Lecture slides for

Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares

Stephen Boyd Lieven Vandenberghe

1. Vectors

Outline

Vectors

- a *vector* is an ordered list of numbers
- written as

$$\begin{bmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{bmatrix} \quad \text{or} \quad \begin{pmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{pmatrix}$$

  or $(-1.1, 0, 3.6, -7.2)$
- numbers in the list are the *elements* (*entries*, *coefficients*, *components*)
- number of elements is the *size* (*dimension*, *length*) of the vector
- vector above has dimension 4; its third entry is 3.6
- vector of size n is called an *n-vector*
- numbers are called *scalars*

Vectors via symbols

- we'll use symbols to denote vectors, *e.g.*, a, X, p, β, E^{aut}
- other conventions: $\mathbf{g}$, $\vec{a}$
- ith element of n-vector a is denoted a_i
- if a is vector above, $a_3 = 3.6$
- in a_i, i is the *index*
- for an n-vector, indexes run from $i = 1$ to $i = n$
- *warning:* sometimes a_i refers to the ith vector in a list of vectors
- two vectors a and b *of the same size* are *equal* if $a_i = b_i$ for all i
- we overload $=$ and write this as $a = b$

Block vectors

- suppose b, c, and d are vectors with sizes m, n, p
- the *stacked vector* or *concatenation* (of b, c, and d) is

$$a = \begin{bmatrix} b \\ c \\ d \end{bmatrix}$$

- also called a *block vector*, with (block) entries b, c, d
- a has size $m + n + p$

$$a = (b_1, b_2, \ldots, b_m, c_1, c_2, \ldots, c_n, d_1, d_2, \ldots, d_p)$$

Zero, ones, and unit vectors

- n-vector with all entries 0 is denoted 0_n or just 0
- n-vector with all entries 1 is denoted $\mathbf{1}_n$ or just $\mathbf{1}$
- a *unit vector* has one entry 1 and all others 0
- denoted e_i where i is entry that is 1
- unit vectors of length 3:

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \qquad e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

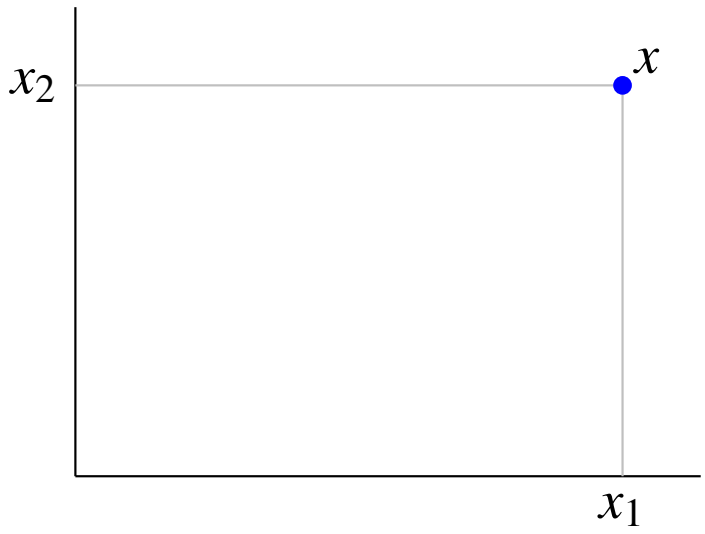
Sparsity

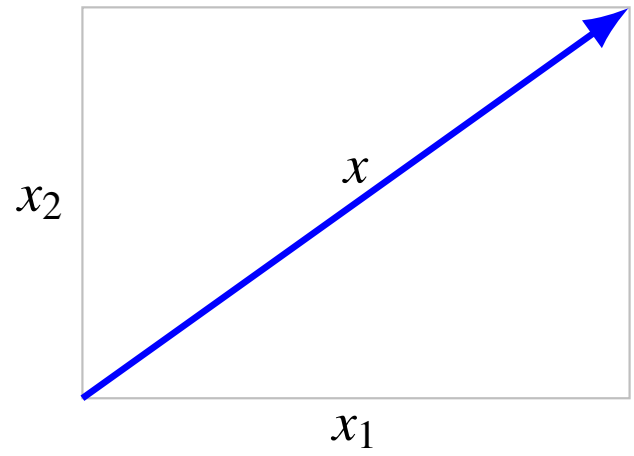
- a vector is *sparse* if many of its entries are 0
- can be stored and manipulated efficiently on a computer
- $\mathbf{nnz}(x)$ is number of entries that are nonzero
- examples: zero vectors, unit vectors

Outline

Location or displacement in 2-D or 3-D

2-vector (x_1, x_2) can represent a location or a displacement in 2-D





More examples

- color: $(\mathrm{R}, \mathrm{G}, \mathrm{B})$
- quantities of n different commodities (or resources), *e.g.*, bill of materials
- portfolio: entries give shares (or \$ value or fraction) held in each of n assets, with negative meaning short positions
- cash flow: x_i is payment in period i to us
- audio: x_i is the acoustic pressure at sample time i
  (sample times are spaced 1/44100 seconds apart)
- features: x_i is the value of ith *feature* or *attribute* of an entity
- customer purchase: x_i is the total \$ purchase of product i by a customer over some period
- word count: x_i is the number of times word i appears in a document

Word count vectors

- a short document:

  Word count vectors are used **in** computer based **document** analysis. Each entry of the **word** count vector is the **number** of times the associated dictionary **word** appears **in** the **document**.

- a small dictionary (left) and word count vector (right)

$$\begin{array}{ll} \text{word} & \begin{bmatrix} 3 \\ 2 \\ 1 \\ 0 \\ 4 \\ 2 \end{bmatrix} \\ \text{in} & \\ \text{number} & \\ \text{horse} & \\ \text{the} & \\ \text{document} & \end{array}$$

- dictionaries used in practice are much larger

Outline

Vector addition

- n-vectors a and b can be added, with sum denoted $a + b$
- to get sum, add corresponding entries:

$$\begin{bmatrix} 0 \\ 7 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 9 \\ 3 \end{bmatrix}$$

- subtraction is similar

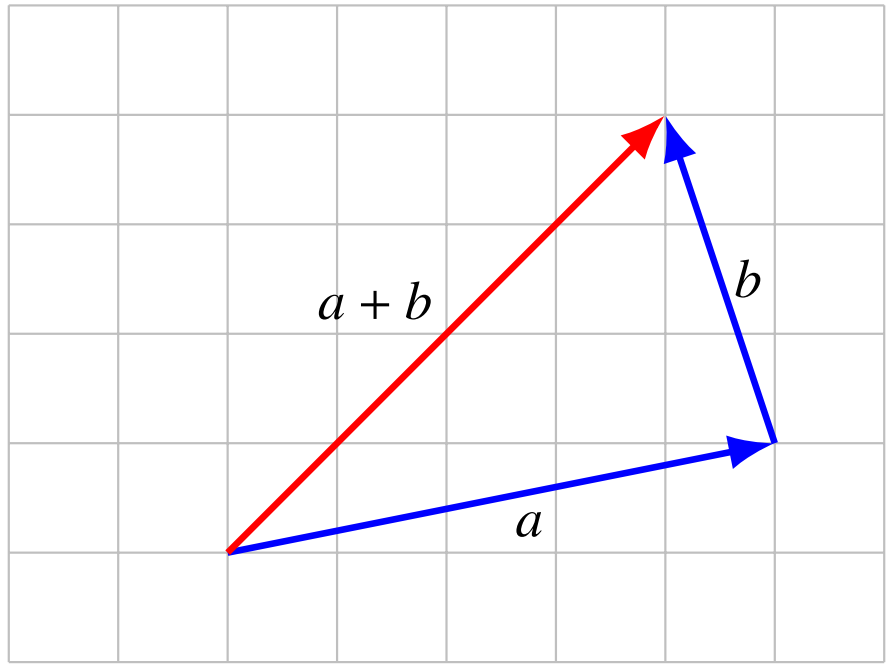
Properties of vector addition

- *commutative*: $a + b = b + a$
- *associative*: $(a + b) + c = a + (b + c)$
  (so we can write both as $a + b + c$)
- $a + 0 = 0 + a = a$
- $a - a = 0$

these are easy and boring to verify

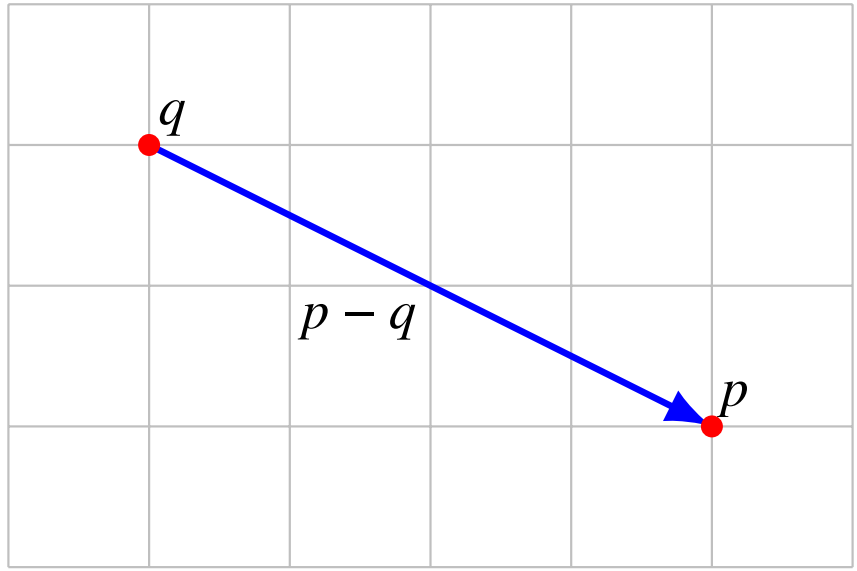
Adding displacements

if 3-vectors a and b are displacements, $a + b$ is the sum displacement



Displacement from one point to another

displacement from point q to point p is $p - q$



Scalar-vector multiplication

- scalar β and n-vector a can be multiplied

$$\beta a = (\beta a_1, \ldots, \beta a_n)$$

- also denoted $a\beta$
- example:

$$(-2)\begin{bmatrix} 1 \\ 9 \\ 6 \end{bmatrix} = \begin{bmatrix} -2 \\ -18 \\ -12 \end{bmatrix}$$

Properties of scalar-vector multiplication

- associative: $(\beta\gamma)a = \beta(\gamma a)$
- left distributive: $(\beta + \gamma)a = \beta a + \gamma a$
- right distributive: $\beta(a + b) = \beta a + \beta b$

these equations look innocent, but be sure you understand them perfectly

Linear combinations

- for vectors $a_1, \ldots, a_m$ and scalars $\beta_1, \ldots, \beta_m$,

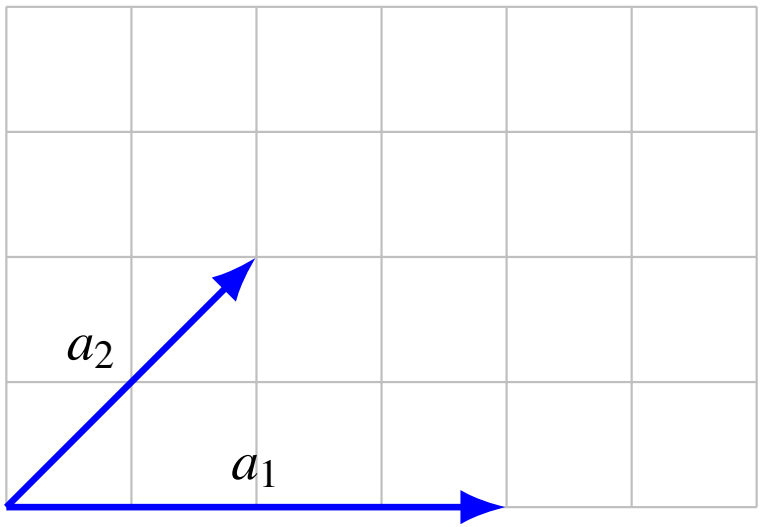
$$\beta_1 a_1 + \cdots + \beta_m a_m$$

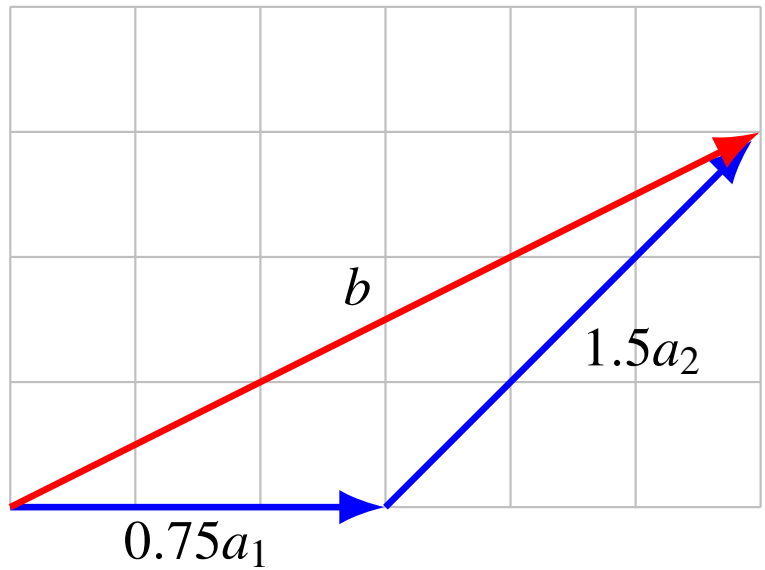
  is a *linear combination* of the vectors
- $\beta_1, \ldots, \beta_m$ are the *coefficients*
- a *very* important concept
- a simple identity: for any n-vector b,

