fit@hcmus

Vectors

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

Stephen Boyd   Lieven Vandenberghe

VECTOR

fit@hcmus

- *Vector*
- Norm and distance
- *Linear independence*
- *Applications*

Vector

Vectors

- A *vector* is an ordered list of numbers
- Written as $\begin{bmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{bmatrix}$ or $\begin{pmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{pmatrix}$

  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: $\mathbf{g}$, $\tilde{a}$
- 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:

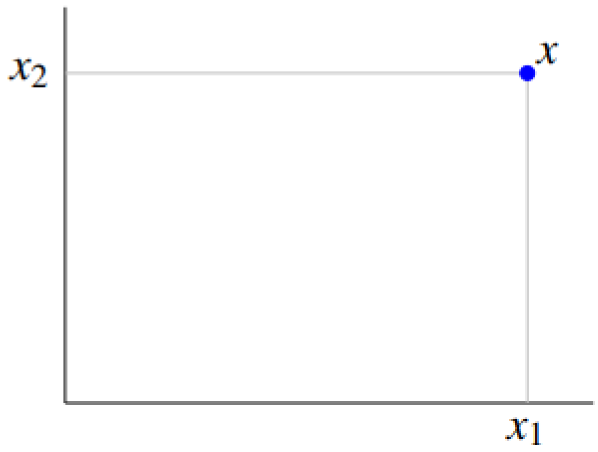
$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \qquad e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

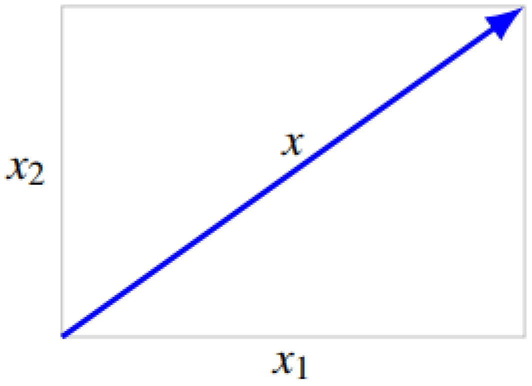
Sparsity

- 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

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)

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

- dictionaries used in practice are much larger

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

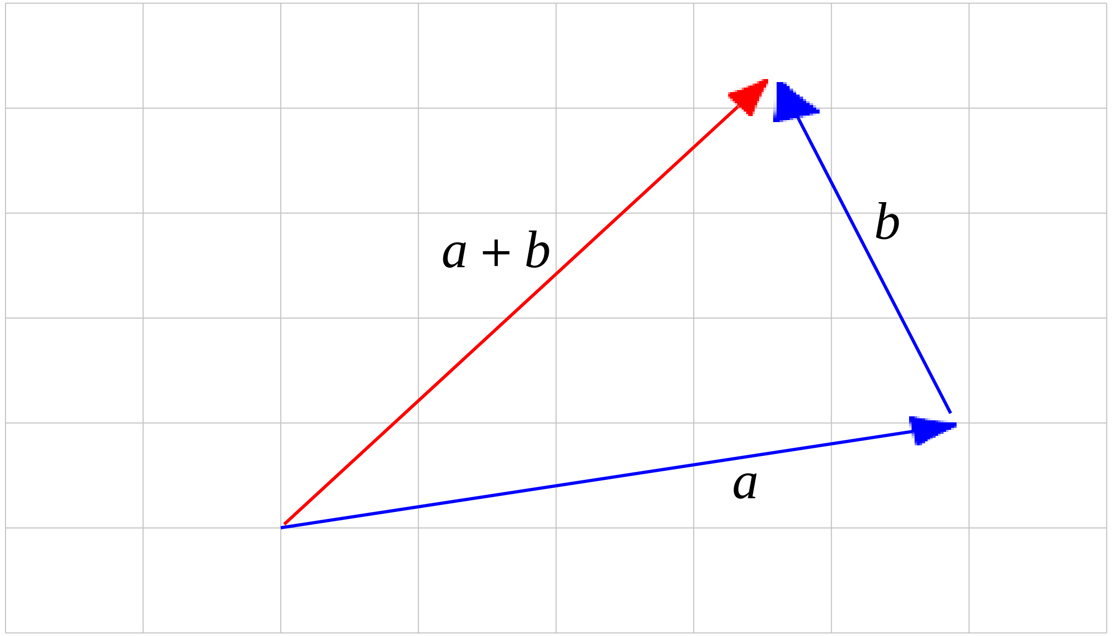
Properties of vector addition

- *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

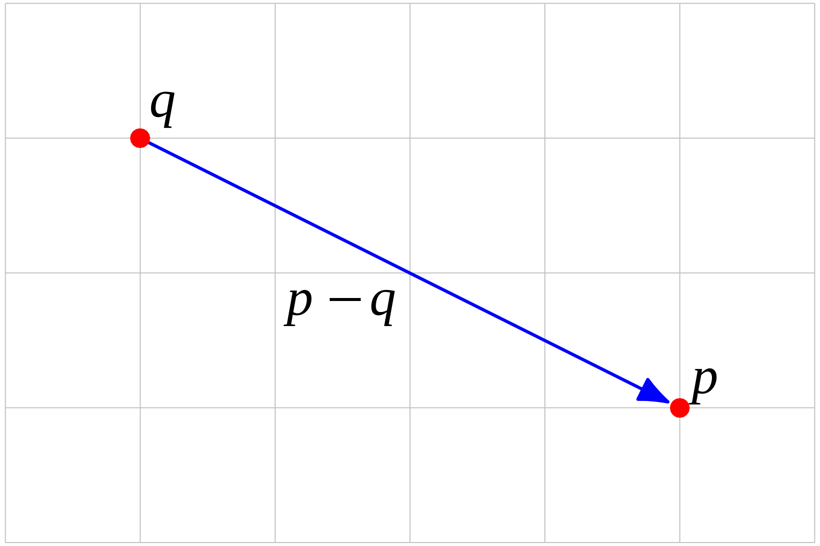
Adding displacements

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: $\beta(a + b) = \beta a + \beta b$

Linear combinations

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

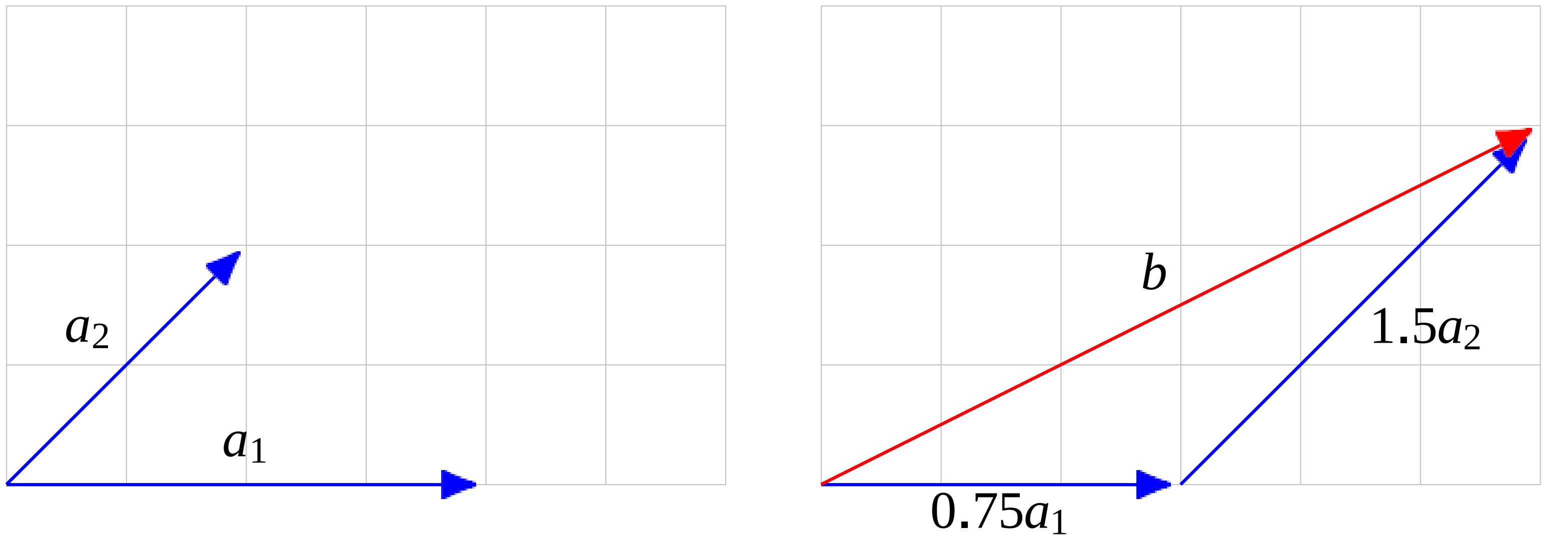
$$\beta_1 a_1 + \cdots + \beta_m a_m$$

  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

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: (a, b), $(a|b)$, (a, b), $a \cdot b$, $<a, b>$

- 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

- $a^T b = b^T a$
- $(ka)^T b = k(a^T b)$
- $(a + b)^T c = a^T c + b^T c$

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

Regression model

- *regression model* is (the affine function of x)

$$\hat{y} = x^T \beta + v$$

- x is a feature vector; its elements x_i are called *regressors*
- n-vector β is the *weight vector*
- scalar v is the *offset*
- scalar $\hat{y}$ is the *prediction*
  (of some actual outcome or *dependent variable*, denoted y)

