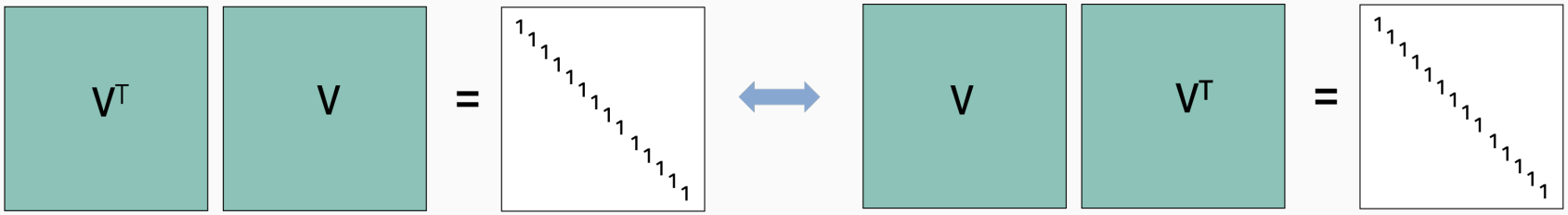
CS-GY 6763: Lecture 10
Singular value decomposition, low-rank approximation, Krylov subspace methods

NYU Tandon School of Engineering, Prof. Christopher Musco

LINEAR ALGEBRA REMINDER

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



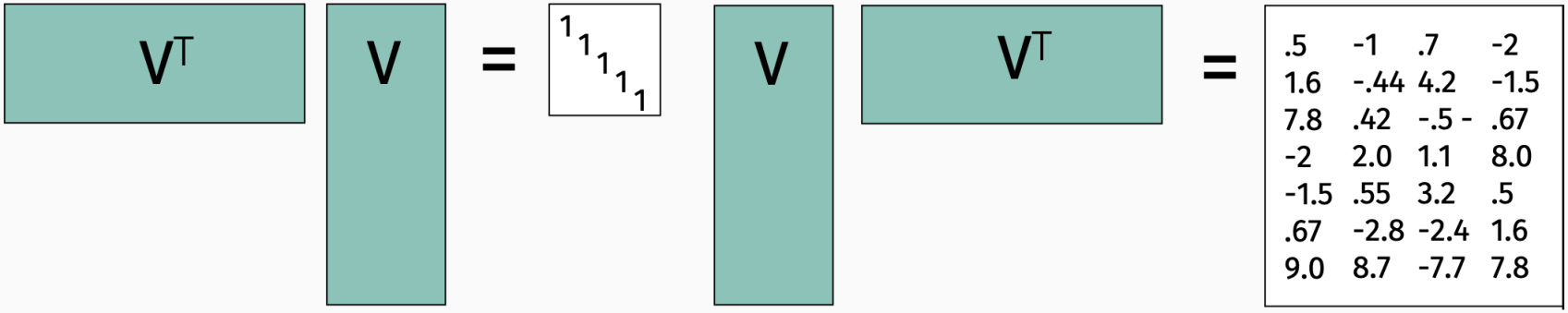
$$V^T V = I = V V^T$$

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

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 $\|\mathbf{V}^T\mathbf{X}\|_F^2 = \|\mathbf{X}\|_F^2$.

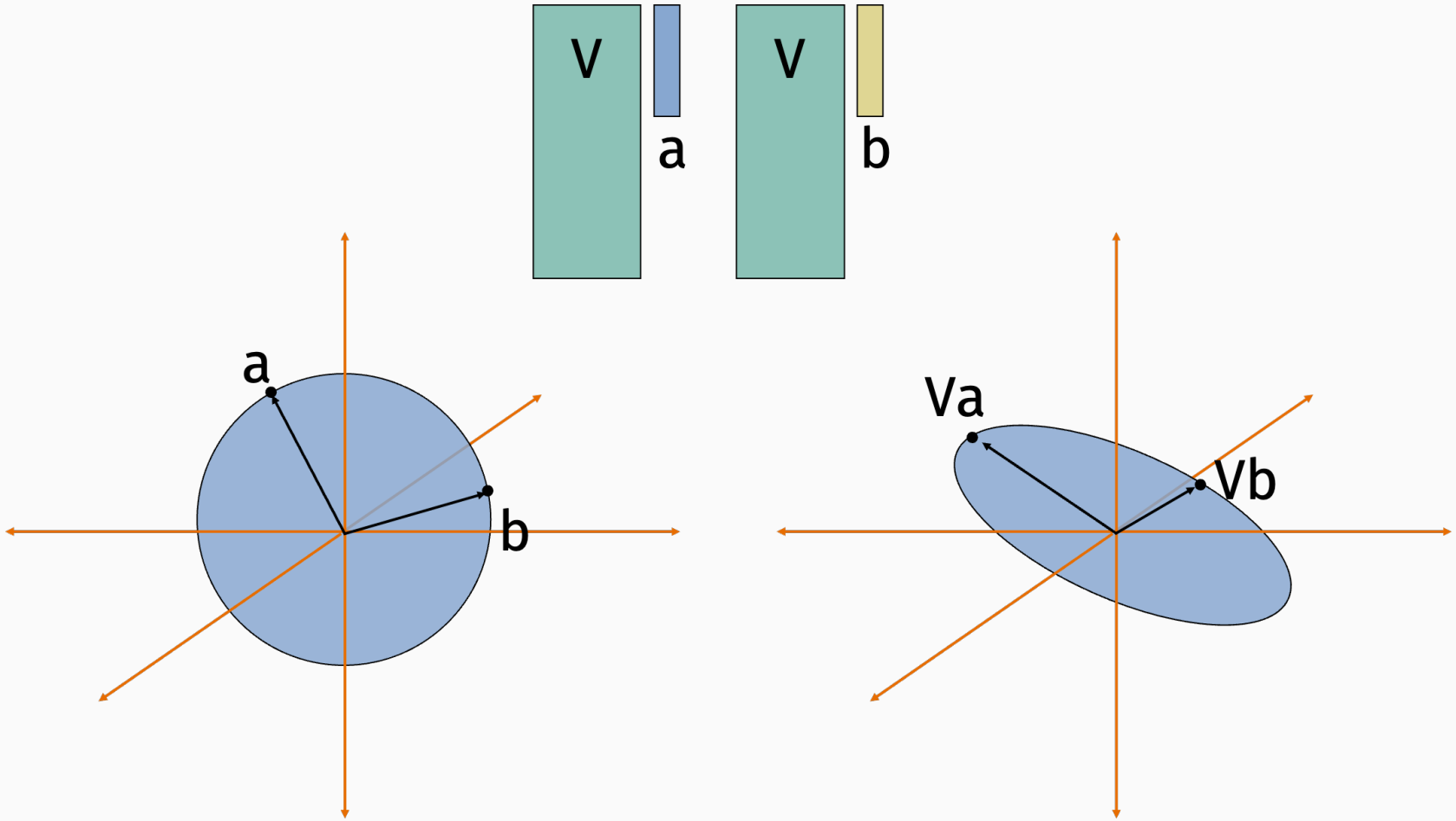
The same is not true for rectangular matrices:



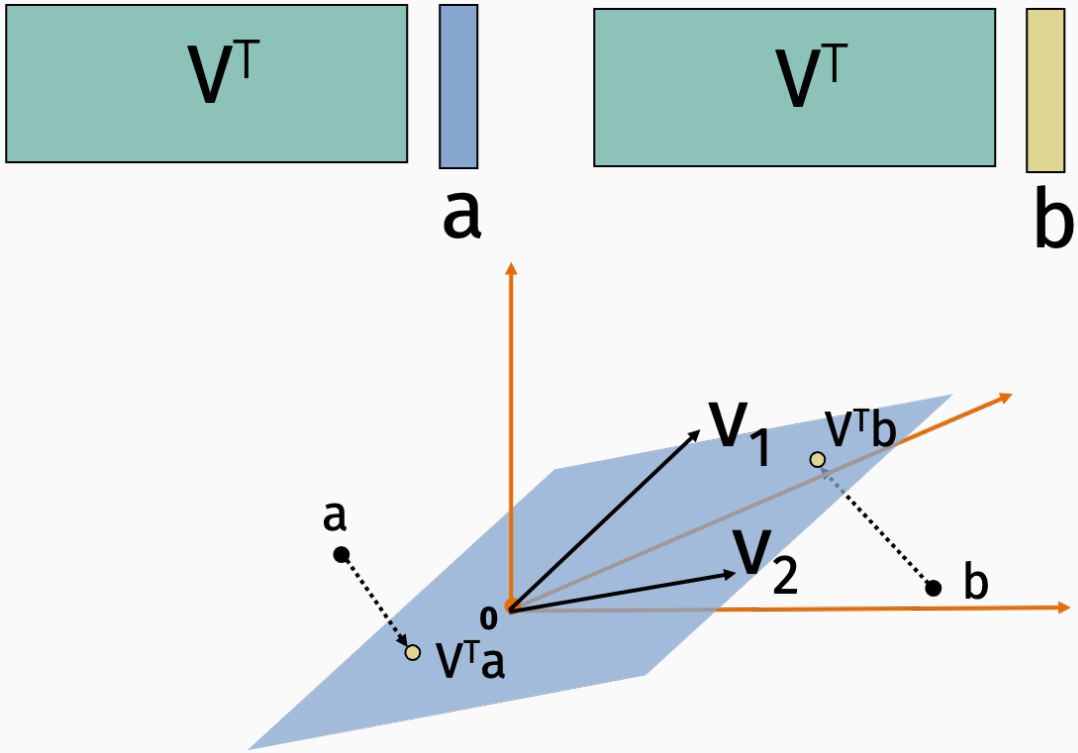
$$V^T V = I \qquad \text{but} \qquad VV^T \neq I$$

For any x, $\|Vx\|_2^2 = \|x\|_2^2$ but $\|V^T x\|_2^2 \neq \|x\|_2^2$ 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).