$$b = b_1 e_1 + \cdots + b_n e_n$$

Example

two vectors a_1 and a_2, and linear combination $b = 0.75a_1 + 1.5a_2$





Replicating a cash flow

- $c_1 = (1, -1.1, 0)$ is a \$1 loan from period 1 to 2 with 10% interest
- $c_2 = (0, 1, -1.1)$ is a \$1 loan from period 2 to 3 with 10% interest
- linear combination

$$d = c_1 + 1.1c_2 = (1, 0, -1.21)$$

  is a two period loan with 10% compounded interest rate
- we have *replicated* a two period loan from two one period loans

Outline

Inner product

- *inner product* (or *dot product*) of n-vectors a and b is

$$a^T b = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$$

- other notation used: $\langle a, b \rangle$, $\langle a | b \rangle$, (a, b), $a \cdot b$

- example:

$$\begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \\ -3 \end{bmatrix} = (-1)(1) + (2)(0) + (2)(-3) = -7$$

Properties of inner product

- $a^T b = b^T a$
- $(\gamma a)^T b = \gamma(a^T b)$
- $(a + b)^T c = a^T c + b^T c$

can combine these to get, for example,

$$(a + b)^T (c + d) = a^T c + a^T d + b^T c + b^T d$$

General examples

- $e_i^T a = a_i$ (picks out ith entry)
- $\mathbf{1}^T a = a_1 + \cdots + a_n$ (sum of entries)
- $a^T a = a_1^2 + \cdots + a_n^2$ (sum of squares of entries)

Examples

- w is weight vector, f is feature vector; $w^T f$ is score
- p is vector of prices, q is vector of quantities; $p^T q$ is total cost
- c is cash flow, d is discount vector (with interest rate r):

$$d = (1, 1/(1+r), \ldots, 1/(1+r)^{n-1})$$

  $d^T c$ is net present value (NPV) of cash flow

- s gives portfolio holdings (in shares), p gives asset prices; $p^T s$ is total portfolio value

Outline

Flop counts

- computers store (real) numbers in *floating-point format*
- basic arithmetic operations (addition, multiplication, ...) are called *floating point operations* or flops
- complexity of an algorithm or operation: total number of flops needed, as function of the input dimension(s)
- this can be *very grossly approximated*
- crude approximation of time to execute: (flops needed)/(computer speed)
- current computers are around 1Gflop/sec (10^9 flops/sec)
- but this can vary by factor of 100

Complexity of vector addition, inner product

- $x + y$ needs n additions, so: n flops
- $x^T y$ needs n multiplications, $n - 1$ additions so: $2n - 1$ flops
- we simplify this to $2n$ (or even n) flops for $x^T y$
- and much less when x or y is sparse

2. Linear functions

Outline

Linear and affine functions

Taylor approximation

Regression model

Superposition and linear functions

- $f : \mathbf{R}^n \to \mathbf{R}$ means f is a function mapping n-vectors to numbers
- f satisfies the *superposition property* if

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$$

  holds for all numbers α, β, and all n-vectors x, y
- be sure to parse this very carefully!
- a function that satisfies superposition is called *linear*

The inner product function

- with a an n-vector, the function

$$f(x) = a^T x = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n$$

  is the *inner product function*

- $f(x)$ is a weighted sum of the entries of x
- the inner product function is linear:

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) \\ &= a^T(\alpha x) + a^T(\beta y) \\ &= \alpha(a^T x) + \beta(a^T y) \\ &= \alpha f(x) + \beta f(y) \end{aligned}$$

. . . and all linear functions are inner products

- suppose $f : \mathbf{R}^n \to \mathbf{R}$ is linear
- then it can be expressed as $f(x) = a^T x$ for some a
- specifically: $a_i = f(e_i)$
- follows from

$$\begin{aligned} f(x) &= f(x_1 e_1 + x_2 e_2 + \cdots + x_n e_n) \\ &= x_1 f(e_1) + x_2 f(e_2) + \cdots + x_n f(e_n) \end{aligned}$$

Affine functions

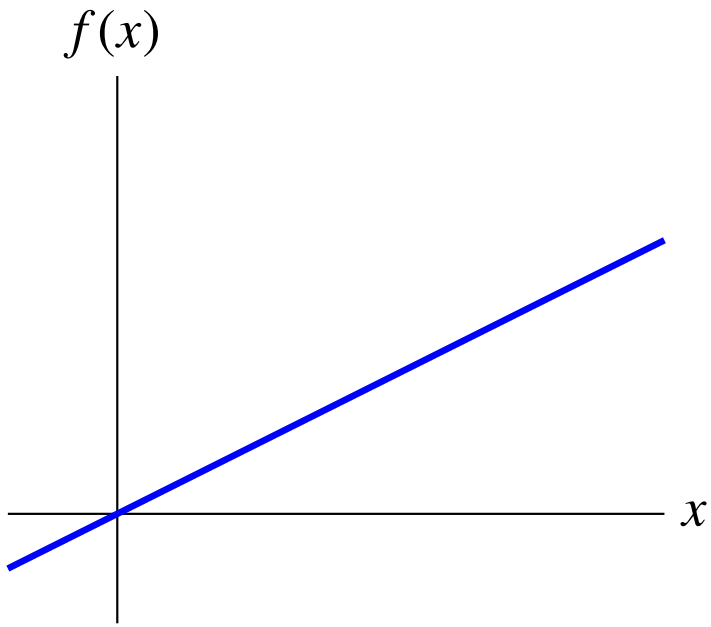
- a function that is linear plus a constant is called *affine*
- general form is $f(x) = a^T x + b$, with a an n-vector and b a scalar
- a function $f : \mathbf{R}^n \to \mathbf{R}$ is affine if and only if

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$$

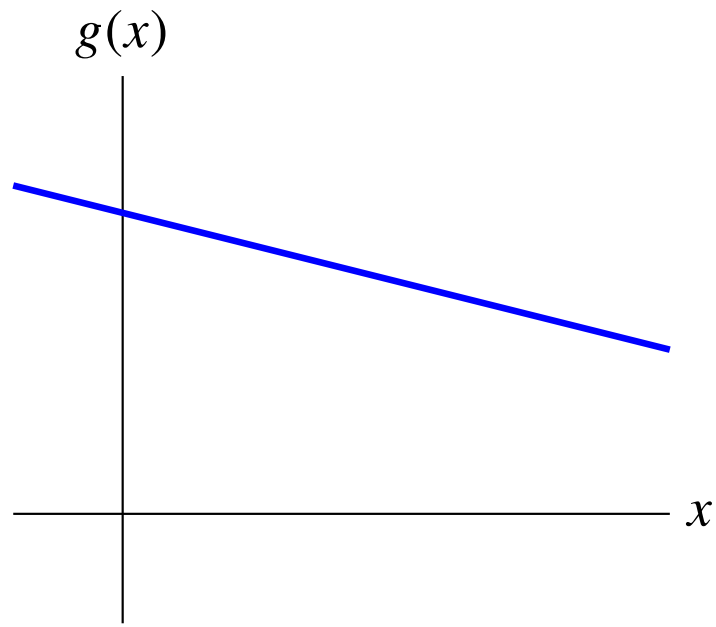
  holds for all α, β with $\alpha + \beta = 1$, and all n-vectors x, y
- sometimes (ignorant) people refer to affine functions as linear

Linear versus affine functions

f is linear



g is affine, not linear



Outline

First-order Taylor approximation

- suppose $f : \mathbf{R}^n \to \mathbf{R}$
- *first-order Taylor approximation* of f, near point z:

$$\hat{f}(x) = f(z) + \frac{\partial f}{\partial x_1}(z)(x_1 - z_1) + \cdots + \frac{\partial f}{\partial x_n}(z)(x_n - z_n)$$

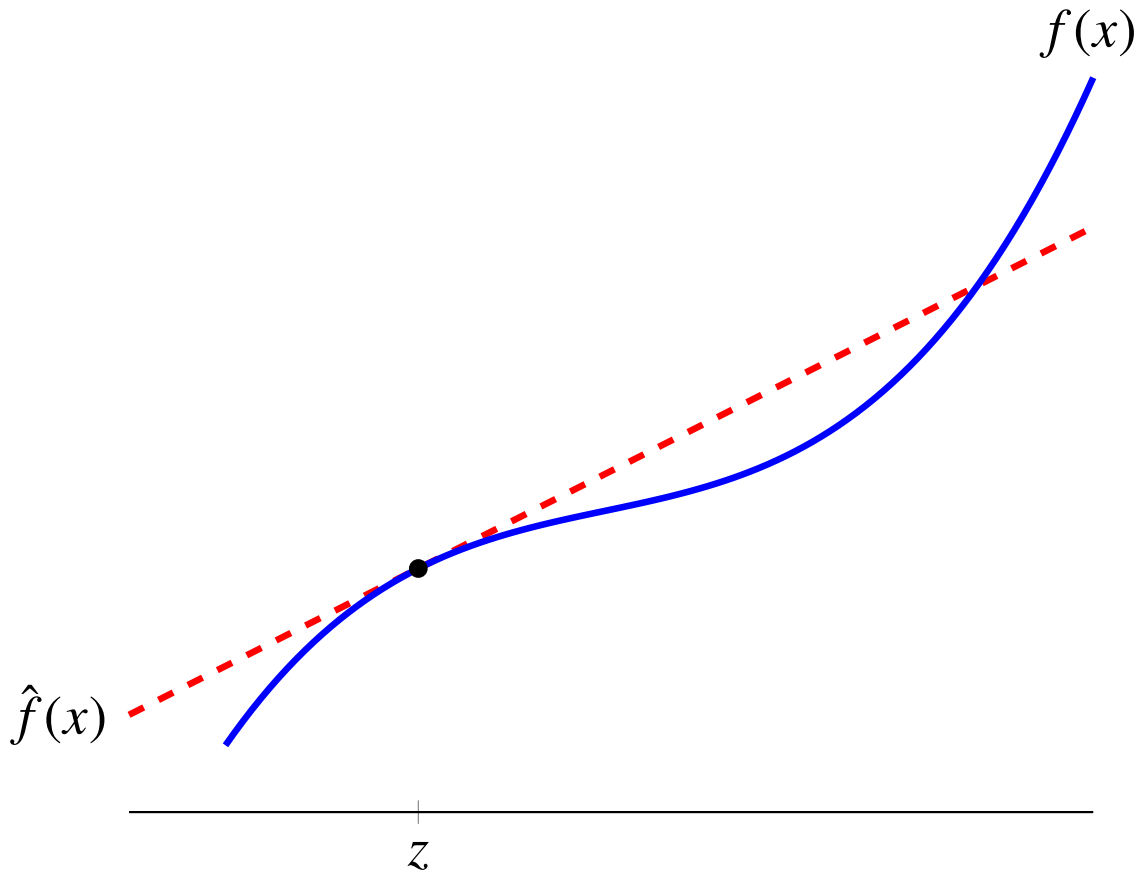
- $\hat{f}(x)$ is *very* close to $f(x)$ when x_i are all near z_i
- $\hat{f}$ is an affine function of x
- can write using inner product as

$$\hat{f}(x) = f(z) + \nabla f(z)^T (x - z)$$

  where n-vector $\nabla f(z)$ is the *gradient* of f at z,

$$\nabla f(z) = \left(\frac{\partial f}{\partial x_1}(z), \ldots, \frac{\partial f}{\partial x_n}(z) \right)$$

