Announcement

- HW1 assigned, due Fri/Sun (4/8, 4/10)
- Maryam's OH: Wed 2-3pm, virtual (Zoom link same as class)
- see website for updated course logistics (& details on Python exercises, self-grading scheme for HWs, etc.)

EE445 Mod1-Lec3: Linear Algebra III

References:

- [VMLS]: Chapter 5, 6, 7

Gram-Schmidt (orthogonalization) algorithm

- an algorithm to check if $a_1, \ldots, a_k$ are linearly independent
- we'll see later it has many other uses
- useful properties:
  - suppose you've orthogonalized vectors $a_1, \ldots, a_k$, and a new vector a_{k+1} is then added to the list. G-S lets you update the previous solution easily (and efficiently).
  - an "incremental" algorithm that handles new data arriving—related to "online" learning

e.g., online representation learning
streaming input data, ...

Gram-Schmidt algorithm

given n-vectors $a_1, \dots, a_k$
for $i = 1, \dots, k$,

1. orthogonalization: $\tilde{q}_i = a_i - (q_1^T a_i) q_1 - \dots - (q_{i-1}^T a_i) q_{i-1}$
2. test for lin. independence: if $\tilde{q}_i = 0$, quit
3. normalization: $q_i = \tilde{q}_i / \|\tilde{q}_i\|$

$\underline{i=1:}$
$\tilde{q}_1 = a_1$ (no other terms, $i-1=0$ so no q's exist yet)
$q_1 = \tilde{q}_1 / \|\tilde{q}_1\|$

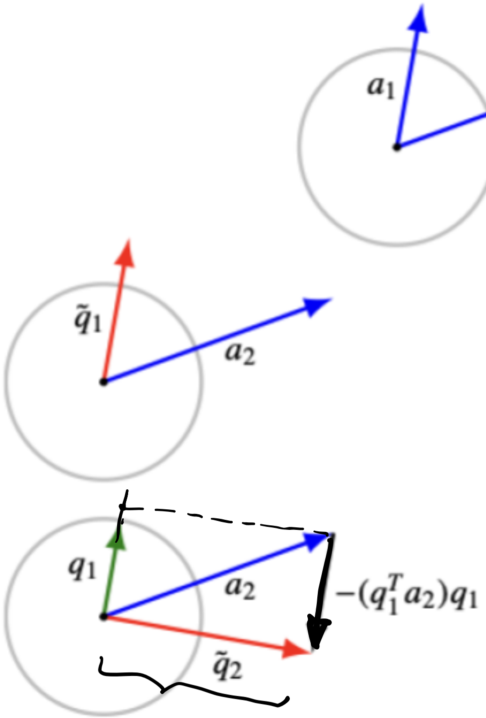
$\underline{i=2:}$
$\tilde{q}_2 = a_2 - (q_1^T a_2) q_1$
$q_2 = \tilde{q}_2 / \|\tilde{q}_2\|$

- if G-S stops early in iteration $i = j$, then a_j is a linear combination of $a_1, \dots, a_{j-1}$ (so set of vectors is linearly dependent)
- if G-S doesn't stop early, then linearly independent

Example

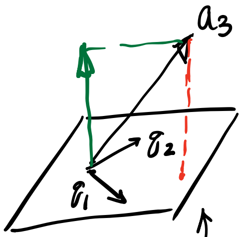
$\underline{i=1}$

$\tilde{q}_1 = a_1$



$\underline{i=2}$:

$\tilde{q}_2 = a_2 - (q_1^T a_2)\, q_1$



3D version

Analysis of G-S algorithm

we show $q_1, \dots, q_i$ are orthonormal, by induction

- assume it's true for $i-1$. orthogonalization step ensures

$$\tilde{q}_i \perp q_1, \quad \dots \quad , \tilde{q}_i \perp q_{i-1}$$

- to see this, take inner product of both sides with q_j, $j < i$: $j=1,\dots,i-1$

$$q_j^T \tilde{q}_i = q_j^T a_i - (q_1^T a_i)\underbrace{(q_j^T q_1)}_{=0} - \cdots - (q_j^T a_i)\underbrace{(q_j^T q_j)}_{=1} - \cdots - (q_{i-1}^T a_i)\underbrace{(q_j^T q_{i-1})}_{=0}$$

$$= q_j^T a_i - q_j^T a_i = 0$$

so: $q_i \perp q_1, \dots, q_i \perp q_{i-1}$

- normalization step ensures $\|q_i\| = 1$

Analysis of G-S algorithm

assume G–S has not terminated before step i: then

- a_i is a lin. comb. of $q_1, \dots, q_i$:

$$a_i = \|\tilde{q}_i\| q_i + (q_1^T a_i) q_1 + \dots + (q_{i-1}^T a_i) q_{i-1}$$

