CS-GY 6763: Lecture 11
Linear Programming, Singular Value Decomposition

NYU Tandon School of Engineering, Prof. Christopher Musco

DIMENSION DEPENDENT CONVEX OPTIMIZIATION

Consider a convex function $f(\mathbf{x})$ be bounded between $[-B, B]$ on a constraint set $\mathcal{S}$.

Theorem (Dimension Dependent Convex Optimization)

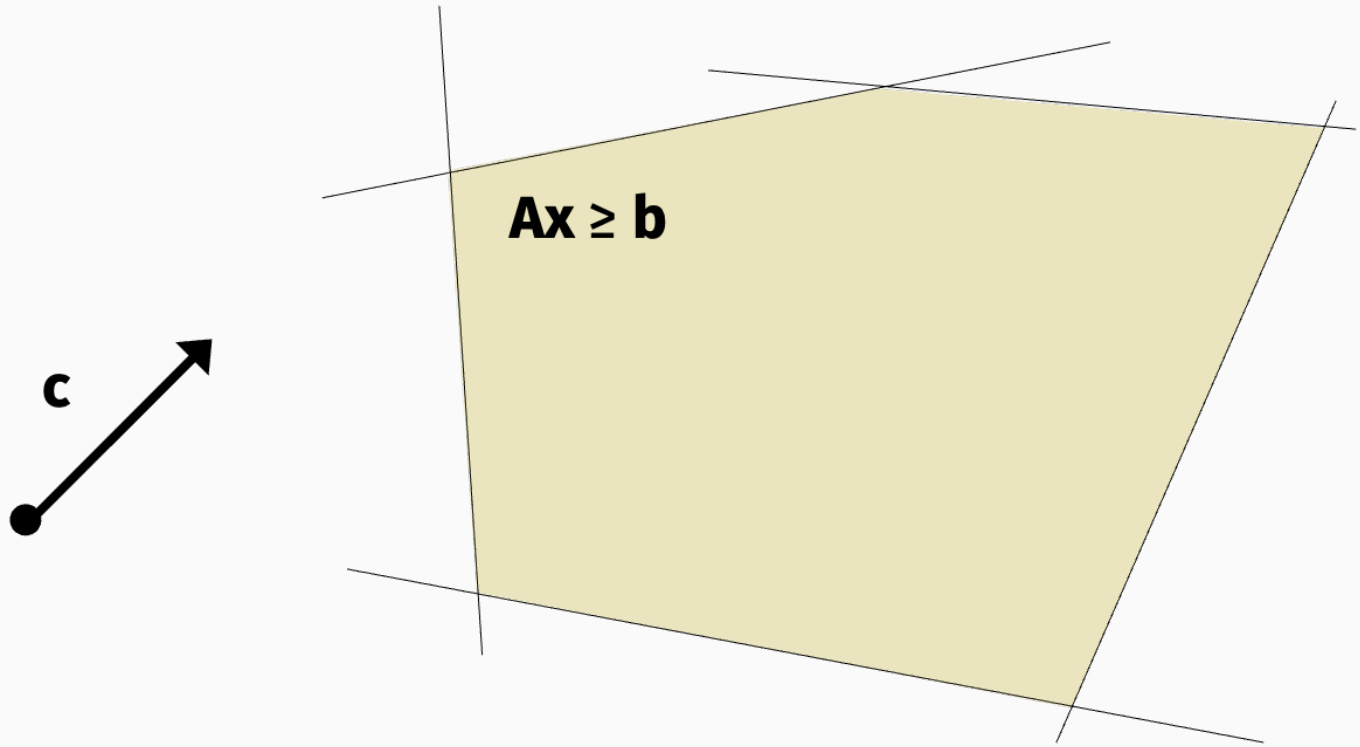
The Center-of-Gravity Method finds $\hat{\mathbf{x}}$ *satisfying* $f(\hat{\mathbf{x}}) \leq \min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}) + \epsilon$ *using* $O(d \log(B/\epsilon))$ *calls to a function and gradient oracle for convex* f.

The center-of-gravity method is not computationally efficient, but inspired the polynomial time ellipsoid method.

KILLER APPLICATION: LINEAR PROGRAMMING

$$\min f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$$

subject to $\mathbf{Ax} \geq \mathbf{b}$.



Linear programs (LPs) are one of the most basic convex constrained, convex optimization problems:

Let $\mathbf{c} \in \mathbb{R}^d, \mathbf{b} \in \mathbb{R}^n, \mathbf{A} \in \mathbb{R}^{n \times d}$ be fixed vectors that define the problem, and let $\mathbf{x}$ be our variable parameter.

$$\min f(\mathbf{x}) = \mathbf{c}^T \mathbf{x}$$
$$\text{subject to } \mathbf{A}\mathbf{x} \geq \mathbf{b}.$$

Think about $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ as a union of half-space constraints:

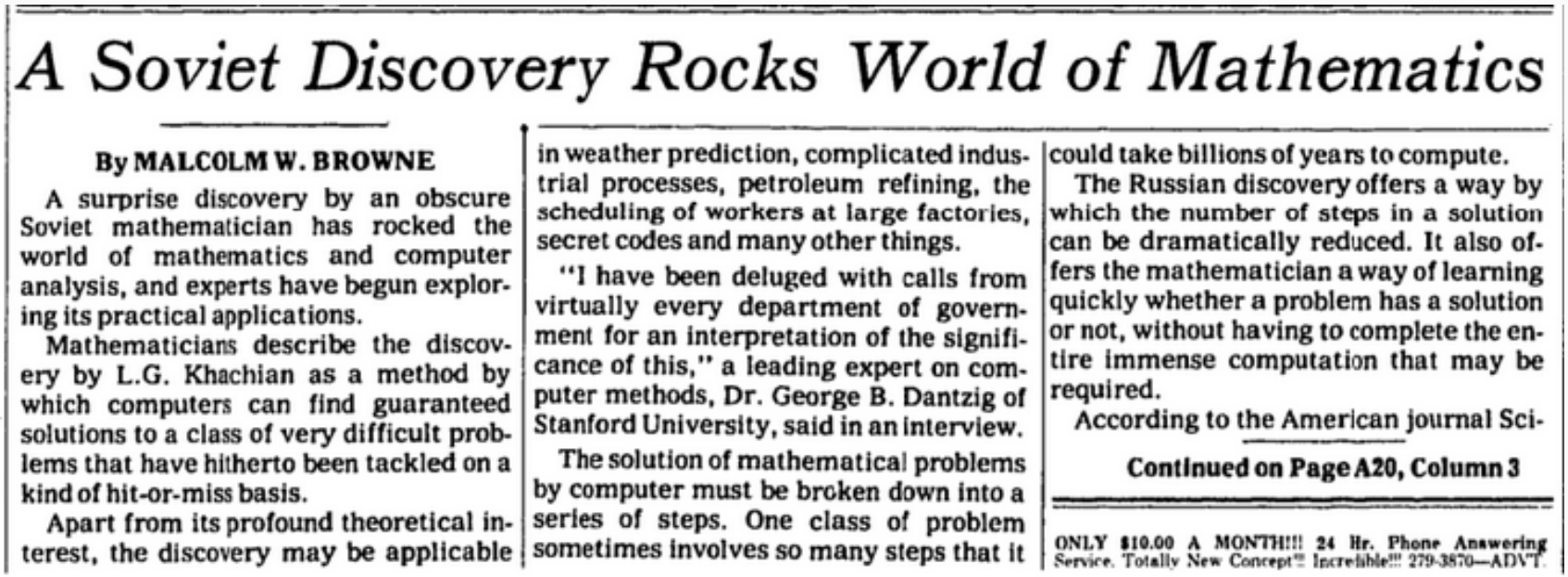
$$\{\mathbf{x} : \mathbf{a}_1^T \mathbf{x} \geq b_1\}$$
$$\{\mathbf{x} : \mathbf{a}_2^T \mathbf{x} \geq b_2\}$$
$$\vdots$$
$$\{\mathbf{x} : \mathbf{a}_n^T \mathbf{x} \geq b_n\}$$

LINEAR PROGRAMMING APPLICATIONS

- Classic optimization applications: industrial resource optimization problems were important original appications in the 70s.
- Robust regression: $\min_x \|Ax - b\|_1$.
- *L*1 constrained regression: $\min_x \|x\|_1$ subject to $Ax = b$. Lots of applications in sparse recovery/compressed sensing.
- Solve $\min_x \|Ax - b\|_\infty$.
- Polynomial time algorithms for Markov Decision Processes (reinforcement learning).
- Many combinatorial optimization problems can be solved via LP relaxation.

Theorem (Khachiyan, 1979)

Assume $n = d$. The ellipsoid method solves any linear program with L-bit integer valued constraints exactly in $O(n^4 L)$ time.



A Soviet Discovery Rocks World of Mathematics

By MALCOLM W. BROWNE

A surprise discovery by an obscure Soviet mathematician has rocked the world of mathematics and computer analysis, and experts have begun exploring its practical applications.

Mathematicians describe the discovery by L.G. Khachian as a method by which computers can find guaranteed solutions to a class of very difficult problems that have hitherto been tackled on a kind of hit-or-miss basis.

Apart from its profound theoretical interest, the discovery may be applicable in weather prediction, complicated industrial processes, petroleum refining, the scheduling of workers at large factories, secret codes and many other things.

"I have been deluged with calls from virtually every department of government for an interpretation of the significance of this," a leading expert on computer methods, Dr. George B. Dantzig of Stanford University, said in an interview.

The solution of mathematical problems by computer must be broken down into a series of steps. One class of problem sometimes involves so many steps that it could take billions of years to compute.

The Russian discovery offers a way by which the number of steps in a solution can be dramatically reduced. It also offers the mathematician a way of learning quickly whether a problem has a solution or not, without having to complete the entire immense computation that may be required.

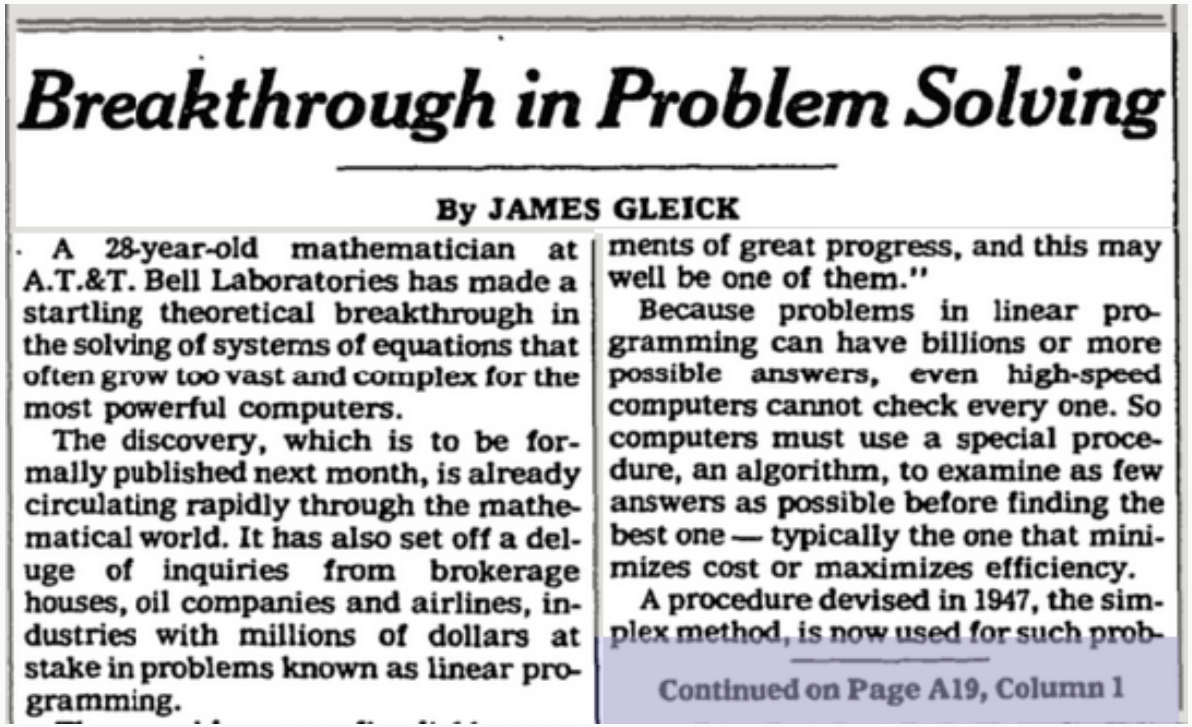
According to the American journal Sci-

Continued on Page A20, Column 3

Front page of New York Times, November 9, 1979.

Theorem (Karmarkar, 1984)

Assume $n = d$. The interior point method solves any linear program with L-bit integer valued constraints in $O(n^{3.5}L)$ time.



Breakthrough in Problem Solving

By JAMES GLEICK

A 28-year-old mathematician at A.T.&T. Bell Laboratories has made a startling theoretical breakthrough in the solving of systems of equations that often grow too vast and complex for the most powerful computers.

The discovery, which is to be formally published next month, is already circulating rapidly through the mathematical world. It has also set off a deluge of inquiries from brokerage houses, oil companies and airlines, industries with millions of dollars at stake in problems known as linear programming.

ments of great progress, and this may well be one of them."

