ CA^{t-1}B & \text{if } t > 0 \end{cases}$$

- $H_{ij}(0), H_{ij}(1), \ldots$ is the response of output i to an impulse applied to input j

block Toeplitz matrices

we have

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ \vdots \\ y(t) \end{bmatrix} = \begin{bmatrix} D & & & & \\ CB & D & & & \\ CAB & CB & D & & \\ \vdots & & & \ddots & \\ CA^{t-1}B & CA^{t-2}B & \dots & CB & D \end{bmatrix} \begin{bmatrix} u(0) \\ u(1) \\ u(2) \\ \vdots \\ u(t) \end{bmatrix} + \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^t \end{bmatrix} x(0)$$

- this matrix gives the output sequence $y(0), y(1), \dots$ in terms of the input sequence $u(0), u(1), \dots$ and the initial state $x(0)$
- *block Toeplitz* means blocks are constant along diagonals from top-left to bottom right
- we can use this to find controllers and estimators

example: hovercraft

a hovercraft, mass 1, with thrusters in directions

$$v_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad v_2 = \begin{bmatrix} -0.5 \\ 1 \end{bmatrix} \qquad v_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

has dynamics

$$x(k+1) = \begin{bmatrix} 1 & h & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{1}{2}h^2 & 0 \\ h & 0 \\ 0 & \frac{1}{2}h^2 \\ 0 & h \end{bmatrix} \begin{bmatrix} v_1 & v_2 & v_2 \end{bmatrix} \begin{bmatrix} u_1(k) \\ u_2(k) \\ u_3(k) \end{bmatrix}$$

$$y(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x(k)$$

here

- $x_1, x_2 =$ position, velocity in **x**-direction
  $x_3, x_3 =$ position, velocity in **y**-direction
- $h =$ sample time; we'll use $h = 1$.
- $u_i(k)$ power applied to thruster i at time k

example: hovercraft

we would like to drive it through the positions

$$y(20) = \begin{bmatrix} 5 \\ 3 \end{bmatrix} \qquad y(40) = \begin{bmatrix} 10 \\ -1 \end{bmatrix} \qquad y(70) = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$$

at the above times

we have

$$y(t) = \sum_{\tau=0}^{t-1} CA^{t-1-\tau} Bu(\tau) + Du(t)$$

this gives the rows of

$$\begin{bmatrix} y(20) \\ y(40) \\ y(70) \end{bmatrix} = A_{\text{act}} \begin{bmatrix} u(0) \\ \vdots \\ u(70) \end{bmatrix}$$

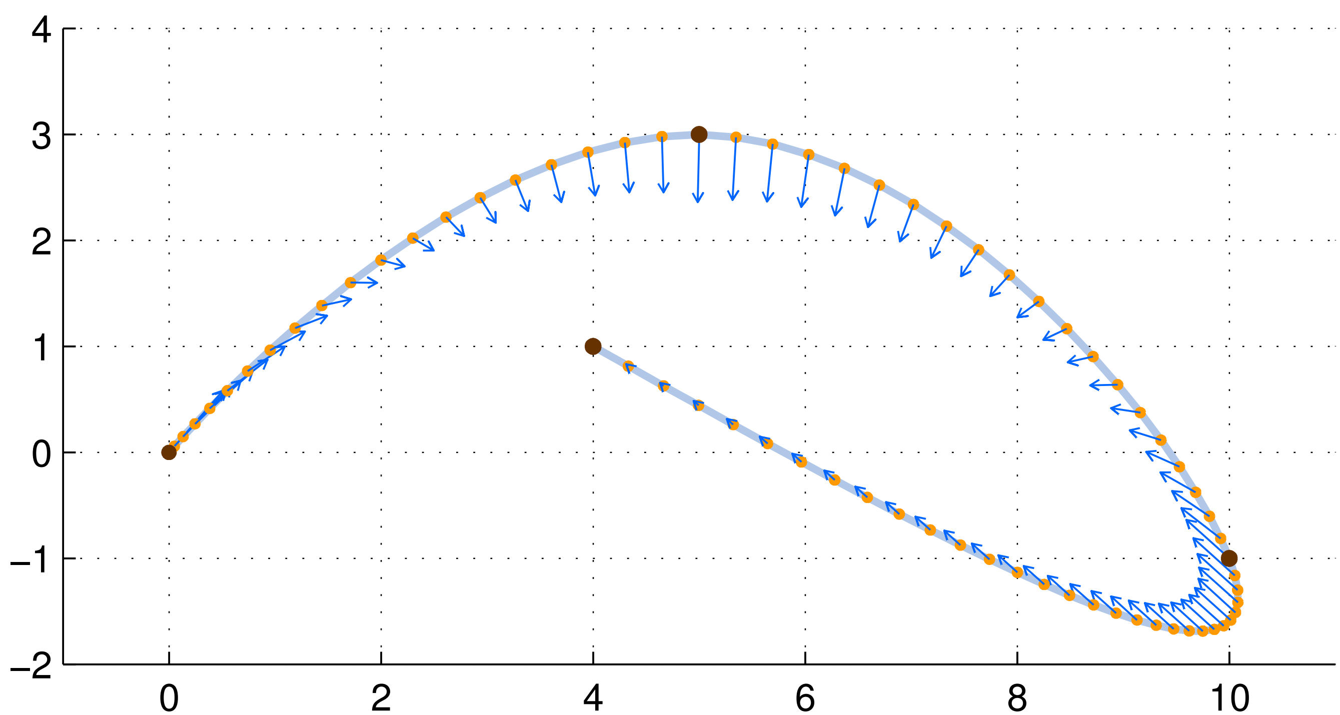
here A_{act} is 6×213.

