COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Fall 2019.

Lecture 1

MOTIVATION FOR THIS CLASS

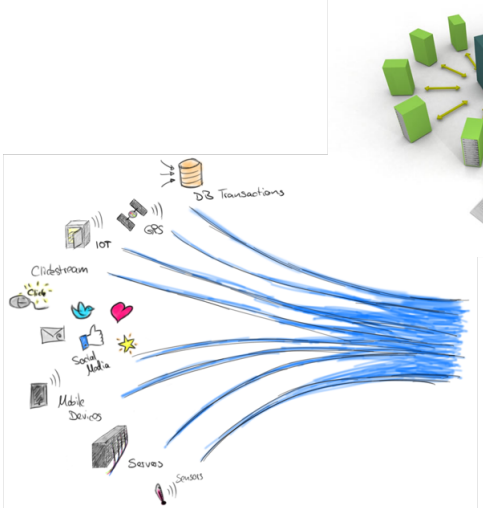
People are increasingly interested in analyzing and learning from massive datasets.

- Twitter receives 6,000 tweets per second, 500 million/day. Google receives 60,000 searches per second, 5.6 billion/day.
  - How do they process them to target advertisements? To predict trends? To improve their products?
- The Large Synoptic Survey Telescope will take high definition photographs of the sky, producing 15 terabytes of data/night.
  - How do they denoise and compress the images? How do they detect anomalies such as changing brightness or position of objects to alert researchers?

A NEW PARADIGM FOR ALGORITHM DESIGN

- Traditionally, algorithm design focuses on fast computation when data is stored in an efficiently accessible centralized manner (e.g., in RAM on a single machine).
- Massive data sets require storage in a distributed manner or processing in a continuous stream.

VS.



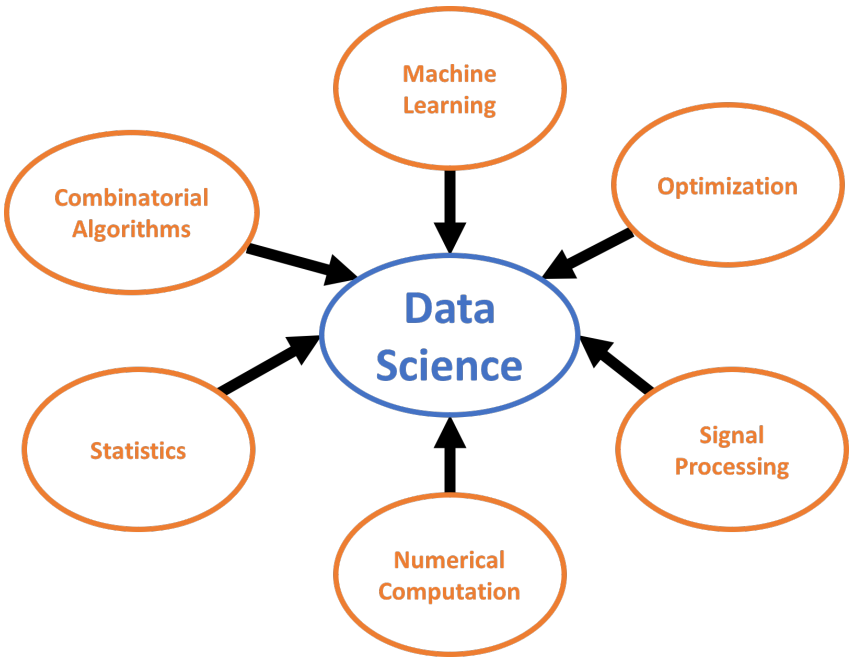
- Even ‘simple’ problems become very difficult in this setting.

For Example:

- How can Twitter rapidly detect if an incoming Tweet is an exact duplicate of another Tweet made in the last year? Given that no machine can store all Tweets made in a year.
- How can Google estimate the number of unique search queries that are made in a given week? Given that no machine can store the full list of queries.
- When you use Shazam to identify a song from a recording, how does it provide an answer in < 10 seconds, without scanning over all ~ 8 million audio files in its database.