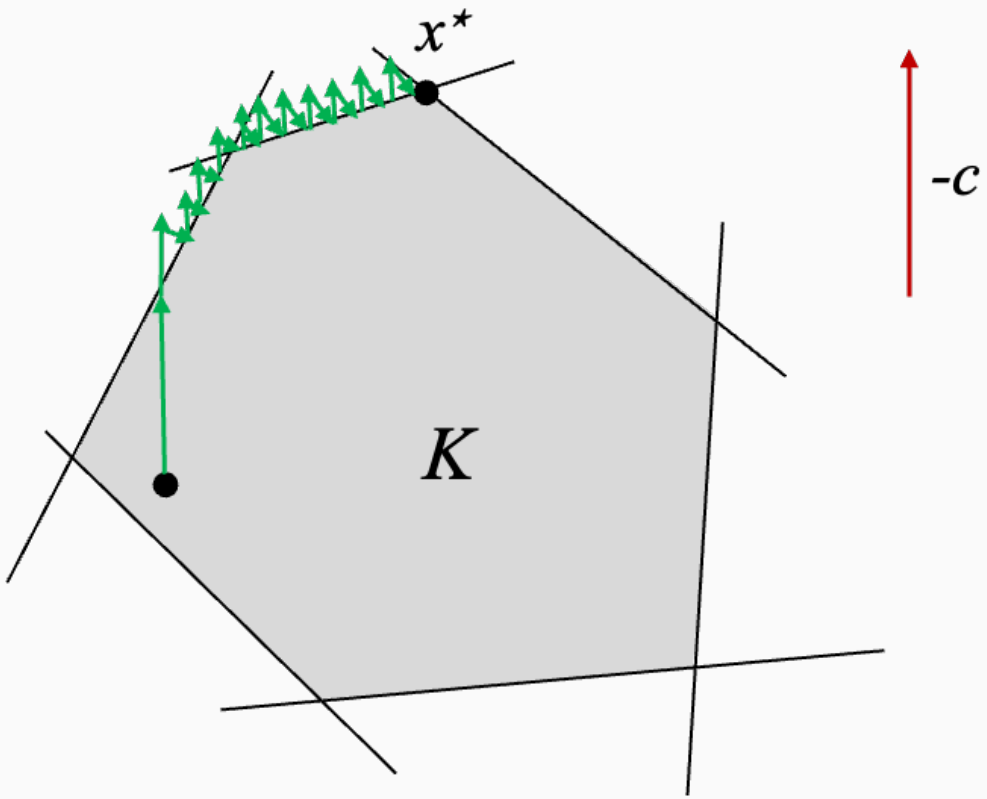
Because problems in linear programming can have billions or more possible answers, even high-speed computers cannot check every one. So computers must use a special procedure, an algorithm, to examine as few answers as possible before finding the best one — typically the one that minimizes cost or maximizes efficiency.

A procedure devised in 1947, the simplex method, is now used for such prob-

Continued on Page A19, Column 1

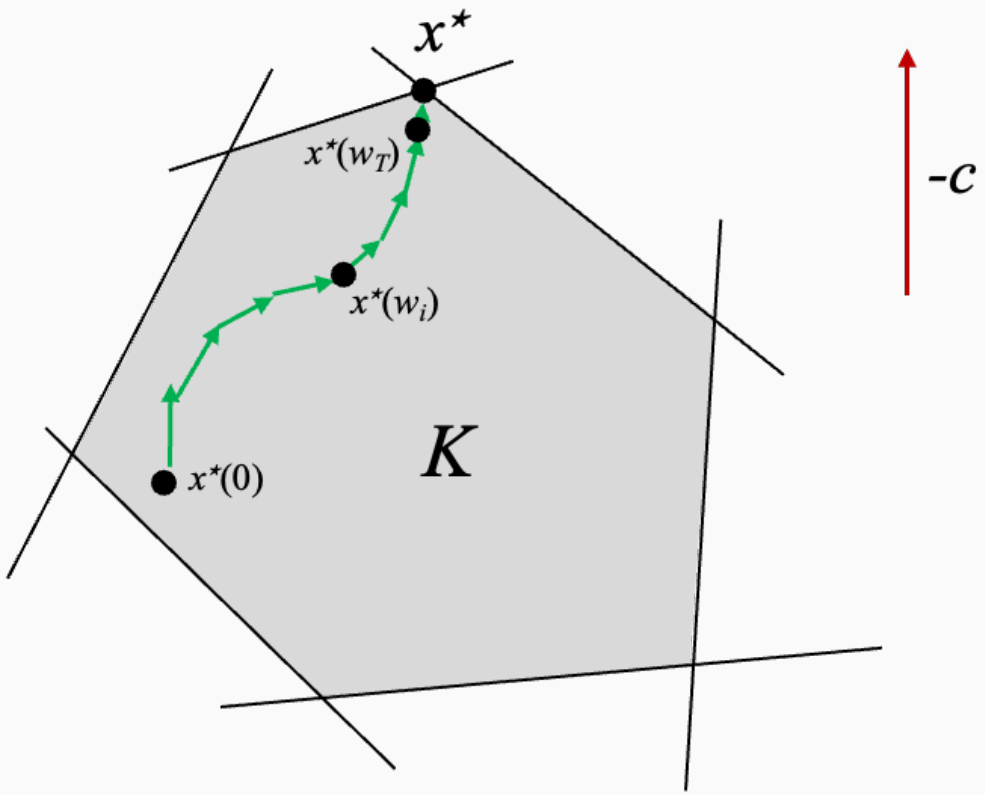
Front page of New York Times, November 19, 1984.

Lecture notes are posted on the website (optional reading).



Projected Gradient Descent Optimization Path

Lecture notes are posted on the website (optional reading).



Ideal Interior Point Optimization Path

POLYNOMIAL TIME LINEAR PROGRAMMING

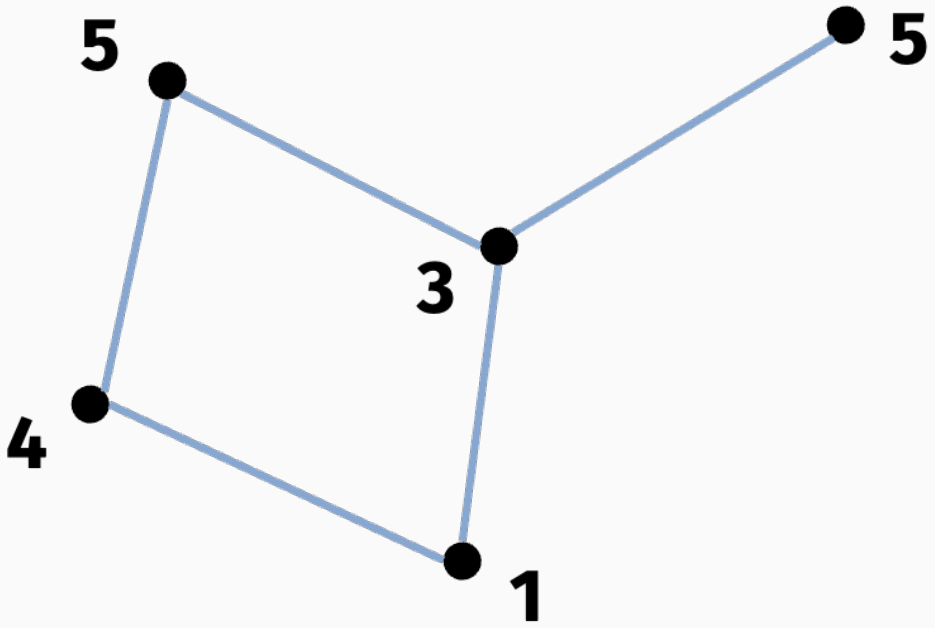
Both results had a huge impact on the theory of optimization, although at the time neither the ellipsoid method or interior point method were faster than a heuristic known at the Simplex Method.

These days, improved interior point methods compete with and often outperform simplex.

Polynomial time linear programming algorithms have also had a huge impact of combinatorial optimization. They are often the work-horse behind approximation algorithms for NP-hard problems.

EXAMPLE: VERTEX COVER

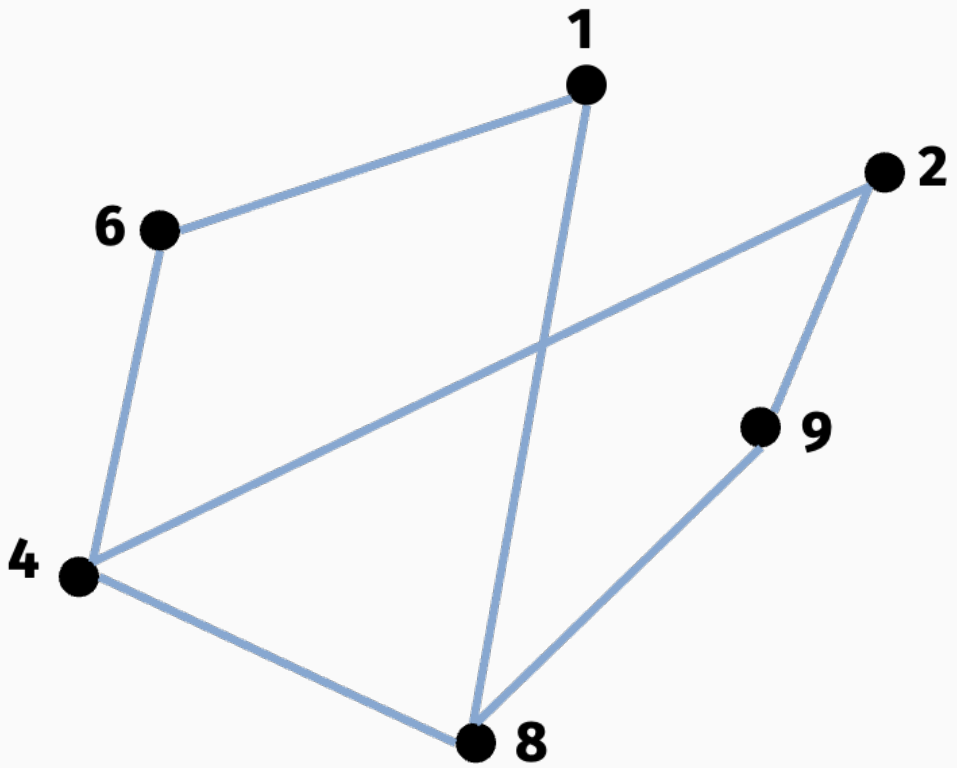
Given a graph G with n nodes and edge set E. Each node is assigned a weight $w_1, \ldots, w_n$.



Goal: Select subset of nodes with minimum total weight that covers all edges.

EXAMPLE: VERTEX COVER

NP-hard to solve exactly.



EXAMPLE: VERTEX COVER

Given a graph G with n nodes and edge set E. Each node is assigned a weight $w_1, \ldots, w_n$.

Formally: Denote if node i is selected by assigning variable x_i to 0 or 1. Let $\mathbf{x} = [x_1, \ldots, x_n]$.

$$\min_{\mathbf{x}} \sum_{i=1}^{n} x_i w_i \qquad \text{subject to} \qquad x_i \in \{0, 1\} \text{ for all } i$$

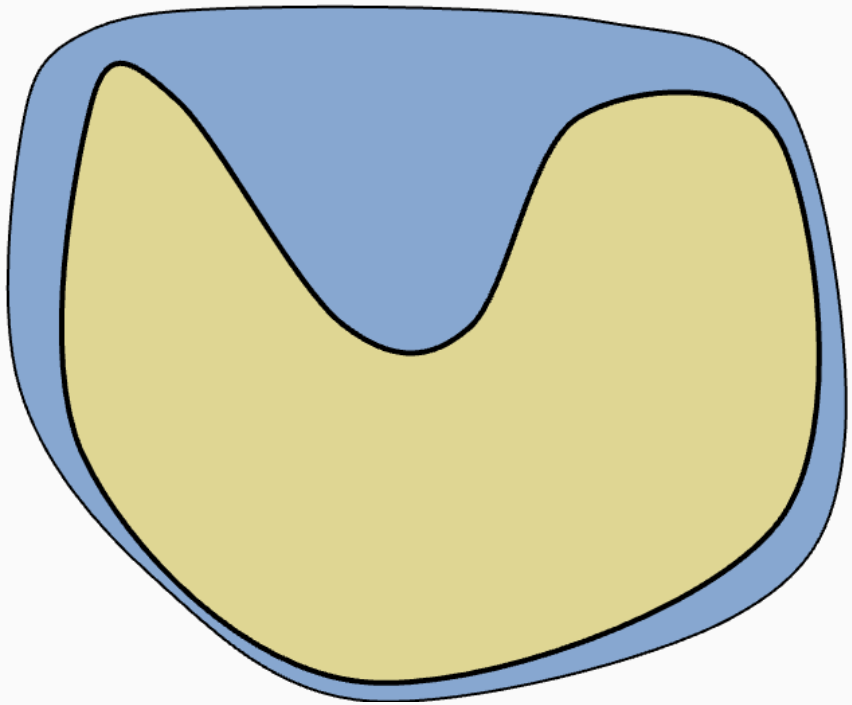
$$x_i + x_j \geq 1 \text{ for all } (i, j) \in E$$

We will use convex optimization give a 2-approximation in polynomial time.

Function to minimize is linear (so convex) but constraint set is not convex. Why?

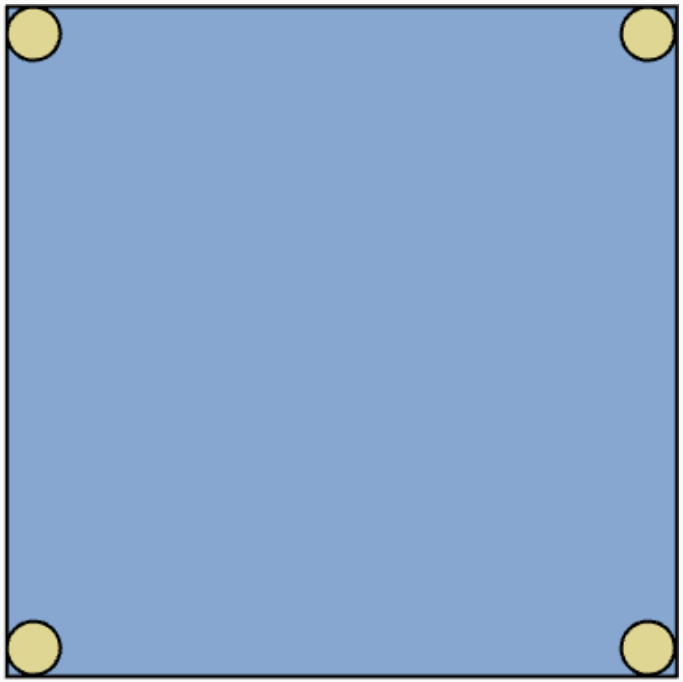
High level approach:

- Relax to a problem with convex constraints.
- Round optimal solution of convex problem back to original constraint set.