Example



Outline

Linear and affine functions

Taylor approximation

Regression model

3. Norm and distance

Outline

Norm

- the *Euclidean norm* (or just *norm*) of an n-vector x is

$$\|x\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} = \sqrt{x^T x}$$

- used to measure the size of a vector
- reduces to absolute value for $n = 1$

Properties

for any n-vectors x and y, and any scalar β

- *homogeneity:* $\|\beta x\| = |\beta| \|x\|$
- *triangle inequality:* $\|x + y\| \leq \|x\| + \|y\|$
- *nonnegativity:* $\|x\| \geq 0$
- *definiteness:* $\|x\| = 0$ only if $x = 0$

easy to show except triangle inequality, which we show later

RMS value

- *mean-square value* of n-vector x is

$$\frac{x_1^2 + \cdots + x_n^2}{n} = \frac{\|x\|^2}{n}$$

- *root-mean-square value* (RMS value) is

$$\mathbf{rms}(x) = \sqrt{\frac{x_1^2 + \cdots + x_n^2}{n}} = \frac{\|x\|}{\sqrt{n}}$$

- $\mathbf{rms}(x)$ gives 'typical' value of $|x_i|$
- *e.g.*, $\mathbf{rms}(\mathbf{1}) = 1$ (independent of n)
- RMS value useful for comparing sizes of vectors of different lengths

Norm of block vectors

- suppose a, b, c are vectors
- $\|(a,b,c)\|^2 = a^T a + b^T b + c^T c = \|a\|^2 + \|b\|^2 + \|c\|^2$
- so we have

$$\|(a,b,c)\| = \sqrt{\|a\|^2 + \|b\|^2 + \|c\|^2} = \|(\|a\|, \|b\|, \|c\|)\|$$

  (parse RHS very carefully!)

- we'll use these ideas later

Outline

Norm

Distance

Standard deviation

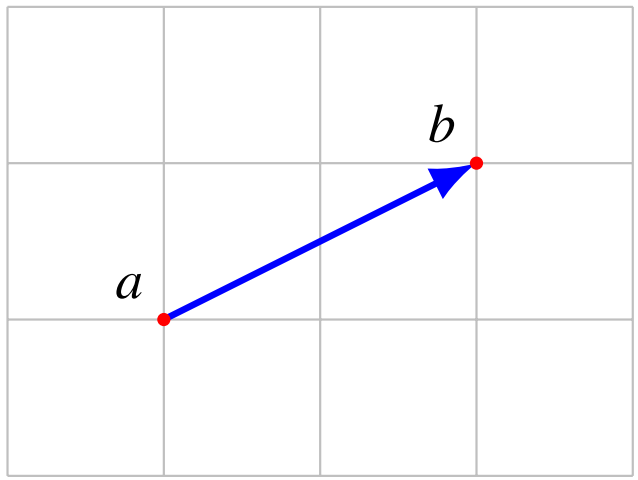
Angle

Distance

- (Euclidean) *distance* between n-vectors a and b is

$$\mathbf{dist}(a,b) = \|a-b\|$$

- agrees with ordinary distance for $n = 1,2,3$



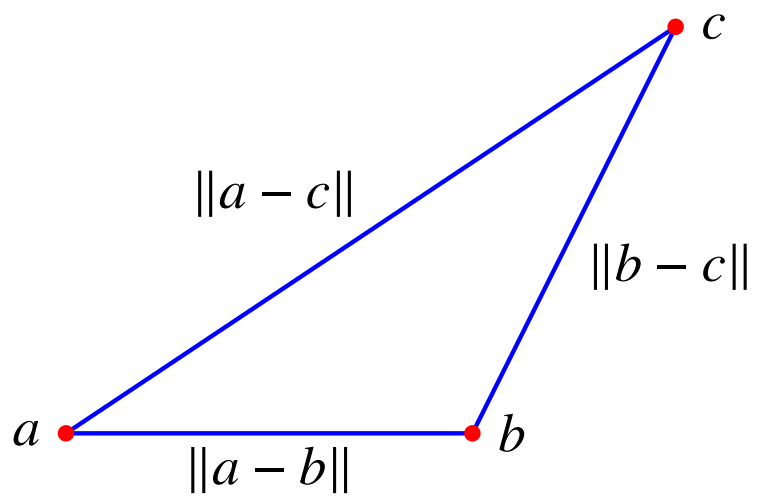
- $\mathbf{rms}(a-b)$ is the *RMS deviation* between a and b

Triangle inequality

- triangle with vertices at positions a, b, c
- edge lengths are $\|a-b\|$, $\|b-c\|$, $\|a-c\|$
- by triangle inequality

$$\|a-c\| = \|(a-b)+(b-c)\| \leq \|a-b\| + \|b-c\|$$

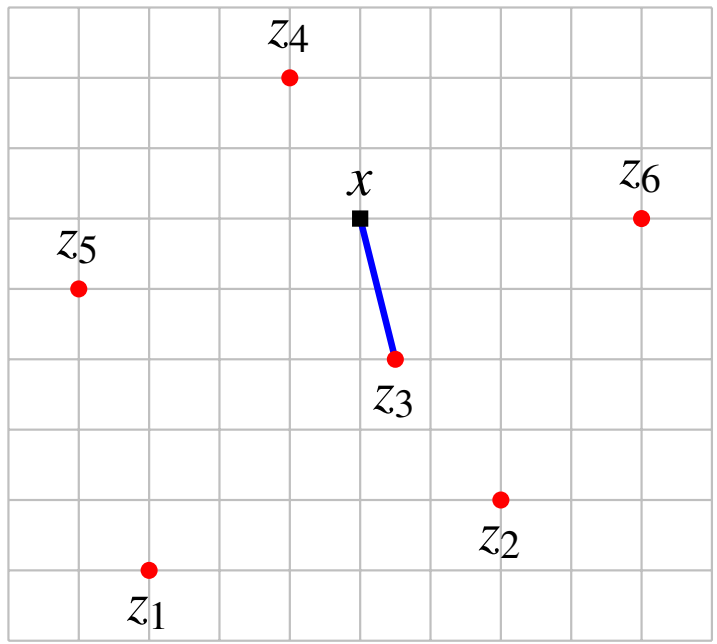
i.e., third edge length is no longer than sum of other two



Feature distance and nearest neighbors

- if x and y are feature vectors for two entities, $\|x - y\|$ is the *feature distance*
- if $z_1, \ldots, z_m$ is a list of vectors, z_j is the *nearest neighbor* of x if

$$\|x - z_j\| \leq \|x - z_i\|, \quad i = 1, \ldots, m$$



- these simple ideas are very widely used

Document dissimilarity

- 5 Wikipedia articles: 'Veterans Day', 'Memorial Day', 'Academy Awards', 'Golden Globe Awards', 'Super Bowl'
- word count histograms, dictionary of 4423 words
- pairwise distances shown below

	Veterans Day	Memorial Day	Academy Awards	Golden Globe Awards	Super Bowl
Veterans Day	0	0.095	0.130	0.153	0.170
Memorial Day	0.095	0	0.122	0.147	0.164
Academy A.	0.130	0.122	0	0.108	0.164
Golden Globe A.	0.153	0.147	0.108	0	0.181
Super Bowl	0.170	0.164	0.164	0.181	0

Cauchy–Schwarz inequality

- for two n-vectors a and b, $|a^T b| \leq \|a\|\|b\|$
- written out,

$$|a_1 b_1 + \cdots + a_n b_n| \leq \left(a_1^2 + \cdots + a_n^2\right)^{1/2} \left(b_1^2 + \cdots + b_n^2\right)^{1/2}$$

- now we can show triangle inequality:

$$\begin{aligned} \|a+b\|^2 &= \|a\|^2 + 2a^T b + \|b\|^2 \\ &\leq \|a\|^2 + 2\|a\|\|b\| + \|b\|^2 \\ &= (\|a\| + \|b\|)^2 \end{aligned}$$

Derivation of Cauchy–Schwarz inequality

- it's clearly true if either a or b is 0
- so assume $\alpha = \|a\|$ and $\beta = \|b\|$ are nonzero
- we have

$$\begin{aligned} 0 &\leq \|\beta a - \alpha b\|^2 \\ &= \|\beta a\|^2 - 2(\beta a)^T(\alpha b) + \|\alpha b\|^2 \\ &= \beta^2\|a\|^2 - 2\beta\alpha(a^T b) + \alpha^2\|b\|^2 \\ &= 2\|a\|^2\|b\|^2 - 2\|a\|\,\|b\|(a^T b) \end{aligned}$$

- divide by $2\|a\|\,\|b\|$ to get $a^T b \leq \|a\|\,\|b\|$
- apply to $-a$, b to get other half of Cauchy–Schwarz inequality

Angle

- *angle* between two nonzero vectors a, b defined as

$$\angle(a,b) = \arccos\left(\frac{a^T b}{\|a\|\,\|b\|}\right)$$

- $\angle(a,b)$ is the number in $[0,\pi]$ that satisfies

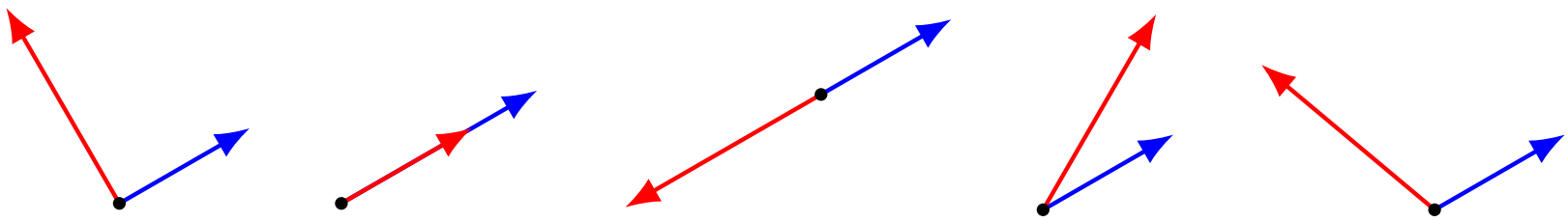
$$a^T b = \|a\|\,\|b\|\cos\left(\angle(a,b)\right)$$

- coincides with ordinary angle between vectors in 2-D and 3-D

Classification of angles

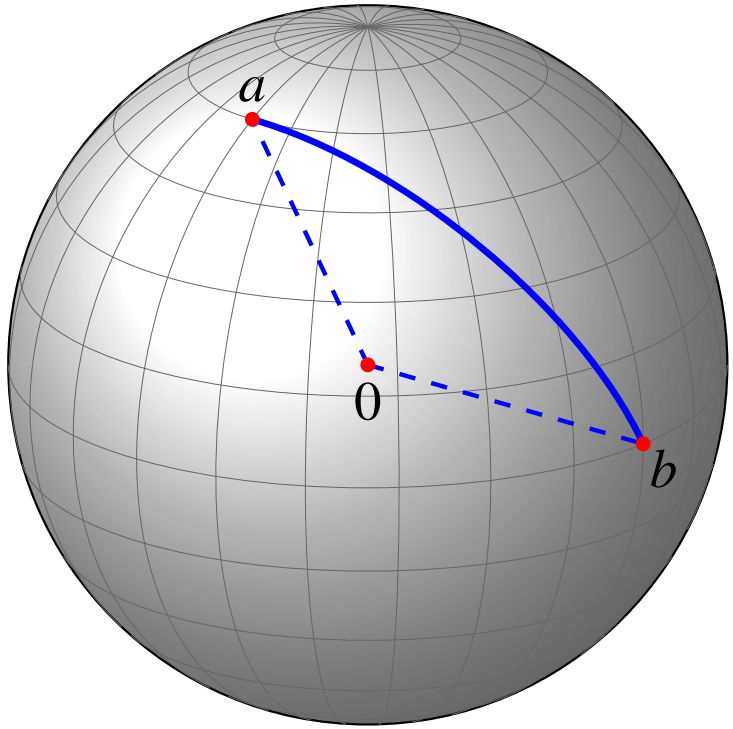
$\theta = \angle(a,b)$

- $\theta = \pi/2 = 90^\circ$: a and b are *orthogonal*, written $a \perp b$ ($a^T b = 0$)
- $\theta = 0$: a and b are *aligned* ($a^T b = \|a\|\|b\|$)
- $\theta = \pi = 180^\circ$: a and b are *anti-aligned* ($a^T b = -\|a\|\,\|b\|$)
- $\theta \leq \pi/2 = 90^\circ$: a and b make an *acute angle* ($a^T b \geq 0$)
- $\theta \geq \pi/2 = 90^\circ$: a and b make an *obtuse angle* ($a^T b \leq 0$)