MOTIVATION FOR THIS CLASS

A Second Motivation: Data Science is highly interdisciplinary.



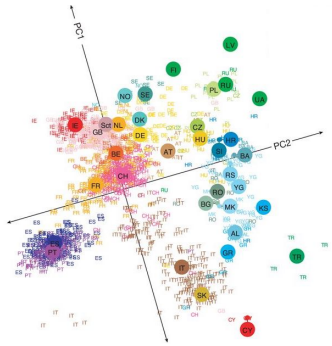
- Many techniques that aren't covered in the traditional CS algorithms curriculum.

Section 1: Randomized Methods & Sketching

How can we efficiently compress large data sets in a way that let's us answer important algorithmic questions rapidly?

- Probability tools and concentration inequalities.
- Randomized hashing for efficient lookup, load balancing, and estimation. Bloom filters.
- Locality sensitive hashing and nearest neighbor search.
- Streaming algorithms: identifying frequent items in a data stream, counting distinct items, etc.
- Random compression of high-dimensional vectors: the Johnson-Lindenstrauss lemma and its applications.
- Randomly sampling datasets: importance sampling and coresets.

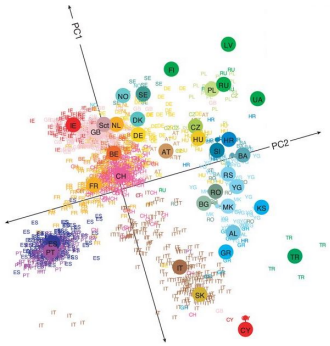
Section 2: Spectral Methods



How do we identify the most important directions and features in a dataset using linear algebraic techniques?

- Principal component analysis, low-rank approximation, dimensionality reduction.
- The singular value decomposition (SVD) and its applications to PCA, low-rank approximation, LSI, MDS, ...
- Spectral graph theory. Spectral clustering, community detection, network visualization.
- Computing the SVD on large datasets via iterative methods.

Section 2: Spectral Methods

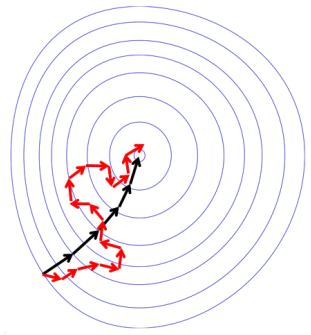


How do we identify the most important directions and features in a dataset using linear algebraic techniques?

If you open up the codes that are underneath [most data science applications] this is all linear algebra on arrays.

– Michael Stonebraker

Section 3: Optimization

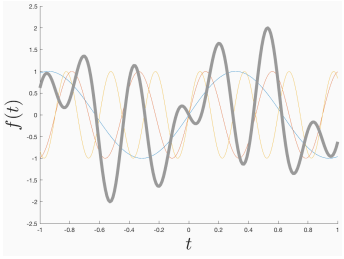


Fundamental continuous optimization approaches that drive methods in machine learning and statistics.

- Gradient descent. Analysis for convex functions.
- Stochastic and online gradient descent. Application to neural networks, non-convex analysis.
- Optimization for hard problems: alternating minimization and the EM algorithm. k-means clustering.

A small taste of what you can find in COMPSCI 590OP.

Section 4: Assorted Topics



- Compressed sensing, restricted isometry property, basis pursuit.
- Discrete Fourier transform, fast Fourier transform.
- High-dimensional geometry, isoperimetric inequality.
- Differential privacy, algorithmic fairness.

Some flexibility here. Let me know what you are interested in!

IMPORTANT TOPICS WE WON'T COVER

- Systems/Software Tools.

  - COMPSCI 532: Systems for Data Science
- Machine Learning/Data Analysis Methods and Models.
  - E.g., least squares regression, logistic regression, kernel methods, random forests, SVM, deep neural networks.
  - COMPSCI 589: Machine Learning

This is a theory centered course.

- Idea is to build general tools and algorithmic strategies that can be applied to a wide range of specific problems.
- Assignments will emphasize algorithm design, correctness proofs, and asymptotic analysis.
- A strong background in algorithms and a strong mathematical background (particularly in linear algebra and probability) are required.
- UMass prereqs: COMPSCI 240 and COMPSCI 311.