High level approach:

- Relax to a problem with convex constraints.
- Round optimal solution of convex problem back to original constraint set.



High level approach:

- Relax to a problem with convex constraints.
- Round optimal solution of convex problem back to original constraint set.

Let $\bar{\mathcal{S}} \supseteq \mathcal{S}$ be the relaxed constraint set. Let $\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x})$ and let $\bar{\mathbf{x}}^* = \arg\min_{\mathbf{x} \in \bar{\mathcal{S}}} f(\mathbf{x})$. We always have that:

$$f(\bar{\mathbf{x}}^*) \leq f(\mathbf{x}^*).$$

So typically the goal is to round $\bar{\mathbf{x}}^*$ to $\mathcal{S}$ in such a way that we don't increase the function value too much.

RELAXING VERTEX COVER

Vertex Cover:

$$\min_x \sum_{i=1}^{n} x_i w_i \qquad \text{subject to} \qquad x_i \in \{0,1\} \text{ for all } i$$

$$x_i + x_j \geq 1 \text{ for all } (i,j) \in E$$

Relaxed Vertex Cover:

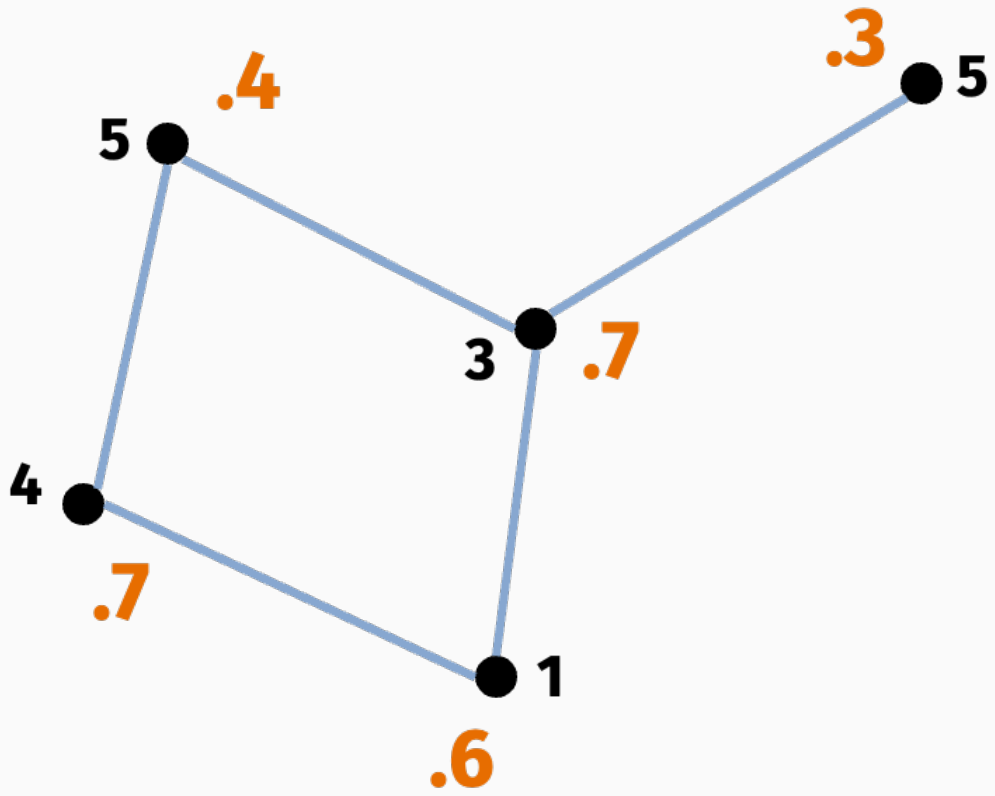
$$\min_x \sum_{i=1}^{n} x_i w_i \qquad \text{subject to} \qquad 0 \leq x_i \leq 1 \text{ for all } i$$

$$x_i + x_j \geq 1 \text{ for all } (i,j) \in E$$

The second problem is a linear program! It can be solved in $\mathsf{poly}(n)$ time!

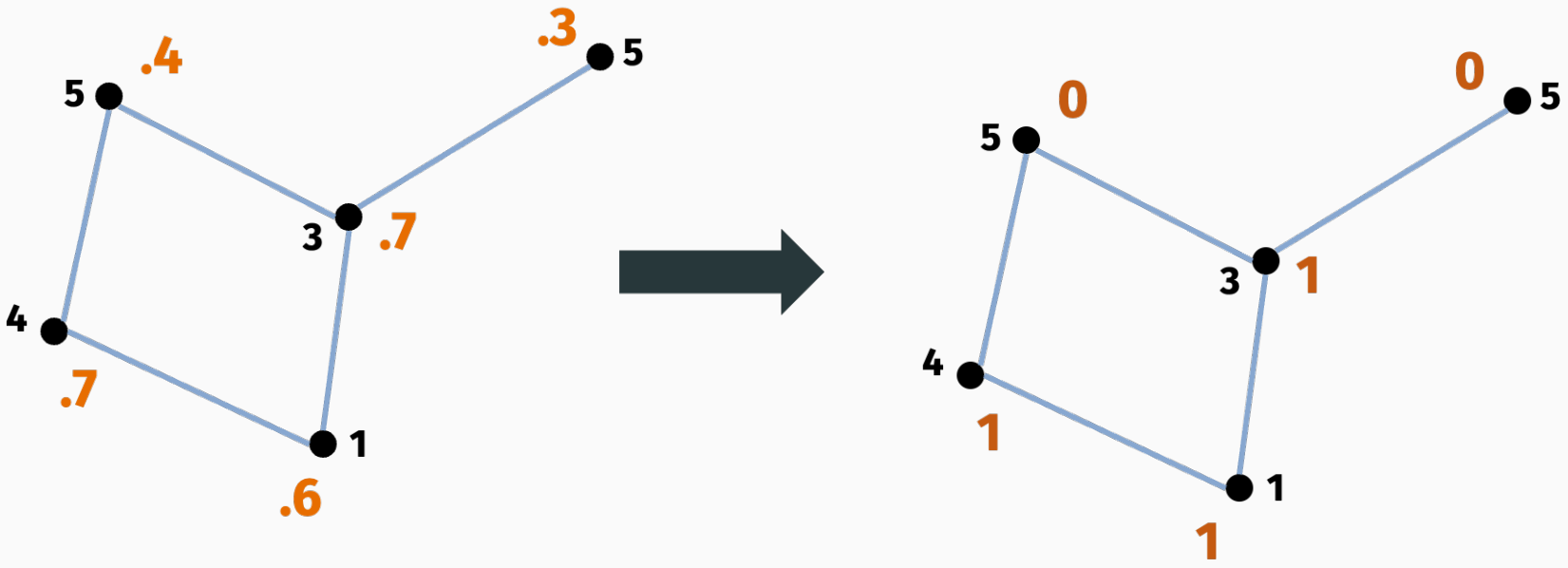
ROUNDING VERTEX COVER

Any ideas on how to round this to a solution to the original problem? I.e., with constraints $x_i \in \{0, 1\}$ for all i.



ROUNDING VERTEX COVER

Simply set all variable $x_i = 1$ of $\bar{x}_i^* \geq 1/2$ and $x_i = 0$ otherwise.



Observation 1: All edges remain covered. I.e., the constraint $x_i + x_j \geq 1$ for all $(i, j) \in E$ is not violated.

Observation 2: Let $\mathbf{x}$ be the rounded version of $\bar{\mathbf{x}}^*$. We have $f(\mathbf{x}) \leq 2 \cdot f(\bar{\mathbf{x}})$, and thus $f(\mathbf{x}) \leq 2 \cdot f(\mathbf{x}^*)$.

Proof:

VERTEX COVER

So, a polynomial time algorithm for solving LPs immediately yields a 2-approximation algorithm for the NP-hard problem of vertex cover.

- Proven that it is NP-hard to do better than a 1.36 approximation in [Dinur, Safra, 2002].
- Recently improved to $\sqrt{2} \approx 1.41$ in [Khot, Minzer, Safra 2018], which proved the 2-to-2 games conjecture.
- Widely believed that doing better than $2 - \epsilon$ is NP-hard for any $\epsilon > 0$, and this is implied by Subhash Khot's Unique Games Conjecture.

There is a simpler greedy 2-approximation algorithm that doesn't use optimization at all!

BREAK

SPECTRAL METHODS

Next section of course: Spectral methods and numerical linear algebra.

Spectral methods generally refer to methods based on the "spectrum" of a matrix. I.e. on it's eigenvectors/eigenvalues and singular vectors/singular values. We will look at

- Applications to low-rank approximation and dimensionality reduction.
- Applications to graph problems.
- Fast algorithms for computing spectral information.

Reminder: A vector $\mathbf{v} \in \mathbb{R}^d$ is an eigenvector of a matrix $\mathbf{X} \in \mathbb{R}^{d \times d}$, if there exists a scalar λ such that

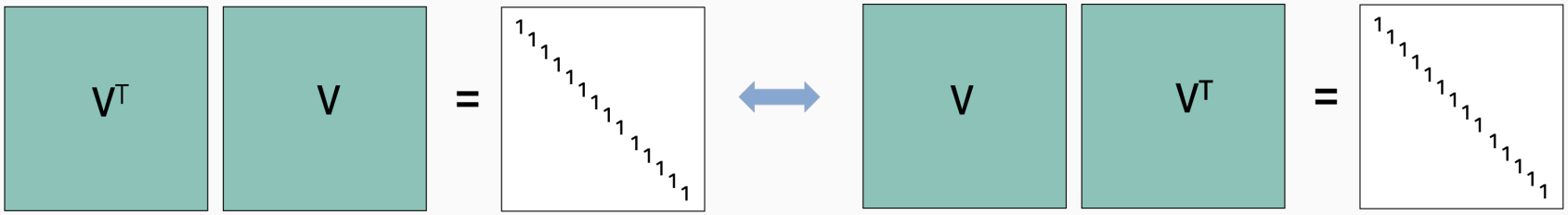
$$\mathbf{X}\mathbf{v} = \lambda \mathbf{v}$$

The scalar λ is called the eigenvalue associated with $\mathbf{v}$.

Matrices can often be written completely in terms of their eigenvectors and eigenvalues. This is called eigendecomposition.

We will actually focus on a related tool called singular value decomposition.

If a square matrix has orthonormal rows, it also has orthonormal columns:



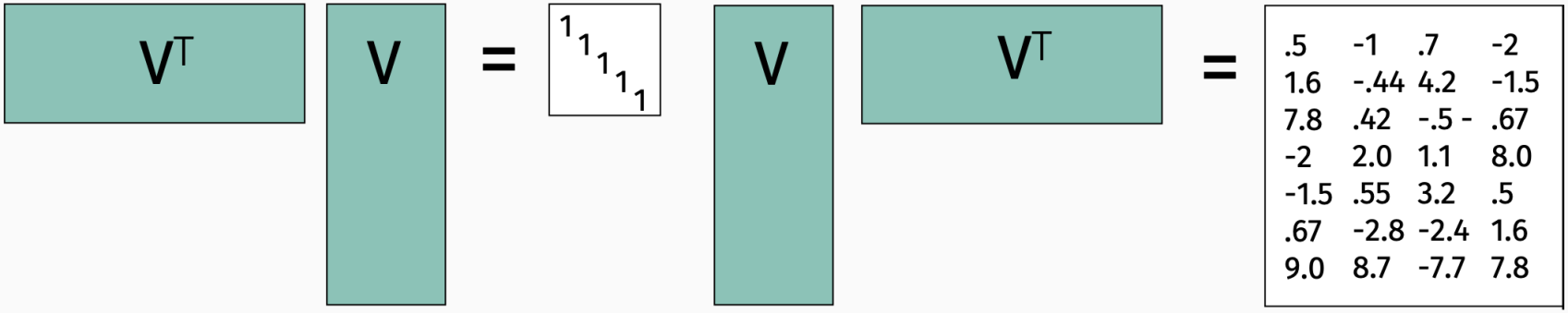
$$V^T V = I = V V^T$$

$$\begin{bmatrix} -0.62 & 0.78 & -0.11 \\ -0.28 & -0.35 & -0.89 \\ -0.73 & -0.52 & 0.44 \end{bmatrix} \cdot \begin{bmatrix} -0.62 & -0.28 & -0.73 \\ 0.78 & -0.35 & -0.52 \\ -0.11 & -0.89 & 0.44 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Implies that for any vector $\mathbf{x}$, $\|\mathbf{V}\mathbf{x}\|_2^2 = \|\mathbf{x}\|_2^2$ and $\|\mathbf{V}^T\mathbf{x}\|_2^2$.

Same thing goes for Frobenius norm: for any matrix $\mathbf{X}$, $\|\mathbf{V}\mathbf{X}\|_F^2 = \|\mathbf{X}\|_F^2$ and $\|\mathbf{V}^T\mathbf{X}\|_F^2 = \|\mathbf{X}\|_F^2$.

