

# Linear Dynamic Systems Notes

Chris Beard

10 August 2011

## Contents

<b>1</b>	<b>Lecture 2</b>	<b>3</b>
1.1	Interpretation for $y = Ax$	3
1.2	Example in notes	4
1.2.1	Linear elastic structure	4
1.2.2	Force/Torque on rigid body	4
1.2.3	Thermal System	4
<b>2</b>	<b>Lecture 3</b>	<b>4</b>
2.1	Review of Linearization (affine approximation)	4
2.1.1	Good intuition range-finding problem	4
2.2	Intepretations of $Ax = y$	4
2.2.1	Computing $C = AB$	4
2.2.2	Zero nullspace (in notes)	5
2.2.3	Interpretations of nullspace	5
2.2.4	Range	5
2.2.5	Rank of matrix	7
2.2.6	Various wrap-up items	7
<b>3</b>	<b>Lecture 5</b>	<b>7</b>
3.1	Geometric properties of orthonormal vectors	8
3.2	Orthonormal basis for $\mathbb{R}^n$	8
3.3	Expansion in orthonormal basis	8
3.4	Gram-Schmidt procedure	8
3.5	$QR$ decomposition	9
3.6	General Gram Schmidt procedure ('rank revealing QR algorithm')	9
3.7	Applications	9
3.8	Least Squares Approximation	9
<b>4</b>	<b>Lecture 6</b>	<b>9</b>
4.1	Overdetermined equations	9
4.2	Least Squares 'Solution'	10
4.3	Projection on $\mathcal{R}(A)$	10
4.4	Orthogonality principle	10
4.5	Least-squares via $QR$ factorization	10
4.6	Full $QR$ factorization	10
4.7	Applications for least squares approximations	11
4.8	BLUE: Best linear unbiased estimator	11
4.9	Range-finding example	11
4.10	Quantizer example	11
4.11	Least Squares data-fitting	12
4.12	Least-squares polynomial fitting	12

<b>5</b>	<b>Lecture 7</b>	<b>12</b>
5.1	Least-squares polynomial fitting, cont'd	12
5.2	Growing sets of regressors	12
5.3	Least-squares system identification (important topic)	13
5.4	Model order selection	13
5.4.1	Cross-validation	13
5.5	Growing sets of measurements	13
5.6	Recursive ways to do least squares	13
5.7	Fast update algorithm for recursive LS	13
5.8	Multi-objective least squares	14
<b>6</b>	<b>Lecture 8</b>	<b>14</b>
6.1	Plot of achievable objective pairs	14
6.2	Minimizing weighted-sum objective	15
6.3	Example: frictionless table	15
6.4	Regularized least-squares	15
6.5	Nonlinear least squares (NLLS) problem	16
6.6	Gauss-Newton method for NLLS	16
6.7	G-N example	17
6.8	Underdetermined linear equations	17
6.9	Least norm solution	17
<b>7</b>	<b>Lecture 9 pt 2</b>	<b>17</b>
7.1	General norm minimization with equality constraints	18
7.2	Autonomous linear dynamical systems	18
7.3	Block diagrams	18
7.4	Linear circuit example	18
<b>8</b>	<b>Lecture 10</b>	<b>18</b>
8.1	Example: Series reaction $A \rightarrow B \rightarrow C$	18
8.2	Discrete time Markov chain	19
8.3	Numerical integration of continuous system	19
8.4	Higher order linear dynamical systems ( $\dot{x} = Ax$ )	19
8.5	Example: Mechanical systems	19
8.6	Linearization near equilibrium point	19
8.7	Example: pendulum linearization	20
<b>9</b>	<b>Lecture 11</b>	<b>20</b>
9.1	Laplace transform	20
9.2	Example: Harmonic oscillator	20
9.3	Example: Double Integrator	21
9.4	Characteristic polynomial	21
9.5	Get eigenvalues of $A$ and poles of resolvent	21
9.6	Matrix exponential	21
9.7	Time transfer property	21
9.8	Application: sampling a continuous time system	22
<b>10</b>	<b>Lecture 12</b>	<b>22</b>
10.1	Stability	22
10.2	Eigenvectors and diagonalization	22
10.3	Scaling interpretation	22
10.4	Dynamic interpretation	22
10.5	Invariant set	23
10.6	Complex eigenvectors	23
10.7	Dynamic interpretation: left eigenvectors	23

10.8 Summary: . . . . .	23
10.9 Example- companion matrix . . . . .	23
<b>11 Lecture 13</b>	<b>23</b>
11.1 Example: Markov chain . . . . .	23
11.2 Diagonalization . . . . .	24
11.3 Not all matrices diagonalizable . . . . .	24
11.4 Distinct eigenvalues . . . . .	24
11.5 Diagonalization and left eigenvectors . . . . .	24
11.6 Modal form . . . . .	25
11.7 Real modal form . . . . .	25
11.8 Diagonalization simplification . . . . .	25
11.9 Simplify for analytical functions of a matrix . . . . .	25
11.10 Solution via diagonalization . . . . .	25
11.11 Interpretation . . . . .	26
11.12 Application . . . . .	26
11.13 Stability of discrete-time systems . . . . .	26
11.14 Jordan Canonical form . . . . .	26
11.15 Jordan block example . . . . .	27
<b>12 Lecture 14</b>	<b>27</b>
<b>13 Appendix</b>	<b>27</b>
13.1 Inverse, transpose properties . . . . .	27
13.2 Invertibility implications . . . . .	27
13.3 Tips . . . . .	28
<b>14 Homework assignments</b>	<b>28</b>

Following notes from the EE263 Course Reader. Orgmode  $\rightarrow$  L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>

Matlab files   Linear Algebra  
Index

## 1 Lecture 2

Official Lecture Notes

### 1.1 Interpretation for $y = Ax$

- $A$  a transformation matrix
- $y$  an observed measurement,  $x$  an unknown
- $x$  is input,  $y$  is output; for rows  $i$  and columns  $j$  in  $A$ ,  $\mathbf{i} \rightarrow \mathbf{y}$ ,  $\mathbf{j} \rightarrow \mathbf{x}$ 
  - indexed as **output**, **input**
- Lower diagonal Matrix should make you expect or have a vague thought about causality.
  - $a_{ij} = 0$  for  $i < j \Rightarrow y_i$  only depends on  $x_1, \dots, x_i$
  - $A$  is diagonal; output only depends on input
- Sparsity pattern (block of zeroes) in a matrix should have you wonder why... usually not coincidental
- $A_{35}$  positive  $\Rightarrow x_5$  increased corresponds to an increase in 3rd output,  $y_3$

## 1.2 Example in notes

### 1.2.1 Linear elastic structure

- $a_{11}$  probably positive
  - $x_1$  input gives positive push to  $y_1$  output

### 1.2.2 Force/Torque on rigid body

- Net torque/force on body is linearly related to force inputs

### 1.2.3 Thermal System

- despite normally needing a Poisson equation, the steady state heat of the different locations can be represented as a linear system  $Ax = y$

## 2 Lecture 3

### 2.1 Review of Linearization (affine approximation)

1. If  $f : \mathbf{R}^n \rightarrow \mathbf{R}^m$  is differentiable at  $x_0 \in \mathbf{R}^n$ , then  $x$  near  $x_0 \Rightarrow f(x)$  very near  $f(x_0) + Df(x_0)(x - x_0)$  where

$$Df(x_0)_{ij} = \left. \frac{\partial f_i}{\partial x_j} \right|_{x_0}$$

is the derivative (Jacobian) matrix.