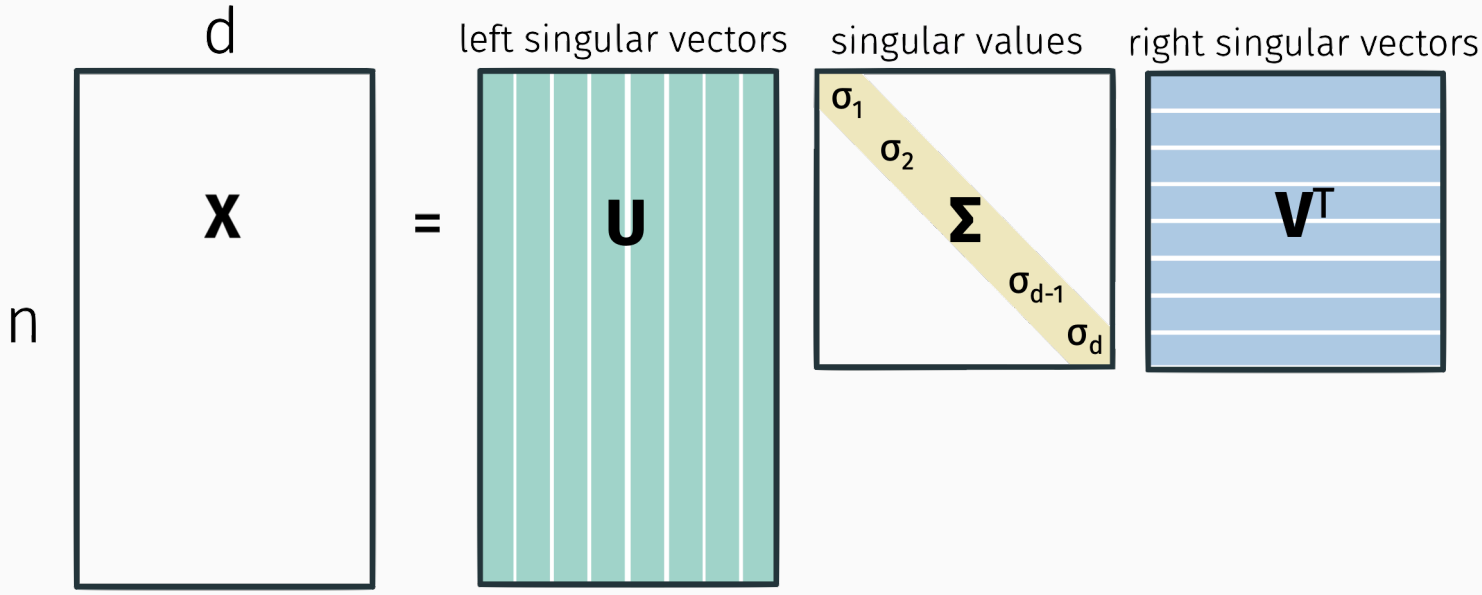


So we always have that $\|\mathbf{V}^T\mathbf{x}\|_2 \leq \|\mathbf{x}\|_2$.

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 $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_d \geq 0$.

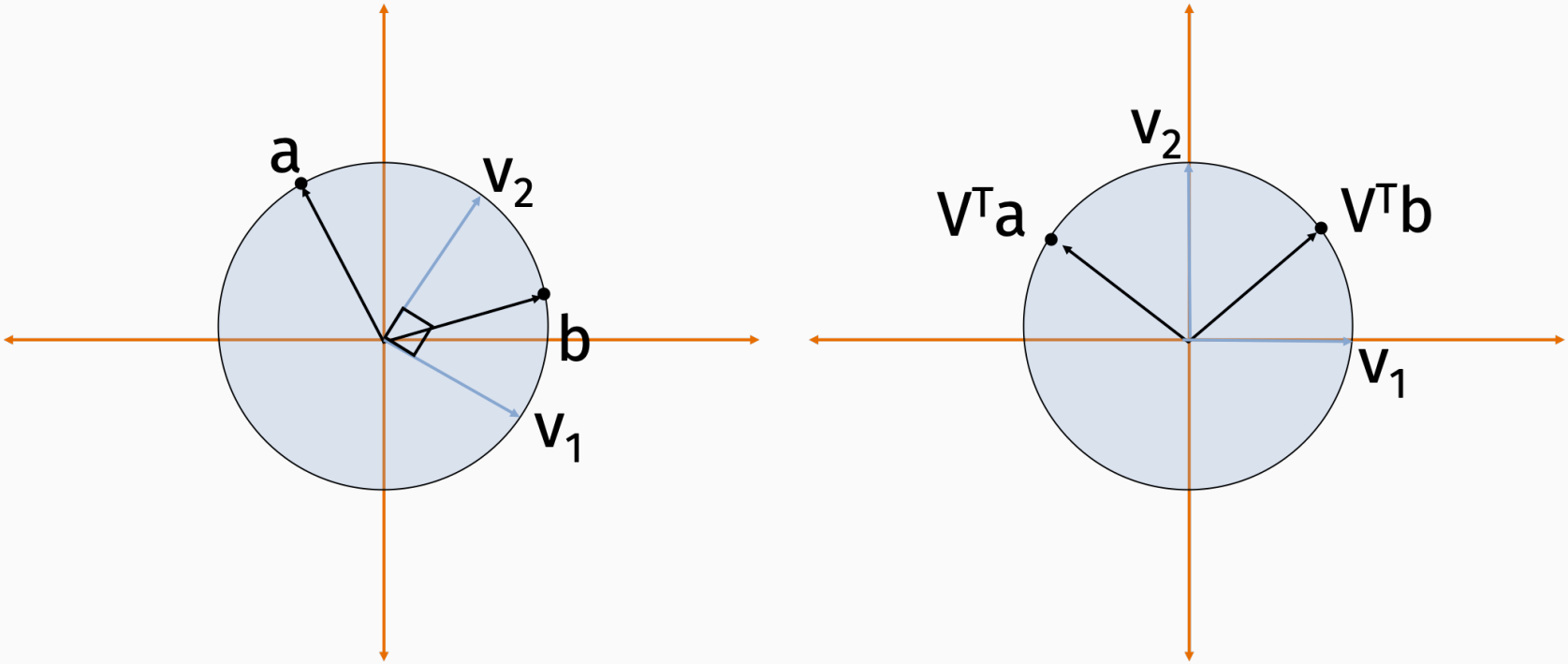
Singular values are unique. Factors are not. Would still get a valid SVD by multiplying both i^{th} column of $\mathbf{V}$ and $\mathbf{U}$ by -1.

Important take away from singular value decomposition.

