COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco
University of Massachusetts Amherst. Spring 2020.
Lecture 13

- **Midterm is on Thursday.**
- No calculators, cheatsheets, or other aids.
- Very important to do some practice problems and to try them first with no resources, to simulate the exam.
- Make sure you can recognize when to apply the fundamentals: union bound, linearity of expectation and variance, Markov's inequality, Chebyshev's inequality, indicator random variables.
- Understand the goal of each algorithm/data structure. I.e., what problem it solves with what guarantees. No need to memorize proofs.

Random hashing / hash tables

→ formally analyse hash table using probability tools

→ Markovs inequality, linearity of exp, [2-universal + pairwise hashing]

Optimizing hash tables

2 level hashing + bloom filters

— more practice using random hash functions

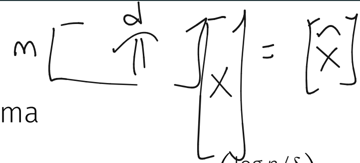
Locality sensitive hashing: approximate queries

- want collisions , SimHash , MinHash
- length-r signatures + + hash tables, S-curve.
- why does minHash.

Streaming algos → Distinct Elements (MinHash) (ε, δ)
Frequent Elements (Count-min sketch)

Last Few Classes:

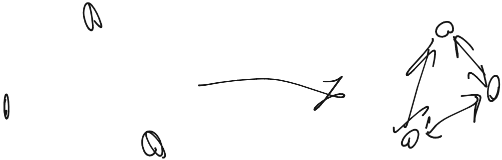
- given ϵ, δ set some other parameter to achieve these bounds
- prove some prob. bound $\Pr(|\hat{S} - \mathbb{E}S| \geq t) \leq \ldots$
- what are the requirements for each inequality
  - median trick
  - law of large numbers
  - distinct elements
- Markov's: $\mathbb{E}[X]$, $X > 0$
- Cheby: $Var(X)$, $\mathbb{E}[X]$
- Bern's: sum of ind. r.v.s., $\mathbb{E}$, Var, $[-M, M]$
- Chernoff: ind. r.v.s, r.v.s are binary.



Last Few Classes:

The Johnson-Lindenstrauss Lemma

- Reduce n data points in any dimension d to $O\left(\frac{\log n/\delta}{\epsilon^2}\right)$ dimensions and preserve (with probability $\geq 1-\delta$) all pairwise distances up to $1 \pm \epsilon$.
- Compression is linear via multiplication with a random, data oblivious, matrix (linear compression)



Last Few Classes:

The Johnson-Lindenstrauss Lemma

- Reduce n data points in any dimension d to $O\left(\frac{\log n/\delta}{\epsilon^2}\right)$ dimensions and preserve (with probability $\geq 1-\delta$) all pairwise distances up to $1 \pm \epsilon$.
- Compression is linear via multiplication with a random, data oblivious, matrix (linear compression)

High-Dimensional Geometry

- Why high-dimensional space is so different than low-dimensional space.
- How the JL Lemma can still work.

Next Few Classes: Low-rank approximation, the SVD, and principal component analysis (PCA).

- Reduce d-dimesional data points to a smaller dimension m.
- Like JL, compression is linear – by applying a matrix.
- Chose this matrix carefully, taking into account structure of the dataset.
- Can give better compression than random projection.

Next Few Classes: Low-rank approximation, the SVD, and principal component analysis (PCA).

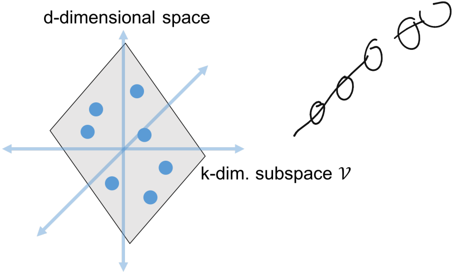
- Reduce d-dimesional data points to a smaller dimension m.
- Like JL, compression is linear – by applying a matrix.
- Chose this matrix carefully, taking into account structure of the dataset.
- Can give better compression than random projection.

Will be using a fair amount of linear algebra: orthogonal basis, column/row span, eigenvectors, etc,

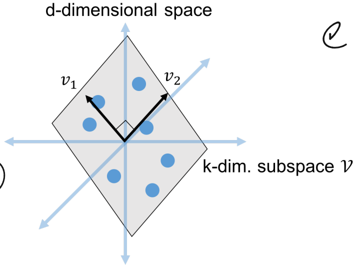
RANDOMIZED ALGORITHMS UNIT TAKEAWAYS

- Randomization is an important tool in working with large datasets.
- Lets us solve 'easy' problems that get really difficult on massive datasets. Fast/space efficient look up (hash tables and bloom filters), distinct items counting, frequent items counting, near neighbor search, etc.
- The analysis of randomized algorithms leads to complex output distributions, which we can't compute exactly.
- We use concentration inequalities to bound these distributions and behaviors like accuracy, space usage, and runtime.
- Concentration inequalities and probability tools used in randomized algorithms are also fundamental in statistics, machine learning theory, probabilistic modeling of complex systems, etc.

Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



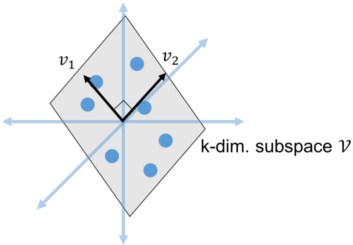
Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Claim: Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns. For all $\vec{x}_i, \vec{x}_j$:

$$\|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



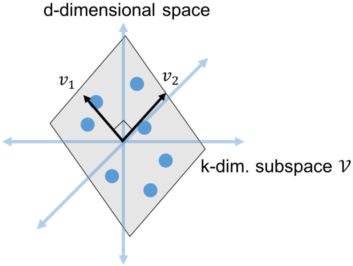
Claim: Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns. For all $\vec{x}_i, \vec{x}_j$:

$$\|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

- $\mathbf{V}^T \in \mathbb{R}^{k \times d}$ is a linear embedding of $\vec{x}_1, \ldots, \vec{x}_n$ into k dimensions with no distortion.

EMBEDDING WITH ASSUMPTIONS

Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie in any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Claim: Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns. For all $\vec{x}_i, \vec{x}_j$:

$$\|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2 = \|\vec{x}_i - \vec{x}_j\|_2.$$

- $\mathbf{V}^T \in \mathbb{R}^{k \times d}$ is a linear embedding of $\vec{x}_1, \ldots, \vec{x}_n$ into k dimensions with no distortion.
- An actual projection, analogous to a JL random projection $\mathbf{\Pi}$.

DOT PRODUCT TRANSFORMATION

Claim: Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns. For all $\vec{x}_i, \vec{x}_j \in \mathcal{V}$:

$$\|\mathbf{V}^T\vec{x}_i - \mathbf{V}^T\vec{x}_j\|_2^2 = \|\vec{x}_i - \vec{x}_j\|_2^2.$$

$I c_i = c_i$

$V V^T \neq I$

$V^T V = I$

$(V^T V)^T = I^T$

$x_i = V c_i$ for some $c_i \in \mathbb{R}^k$

$\|V^T V c_i - V^T V c_j\|_2^2$

$\|I c_i - I c_j\|_2^2$

$\|c_i - c_j\|_2^2$

$= \|V c_i - V c_j\|_2^2$

$= \|V(c_i - c_j)\|_2^2$

$= (c_i - c_j)^T V^T V (c_i - c_j)$

$= (c_i - c_j)^T (c_i - c_j)$

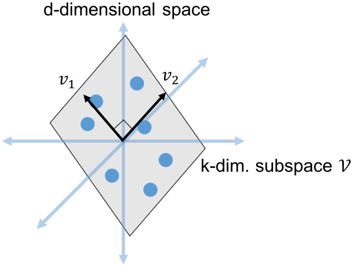
$= \|c_i - c_j\|_2^2$

$V^T V = I_{k \times k}$

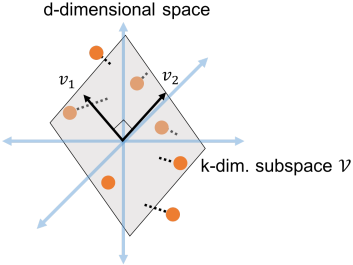
$(V^T V)_{ij} = \langle v_i, v_j \rangle$

$y^T y = \langle y, y \rangle = \|y\|_2^2$

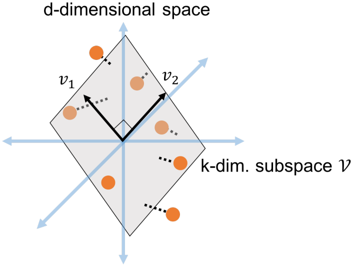
Main Focus of Today: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Main Focus of Today: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



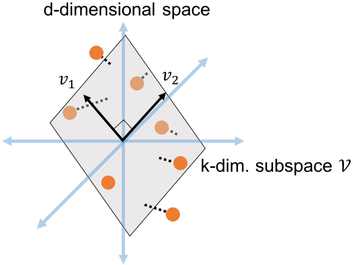
Main Focus of Today: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Letting $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns, $\mathbf{V}^T\vec{x}_i \in \mathbb{R}^k$ is still a good embedding for $x_i \in \mathbb{R}^d$.

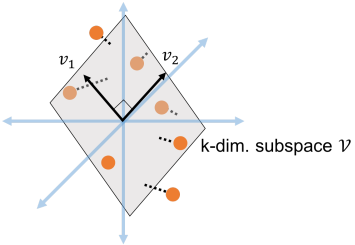
EMBEDDING WITH ASSUMPTIONS

Main Focus of Today: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Letting $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns, $\mathbf{V}^T\vec{x}_i \in \mathbb{R}^k$ is still a good embedding for $x_i \in \mathbb{R}^d$. The key idea behind low-rank approximation and principal component analysis (PCA).

Main Focus of Today: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$.



Letting $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns, $\mathbf{V}^T \vec{x}_i \in \mathbb{R}^k$ is still a good embedding for $x_i \in \mathbb{R}^d$. The key idea behind low-rank approximation and principal component analysis (PCA).

- How do we find $\mathcal{V}$ and $\mathbf{V}$?
- How good is the embedding?