example: hovercraft

let's find the minimum norm sequence of thrusts that meets the specifications

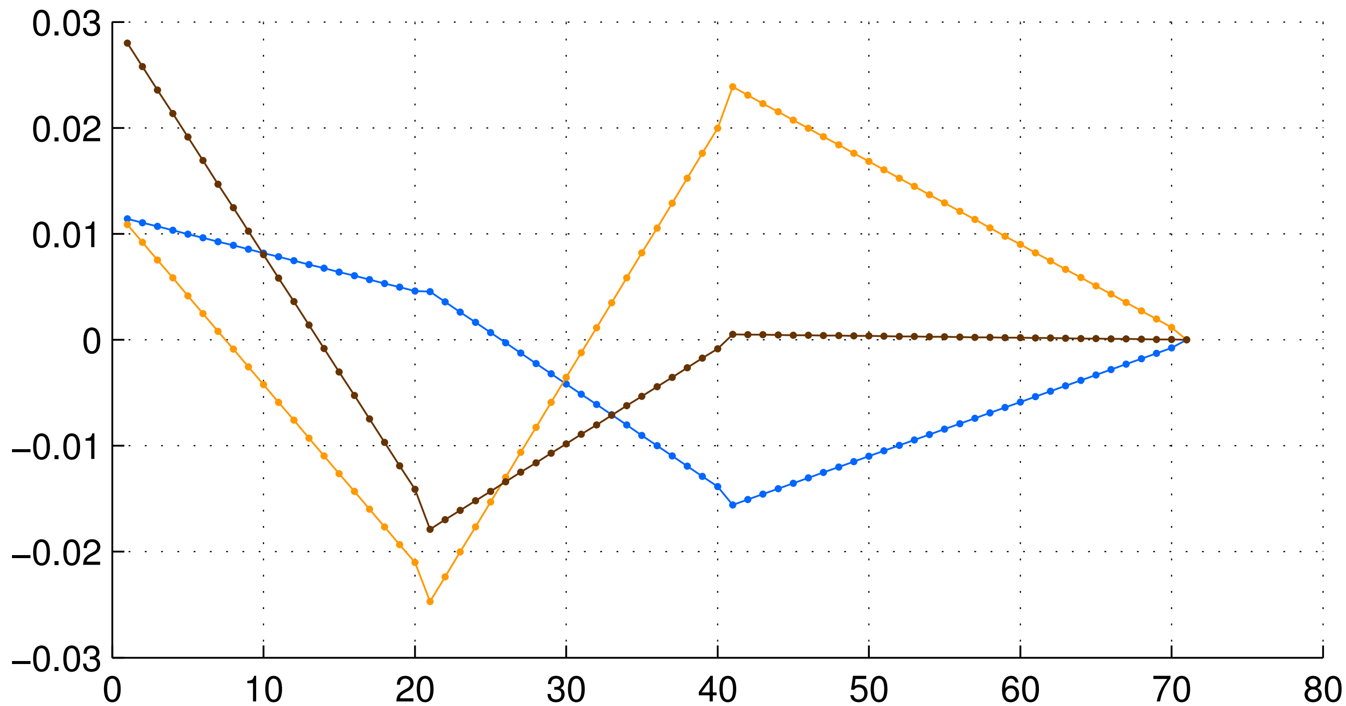
$$\begin{bmatrix} u(0) \\ \vdots \\ u(70) \end{bmatrix} = A_{\text{act}}^{\dagger} \begin{bmatrix} 5 \\ 3 \\ 10 \\ -1 \\ 4 \\ 1 \end{bmatrix}$$

hovercraft trajectory is



example: hovercraft

sequence of thrust inputs is



comments on this approach

- discrete-time systems have the wonderful property that

  the relationship between the sequence of inputs and the sequence of outputs is just a linear equation

- so we can apply

  least squares to find the inputs, given measurements of the outputs
  minimum-norm solutions to steer the output along some desired trajectory

- but

  - as the number of time-steps increases, so does the size of the least-squares problems
  - computation time grows as $(\text{number-of-time-steps})^3$
  - A_{act} contains all powers of A from 1 to the number of time-steps; so *condition numbers* grow very large
  - worst of all: controllers and estimators are *open-loop*

- we'll fix all this