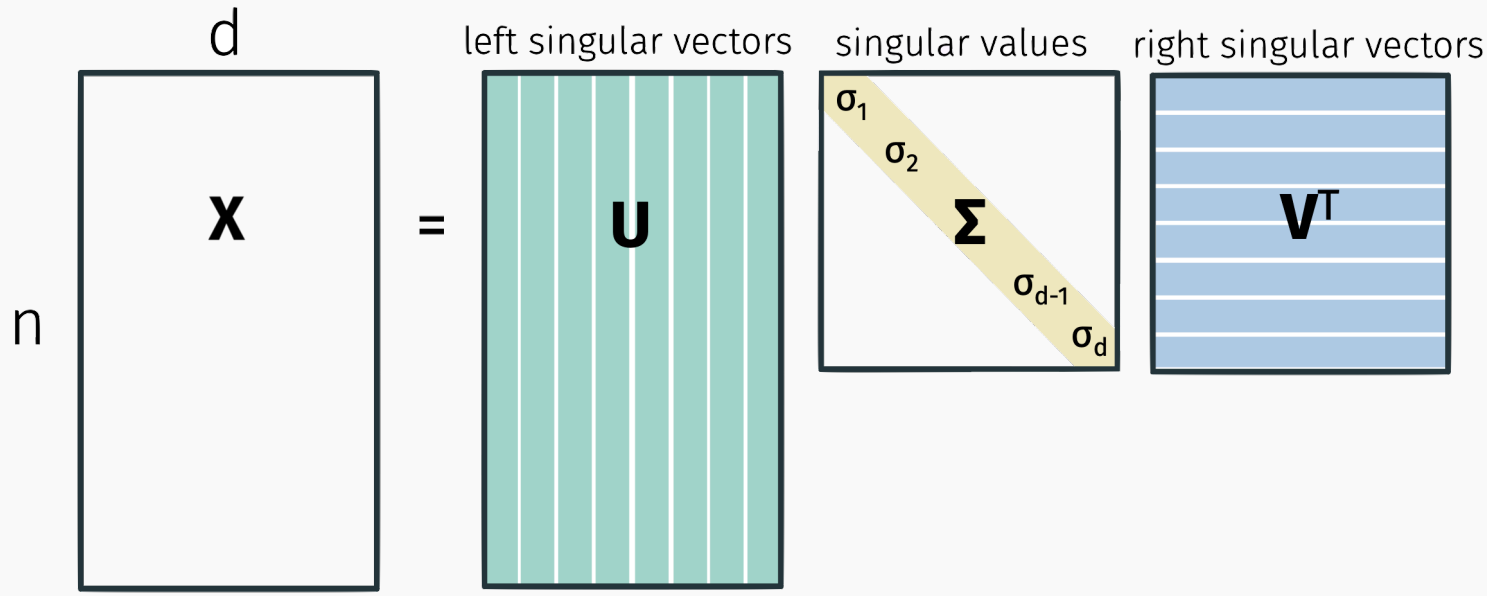
CS-GY 6763: Lecture 11
Power Method, Krylov Subspace Methods, Spectral Clustering

NYU Tandon School of Engineering, Prof. Christopher Musco

SINGULAR VALUE DECOMPOSITION

One of the most fundamental results in linear algebra.

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

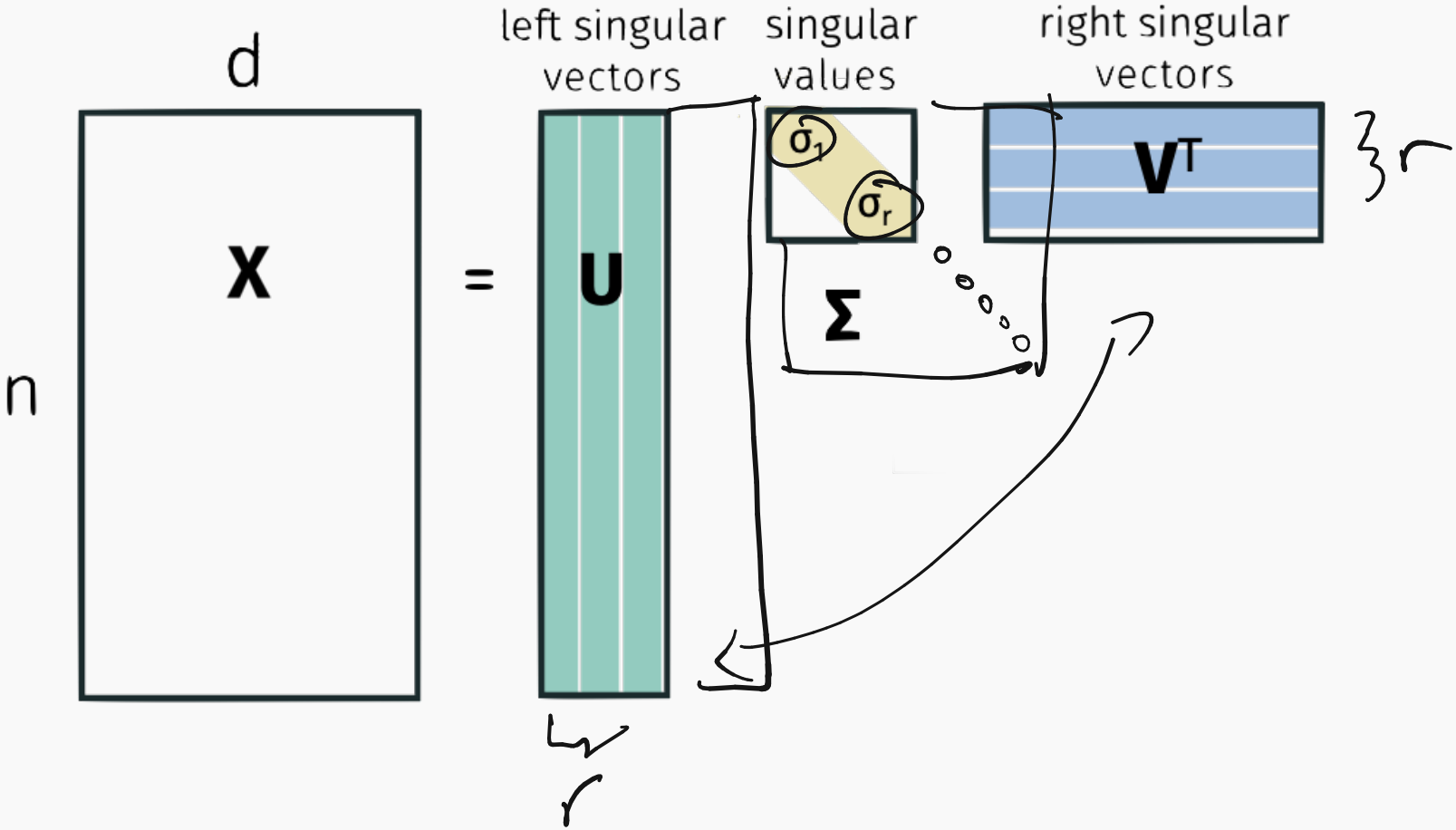


Where $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}$, and $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_d \geq 0$.

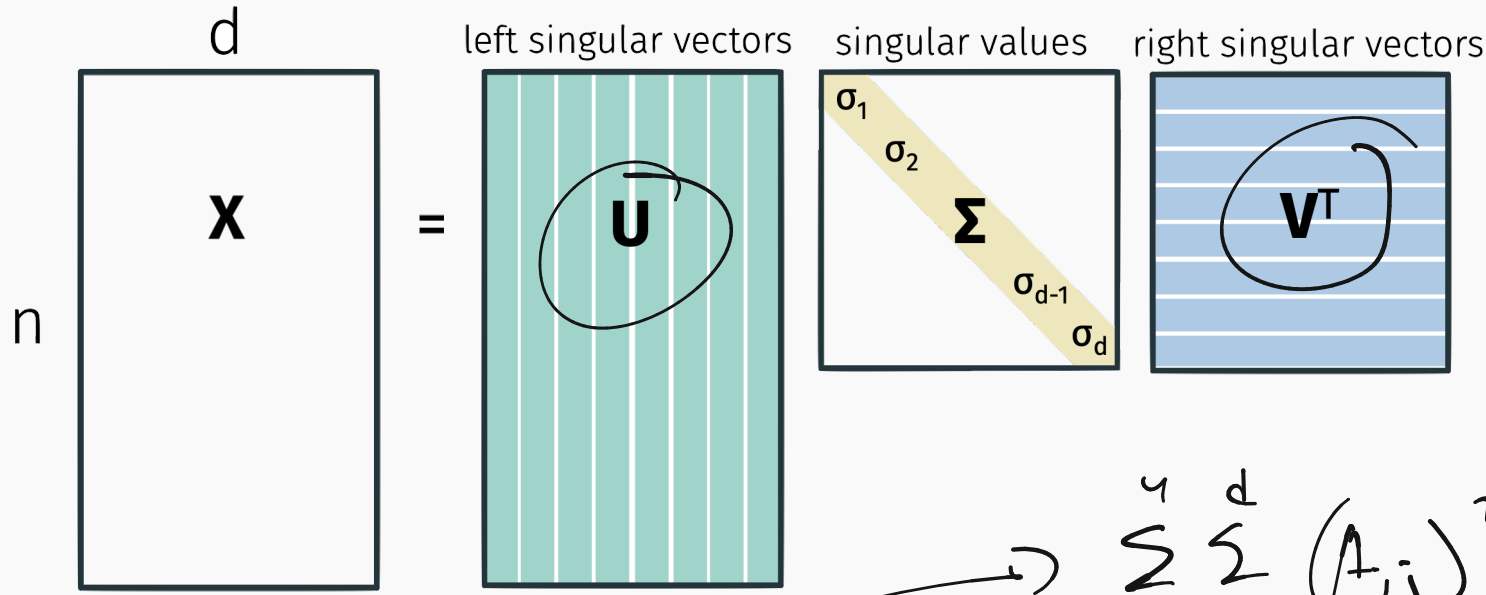
Singular values are unique. Factors are not. E.g. would still get a valid SVD by multiplying both i^{th} column of $\mathbf{V}$ and $\mathbf{U}$ by -1.

IMPORTANT NOTE FOR PROBLEM SET

If $\mathbf{X}$ has rank $r \leq \min(n, d)$ it only have r non-zero singular values. Some software packages will still return a full size $\mathbf{U}$ and $\mathbf{V}$ matrix.



OTHER THINGS TO NOTE



$$\sum_{i=1}^{n}\sum_{j=1}^{d}(A_{ij})^2$$

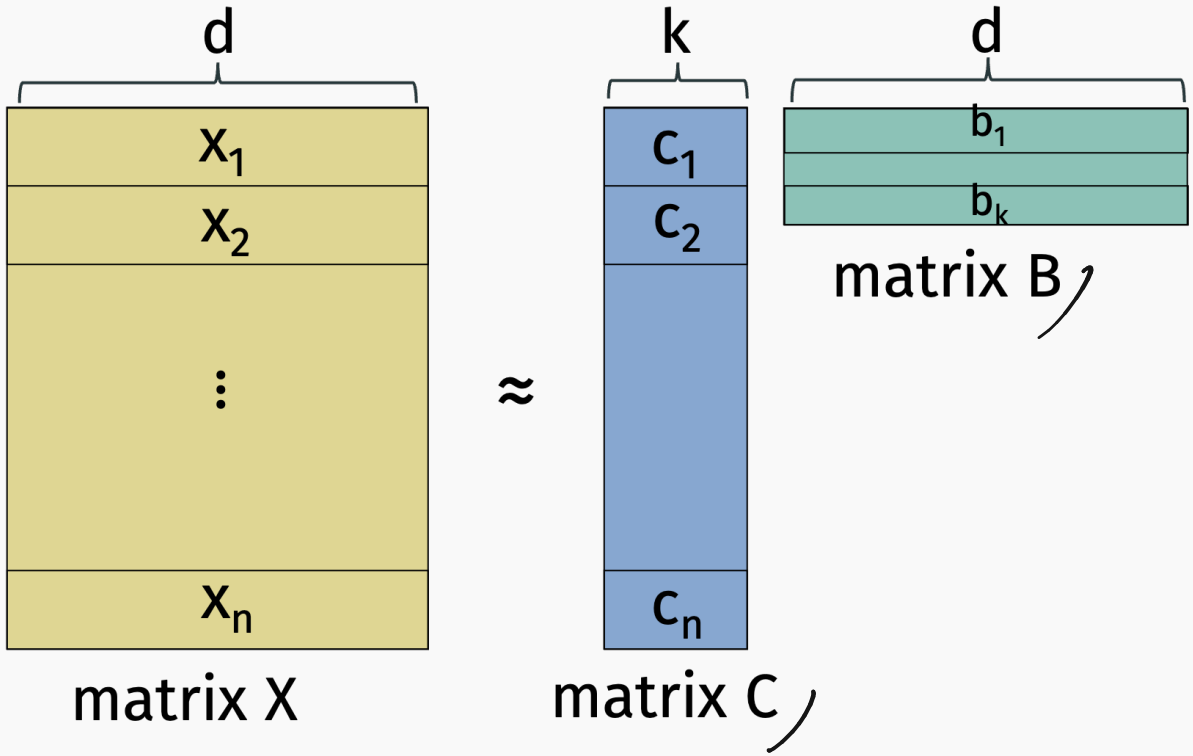
$$\|X\|_F^2 = \sum_{i=1}^{d}\sigma_i^2$$

$$\max_w \frac{\|Xw\|_2}{\|w\|_2}$$

$$\|X\|_2 = \sigma_1$$

LOW-RANK APPROXIMATION

Approximate $\mathbf{X}$ as the product of two rank k matrices:



Typically choose $\mathbf{C}$ and $\mathbf{B}$ to minimize: → rank k

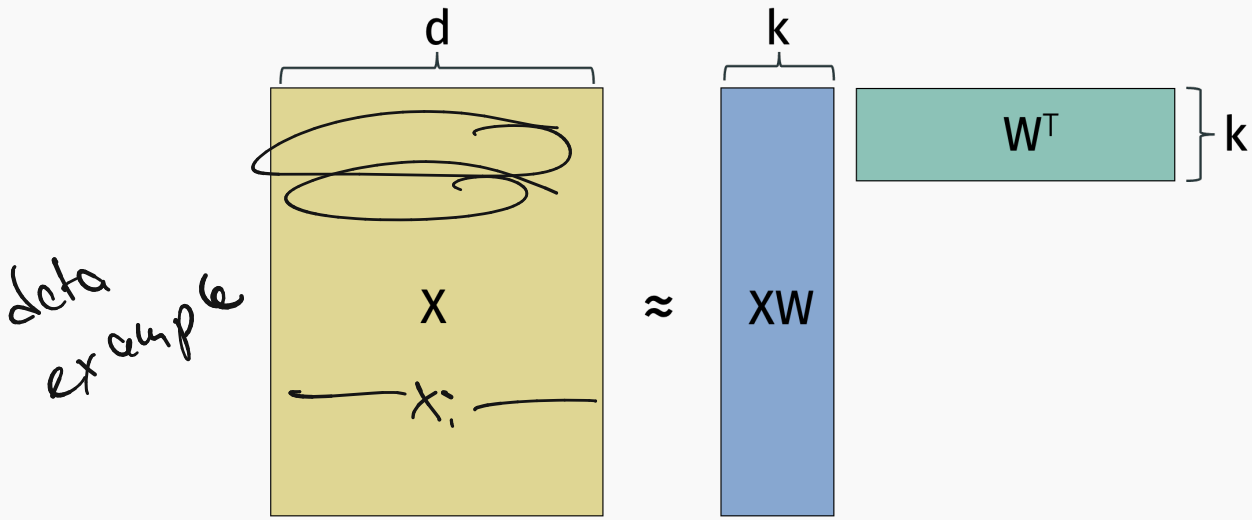
$$\min_{\mathbf{B},\mathbf{C}} \|\mathbf{X} - \mathbf{CB}\|$$

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

EQUIVALENT FORMULATION

When measuring error with the Frobenius norm (or spectral norm) it suffices to find $d \times k$ orthogonal matrix $\mathbf{W}$ minimizing:

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



I.e., best low-rank approximation projects $\mathbf{X}$'s rows to a lower dimensional space.