Spherical distance

if a, b are on sphere of radius R, distance *along the sphere* is $R\angle(a,b)$



Document dissimilarity by angles

- measure dissimilarity by angle of word count histogram vectors
- pairwise angles (in degrees) for 5 Wikipedia pages shown below

	Veterans Day	Memorial Day	Academy Awards	Golden Globe Awards	Super Bowl
Veterans Day	0	60.6	85.7	87.0	87.7
Memorial Day	60.6	0	85.6	87.5	87.5
Academy A.	85.7	85.6	0	58.7	85.7
Golden Globe A.	87.0	87.5	58.7	0	86.0
Super Bowl	87.7	87.5	86.1	86.0	0

Correlation coefficient

- vectors a and b, and de-meaned vectors

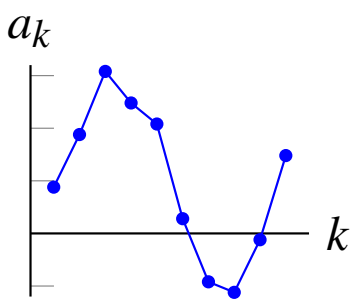
$$\tilde{a} = a - \mathbf{avg}(a)\mathbf{1}, \qquad \tilde{b} = b - \mathbf{avg}(b)\mathbf{1}$$

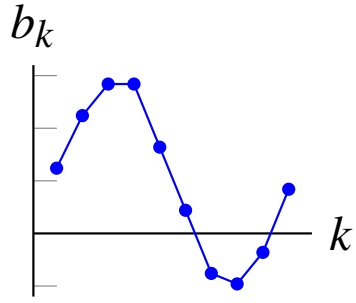
- *correlation coefficient* (between a and b, with $\tilde{a} \neq 0$, $\tilde{b} \neq 0$)

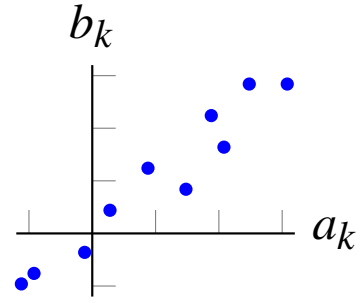
$$\rho = \frac{\tilde{a}^T \tilde{b}}{\|\tilde{a}\| \, \|\tilde{b}\|}$$

- $\rho = \cos \angle(\tilde{a}, \tilde{b})$
  - $\rho = 0$: a and b are *uncorrelated*
  - $\rho > 0.8$ (or so): a and b are *highly correlated*
  - $\rho < -0.8$ (or so): a and b are *highly anti-correlated*
- very roughly: highly correlated means a_i and b_i are typically both above (below) their means together

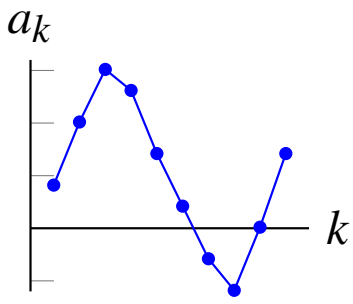
Examples

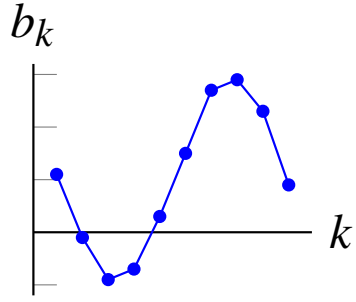


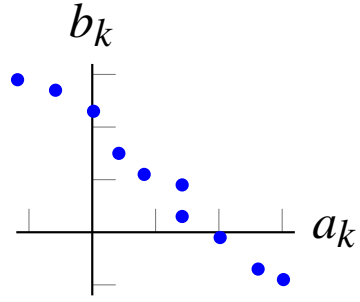




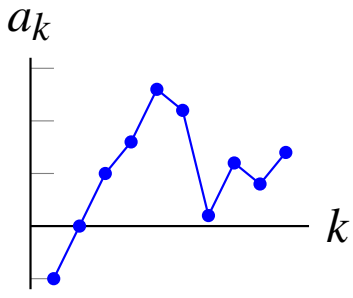
$\rho = 97\%$

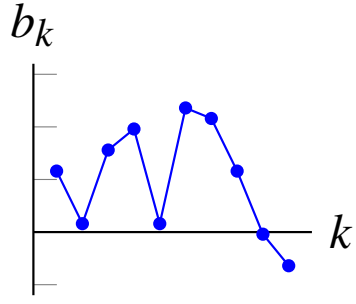


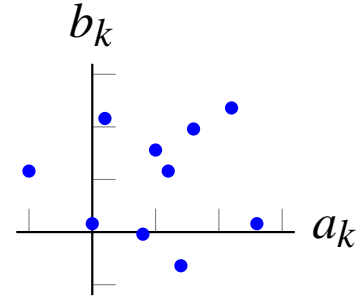




$\rho = -99\%$







$\rho = 0.4\%$

Examples

- highly correlated vectors:
  - rainfall time series at nearby locations
  - daily returns of similar companies in same industry
  - word count vectors of closely related documents (*e.g.*, same author, topic, ...)
  - sales of shoes and socks (at different locations or periods)
- approximately uncorrelated vectors
  - unrelated vectors
  - audio signals (even different tracks in multi-track recording)
- (somewhat) negatively correlated vectors
  - daily temperatures in Palo Alto and Melbourne

5. Linear independence

Outline

Linear dependence

- set of n-vectors $\{a_1, \ldots, a_k\}$ (with $k \geq 1$) is *linearly dependent* if

$$\beta_1 a_1 + \cdots + \beta_k a_k = 0$$

  holds for some $\beta_1, \ldots, \beta_k$, that are not all zero
- equivalent to: at least one a_i is a linear combination of the others
- we say '$a_1, \ldots, a_k$ are linearly dependent'

- $\{a_1\}$ is linearly dependent only if $a_1 = 0$
- $\{a_1, a_2\}$ is linearly dependent only if one a_i is a multiple of the other
- for more than two vectors, there is no simple to state condition

Example

- the vectors

$$a_1 = \begin{bmatrix} 0.2 \\ -7 \\ 8.6 \end{bmatrix}, \qquad a_2 = \begin{bmatrix} -0.1 \\ 2 \\ -1 \end{bmatrix}, \qquad a_3 = \begin{bmatrix} 0 \\ -1 \\ 2.2 \end{bmatrix}$$

  are linearly dependent, since $a_1 + 2a_2 - 3a_3 = 0$

- can express any of them as linear combination of the other two, *e.g.*,

$$a_2 = (-1/2)a_1 + (3/2)a_3$$

Linear independence

- set of n-vectors $\{a_1, \ldots, a_k\}$ (with $k \geq 1$) is *linearly independent* if it is not linearly dependent, *i.e.*,

$$\beta_1 a_1 + \cdots + \beta_k a_k = 0$$

  holds only when $\beta_1 = \cdots = \beta_k = 0$

- we say '$a_1, \ldots, a_k$ are linearly independent'
- equivalent to: no a_i is a linear combination of the others

- example: the unit n-vectors $e_1, \ldots, e_n$ are linearly independent

Linear combinations of linearly independent vectors

- suppose x is linear combination of linearly independent vectors $a_1, \ldots, a_k$:

$$x = \beta_1 a_1 + \cdots + \beta_k a_k$$

- the coefficients $\beta_1, \ldots, \beta_k$ are *unique*, *i.e.*, if

$$x = \gamma_1 a_1 + \cdots + \gamma_k a_k$$

  then $\beta_i = \gamma_i$ for $i = 1, \ldots, k$

- this means that (in principle) we can deduce the coefficients from x
- to see why, note that

$$(\beta_1 - \gamma_1) a_1 + \cdots + (\beta_k - \gamma_k) a_k = 0$$

  and so (by linear independence) $\beta_1 - \gamma_1 = \cdots = \beta_k - \gamma_k = 0$

Outline

Independence-dimension inequality

- *a linearly independent set of n-vectors can have at most n elements*
- put another way: *any set of $n + 1$ or more n-vectors is linearly dependent*

Basis

- a set of n linearly independent n-vectors $a_1, \ldots, a_n$ is called a *basis*
- any n-vector b can be expressed as a linear combination of them:

$$b = \beta_1 a_1 + \cdots + \beta_n a_n$$

  for some $\beta_1, \ldots, \beta_n$
- and these coefficients are unique
- formula above is called *expansion of b in the $a_1, \ldots, a_n$ basis*
- example: $e_1, \ldots, e_n$ is a basis, expansion of b is

$$b = b_1 e_1 + \cdots + b_n e_n$$

Outline

Orthonormal vectors

- set of n-vectors $a_1, \ldots, a_k$ are *(mutually) orthogonal* if $a_i \perp a_j$ for $i \neq j$
- they are *normalized* if $\|a_i\| = 1$ for $i = 1, \ldots, k$
- they are *orthonormal* if both hold
- can be expressed using inner products as

$$a_i^T a_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

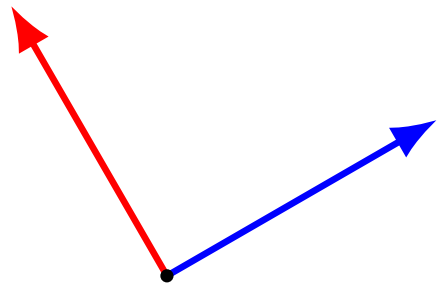
- orthonormal sets of vectors are linearly independent
- by independence-dimension inequality, must have $k \leq n$
- when $k = n$, $a_1, \ldots, a_n$ are an *orthonormal basis*

Examples of orthonormal bases

- standard unit n-vectors $e_1, \ldots, e_n$
- the 3-vectors

$$\begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}, \qquad \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \qquad \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

- the 2-vectors shown below



Orthonormal expansion

- if $a_1, \ldots, a_n$ is an orthonormal basis, we have for any n-vector x

$$x = (a_1^T x)a_1 + \cdots + (a_n^T x)a_n$$

- called *orthonormal expansion of* x (in the orthonormal basis)
- to verify formula, take inner product of both sides with a_i