- q_i is also a lin. comb. of $a_1, \dots, a_i$:
  if (by induction assumption) each $q_1, \dots, q_{i-1}$ is a lin. comb. of $a_i, \dots, a_{i-1}$, then for

$$q_i = \frac{1}{\|\tilde{q}_i\|} \left(a_i - (q_1^T a_i) q_1 - \dots - (q_{i-1}^T a_i) q_{i-1} \right)$$

assume G-S terminates at step j: then a_j is a linear combination of $a_1, \dots, a_{j-1}$

Analysis of G-S algorithm

- assume G–S has not terminated before step i: then
  - a_i is a lin. comb. of $q_1, \ldots, q_i$:

$$a_i = \|\tilde{q}_i\| q_i + (q_1^T a_i) q_1 + \ldots + (q_{i-1}^T a_i) q_{i-1}$$

  - q_i is also a lin. comb. of $a_1, \ldots, a_i$:
    if (by induction assumption) each $q_1, \ldots, q_{i-1}$ is a lin. comb. of $a_i, \ldots, a_{i-1}$, then for

$$q_i = \frac{1}{\|\tilde{q}_i\|} \left(a_i - (q_1^T a_i) q_1 - \ldots - (q_{i-1}^T a_i) q_{i-1} \right)$$

so q_i is a lin. comb. of $a_1, \cdots, a_{i-1}, a_i$

- assume G-S terminates at step j: then a_j is a linear combination of $a_1, \ldots, a_{j-1}$

when $\tilde{q}_j = 0$

Review of Matrices

- a $m \times n$ matrix is a rectangular array of numbers, denoted as $\underline{A \in \mathbf{R}^{m\times n}}$, e.g.,

$$\begin{bmatrix} 0 & 1 & -2.3 & 0.1 \\ 1.3 & 4 & -0.1 & 0 \\ 4.1 & -1 & 0 & 1.5 \end{bmatrix}_{3\times 4}$$

$A_{31} = 4.1$

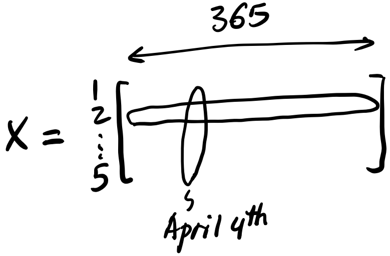
- $\underline{A_{ij}}$ is the i, jth element (entry); transpose: $(A^T)_{ij} = A_{ji}$ $\begin{bmatrix} \times & 0 \\ 0 & \times \end{bmatrix}$ $\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ 0 & & & \times \end{bmatrix}$
- shapes: tall ($m > n$), wide ($m < n$), square ($m = n$), diagonal, upper triangular,. . .
- column & row representation of matrix (a_i are column m-vectors, b_i are row n-vectors):

$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \dots & a_n \\ | & | & & | \end{bmatrix}, \qquad A = \begin{bmatrix} - b_1 - \\ - b_2 - \\ \vdots \\ - b_m - \end{bmatrix}$$

$a_i \in \mathbb{R}^m$ $\qquad$ $b_i \in \mathbb{R}^n$

Examples

- *image:* X_{ij} is the pixel value in a gray-scale image
- *rainfall data:* X_{ij} is rainfall at location i on day j
- *feature matrix:* X_{ij} is value of feature i for entity j



e.g. customer, patient, ...

a block matrix: $\quad A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}$

ex: $B = \begin{bmatrix} 0 & 2 & 3 \end{bmatrix}$, $C = [-1]$, $D = \begin{bmatrix} 2 & 2 & 1 \\ 1 & 3 & 5 \end{bmatrix}$, $E = \begin{bmatrix} 4 \\ 4 \end{bmatrix}$

$$\left[\begin{array}{ccc|c} 0 & 2 & 3 & -1 \\ \hline 2 & 2 & 1 & 4 \\ 1 & 3 & 5 & 4 \end{array}\right]$$