EQUIVALENT FORMULATION

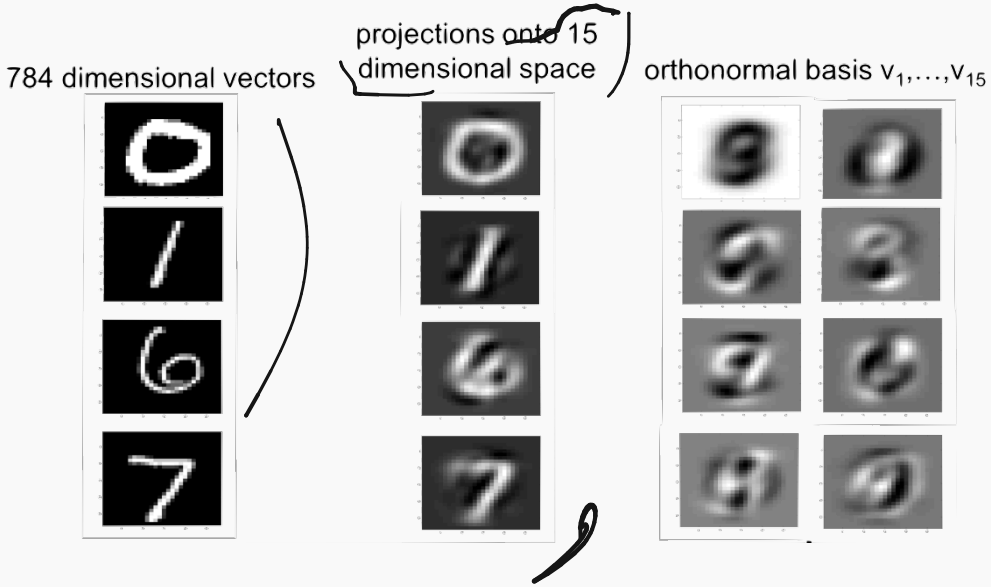
Alternatively, suffices to find $n \times k$ orthogonal matrix $\mathbf{Z}$ minimizing:

$$\|\mathbf{X} - \mathbf{Z}\mathbf{Z}^T\mathbf{X}\|_F$$

C B

WHY IS DATA LOW-RANK

Row redundancy: If a data set only had k unique data points, it would be exactly rank k. If it has k "clusters" of data points (e.g. the 10 digits) it's often very close to rank k.



WHY IS DATA LOW-RANK

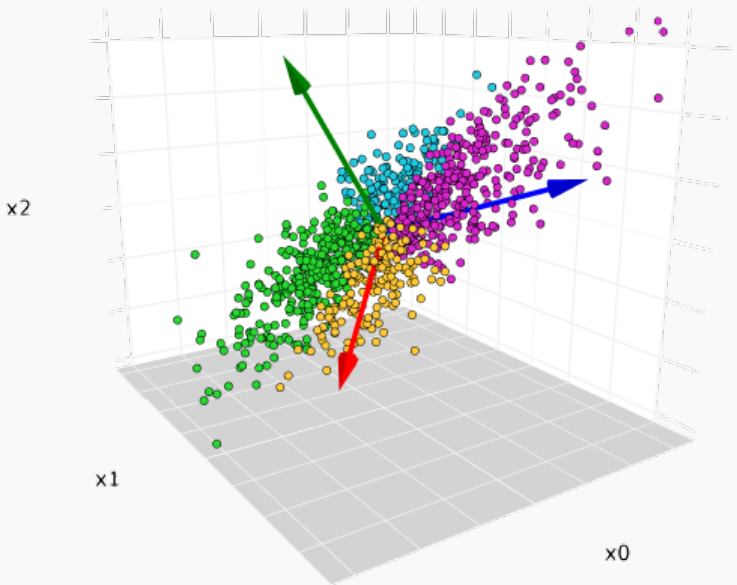
Column redundancy: Colinearity/correlation of data features leads to a low-rank data matrix. $d-1$

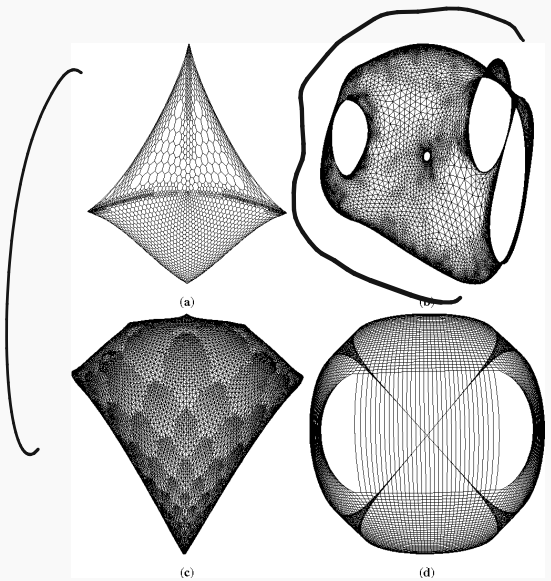
	bedrooms	bathrooms	sq.ft.	floors	list price	sale price
home 1	2	2	1800	2	200,000	195,000
home 2	4	2.5	2700	1	300,000	310,000
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
home n	5	3.5	3600	3	450,000	450,000

APPLICATIONS OF LOW-RANK APPROXIMATION

Fact that $\|\mathbf{x}_i - \mathbf{x}_j\|_2 \approx \|\mathbf{x}_i^T \mathbf{W}\mathbf{W}^T - \mathbf{x}_j^T \mathbf{W}\mathbf{W}^T\|_2 = \|\mathbf{c}_i - \mathbf{c}_i\|_2$ leads to lots of applications.

- Data compression. E.g. used in state-of-the-art data dependent methods for nearest neighbor search.
- Data visualization when $k = 2$ or 3.

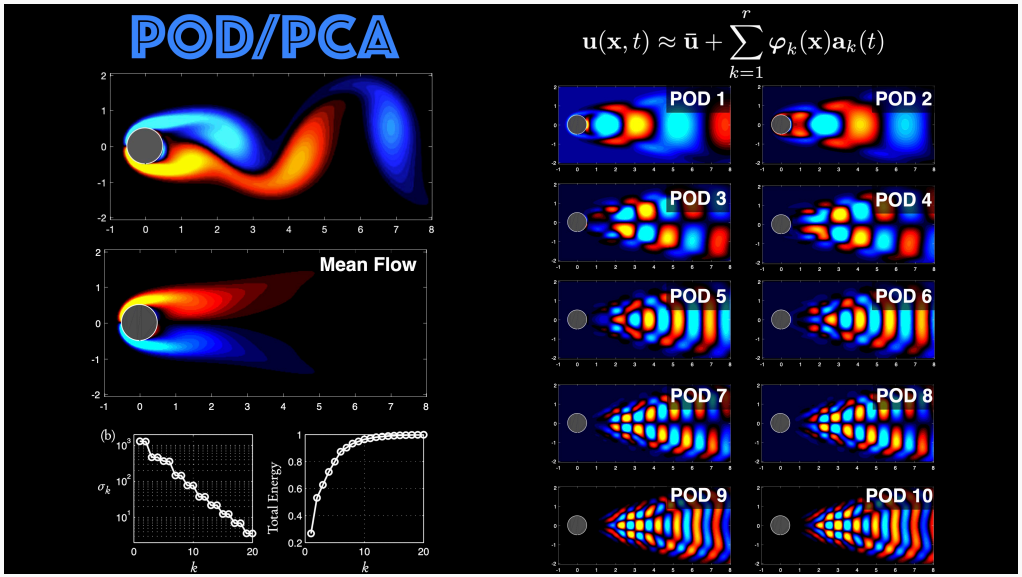




- Data embeddings (e.g. word2vec, node2vec).

APPLICATIONS OF LOW-RANK APPROXIMATION

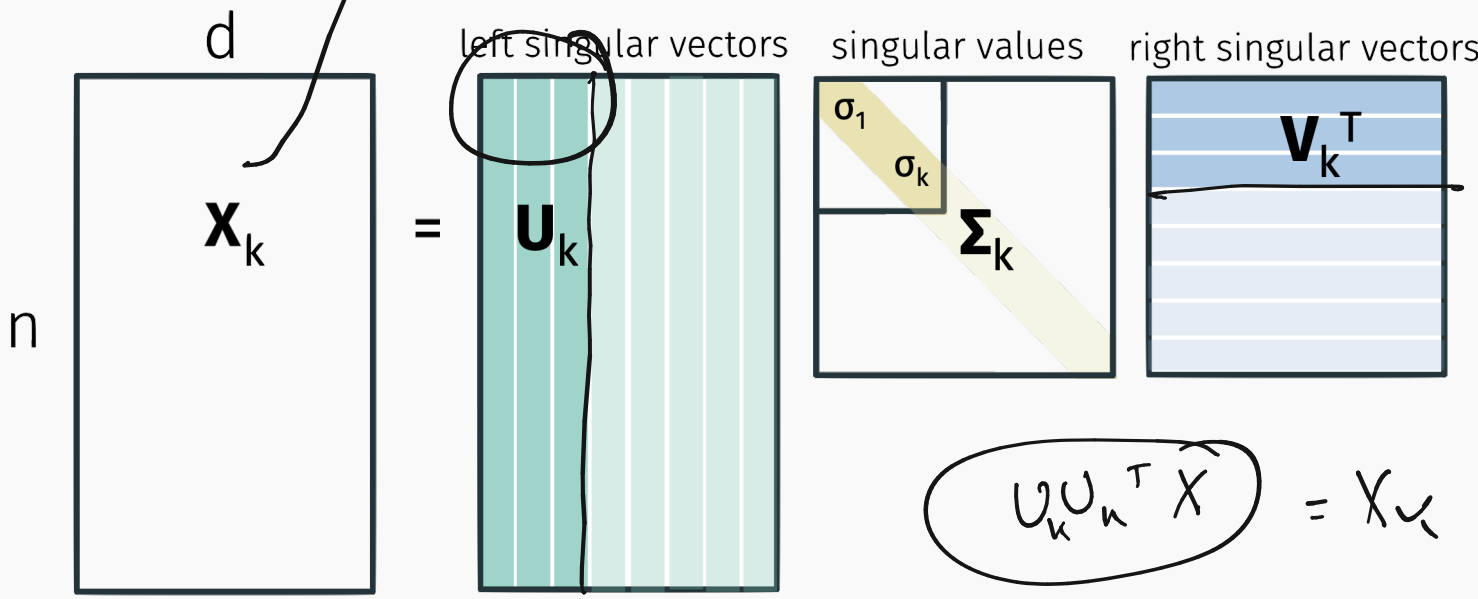
- Reduced order modeling for solving physical equations.



- Constructing preconditioners in optimization.
- Noisy triangulation (on problem set).

Key result: Can find the best projection from the singular value decomposition.

optimal rank k approx. of X



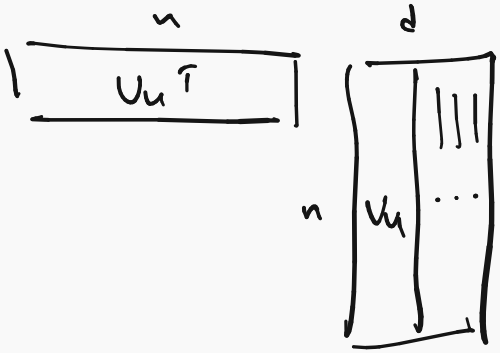
$U_k U_k^T X = X_k = U_k \Sigma_k V_k^T$

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

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

OPTIMAL LOW-RANK APPROXIMATION

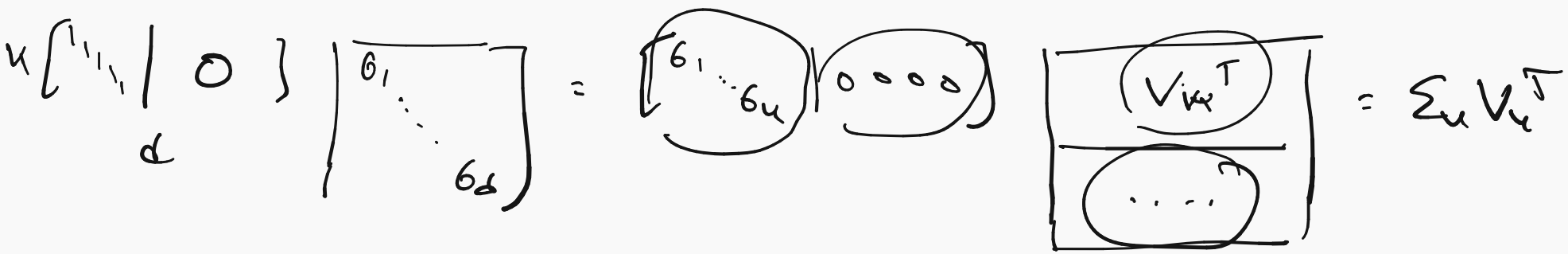
Claim: $X_k = U_k \Sigma_k V_k^T = U_k U_k^T X = X V_k V_k^T$.