For example: Baye's rule in conditional probability. What it means for a vector x to be an eigenvector of a matrix A. Greedy algorithms, divide-and-conquer algorithms.

COURSE LOGISTICS

See course webpage for logistics, policies, lecture notes, assignments, etc.:

http://people.cs.umass.edu/~cmusco/CS514F19/

PERSONNEL

Professor: Cameron Musco

- Email: cmusco@cs.umass.edu
- Office Hours: Tuesdays, 11:30am-12:30pm, CS 234.

TAs:

- Raj Kumar Maity
- Xi Chen
- Pratheba Selvaraju

See website for office hours/contact info.

PIAZZA

We will use Piazza for class discussion and questions.

- See website for link to sign up.
- We encourage good question asking and answering with up to 5% extra credit.

HOMEWORK

We will have 4 problem sets, completed in **groups of 3**.

- Groups will remain fixed for the full semester. After you pick a group, have one member email me the members/group name by **next Thursday 9/12**.
- See Piazza for a thread to help you organize groups.

Problem set submissions will be via Gradescope.

- See website for a link to join. **Entry Code: MRVWB2**
- Since your emails, names, and grades will be stored in Gradescope we need your consent to use. See Piazza for a poll to give consent. Please complete by **next Thursday 9/12**.

Grade Breakdown:

- Problem Sets (4 total): 40%, weighted equally.
- In Class Midterm (10/17): 30%.
- Final (12/19, 10:30am-12:30pm): 30%.

Extra Credit: Up to 5% extra credit will be awarded for participation. Asking good clarifying questions in class and on Piazza, answering instructors questions in class, answering other students' questions on Piazza, etc.

DISABILITIES

UMass Amherst is committed to making reasonable, effective, and appropriate accommodations to meet the needs to students with disabilities.

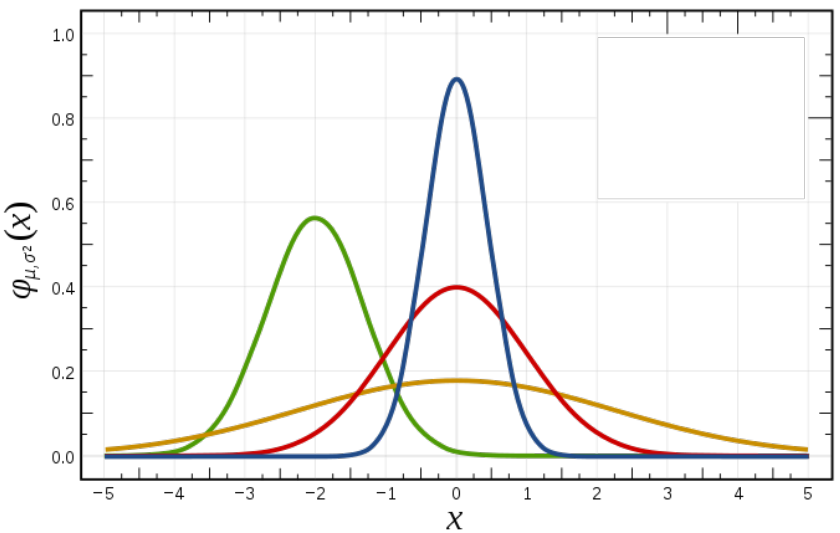
- If you have a documented disability **on file with Disability Services**, you may be eligible for reasonable accommodations in this course.
- If your disability requires an accommodation, please notify me by **next Thursday 9/12** so that we can make arrangements.

Questions?

Section 1: Randomized Methods & Sketching

Consider a random X variable taking values in some finite set $S \subset \mathbb{R}$. E.g., for a random dice roll, $S = \{1, 2, 3, 4, 5, 6\}$.

- **Expectation:** $\mathbb{E}[X] = \sum_{s \in S} \Pr(X = s) \cdot s.$
- **Variance:** $\mathrm{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2].$



Consider two random events A and B.