Matrix Frobenius norm

- for $m \times n$ matrix A,

$$\|A\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}^2 \right)^{1/2}$$

  (in the book, F subscript is often dropped). agrees with vector norm if $n = 1$.
- satisfies norm properties:
  - $\|\alpha A\|_F = |\alpha| \|A\|_F$
  - $\|A + B\|_F \leq \|A\|_F + \|B\|_F$
  - $\|A\|_F \geq 0$; and $\|A\|_F = 0$ only if $A = 0$
- distance between two matrices: $\|A - B\|_F$
- (there are many other matrix norms, will see some later)

Examples

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_3 \\ x_2 \\ x_1 \end{bmatrix}$$

- **reversal matrix:** $f(x) = Ax = (x_n, \ldots, x_1)$
  Saw in HW0, P6.

- **running sum:** $f(x) = Ax = (x_1, x_1 + x_2, x_1 + x_2 + x_3, \ldots, \sum_{i=1}^{n} x_i)$ with

$$\underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ 1 & 1 & 1 \cdots & 0 \\ \vdots & & & \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_1 + x_2 \\ \vdots \\ \vdots \end{bmatrix}$$

Examples

$\tilde{x} = x - \text{avg}(x)\mathbf{1} = x - \frac{1}{n}(\mathbf{1}^T x)\mathbf{1}$

$$\begin{aligned}\tilde{x} &= x - \frac{1}{n}\underbrace{(\mathbf{1}^T x)}_{\text{scalar}}\mathbf{1} \\ &= x - \frac{1}{n}\mathbf{1}(\mathbf{1}^T x) \\ &= x - \frac{1}{n}(\mathbf{1}\mathbf{1}^T)x = \underbrace{\left(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)}_{A}x\end{aligned}$$

$\begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix}[1 \cdots 1]$

- **centering matrix:** $\tilde{x} = Ax$ is *centered* (de-meaned) version of x with

$$A = \begin{bmatrix} 1-1/n & -1/n & \dots & -1/n \\ -1/n & 1-1/n & \dots & -1/n \\ \vdots & & \ddots & \vdots \\ -1/n & -1/n & \dots & 1-1/n \end{bmatrix}$$

$(Ax)_i = x_i \underbrace{- \frac{1}{n}x_1 - \cdots - \frac{1}{n}x_n}$

- **difference matrix** D and $y = Dx$ (vector of differences of consecutive entries of x):

$$D = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad Dx = \begin{bmatrix} x_2 - x_1 \\ x_3 - x_2 \\ \vdots \\ x_n - x_{n-1} \end{bmatrix} \qquad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$\|Dx\|^2 = (x_2 - x_1)^2 + \cdots + (x_n - x_{n-1})^2 \rightsquigarrow$ "wiggliness"

Matrix-vector product

- Define $y = Ax$, for $m \times n$ matrix A and n-vector x, as

$$y_i = A_{i1}x_1 + \ldots + A_{in}x_n, \quad i = 1, \ldots, m$$

- *row interpretation:*
  $y_i = b_i^T x$, $i = 1, \ldots, m$, where $b_1^T, \ldots, b_m^T$ are rows of A (so $y = Ax$ gives inner product of all rows of A with x)
- example: $(A\mathbf{1})_i =$ sum across row i
- *column interpretation:*
  $y = x_1 a_1 + x_2 a_2 + \ldots + x_n a_n$, where $a_1, \ldots, a_n$ are columns of A
- example: $Ae_j =$

Matrix-vector product

- Define $y = Ax$, for $m \times n$ matrix A and n-vector x, as

$$y_i = A_{i1}x_1 + \ldots + A_{in}x_n, \quad i = 1, \ldots, m$$

- *row interpretation:*
  $y_i = b_i^T x$, $i = 1, \ldots, m$, where $b_1^T, \ldots, b_m^T$ are rows of A (so $y = Ax$ gives inner product of all rows of A with x)
- example: $A\mathbf{1} =$
- *column interpretation:*
  $y = x_1 a_1 + x_2 a_2 + \ldots + x_n a_n$, where $a_1, \ldots, a_n$ are columns of A
- example: $Ae_j = a_j$

Ex: Feature matrix-weight vector

- $X = \left[\begin{array}{ccc} x_1 & \dots & x_N \end{array} \right]$ is an $n \times N$ *feature matrix*
- column x_j is feature n-vector for object/example j
- X_{ij} is value of feature i for example j
- n-vector $\underline{w}$ is *weight* vector
- $s = \underset{N\times n}{X^T} \underset{n\times 1}{w}$ is vector of *scores*, for each example: $s_j = x_j^T w$

score for example j is a weighted sum of its features

e.g. credit score (for bank loans)