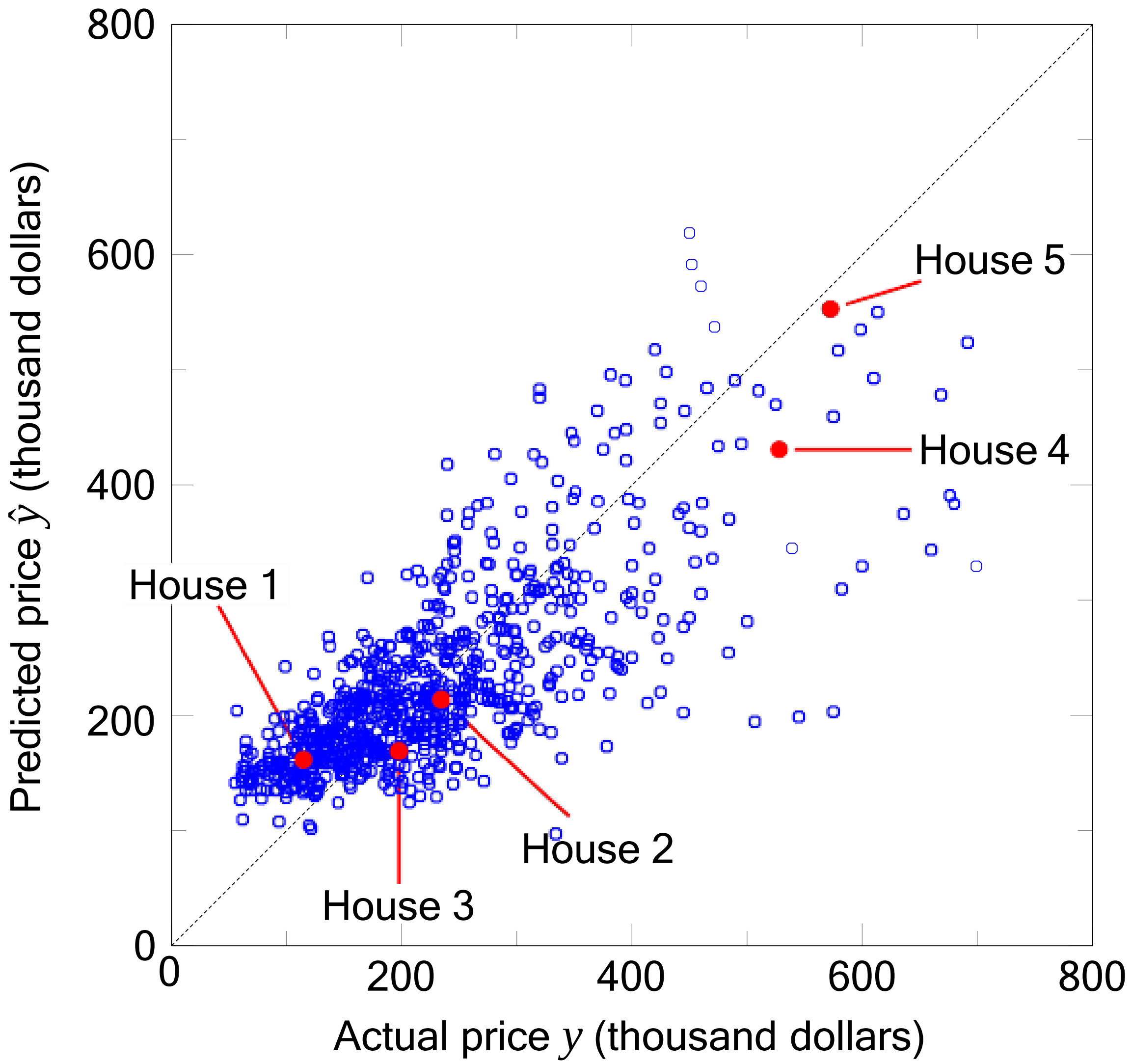
Example

- y is selling price of house in $\$1000$ (in some location, over some period)
- regressor is
$$x = (\text{house area}, \#\ \text{bedrooms})$$
(house area in 1000 sq.ft.)
- regression model weight vector and offset are
$$\beta = (148.73, -18.85), \qquad v = 54.40$$
- we'll see later how to guess β and v from sales data

Example

House	x_1 (area)	x_2 (beds)	y (price)	$\hat{y}$ (prediction)
1	0.846	1	115.00	161.37
2	1.324	2	234.50	213.61
3	1.150	3	198.00	168.88
4	3.037	4	528.00	430.67
5	3.984	5	572.50	552.66

Example



800
600
400
200
0
0
200
400
600
800
Predicted price $\hat{y}$ (thousand dollars)
Actual price y (thousand dollars)
House 1
House 2
House 3
House 4
House 5

2. Norm and distance

Outline

Norm

Distance

Standard deviation

Angle

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 $x = 0$

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

Chebyshev inequality

- suppose that k of the numbers $|x_1|, \ldots, |x_n|$ are $\geq a$
- then k of the numbers $x_1^2, \ldots, x_n^2$ are $\geq a^2$
- so $\|x\|^2 = x_1^2 + \cdots + x_n^2 \geq ka^2$
- so we have $k \leq \|x\|^2 / a^2$
- number of x_i with $|x_i| \geq a$ is no more than $\|x\|^2 / a^2$
- this is the *Chebyshev inequality*
- in terms of RMS value:

$$\text{fraction of entries with } |x_i| \geq a \text{ is no more than } \left(\frac{\mathbf{rms}(x)}{a}\right)^2$$

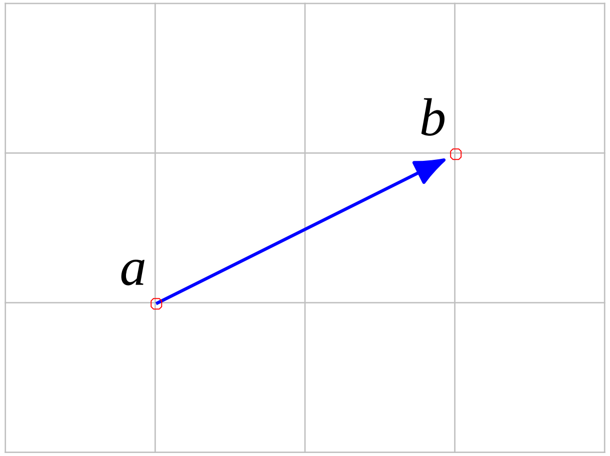
- example: no more than 4% of entries can satisfy $|x_i| \geq 5\ \mathbf{rms}(x)$

Distance

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

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

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



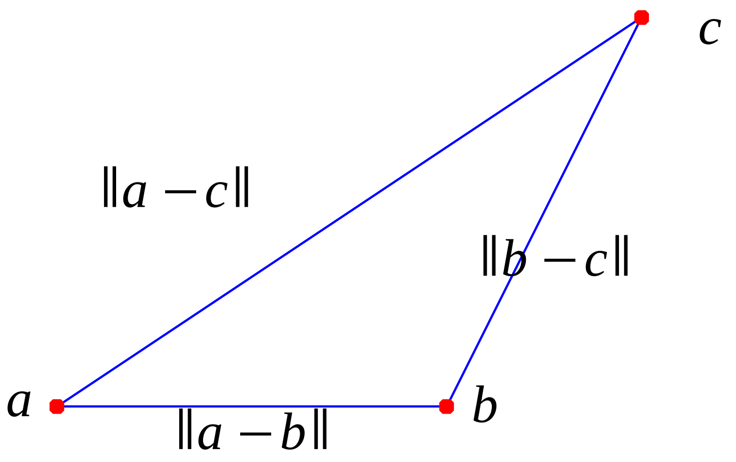
- $\textbf{rms}(a - b)$ is the *RMS deviation* between a and b

Triangle inequality

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

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

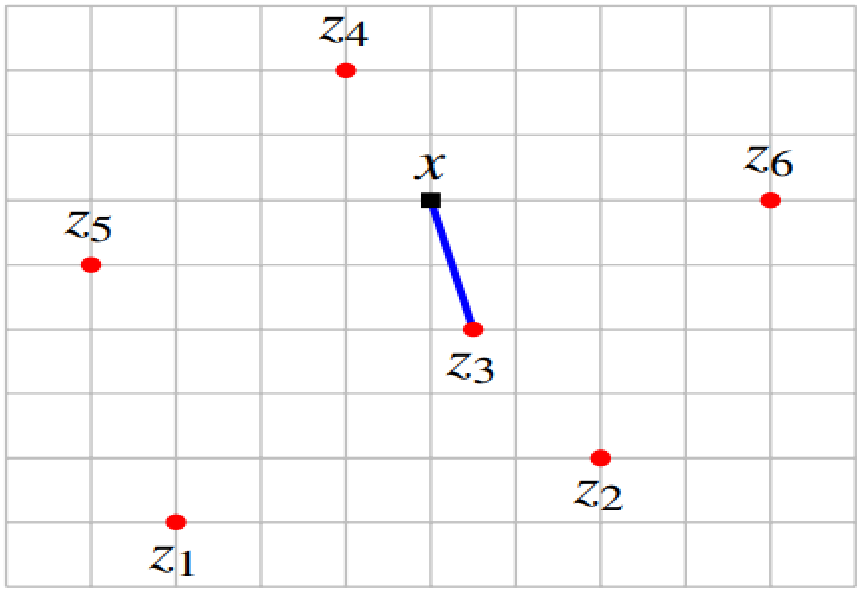
i.e., third edge length is no longer than sum of other two