- **Conditional Probability:**

$$\Pr(A|B) = \frac{\Pr(A \cap B)}{\Pr(B)}.$$

- **Independence:** A and B are independent if:

$$\Pr(A|B) = \Pr(A).$$

Using the definition of conditional probability, independence means:

$$\frac{\Pr(A \cap B)}{\Pr(B)} = \Pr(A) \implies \Pr(A \cap B) = \Pr(A) \cdot \Pr(B).$$

For Example: What is the probability that for two independent dice rolls the first is a 6 and the second is odd?

$$\begin{aligned}\Pr(D_1 = 6 \cap D_2 \in \{1,3,5\}) &= \Pr(D_1 = 6) \cdot \Pr(D_2 \in \{1,3,5\}) \\ &= \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{12}\end{aligned}$$

Independent Random Variables: Two random variables X, Y are independent if for all s, t, $X = s$ and $Y = t$ are independent events. In other words:

$$\Pr(X = s \cap Y = t) = \Pr(X = s) \cdot \Pr(Y = t).$$

LINEARITY OF EXPECTATION AND VARIANCE

When are the expectation and variance linear? I.e.,

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$$

and

$$\mathrm{Var}[X + Y] = \mathrm{Var}[X] + \mathrm{Var}[Y].$$

$\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ for any random variables X and Y.

Proof:

$$\begin{aligned}
\mathbb{E}[X+Y] &= \sum_{s\in S}\sum_{t\in T} \Pr(X=s \cap Y=t)\cdot(s+t) \\
&= \sum_{s\in S}\sum_{t\in T} \Pr(X=s \cap Y=t)\cdot s + \sum_{s\in S}\sum_{t\in T} \Pr(X=s \cap Y=t)\cdot t \\
&= \sum_{s\in S} s\cdot\sum_{t\in T} \Pr(X=s \cap Y=t) + \sum_{t\in T} t\cdot\sum_{s\in S} \Pr(X=s \cap Y=t) \\
&= \sum_{s\in S} s\cdot \Pr(X=s) + \sum_{t\in T} t\cdot \Pr(Y=t) = \mathbb{E}[X] + \mathbb{E}[Y].
\end{aligned}$$

$\mathrm{Var}[X+Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$ when X and Y are independent.

Claim 1: $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$ (via linearity of expectation)

Claim 2: $\mathbb{E}[XY] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$ when X, Y are independent.

Together give:

$$\begin{aligned}
\mathrm{Var}[X+Y] &= \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2 \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2 \\
&\qquad\qquad \text{(linearity of expectation)} \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}[X] \cdot \mathbb{E}[Y] - \mathbb{E}[Y]^2 \\
&= \mathrm{Var}[X] + \mathrm{Var}[Y].
\end{aligned}$$

AN ALGORITHMIC APPLICATION

You have contracted with a new company to provide CAPTCHAS for your website.

- They claim that they have a database of 1000000 unique CAPTCHAS. A random one is chosen for each security check.
- You want to independently verify this claimed database size.
- You could make test checks until you see 1000000 unique CAPTCHAS: would take $\geq$ 1000000 checks!

A Clever Idea: You run some test security checks and see if any duplicate CAPTCHAS show up. If you're seeing duplicates after not too many checks, the database size is probably not too big.

- 'Mark and recapture' method in ecology.

If you run m security checks, and there are n unique CAPTCHAS, how many pairwise duplicates do you see in expectation?

If e.g. the same CAPTCHA shows up three times, on your i^{th}, j^{th}, and k^{th} test, this is three duplicates: (i, j), (i, k) and (j, k).

Let $D_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[D] = \sum_{i,j} \mathbb{E}[D_{i,j}].$$

For any pair i, j: $\mathbb{E}[D_{i,j}] = \Pr[D_{i,j} = 1] = \frac{1}{n}$.

$$\mathbb{E}[D] = \sum_{i,j} \frac{1}{n} = \frac{\binom{m}{2}}{n} = \frac{m(m-1)}{2n}.$$

Note that the $D_{i,j}$ random variables are not independent!