Outline

Linear independence

Basis

Orthonormal vectors

Gram–Schmidt algorithm

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

Gram–Schmidt algorithm

---

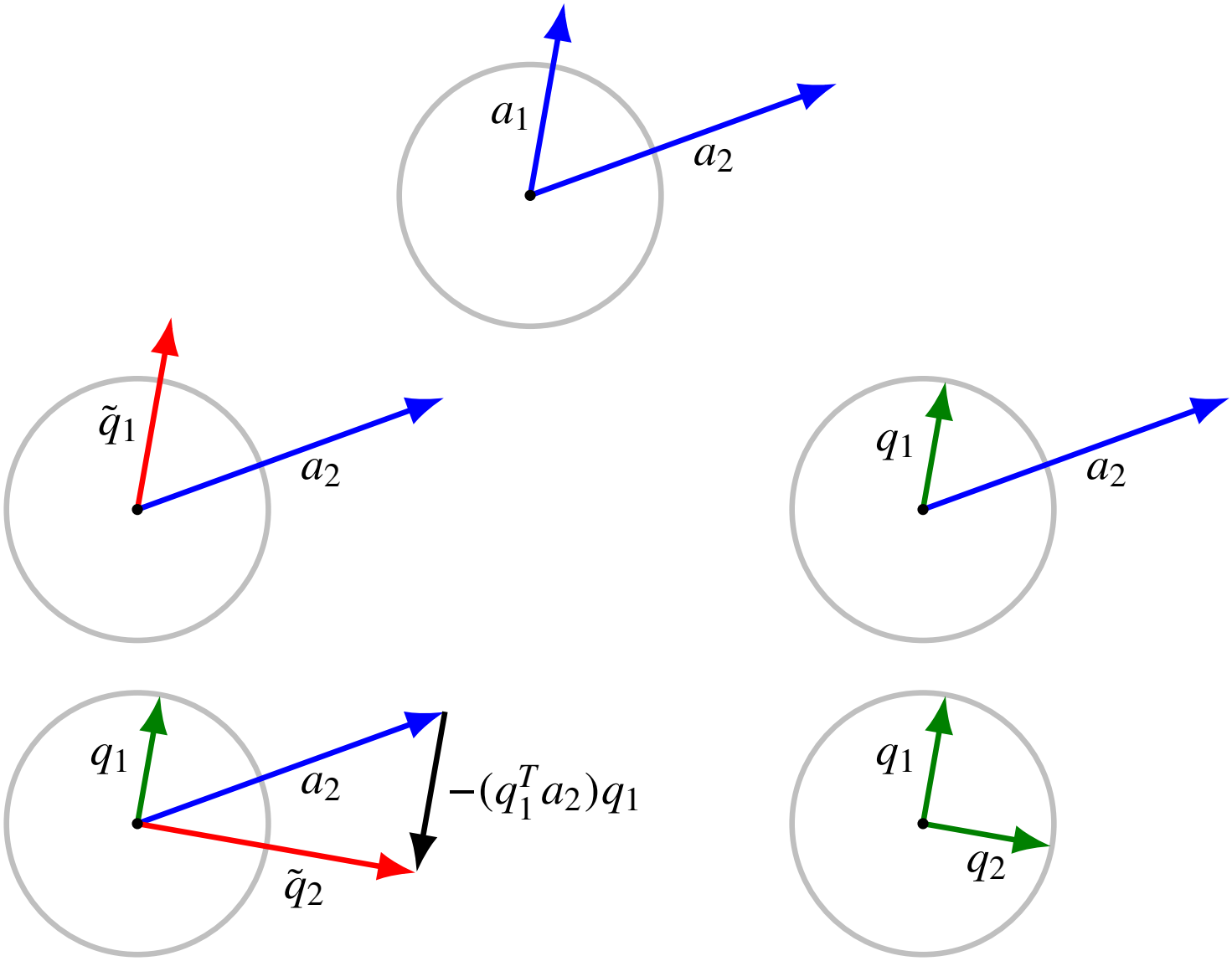
given n-vectors $a_1, \ldots, a_k$

for $i = 1, \ldots, k$

1. *Orthogonalization:* $\tilde{q}_i = a_i - (q_1^T a_i)q_1 - \cdots - (q_{i-1}^T a_i)q_{i-1}$
2. *Test for linear dependence:* if $\tilde{q}_i = 0$, quit
3. *Normalization:* $q_i = \tilde{q}_i / \|\tilde{q}_i\|$

- if G–S does not stop early (in step 2), $a_1, \ldots, a_k$ are linearly independent
- if G–S stops early in iteration $i = j$, then a_j is a linear combination of $a_1, \ldots, a_{j-1}$ (so $a_1, \ldots, a_k$ are linearly dependent)

Example



Analysis

let's show by induction that $q_1, \ldots, q_i$ are orthonormal

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

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

- to see this, take inner product of both sides with q_j, $j < i$

$$\begin{aligned} q_j^T \tilde{q}_i &= q_j^T a_i - (q_1^T a_i)(q_j^T q_1) - \cdots - (q_{i-1}^T a_i)(q_j^T q_{i-1}) \\ &= q_j^T a_i - q_j^T a_i = 0 \end{aligned}$$

- so $q_i \perp q_1, \ldots, q_i \perp q_{i-1}$
- normalization step ensures that $\|q_i\| = 1$

Analysis

assuming G–S has not terminated before iteration i

- a_i is a linear combination of $q_1, \ldots, q_i$:

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

- q_i is a linear combination of $a_1, \ldots, a_i$: by induction on i,

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

  and (by induction assumption) each $q_1, \ldots, q_{i-1}$ is a linear combination of $a_1, \ldots, a_{i-1}$

Early termination

suppose G–S terminates in step j

- a_j is linear combination of $q_1, \ldots, q_{j-1}$

$$a_j = (q_1^T a_j) q_1 + \cdots + (q_{j-1}^T a_j) q_{j-1}$$

- and each of $q_1, \ldots, q_{j-1}$ is linear combination of $a_1, \ldots, a_{j-1}$
- so a_j is a linear combination of $a_1, \ldots, a_{j-1}$

Complexity of Gram–Schmidt algorithm

- step 1 of iteration i requires $i-1$ inner products,

$$q_1^T a_i, \ldots, q_{i-1}^T a_i$$

  which costs $(i-1)(2n-1)$ flops

- $2n(i-1)$ flops to compute $\tilde{q}_i$
- $3n$ flops to compute $\|\tilde{q}_i\|$ and q_i
- total is

$$\sum_{i=1}^{k}((4n-1)(i-1)+3n) = (4n-1)\frac{k(k-1)}{2} + 3nk \approx 2nk^2$$

  using $\sum_{i=1}^{k}(i-1) = k(k-1)/2$

6. Matrices

Outline

Matrices

Matrix-vector multiplication

Examples

Matrices

- a *matrix* is a rectangular array of numbers, *e.g.*,

$$\begin{bmatrix} 0 & 1 & -2.3 & 0.1 \\ 1.3 & 4 & -0.1 & 0 \\ 4.1 & -1 & 0 & 1.7 \end{bmatrix}$$

- its *size* is given by (row dimension) $\times$ (column dimension)
  e.g., matrix above is 3×4
- *elements* also called *entries* or *coefficients*
- B_{ij} is i,j element of matrix B
- i is the *row index*, j is the *column index*; indexes start at 1
- two matrices are *equal* (denoted with $=$) if they are the same size and corresponding entries are equal

Matrix shapes

an $m \times n$ matrix A is

- *tall* if $m > n$
- *wide* if $m < n$
- *square* if $m = n$

Column and row vectors

- we consider an $n \times 1$ matrix to be an n-vector
- we consider a 1×1 matrix to be a number
- a $1 \times n$ matrix is called a *row vector*, *e.g.*,

$$\begin{bmatrix} 1.2 & -0.3 & 1.4 & 2.6 \end{bmatrix}$$

  which is *not* the same as the (column) vector

$$\begin{bmatrix} 1.2 \\ -0.3 \\ 1.4 \\ 2.6 \end{bmatrix}$$

Columns and rows of a matrix

- suppose A is an $m \times n$ matrix with entries A_{ij} for $i = 1, \ldots, m$, $j = 1, \ldots, n$
- its jth *column* is (the m-vector)

$$\begin{bmatrix} A_{1j} \\ \vdots \\ A_{mj} \end{bmatrix}$$

- its ith *row* is (the n-row-vector)

$$\begin{bmatrix} A_{i1} & \cdots & A_{in} \end{bmatrix}$$

- *slice* of matrix: $A_{p:q,r:s}$ is the $(q-p+1) \times (s-r+1)$ matrix

$$A_{p:q,r:s} = \begin{bmatrix} A_{pr} & A_{p,r+1} & \cdots & A_{ps} \\ A_{p+1,r} & A_{p+1,r+1} & \cdots & A_{p+1,s} \\ \vdots & \vdots & & \vdots \\ A_{qr} & A_{q,r+1} & \cdots & A_{qs} \end{bmatrix}$$

Block matrices

- we can form *block matrices*, whose entries are matrices, such as

$$A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}$$

  where B, C, D, and E are matrices (called *submatrices* or *blocks* of A)
- matrices in each block row must have same height (row dimension)
- matrices in each block column must have same width (column dimension)
- example: if

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

  then

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

Column and row representation of matrix

- A is an $m \times n$ matrix
- can express as block matrix with its (m-vector) columns $a_1, \ldots, a_n$

$$A = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$$

- or as block matrix with its (n-row-vector) rows $b_1, \ldots, b_m$

$$A = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Examples

- *image:* X_{ij} is i,j pixel value in a monochrome image
- *rainfall data:* A_{ij} is rainfall at location i on day j
- *multiple asset returns:* R_{ij} is return of asset j in period i
- *contingency table:* A_{ij} is number of objects with first attribute i and second attribute j
- *feature matrix:* X_{ij} is value of feature i for entity j

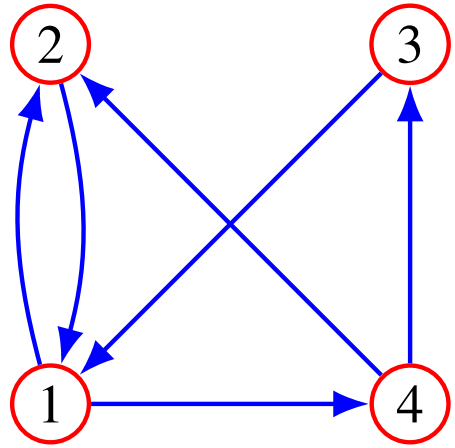
in each of these, what do the rows and columns mean?

Graph or relation

- a *relation* is a set of pairs of *objects*, labeled $1, \ldots, n$, such as

$$\mathcal{R} = \{(1,2),\ (1,3),\ (2,1),\ (2,4),\ (3,4),\ (4,1)\}$$

- same as *directed graph*



- can be represented as $n \times n$ matrix with $A_{ij} = 1$ if $(i,j) \in \mathcal{R}$

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

Special matrices

- *$m \times n$ zero matrix* has all entries zero, written as $0_{m\times n}$ or just 0
- *identity matrix* is square matrix with $I_{ii} = 1$ and $I_{ij} = 0$ for $i \neq j$, *e.g.*,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- *sparse matrix:* most entries are zero
  - examples: 0 and I
  - can be stored and manipulated efficiently
  - $\mathbf{nnz}(A)$ is number of nonzero entries

Diagonal and triangular matrices

- *diagonal matrix*: square matrix with $A_{ij} = 0$ when $i \neq j$
- $\mathbf{diag}(a_1, \ldots, a_n)$ denotes the diagonal matrix with $A_{ii} = a_i$ for $i = 1, \ldots, n$
- example:

$$\mathbf{diag}(0.2, -3, 1.2) = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 1.2 \end{bmatrix}$$

- *lower triangular matrix*: $A_{ij} = 0$ for $i < j$
- *upper triangular matrix*: $A_{ij} = 0$ for $i > j$
- examples:

$$\begin{bmatrix} 1 & -1 & 0.7 \\ 0 & 1.2 & -1.1 \\ 0 & 0 & 3.2 \end{bmatrix} \text{ (upper triangular)}, \qquad \begin{bmatrix} -0.6 & 0 \\ -0.3 & 3.5 \end{bmatrix} \text{ (lower triangular)}$$

Transpose

- the *transpose* of an $m \times n$ matrix A is denoted A^T, and defined by

$$(A^T)_{ij} = A_{ji}, \quad i = 1, \ldots, n, \quad j = 1, \ldots, m$$

- for example,

$$\begin{bmatrix} 0 & 4 \\ 7 & 0 \\ 3 & 1 \end{bmatrix}^T = \begin{bmatrix} 0 & 7 & 3 \\ 4 & 0 & 1 \end{bmatrix}$$

- transpose converts column to row vectors (and vice versa)
- $(A^T)^T = A$

Addition, subtraction, and scalar multiplication

- (just like vectors) we can add or subtract matrices of the same size:

$$(A+B)_{ij} = A_{ij} + B_{ij}, \quad i = 1,\ldots,m, \quad j = 1,\ldots,n$$

  (subtraction is similar)

- scalar multiplication:

$$(\alpha A)_{ij} = \alpha A_{ij}, \quad i = 1,\ldots,m, \quad j = 1,\ldots,n$$

- many obvious properties, *e.g.*,

$$A + B = B + A, \qquad \alpha(A+B) = \alpha A + \alpha B, \qquad (A+B)^T = A^T + B^T$$

Matrix norm

- for $m \times n$ matrix A, we define

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

- agrees with vector norm when $n = 1$
- satisfies norm properties:

$$\|\alpha A\| = |\alpha| \|A\|$$

$$\|A + B\| \leq \|A\| + \|B\|$$

$$\|A\| \geq 0$$

$$\|A\| = 0 \text{ only if } A = 0$$

- distance between two matrices: $\|A - B\|$
- (there are other matrix norms, which we won't use)

Outline

Matrices

Matrix-vector multiplication

Examples

Matrix-vector product

- *matrix-vector product* of $m \times n$ matrix A, n-vector x, denoted $y = Ax$, with

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

- for example,

$$\begin{bmatrix} 0 & 2 & -1 \\ -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 3 \\ -4 \end{bmatrix}$$