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

Standard deviation

- for n-vector x, $\mathbf{avg}(x) = \mathbf{1}^T x/n$
- *de-meaned vector* is $\tilde{x} = x - \mathbf{avg}(x)\mathbf{1}$ (so $\mathbf{avg}(\tilde{x}) = 0$)
- *standard deviation* of x is

$$\mathbf{std}(x) = \mathbf{rms}(\tilde{x}) = \frac{\|x - (\mathbf{1}^T x/n)\mathbf{1}\|}{\sqrt{n}}$$

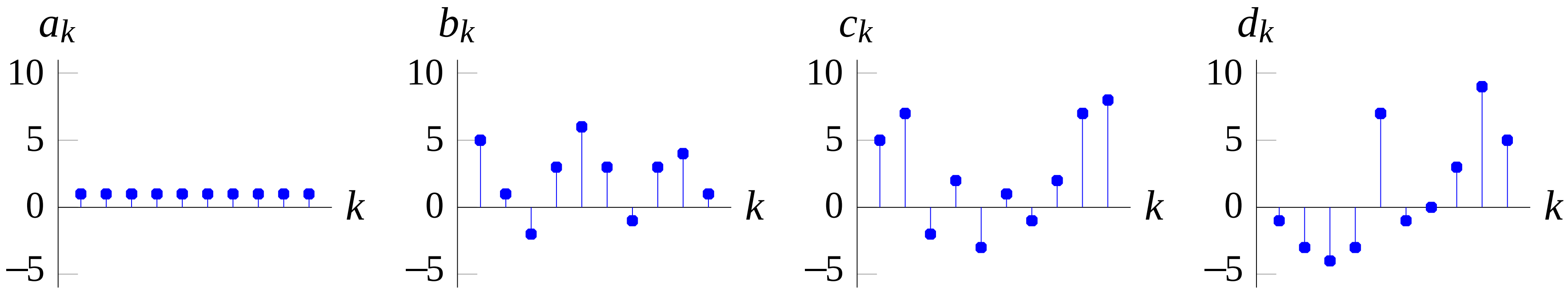
- $\mathbf{std}(x)$ gives 'typical' amount x_i vary from $\mathbf{avg}(x)$
- $\mathbf{std}(x) = 0$ only if $x = \alpha\mathbf{1}$ for some α
- greek letters μ, σ commonly used for mean, standard deviation
- a basic formula:

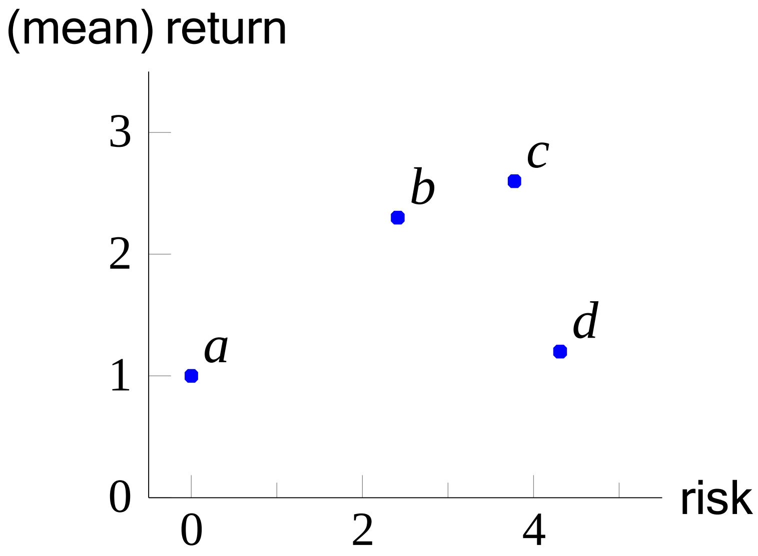
$$\mathbf{rms}(x)^2 = \mathbf{avg}(x)^2 + \mathbf{std}(x)^2$$

Mean return and risk

- x is time series of returns (say, in %) on some investment or asset over some period
- $\mathbf{avg}(x)$ is the mean return over the period, usually just called *return*
- $\mathbf{std}(x)$ measures how variable the return is over the period, and is called the *risk*
- multiple investments (with different return time series) are often compared in terms of return and risk
- often plotted on a *risk-return plot*

Risk-return example





Chebyshev inequality for standard deviation

- x is an n-vector with mean $\mathbf{avg}(x)$, standard deviation $\mathbf{std}(x)$
- rough idea: most entries of x are not too far from the mean
- by Chebyshev inequality, fraction of entries of x with

$$|x_i - \mathbf{avg}(x)| \geq \alpha \ \mathbf{std}(x)$$

  is no more than $1/\alpha^2$ (for $\alpha > 1$)
- for return time series with mean 8% and standard deviation 3%, loss ($x_i \leq 0$) can occur in no more than $(3/8)^2 = 14.1\%$ of periods

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

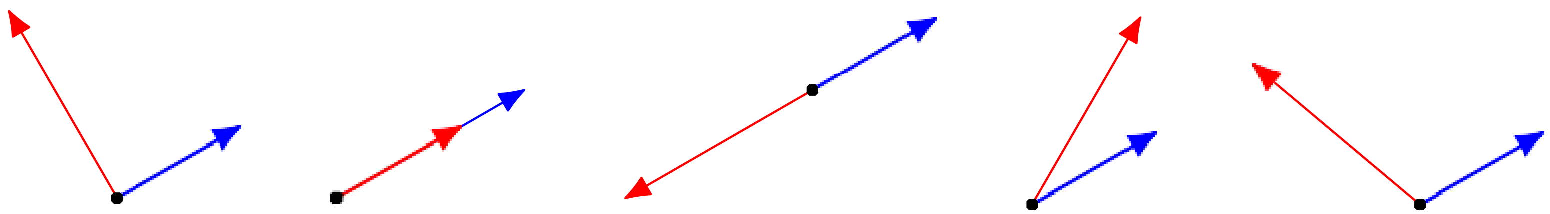
$$a^T b = \|a\|\,\|b\| \cos\left(\angle(a,b)\right)$$

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

Classification of angles

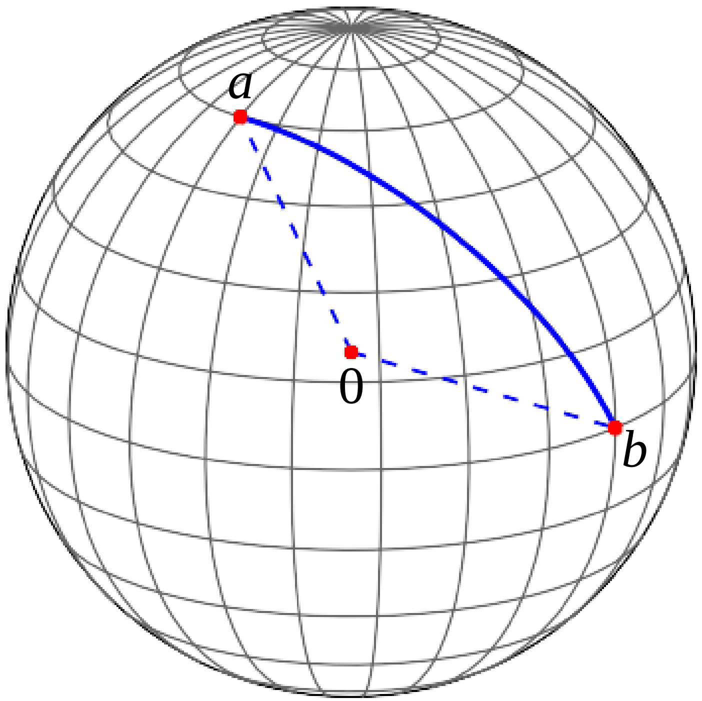
$\theta = \measuredangle(a, b)$

- $\theta = \pi/2 = 90^\circ$: 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* ($a^T b = -\|a\| \, \|b\|$)
- $\theta \le \pi/2 = 90^\circ$: a and b make an *acute angle* ($a^T b \ge 0$)
- $\theta \ge \pi/2 = 90^\circ$: a and b make an *obtuse angle* ($a^T b \le 0$)