You take $m = 1000$ samples. If the database size is as claimed ($n = 1000000$) then expected number of duplicates is:

$$\mathbb{E}[D] = \frac{m(m-1)}{2n} = .4995$$

You see 10 pairwise duplicates. And suspect that something is up. But how confident can you be in your test?

Concentration Inequalities: Bounds on the probability that a random variable deviates a certain distance from its mean.

- Useful in understanding how statistical tests perform, the behavior of randomized algorithms, the behavior of data drawn from different distributions, etc.

The most fundamental concentration bound: **Markov's inequality.**

For any non-negative random variable X:

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

Proof:

$$\begin{aligned}\mathbb{E}[X] = \sum_s \Pr(X = s) \cdot s &\geq \sum_{s \geq t} \Pr(X = s) \cdot s \\ &\geq \sum_{s \geq t} \Pr(X = s) \cdot t \\ &= t \cdot \Pr(X \geq t).\end{aligned}$$

The most fundamental concentration bound: **Markov's inequality.**

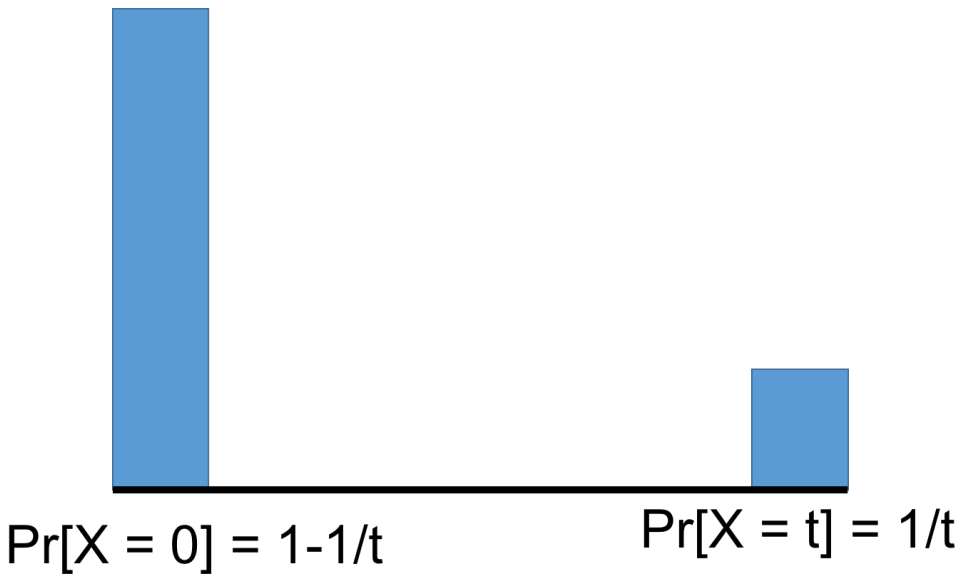
For any non-negative random variable X:

$$\Pr[X \geq t \cdot \mathbb{E}[X]] \leq \frac{1}{t}.$$

Proof:

$$\begin{aligned}\mathbb{E}[X] = \sum_s \Pr(X = s) \cdot s &\geq \sum_{s \geq t} \Pr(X = s) \cdot s \\ &\geq \sum_{s \geq t} \Pr(X = s) \cdot t \\ &= t \cdot \Pr(X \geq t).\end{aligned}$$

Given no other assumptions on X besides non-negativity, can you prove a stronger bound than Markov's? No!



$$\Pr[X \geq t] = \frac{\mathbb{E}[X]}{t}.$$

Expected number of duplicate CAPTCHAS:

$\mathbb{E}[D] = \frac{m(m-1)}{2n} = .4995.$

You see $D = 10$.

Applying Markov's inequality, if the real database size is $n = 1000000$ the probability of this happening is:

$$\Pr[D \geq 10] \leq \frac{\mathbb{E}[D]}{10} = \frac{.4995}{10} \approx .05$$

This is pretty small – you feel pretty sure the number of unique CAPTCHAS is much less than 1000000. But how can you boost your confidence?We'll discuss next class.