2. with  $y = f(x)$ ,  $y_0 = f(x_0)$ , define ‘input deviation’  $\delta x := x - x_0$ , ‘output deviation’  $\delta y := y - y_0$
3. then we have  $\delta y \approx Df(x_0)\delta x$ 
  - i.e., we get a linear function for looking at how the output changes with small changes in the input
4. When deviations are small, they are approximately related by a linear function

#### 2.1.1 Good intuition range-finding problem

### 2.2 Interpretations of $Ax = y$

- Scaled combination of columns in  $A$ 

$$A = [a_1 a_2 \dots a_n] \rightarrow y = x_1 a_1 + x_2 a_2 + \dots + x_n a_n$$
- Looking at the rows: the multiplication is the inner product of rows and vector  $x$
- I/O ordering is backwards in control; in  $A_{ij}$ ,  $j$  refers to input and  $i$  refers to output
  - indexing is likewise backwards in block diagram indexing

$AB = I$ ;  $\tilde{a}_i^T \cdot b_j = 0$  if  $i \neq j$ , where  $\tilde{a}_i$  is the  $i$ th row in  $A$ , and  $b_j$  is  $j$ th column in  $B$

#### 2.2.1 Computing $C = AB$

How to compute this faster than using formula  $c_{ij} = \sum_{k=1} A_{ik} B_{kj}$ : have computer break it up into submatrices and do the multiplication on them

**2.2.2 Zero nullspace (in notes)**

- if 0 is only element in  $\mathcal{N}(A)$ 
  - $A$  is one-to-one
    - \* linear transformation doesn't lose information
  - columns of  $A$  are independent, and basis for their span
  - $A$  has a left inverse  $\Rightarrow$  you can use this to undo the transformation and find the input  $x$ , given the output  $y$
  - $|A^T A| \neq 0$

**2.2.3 Interpretations of nullspace**

supposing  $z \in \mathcal{N}(A)$

- Measurements
  - $z$  is undetectable from sensors
  - $x$  and  $x + z$  are indistinguishable from sensors:  $Ax = A(x + z)$
  - the nullspace characterizes ambiguity in  $x$  from measurement  $y = Ax$ 
    - \* large nullspace is bad
- $y = Ax$  is output from input  $x$ 
  - $z$  is input with no result
  - $x$  and  $x + z$  have same result
  - the nullspace characterizes freedom of input choice for given result
    - \* large nullspace is good: more room for optimization because of more input possibilities

**2.2.4 Range**

Range of  $A \in \mathbb{R}^{m \times n}$  defined as  $\mathcal{R}(A) = \{Ax | x \in \mathbb{R}^n\} \subseteq \mathbb{R}^m$

- The set of vectors that can be 'hit' by linear mapping  $y = Ax$
- span of columns of  $A$
- set of vectors  $y$  for which  $Ax = y$  has a solution
- possible sensor measurement
  - in design, you'll want to throw an exception if a measurement is outside the range; the sensor is bad
  - test for a bad 13th sensor: remove 13th row in  $A$ ; if the reduced  $y$  is in the range of the reduced matrix, the 13th sensor might not have been bad
- Onto matrices
 

$A$  is 'onto' if  $\mathcal{R}(A) = \mathbb{R}^m$

  - you can solve  $Ax = y$  for any  $y$
  - columns of  $A$  span  $\mathbb{R}^m$
  - $A$  has a right inverse  $B$  s.t.  $AB = I$ 
    - \* can do  $AB y = A(B y) = y$ : you want an  $x$  that gives you  $y$ ? Here it is.
    - \* Design procedure
  - rows of  $A$  are independent

- \* a.k.a.,  $\mathcal{N}(A^T) = \{0\}$
- $|AA^T| \neq 0$

- Interpretations of range

- supposing  $v \in \mathcal{R}(A)$ 
  - \*  $v$  reachable
  - \* else, not reachable

- Inverse

Note: square matrices are impractical for engineering. They don't let you take advantage of redundant sensors/controllers, or let you build a robust system to take care of broken sensors

- $A \in \mathbb{R}^{n \times n}$  is invertible or nonsingular if  $\det A \neq 0$ 
  - \* columns of  $A$  are basis for  $\mathbb{R}^n$
  - \* rows of  $A$  are basis for  $\mathbb{R}^n$
  - \*  $y = Ax$  has a unique solution  $x$  for every  $y \in \mathbb{R}^n$
  - \*  $A$  has left and right inverse  $A^{-1} \in \mathbb{R}^{n \times n}$ , s.t.  $AA^{-1} = A^{-1}A = I$
  - \*  $\mathcal{N}(A) = \{0\}$
  - \*  $\mathcal{R}(A) = \mathbb{R}^n$
  - \*  $\det A^T A = |AA^T| \neq 0$
- Dual basis interpretation of inverse
  - $a_i$  are columns of  $A$ , and  $\tilde{b}_i^T$  are rows of  $B = A^{-1}$ 
    - \*  $y = Ax$ , column by column, looks like  $y = x_1 a_1 + \dots + x_n a_n$ 
      - multiply both sides of  $y = Ax$  by  $A^{-1} = B$  gives  $x = By$
      - so  $x_i = \tilde{b}_i^T y$

$$\begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ a_1 & a_2 & \dots & a_n \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$x = A^{-1}y$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} \dots & \tilde{b}_1^T & \dots \\ \dots & \tilde{b}_2^T & \dots \\ \dots & \vdots & \dots \\ \dots & \tilde{b}_n^T & \dots \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = [x_1 a_1 + \dots + x_n a_n] = [(\tilde{b}_1^T y) a_1 + \dots + (\tilde{b}_n^T y) a_n]$$

Beautiful thing:

$$y = \sum_{i=1}^n (\tilde{b}_i^T y) a_i$$

### 2.2.5 Rank of matrix

Rank of  $A \in \mathbb{R}^{m \times n}$  as  $\text{rank}(A) = \dim \mathcal{R}(A)$

- $\text{rank}(A) = \text{rank}(A^T)$
- $\text{rank}(A)$  is maximum number of independent columns or rows of  $A$ :  $\text{rank}(A) \leq \min(m, n)$
- $\text{rank}(A) + \dim \mathcal{N}(A) = n$
- Conservation of degrees of freedom (dimension)
  - $\text{rank}(A)$  is dimension of set ‘hit’ by mapping  $y = Ax$
  - $\dim \mathcal{N}(A)$  is dimension of set of  $x$  ‘crushed’ to zero by  $y = Ax$
  - Example
    - \*  $A \in \mathbb{R}^{20 \times 10}$   $\text{rank}(A) = 8$ 
      - you can do 8 dimensions worth of stuff
      - 10 knobs, 2 redundant knobs, which is  $\dim \mathcal{N}(A) = 2$
- Coding interpretation of rank
  - rank of product:  $\text{rank}(BC) \leq \min\{\text{rank}(B), \text{rank}(C)\}$
  - supposedly really cool stuff based on this
  - low rank matrices let you do fast computations

### 2.2.6 Various wrap-up items

- RMS

$$\text{rms}(x) = \left( \frac{1}{n} \sum_{i=1}^n \right)^{1/2} = \frac{\|x\|}{\sqrt{n}}$$

- Inner product
  - $\langle x, y \rangle := x_1 y_1 + x_2 y_2 + \cdots + x_n y_n = x^T y$
  - interpretation of inner product signs:
    - $x^T y > 0$ : acute; roughly point in same direction
    - $x^T y < 0$ : obtuse; roughly point in opposite direction
- Orthonormal set of vectors
  - set of  $k$  vectors  $u_1, u_2, \dots, u_k \in \mathbb{R}^n$  orthonormal;  $U = [u_1 \cdots u_k]$
  - $U^T U = I_k \Leftrightarrow$  set of column vectors of  $U$  are orthonormal
  - **warning:**  $U U^T \neq I_n$  if  $k < n$ 
    - \* say  $U$  is  $10 \times 3$ ,  $U^T$  is  $3 \times 10$ , rank of  $U$  is 3  $\Rightarrow$  rank of  $U U^T$  is at most 3
    - \* but  $U U^T$  will be a  $10 \times 10$  matrix, so it can't be the identity matrix

## 3 Lecture 5

A good source for more on orthogonality at University of Minnesota

### 3.1 Geometric properties of orthonormal vectors

- columns of  $U$  are ON  $\Rightarrow$  mapping under  $U$  preserves distances

$$- w = Uz \Rightarrow \|w\| = \|z\|$$

- Also preserves inner product
- Also preserves angles
- Something like a rigid transformation