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

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

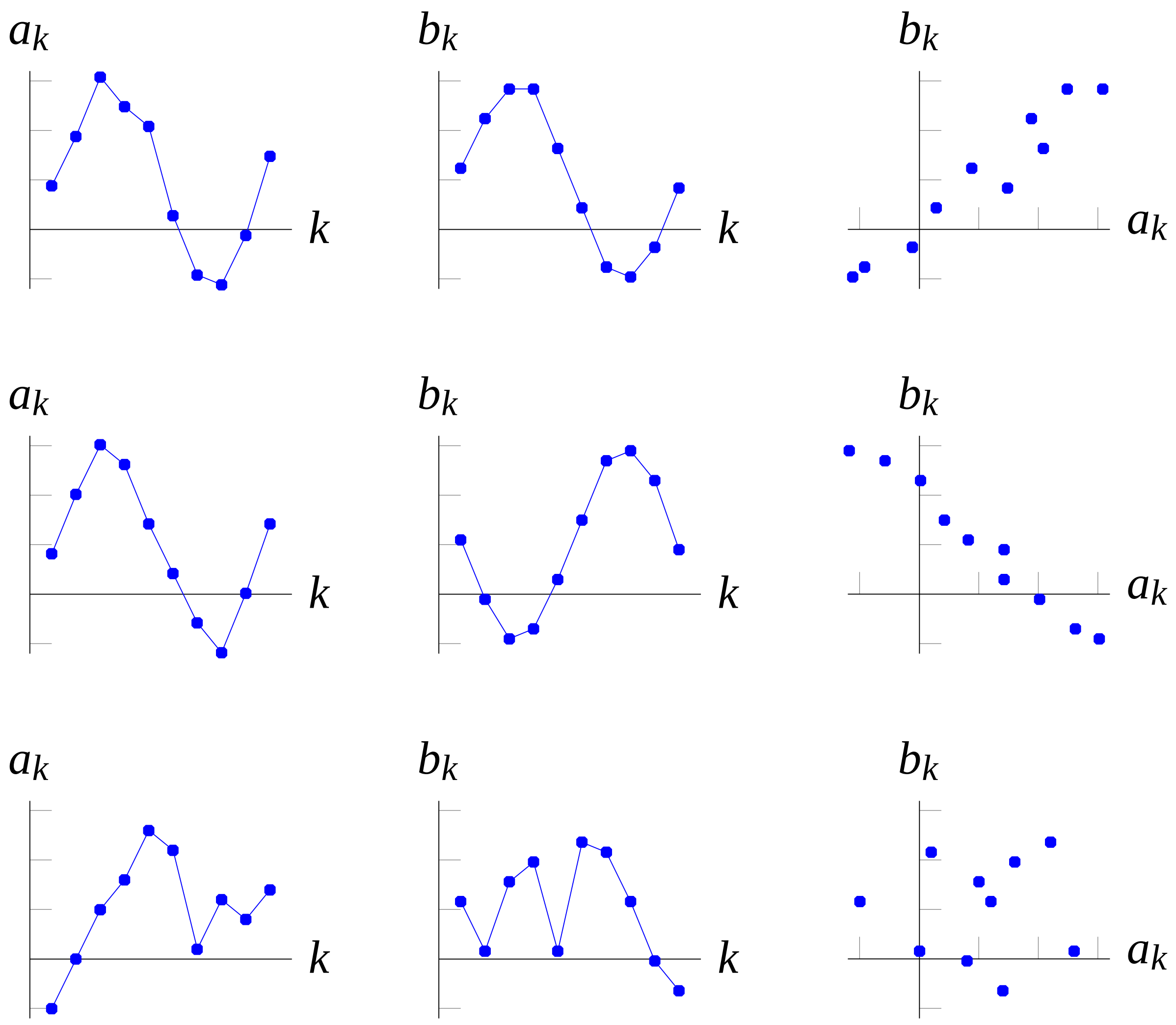
- *correlation coefficient* (between a and b, with $\tilde{a} \neq 0$, $\tilde{b} \neq 0$)

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

- $\rho = \cos \measuredangle(\tilde{a}, \tilde{b})$
  - $\rho = 0$: 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

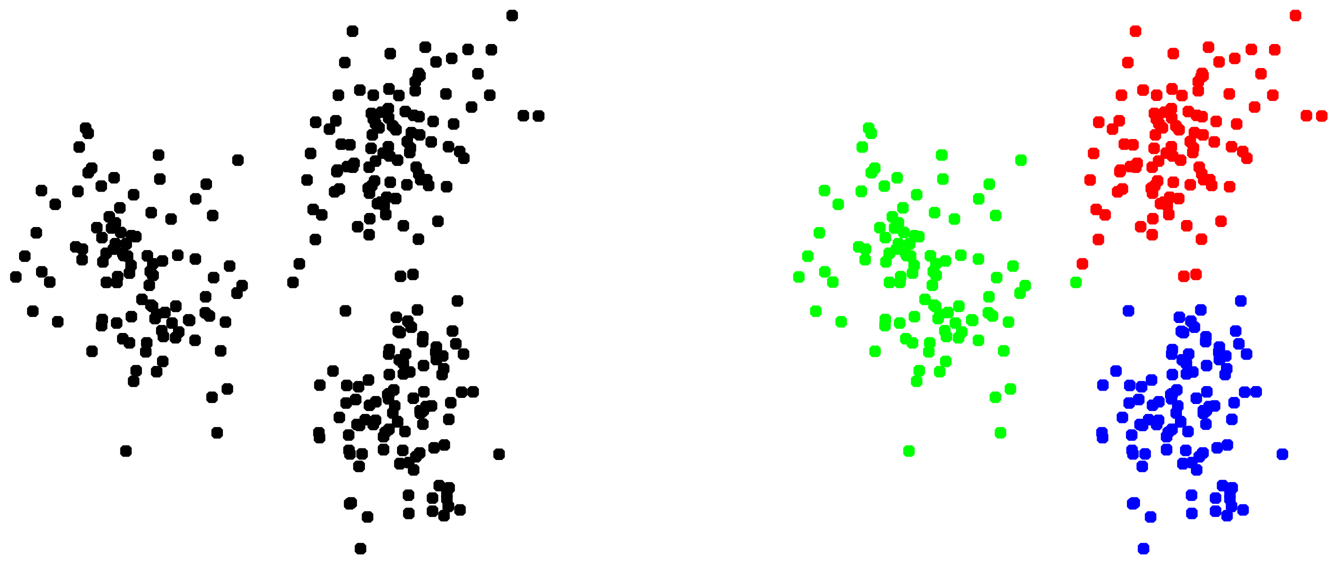
Applications

Clustering

Outline

Clustering

- given N n-vectors $x_1, \ldots, x_N$
- goal: partition (divide, cluster) into k groups
- want vectors in the same group to be close to one another



Example settings

- topic discovery and document classification
  - x_i is word count histogram for document i
- patient clustering
  - x_i are patient attributes, test results, symptoms
- customer market segmentation
  - x_i is purchase history and other attributes of customer i
- color compression of images
  - x_i are RGB pixel values
- financial sectors
  - x_i are n-vectors of financial attributes of company i

Clustering objective

- $G_j \subset \{1, \ldots, N\}$ is group j, for $j = 1, \ldots, k$
- c_i is group that x_i is in: $i \in G_{c_i}$
- group *representatives*: n-vectors $z_1, \ldots, z_k$
- clustering objective is

$$J^{\text{clust}} = \frac{1}{N} \sum_{i=1}^{N} \|x_i - z_{c_i}\|^2$$

  mean square distance from vectors to associated representative
- J^{clust} small means good clustering
- goal: choose clustering c_i and representatives z_j to minimize J^{clust}

Outline

Clustering

Algorithm

Examples

Applications

Partitioning the vectors given the representatives

- suppose representatives $z_1, \ldots, z_k$ are given
- how do we assign the vectors to groups, *i.e.*, choose $c_1, \ldots, c_N$?

- c_i only appears in term $\|x_i - z_{c_i}\|^2$ in J^{clust}
- to minimize over c_i, choose c_i so $\|x_i - z_{c_i}\|^2 = \min_j \|x_i - z_j\|^2$
- *i.e., assign each vector to its nearest representative*

Choosing representatives given the partition

- given the partition $G_1, \ldots, G_k$, how do we choose representatives $z_1, \ldots, z_k$ to minimize J^{clust}?
- J^{clust} splits into a sum of k sums, one for each z_j:

$$J^{\text{clust}} = J_1 + \cdots + J_k, \qquad J_j = (1/N) \sum_{i \in G_j} \|x_i - z_j\|^2$$

- so we choose z_j to minimize mean square distance to the points in its partition
- this is the mean (or average or centroid) of the points in the partition:

$$z_j = (1/|G_j|) \sum_{i \in G_j} x_i$$

k-means algorithm

- alternate between updating the partition, then the representatives
- a famous algorithm called *k-means*
- objective J^{clust} decreases in each step

---

given $x_1, \ldots, x_N \in \mathbf{R}^n$ and $z_1, \ldots, z_k \in \mathbf{R}^n$

repeat

Update partition: assign i to G_j, $j = \operatorname{argmin}_{j'} \|x_i - z_{j'}\|^2$