$U_k U_k^T X = U_k U_k^T U \Sigma V^T = U_k \Sigma_k V_k^T$

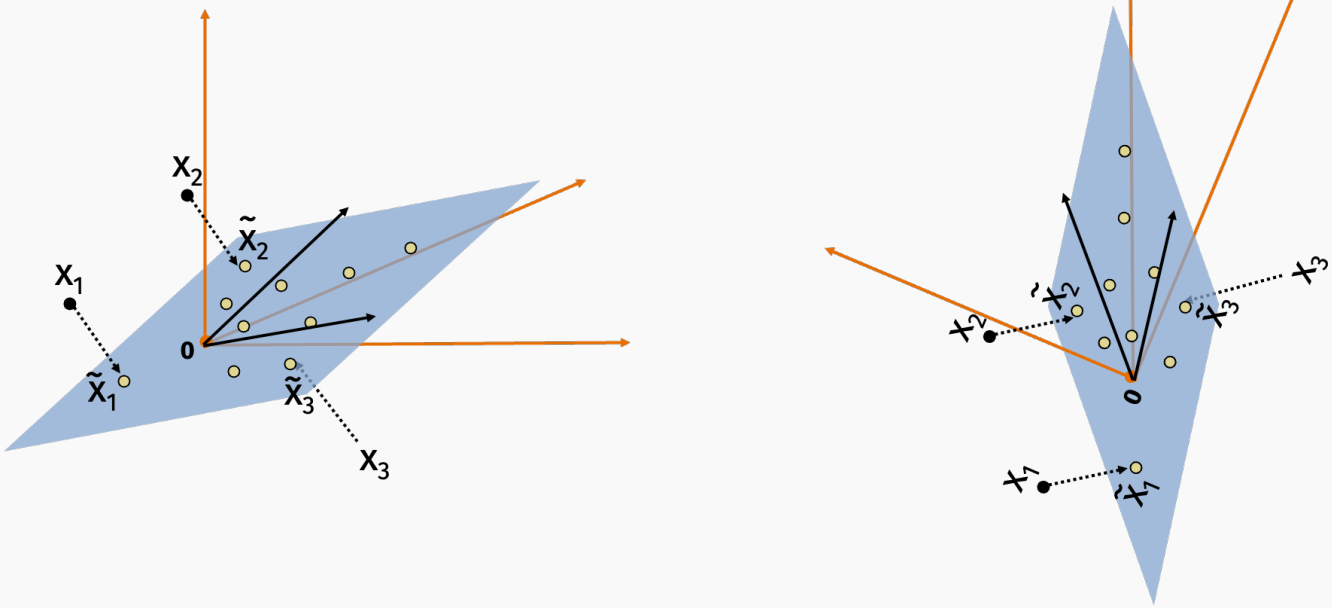
(want)

Suffices to show $U_k^T U \Sigma V^T = \Sigma_k V_k^T$



Claim 1:

$$\arg\min_{\text{rank } k\ \mathbf{B}} \|\mathbf{X} - \mathbf{B}\|_F^2 = \left[\arg\min_{\text{rank } k\mathbf{B}} \|\mathbf{U\Sigma} - \mathbf{B}\|_F^2\right] \cdot \mathbf{V}^T$$



Claim 2:

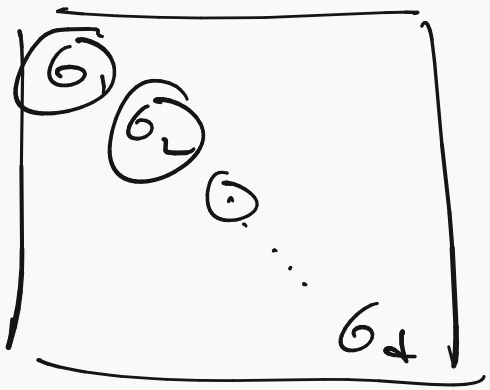
$$\arg\min_{\text{rank } k \; B} \|U\Sigma - B\|_F^2 = \arg\min_{\text{rank } k \; B} \|\Sigma - U^T B\|_F^2$$

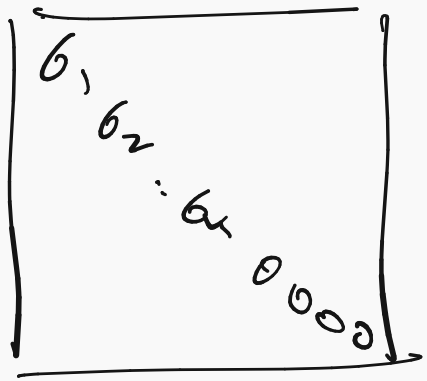
$U^T(U\Sigma - B) = \Sigma - U^T B$

Choose B so that $U^T B$ is an optimal rank k approximation of Σ. I.e., Σ_k.

$U^T B$

$$U^T B = \begin{bmatrix} \Sigma_k & \\ & 0 \end{bmatrix}$$

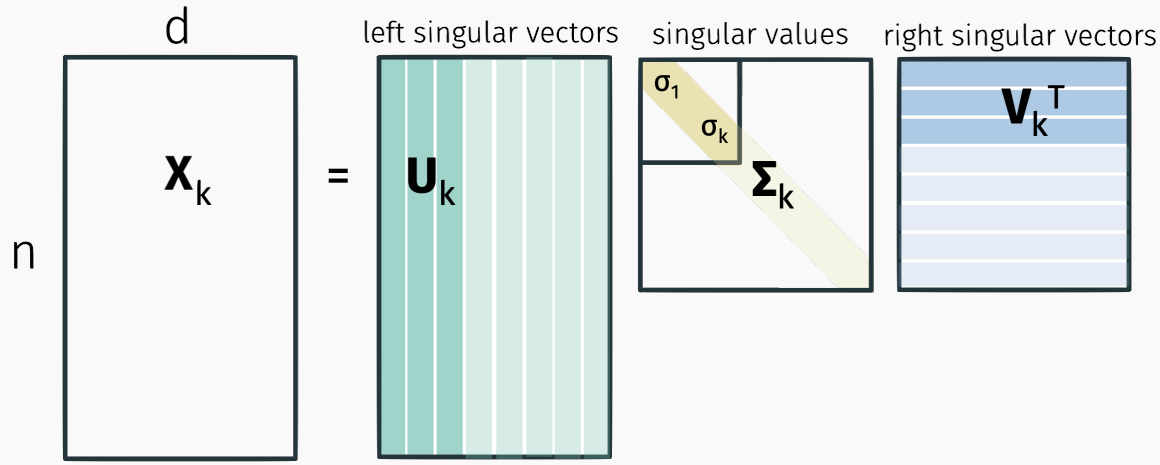




$$B = U \begin{bmatrix} \Sigma_k & \\ & 0 \end{bmatrix}$$

$$U \begin{bmatrix} \Sigma_k & \\ & 0 \end{bmatrix} V^T = U_k \Sigma_k V_k^T$$

USEFUL OBSERVATIONS



Observation 1:

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

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

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

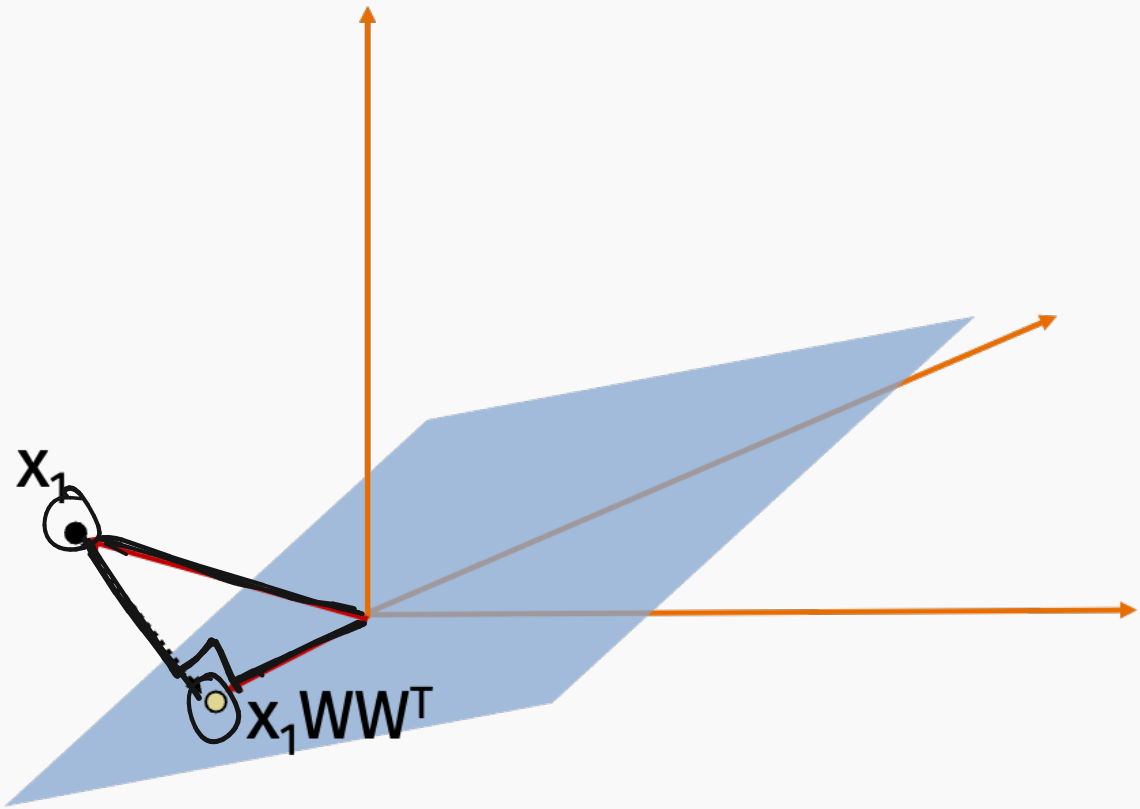
This is often the perspective people take when thinking about Principal Component Analysis.

USEFUL OBSERVATIONS

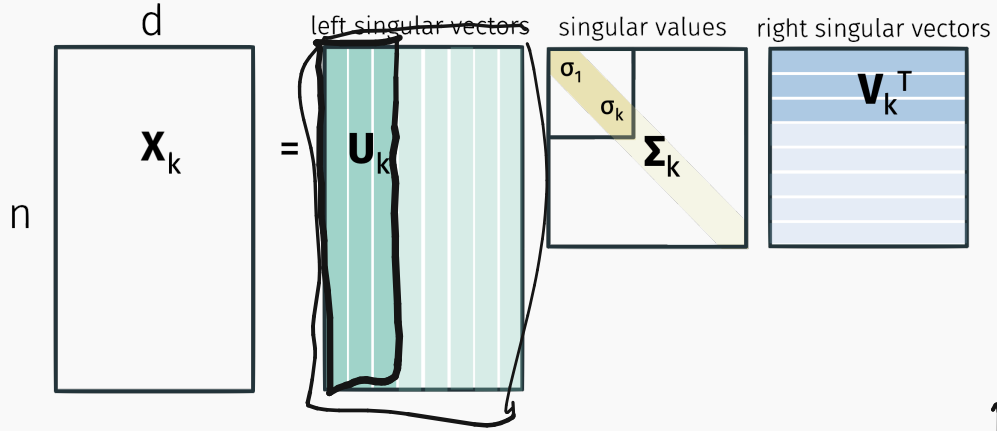
Claim:

$$\sum_{i=1}^{n} \|x_i - x_i WW^T\|_2^2$$

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



USEFUL OBSERVATIONS



$n \, d$

$\frac{\|X_k\|_F^2}{\|X\|_F^2} \geq \cdot \, q$

Observation 2: 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:

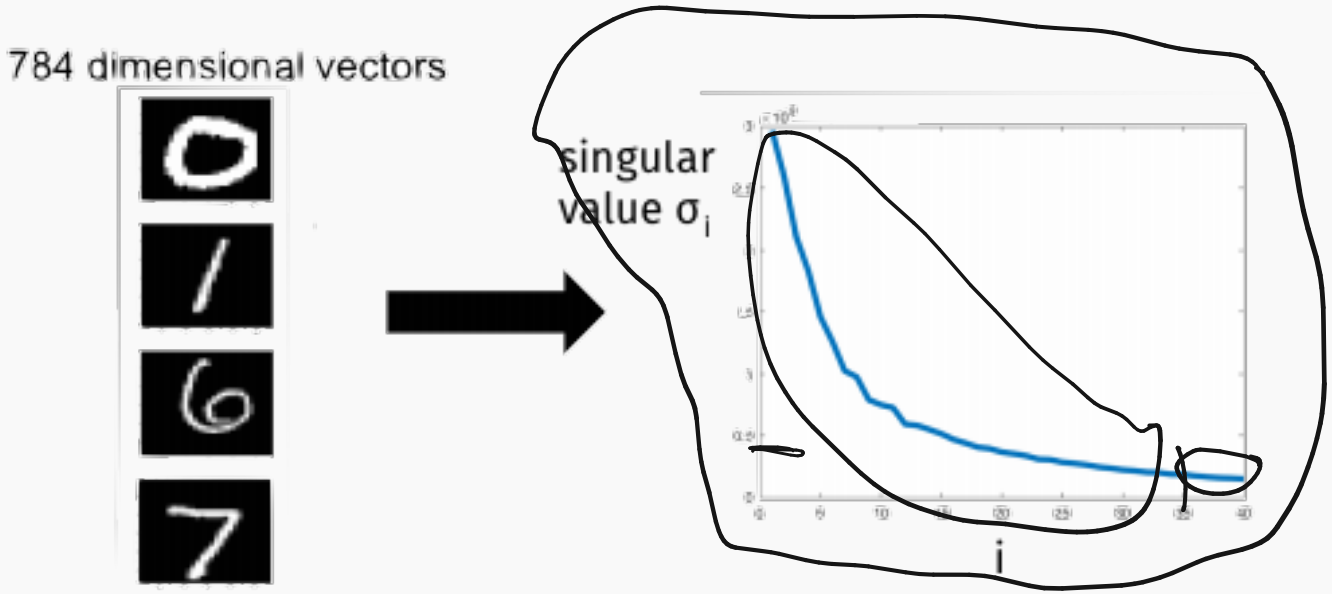
$$E_k = \sum_{i=k+1}^{d} \sigma_i^2 .$$

$\sum_{i=1}^{d} \sigma_i^2 - \sum_{i=1}^{k} \sigma_i^2$

Observation 2: 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:

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

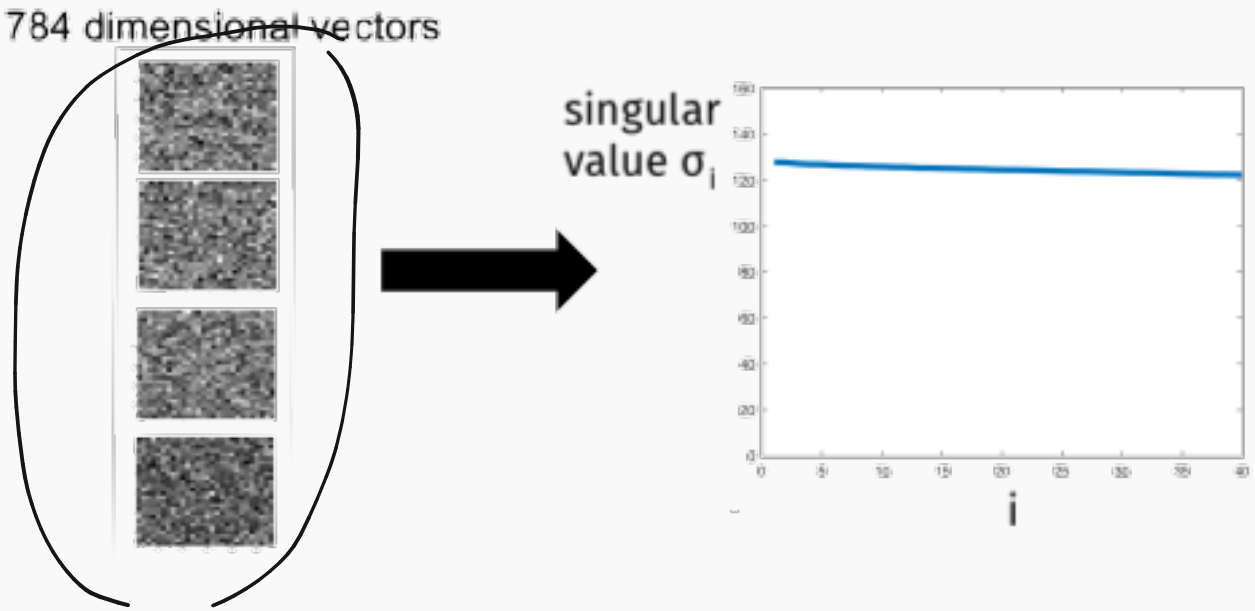
Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



Observation 2: 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:

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

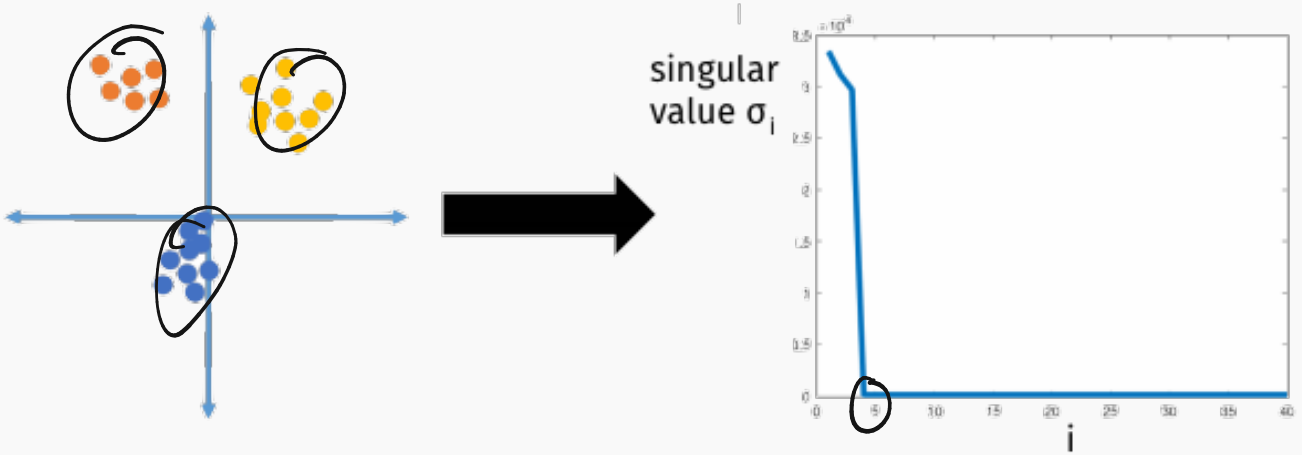
Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



Observation 2: 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:



COMPUTING THE SVD

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 $\mathbf{U}_i = \mathbf{L}_i / \|\mathbf{L}_i\|_2$.

$O(d^3)$

$X = U\Sigma V^T \qquad XV = U\Sigma$

Total runtime $\approx$ $O(nd^2)$

COMPUTING THE SVD (FASTER)

How to go faster?

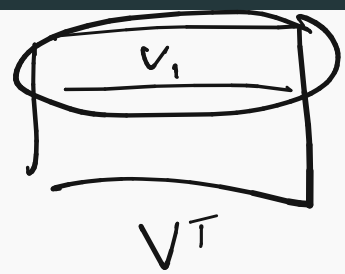
- Compute approximate solution.
- Only compute top k singular vectors/values.
- Iterative algorithms achieve runtime $\approx O(ndk)$ vs. $O(nd^2)$ time.
  - **Krylov subspace methods** like the Lanczos method are most commonly used in practice.
  - **Power method** is the simplest Krylov subspace method,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

POWER METHOD

Today: What about when $k = 1$?

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

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



$\|z - v_1\|_2 < \varepsilon$

Each iteration costs $O(nd)$

Power method:

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

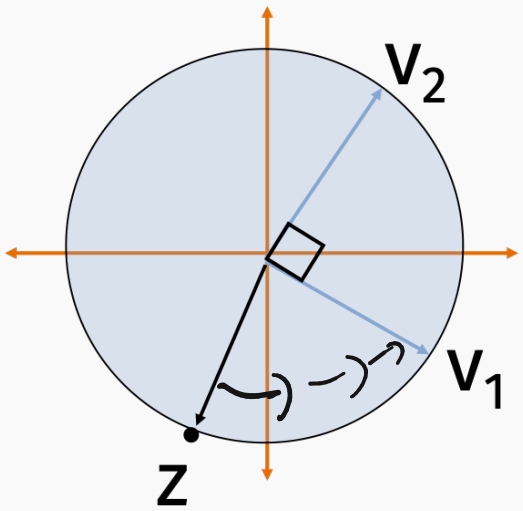
  Return $\mathbf{z}^{(T)}$

$z^{(t)} = c \cdot (X^T X)(X^T X) \cdots (X^T X) z^{(0)}$

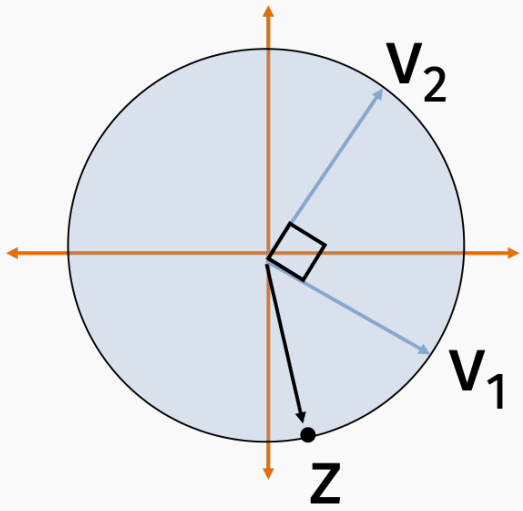
matrix power of $X^T X$

POWER METHOD INTUITION

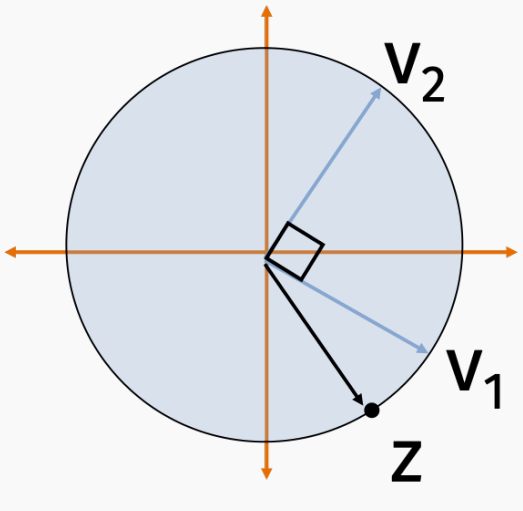
0 iterations



1 iterations



2 iterations



POWER METHOD FORMAL CONVERGENCE

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:

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

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

ONE STEP ANALYSIS OF POWER METHOD

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

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

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

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

$\|c^{(i)}\|_2 = \|V^T z^{(i)}\|_2 = \|z^{(i)}\|_2$

ONE STEP ANALYSIS OF POWER METHOD

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

$$z^{(i)} = \frac{1}{n_i}X^T X z^{(i-1)} = \frac{1}{n_i} V\Sigma^2 V^T z^{(i-1)} = \frac{1}{n_i} V\Sigma^2 c^{(i-1)}$$

$$X^TX = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T \qquad X = U\Sigma V^T$$

$$V^T z^{(i)} = \frac{1}{n_i} V^T V\Sigma^2 c^{(i-1)}$$

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

MULTI-STEP ANALYSIS OF POWER METHOD

Claim: After T updates:

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

$= 0$

$c^{(1)} = \frac{1}{n_1}\Sigma^2 c^{(0)} \qquad c^{2} = \frac{1}{n_2}\Sigma^2 c^{(1)} \ldots$

$c^{(T)} = \frac{1}{n_1 n_2 \ldots n_T}\Sigma^2\Sigma^2\Sigma^2\ldots\Sigma^2 c^{(0)}$

$= \frac{1}{\prod n_i}\Sigma^{2T} c^{(0)}$

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 $j \neq 1$.

POWER METHOD FORMAL CONVERGENCE

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 $\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2^2 \leq \epsilon$.

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

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

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

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

$$\|\mathbf{v}_1 - \mathbf{z}^{(T)}\|_2^2 = \|\mathbf{v}_1\|_2^2 + \|\mathbf{z}^{(T)}\|_2^2 - 2\langle \mathbf{v}_1, \mathbf{z}^{(T)} \rangle = 2 - 2\alpha_1 \leq \epsilon$$

POWER METHOD FORMAL CONVERGENCE

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 $\alpha_j = \frac{1}{\prod_{i=1}^{T} n_i} c_j^{(0)} \sigma_j^{2T}$

Assumption: Starting coefficient on first eigenvector is not too small:

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

$\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$

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

≤ 1

$$\frac{|\alpha_j|}{|\alpha_1|} = \frac{\sigma_j^{2T}}{\sigma_1^{2T}} \cdot \frac{|c_j^{(0)}|}{|c_1^{(0)}|} \leq \left(\frac{\sigma_j}{\sigma_1}\right)^{2T} \cdot \sqrt{d} \leq \left(\frac{\sigma_2}{\sigma_1}\right)^{2T} \cdot \sqrt{d}$$

$\geq O(1/\sqrt{d})$

$$= (1-\gamma)^{2T} \sqrt{d} \leq \sqrt{\frac{\epsilon}{d}}$$

want

Need $T = 1/\gamma \cdot \log(d/\epsilon) = O(1/\gamma \log(d/\epsilon))$.

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

STARTING COEFFICIENT ANALYSIS

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),

$\|\mathbf{g}\|_2^2 = \sum_{i=1}^{d} g_i^2$

$\|\mathbf{g}\|_2^2 \leq 2d$ with super high probability.

$\|\mathbf{g}\|_2 \leq O(\sqrt{d})$

$\mathbb{E}[\|\mathbf{g}\|_2^2] = d$

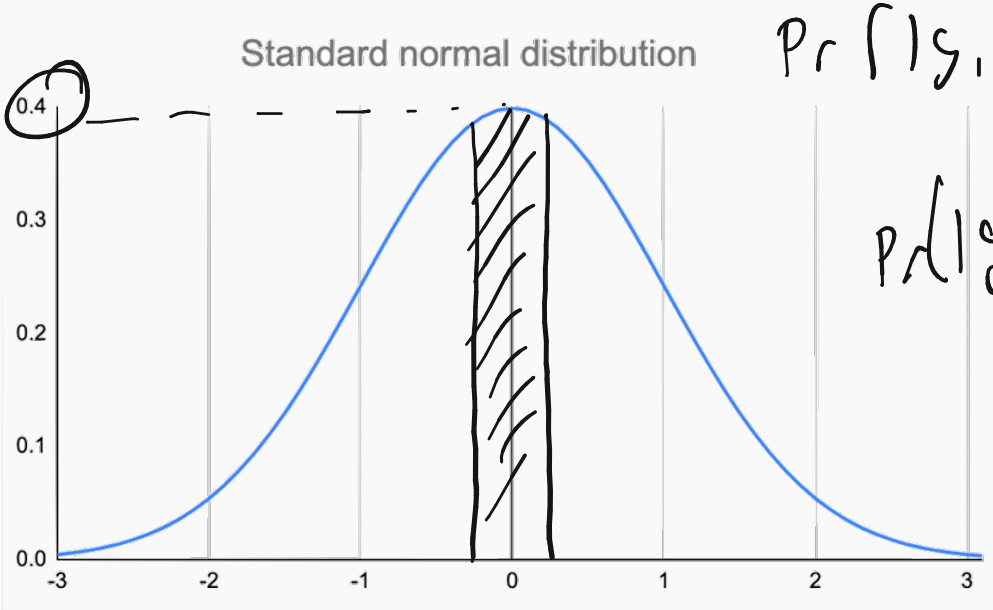
Need to show that with high probability, the magnitude of the first entry of $\mathbf{g} \geq c$ for a constant c. Think e.g. $c = 1/10$.

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

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

$1 - O(1/100)$

$1/100$



$\Pr[|g_1| \leq \alpha] \leq 2\alpha \cdot .4$

$\leq \alpha$

$\Pr[|g_1| \geq \alpha] \geq 1 - \alpha$

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

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

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

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

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

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.

$d \times (q)$

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.

LANCZOS METHOD ANALYSIS

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.