### 3.2 Orthonormal basis for $\mathbb{R}^n$

- if there are  $n$  orthonormal vectors (remember, with dimension  $n$ ), it forms an orthonormal basis for  $\mathbb{R}^n$

- $U^{-1} = U^T$

$$- \boxed{U^T U = I \Leftrightarrow U\text{'s column vectors form an orthonormal basis for } \mathbb{R}^n}$$

- $-\sum_{i=1}^n u_i u_i^T = I \in \mathbb{R}^{n \times n}$  (known as a dyad, or outer product; inner products reverses the two and gives a scalar, outer gives a matrix)
- outer products take 2 vectors, possibly of different sizes, and multiplies every combination of elements one with another

### 3.3 Expansion in orthonormal basis

- $U$  orthogonal  $\Rightarrow x = U U^T x$

$$- x = \sum_{i=1}^n (u_i^T x) u_i$$

- because  $U^T U = I$ , the thing in sum is really  $u_i u_i^T x$
- $u_i^T x$  is really a scalar, so this can be moved to the front of  $u_i$ , giving our result
- This says  $x$  is a linear combination of  $u_i$ 's

### 3.4 Gram-Schmidt procedure

- $a_1, \dots, a_k \in \mathbb{R}^n$  are LI; G-S finds ON vectors  $q_1, \dots, q_k$  s.t.

$$\text{span}(a_1, \dots, a_r) = \text{span}(q_1, \dots, q_r)$$

for  $r \leq k$

- so  $q_1, \dots, q_r$  is an ON basis for  $\text{span}(a_1, \dots, a_r)$
- Basic method: orthogonalize each vector wrt the previous ones, then normalize result

- $\tilde{q}_1 = a_1$
- normalize:  $q_1 = \tilde{q}_1 / \|\tilde{q}_1\|$
- remove  $q_1$  component from  $a_2$ :  $\tilde{q}_2 = a_2 - (q_1^T a_2) q_1$
- normalize  $q_2$
- remove  $q_1, q_2$  components:  $\tilde{q}_3 = a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2$
- normalize  $q_3$

- $a_i = (q_1^T a_i) q_1 + (q_2^T a_i) q_2 + \dots + (q_{i-1}^T a_i) q_{i-1} + \|\tilde{q}_i\| q_i$
- $= r_{1i} q_1 + r_{2i} q_2 + \dots + r_{ii} q_i$  ( $r_{ii} \geq 0$  is the length of  $\tilde{q}_i$ )



### 3.5 QR decomposition

This can be written as  $A = QR$ , where  $A \in \mathbb{R}^{n \times k}$ ,  $Q \in \mathbb{R}^{n \times k}$ ,  $R \in \mathbb{R}^{k \times k}$

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_k \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \cdots & q_k \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1k} \\ 0 & r_{22} & \cdots & r_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{kk} \end{bmatrix}$$

- $R$  triangular because computation of  $a_i$  only involves up to  $q_i$ 
  - a sort of causality, since you can calculate  $q_7$  without seeing  $q_8$
- Columns of  $Q$  are ON basis for  $\mathcal{R}(A)$

### 3.6 General Gram Schmidt procedure ('rank revealing QR algorithm')

- Basically the same, but if one of the  $\tilde{q}_i$ 's is zero (meaning  $a_i$  is dependent on previous  $a$  vectors), then just go to the next column
- referring to notes, upper staircase notation shows which vectors are dependent on previous ones (columns without the x's)
  - entries with x are 'corner' entries

### 3.7 Applications

- check if  $b \in \text{span}(a_1, a_2, \dots, a_k)$
- Factorize matrix  $A$

### 3.8 Least Squares Approximation

- Overdetermined linear equation (tall, skinny, more equations than unknowns, dimensionally redundant system of equations)

## 4 Lecture 6

On skinny, full rank matrices

### 4.1 Overdetermined equations

- Skinny, more equations than unknowns
- Given  $y = Ax$ ,  $A \in \mathbb{R}^{m \times n}$ , a randomly-chosen  $y$  in  $\mathbb{R}^m$  has 0 probability of being in the range of  $A$
- To *approximately* solve for  $y$ , minimize norm of error (residual)  $r = Ax - y$
- find  $x = x_{ls}$  (least squares approx.) that minimizes  $\|r\|$

## 4.2 Least Squares ‘Solution’

- square  $\|r\|$ , get expansion, set gradient wrt  $x$  equal to zero
- $x_{ls} = (A^T A)^{-1} A^T y = B_{ls} y$  (linear operation)
- $A^T A$  should be invertible, square, full rank
- $(A^T A)^{-1} A^T$  is a generalized inverse (is only inverse for square matrices, though)
  - Also known as the  $A^\dagger$ , ‘pseudo-inverse’
  - Which is a left inverse of  $A$

## 4.3 Projection on $\mathcal{R}(A)$

$Ax_{ls}$  is the point closest to  $y$  (i.e., projection of  $y$  onto  $\mathcal{R}(A)$ )

- $Ax_{ls} = \mathbf{proj}_{\mathcal{R}(A)}(y) = (A(A^T A)^{-1} A^T) y$

## 4.4 Orthogonality principle

The optimal residual is orthogonal to  $C(A)$

- $r = Ax_{ls} - y = (A(A^T A)^{-1} A^T - I)y$  orthogonal to  $C(A)$
- $\langle r, Az \rangle = y^T (A(A^T A)^{-1} A^T - I)^T Az = 0$  for all  $z \in \mathbb{R}^n$

## 4.5 Least-squares via $QR$ factorization

$A$  is still skinny, full rank

- Factor as  $A = QR$ ;  $Q^T Q = I_n$ ,  $R \in \mathbb{R}^{n \times n}$  upper triangular, invertible
- pseudo-inverse:  $(A^T A)^{-1} A^T = R^{-1} Q^T \Rightarrow R^{-1} Q^T y = x_{ls}$
- Pretty straight-forward
- Matlab for least squares approximation

```
x1 = inv(A' * A)*A'y; # So common that has shorthand in MATLAB
x1 = A\y;             # Works for non-skinny matrices, may do unexpected things
```

## 4.6 Full $QR$ factorization

- $A = [Q_1 \ Q_2] \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$ 
  - New  $Q$  is square, orthogonal matrix;  $R_1$  is square, upper triangular, invertible
- Remember, multiplying by orthogonal matrix doesn’t change the norm:
  - $\|Ax - y\|^2 = \|R_1 x - Q_1^T y\|^2 + \|Q_2^T y\|^2$
  - Find least squares approximation with  $x_{ls} = R_1^{-1} Q_1^T y$  (zeroes first term)

## 4.7 Applications for least squares approximations

- if there is some noise  $v$  in  $y = Ax + v$ 
  - you can't reconstruct  $x$ , but you can get close with the approximation
- Estimation: choose some  $\hat{x}$  that minimizes  $\|A\hat{x} - y\|$ , which is the deviation between the think we observed, and what we would have observed in the absence of noise

## 4.8 BLUE: Best linear unbiased estimator

- $A$  still full rank and skinny; have a 'linear estimator'  $\hat{x} = By$  ( $B$  is fat)
  - $\hat{x} = B(Ax + v)$
- Called unbiased if there is no estimation error when there's no noise; the estimator works perfectly in the absence of noise
  - if  $v = 0$  and  $BA = I$ ;  $B$  is left inverse/perfect reconstructor
- Estimation error of unbiased linear estimator is  $x - \hat{x} = sBv$ , so we want  $B$  to be small and  $BA = I$ ; small means error isn't sensitive to the noise
- The pseudo-inverse is the smallest left inverse of  $A$ :