Update centroids: $z_j = \frac{1}{|G_j|} \sum_{i \in G_j} x_i$

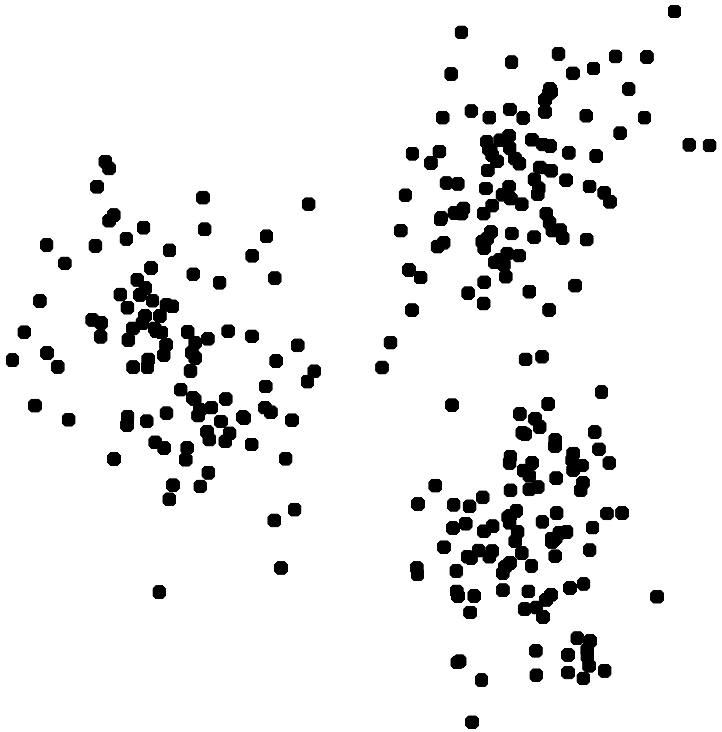
until $z_1, \ldots, z_k$ stop changing

Convergence of k-means algorithm

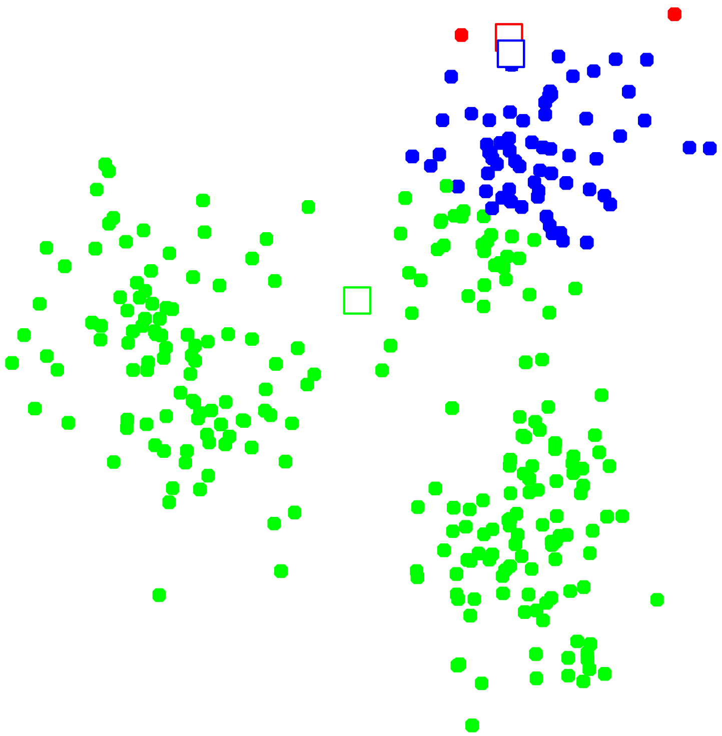
- J^{clust} goes down in each step, until the z_j's stop changing
- but (in general) the k-means algorithm *does not find the partition that minimizes* J^{clust}

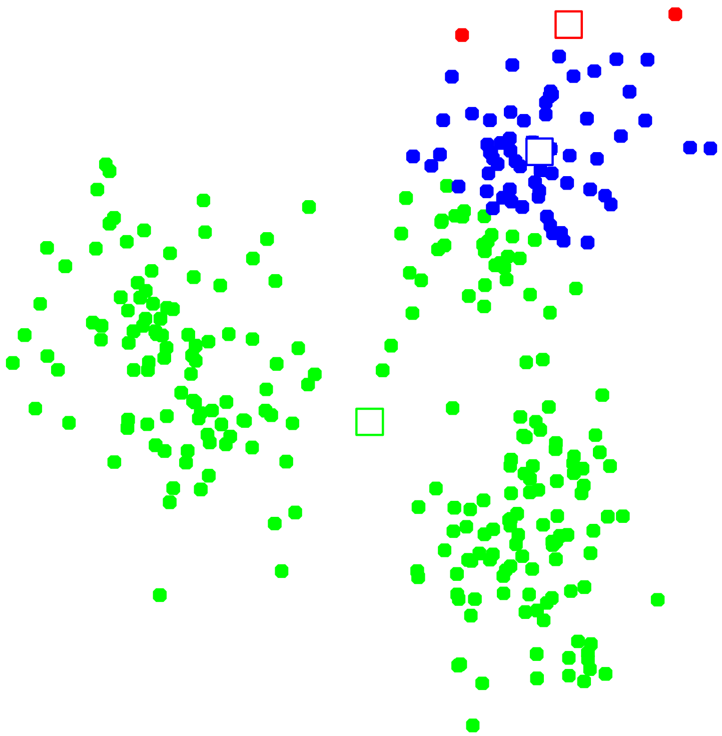
- k-means is a *heuristic*: it is not guaranteed to find the smallest possible value of J^{clust}
- the final partition (and its value of J^{clust}) can depend on the initial representatives
- common approach:
  - run k-means 10 times, with different (often random) initial representatives
  - take as final partition the one with the smallest value of J^{clust}