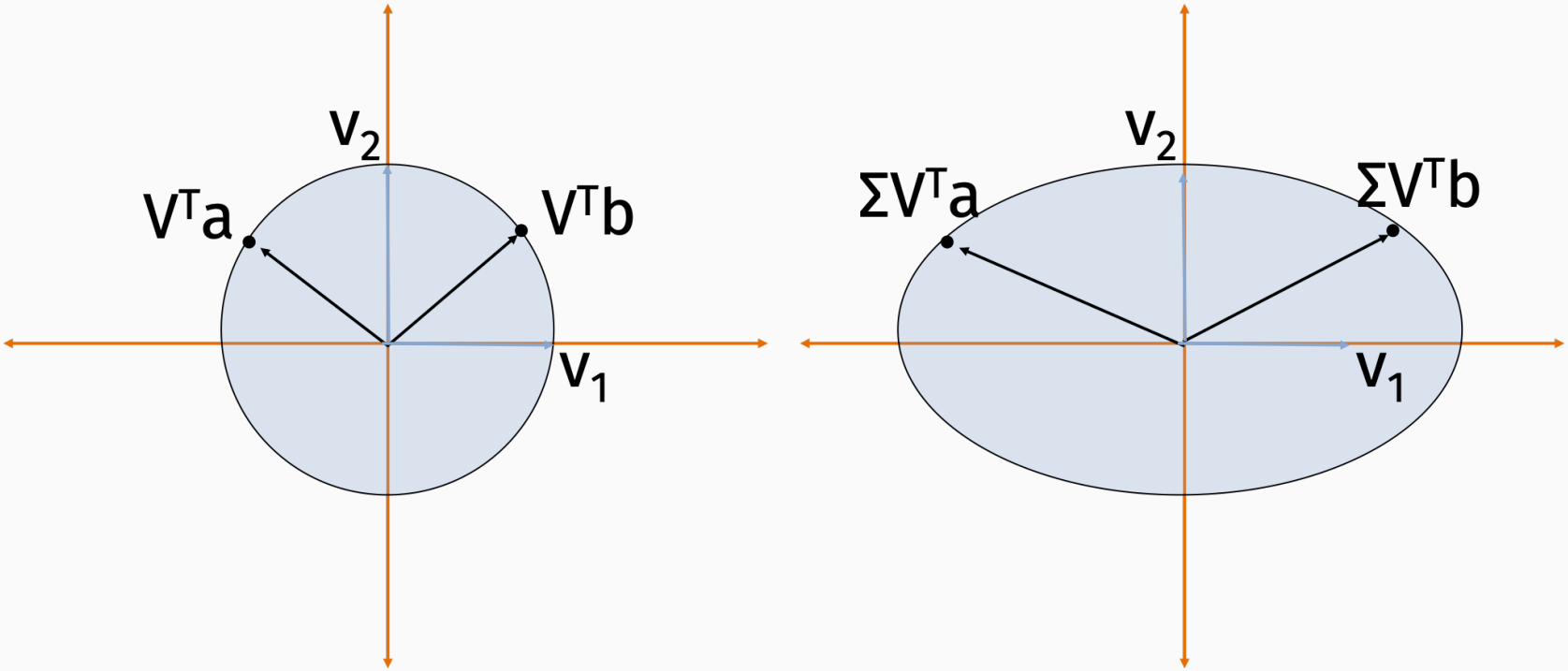
Multiplying any vector $\mathbf{a}$ by a matrix $\mathbf{X}$ to form $\mathbf{Xa}$ can be viewed as a composition of 3 operations:

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 $\mathbf{U}$).

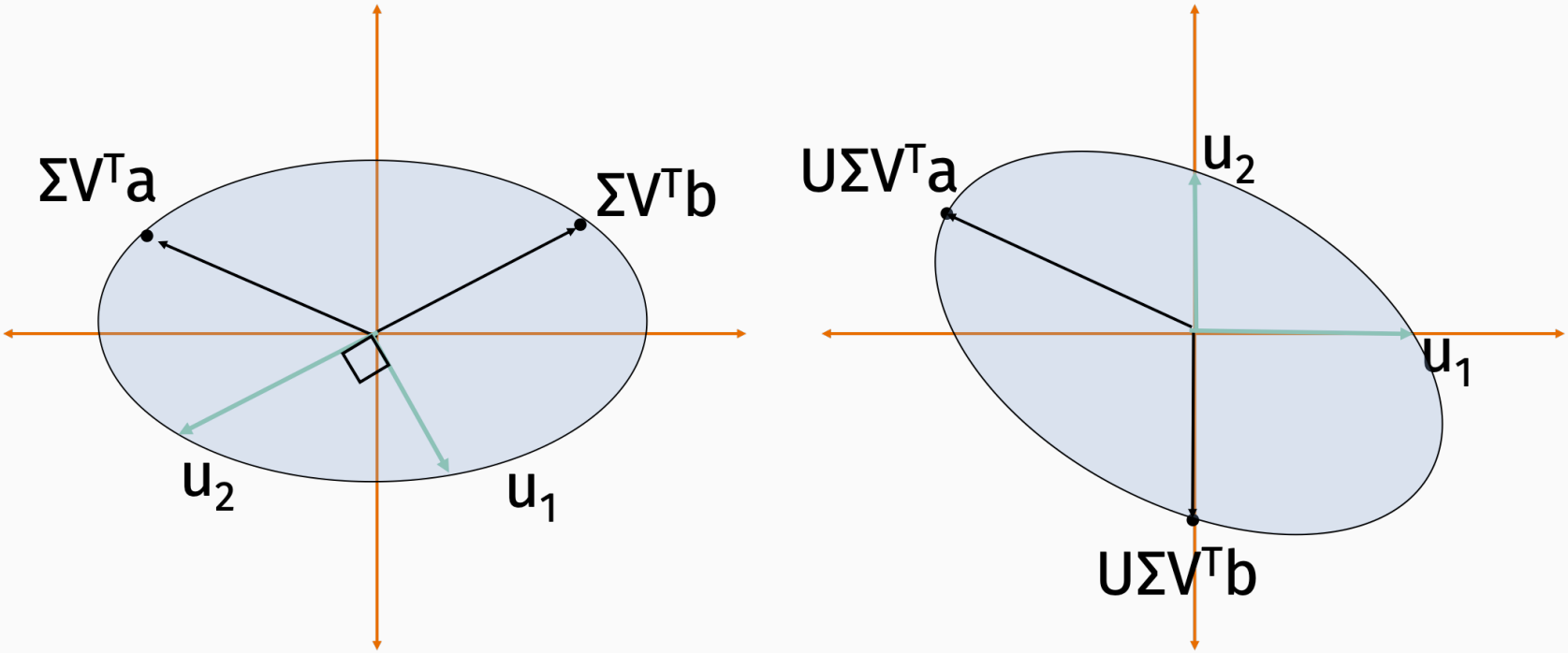
SINGULAR VALUE DECOMPOSITION: ROTATE/REFLECT



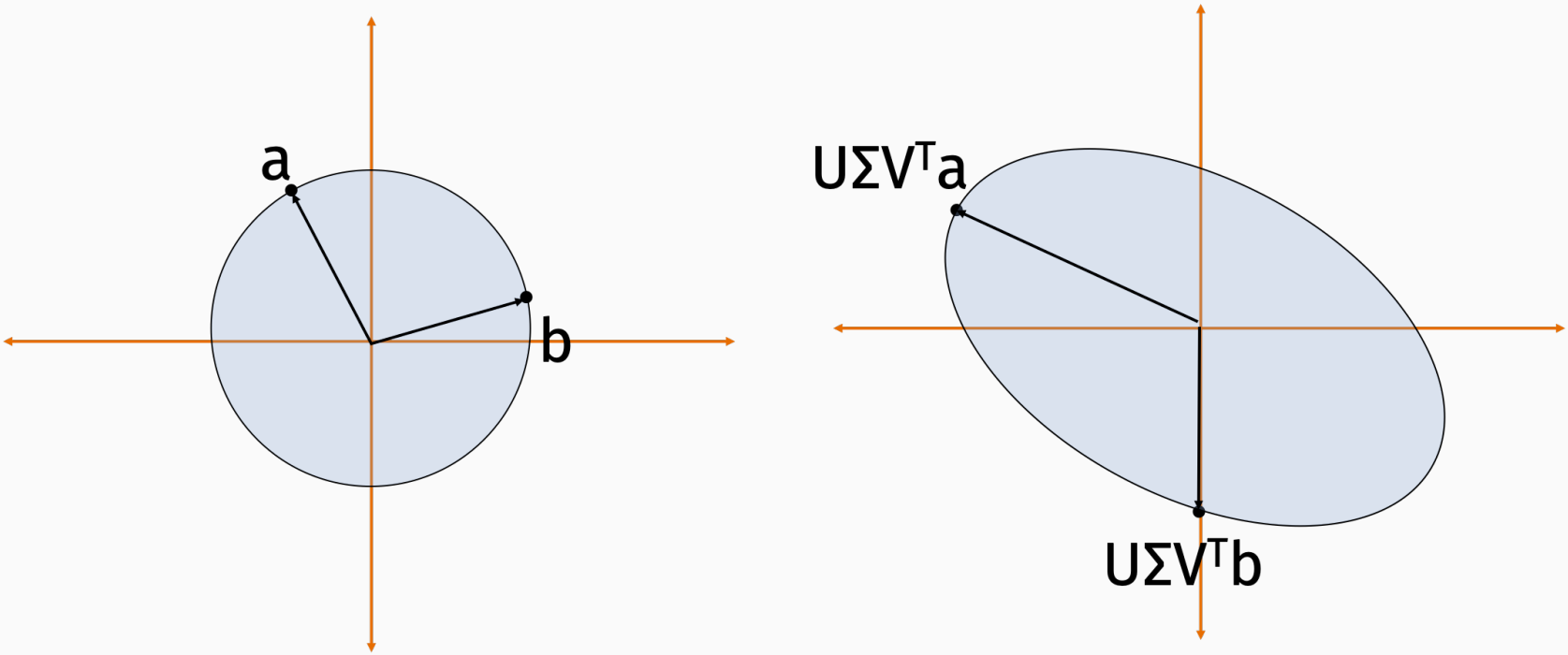
SINGULAR VALUE DECOMPOSITION: STRETCH



SINGULAR VALUE DECOMPOSITION: ROTATE/REFLECT



SINGULAR VALUE DECOMPOSITION



COMPARISON TO EIGENDECOMPOSITION

Recall that an eigenvalue of a square matrix $\mathbf{X} \in \mathbb{R}^{d \times d}$ is any vector $\mathbf{v}$ such that $\mathbf{X}\mathbf{v} = \lambda\mathbf{v}$. A 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 $\lambda_1, \ldots, \lambda_n$ it is called "diagonalizable" and can be written as:

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

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 or negative.
- 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 $\mathbf{X}^T\mathbf{X}$.
- $\sigma_i^2 = \lambda_i(\mathbf{X}\mathbf{X}^T) = \lambda_i(\mathbf{X}^T\mathbf{X})$

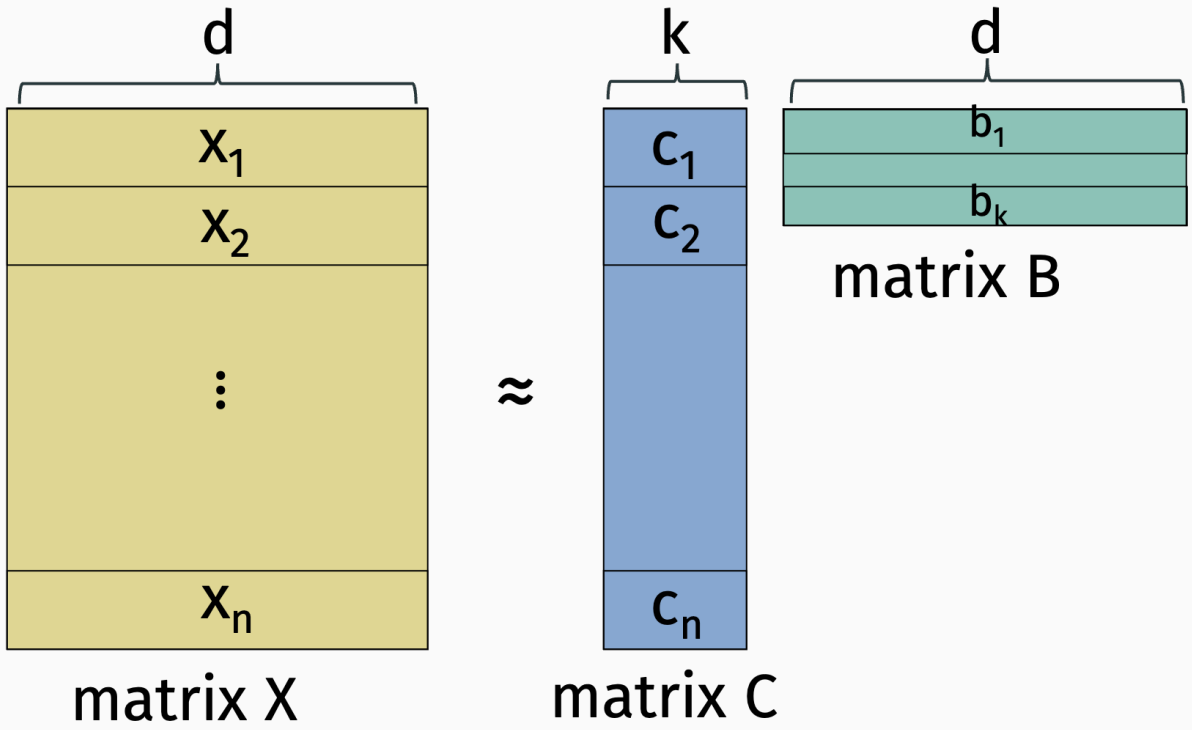
SVD APPLICATIONS

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 approximations for $\mathbf{X}$.

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

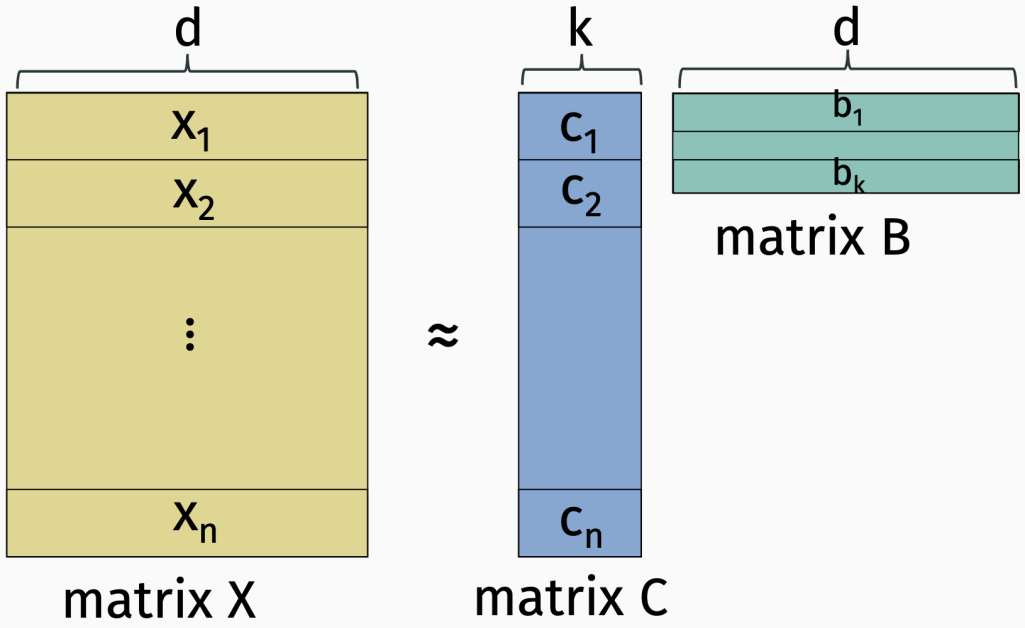


Typically choose $\mathbf{C}$ and $\mathbf{B}$ to minimize:

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

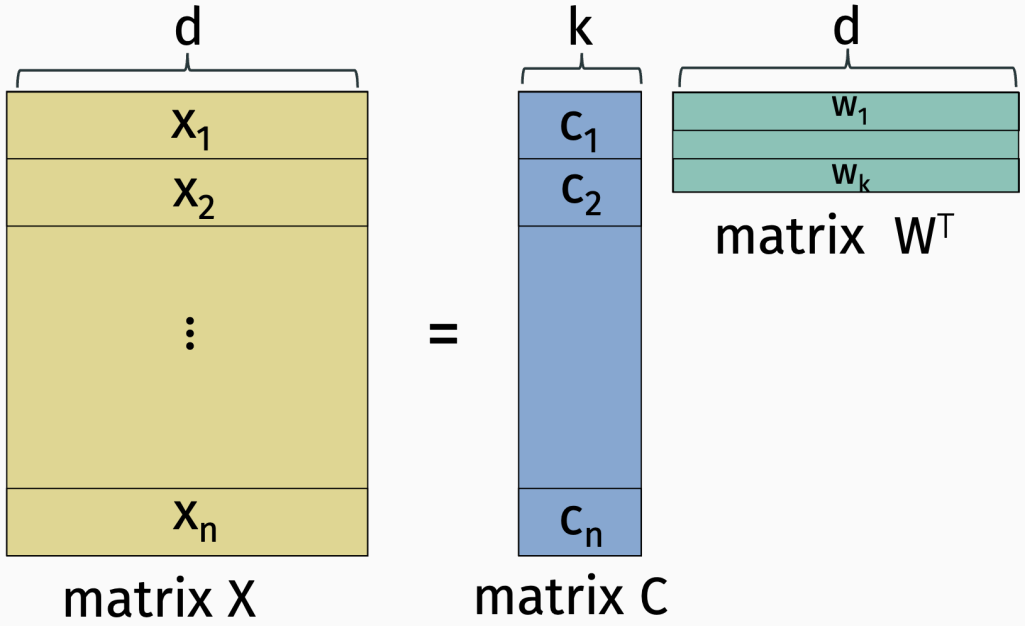
for some matrix norm. Common choice is $\|\mathbf{X} - \mathbf{CB}\|_F^2$.

APPLICATIONS OF LOW-RANK APPROXIMATION



- **CB** takes $O(k(n+d))$ space to store instead of $O(nd)$.
- Regression problems involving **CB** can be solved in $O(nk^2)$ instead of $O(nd^2)$ time.
- Will see a bunch more in a minute.

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



Then we should choose C to minimize:

$$\min_C \|X - CW^T\|_F^2$$

This is just n least squares regression problems!