Row interpretation

- $y = Ax$ can be expressed as

$$y_i = b_i^T x, \quad i = 1, \ldots, m$$

  where $b_1^T, \ldots, b_m^T$ are rows of A
- so $y = Ax$ is a 'batch' inner product of all rows of A with x
- example: $A\mathbf{1}$ is vector of row sums of matrix A

Column interpretation

- $y = Ax$ can be expressed as

$$y = x_1 a_1 + x_2 a_2 + \cdots + x_n a_n$$

  where $a_1, \ldots, a_n$ are columns of A
- so $y = Ax$ is linear combination of columns of A, with coefficients $x_1, \ldots, x_n$
- important example: $Ae_j = a_j$
- columns of A are linearly independent if $Ax = 0$ implies $x = 0$

Outline

Matrices

Matrix-vector multiplication

Examples

General examples

- $0x = 0$, *i.e.*, multiplying by zero matrix gives zero
- $Ix = x$, *i.e.*, multiplying by identity matrix does nothing
- inner product $a^T b$ is matrix-vector product of $1 \times n$ matrix a^T and n-vector b
- $\tilde{x} = Ax$ is de-meaned version of x, with

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

Difference matrix

- $(n-1) \times n$ *difference matrix* is

$$D = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

  $y = Dx$ is $(n-1)$-vector of differences of consecutive entries of x:

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

- *Dirichlet energy:* $\|Dx\|^2$ is measure of wiggliness for x a time series

Return matrix – portfolio vector

- R is $T \times n$ matrix of asset returns
- R_{ij} is return of asset j in period i (say, in percentage)
- n-vector w gives portfolio (investments in the assets)
- T-vector Rw is time series of the portfolio return
- $\mathbf{avg}(Rw)$ is the portfolio (mean) return, $\mathbf{std}(Rw)$ is its risk

Feature matrix – weight vector

- $X = [x_1 \ \cdots \ x_N]$ is $n \times N$ *feature matrix*
- column x_j is feature n-vector for object or example j
- X_{ij} is value of feature i for example j
- n-vector w is weight vector
- $s = X^T w$ is vector of scores for each example; $s_j = x_j^T w$

Input – output matrix

- A is $m \times n$ matrix
- $y = Ax$
- n-vector x is *input* or *action*
- m-vector y is *output* or *result*
- A_{ij} is the factor by which y_i depends on x_j
- A_{ij} is the *gain* from input j to output i
- *e.g.*, if A is lower triangular, then y_i only depends on $x_1, \ldots, x_i$

Complexity

- $m \times n$ matrix stored A as $m \times n$ array of numbers
  (for sparse A, store only $\mathbf{nnz}(A)$ nonzero values)
- matrix addition, scalar-matrix multiplication cost mn flops
- matrix-vector multiplication costs $m(2n-1) \approx 2mn$ flops
  (for sparse A, around $2\mathbf{nnz}(A)$ flops)

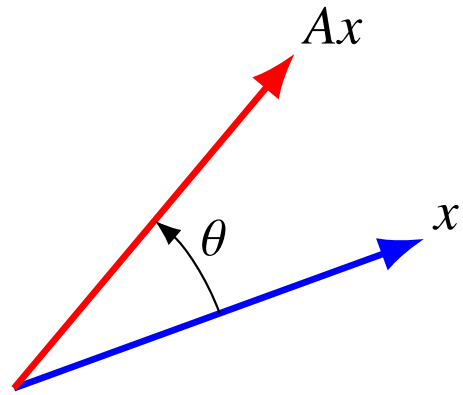
7. Matrix examples

Outline

Geometric transformations

- many geometric transformations and mappings of 2-D and 3-D vectors can be represented via matrix multiplication $y = Ax$
- for example, rotation by θ:

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



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

Outline

Geometric transformations

Selectors

Incidence matrix

Convolution

Selectors

- an $m \times n$ *selector matrix*: each row is a unit vector (transposed)

$$A = \begin{bmatrix} e_{k_1}^T \\ \vdots \\ e_{k_m}^T \end{bmatrix}$$

- multiplying by A selects entries of x:

$$Ax = (x_{k_1}, x_{k_2}, \ldots, x_{k_m})$$

- example: the $m \times 2m$ matrix

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

  'down-samples' by 2: if x is a $2m$-vector then $y = Ax = (x_1, x_3, \ldots, x_{2m-1})$

- other examples: image cropping, permutation, ...

Outline

Convolution

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

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

- for example with $n = 4$, $m = 3$, we have

$$\begin{aligned}
c_1 &= a_1 b_1 \\
c_2 &= a_1 b_2 + a_2 b_1 \\
c_3 &= a_1 b_3 + a_2 b_2 + a_3 b_1 \\
c_4 &= a_2 b_3 + a_3 b_2 + a_4 b_1 \\
c_5 &= a_3 b_3 + a_4 b_2 \\
c_6 &= a_4 b_3
\end{aligned}$$

- example: $(1, 0, -1) * (2, 1, -1) = (2, 1, -3, -1, 1)$

Polynomial multiplication

- a and b are coefficients of two polynomials:

$$p(x) = a_1 + a_2 x + \cdots + a_n x^{n-1}, \qquad q(x) = b_1 + b_2 x + \cdots + b_m x^{m-1}$$

- convolution $c = a * b$ gives the coefficients of the product $p(x)q(x)$:

$$p(x)q(x) = c_1 + c_2 x + \cdots + c_{n+m-1} x^{n+m-2}$$

- this gives simple proofs of many properties of convolution; for example,

$$a * b = b * a$$

$$(a * b) * c = a * (b * c)$$

$a * b = 0$ only if $a = 0$ or $b = 0$

Toeplitz matrices

- can express $c = a * b$ using matrix-vector multiplication as $c = T(b)a$, with

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

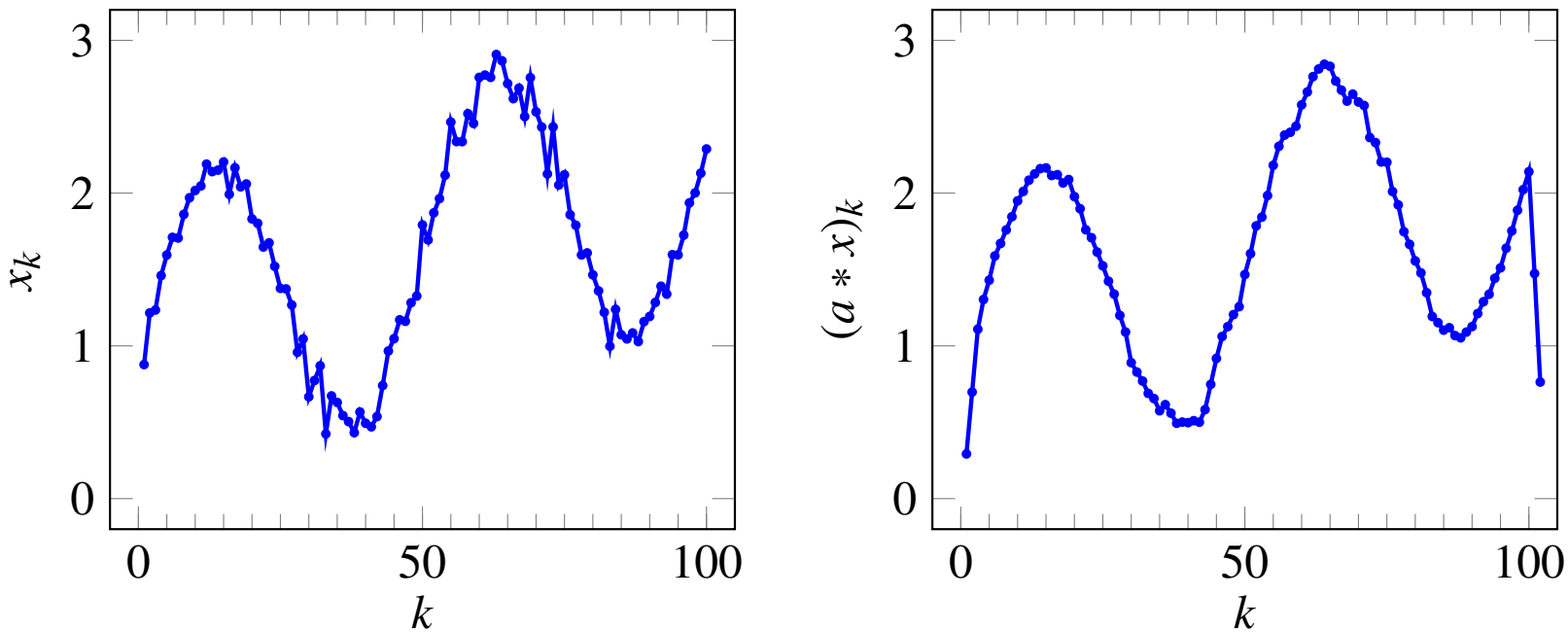
- $T(b)$ is a Toeplitz matrix (values on diagonals are equal)

Moving average of time series

- n-vector x represents a time series
- convolution $y = a * x$ with $a = (1/3, 1/3, 1/3)$ is 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 interpreted as zero for $k < 1$ and $k > n$)



Input-output convolution system

- m-vector u represents a time series *input*
- $m + n - 1$ vector y represents a time series *output*
- $y = h * u$ is a *convolution model*
- n-vector h is called the *system impulse response*
- we have

$$y_i = \sum_{j=1}^{n} u_{i-j+1} h_j$$

  (interpreting u_k as zero for $k < n$ or $k > n$)

- interpretation: y_i, output at time i is a linear combination of $u_i, \ldots, u_{i-n+1}$
- h_3 is the factor by which current output depends on what the input was 2 time steps before

10. Matrix multiplication

Outline

Matrix multiplication

Composition of linear functions

Matrix powers

QR factorization

Matrix multiplication

- can multiply $m \times p$ matrix A and $p \times n$ matrix B to get $C = AB$:

$$C_{ij} = \sum_{k=1}^{p} A_{ik}B_{kj} = A_{i1}B_{1j} + \cdots + A_{ip}B_{pj}$$

  for $i = 1, \ldots, m, j = 1, \ldots, n$

- to get C_{ij}: move along ith row of A, jth column of B
- example:

$$\begin{bmatrix} -1.5 & 3 & 2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 \\ 0 & -2 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 3.5 & -4.5 \\ -1 & 1 \end{bmatrix}$$

Special cases of matrix multiplication

- scalar-vector product (with scalar on right!) $x\alpha$
- inner product $a^T b$
- matrix-vector multiplication Ax
- *outer product* of m-vector a and n-vector b

$$ab^T = \begin{bmatrix} a_1b_1 & a_1b_2 & \cdots & a_1b_n \\ a_2b_1 & a_2b_2 & \cdots & a_2b_n \\ \vdots & \vdots & & \vdots \\ a_mb_1 & a_mb_2 & \cdots & a_mb_n \end{bmatrix}$$

Properties

- $(AB)C = A(BC)$, so both can be written ABC
- $A(B + C) = AB + AC$
- $(AB)^T = B^T A^T$
- $AI = A$ and $IA = A$
- $AB = BA$ *does not hold in general*

Block matrices

block matrices can be multiplied using the same formula, *e.g.*,

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

(provided the products all make sense)

Column interpretation

- denote columns of B by b_i:

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

- then we have

$$\begin{aligned} AB &= A \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} \\ &= \begin{bmatrix} Ab_1 & Ab_2 & \cdots & Ab_n \end{bmatrix} \end{aligned}$$

- so AB is 'batch' multiply of A times columns of B

Multiple sets of linear equations

- given k systems of linear equations, with same $m \times n$ coefficient matrix

$$Ax_i = b_i, \quad i = 1, \ldots, k$$

- write in compact matrix form as $AX = B$
- $X = [x_1 \ \cdots \ x_k]$, $B = [b_1 \ \cdots \ b_k]$

Inner product interpretation

- with a_i^T the rows of A, b_j the columns of B, we have

$$AB = \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdots & a_1^T b_n \\ a_2^T b_1 & a_2^T b_2 & \cdots & a_2^T b_n \\ \vdots & \vdots & & \vdots \\ a_m^T b_1 & a_m^T b_2 & \cdots & a_m^T b_n \end{bmatrix}$$

- so matrix product is all inner products of rows of A and columns of B, arranged in a matrix