Data

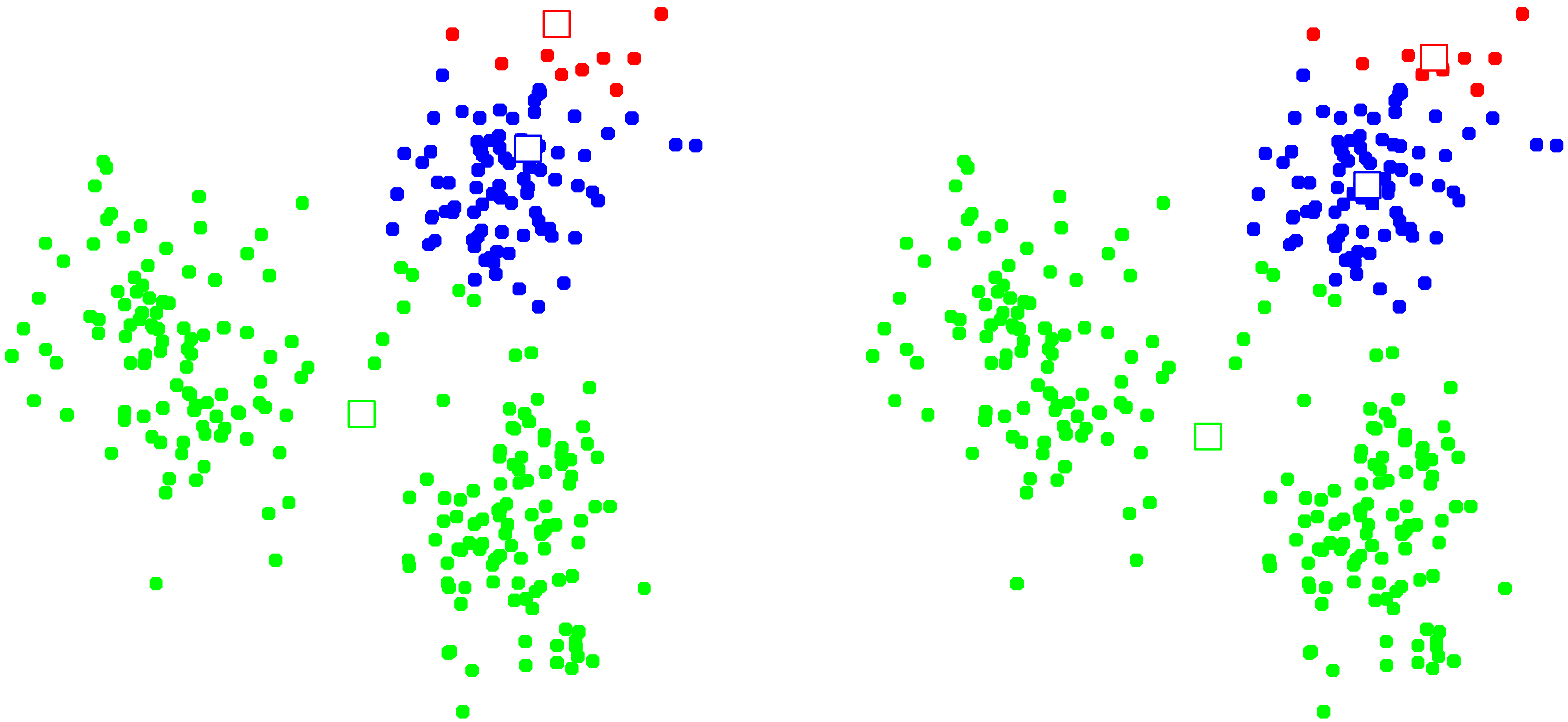


Iteration 1

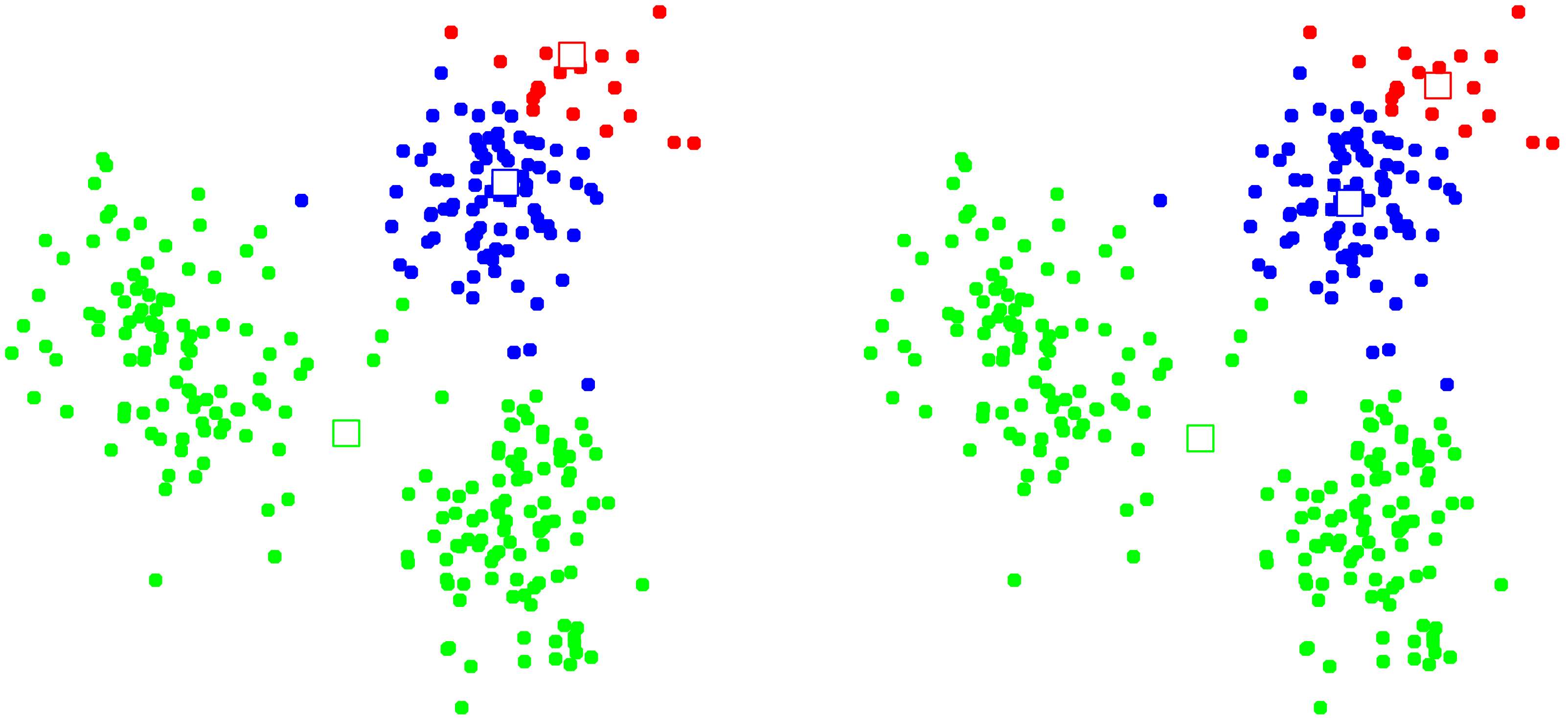




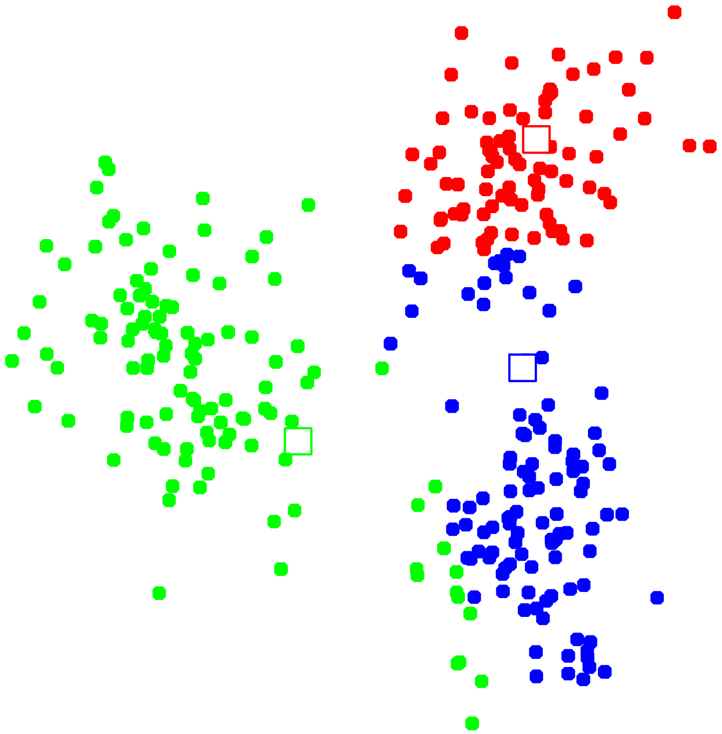
Iteration 2

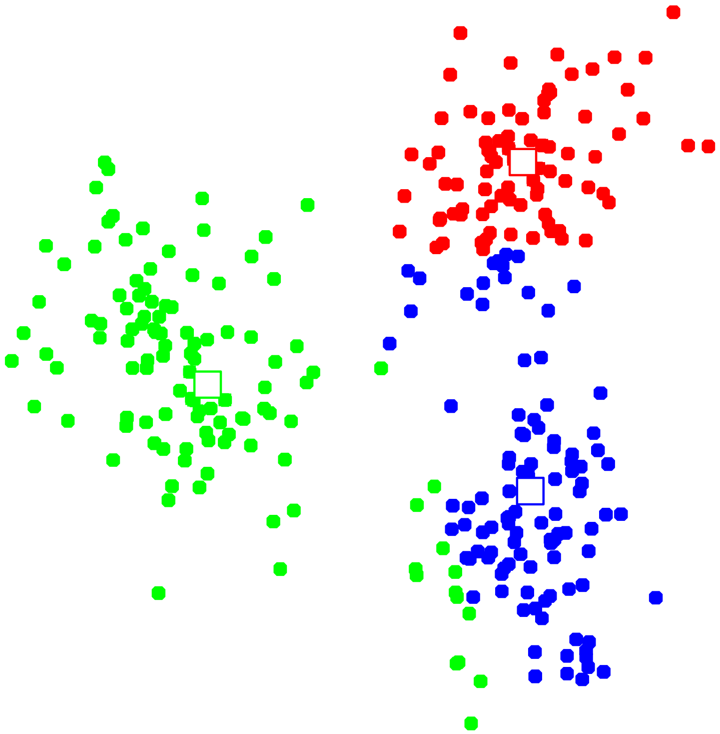


Iteration 3

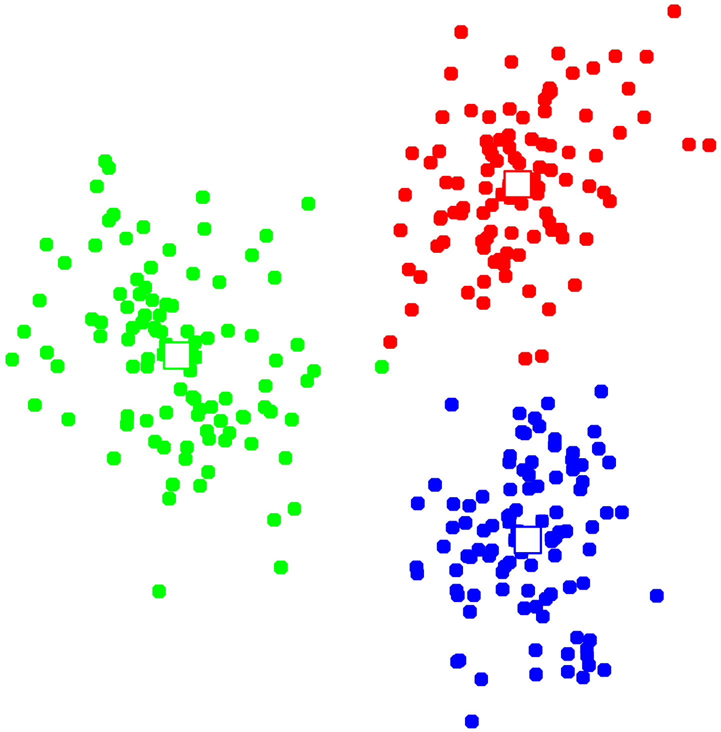


Iteration 10

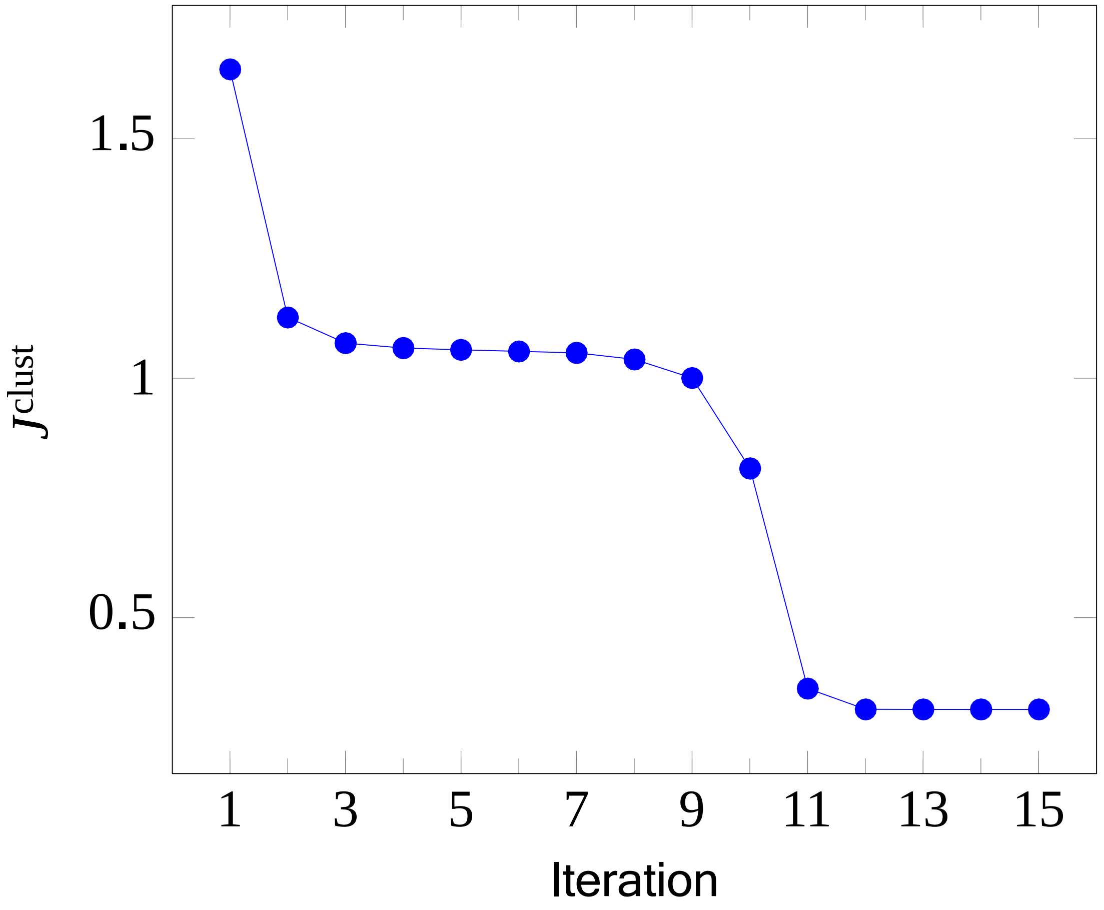




Final clustering



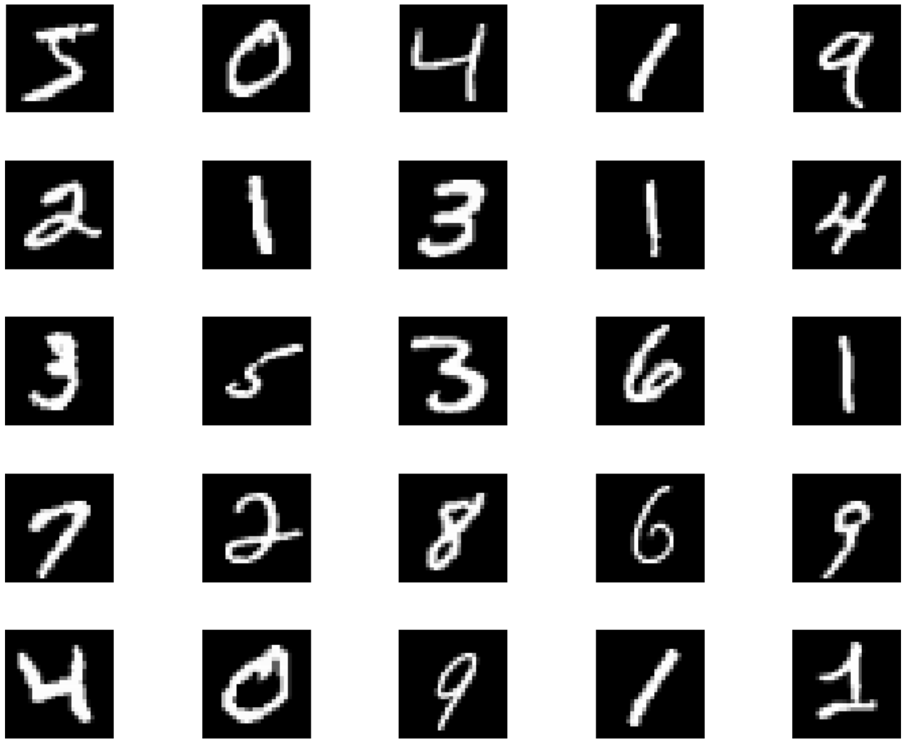
Convergence