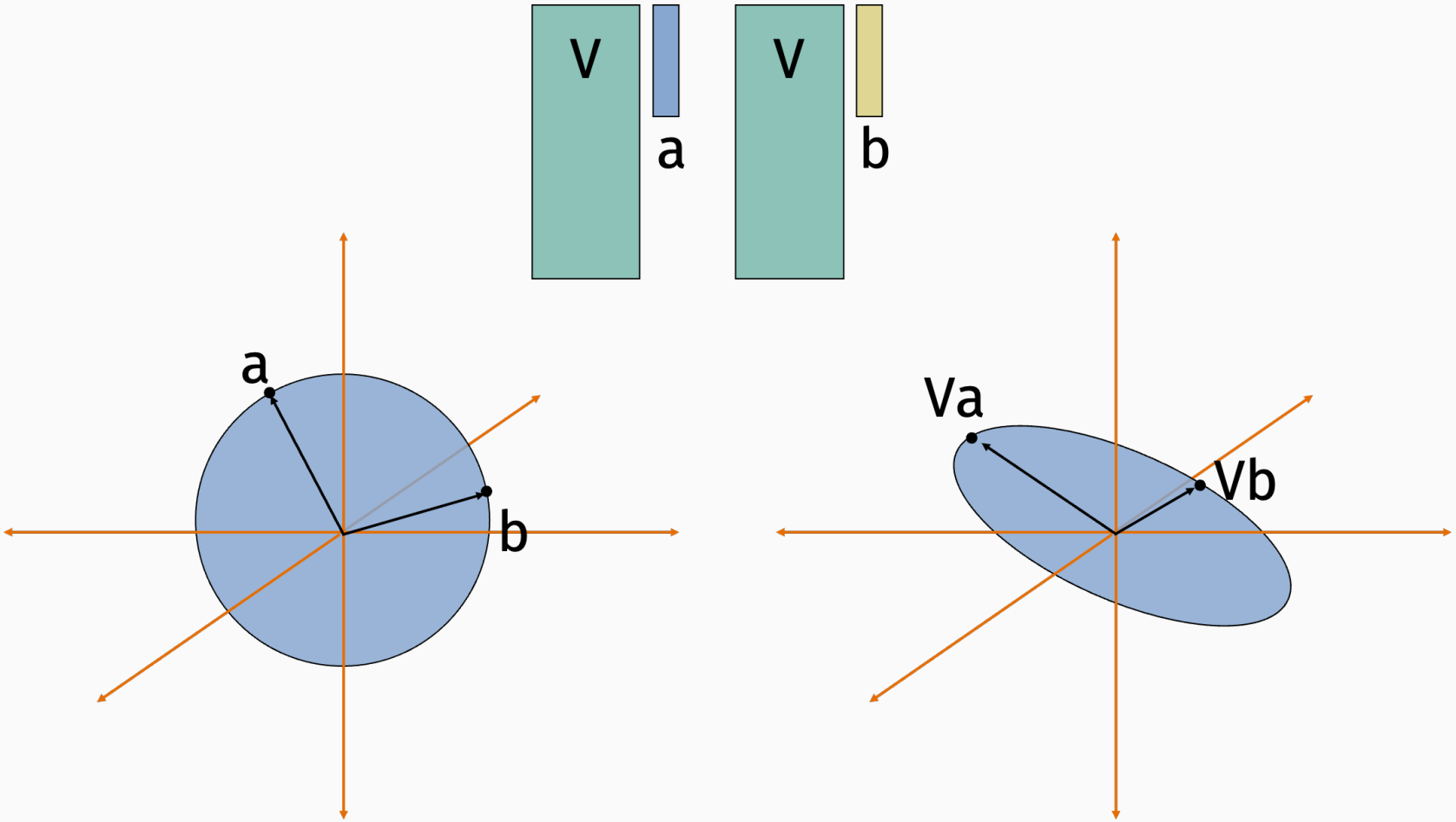
The same is not true for rectangular matrices.



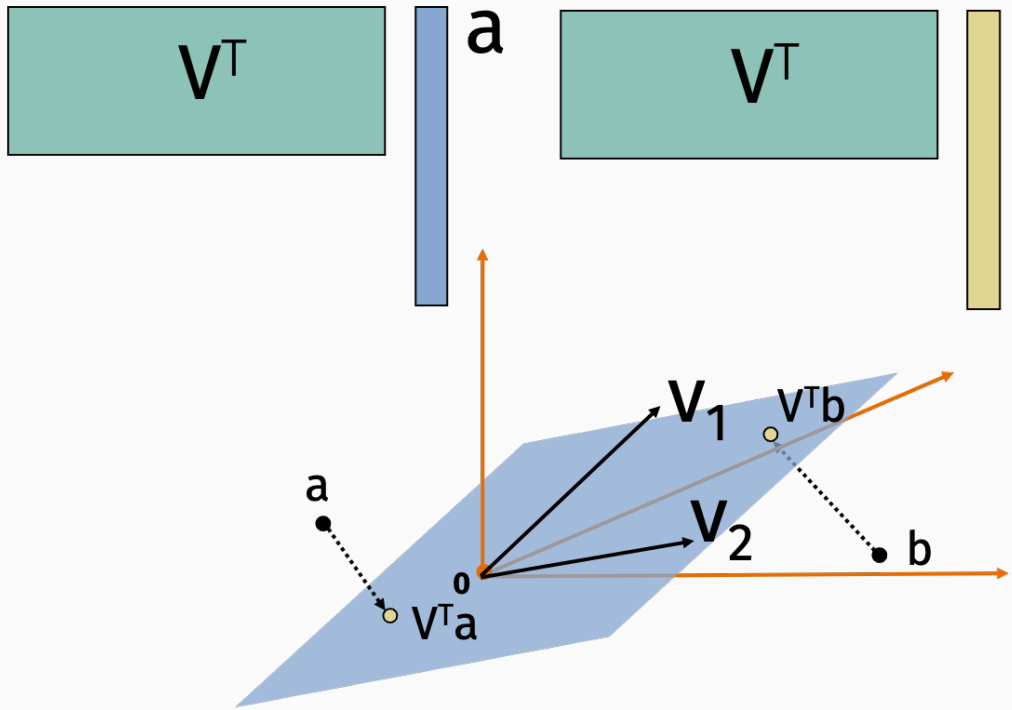
$$\mathbf{V}^T\mathbf{V} = \mathbf{I} \qquad \text{but} \qquad \mathbf{V}\mathbf{V}^T \neq \mathbf{I}$$

For any $\mathbf{x}$, $\|\mathbf{V}\mathbf{x}\|_2^2 = \|\mathbf{x}\|_2^2$ but $\|\mathbf{V}^T\mathbf{x}\|_2^2 \neq \|\mathbf{x}\|_2^2$ in general.

Multiplying a vector by **V** with orthonormal columns rotates and/or reflects the vector.



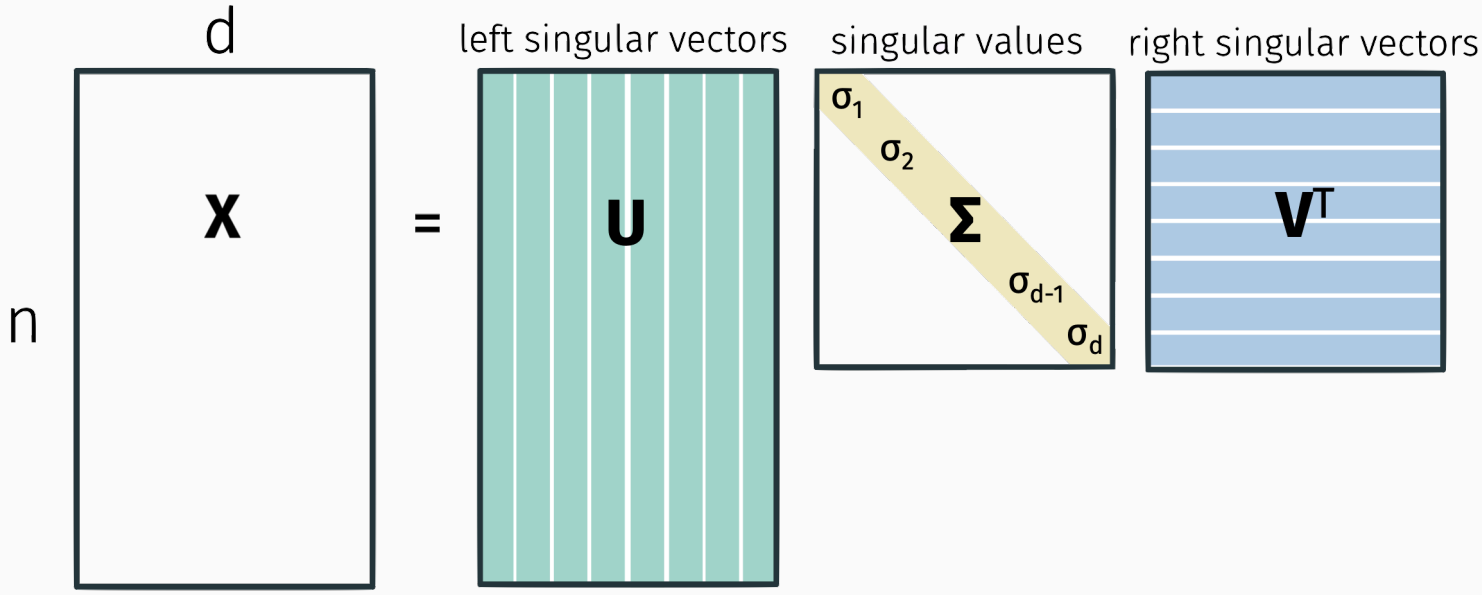
Multiplying a vector by a rectangular matrix $\mathbf{V}^T$ with orthonormal rows projects the vector (representing it as coordinates in the lower dimensional space).



So we always have that $\|\mathbf{V}^T\mathbf{x}\|_2 \leq \|\mathbf{x}\|_2$.

One of the most fundamental results in linear algebra.

Any matrix $\mathbf{X}$ can be written:



Where $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}$, and $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_d \geq 0$.

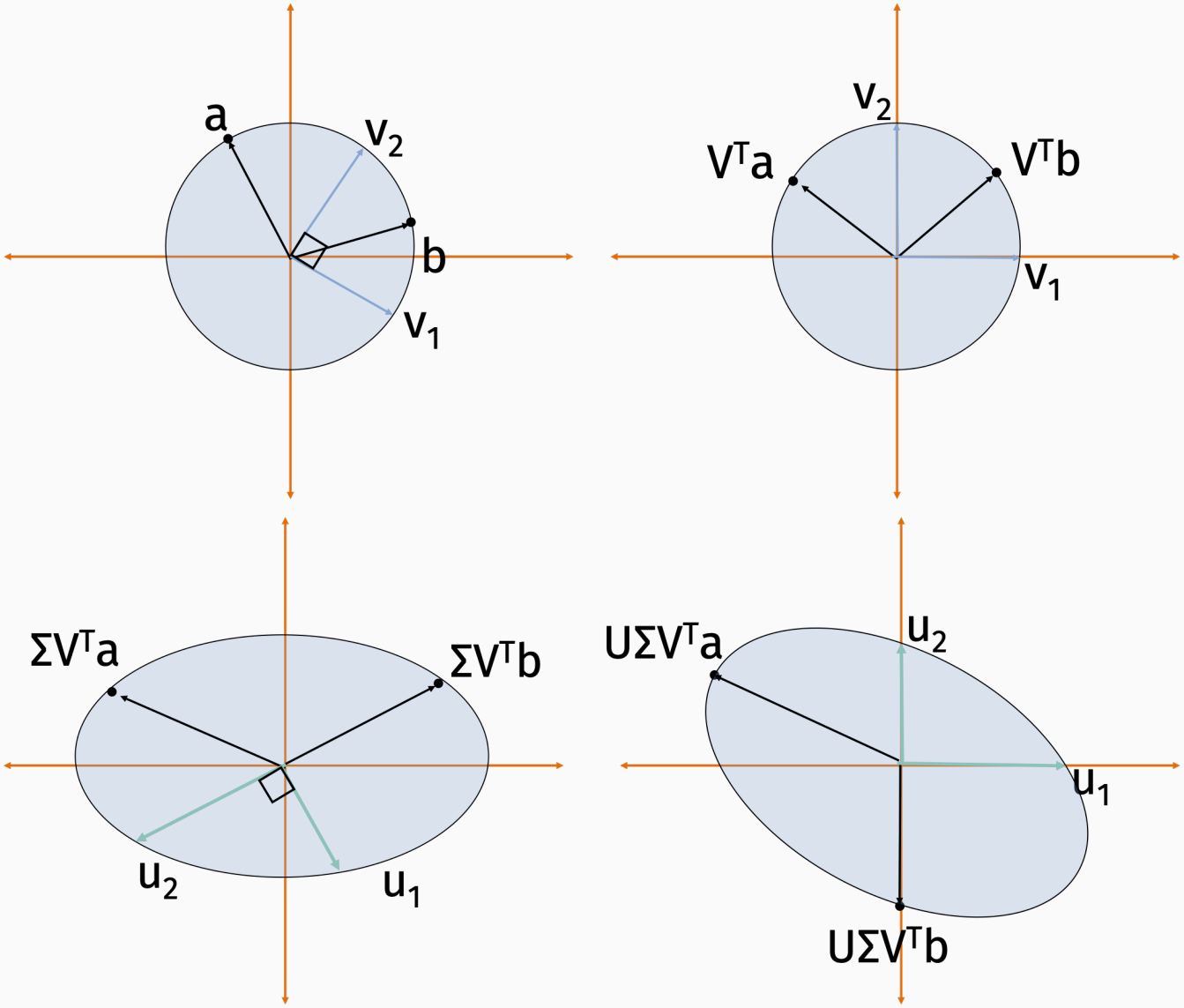
Singular values are unique. Factors are not. E.g. would still get a valid SVD by multiplying both i^{th} column of $\mathbf{V}$ and $\mathbf{U}$ by -1.

Important take away from singular value decomposition.

Multiplying any vector $\mathbf{a}$ by a matrix $\mathbf{X}$ to form $\mathbf{Xa}$ can be viewed as a composition of 3 operations:

1. Rotate/reflect the vector (multiplication by to $\mathbf{V}^T$).
2. Scale the coordinates (multiplication by $\mathbf{\Sigma}$.
3. Rotate/reflect the vector again (multiplication by $\mathbf{U}$).