Gram matrix

- let A be an $m \times n$ matrix with columns $a_1, \ldots, a_n$
- the *Gram matrix* of A is

$$G = A^T A = \begin{bmatrix} a_1^T a_1 & a_1^T a_2 & \cdots & a_1^T a_n \\ a_2^T a_1 & a_2^T a_2 & \cdots & a_2^T a_n \\ \vdots & \vdots & \ddots & \vdots \\ a_n^T a_1 & a_n^T a_2 & \cdots & a_n^T a_n \end{bmatrix}$$

- Gram matrix gives all inner products of columns of A
- example: $G = A^T A = I$ means columns of A are orthonormal

Complexity

- to compute $C_{ij} = (AB)_{ij}$ is inner product of p-vectors
- so total required flops is $(mn)(2p) = 2mnp$ flops
- multiplying two 1000×1000 matrices requires 2 billion flops
- . . . and can be done in well under a second on current computers

Outline

Matrix multiplication

Composition of linear functions

Matrix powers

QR factorization

Composition of linear functions

- A is an $m \times p$ matrix, B is $p \times n$
- define $f : \mathbf{R}^p \to \mathbf{R}^m$ and $g : \mathbf{R}^n \to \mathbf{R}^p$ as

$$f(u) = Au, \qquad g(v) = Bv$$

- f and g are linear functions
- *composition* of f and g is $h : \mathbf{R}^n \to \mathbf{R}^m$ with $h(x) = f(g(x))$
- we have

$$h(x) = f(g(x)) = A(Bx) = (AB)x$$

- composition of linear functions is linear
- associated matrix is product of matrices of the functions

Second difference matrix

- D_n is $(n-1) \times n$ difference matrix:

$$D_n x = (x_2 - x_1, \ldots, x_n - x_{n-1})$$

- D_{n-1} is $(n-2) \times (n-1)$ difference matrix:

$$D_n y = (y_2 - y_1, \ldots, y_{n-1} - y_{n-2})$$

- $\Delta = D_{n-1} D_n$ is $(n-2) \times n$ second difference matrix:

$$\Delta x = (x_1 - 2x_2 + x_3, x_2 - 2x_3 + x_4, \ldots, x_{n-2} - 2x_{n-1} + x_n)$$

- for $n = 5$, $\Delta = D_{n-1} D_n$ is

$$\begin{bmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Outline

Matrix multiplication

Composition of linear functions

Matrix powers

QR factorization

Matrix powers

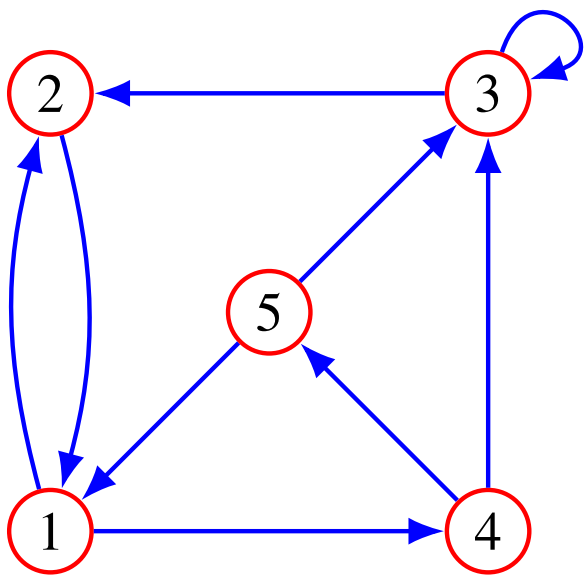
- for A square, A^2 means AA, and same for higher powers
- with convention $A^0 = I$ we have $A^k A^l = A^{k+l}$
- negative powers later; fractional powers in other courses

Directed graph

- $n \times n$ matrix A is adjacency matrix of directed graph:

$$A_{ij} = \begin{cases} 1 & \text{there is a edge from vertex } j \text{ to vertex } i \\ 0 & \text{otherwise} \end{cases}$$

- example:



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

Paths in directed graph

- square of adjacency matrix:

$$(A^2)_{ij} = \sum_{k=1}^{n} A_{ik}A_{kj}$$

- $(A^2)_{ij}$ is number of paths of length 2 from j to i
- for the example,

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 2 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

  e.g., there are two paths from 4 to 3 (via 3 and 5)

- more generally, $(A^\ell)_{ij}$ = number of paths of length ℓ from j to i

Outline

Matrix multiplication

Composition of linear functions

Matrix powers

QR factorization

Gram–Schmidt in matrix notation

- run Gram–Schmidt on columns $a_1, \ldots, a_k$ of $n \times k$ matrix A
- if columns are linearly independent, get orthonormal $q_1, \ldots, q_k$
- define $n \times k$ matrix Q with columns $q_1, \ldots, q_k$
- $Q^T Q = I$
- from Gram–Schmidt algorithm

$$\begin{aligned} a_i &= (q_1^T a_i) q_1 + \cdots + (q_{i-1}^T a_i) q_{i-1} + \|\tilde{q}_i\| q_i \\ &= R_{1i} q_1 + \cdots + R_{ii} q_i \end{aligned}$$

  with $R_{ij} = q_i^T a_j$ for $i < j$ and $R_{ii} = \|\tilde{q}_i\|$
- defining $R_{ij} = 0$ for $i > j$ we have $A = QR$
- R is upper triangular, with positive diagonal entries

QR factorization

- $A = QR$ is called *QR factorization* of A
- factors satisfy $Q^T Q = I$, R upper triangular with positive diagonal entries
- can be computed using Gram–Schmidt algorithm (or some variations)
- has a *huge* number of uses, which we'll see soon

11. Matrix inverses

Outline

Left inverses

- a number x that satisfies $xa = 1$ is called the inverse of a
- inverse (*i.e.*, $1/a$) exists if and only if $a \neq 0$, and is unique
- a matrix X that satisfies $XA = I$ is called a *left inverse* of A
- if a left inverse exists we say that A is *left-invertible*
- example: the matrix

$$A = \begin{bmatrix} -3 & -4 \\ 4 & 6 \\ 1 & 1 \end{bmatrix}$$

  has two different left inverses:

$$B = \frac{1}{9}\begin{bmatrix} -11 & -10 & 16 \\ 7 & 8 & -11 \end{bmatrix}, \qquad C = \frac{1}{2}\begin{bmatrix} 0 & -1 & 6 \\ 0 & 1 & -4 \end{bmatrix}$$

Left inverse and column independence

- if A has a left inverse C then the columns of A are linearly independent
- to see this: if $Ax = 0$ and $CA = I$ then

$$0 = C0 = C(Ax) = (CA)x = Ix = x$$

- we'll see later the converse is also true, so

  a matrix is left-invertible if and only if its columns are linearly independent

- matrix generalization of

  a number is invertible if and only if it is nonzero

- so left-invertible matrices are tall or square

Solving linear equations with a left inverse

- suppose $Ax = b$, and A has a left inverse C
- then $Cb = C(Ax) = (CA)x = Ix = x$
- so multiplying the right-hand side by a left inverse yields the solution

Example

$$A = \begin{bmatrix} -3 & -4 \\ 4 & 6 \\ 1 & 1 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix}$$

- over-determined equations $Ax = b$ have (unique) solution $x = (1, -1)$
- A has two different left inverses,

$$B = \frac{1}{9}\begin{bmatrix} -11 & -10 & 16 \\ 7 & 8 & -11 \end{bmatrix}, \qquad C = \frac{1}{2}\begin{bmatrix} 0 & -1 & 6 \\ 0 & 1 & -4 \end{bmatrix}$$

- multiplying the right-hand side with the left inverse B we get

$$Bb = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

- and also

$$Cb = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Right inverses

- a matrix X that satisfies $AX = I$ is a *right inverse* of A
- if a right inverse exists we say that A is *right-invertible*
- A is right-invertible if and only if A^T is left-invertible:

$$AX = I \iff (AX)^T = I \iff X^T A^T = I$$

- so we conclude

  A is right-invertible if and only if its rows are linearly independent

- right-invertible matrices are wide or square

Solving linear equations with a right inverse

- suppose A has a right inverse B
- consider the (square or underdetermined) equations $Ax = b$
- $x = Bb$ is a solution:

$$Ax = A(Bb) = (AB)b = Ib = b$$

- so $Ax = b$ has a solution for *any* b

Example

- same A, B, C in example above
- C^T and B^T are both right inverses of A^T
- under-determined equations $A^T x = (1,2)$ has (different) solutions

$$B^T(1,2) = (1/3, 2/3, -2/3), \qquad C^T(1,2) = (0, 1/2, -1)$$

(there are many other solutions as well)

Outline

Left and right inverses

Inverse

Solving linear equations

Examples

Pseudo-inverse

Inverse

- if A has a left and a right inverse, they are unique and equal (and we say that A is *invertible*)
- so A must be square
- to see this: if $AX = I$, $YA = I$

$$X = IX = (YA)X = Y(AX) = YI = Y$$

- we denote them by A^{-1}:

$$A^{-1}A = AA^{-1} = I$$

- inverse of inverse: $(A^{-1})^{-1} = A$

Solving square systems of linear equations

- suppose A is invertible
- for any b, $Ax = b$ has the unique solution

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

- matrix generalization of simple scalar equation $ax = b$ having solution $x = (1/a)b$ (for $a \neq 0$)

- simple-looking formula $x = A^{-1}b$ is basis for many applications

Invertible matrices

the following are equivalent for a square matrix A:

- A is invertible
- columns of A are linearly independent
- rows of A are linearly independent
- A has a left inverse
- A has a right inverse

if any of these hold, all others do

Examples

- $I^{-1} = I$
- if Q is orthogonal, *i.e.*, square with $Q^T Q = I$, then $Q^{-1} = Q^T$
- 2×2 matrix A is invertible if and only $A_{11}A_{22} \neq A_{12}A_{21}$

$$A^{-1} = \frac{1}{A_{11}A_{22} - A_{12}A_{21}} \begin{bmatrix} A_{22} & -A_{12} \\ -A_{21} & A_{11} \end{bmatrix}$$

  - you need to know this formula
  - there are similar but *much* more complicated formulas for larger matrices (and no, you do not need to know them)

Non-obvious example

$$A = \begin{bmatrix} 1 & -2 & 3 \\ 0 & 2 & 2 \\ -3 & -4 & -4 \end{bmatrix}$$

- A is invertible, with inverse

$$A^{-1} = \frac{1}{30} \begin{bmatrix} 0 & -20 & -10 \\ -6 & 5 & -2 \\ 6 & 10 & 2 \end{bmatrix}.$$

- verified by checking $AA^{-1} = I$ (or $A^{-1}A = I$)
- we'll soon see how to compute the inverse

Properties

- $(AB)^{-1} = B^{-1}A^{-1}$ (provided inverses exist)
- $(A^T)^{-1} = (A^{-1})^T$ (sometimes denoted A^{-T})
- negative matrix powers: $(A^{-1})^k$ is denoted A^{-k}
- with $A^0 = I$, identity $A^k A^l = A^{k+l}$ holds for any integers k, l

Triangular matrices

- lower triangular L with nonzero diagonal entries is invertible
- so see this, write $Lx = 0$ as

$$\begin{aligned} L_{11}x_1 &= 0 \\ L_{21}x_1 + L_{22}x_2 &= 0 \\ &\vdots \\ L_{n1}x_1 + L_{n2}x_2 + \cdots + L_{n,n-1}x_{n-1} + L_{nn}x_n &= 0 \end{aligned}$$

  - from first equation, $x_1 = 0$ (since $L_{11} \neq 0$)
  - second equation reduces to $L_{22}x_2 = 0$, so $x_2 = 0$ (since $L_{22} \neq 0$)
  - and so on

  this shows columns of L are linearly independent, so L is invertible

- upper triangular R with nonzero diagonal entries is invertible

Inverse via QR factorization

- suppose A is square and invertible
- so its columns are linearly independent
- so Gram–Schmidt gives QR factorization
  - $A = QR$
  - Q is orthogonal: $Q^T Q = I$
  - R is upper triangular with positive diagonal entries, hence invertible
- so we have

$$A^{-1} = (QR)^{-1} = R^{-1}Q^{-1} = R^{-1}Q^T$$

Outline

Back substitution

- suppose R is upper triangular with nonzero diagonal entries
- write out $Rx = b$ as

$$\begin{aligned} R_{11}x_1 + R_{12}x_2 + \cdots + R_{1,n-1}x_{n-1} + R_{1n}x_n &= b_1 \\ &\vdots \\ R_{n-1,n-1}x_{n-1} + R_{n-1,n}x_n &= b_{n-1} \\ R_{nn}x_n &= b_n \end{aligned}$$