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

$$- \sum_{i,j} B_{ij}^2 \geq \sum_{i,j} A_{ij}^{\dagger 2}$$

## 4.9 Range-finding example

- Find ranges to 4 beacons from an unknown position  $x$
- $y = - \begin{bmatrix} k_1^T \\ k_2^T \\ k_3^T \\ k_4^T \end{bmatrix} x + v$
- actual position  $x = (5.59, 10.58)$ ; measurement  $y = (-11.95, -2.84, -9.81, 2.81)$ 
  - these numbers aren't consistent in  $Ax = y$ , since there's also the error; there is no such  $x$  value that can give this  $y$  value
- There are 2 redundant sensors (2 more  $y$  values than  $x$  values); one method for estimating  $\hat{x}$  is 'just enough' method: you only need 2  $y$  values; take inverse of top half of  $A$  and pad the rest of the matrix with 0's
- use  $\hat{x} = B_{justenough} y = \begin{bmatrix} [k_1^T]^{-1} & 0 & 0 \\ [k_2^T] & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1.0 & 0 & 0 \\ -1.12 & .5 & 0 & 0 \end{bmatrix} y = \begin{bmatrix} 2.84 \\ 11.9 \end{bmatrix}$
- Least Squares method:  $\hat{x} A^\dagger y =$  this has a much smaller norm of error
- Just enough estimator doesn't seem to have good performance... unless last two measurements were really off, since JEM only takes 2 measurements into account

## 4.10 Quantizer example

Super-impressive least squares estimate; more precise than A-D converter

### 4.11 Least Squares data-fitting

- use functions  $f_1, f_2, \dots, f_n : S \rightarrow \mathbb{R}$  are called regressors or basis functions
- applications
  - interpolation-, extrapolation, smoothing of data

#### Applications

---

interpolation	don't have sensors in specific location, but want the temperature
extrapolation	get good basis functions for better interpolation
data smoothing	de-noise measurements
simple, approximate data model	Get a million samples, use the data-fitting to get a simple approximate function

---

### 4.12 Least-squares polynomial fitting

- Vandermonde matrix?

## 5 Lecture 7

### 5.1 Least-squares polynomial fitting, cont'd

- have data samples  $(t_i, y_i), i = 1, \dots, m$
- fit coefficients  $a_i$  of polynomial  $p(t) = a_0 + a_1t + \dots + a_{n-1}t^{n-1}$  so that when evaluated at  $t_i$  it will give you the associated  $y$  value
- basis functions are  $f_j(t) = t^{j-1}, j = 1, \dots, n$
- use Vandermonde matrix  $A$  ('polynomial evaluator matrix'):

$$A = \begin{bmatrix} 1 & t_1 & t_1^2 & \dots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \dots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^{n-1} \end{bmatrix}$$

- side note: use this when you want to fit throughout an interval, use a Taylor series fit if you want it close to a point

### 5.2 Growing sets of regressors

- Given ordered set of vectors; find best fit with first vector, then best fit with first and second, then best fit with first three...
- These vectors called *regressors*, or columns
- Say you have some *master list*  $A$  with  $n$  columns, and  $A^{(p)}$  will be the matrix with the first  $p$  columns of it
  - we want to minimize different sets of  $\|A^{(p)}x - y\|$
  - i.e., project  $y$  onto a growing span  $\{a_1, a_2, \dots, a_p\}$
- Solution for each  $p \leq n$  given by  $x_{ls}^{(p)} = (A_p^T A_p)^{-1} A_p^T y = R_p^{-1} Q_p^T y$ 
  - In MATLAB, `A(:,1:p)\y`, though technically it's faster to do a sort of `for` loop
- Residual,  $\|\sum_{i=1}^p x_i a_i - y\|$  reduces as  $p$  (number of columns) increases
  - though it may be same as residual with previous value of  $p$  if the optimal  $x_1 = 0$ , when  $y \perp a_1$
  - if the residual drops 15% from that of previous value of  $p$ , you say that  $a_1$  explains 15% of  $y$

### 5.3 Least-squares system identification (important topic)

- measure input, output  $u(t), y(t)$  for  $t = 0, \dots, N$  of unknown system, and try to get a model of system
- example: moving average (MA) model with  $n$  delays (try to approximate what are the weights  $h_i$  for each delay)
  - see equation/matrix in notes, though there are different ways to write it
  - get best answer with LSA

### 5.4 Model order selection

- how large should  $n$  be?
- the larger, the smaller prediction error on *data used to form model*
- but at a certain point, predictive ability of model on other I/O data from same system worsens
- probably best to choose the ‘knee’ on the graph on notes slide for prediction of new data

#### 5.4.1 Cross-validation

- check with new data, only if you’re getting small residuals on data you’ve already seen
- when  $n$  gets too large (greater than  $n = 10$  on graph), the error with ‘validation data’ actually gets larger
- this example is ideal, since  $n = 10$  is the obvious order for the model
- **Application note:** in medical, many industries, there’s a firm wall between validation data and model-developing data, so someone *else* tests your model
- in this example, it is known as *overfit* when the validation data error gets larger for  $n$  too large

### 5.5 Growing sets of measurements

- similar to GSo Regressors, except you add new rows, not columns
- this would happen if we’re estimating a parameter  $x$  (which is constant)
- Solution:  $x_{ls} = \left( \sum_{i=1}^m a_i a_i^T \right)^{-1} \sum_{i=1}^m y_i a_i$
- new way to think of least squares equation

### 5.6 Recursive ways to do least squares

- don’t have to re-add for each new measurement
  - i.e., memory is bounded
  - use equation from notes; solution is  $x_{ls}(m) = P(m)^{-1}q(m)$

### 5.7 Fast update algorithm for recursive LS

- Was a big deal back in the day; somewhat still

## 5.8 Multi-objective least squares

- Sometimes you have 2+ objectives to minimize
  - say  $J_1 = \|Ax - y\|^2$  (what we've done so far)
  - and  $J_2 = \|Fx - g\|^2$
  - these are usually competing (minimize one at cost of other)
- Variable in question is  $x \in \mathbb{R}^n$
- Plot in notes shows plot of  $(J_1(x_i), J_2(x_i))$
- Some points are unambiguously worse than others, but there is some ambiguity when  $J_1(x_1) < J_1(x_2)$ , while  $J_2(x_1) > J_2(x_2)$
- Fix this ambiguity with 'weighted-sum objective'
- $J_1 + \mu J_2 = \|Ax - y\|^2 + \mu \|Fx - g\|^2$ 
  - Say, there's a trade-off between smoothness (no noise) and better fit;  $\mu$  can have different dimensions if  $J_2$  does
- Use slope of  $\mu$  in graph ('indifference curve', in economics) [slide 7-6]

## 6 Lecture 8

Multi-objective least-squares

### 6.1 Plot of achievable objective pairs

- if it approximates an L shape (has a 'knee'), the knee is usually the obvious optimal location, so least-squares isn't as helpful
  - optimal point isn't very sensitive to  $\mu$
- Other extreme: trade-off curve looks linear (negative slope), where it's zero-sum
  - optimal point very sensitive to  $\mu$
  - slope commonly called *exchange rate curve*
- In this class, they must be convex curves (cup up/outward)
- To find Pareto optimal points, minimize  $J_1 + \mu J_2 = \alpha$ 
  - on plot, can have level curves with slope  $\mu$
  - Find point on Pareto Optimal Curve that has slope  $\mu$

## 6.2 Minimizing weighted-sum objective

- note: norm-squared of a stacked vector is norm-square of the top+norm-square of bottom

$$\begin{aligned} J_1 + \mu J_2 &= \|Ax - y\|^2 + \mu \|Fx - g\|^2 = \left\| \begin{bmatrix} A \\ \sqrt{\mu}F \end{bmatrix} x - \begin{bmatrix} y \\ \sqrt{\mu}g \end{bmatrix} \right\|^2 \\ &= \|\tilde{A}x - \tilde{y}\| \end{aligned}$$

where

$$\tilde{A} = \begin{bmatrix} A \\ \sqrt{\mu}F \end{bmatrix}, \tilde{y} = \begin{bmatrix} y \\ \sqrt{\mu}g \end{bmatrix}$$