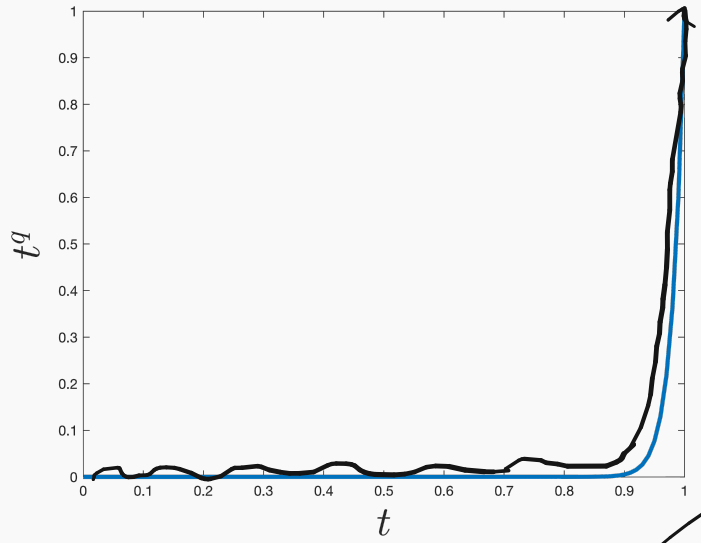
Power method returns:

$$\mathbf{v}_{X^t}.$$

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 x^q up to error Δ on $[0, \sigma_1^2]$.



optimal poly.

power methods result at q iteration.

$$\|X - Xv_{p^*}v_{p^*}^T\|_F^2 \leq \|X - Xv_{\hat{p}}v_{\hat{p}}^T\|_F^2 \approx \|X - Xv_{X^q}v_{X^q}^T\|_F^2 \approx \|X - Xv_1v_1^T\|_F^2$$

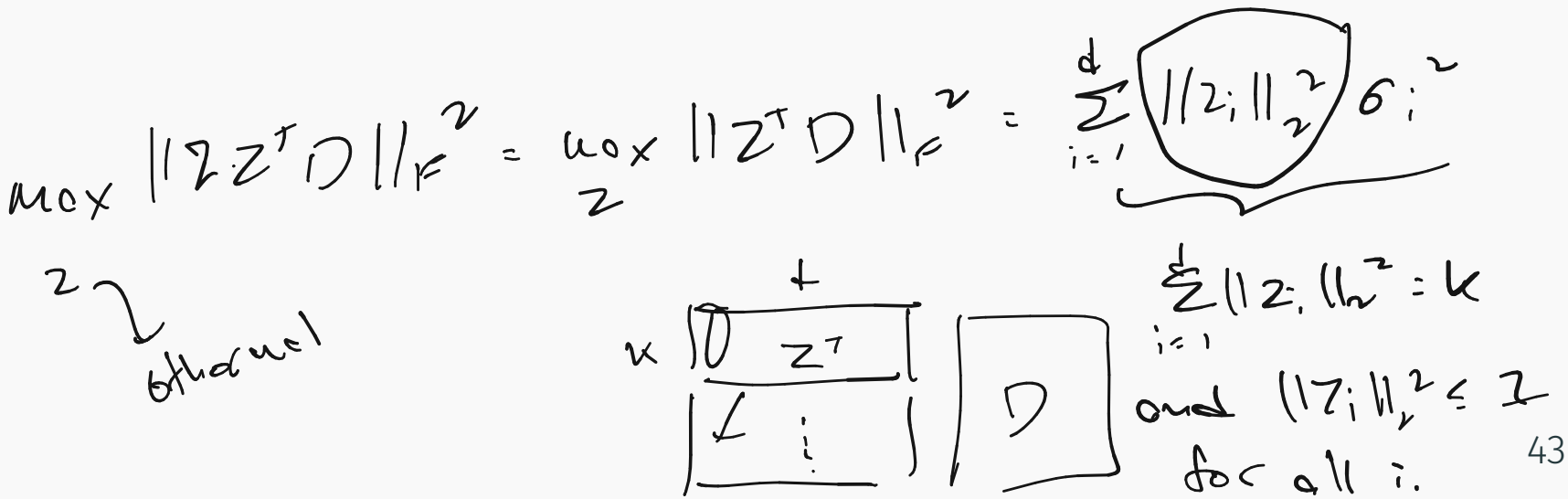
Runtime: $O\left(\frac{\log(d/\epsilon)}{\sqrt{\epsilon}} \cdot \text{nnz}(X)\right)$ vs. $O\left(\frac{\log(d/\epsilon)}{\epsilon} \cdot \text{nnz}(X)\right)$

GENERALIZATIONS TO LARGER k

- Block Krylov methods

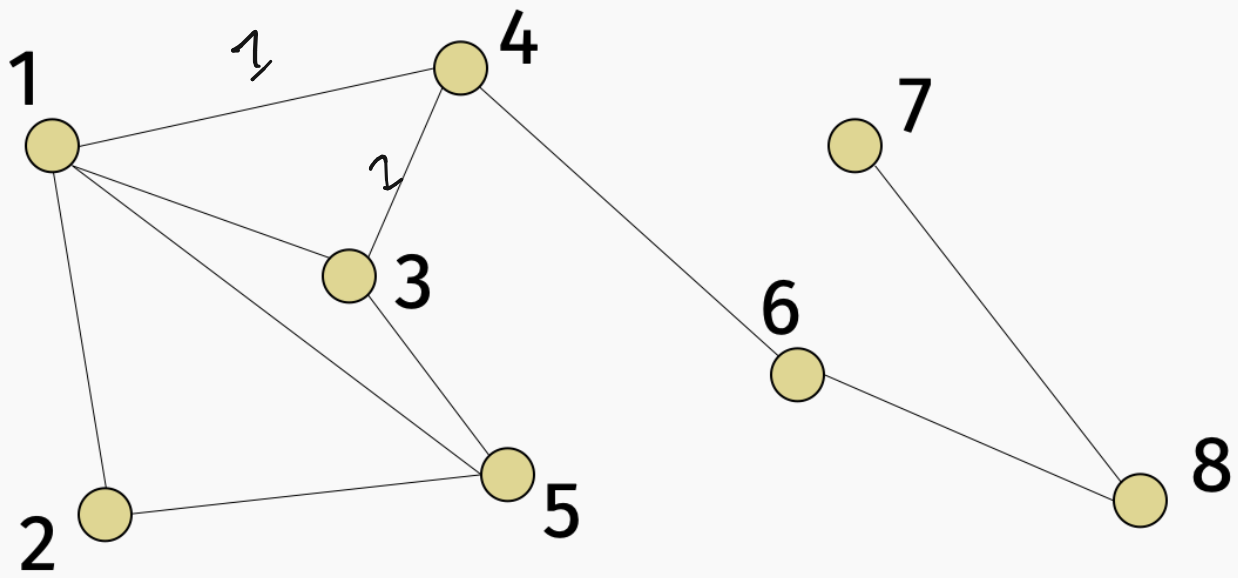
- Let $\mathbf{G} \in \mathbb{R}^{d \times k}$ be a random Gaussian matrix.
- $\mathcal{K}_q = \left[\mathbf{G}, (\mathbf{X}^T\mathbf{X}) \cdot \mathbf{G}, (\mathbf{X}^T\mathbf{X})^2 \cdot \mathbf{G}, \ldots, (\mathbf{X}^T\mathbf{X})^q \cdot \mathbf{G}\right]$

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.



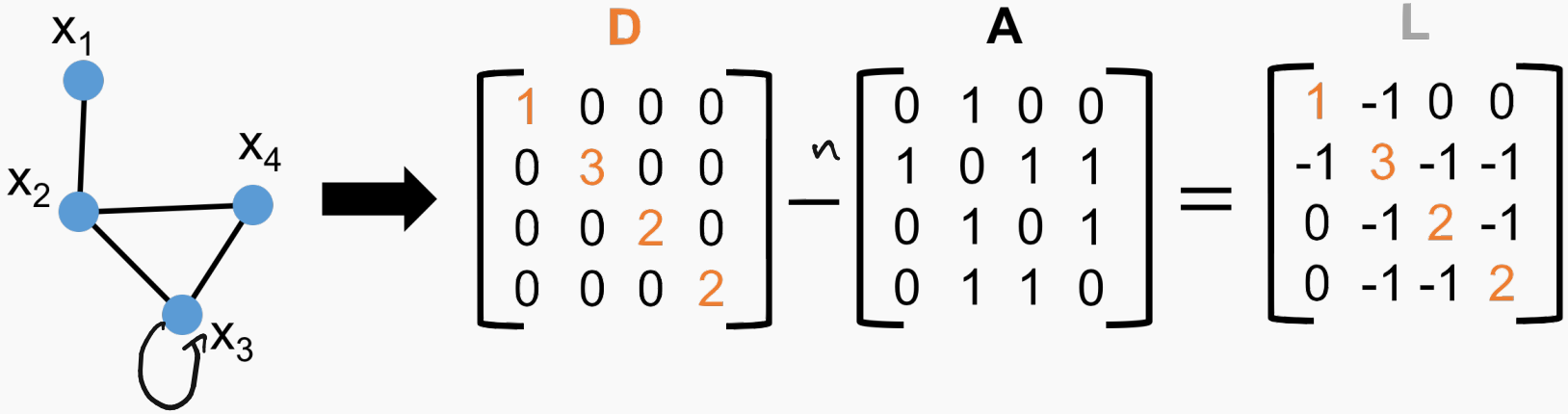
BREAK

Main idea: Understand graph data by constructing natural matrix representations, and studying that matrix's spectrum (eigenvalues/eigenvectors).



For now assume $G = (V, E)$ is an undirected, unweighted graph with n nodes.

Two most common representations: $n \times n$ adjacency matrix $\mathbf{A}$ and graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ where $\mathbf{D}$ is the diagonal degree matrix.



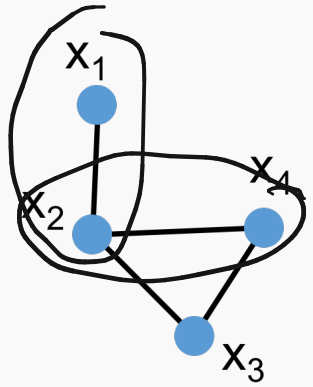
Also common to look at normalized versions of both of these: $\bar{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ and $\bar{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$.

SPECTRAL GRAPH THEORY TIDBITS

- If $\mathbf{L}$ have k eigenvalues equal to 0, then G has k connected components.
- Sum of cubes of $\mathbf{A}$'s eigenvalues is equal to number of triangles in the graph times 6.
- Sum of eigenvalues to the power q is proportional to the number of q cycles.
- Today: Eigenvectors of super useful in clustering graph data.

THE LAPLACIAN VIEW

m = # edges in graph



$$\underset{D}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}} - \underset{A}{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}} = \underset{L}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}}$$

$L = B^T B$ where B is the signed "edge-vertex incidence" matrix.

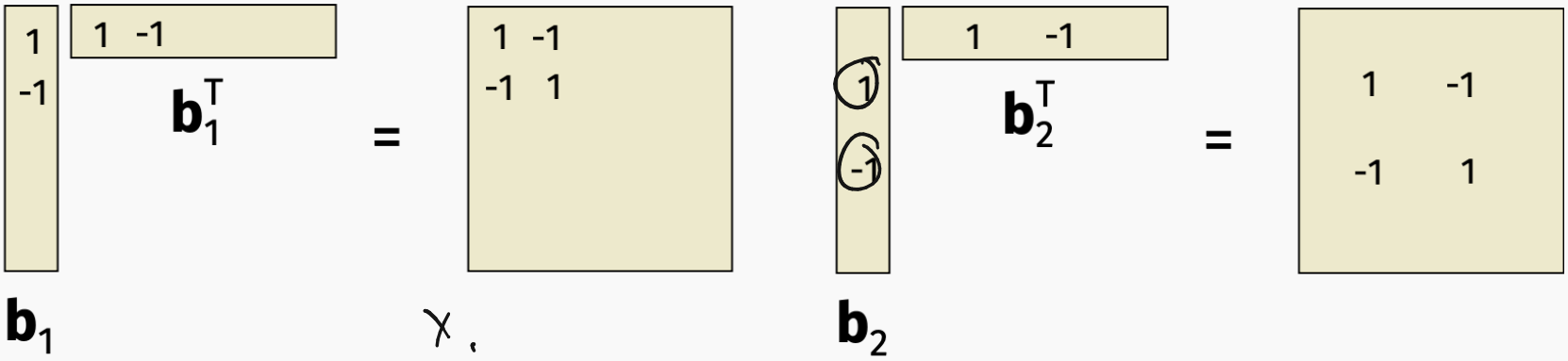
$$B = \begin{matrix} \\ \text{\# edges} \end{matrix} \begin{bmatrix} 1 & -1 & & \\ & 1 & -1 & \\ & & & \\ & & & \end{bmatrix}$$

\# nodes

$L = B^T B = \sum_{i=1}^{m} b_i b_i^T$ → outer product

$$L = B^T B = b_1 b_1^T + b_2 b_2^T + \ldots + b_m b_m^T,$$

where b_i is the i^{th} row of B (each row corresponds to a single edge).