LOW-RANK APPROXIMATION

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

$$c_i = W^T x_i$$

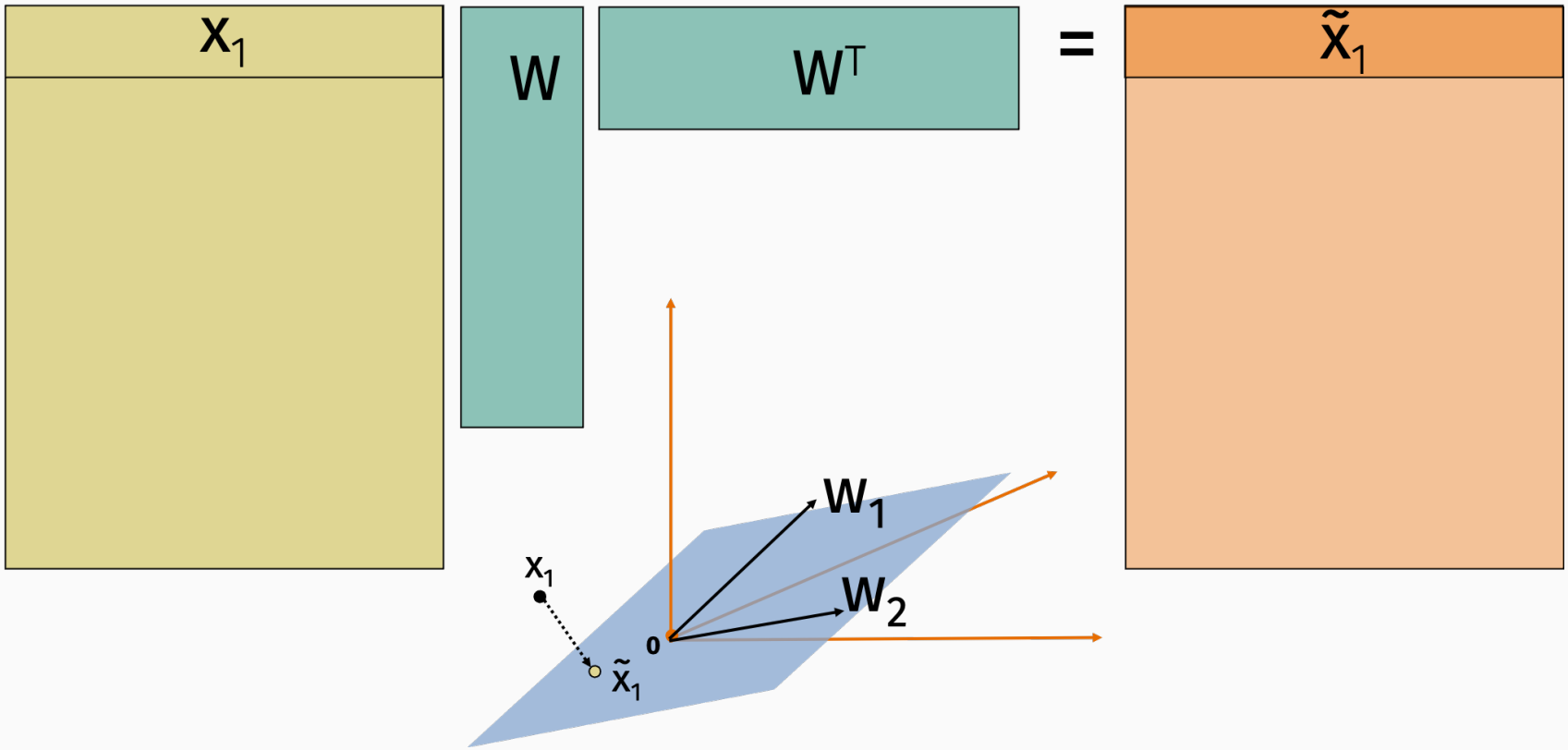
$$C = XW$$

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

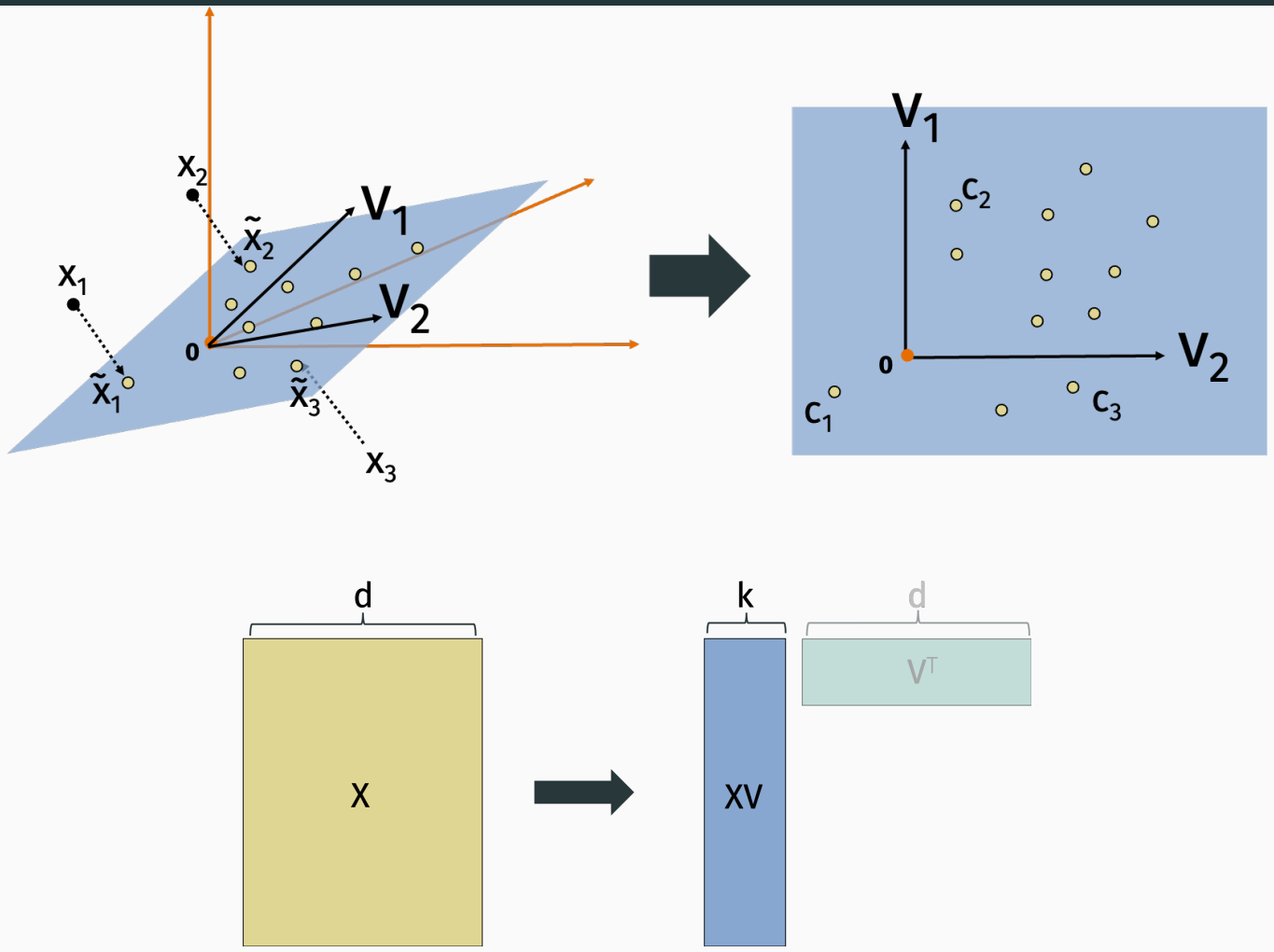
$$X \approx XWW^T$$

PROJECTION MATRICES

WW^T is a symmetric projection matrix.



LOW-RANK APPROXIMATION



$C = XW$ can be used as a compressed version of data matrix X.

Let $\mathbf{C} = \mathbf{XW}$. We have that:

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

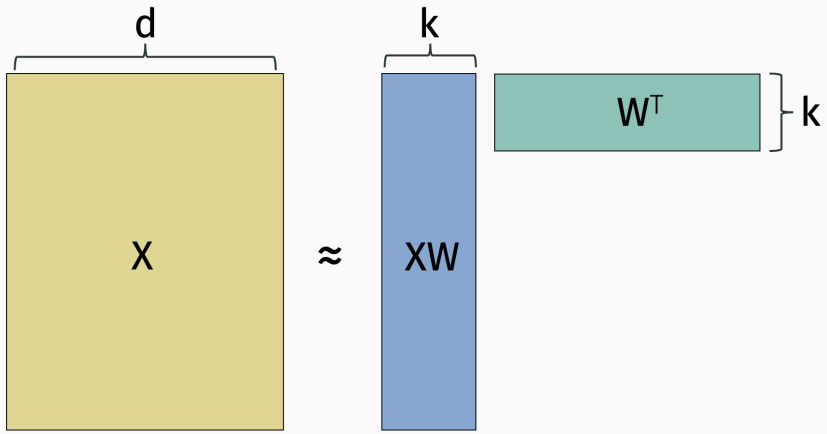
Similarly, we expect that:

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

How does this compare to Johnson-Lindenstrauss projection?