If  $\tilde{A}$  is full rank,

$$x = (\tilde{A}^T \tilde{A})^{-1} \tilde{A}^T \tilde{y} \quad (1)$$

$$= (A^T A + \mu F^T F)^{-1} (A^T y + \mu F^T g) \quad (2)$$

Note: to plot the tradeoff curve, calculate the minimizer  $x_\mu$ , and plot the resulting pairs  $(J_1, J_2)$  In MATLAB, `[A; sqrt(mu) * F] \ [y; sqrt(mu) * g]`

## 6.3 Example: frictionless table

- $y$  is final position at  $t = 10$ ;  $y = a^T x$ ,  $a \in \mathbb{R}^{10}$
- $J_1 = (y - 1)^2$ , (final position difference from  $y = 1$  squared)
- $J_2 = \|x\|^2$  sum of force squares
- Q: Why do we often care about sum of squares? A: **It's easy to analyze** (not necessarily because it corresponds to energy)
  - $\max |x_i|$  corresponds to maximum thrust
  - $\sum |x_i|$  corresponds to fuel use
- Optimal tradeoff curve is quadratic

## 6.4 Regularized least-squares

- famous example of multi-objective least squares
  - second  $J$  term is simply  $J_2 = \|x\|^2$ , though first is the same:  $J_1 = \|Ax - y\|^2$
- Tychonov regularization works for *any*  $A$ 
  - *regularized* least-squares solution:  $x_\mu = (A^T A + \mu I)^{-1} A^T y$ , for  $F = I, g = 0$

Show  $(A^T A + \mu I)$  is invertible, no matter what size/values of  $A$  (assuming  $\mu > 0$ ): If this is *not* invertible (singular), it means some nonzero vector  $z$  gets mapped to zero ( $z \in \mathcal{N}(A)$ )

$$(A^T A + \mu I)z = 0, z \neq 0 \quad (3)$$

$$z^T (A^T A + \mu I)z = 0 \text{ since } z^T \vec{0} = 0 \quad (4)$$

$$z^T A^T A z + \mu z^T z = 0 \quad (5)$$

$$\|Az\|^2 + \mu \|z\|^2 = 0 \quad (6)$$

$$z = \vec{0} \quad (7)$$

So,  $z$  can only be zero, meaning  $\mathcal{N}(A) = \{0\} \Rightarrow (A^T A - \mu I)$  is invertible. This is also why  $\mu$  must be positive. Or, you know it's invertible, since it is full rank (and skinny) when you stack  $\mu I$  below it (see definition of  $\tilde{A}$ ).

- Application of Regularized least-squares
  - estimation/inversion
  - $Ax - y$  is sensor residual
  - prior information that  $x$  is really small
  - or, model only accurate for small  $x$
  - Tychonov solution trades off sensor fit and size of  $x$
- Image processing example
  - Laplacian regularization
    - \* image reconstruction problem
  - $x$  is vectorized version of image
  - $\|Ax - y\|^2$  is difference from real image
  - Want new objective to minimize roughness
    - \* vector  $Dx$  (from new matrix  $D$ ) which has difference between neighboring pixels as elements
      - $D_v x$  measures vertical difference
      - $D_h x$  measures horizontal difference
      - Nullspace is vector where there is no variation between pixels
  - minimize  $\|Ax - y\|^2 + \mu\|[D_h x \ D_v x]^T\|^2$ 
    - \* if  $\mu$  is turned way up, it'll be all smoothed out
    - \* if you care about total size of image, you can add another parameter  $\lambda$ :  $\|Ax - y\|^2 + \mu\|[D_h x \ D_v x]^T\|^2 + \lambda\|x\|^2$

## 6.5 Nonlinear least squares (NLLS) problem

- find  $x \in \mathbb{R}^n$  that minimizes  $\|r(x)\|^2 = \sum_{i=1}^m r_i(x)^2$
- $r(x)$  is vector of residuals;  $r(x) = Ax - y \Rightarrow$  problem reduces to linear least squares problem
- in general, can't **really** solve a NLLS problem, but can find good heuristics to get a locally optimal solution

## 6.6 Gauss-Newton method for NLLS

- Start guess for  $x$
- Loop
  - linearize  $r$  near current guess
  - new guess is linear LS solution, using linearized  $r$
  - if convergence, stop
- Linearize?
  - Jacobian:  $(Dr)_{ij} = \partial r_i / \partial x_j$
  - Linearization:  $r(x) \approx r(x^{(k)}) + Dr(x^{(k)})(x - x^{(k)})$
  - Set this linearized approximation equal to  $r(x) \approx A^{(k)}x - b^{(k)}$ 
    - \*  $A^{(k)} = Dr(x^{(k)})$
    - \*  $b^{(k)} = Dr(x^{(k)})x^{(k)} - r(x^{(k)})$



- See rest in notes
- At  $k$  th iteration, approximate NLLS problem by linear LS problem:
  - \*  $\|r(x)\|^2 \approx \|A^{(k)}x - b^{(k)}\|^2$ 
    - if you wanna make this really cool add a  $\mu\|x - x^{(k)}\|^2$  term on RHS
    - called a ‘trust region term’;
    - first (original) part says to minimize sum of squares for *model*
    - trust region term says ‘but don’t go far from where you are now’
- Could also linearize without calculus; works really well
  - See ‘particle filter’

## 6.7 G-N example

- Nice graph and residual plot
- As practical matter, good to run simulation several times (with different initial guesses)
- ‘exhaustive simulation’

## 6.8 Underdetermined linear equations

- $A \in \mathbb{R}^{m \times n}, m < n$  ( $A$  is fat)
- more variables than equations
- $x$  is underspecified
- For this section **assume  $A$  is full rank**
- Set of all solutions has form  $\{x | Ax = y\} = \{x_p + z | z \in \mathcal{N}(A)\}$
- solution has  $\dim \mathcal{N}(A) = n - m$  ‘degrees of freedom’
  - many DOF: good for design (flexibility), bad for estimation (stuff you don’t/can’t know with available measurements)

## 6.9 Least norm solution

- $x_{ls} = A^T(AA^T)^{-1}y$ 
  - similar to our familiar skinny  $A$  version:  $x_{ls} = (A^T A)^{-1}A^T y$
  - mnemonic:  $(\cdot)^{-1}$  thing must be square
    - \* if  $A$  skinny, both  $A A^T$  and  $A^T A$  could be square (syntactically)
    - \* semantically, you need the up and down patterns that will form the smallest square, i.e., full rank matrix

## 7 Lecture 9 pt 2

Thank you fucking Suncheon.

## 7.1 General norm minimization with equality constraints

- Problem:  $\boxed{\text{minimize } \|Ax - b\| \text{ subject to } Cx = d, \text{ with variable } x}$
- Least squares/least norm are special cases
  - Least norm: set  $A = I, b = 0$ , then you just have norm of  $x$  subject to some linear equations
- Same as: minimize  $(1/2)\|Ax - b\|^2$  subject to  $Cx = d$
- Lagrangian is... long ugly thing... look at notes
  - a bit easier to look at block matrix format

$$\begin{bmatrix} A^T A & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} A^T b \\ d \end{bmatrix}$$

- recover least squares (maybe) by eliminating  $C$  from matrix (not setting to zero, but only having 1 row/column in first matrix)

## 7.2 Autonomous linear dynamical systems

“What the class is nominally about”

- In continuous time, autonomous LDS has form  $\dot{x} = Ax$
- Solution:  $x(t) = e^{tA}x(0)$
- $x(t) \in \mathbb{R}^n$  is called the state
  - $n$  is state dimension
- $A$  basically maps where you are ( $x$ ) to where you’re going ( $\dot{x}$ )
  - has units of  $s^{-1}$ , frequency
- Example illustration: vector fields

## 7.3 Block diagrams

- use integrators to express  $\dot{x} = Ax$  instead of differentiators
  - block called ‘bank of integrators’
  - historically used because of analog, mechanical computers
- notches to express  $n$  signals