Outline

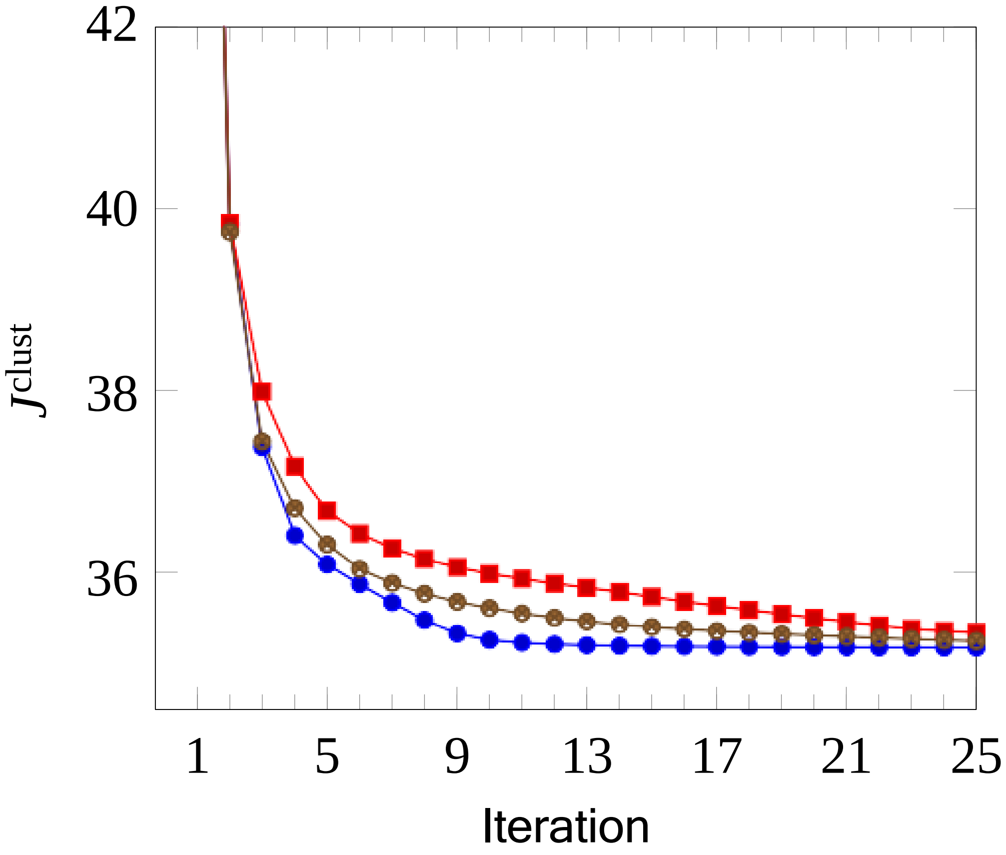
Handwritten digit image set

- MNIST images of handwritten digits (via Yann Lecun)
- $N = 60,000$ 28×28 images, represented as 784-vectors x_i
- 25 examples shown below

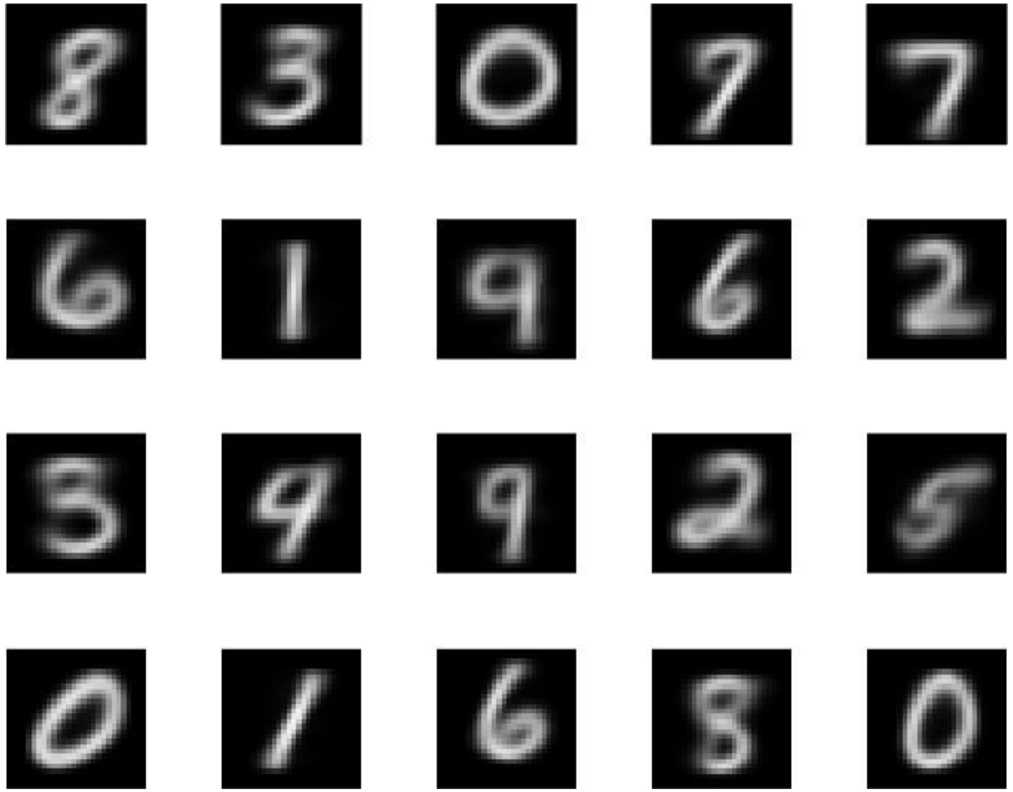


k-means image clustering

- $k = 20$, run 20 times with different initial assignments
- convergence shown below (including best and worst)