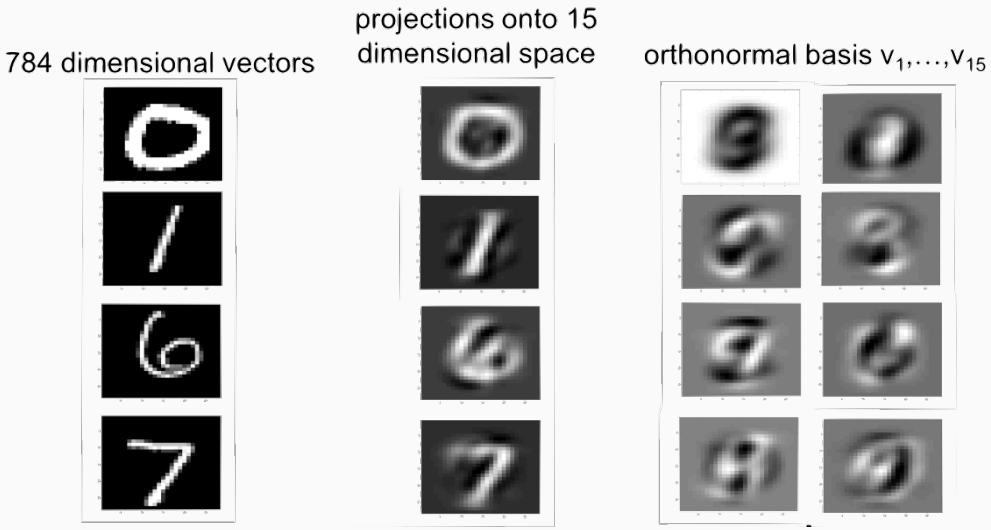
WHY IS DATA APPROXIMATELY LOW-RANK?

Rows of $\mathbf{X}$ (data points) are approximately spanned by k vectors. Columns of $\mathbf{X}$ (data features) are approximately spanned by k vectors.



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.



COLUMN REDUNDANCY

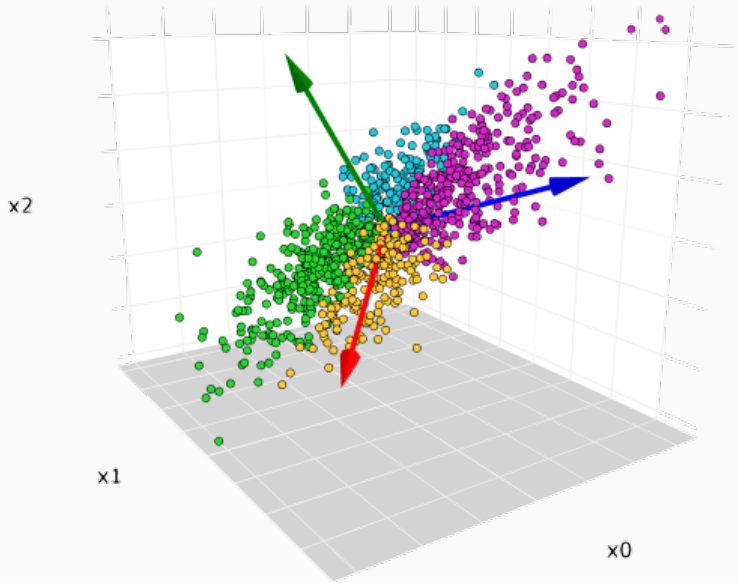
Colinearity/correlation of data features leads to a low-rank data matrix.

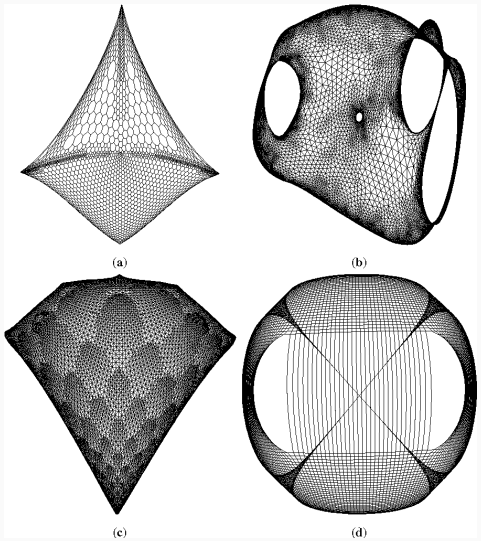
	bedrooms	bathrooms	sq.ft.	floors	list price	sale price
home 1	2	2	1800	2	200,000	195,000
home 2	4	2.5	2700	1	300,000	310,000
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
home n	5	3.5	3600	3	450,000	450,000

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 dependence methods for nearest neighbor search.
- Data visualization when $k = 2$ or 3.

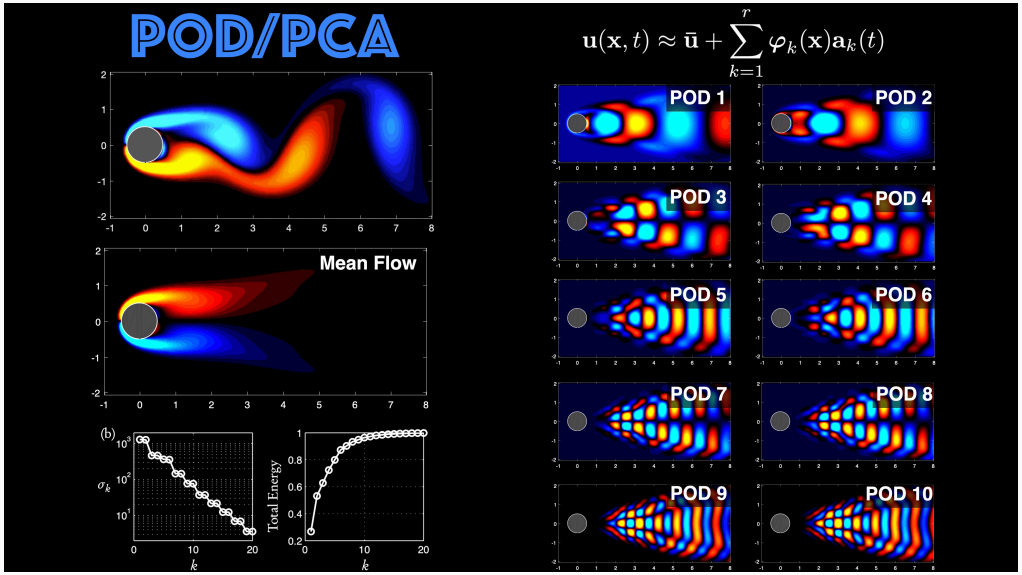




- Entity embeddings (next lecture).

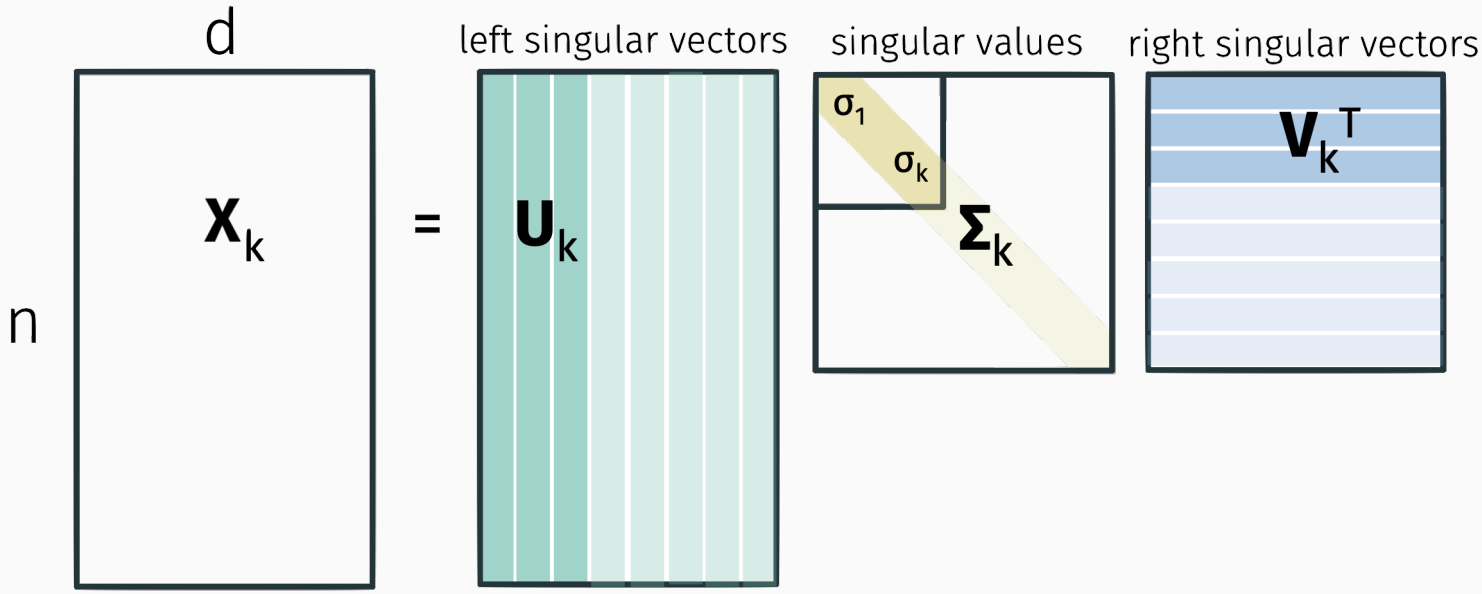
APPLICATIONS OF LOW-RANK APPROXIMATION

- Reduced order modeling for solving physical equations.