## 7.4 Linear circuit example

# 8 Lecture 10

Examples of autonomous linear dynamical systems,  $\dot{x} = Ax$

## 8.1 Example: Series reaction $A \rightarrow B \rightarrow C$

$$\dot{x} = \begin{bmatrix} -k_1 & 0 & 0 \\ k_1 & -k_2 & 0 \\ 0 & k_2 & 0 \end{bmatrix} x$$

- For second row, first term on rhs of  $\dot{x}_2 = k_1x_1 - k_2x_1$  is *buildup*
- Note: Column sums are 0 implies conservation of mass/materials;

## 8.2 Discrete time Markov chain

- $x(t+1) = Ax(t)$
- $x(t) = A^t x(0)$
- Given current state, the matrix of *transition probabilities*  $P$  will tell you probabilities of the next state, given the current state

## 8.3 Numerical integration of continuous system

- for a small time step  $h$ , find about where you'll be in  $h$  seconds
- $x(t+h) \approx x(t) + h\dot{x}(t) = (I + hA)x(t)$
- problem: when you do it for a long time, error can build up pretty high

## 8.4 Higher order linear dynamical systems ( $\dot{x} = Ax$ )

$$x^{(k)} = A_{k-1}x^{(k-1)} + \dots + A_1x^{(1)} + A_0x, x(t) \in \mathbb{R}^n$$

- define new variable

$$z = \begin{bmatrix} x \\ x^{(1)} \\ \vdots \\ x^{(k-1)} \end{bmatrix} \in \mathbb{R}^{nk}, \dot{z} = \begin{bmatrix} x^{(1)} \\ \vdots \\ x^{(k)} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & 0 & \cdots & I \\ A_0 & A_1 & A_2 & \cdots & A_{k-1} \end{bmatrix} z$$

- 'upshift  $x$ , and zero-pad'
- $z$  is the state, not  $x$
- in notes, black diagram with chain of integrators

## 8.5 Example: Mechanical systems

- Ex:  $K_{12}$  is 'cross-stiffness', how much stiffness you'd feel at node 1 from node 2

## 8.6 Linearization near equilibrium point

Equilibrium point corresponds to constant solution ( $f(x_e) = 0, x(t) = x_e$ )

- if you start at an equilibrium point, you'll stay there
- if you start *near* equilibrium point
  - veer off (unstable)
  - go towards equilibrium (stable)
  - something in between
- but, you never stay at an unstable equilibrium position, since equation is really  $\dot{x} = f(x) + w(t)$ , where  $w(t)$  is noise
- Near equilibrium point,  $\dot{\delta x}(t) \approx Df(x_e)\delta x(t)$ , where  $D$  is the Jacobian
  - similar to euler forward equation
- Don't fully trust approximations on approximations (but hope they work)

## 8.7 Example: pendulum linearization

- $ml^2\ddot{\theta} = -lmg \sin \theta$
- rewrite as 1st order DE with state  $x = [\theta \ \dot{\theta}]^T = [x_1 \ x_2]^T$ :

$$\dot{x} = \begin{bmatrix} x_2 \\ -(g/l) \sin x_1 \end{bmatrix}$$

- $\exists$  equilibrium point at  $x = 0$  (and  $\pi$ ), so we linearize system near  $x_e = 0$ , using a Jacobian matrix:

$$\dot{\delta x} = \begin{bmatrix} \frac{\partial x_2}{\partial x_1} (-g/l) \sin x_1|_{x_1=0} & \frac{\partial x_2}{\partial x_2} (-g/l) \sin x_1 \end{bmatrix} \delta x = \begin{bmatrix} 0 & 1 \\ -g/l & 0 \end{bmatrix} \delta x$$

## 9 Lecture 11

Solution via Laplace transform and matrix exponential Remember, we've already overloaded  $\dot{x} = ax$ . Now, we'll overload exponentials to apply to matrices  $x(t) = e^{ta}x(0)$ .

### 9.1 Laplace transform

- $z : \mathbb{R}_+ \rightarrow \mathbb{R}^{p \times q}$  (function that maps non-negative real scalars to matrices)
- Laplace transform:  $Z = \mathcal{L}(z)$ , defined by  $Z(s) = \int_0^\infty e^{-st} z(t) dt$
- Region of convergence of  $Z$  is mostly for confusing students
- Derivative property:  $\mathcal{L}(\dot{z}) = sZ(s) - z(0)$

So, we can use the Laplace transform to solve  $\dot{x} = Ax$ . Take Laplace:  $sX(s) - x(0) = AX(s)$ , rewrite as  $(sI - A)X(s) = x(0)$ , so  $X(s) = (sI - A)^{-1}x(0)$ . Then take the inverse transform:  $x(t) = \mathcal{L}^{-1}((sI - A)^{-1})x(0)$

- takes advantage if linearity of the Laplace transform
- $(sI - A)^{-1}$  is called the *resolvent* of  $A$ 
  - but not defined for eigenvalues of  $A$ ;  $s$ , ST  $\det(sI - A) = 0$
- $\Phi = \mathcal{L}^{-1}((sI - A)^{-1})$  is called the *state-transition matrix*, which maps the initial state to state at time  $t$ :  
 $x(t) = \Phi(t)x(0)$

### 9.2 Example: Harmonic oscillator

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x$$

- To solve for  $s$ , get the resolvent, then apply the Laplacian to it *elementwise*, getting

$$x(t) = \begin{bmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{bmatrix} x(0)$$

Which is a circular rotation matrix. The solutions to  $\dot{x} = ax$  is  $x(t) = e^{ta}x(0)$

- $a$  positive: exponential growth
- $a$  negative: exponential decay
- $a = 0$ : constant

### 9.3 Example: Double Integrator

- Note, with scalars,  $x$  in  $\dot{x} = ax$  grows exponentially in time, and cannot grow linearly, as with matrices (can have a  $t$  element in matrix)
- What is first column of  $\Phi(t)$  say? It tells what the state trajectory is if the initial condition was  $e_1$  (second column tells what it is if  $x(0) = e_2$ )
- First row says the linear combination that  $x_1$  is at time  $t$  given  $x(0)$

### 9.4 Characteristic polynomial

$\mathcal{X}(s) = \det(sI - A)$ ; called a *monic* polynomial

- roots of  $\mathcal{X}$  are eigenvalues of  $A$ , and  $\mathcal{X}$  has real coefficients, so e-values are real or occur in conjugate pairs

### 9.5 Get eigenvalues of $A$ and poles of resolvent

Use Cramer's rule to get  $i, j$  entry:

$$(-1)^{i+j} \frac{\det \Delta_{ij}}{\det(sI - A)},$$

where  $\Delta_{ij}$  is  $sI - A$  with  $j$  th row and  $i$  th column deleted. Poles of entries of resolvent **must** be eigenvalues of  $A$ .

### 9.6 Matrix exponential

How to overload exponentials for matrices; start with  $(I - C)^{-1} = I + C + C^2 + \dots$ . Series converges if  $|\text{eigenvalues of } C| < 1$ . Do series expansion of resolvent, then take the Laplacian of the series, which looks like the form for the expansion of  $e^{tA}$  (though square matrices replace scalars). So we end by learning that the state transition matrix,  $\Phi(t)$  is the matrix exponential  $e^{tA}$ .

- Many scalar exponential properties don't extend to matrix exponential; with scalars, this is wrong:  $e^{A+B} = e^A e^B$  (unless  $A$  and  $B$  commute:  $AB = BA$ )
- But this is ok:  $e^{-A} = (e^A)^{-1}$
- So, how do you find the matrix exponential:

Find  $e^A$ ,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

Found  $e^{tA} = \mathcal{L}^{-1}(sI - A)^{-1}$  in earlier example, so just plug in  $t = 1$ .

- Matlab: `expm(A)`, not elementwise `exp(A)`

### 9.7 Time transfer property

Summary: for  $\dot{x} = Ax$ ,  $x(t) = \Phi(t)x(0) = \boxed{e^{tA}x(0)}$ . The matrix  $e^{tA}$  propagates initial condition into state at time  $t$ . Also propagates backward in time if  $t < 0$ .