SINGULAR VALUE DECOMPOSITION: ROTATE/REFLECT



COMPARISON TO EIGENDECOMPOSITION

A square matrix has at most d linearly independent eigenvectors. If a matrix has a full set of d eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_d$ with eigenvalues $\lambda_1, \ldots, \lambda_d$ it is called "diagonalizable" and can be written as:

$$\mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}.$$

$\mathbf{V}$'s columns are $\mathbf{v}_1, \ldots, \mathbf{v}_d$.

COMPARISON TO EIGENDECOMPOSITION

Singluar value decomposition

- Exists for all matrices, square or rectangular.
- Singular values are always positive.
- Factors $\mathbf{U}$ and $\mathbf{V}$ are orthogonal.

Eigendecomposition

- Exists for some square matrices.
- Eigenvalues can be positive, negative, or imaginary. Real if $\mathbf{X}$ is symmetric.
- Factor $\mathbf{V}$ is orthogonal if and only if $\mathbf{X}$ is symmetric.

CONNECTION TO EIGENDECOMPOSITION

- $\mathbf{U}$ contains the orthogonal eigenvectors of $\mathbf{X}\mathbf{X}^T$.
- $\mathbf{V}$ contains the orthogonal eigenvectors of $\mathbf{X}^T\mathbf{X}$.
- $\sigma_i^2 = \lambda_i(\mathbf{X}\mathbf{X}^T) = \lambda_i(\mathbf{X}^T\mathbf{X})$

Lots of applications.

- Compute pseudoinverse $\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T$.
- Read off condition number of $\mathbf{X}$, σ_1^2/σ_d^2.
- Compute matrix norms. E.g. $\|\mathbf{X}\|_2 = \sigma_1$, $\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{d} \sigma_i^2}$.
- Compute matrix square root – i.e. find a matrix $\mathbf{B}$ such that $\mathbf{B}\mathbf{B}^T = \mathbf{X}$. Used e.g. in sampling from Gaussian with covariance $\mathbf{X}$.
- Principal component analysis.

Killer app: Read off optimal low-rank approximation for $\mathbf{X}$.

RANK

The column span of a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the set of all vectors that can be written as $\mathbf{Xa}$ for some $\mathbf{a}$.

The dimension of the column span, D_c, is the maximum number of linear independent vectors in that set.

The row span of a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the set of all vectors that can be written as $\mathbf{b}^T\mathbf{X}$ for some $\mathbf{b}$.

The dimension of the row span, D_r, is the maximum number of linear independent vectors in that set.

RANK

For a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ we have:

$$D_c \leq d$$

$$D_r \leq n$$

$$D_c = D_r.$$

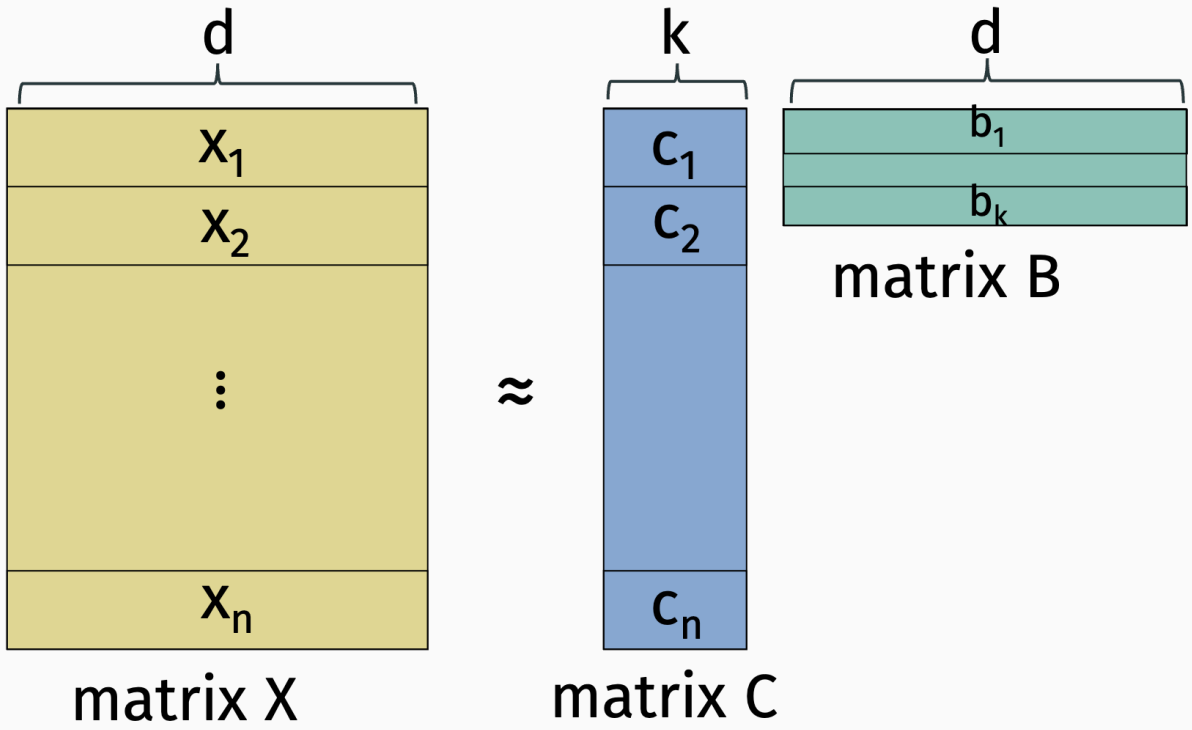
We call the value of $D_c = D_r$ the rank of $\mathbf{X}$.

We always have that:

$$\text{rank}\left(\mathsf{A} \cdot \mathsf{B} \cdot \mathsf{C} \cdot \ldots\right) \leq \min\left(\text{rank}(\mathsf{A}), \text{rank}(\mathsf{B}), \text{rank}(\mathsf{C}), \ldots\right).$$

LOW-RANK APPROXIMATION

Approximate $\mathbf{X}$ as a rank k matrix:

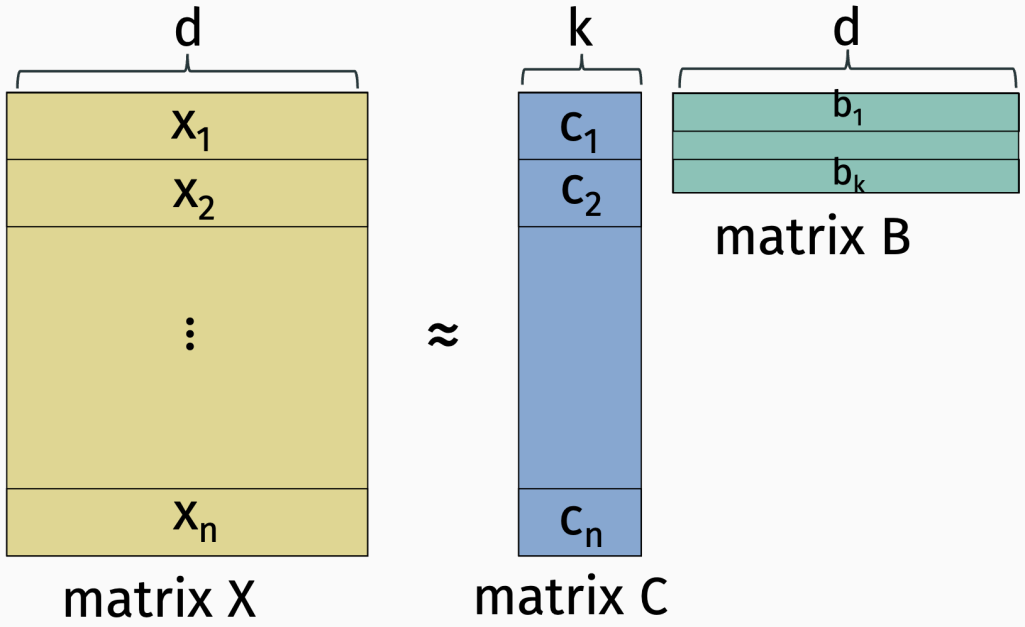


Choose $\mathbf{C}$ and $\mathbf{B}$ to minimize:

$$\min_{\mathbf{B},\mathbf{C}} \|\mathbf{X} - \mathbf{CB}\|$$

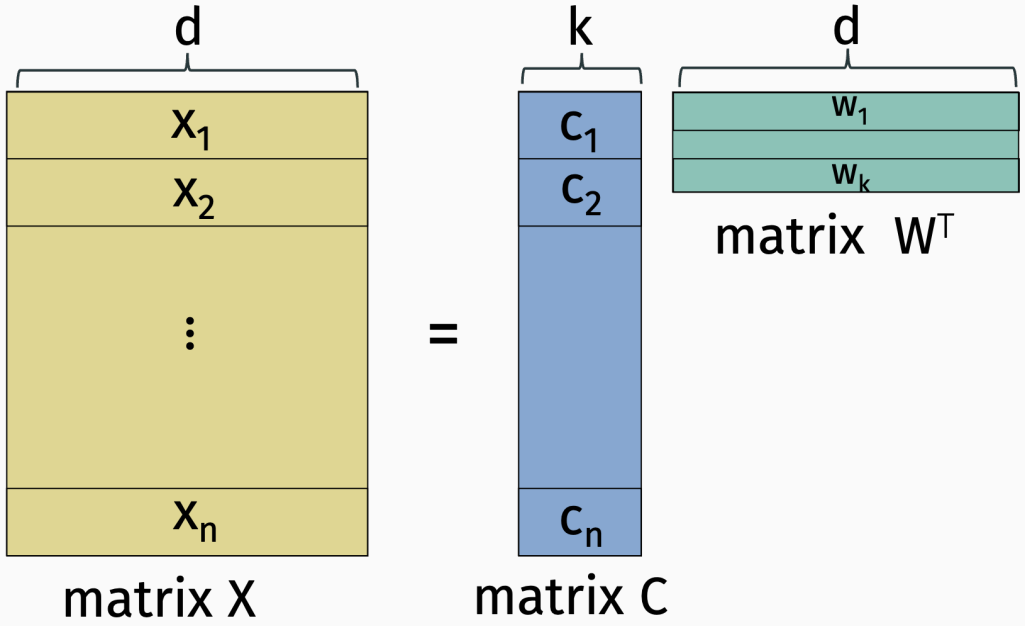
for some matrix norm. Common choice is $\|\mathbf{X} - \mathbf{CB}\|_F^2$.

APPLICATIONS OF LOW-RANK APPROXIMATION



- **CB** takes $O(k(n+d))$ space to store instead of $O(nd)$.
  - Important in many applications, including e.g. LoRA: Low-Rank Adaptation of Large Language Models
  - Can be used to compress vector databases.
  - Many more applications.
- Many linear algebraic problems involving **CB** can be solved in $O(nk^2)$ instead of $O(nd^2)$ time.

Without loss of generality can assume that the right matrix is orthogonal. I.e. $\mathbf{W}^T$ with $\mathbf{W}^T\mathbf{W} = \mathbf{I}$



Then we should choose left matrix $\mathbf{C}$ to minimize:

$$\min_{\mathbf{C}} \|\mathbf{X} - \mathbf{C}\mathbf{W}^T\|_F^2$$

This is just n least squares regression problems!

$$c_i = \arg\min_c \|Wc - x_i\|_2^2$$

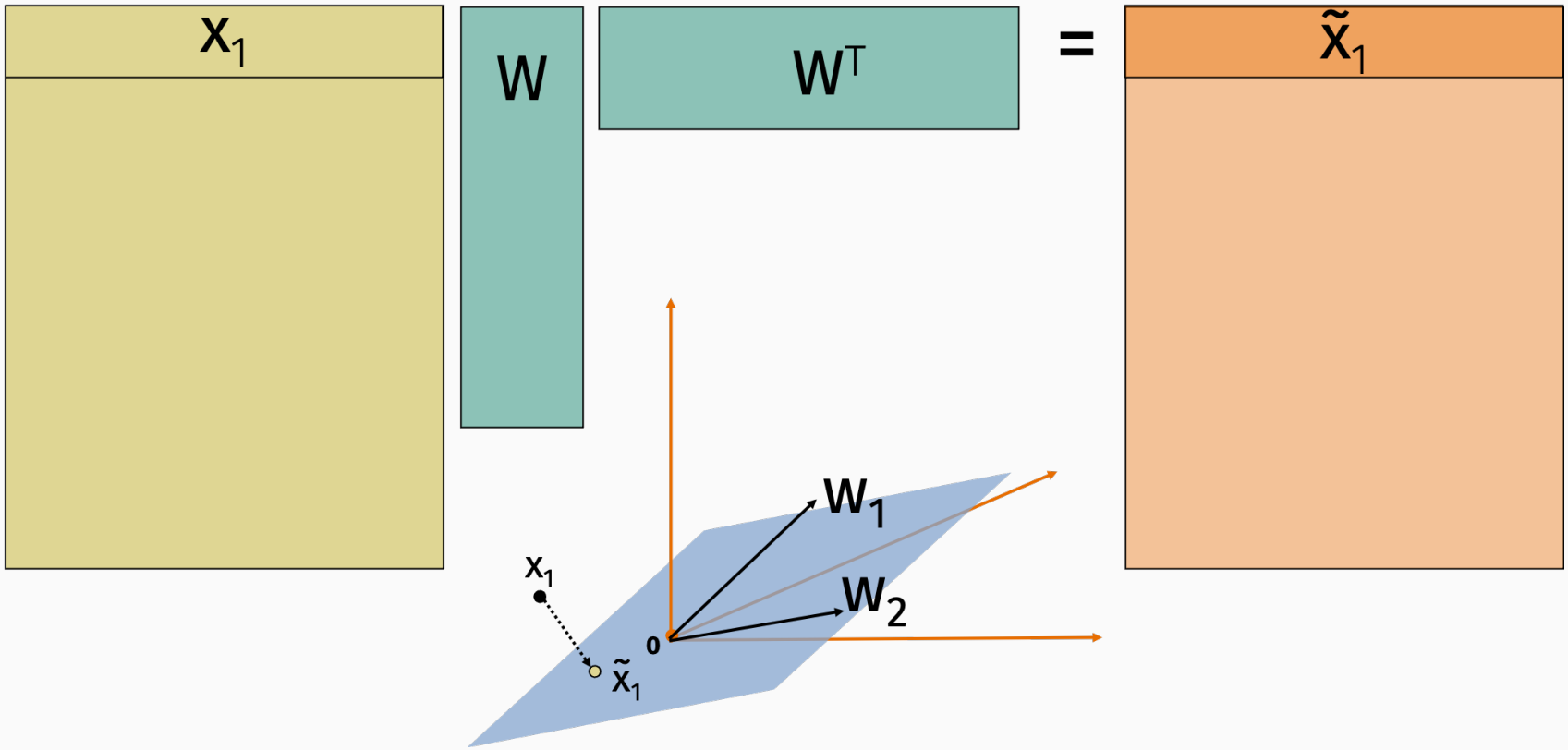
$$c_i = W^T x_i$$

$$C = XW$$

So our optimal low-rank approximation always has the form:

$$X \approx XWW^T$$

WW^T is a symmetric projection matrix.