- Constructing preconditioners in optimization.
- Many more.

Can find the best projection from the singular value decomposition.



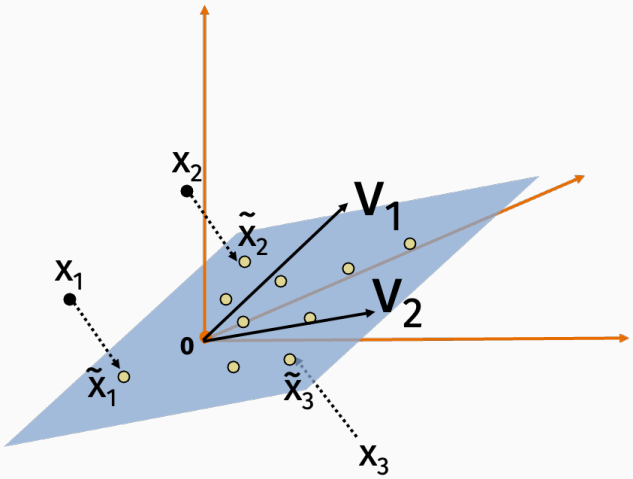
$$\mathbf{V}_k = \underset{\text{orthogonal } \mathbf{W} \in \mathbb{R}^{d \times k}}{\arg\min} \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2$$

OPTIMAL LOW-RANK APPROXIMATION

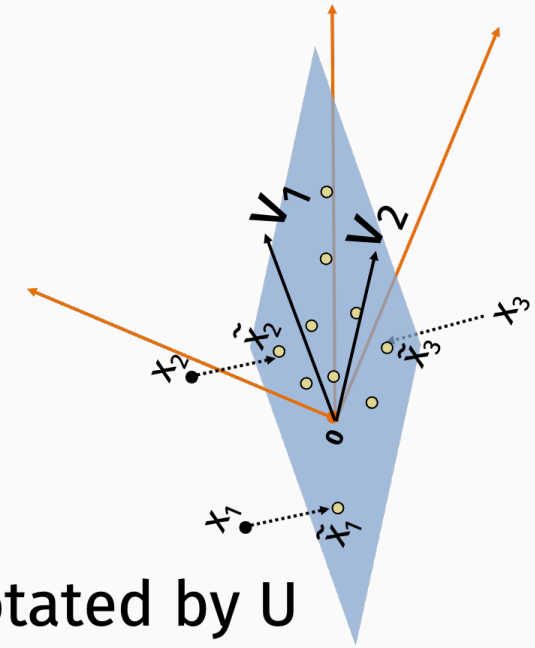
Claim: $X_k = U_k \mathbf{\Sigma}_k V_k^T = X V_k V_k^T$.

Claim 1:

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



ΣV^T



Rotated by U on the left

Claim 2:

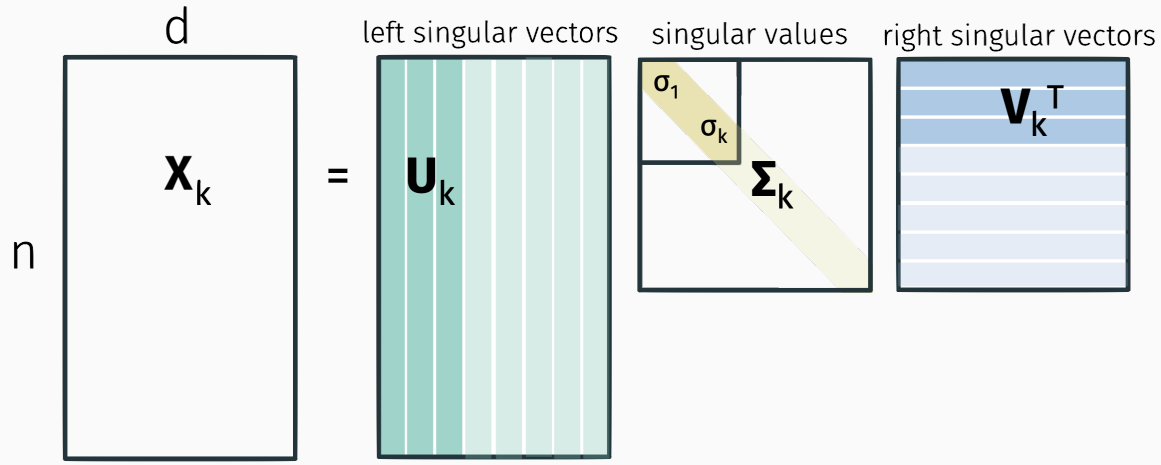
$$\underset{\text{rank } k\ \mathbf{B}}{\arg\min} \|\mathbf{\Sigma}\mathbf{V}^T - \mathbf{B}\|_F^2 = \underset{\text{rank } k\ \mathbf{B}}{\arg\min} \|\mathbf{V}\mathbf{\Sigma} - \mathbf{B}^T\|_F^2$$

Claim 3:

$$\underset{\text{rank } k\ \mathbf{B}}{\arg\min} \|\mathbf{V}\mathbf{\Sigma} - \mathbf{B}^T\|_F^2 = \underset{\text{rank } k\ \mathbf{B}}{\arg\min} \|\mathbf{\Sigma} - \mathbf{V}^T\mathbf{B}^T\|_F^2$$

Chose $\mathbf{B}^T$ so that $\mathbf{V}^T\mathbf{B}^T = \mathbf{\Sigma}_k$.

USEFUL OBSERVATIONS



Observation 1:

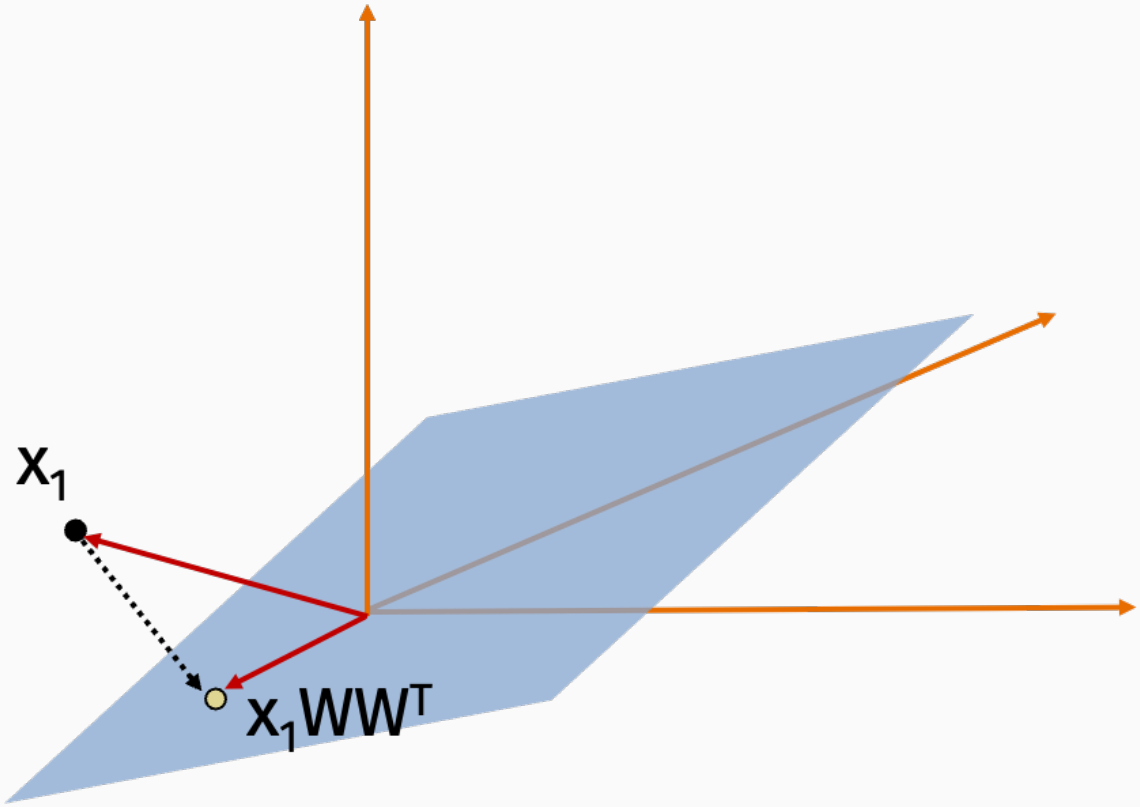
$$\arg\min_{\mathbf{W}\in\mathbb{R}^{d\times k}} \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2 = \arg\max_{\mathbf{W}\in\mathbb{R}^{d\times k}} \|\mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2$$