If given  $x(12)$ , find  $x(0)$  via  $e^{-12A}x(12)$ .

- Can use first order forward Euler approximate state update for small  $t$
- Discretized autonomous LDS:  $z(k+1) = e^{hA}z(k)$  (not an approximation for these equations)

## 9.8 Application: sampling a continuous time system

## 10 Lecture 12

Piecewise constant system:  $A$  is constant for certain intervals of time.

- Qualitative behavior of  $x(t)$ 
  - Eigenvalues determine (possible) behavior of  $x$
  - Can plot eigenvalues on complex axes; like pole plot
  - Can put  $x$  in summation form with polynomial coefficient and exponential terms

### 10.1 Stability

- $\dot{x} = Ax$  is stable if  $e^{tA} \rightarrow 0$  as  $t \rightarrow \infty$ 
  - means that state  $x(t)$  converges to 0 as  $t \rightarrow \infty$ , no matter  $x(0)$
  - all trajectories of  $\dot{x} = Ax$  converge to 0 as  $t \rightarrow \infty$
  - $\dot{x} = Ax$  is stable iff all eigenvalues of  $A$  have negative real part

### 10.2 Eigenvectors and diagonalization

- $\lambda \in \mathbb{C}$  is an eigenvalue of  $A \in \mathbb{C}^{n \times n}$  if (characteristic polynomial)
 
$$\mathcal{X}(\lambda) = \det(\lambda I - A) = 0$$
- i.e.,  $(\lambda I - A)$  is singular, not invertible,  $\mathcal{N}$  not equal to the 0 set

Equivalent to:

- $\exists$  nonzero  $v \in \mathbb{C}^n$  s.t.  $(\lambda I - A)v = 0$ :  $\boxed{Av = \lambda v}$  ( $v$  is the eigenvector)
  - columns are dependent
- $\exists$  nonzero  $w \in \mathbb{C}^n$  s.t.  $w^T(\lambda I - A) = 0$ :  $\boxed{w^T A = \lambda w^T}$  ( $w$  is the left eigenvector)
  - rows are dependent
- real  $A$  can still have complex e-pairs
- $A, \lambda$  real  $\Rightarrow \lambda$  is associated with a real  $v$
- conjugate (negate imaginary term of complex number[s])
- hermitian conjugate (and transpose)

### 10.3 Scaling interpretation

$Av$  is simply scaled version of  $v$  ( $\lambda$  times); all components get magnified by the same amount

### 10.4 Dynamic interpretation

For  $Av = \lambda v$ , if  $\dot{x} = Ax, x(0) = v \Rightarrow \boxed{x(t) = e^{\lambda t} v} = e^{tA} v$ .

- $A^2 v = \lambda^2 v$
- So you just need a scalar in front of the  $v$  to calculate  $x(t)$ !
- $\boxed{\text{An eigenvector is an initial condition } x(0) \text{ for which the entire trajectory is really simple.}}$
- solution  $x(t) = e^{\lambda t} v$  is a mode of  $\dot{x} = Ax$  (associated with eigenvalue  $\lambda$ )

## 10.5 Invariant set

a set  $S \subseteq \mathbb{R}^n$  is *invariant* under  $\dot{x} = Ax$  if whenever  $x(t) \in S$ , then  $x(\tau) \in S$  for all  $\tau \geq t$  (you stay stuck within the set)

- vector field interpretation: trajectories only cut *into*  $S$

If a single point is an invariant set, it must be in the nullspace;  $S = \{x_0\} \Leftrightarrow x_0 \in \mathcal{N}(A)$ , so  $Ax_0 = 0 = \dot{x}$ .

- line  $\{tv | t \in \mathbb{R}\}$  is invariant for eigenvector  $v$

## 10.6 Complex eigenvectors

- for  $a \in \mathbb{C}$ , complex trajectory  $ae^{\lambda t}v$  satisfies  $\dot{x} = Ax$ , as well as *real* part

$$\begin{aligned} x(t) &= \text{Re}(ae^{\lambda t}v) \\ &= e^{\sigma t} \begin{bmatrix} v_{re} & v_{im} \end{bmatrix} \begin{bmatrix} \cos \omega t & \sin \omega t \\ -\sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} \alpha \\ -\beta \end{bmatrix} \end{aligned}$$

where

$$v = v_{re} + jv_{im}, \lambda = \sigma + j\omega, a = \alpha + j\beta$$

- $\sigma$  gives logarithmic growth/decay factor
- $\omega$  gives angular velocity of rotation in plane
- trajectory stays in *invariant plane*  $\text{span} \{v_{re}, v_{im}\}$

## 10.7 Dynamic interpretation: left eigenvectors

### 10.8 Summary:

- *right eigenvectors* are initial conditions from which resulting motion is simple (i.e., remains on line or in plane)
- *left eigenvectors* give linear functions of state that are simple, for any initial condition

## 10.9 Example- companion matrix

- Easy to get the characteristic polynomial
- General truth: with these matrices you can't generally tell the system behavior by just looking at it
- If you push a signal through an integrator, it gets less wiggly
- By multiplying by the left eigenvector, you've filtered out the sinusoid?

# 11 Lecture 13

## 11.1 Example: Markov chain

Probability vector  $p \in \mathbb{R}^n$  that you're in each of  $n$  states:  $p(t+1) = Pp(t)$ . This probability evolves in time by being multiplied by state transition matrix  $P$ .

- $p_i(t) = \text{Prob}(z(t) = i) \Rightarrow \sum_{i=1}^n p_i(t) = 1$
- sum of each column is 1
  - called stochastic
- i.e.,  $[1 \ 1 \ \cdots \ 1]$  is a left eigenvector of  $P$  with  $\lambda = 1$

- so  $\det(I - P) = 0$ , so there's also a nonzero right eigenvector s.t.  $Pv = v$ 
  - $v$  can always be chosen to have non-negative elements, and can be normalized
- **Interpretation:**  $v$  is an equilibrium distribution; you don't change your *probability* distribution in time; always in  $v$ 
  - if  $v$  unique, it's called the steady-state distribution of the Markov chain

## 11.2 Diagonalization

- $v_1, \dots, v_n$  is LI set of eigenvectors of  $A \in \mathbb{R}^{n \times n}$ :  $Av_i = \lambda_i v_i$
- Concatenate in matrix language:

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

or,  $AT = T\Lambda$ , or  $T^{-1}AT = \Lambda$

- note,  $T$  is invertible, since its columns are linearly independent
- This is why, while  $Av = \lambda v$  is more commonly used for a scalar eigenvalue,  $\boxed{Av = v\lambda}$  is more general, as it can represent a vector of eigenvalues  $\lambda$ .
- so,  $A$  is diagonalizable if
  - $\exists T$  s.t.  $T^{-1}AT = \Lambda$  is diagonal
  - $A$  has a set of linearly independent eigenvectors
    - \* if  $A$  not diagonalizable, it is called defective

## 11.3 Not all matrices diagonalizable

i.e.,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

## 11.4 Distinct eigenvalues

**fact:** distinct eigenvalues in  $A \Rightarrow A$  diagonalizable

- converse not true, i.e.,  $I \in \mathbb{R}^{7 \times 7}$

## 11.5 Diagonalization and left eigenvectors

rewrite  $T^{-1}AT = \Lambda$  as  $T^{-1}A = \Lambda T^{-1}$ :

$$\begin{bmatrix} w_1^T \\ \vdots \\ w_n^T \end{bmatrix} A = \Lambda \begin{bmatrix} w_1^T \\ \vdots \\ w_n^T \end{bmatrix}$$

- remember that  $\Lambda$  is diagonal matrix, and multiplying by a diagonal matrix on the left is equivalent to scaling rows of the matrix
  - on the right scales the columns



Remember left/right multiplication results (whether it scales columns or rows) with  $2 \times 2$  matrix multiplication:

$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix} = \begin{bmatrix} 2x_1 & 2x_2 \\ 3y_1 & 3y_2 \end{bmatrix}$$

I.e., right multiplication of diagonal matrix scales the rows.