$\mathbf{C} = \mathbf{XW}$ can be used as a meaningful compressed version of data matrix $\mathbf{X}$. We have that:

$$\|\mathbf{x}_i - \mathbf{x}_j\|_2 \approx \|\mathbf{W}\mathbf{W}^T\mathbf{x}_i - \mathbf{W}\mathbf{W}^T\mathbf{x}_j\|_2 = \|\mathbf{c}_i - \mathbf{c}_i\|_2$$

So we expect that:

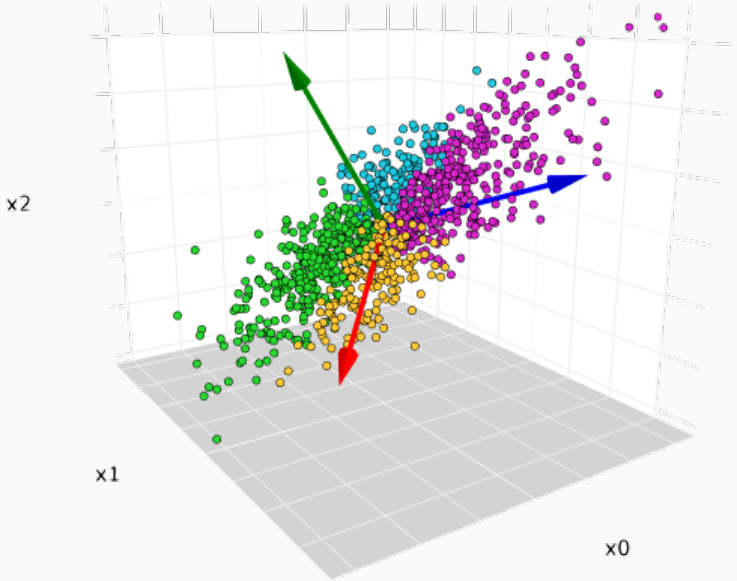
- $\|\mathbf{x}_i\|_2 \approx \|\mathbf{c}_i\|_2$
- $\langle \mathbf{x}_i, \mathbf{x}_j \rangle \approx \langle \mathbf{c}_i, \mathbf{c}_j \rangle$
- etc.

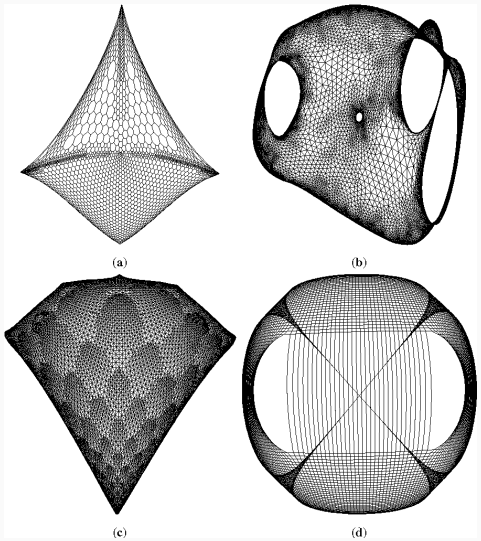
How does this compare to Johnson-Lindenstrauss projection?

APPLICATIONS OF LOW-RANK APPROXIMATION

Also useful in:

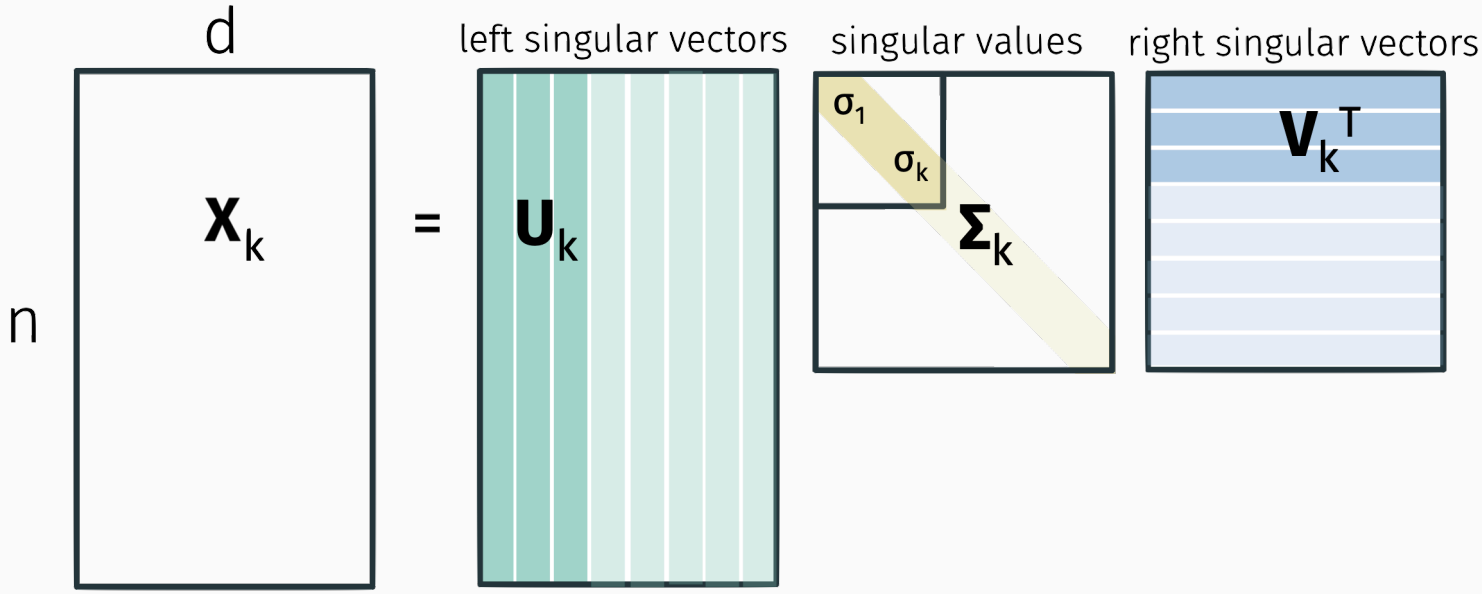
- Data visualization when $k = 2$ or 3.





- Data denoising (e.g. distance triangulation).
- Feature selection.

Key result: Can find the best projection from the singular value decomposition. **Note:** $X_k = U_k \Sigma_k V_k^T = U_k U_k^T X = X V_k V_k^T$.



$$U_k = \underset{\text{orthogonal } Z \in \mathbb{R}^{d \times k}}{\arg\min} \|X - ZZ^T X\|_F^2$$

$$V_k = \underset{\text{orthogonal } W \in \mathbb{R}^{d \times k}}{\arg\min} \|X - XWW^T\|_F^2$$

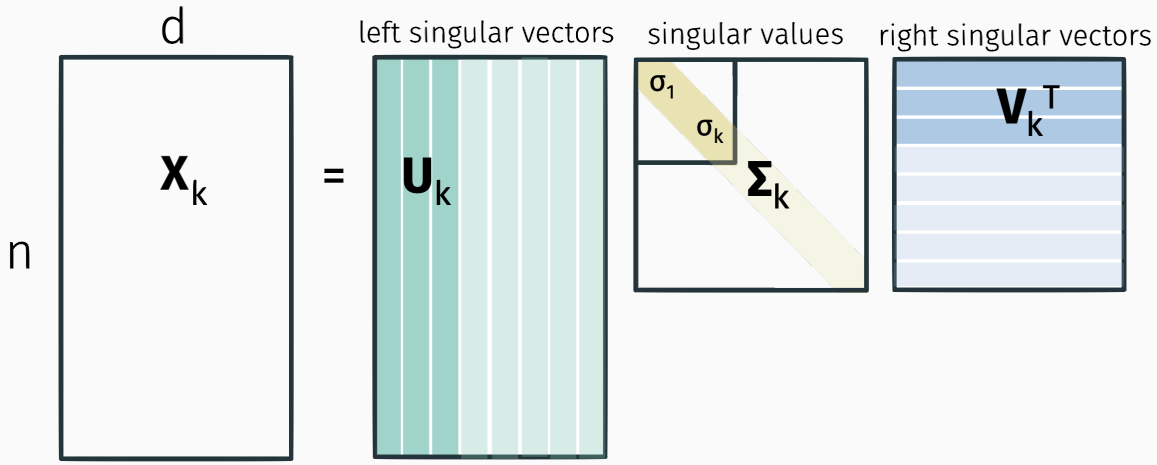
OPTIMAL LOW-RANK APPROXIMATION

Goal: Minimize $\|\mathbf{X} - \mathbf{B}\|_F$.

Claim 1: Without loss of generality, can assume $\mathbf{B} = \mathbf{U}\mathbf{Z}\mathbf{V}^T$ for some other rank k matrix $\mathbf{Z}$.

Goal: Minimize $\|\mathbf{X} - \mathbf{B}\|_F$.

Claim 2: Should choose $\mathbf{Z}$ to be the best rank k approximation to $\mathbf{\Sigma}$. (We will then show this equals $\mathbf{\Sigma}_k$.)



Claim 3:

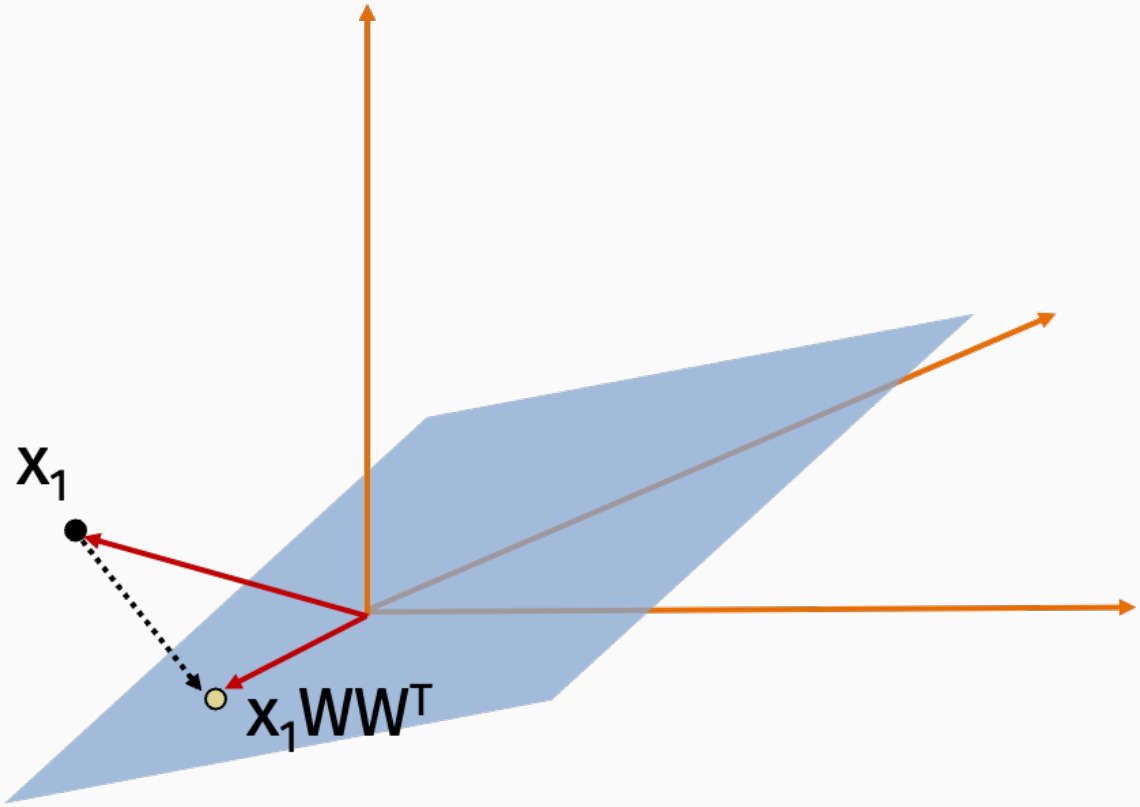
$$\arg\min_{\mathbf{W}\in\mathbb{R}^{d\times k}} \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2 = \arg\max_{\mathbf{W}\in\mathbb{R}^{d\times k}} \|\mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2$$

Follows from fact that for all orthogonal $\mathbf{W}$:

$$\|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2$$

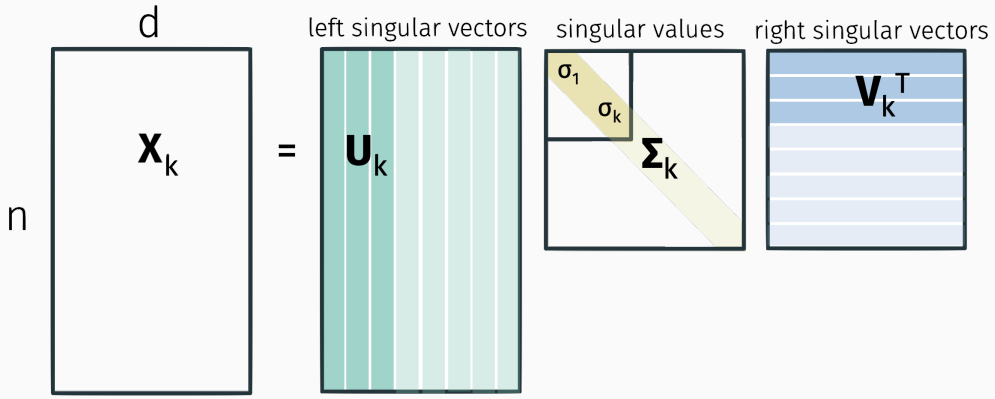
Claim 3:

$$\|X - XWW^T\|_F^2 = \|X\|_F^2 - \|XWW^T\|_F^2$$



Final Step: Let $\mathbf{W}^* \in \mathbb{R}^{d \times k}$ contain the first k standard basis vectors. Then we claim that $\mathbf{W}^* = \arg\max_{\mathbf{W}} \|\mathbf{\Sigma}\mathbf{W}\mathbf{W}^T\|_F^2$.