Ex: Input-output matrix

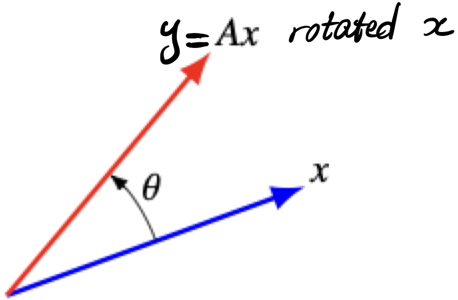
$A \in \mathbb{R}^{m \times n}$

- consider $y = Ax$:
- n-vector x is *input* or action
- m-vector y is *output* or result
- A_{ij} is the *gain* from input j to output i
- e.g., if A is lower triangular, then y_i depends only on $x_1, \ldots, x_i$ (linear system is causal)

Ex: Geometric transformations

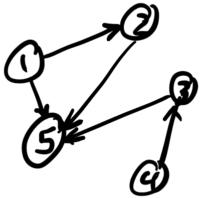
- many geometric transformations and mappings of 2D and 3D vectors can be represented by $y = Ax$
- e.g., rotation by θ:

$$y = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{A} x$$



(to get the entries, look at Ae_1, Ae_2)

Ex: Incidence matrix in a graph



- graph with n vertices or nodes, m (directed) edges or links.
- incidence matrix is $n \times m$ matrix

$$A_{ij} = \begin{cases} 1, & \text{edge } j \text{ points to node } i \\ -1, & \text{edge } j \text{ points from node } i \\ 0 & \text{otherwise} \end{cases}$$

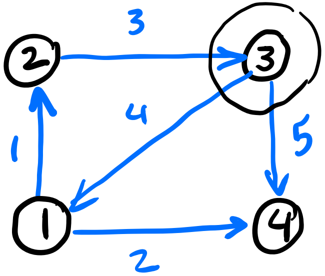
- ex with $n = 4$, $m = 5$:

nodes \ edges

$$A = \begin{bmatrix} -1 & -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_5 \end{bmatrix}$$

node 3

edge 1



Incidence matrix and flow conservation

- m-vector x gives flows (of something) along the edges
- examples: heat, money, power, mass, people,...
- $x_j > 0$ means flow follows edge direction
- Ax is n-vector that gives the total or net flows
- $(Ax)i$ is the net flow into node i
- $Ax = 0$ is *flow conservation*

in electric circuits : KCL

(KVL can be described by $A^T v = e_v$)

node voltages

voltage drops across edges

Ex: Input-output convolution

- for n-vector a, m-vector b, the (discrete-time) convolution $c = a * b$ is the $(n+m-1)$-vector

$$c_k = \sum_{i+j=k+1} a_i b_j, \quad k = 1, \ldots, n+m-1$$

$j = k+1-i$

- as seen in ee341 (and ee235)
- e.g., with $n = 4$, $m = 3$:

$$\left\{\begin{aligned} c_1 &= a_1 b_1 \\ c_2 &= a_1 b_2 + a_2 b_1 \\ c_3 &= a_1 b_3 + a_2 b_2 + a_3 b_1 \\ &\vdots \\ c_6 &= a_4 b_3 \end{aligned}\right.$$

Convolution and Toeplitz matrices

input   impulse response (of linear input/output system)

- can express $c = \underline{a * b}$ using matrices as $c = \underline{T(b)}a$, with the *Toeplitz* matrix

$$T(b) = \begin{bmatrix} b_1 & 0 & 0 & 0 \\ b_2 & b_1 & 0 & 0 \\ b_3 & b_2 & b_1 & 0 \\ 0 & b_3 & b_2 & b_1 \\ 0 & 0 & b_3 & b_2 \\ 0 & 0 & 0 & b_3 \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_4 \end{bmatrix}$$

Convolution example: moving average of time series

- n-vector x represents a time series (time steps $k=1,\cdots,n$)
- convolution $y = a * x$ with $a = (1/3, 1/3, 1/3)$ is a 3-period *moving average*:

$$y_k = \frac{1}{3}(x_k + x_{k-1} + x_{k-2}), \quad k = 1, 2, \ldots, n+2$$

with x_k taken as zero for $k < 1$ and $k > n$.

moving ave =
a low-pass filter
(smoothing)