- from last equation we get $x_n = b_n/R_{nn}$
- from 2nd to last equation we get

$$x_{n-1} = (b_{n-1} - R_{n-1,n}x_n)/R_{n-1,n-1}$$

- continue to get $x_{n-2}, x_{n-3}, \ldots, x_1$

Back substitution

- called *back substitution* since we find the variables in reverse order, substituting the already known values of x_i
- computes $x = R^{-1}b$
- complexity:
  - first step requires 1 flop (division)
  - 2nd step needs 3 flops
  - ith step needs $2i - 1$ flops

  total is $1 + 3 + \cdots + (2n - 1) = n^2$ flops

Solving linear equations via QR factorization

- assuming A is invertible, let's solve $Ax = b$, *i.e.*, compute $x = A^{-1}b$
- with QR factorization $A = QR$, we have

$$A^{-1} = (QR)^{-1} = R^{-1}Q^T$$

- compute $x = R^{-1}(Q^T b)$ by back substitution

Solving linear equations via QR factorization

given an $n \times n$ invertible matrix A and an n-vector b

1. *QR factorization:* compute the QR factorization $A = QR$
2. compute $Q^T b$.
3. *Back substitution:* Solve the triangular equation $Rx = Q^T b$ using back substitution

- complexity $2n^3$ (step 1), $2n^2$ (step 2), n^2 (step 3)
- total is $2n^3 + 3n^2 \approx 2n^3$

Multiple right-hand sides

- let's solve $Ax_i = b_i$, $i = 1, \ldots, k$, with A invertible
- carry out QR factorization *once* ($2n^3$ flops)
- for $i = 1, \ldots, k$, solve $Rx_i = Q^T b_i$ via back substitution ($3kn^2$ flops)
- total is $2n^3 + 3kn^2$ flops
- if k is small compared to n, *same cost as solving one set of equations*

Outline

Polynomial interpolation

- let's find coefficients of a cubic polynomial

$$p(x) = c_1 + c_2 x + c_3 x^2 + c_4 x^3$$

that satisfies

$$p(-1.1) = b_1, \qquad p(-0.4) = b_2, \qquad p(0.1) = b_3, \qquad p(0.8) = b_4$$

- write as $Ac = b$, with

$$A = \begin{bmatrix} 1 & -1.1 & (-1.1)^2 & (-1.1)^3 \\ 1 & -0.4 & (-0.4)^2 & (-0.4)^3 \\ 1 & 0.1 & (0.1)^2 & (0.1)^3 \\ 1 & 0.8 & (0.8)^2 & (0.8)^3 \end{bmatrix}$$

Polynomial interpolation

- (unique) coefficients given by $c = A^{-1}b$, with

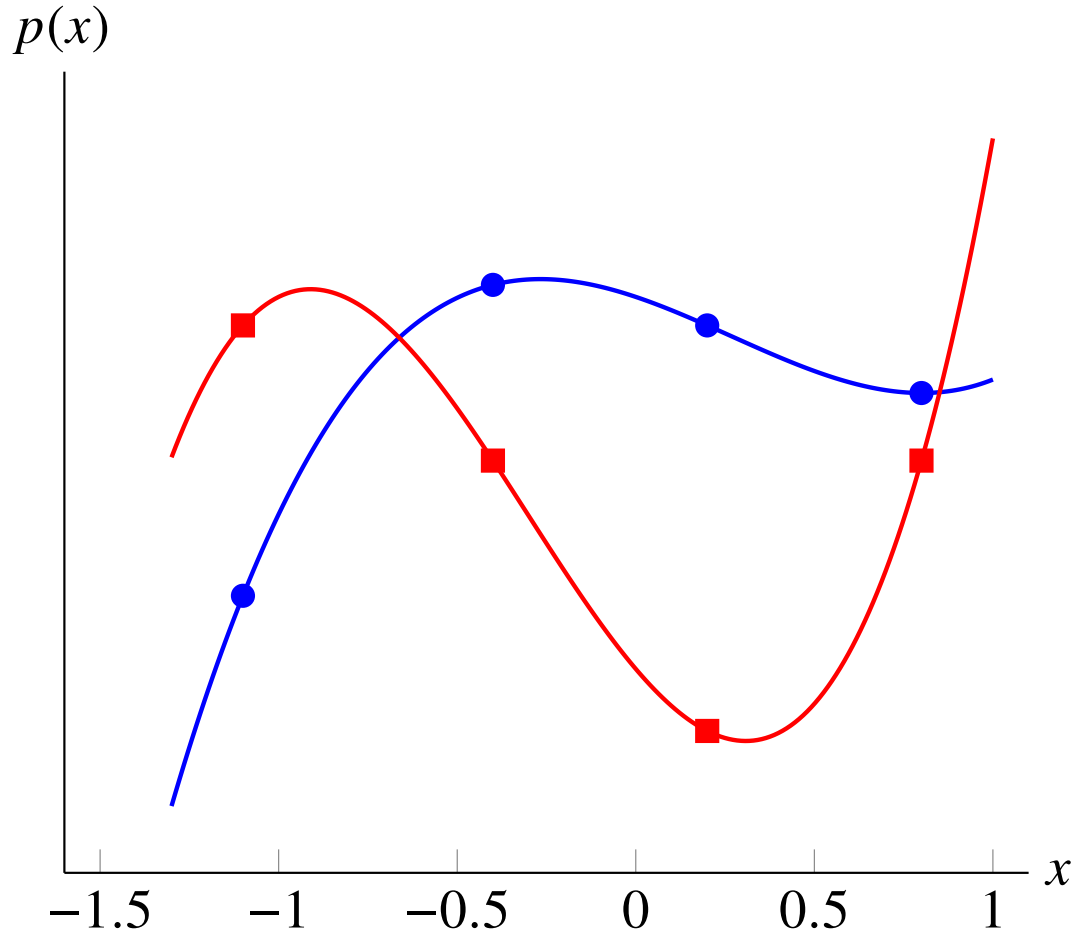
$$A^{-1} = \begin{bmatrix} -0.0370 & 0.3492 & 0.7521 & -0.0643 \\ 0.1388 & -1.8651 & 1.6239 & 0.1023 \\ 0.3470 & 0.1984 & -1.4957 & 0.9503 \\ -0.5784 & 1.9841 & -2.1368 & 0.7310 \end{bmatrix}$$

- so, *e.g.*, c_1 is not very sensitive to b_1 or b_4
- first column gives coefficients of polynomial that satisfies

$$p(-1.1) = 1, \qquad p(-0.4) = 0, \qquad p(0.1) = 0, \qquad p(0.8) = 0$$

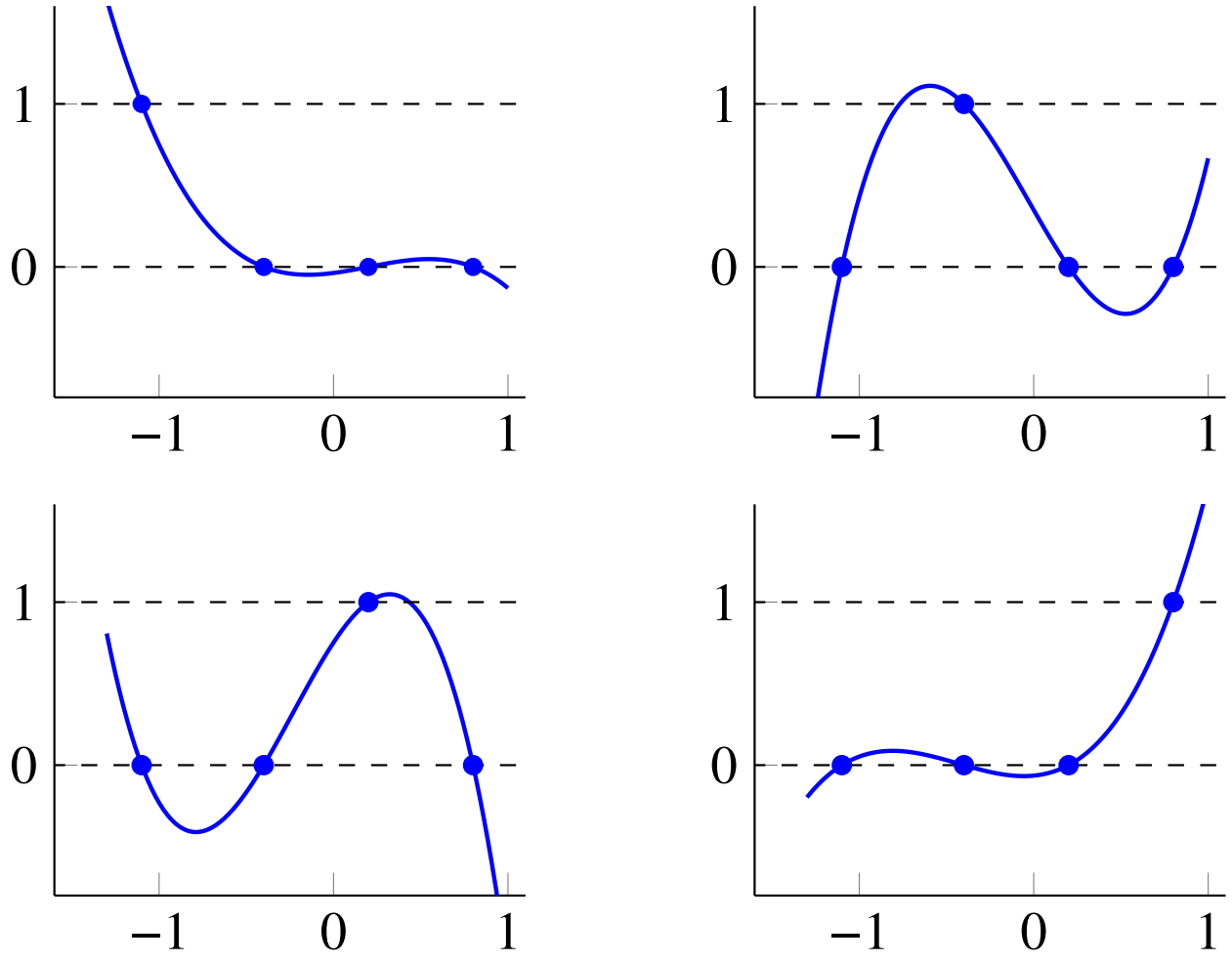
  called (first) *Lagrange polynomial*

Example



Lagrange polynomials

Lagrange polynomials associated with points $-1.1, -0.4, 0.2, 0.8$



Outline

Invertibility of Gram matrix

- A has linearly independent columns if and only if A^TA is invertible
- to see this, we'll show that $Ax = 0 \Leftrightarrow A^TAx = 0$
- $\Rightarrow$: if $Ax = 0$ then $(A^TA)x = A^T(Ax) = A^T0 = 0$
- $\Leftarrow$: if $(A^TA)x = 0$ then

$$0 = x^T(A^TA)x = (Ax)^T(Ax) = \|Ax\|^2 = 0$$

so $Ax = 0$

Pseudo-inverse of tall matrix

- the *pseudo-inverse* of A with independent columns is

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

- it is a left inverse of A:

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

  (we'll soon see that it's a very important left inverse of A)

- reduces to A^{-1} when A is square:

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

Pseudo-inverse of wide matrix

- if A is wide, with linearly independent rows, AA^T is invertible
- pseudo-inverse is defined as

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

- $A^\dagger$ is a right inverse of A:

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

  (we'll see later it is an important right inverse)

- reduces to A^{-1} when A is square:

$$A^T(AA^T)^{-1} = A^TA^{-T}A^{-1} = A^{-1}$$

Pseudo-inverse via QR factorization

- suppose A has linearly independent columns, $A = QR$
- then $A^TA = (QR)^T(QR) = R^TQ^TQR = R^TR$
- so

$$A^\dagger = (A^TA)^{-1}A^T = (R^TR)^{-1}(QR)^T = R^{-1}R^{-T}R^TQ^T = R^{-1}Q^T$$

- can compute $A^\dagger$ using back substitution on columns of Q^T
- for A with linearly independent rows, $A^\dagger = QR^{-T}$