USEFUL OBSERVATIONS



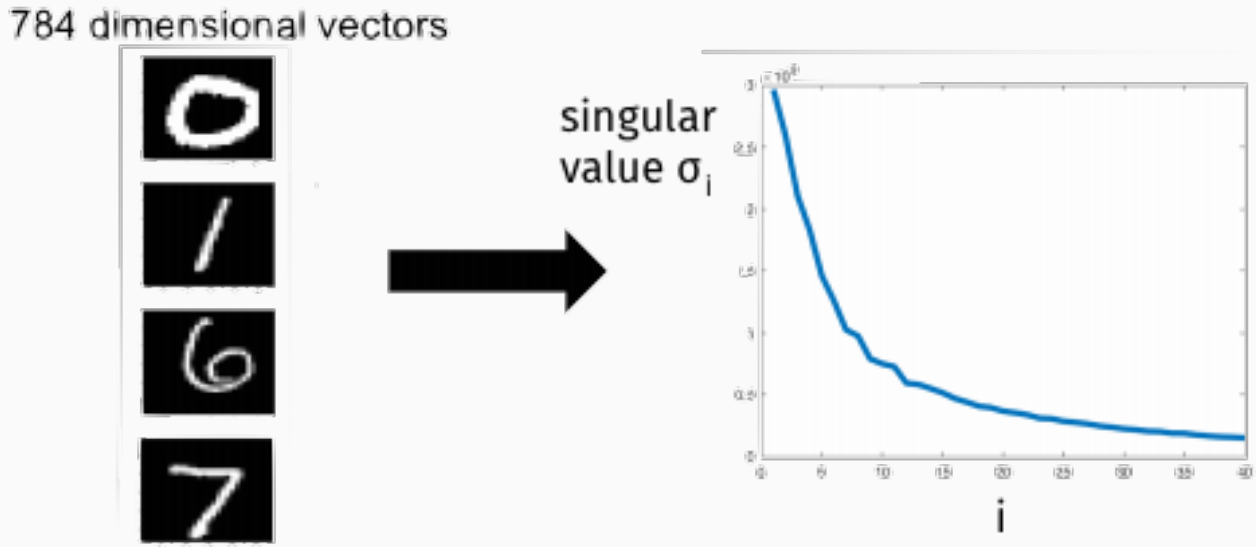
Observation: The optimal low-rank approximation error $E_k = \|\mathbf{X} - \mathbf{X}_k\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}_k\|_F^2$ can be written:

$$E_k = \sum_{i=k+1}^{d} \sigma_i^2.$$

Observation: The optimal low-rank approximation error $E_k = \|\mathbf{X} - \mathbf{X}_k\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}_k\|_F^2$ can be written:

$$E_k = \sum_{i=k+1}^{d} \sigma_i^2.$$

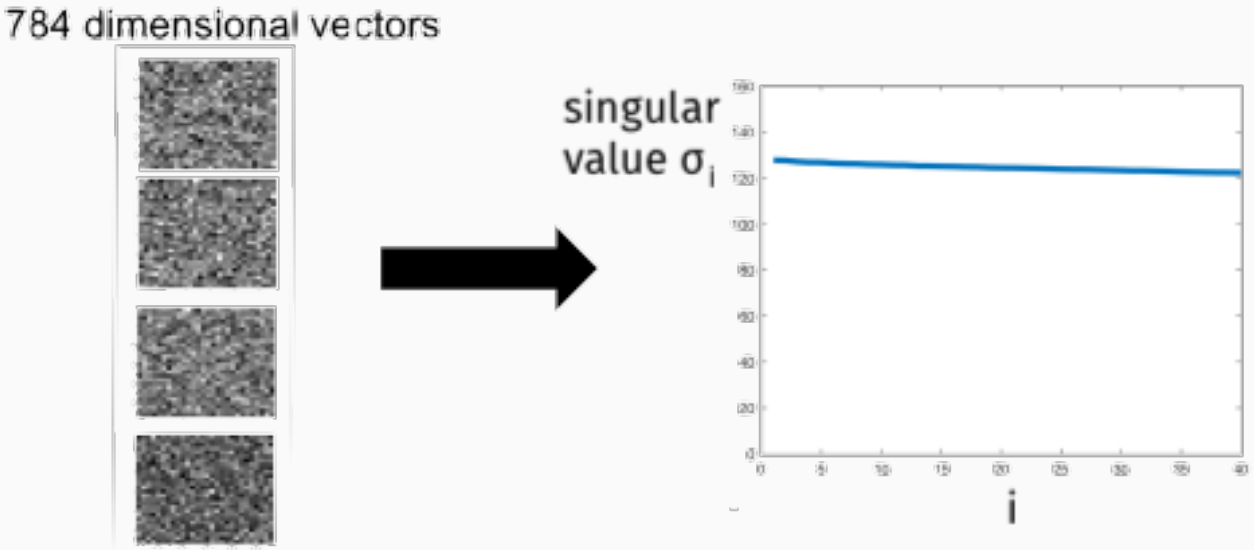
Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



Observation: The optimal low-rank approximation error $E_k = \|\mathbf{X} - \mathbf{X}_k\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}_k\|_F^2$ can be written:

$$E_k = \sum_{i=k+1}^{d} \sigma_i^2.$$

Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



Suffices to compute right singular vectors $\mathbf{V}$:

- Compute $\mathbf{X}^T\mathbf{X}$.
- Find eigendecomposition $\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T = \mathbf{X}^T\mathbf{X}$ using e.g. QR algorithm.
- Compute $\mathbf{L} = \mathbf{X}\mathbf{V}$. Set $\sigma_i = \|\mathbf{L}_i\|_2$ and $\mathbf{U}_i = \mathbf{L}_i / \|\mathbf{L}_i\|_2$.

Total runtime $\approx$

How to go faster?

- Compute approximate solution.
- Only compute top k singular vectors/values.
- Iterative algorithms achieve runtime $\approx O(ndk)$ vs. $O(nd^2)$ time.
  - **Krylov subspace methods** like the Lanczos method are most commonly used in practice.
  - **Power method** is the simplest Krylov subspace method, and still works very well.

POWER METHOD

Today: Consider simlest case when $k = 1$.

Goal: Find some $\mathbf{z} \approx \mathbf{v}_1$.

Input: $\mathbf{X} \in \mathbb{R}^{n \times d}$ with SVD $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.

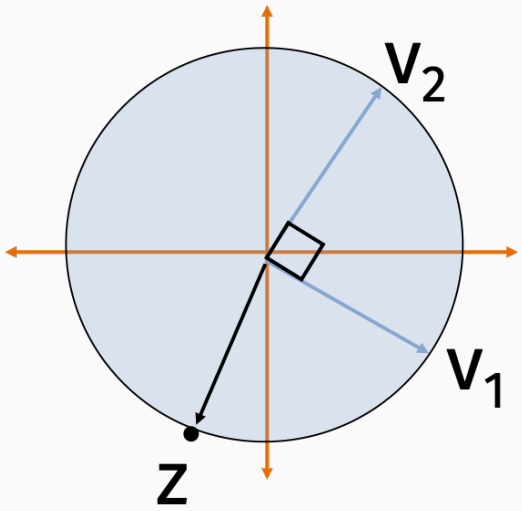
Power method:

- Choose $\mathbf{z}^{(0)}$ randomly. $\mathbf{z}_0 \sim \mathcal{N}(0, 1)$.
- $\mathbf{z}^{(0)} = \mathbf{z}^{(0)} / \|\mathbf{z}^{(0)}\|_2$
- For $i = 1, \ldots, T$
  - $\mathbf{z}^{(i)} = \mathbf{X}^T \cdot (\mathbf{X}\mathbf{z}^{(i-1)})$
  - $n_i = \|\mathbf{z}^{(i)}\|_2$
  - $\mathbf{z}^{(i)} = \mathbf{z}^{(i)} / n_i$

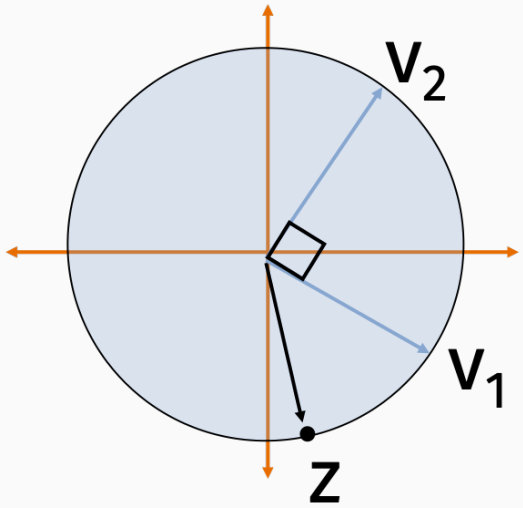
  Return $\mathbf{z}^{(T)}$

POWER METHOD INTUITION

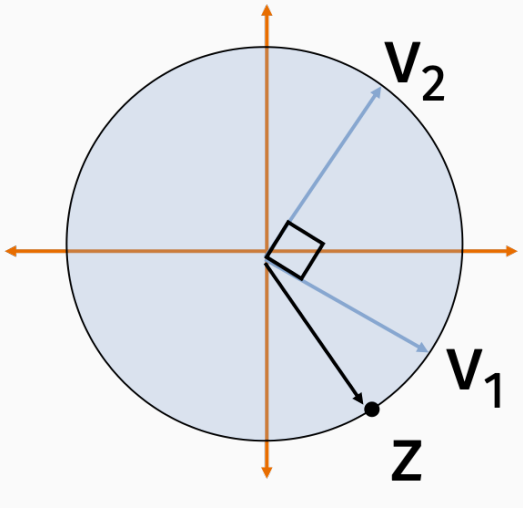
0 iterations



1 iterations



2 iterations



Theorem (Basic Power Method Convergence)

Let $\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$ be parameter capturing the "gap" between the first and second largest singular values. If Power Method is initialized with a random Gaussian vector then, with high probability, after $T = O\left(\frac{\log d/\epsilon}{\gamma}\right)$ steps, we have either:

$$\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2 \leq \epsilon \qquad \text{or} \qquad \|\mathbf{v}_1 - (-\mathbf{z}^{(T)})\|_2 \leq \epsilon.$$

Total runtime: $O\left(nd \cdot \frac{\log d/\epsilon}{\gamma}\right)$

Write $\mathbf{z}^{(i)}$ in the right singular vector basis:

$$\begin{aligned}
\mathbf{z}^{(0)} &= c_1^{(0)}\mathbf{v}_1 + c_2^{(0)}\mathbf{v}_2 + \ldots + c_d^{(0)}\mathbf{v}_d \\
\mathbf{z}^{(1)} &= c_1^{(1)}\mathbf{v}_1 + c_2^{(1)}\mathbf{v}_2 + \ldots + c_d^{(1)}\mathbf{v}_d \\
&\vdots \\
\mathbf{z}^{(i)} &= c_1^{(i)}\mathbf{v}_1 + c_2^{(i)}\mathbf{v}_2 + \ldots + c_d^{(i)}\mathbf{v}_d
\end{aligned}$$

Note: $[c_1^{(i)}, \ldots, c_d^{(i)}] = \mathbf{c}^{(i)} = \mathbf{V}^T\mathbf{z}^{(i)}$.

Also: Since $\mathbf{V}$ is orthogonal and $\|\mathbf{z}^{(i)}\|_2 = 1$, $\|\mathbf{c}^{(i)}\|_2^2 = 1$.

Claim: After update $\mathbf{z}^{(i)} = \frac{1}{n_i}\mathbf{X}^T\mathbf{X}\mathbf{z}^{(i-1)}$,

$$c_j^{(i)} = \frac{1}{n_i}\sigma_j^2 c_j^{(i-1)}$$

$$\mathbf{z}^{(i)} = \frac{1}{n_i}\left[c_1^{(i-1)}\sigma_1^2 \cdot \mathbf{v}_1 + c_2^{(i-1)}\sigma_2^2 \cdot \mathbf{v}_2 + \ldots + c_d^{(i-1)}\sigma_d^2 \cdot \mathbf{v}_d\right]$$

Equivalently: $\mathbf{c}^{(i)} = \frac{1}{n_i}\mathbf{\Sigma}^2\mathbf{c}^{(i-1)}$.

Claim: After T updates:

$$\mathbf{z}^{(T)} = \frac{1}{\prod_{i=1}^{T} n_i}\left[c_1^{(0)}\sigma_1^{2T}\cdot\mathbf{v}_1 + c_2^{(0)}\sigma_2^{2T}\cdot\mathbf{v}_2 + \ldots + c_d^{(0)}\sigma_d^{2T}\cdot\mathbf{v}_d\right]$$

Let $\alpha_j = \frac{1}{\prod_{i=1}^{T} n_i} c_j^{(0)}\sigma_j^{2T}$. **Goal:** Show that $\alpha_j \ll \alpha_1$ for all $j \neq 1$.