- Take LI set of eigenvectors as columns, invert that matrix, then the rows are **left** eigenvectors
- An eigenvector is still an eigenvector after being scaled; so any can be normalized

## 11.6 Modal form

Take a LI set of eigenvectors from  $A$ , shove them together as columns of new matrix  $T$  = “ $A$  is diagonalizable by  $T$ ”

- can define new coordinates by  $x = T\tilde{x}$ :
- $\tilde{x}$  is coordinates of  $x$  in the  $T$  expansion; modal (or eigenvector) expansion
  - $\tilde{x}$  is  $x$  in terms of the eigenvectors

$$T\dot{\tilde{x}} = AT\tilde{x} \Leftrightarrow \dot{\tilde{x}} = T^{-1}AT\tilde{x} \Leftrightarrow \dot{\tilde{x}} = \Lambda\tilde{x}$$

- in new coordinate system, system is diagonal (decoupled)
- normally, with  $\dot{x} = Ax$ , there's a ton of cross-gains from input  $x_i$  to output  $y_j$ , where all the outputs depend on all the inputs (assuming  $A$  has only non-zero entries)
  - diagonalized system decouples it; trajectory consists of  $n$  independent modes:

$$\tilde{x}_i(t) = e^{\lambda_i t} \tilde{x}_i(0)$$

## 11.7 Real modal form

when eigenvalues ( $\Rightarrow T$ ) are complex

- notes show block diagram of complex mode (note if real parts  $\sigma$  are removed, you get harmonic oscillator)

## 11.8 Diagonalization simplification

Simplifies calculation of:

- resolvent
- powers ( $A^k$ )
- exponential ( $e^A = T \mathbf{diag}(e^{\lambda_1}, \dots, e^{\lambda_n}) T^{-1}$ )
- So, diagonalization is largely a conceptual tool, and sometimes gives great computational advantage

## 11.9 Simplify for analytical functions of a matrix

### 11.10 Solution via diagonalization

$\dot{x} = Ax$  solution is  $x(t) = e^{tA}x(0)$

- with diagonalization, solution given as

$$x(t) = \sum_{i=1}^n e^{\lambda_i t} (w_i^T x(0)) v_i$$

### 11.11 Interpretation

- (left eigenvectors) decompose initial state  $x(0)$  into modal components  $w_i^T x(0)$
- $e^{\lambda_i t}$  term propagates  $i$  th mode forward  $t$  seconds
- reconstruct state as linear combination of (right eigenvectors)

### 11.12 Application

Finding  $x(0)$  that gives stable solution.

### 11.13 Stability of discrete-time systems

- powers of complex numbers  $s^k$  go to zero if  $|s| < 1$ 
  - imaginary part tells how much of a rotation at each step you get
- $x(t+1) = Ax(t)$  is stable iff all eigenvalues of  $A$  have magnitude less than one
- spectral radius of  $A$  :  $\rho(A) = \max |\lambda_i|$ 
  - so it is a stable system iff  $\rho(A) < 1$
  - $\rho$  gives rough growth or decay

### 11.14 Jordan Canonical form

- Any matrix  $A \in \mathbb{R}^{n \times n}$  can be expressed in Jordan-canonical form (via ‘similarity transformation,’ for some invertible matrix  $T^{-1}$ )

$$T^{-1}AT = J = \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_q \end{bmatrix}$$

where

$$J_i = \begin{bmatrix} \lambda_i & 1 & & \\ & \lambda_i & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{bmatrix} \in \mathbb{C}^{n_i \times n_i}$$

- $J$  is ‘upper bidiagonal’
- Jordan form is unique (up to permutations of blocks- blocks might be in different places in the diagonal)
- *Almost* strictly a conceptual tool; almost never used for numerical computations
- Jordan forms are inutil if the matrix is already diagonalizable
- When you get into Jordan form, you can use a chain of integrators to represent it in block diagram form
- Jordan blocks refer to dynamics blocks that cannot be decoupled
- Jordan blocks yield:
  - repeated poles in resolvent
  - terms of form  $t^p e^{t\lambda}$  in  $e^{tA}$ : solution to  $\dot{x} = Ax$  is in form of polynomials, with  $t^n e^{\lambda t}$  terms

## 11.15 Jordan block example

$$\dot{x} = \begin{bmatrix} 0 & 1 & & \\ & 0 & 1 & \\ & & 0 & 1 \\ & & & 0 \end{bmatrix} x$$

- Eigenvalues are all zero
- in the solutions you should expect  $t^p e^0$  ('constant') terms: solution of form  $t + t^2 + t^3$

## 12 Lecture 14

- Jordan canonical form: 'closest you can get to diagonalizable form, if matrix isn't diagonalizable'
- via inverse Laplace transform:

$$e^{tJ_\lambda} = e^{t\lambda}(I + tF_1 + \dots + (t^{k-1}/(k-1)!)F_{k-1})$$

$$= e^{t\lambda} \begin{bmatrix} 1 & t & \dots & t^{k-1}/(k-1)! \\ & 1 & \dots & t^{k-2}/(k-2)! \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}$$

## 13 Appendix

Some special things to remember.

### 13.1 Inverse, transpose properties

- $(AB)^{-1} = B^{-1}A^{-1}$
- $(A^{-1})^T = (A^T)^{-1} = A^{-T}$

### 13.2 Invertibility implications

For an  $n$ -by- $n$  matrix  $A$

Invertible	mnemonic
$ A  \neq 0$	$ A  = 0 \Rightarrow$ you can't compute the inverse
non-singular	- (remember base case $2 \times 2$ matrix inverse involves $1/ A $ term)
$A$ is full rank	singular $\Rightarrow$ the matrix sends a nontrivial subspace to the singular subspace, $\{0\}$
$\mathcal{N}(A) = \{0\}$	linearly independent columns (invertibility $\Rightarrow$ 1-to-1/injective)
$\mathcal{R}(A) = \mathbb{R}^n$	linearly independent columns
$Ax = b$ has unique solution for every $b$	linearly independent columns
	- no more than one solution (can't add members of $\mathcal{N}(A)$ for multiple $b$ )
	- one solution, since $\mathcal{R}(A) = \mathbb{R}^n$ ; everything reachable/surjective
	- one solution found using the unique inverse of $A$
$\text{rref}(A) = I_n$	
$A$ is a product of elementary matrices	

### 13.3 Tips

- to sum up all the elements in matrix  $A$ , multiply by one vectors:  $\sum_{i,j} (A)_{i,j} = \mathbf{1}^T A \mathbf{1}$
- each row  $i$  in  $Z$  is a linear combination of rows  $i, \dots, n$  in  $Y : Z = UY$ , where  $U$  is upper triangular:  $U_{ij} = 0$  for  $i > j$ 
  - more in 2.24, hw2

## 14 Homework assignments

Homework 1	Lecture 4	2.1–2.4, 2.6, 2.9, 2.12, +
Homework 2	Lecture 6	3.2, 3.3, 3.10, 3.11, 3.16, 3.17, +
Homework 3	Lecture 8	2.17, 3.13, 4.1–4.3, 5.1, 6.9, +
Homework 4	Lecture 10	5.2, 6.2, 6.5, 6.12, 6.14, 6.26, 7.3, 8.2
Homework 5	Lecture 13	10.2, 10.3, 10.4, +
Homework 6	Lecture 14	9.9, 10.5, 10.6, 10.8, 10.14, 11.3, and 11.6a
Homework 7	Lecture 16	10.9, 10.11, 10.19, 11.13, 12.1, 13.1, +
Homework 8	Lecture 18	13.17, 14.2, 14.3, 14.4, 14.6, 14.8, 14.9, 14.11, 14.13, 14.21, 14.33, +
Homework 9	Lecture 20	14.16, 14.26, 15.2, 15.3, 15.6, 15.8, 15.10, and 15.11