Conclusions from $L = B^T B$

$x^T L x = x^T B^T B x = \|Bx\|_2^2 \geq 0$

- L is positive semidefinite: $x^T L x \geq 0$ for all x.
- $L = V \Sigma^2 V^T$ where $U \Sigma V^T$ is B's SVD. Columns of V are eigenvectors of L.
- For any vector $x \in \mathbb{R}^n$,

$$x^T L x = \sum_{(i,j) \in E} (x(i) - x(j))^2.$$

$\|Bx\|_2^2$

$= x(i) - x(j)$

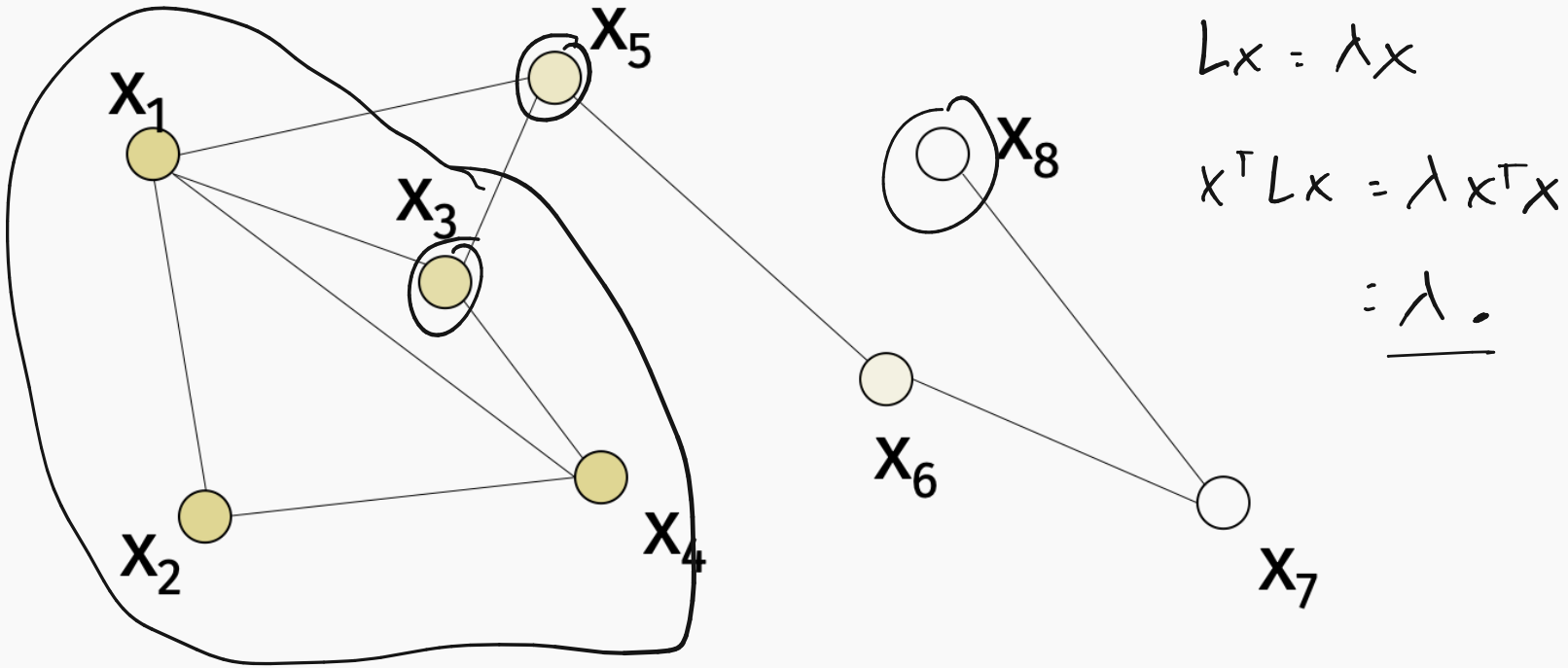
$\mathbf{x}^T L\mathbf{x} = \sum_{(i,j)\in E}(\mathbf{x}(i) - \mathbf{x}(j))^2$. So $\mathbf{x}^T L\mathbf{x}$ is small if $\mathbf{x}$ is a "smooth" function with respect to the graph.

$[1 \quad 2 \quad 1.5 \quad -3 \quad -7 \quad -1]$

$Lx = \lambda x$

$x^T Lx = \lambda x^T x$

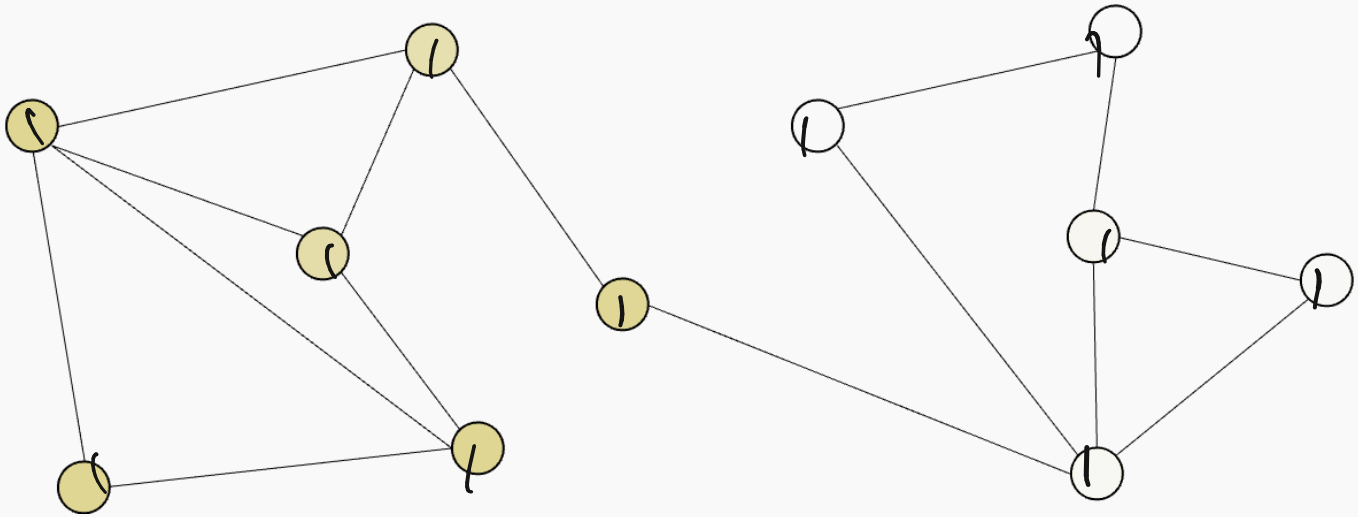
$= \lambda \cdot$



Eigenvectors of the Laplacian with small eigenvalues correspond to smooth functions over the graph.

ANOTHER EXAMPLE OF A SMOOTH FUNCTION

Any function that only has a large change across a small cut in the graph is also smooth.



$$1^T L 1 = \|B 1\|_2^2 = 0$$

Courant–Fischer min-max principle

Let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ be the eigenvectors of $\mathbf{L}$.

$$\mathbf{v}_n = \arg\min_{\|\mathbf{v}\|=1} \mathbf{v}^T \mathbf{L} \mathbf{v}$$

$$\mathbf{v}_{n-1} = \arg\min_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_n} \mathbf{v}^T \mathbf{L} \mathbf{v}$$

$$\mathbf{v}_{n-2} = \arg\min_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_n, \mathbf{v}_{n-1}} \mathbf{v}^T \mathbf{L} \mathbf{v}$$

$$\vdots$$

$$\mathbf{v}_1 = \arg\min_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_n, \ldots, \mathbf{v}_2} \mathbf{v}^T \mathbf{L} \mathbf{v}$$

Courant–Fischer min-max principle

Let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ be the eigenvectors of $\mathbf{L}$.

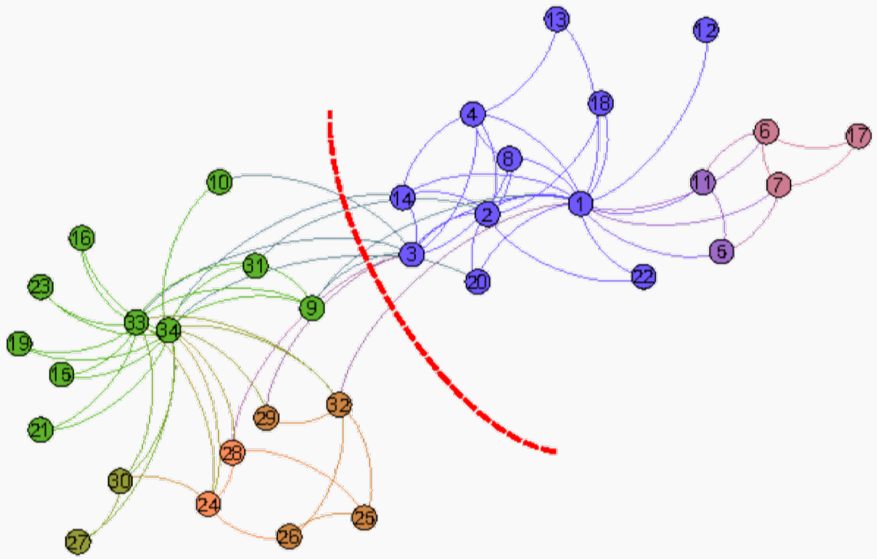
$$\begin{aligned}
\mathbf{v}_1 &= \underset{\|\mathbf{v}\|=1}{\arg\max}\, \mathbf{v}^T \mathbf{L} \mathbf{v} \\
\mathbf{v}_2 &= \underset{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1}{\arg\max}\, \mathbf{v}^T \mathbf{L} \mathbf{v} \\
\mathbf{v}_3 &= \underset{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1, \mathbf{v}_2}{\arg\max}\, \mathbf{v}^T \mathbf{L} \mathbf{v} \\
&\vdots \\
\mathbf{v}_n &= \underset{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}}{\arg\max}\, \mathbf{v}^T \mathbf{L} \mathbf{v}
\end{aligned}$$

EXAMPLE APPLICATION OF SPECTRAL GRAPH THEORY

- Study graph partitioning problem important in 1) understanding social networks 2) nonlinear clustering in unsupervised machine learning (spectral clustering). 3) Graph visualization 4) Mesh partitioning
- See how this problem can be solved heuristically using Laplacian eigenvectors.
- Give a full analysis of the method for a common random graph model.
- Use two tools: matrix concentration and eigenvector perturbation bounds.

Common goal: Given a graph $G = (V, E)$, partition nodes along a cut that:

- Has few crossing edges: $|\{(u, v) \in E : u \in S, v \in T\}|$ is small.
- Separates large partitions: $|S|, |T|$ are not too small.



(a) Zachary Karate Club Graph

Important in understanding community structure in social networks.

Social networks in the 1970s

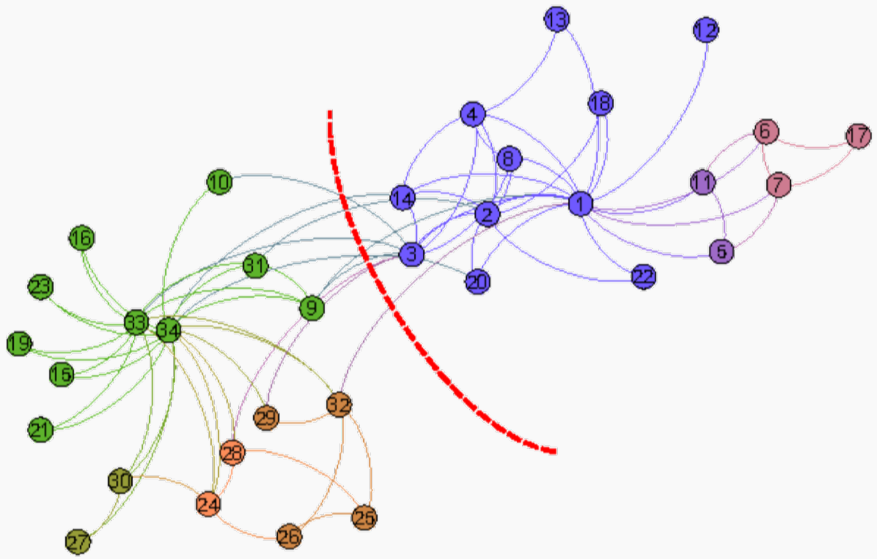
Wayne W. Zachary (1977). An Information Flow Model for Conflict and Fission in Small Groups.

"The network captures 34 members of a karate club, documenting links between pairs of members who interacted outside the club. During the study a conflict arose between the administrator "John A" and instructor "Mr. Hi" (pseudonyms), which led to the split of the club into two. Half of the members formed a new club around Mr. Hi; members from the other part found a new instructor or gave up karate. Based on collected data Zachary correctly assigned all but one member of the club to the groups they actually joined after the split." – Wikipedia

Beautiful paper – definitely worth checking out!

Common goal: Given a graph $G = (V, E)$, partition nodes along a cut that:

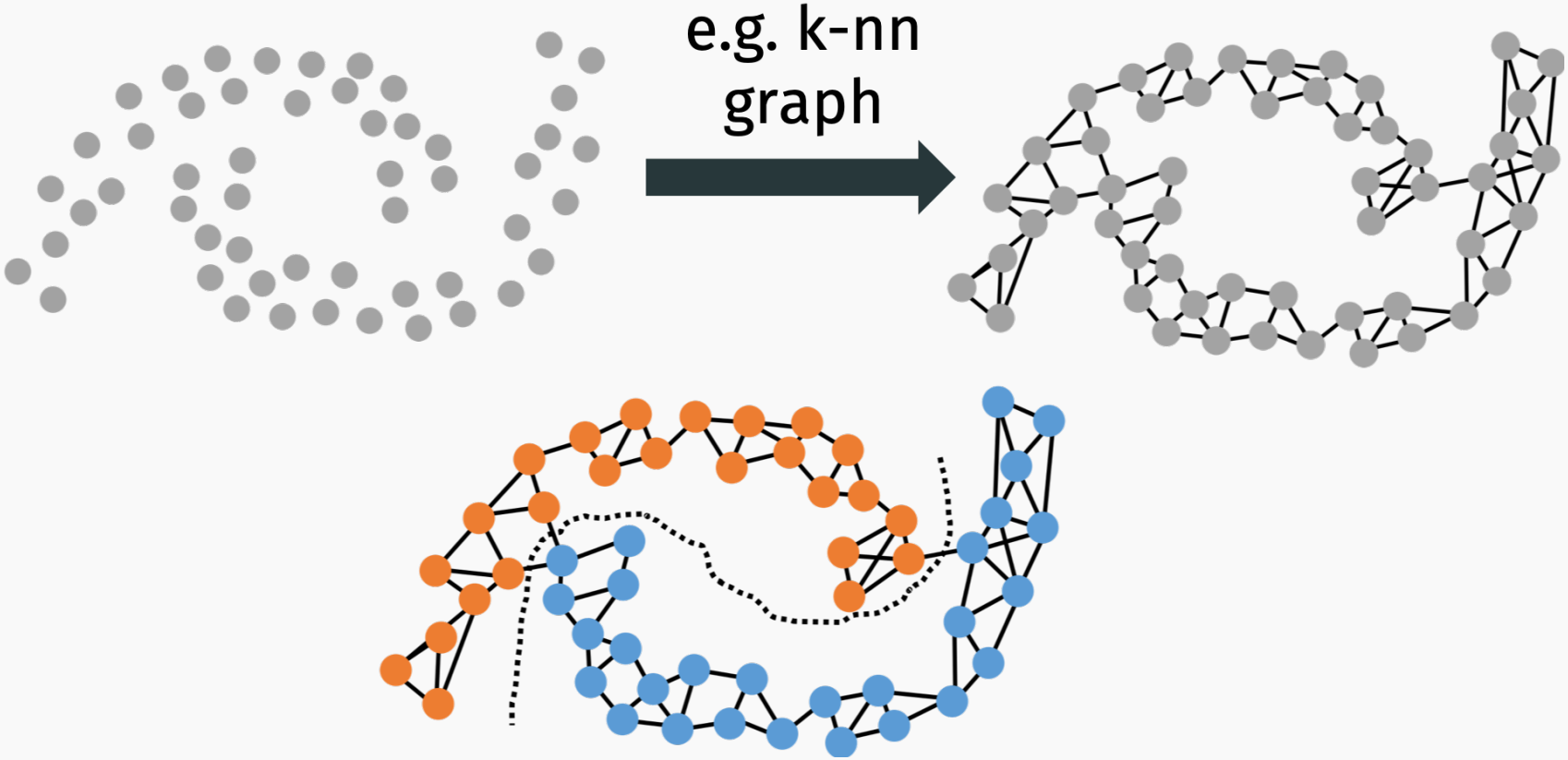
- Has few crossing edges: $|\{(u, v) \in E : u \in S, v \in T\}|$ is small.
- Separates large partitions: $|S|, |T|$ are not too small.