Since $\mathbf{z}^{(T)}$ is a unit vector, $\sum_{i=1}^{d} \alpha_i^2 = 1$. So $|\alpha_1| \leq 1$.

If we can prove that $\left|\frac{\alpha_j}{\alpha_1}\right| \leq \sqrt{\frac{\epsilon}{2d}}$ then we will have that $\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2^2 \leq \epsilon$.

$$\alpha_j^2 \leq \alpha_1^2 \cdot \frac{\epsilon}{2d}$$

$$1 = \alpha_1^2 + \sum_{j=2}^{d} \alpha_d^2 \leq \alpha_1^2 + \frac{\epsilon}{2}$$

$$\alpha_1^2 \geq 1 - \frac{\epsilon}{2}$$

$$|\alpha_1| \geq 1 - \frac{\epsilon}{2}$$

$$\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2^2 = 2 - 2\langle \mathbf{v}_1, \mathbf{z}^{(T)} \rangle \leq \epsilon$$

Let's see how many steps T it takes to ensure that $\left|\frac{\alpha_j}{\alpha_1}\right| \leq \sqrt{\frac{\epsilon}{2d}}$ where $\alpha_j = \frac{1}{\prod_{i=1}^{T} n_i} c_j^{(0)} \sigma_j^{2T}$. Answer will depend on $\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$. **Assumption:** Starting coefficient on first eigenvector is not too small:

$$\left|c_1^{(0)}\right| \geq O\left(\frac{1}{\sqrt{d}}\right).$$

We will prove shortly that it holds with probability 99/100.

$$\frac{|\alpha_j|}{|\alpha_1|} = \frac{\sigma_j^{2T}}{\sigma_1^{2T}} \cdot \frac{|c_j^{(0)}|}{|c_1^{(0)}|} \leq$$

Need to set $T =$

Need to prove: Starting coefficient on first eigenvector is not too small. I.e., with probability 99/100,

$$\left|c_1^{(0)}\right| \geq O\left(\frac{1}{\sqrt{d}}\right).$$

Prove using Gaussian anti-concentration. First use rotational invariance of Gaussian:

$$\mathbf{c}^{(0)} = \frac{\mathbf{V}^T\mathbf{z}^{(0)}}{\|\mathbf{z}^{(0)}\|_2} = \frac{\mathbf{V}^T\mathbf{z}^{(0)}}{\|\mathbf{V}^T\mathbf{z}^{(0)}\|_2} \sim \frac{\mathbf{g}}{\|\mathbf{g}\|_2},$$

where $\mathbf{g} \sim \mathcal{N}(0,1)^d$.

Need to show that with high probability, first entry of $\frac{\mathbf{g}}{\|\mathbf{g}\|_2} \geq c \cdot \frac{1}{\sqrt{d}}$.

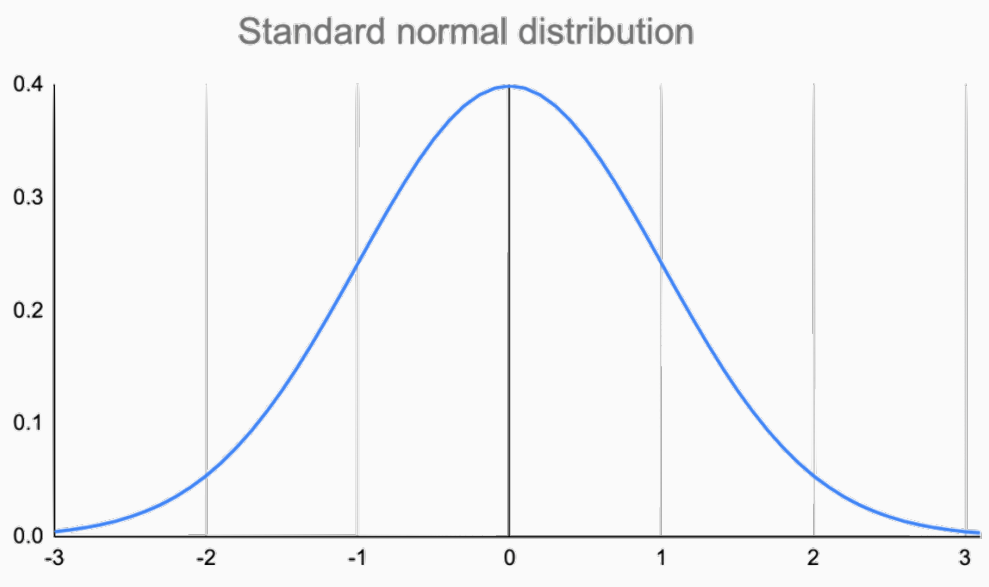
Part 1: With super high probability (e.g. 99/100),

$$\|\mathbf{g}\|_2^2 \leq$$

Need to show that with high probability, the magnitude of the first entry of $\mathbf{g} \geq c$ for a constant c. Think e.g. $c = 1/100$.

Part 2: With probablility $1 - O(\alpha)$,

$$|g_1| \geq \alpha.$$



Theorem (Basic Power Method Convergence)

Let $\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$ be parameter capturing the "gap" between the first and second largest singular values. If Power Method is initialized with a random Gaussian vector then, with high probability, after $T = O\left(\frac{\log d/\epsilon}{\gamma}\right)$ steps, we have either:

$$\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2 \leq \epsilon \qquad \text{or} \qquad \|\mathbf{v}_1 - (-\mathbf{z}^{(T)})\|_2 \leq \epsilon.$$

The method truly won't converge if γ is very small. Consider extreme case when $\gamma = 0$.

$$\mathbf{z}^{(T)} = \frac{1}{\prod_{i=1}^{T} n_i}\left[c_1^{(0)}\sigma_1^{2T} \cdot \mathbf{v}_1 + c_2^{(0)}\sigma_2^{2T} \cdot \mathbf{v}_2 + \ldots + c_d^{(0)}\sigma_d^{2T} \cdot \mathbf{v}_d\right]$$

Theorem (Gapless Power Method Convergence)

If Power Method is initialized with a random Gaussian vector then, with high probability, after $T = O\left(\frac{\log d/\epsilon}{\epsilon}\right)$ *steps, we obtain a* $\mathbf{z}$ *satisfying:*

$$\|\mathbf{X} - \mathbf{X}\mathbf{z}\mathbf{z}^T\|_F^2 \leq (1+\epsilon)\|\mathbf{X} - \mathbf{X}\mathbf{v}_1\mathbf{v}_1^T\|_F^2$$

Intuition: For a good low-rank approximation, we don't actually need to converge to $\mathbf{v}_1$ if σ_1 and σ_2 are the same or very close. Would suffice to return either $\mathbf{v}_1$ or $\mathbf{v}_2$, or some linear combination of the two.

GENERALIZATIONS TO LARGER k

- Block Power Method aka Simultaneous Iteration aka Subspace Iteration aka Orthogonal Iteration

Power method:

- Choose $\mathsf{G} \in \mathbb{R}^{d \times k}$ be a random Gaussian matrix.
- $\mathsf{Z}_0 = \mathsf{orth}(\mathsf{G})$.
- For $i = 1, \ldots, T$
  - $\mathsf{Z}^{(i)} = \mathsf{X}^T \cdot (\mathsf{X}\mathsf{Z}^{(i-1)})$
  - $\mathsf{Z}^{(i)} = \mathsf{orth}(\mathsf{Z}^{(i)})$

  Return $\mathsf{Z}^{(T)}$

Guarantee: After $O\left(\frac{\log d/\epsilon}{\epsilon}\right)$ iterations:

$$\|\mathsf{X} - \mathsf{X}\mathsf{Z}\mathsf{Z}^T\|_F^2 \leq (1+\epsilon)\|\mathsf{X} - \mathsf{X}\mathsf{V}_\mathsf{k}\mathsf{V}_\mathsf{k}{}^T\|_F^2.$$

Runtime: $O(\mathsf{nnz}(\mathsf{X}) \cdot k \cdot T) \leq O(ndk \cdot T)$.

Possible to “accelerate” these methods.

Convergence Guarantee: $T = O\left(\frac{\log d/\epsilon}{\sqrt{\epsilon}}\right)$ iterations to obtain a nearly optimal low-rank approximation:

$$\|\mathbf{X} - \mathbf{X}\mathbf{Z}\mathbf{Z}^T\|_F^2 \leq (1+\epsilon)\|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k{}^T\|_F^2.$$

For a normalizing constant c, power method returns:

$$\mathbf{z}^{(q)} = c \cdot \left(\mathbf{X}^T\mathbf{X}\right)^q \cdot \mathbf{g}$$

Along the way we computed:

$$\mathcal{K}_q = \left[\mathbf{g}, \left(\mathbf{X}^T\mathbf{X}\right) \cdot \mathbf{g}, \left(\mathbf{X}^T\mathbf{X}\right)^2 \cdot \mathbf{g}, \ldots, \left(\mathbf{X}^T\mathbf{X}\right)^q \cdot \mathbf{g}\right]$$

$\mathcal{K}$ is called the Krylov subspace of degree q.

Idea behind Krlyov methods: Don't throw away everything before $\left(\mathbf{X}^T\mathbf{X}\right)^q \cdot \mathbf{g}$.

Want to find $\mathbf{v}$, which minimizes $\|\mathbf{X} - \mathbf{X}\mathbf{v}\mathbf{v}^T\|_F^2$.

Lanczos method:

- Let $\mathbf{Q} \in \mathbb{R}^{d \times k}$ be an orthonormal span for the vectors in $\mathcal{K}$.
- Solve $\min_{\mathbf{v}=\mathbf{Q}\mathbf{w}} \|\mathbf{X} - \mathbf{X}\mathbf{v}\mathbf{v}^T\|_F^2$.
  - Find best vector in the Krylov subspace, instead of just using last vector.
  - Can be done in $O\left(ndk + dk^2\right)$ time.
  - What you're using when you run `svds` or `eigs` in MATLAB or Python.

For a degree t polynomial p, let $\mathbf{v}_p = \frac{p(\mathbf{X}^T\mathbf{X})\mathbf{g}}{\|p(\mathbf{X}^T\mathbf{X})\mathbf{g}\|_2}$. We always have that $\mathbf{v}_p \in \mathcal{K}_t$, the Krylov subspace contructed with t iterations.

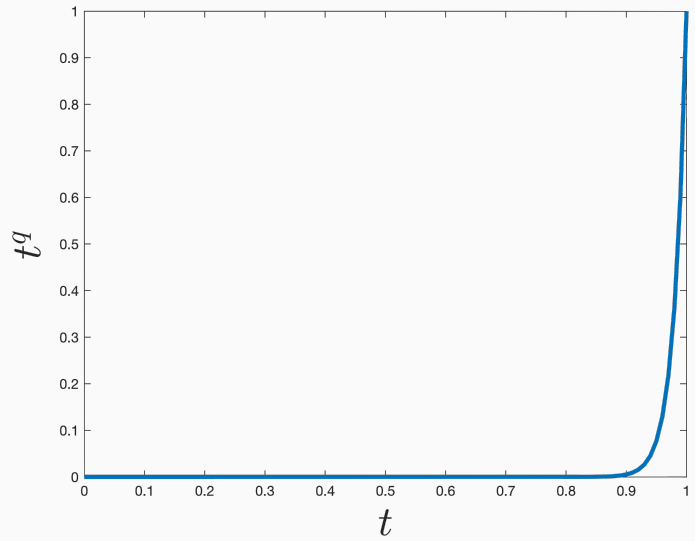
Power method returns:

$$\mathbf{v}_p \text{ where } p = x^q \text{ for } q = 2T.$$

Lanczos method returns $\mathbf{v}_{p^*}$ where:

$$p^* = \underset{\text{degree } t \; p}{\arg\min} \|\mathbf{X} - \mathbf{X}\mathbf{v}_p\mathbf{v}_p^T\|_F^2.$$

Claim: There is a $t = O\left(\sqrt{q \log \frac{1}{\Delta}}\right)$ degree polynomial $\hat{p}$ approximating $\mathbf{x}^q$ up to error Δ on $[0, \sigma_1^2]$.



$$\|\mathbf{X} - \mathbf{X}\mathbf{v}_{p^*}\mathbf{v}_{p^*}^T\|_F^2 \leq \|\mathbf{X} - \mathbf{X}\mathbf{v}_{\hat{p}}\mathbf{v}_{\hat{p}}^T\|_F^2 \approx \|\mathbf{X} - \mathbf{X}\mathbf{v}_{X^q}\mathbf{v}_{X^q}^T\|_F^2 \approx \|\mathbf{X} - \mathbf{X}\mathbf{v}_1\mathbf{v}_1^T\|_F^2$$

Runtime: $O\left(\frac{\log(d/\epsilon)}{\sqrt{\epsilon}} \cdot \text{nnz}(\mathbf{X})\right)$ vs. $O\left(\frac{\log(d/\epsilon)}{\epsilon} \cdot \text{nnz}(\mathbf{X})\right)$

GENERALIZATIONS TO LARGER k

- Block Krylov methods

- Let $\mathbf{G} \in \mathbb{R}^{d \times k}$ be a random Gaussian matrix.
- $\mathcal{K}_q = \left[\mathbf{G}, (\mathbf{X}^T\mathbf{X}) \cdot \mathbf{G}, (\mathbf{X}^T\mathbf{X})^2 \cdot \mathbf{G}, \ldots, (\mathbf{X}^T\mathbf{X})^q \cdot \mathbf{G}\right]$

Runtime: $O\left(\mathsf{nnz}(\mathbf{X}) \cdot k \cdot \frac{\log d/\epsilon}{\sqrt{\epsilon}}\right)$ to obtain a nearly optimal low-rank approximation.