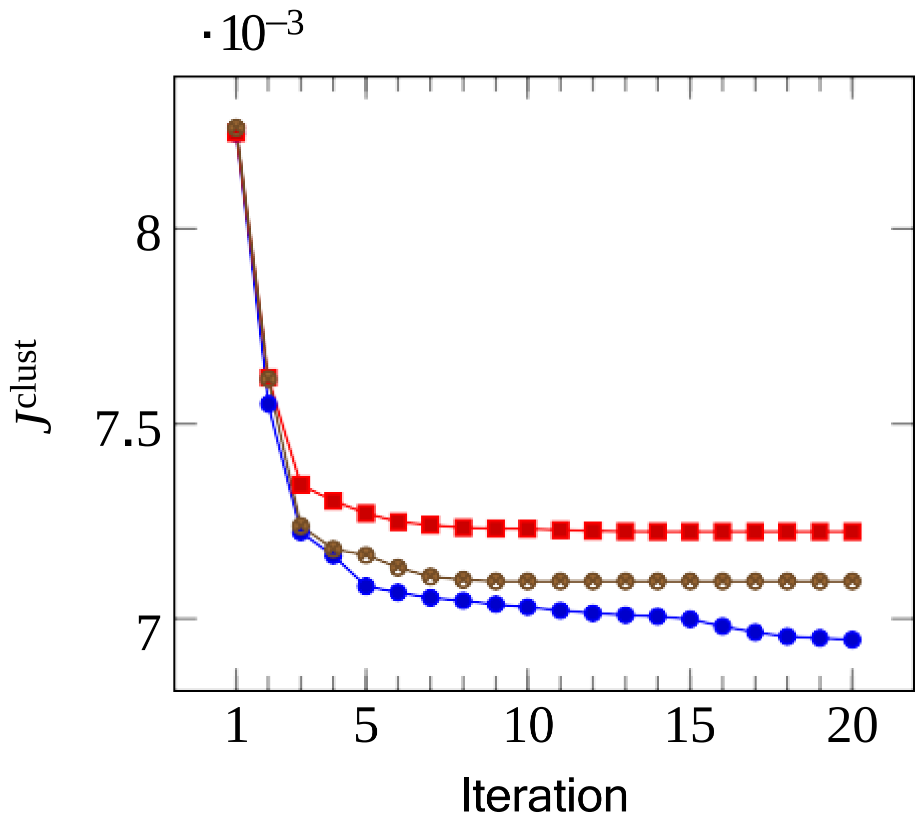


Group representatives, best clustering



Topic discovery

- $N = 500$ Wikipedia articles, word count histograms with $n = 4423$
- $k = 9$, run 20 times with different initial assignments
- convergence shown below (including best and worst)



Topics discovered (clusters 1–3)

- words with largest representative coefficients

Cluster 1		Cluster 2		Cluster 3	
Word	Coef.	Word	Coef.	Word	Coef.
fight	0.038	holiday	0.012	united	0.004
win	0.022	celebrate	0.009	family	0.003
event	0.019	festival	0.007	party	0.003
champion	0.015	celebration	0.007	president	0.003
fighter	0.015	calendar	0.006	government	0.003

- titles of articles closest to cluster representative
  1. “Floyd Mayweather, Jr”, “Kimbo Slice”, “Ronda Rousey”, “José Aldo”, “Joe Frazier”, “Wladimir Klitschko”, “Saul Álvarez”, “Gennady Golovkin”, “Nate Diaz”, . . .
  2. “Halloween”, “Guy Fawkes Night” “Diwali”, “Hanukkah”, “Groundhog Day”, “Rosh Hashanah”, “Yom Kippur”, “Seventh-day Adventist Church”, “Remembrance Day”, . . .
  3. “Mahatma Gandhi”, “Sigmund Freud”, “Carly Fiorina”, “Frederick Douglass”, “Marco Rubio”, “Christopher Columbus”, “Fidel Castro”, “Jim Webb”, . . .

Topics discovered (clusters 4–6)

- words with largest representative coefficients

Cluster 4		Cluster 5		Cluster 6	
Word	Coef.	Word	Coef.	Word	Coef.
album	0.031	game	0.023	series	0.029
release	0.016	season	0.020	season	0.027
song	0.015	team	0.018	episode	0.013
music	0.014	win	0.017	character	0.011
single	0.011	player	0.014	film	0.008

- titles of articles closest to cluster representative
  4. “David Bowie”, “Kanye West” “Celine Dion”, “Kesha”, “Ariana Grande”, “Adele”, “Gwen Stefani”, “Anti (album)”, “Dolly Parton”, “Sia Furler”, . . .
  5. “Kobe Bryant”, “Lamar Odom”, “Johan Cruyff”, “Yogi Berra”, “José Mourinho”, “Halo 5: Guardians”, “Tom Brady”, “Eli Manning”, “Stephen Curry”, “Carolina Panthers”, . . .
  6. “The X-Files”, “Game of Thrones”, “House of Cards (U.S. TV series)”, “Daredevil (TV series)”, “Supergirl (U.S. TV series)”, “American Horror Story”, . . .

Topics discovered (clusters 7–9)

- words with largest representative coefficients

Cluster 7		Cluster 8		Cluster 9	
Word	Coef.	Word	Coef.	Word	Coef.
match	0.065	film	0.036	film	0.061
win	0.018	star	0.014	million	0.019
championship	0.016	role	0.014	release	0.013
team	0.015	play	0.010	star	0.010
event	0.015	series	0.009	character	0.006

- titles of articles closest to cluster representative

7. “Wrestlemania 32”, “Payback (2016)”, “Survivor Series (2015)”, “Royal Rumble (2016)”, “Night of Champions (2015)”, “Fastlane (2016)”, “Extreme Rules (2016)”, . . .
8. “Ben Affleck”, “Johnny Depp”, “Maureen O’Hara”, “Kate Beckinsale”, “Leonardo DiCaprio”, “Keanu Reeves”, “Charlie Sheen”, “Kate Winslet”, “Carrie Fisher”, . . .
9. “Star Wars: The Force Awakens”, “Star Wars Episode I: The Phantom Menace”, “The Martian (film)”, “The Revenant (2015 film)”, “The Hateful Eight”, . . .

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

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

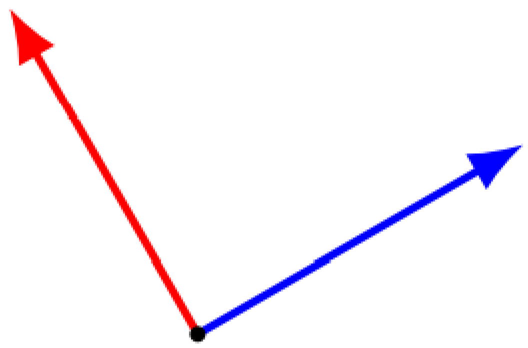
- orthonormal sets of vectors are linearly independent
- by independence-dimension inequality, must have $k \leq n$
- when $k = n$, $a_1, \dots, a_n$ are an *orthonormal basis*

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

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

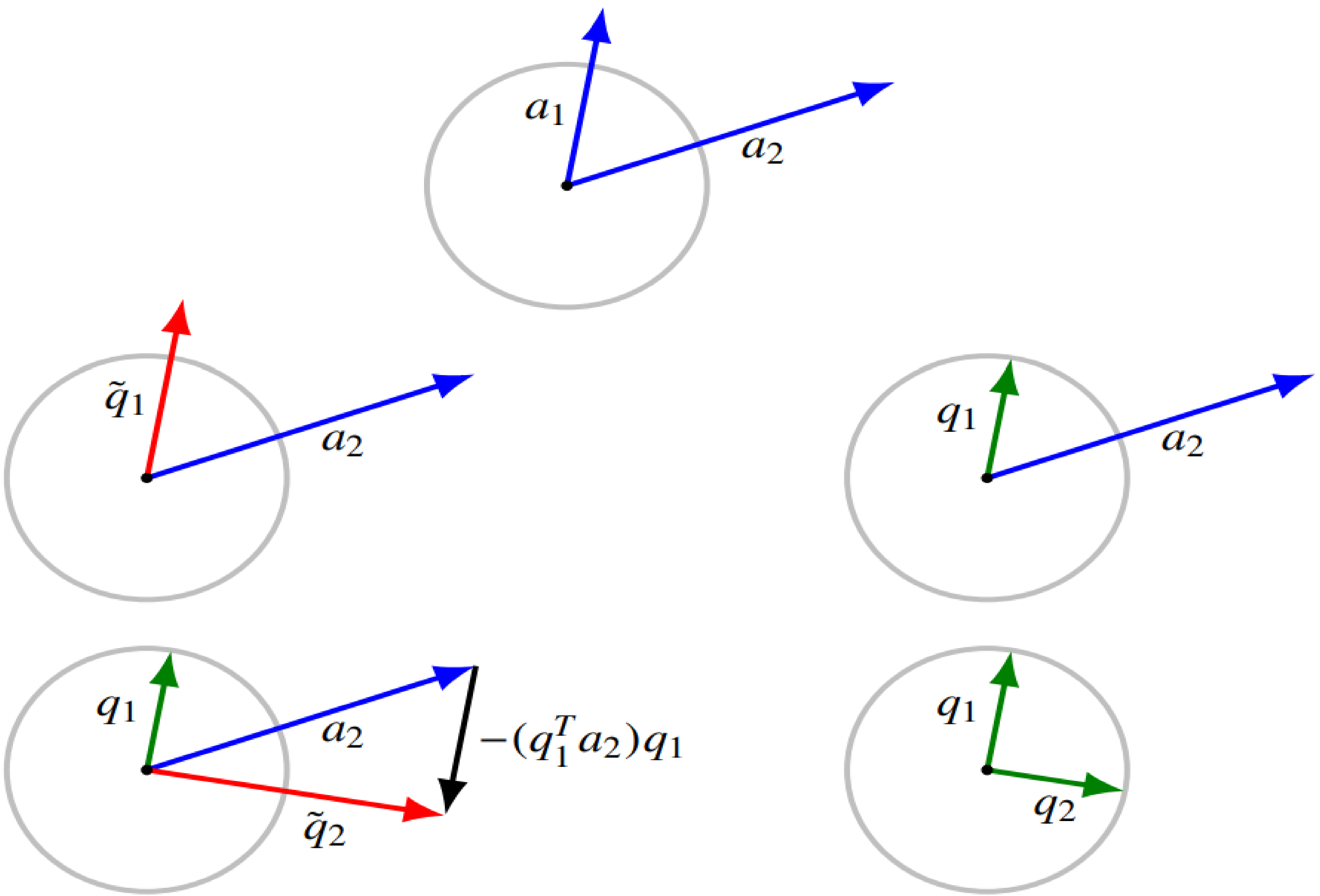
given n-vectors $a_1, \ldots, a_k$

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