(a) Zachary Karate Club Graph

Important in understanding community structure in social networks.

Idea: Construct synthetic graph for data that is hard to cluster.



Spectral Clustering, Laplacian Eigenmaps, Locally linear embedding, Isomap, etc.

There are many way's to formalize Zachary's problem:

β-Balanced Cut:

$$\min_S \text{cut}(S, V \setminus S) \quad \text{such that} \quad \min(|S|, |V \setminus S|) \geq \beta \text{ for } \beta \leq .5$$

Sparsest Cut:

$$\min_S \frac{\text{cut}(S, V \setminus S)}{\min(|S|, |V \setminus S|)}$$

Most formalizations lead to NP-hard problems. Lots of interest in designing polynomial time approximation algorithms, but tend to be slow. In practice, much simpler methods based on the graph spectrum are used.

Spectral methods run in at worst $O(n^3)$ time (faster if you use iterative methods).

Basic spectral clustering method:

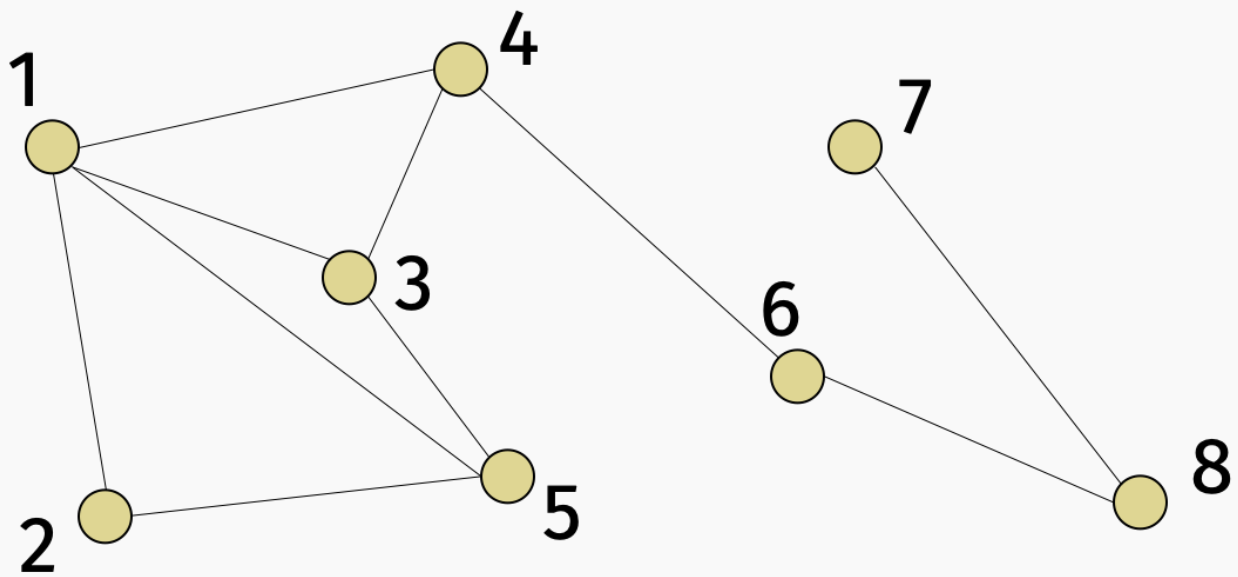
- Compute second smallest eigenvalue of graph, $\mathbf{v}_{n-1}$.
- $\mathbf{v}_{n-1}$ has an entry for every node i in the graph.
- If the i^{th} entry is positive, put node i in T.
- Otherwise if the i^{th} entry is negative, put i in S.

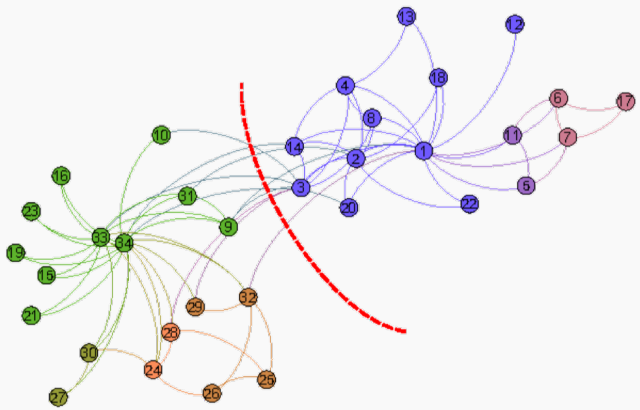
This shouldn't make much sense yet! We will see that is a "relax and round" algorithm in disguise.

Another conclusion from $\mathsf{L} = \mathsf{B}^T\mathsf{B}$:

For a cut indicator vector $\mathbf{c} \in \{-1, 1\}^n$ with $\mathbf{c}(i) = -1$ for $i \in S$ and $\mathbf{c}(i) = 1$ for $i \in T = V \setminus S$:

$$\mathbf{c}^T L \mathbf{c} = \sum_{(i,j)\in E} (\mathbf{c}(i) - \mathbf{c}(j))^2 = 4 \cdot \mathsf{cut}(S, T). \tag{1}$$



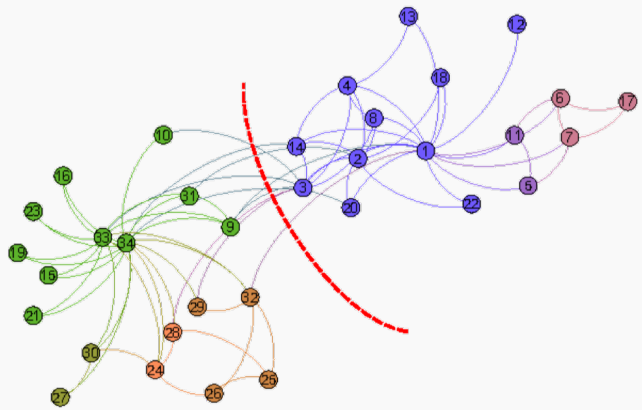


(a) Zachary Karate Club Graph

For a cut indicator vector $\mathbf{c} \in \{-1, 1\}^n$ with $\mathbf{c}(i) = -1$ for $i \in S$ and $\mathbf{c}(i) = 1$ for $i \in T$:

- $\mathbf{c}^T L \mathbf{c} = 4 \cdot cut(S, T)$.
- $\mathbf{c}^T \mathbf{1} = |T| - |S|$.

Want to minimize both $\mathbf{c}^T L \mathbf{c}$ (cut size) and $|\mathbf{c}^T \mathbf{1}|$ (imbalance).



(a) Zachary Karate Club Graph

Equivalent formulation if we divide everything by $\sqrt{n}$ so that $\mathbf{c}$ has norm 1. Then $\mathbf{c} \in \{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\}^n$ and:

- $\mathbf{c}^T L \mathbf{c} = \frac{4}{n} \cdot cut(S, T)$.
- $\mathbf{c}^T \mathbf{1} = \frac{1}{\sqrt{n}}(|T| - |S|)$.

Want to minimize both $\mathbf{c}^T L \mathbf{c}$ (cut size) and $|\mathbf{c}^T \mathbf{1}|$ (imbalance).

Consider the "perfectly balanced" version of the balanced cut problem:

$$\min_{\mathbf{c} \in \{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\}^n} \mathbf{c}^T \mathbf{L} \mathbf{c} \text{ such that } \mathbf{c}^T \mathbf{1} = 0.$$

Claim: If we relax the constraint $\mathbf{c} \in \{-\frac{1}{\sqrt{n}} \frac{1}{\sqrt{n}}\}^n$ to $\|\mathbf{c}\|_2 = 1$, then this problem is exactly minimized by the second smallest eigenvector $\mathbf{v}_{n-1}$ of $\mathbf{L}$.

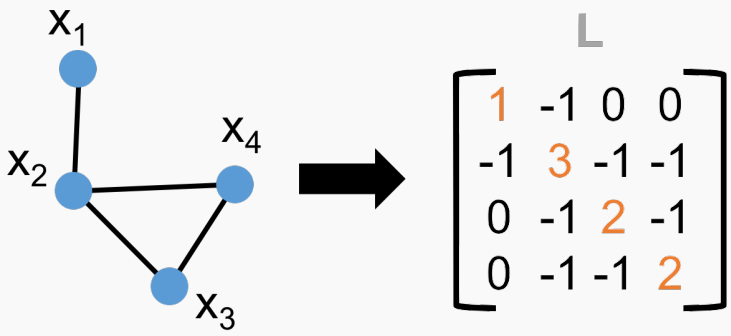
Approach: Relax, find $\mathbf{v}_{n-1}$, then round back to a vector with $-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}$ entries.

SMALLEST LAPLACIAN EIGENVECTOR

The smallest eigenvector/singular vector $\mathbf{v}_n$ satisfies:

$$\mathbf{v}_n = \frac{1}{\sqrt{n}} \cdot \mathbf{1} = \underset{v \in \mathbb{R}^n \text{ with } \|\mathbf{v}\|=1}{\arg\min} \mathbf{v}^T L \mathbf{v}$$

with $\mathbf{v}_n^T L \mathbf{v}_n = 0$.



SECOND SMALLEST LAPLACIAN EIGENVECTOR

By Courant-Fischer, $\mathbf{v}_{n-1}$ is given by:

$$\mathbf{v}_{n-1} = \underset{\|\mathbf{v}\|=1,\ \mathbf{v}_n^T\mathbf{v}=0}{\arg\min} \ \mathbf{v}^T L \mathbf{v}$$

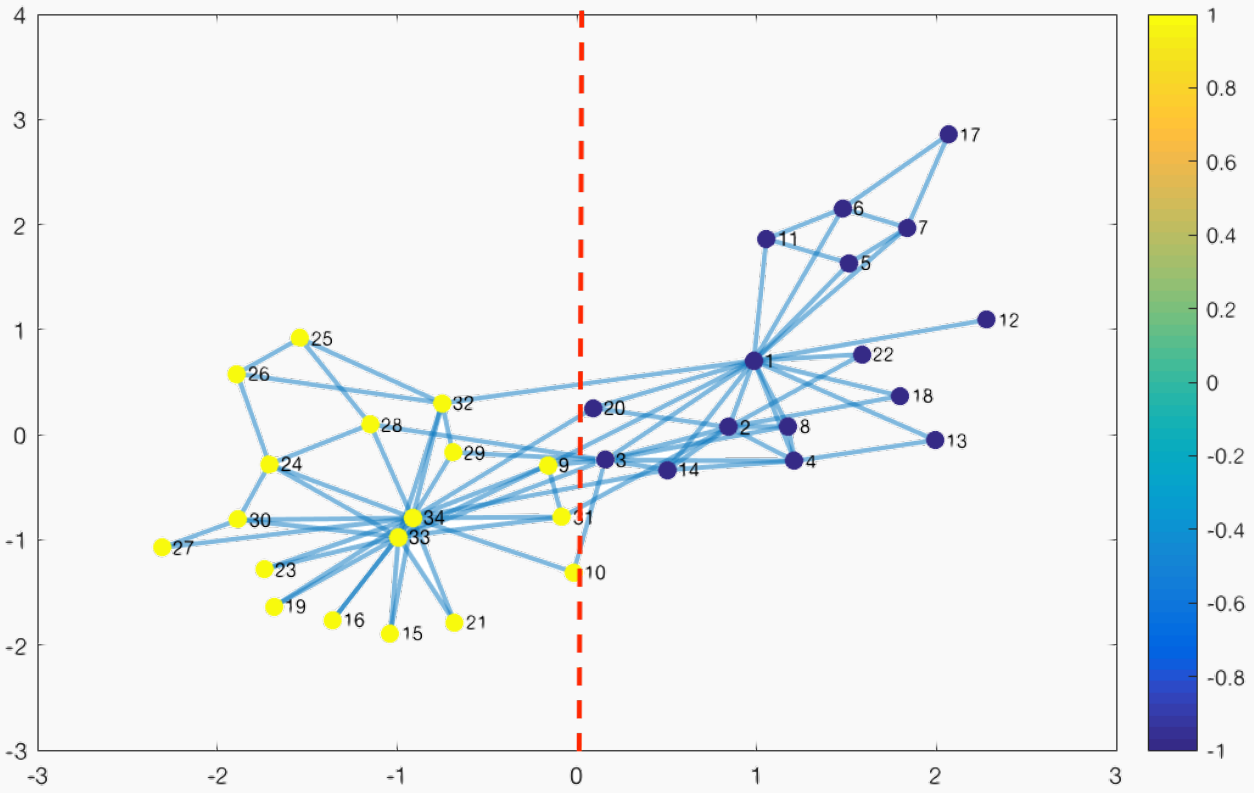
which is equivalent to

$$\mathbf{v}_{n-1} = \underset{\|\mathbf{v}\|=1,\ \mathbf{1}^T\mathbf{v}=0}{\arg\min} \ \mathbf{v}^T L \mathbf{v}.$$

Final relax and round algorithm: Compute

$$\mathbf{v}_{n-1} = \underset{v \in \mathbb{R}^n \text{ with } \|\mathbf{v}\|=1,\ \mathbf{v}^T\mathbf{1}=0}{\arg\min} \mathbf{v}^T L \mathbf{v}$$

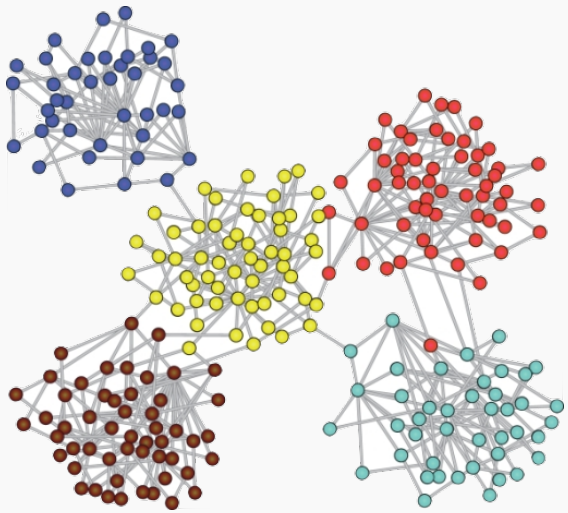
Set S to be all nodes with $\mathbf{v}_{n-1}(i) < 0$, and T to be all with $\mathbf{v}_{n-1}(i) \geq 0$. I.e. set $\mathbf{c} = \mathbf{sign}(\mathbf{v}_{n-1})$



SPECTRAL PARTITIONING IN PRACTICE

The Shi-Malik normalized cuts algorithm is one of the most commonly used variants of this approach, using the normalized Laplacian $\bar{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$.

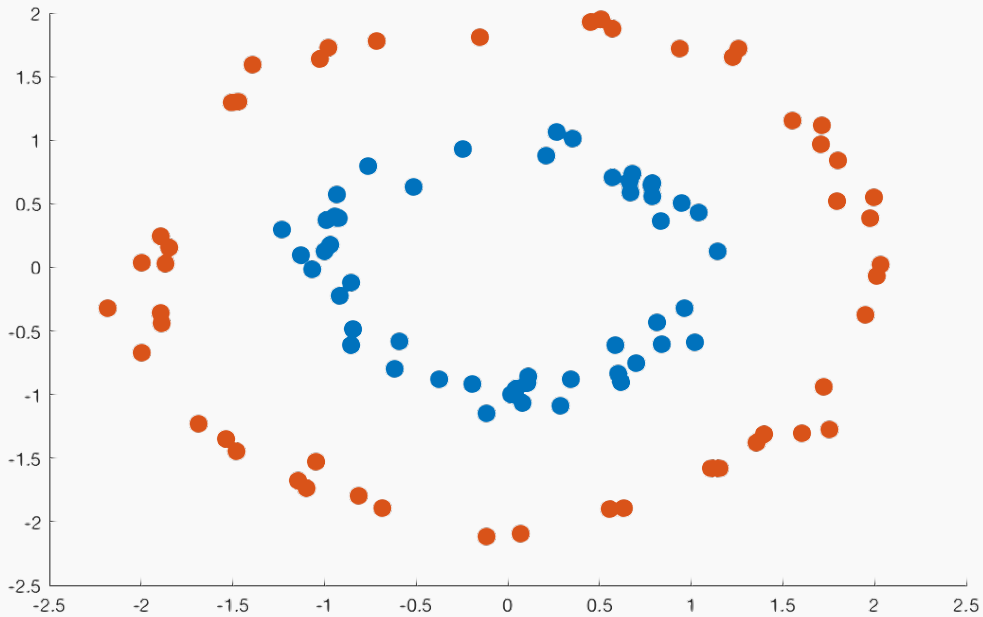
Important consideration: What to do when we want to split the graph into more than two parts?



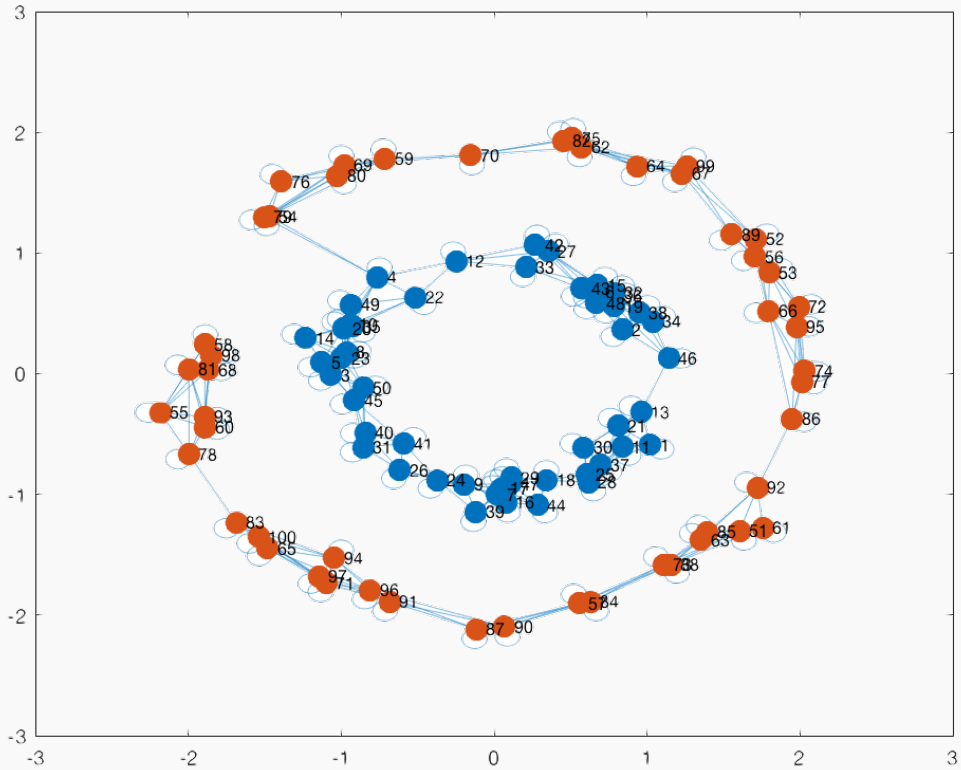
Spectral Clustering:

- Compute smallest k eigenvectors $\mathbf{v}_{n-1}, \ldots, \mathbf{v}_{n-\ell}$ of $\mathbf{L}$.
- Represent each node by its corresponding row in $\mathbf{V} \in \mathbb{R}^{n \times \ell}$ whose rows are $\mathbf{v}_{n-1}, \ldots \mathbf{v}_{n-\ell}$.
- Cluster these rows using k-means clustering (or really any clustering method).
- Often we choose $\ell = k$, but not necessarily.

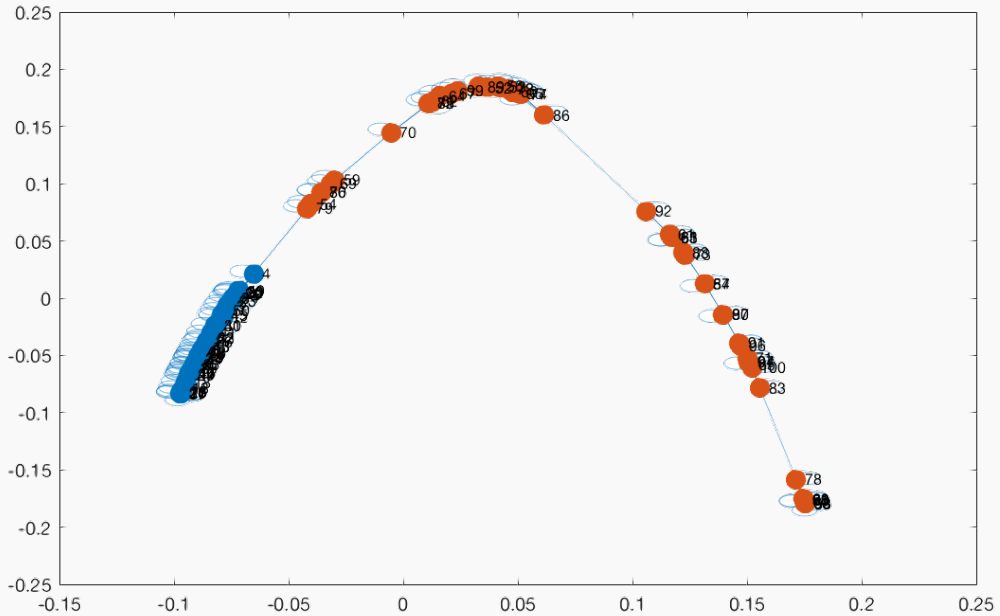
Original Data: (not linearly separable)



k-Nearest Neighbors Graph:



Embedding with eigenvectors v_{n-1}, v_{n-2}: (linearly separable)

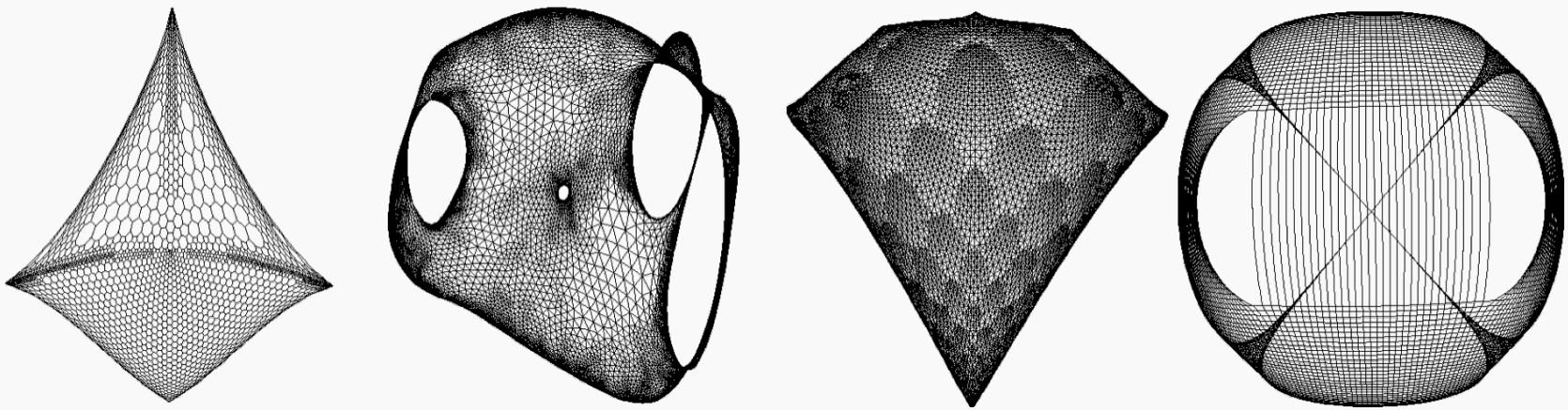


WHY DOES THIS WORK?

Intuitively, since $\mathbf{v} \in \mathbf{v}_1, \ldots \mathbf{v}_k$ are smooth over the graph,

$$\sum_{i,j \in E} (\mathbf{v}[i] - \mathbf{v}[j])^2$$

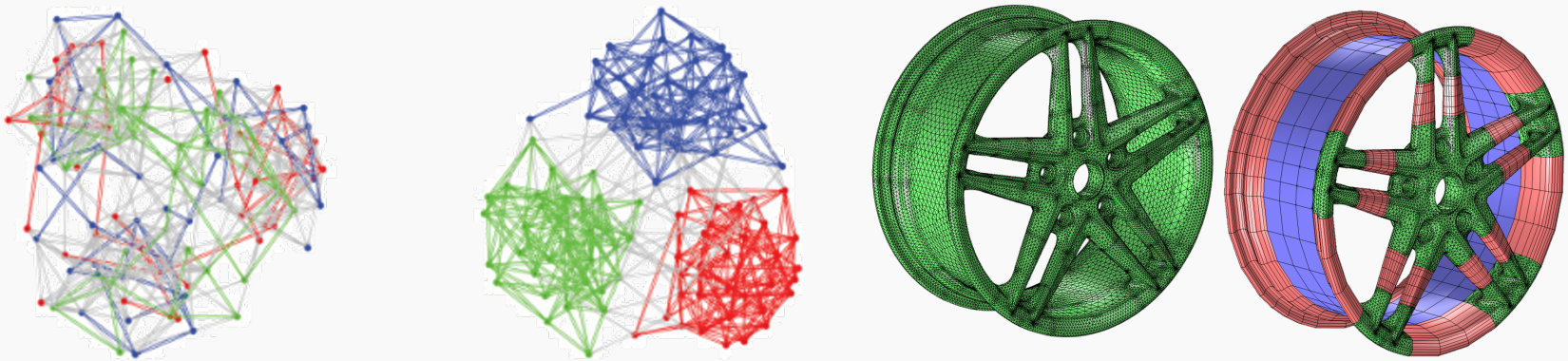
is small for each coordinate. I.e. this embedding explicitly encourages nodes connected by an edge to be placed in nearby locations in the embedding.



Also useful e.g., in graph drawing.

TONS OF OTHER APPLICATIONS!

Fast balanced partitioning algorithms are also use in distributing data in graph databases, for partitioning finite element meshes in scientific computing (e.g., that arise when solving differential equations), and more.



Lots of good software packages (e.g. METIS).

GENERATIVE MODELS

So far: Showed that spectral clustering partitions a graph along a small cut between large pieces.

- No formal guarantee on the 'quality' of the partitioning.
- Difficult to analyze for general input graphs.

Common approach: Design a natural generative model that produces random but realistic inputs and analyze how the algorithm performs on inputs drawn from this model.

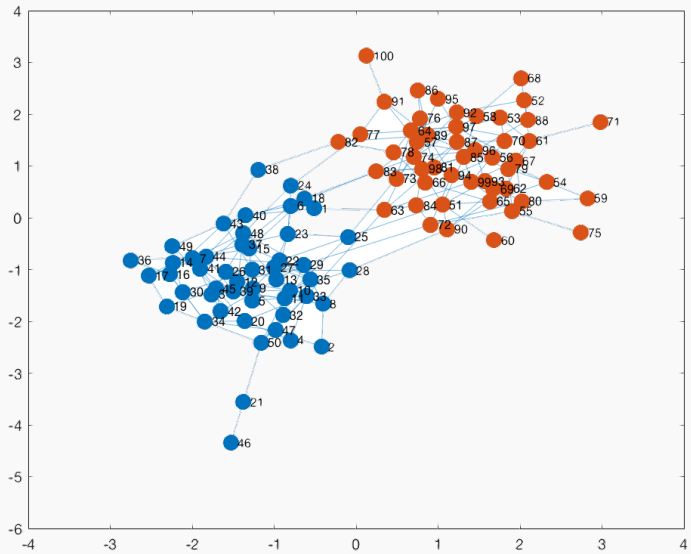
- Very common in algorithm design and analysis. Great way to start approaching a problem.
- This is also the whole idea behind Bayesian Machine Learning (can be used to justify ℓ_2 linear regression, k-means clustering, PCA, etc.)

Ideas for a generative model for **social network graphs** that would allow us to understand partitioning?

Stochastic Block Model (Planted Partition Model):

Let $G_n(p, q)$ be a distribution over graphs on n nodes, split equally into two groups B and C, each with $n/2$ nodes.

- Any two nodes in the same group are connected with probability p (including self-loops).
- Any two nodes in different groups are connected with prob. $q < p$.



Next class we will analyze spectral clustering for SBM graphs.

Have a good Thanksgiving break!