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

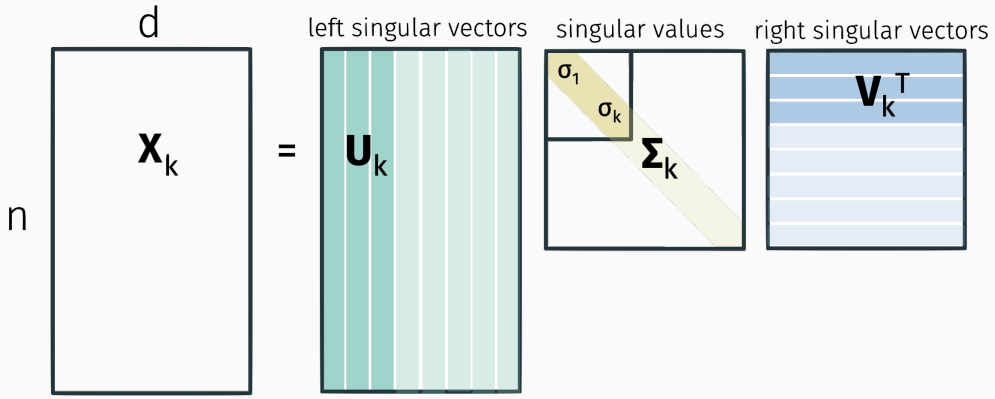
$$\|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}\mathbf{W}\mathbf{W}^T\|_F^2$$

Claim:

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



USEFUL OBSERVATIONS



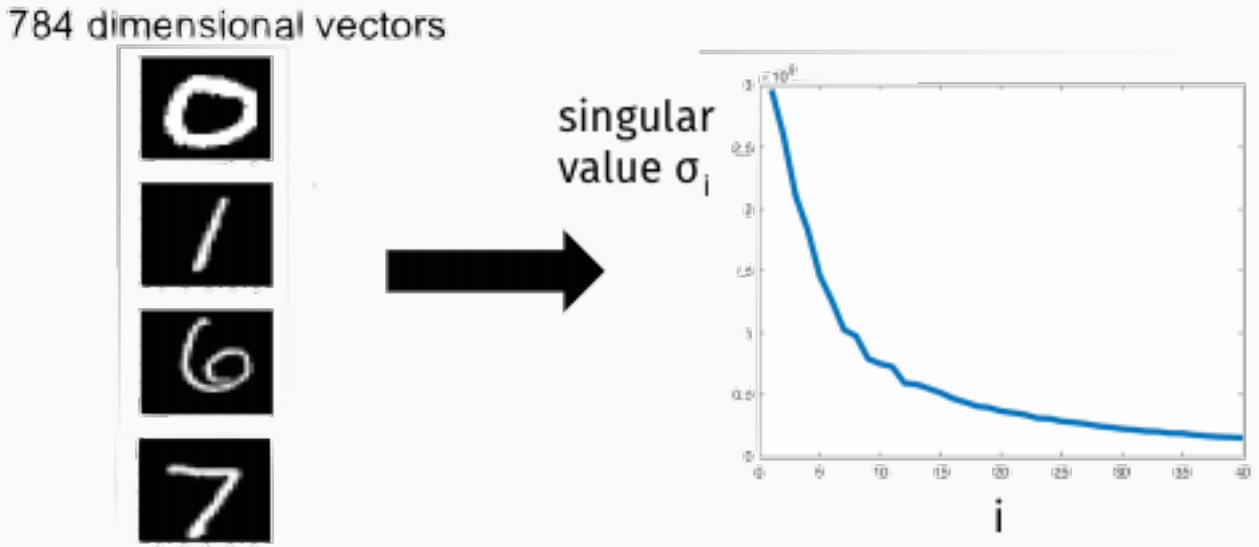
Observation 2: The optimal low-rank approximation error $E_k = \|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2$ can be written:

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

Observation 2: The optimal low-rank approximation error $E_k = \|X - XV_kV_k^T\|_F^2 = \|X\|_F^2 - \|XV_kV_k^T\|_F^2$ can be written:

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

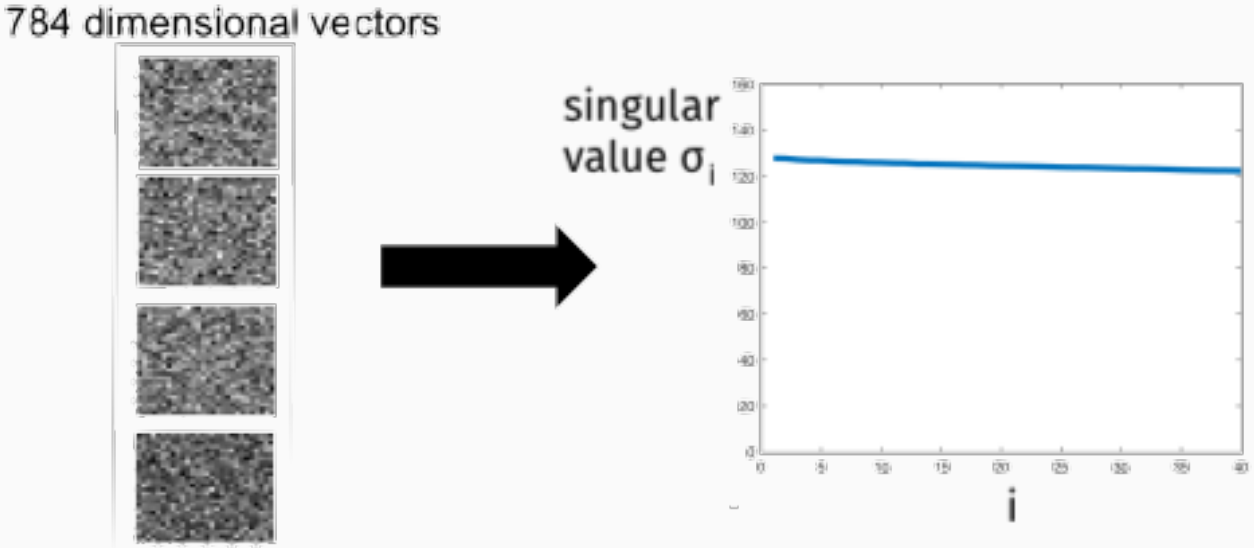
Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



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

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

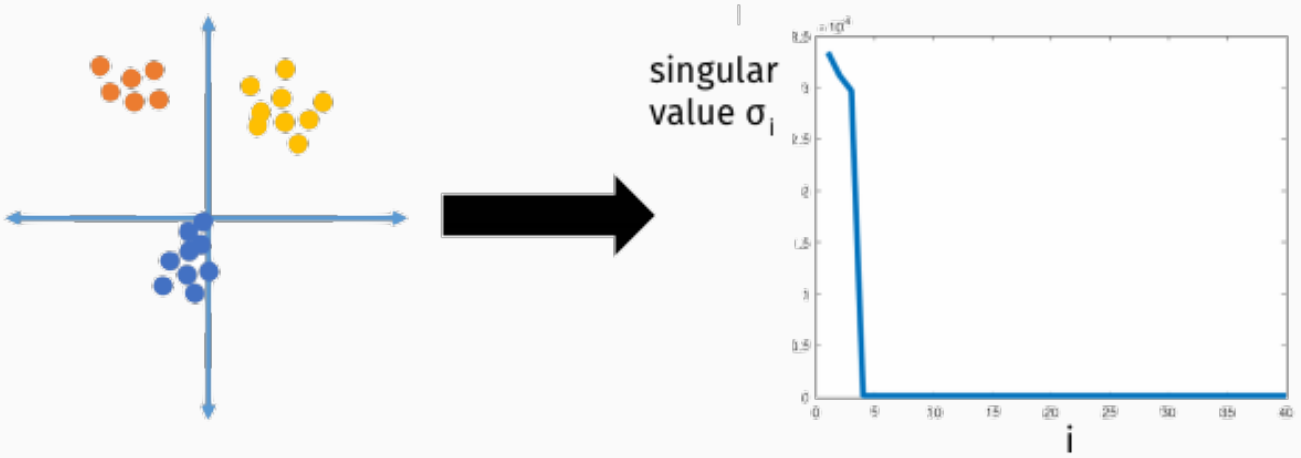
Can immediately get a sense of "how low-rank" a matrix is from it's spectrum:



Observation 2: The optimal low-rank approximation error $E_k = \|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2 = \|\mathbf{X}\|_F^2 - \|\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2$ 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 $\approx$

COMPUTING THE SVD (FASTER)

- Compute approximate solution.
- Only compute top k singular vectors/values. Runtime will depend on k. When $k = d$ we can't do any better than classical algorithms based on eigendecomposition.
- 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.

What we won't discuss today: sketching methods and stochastic methods (which are faster in some settings).

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 $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.

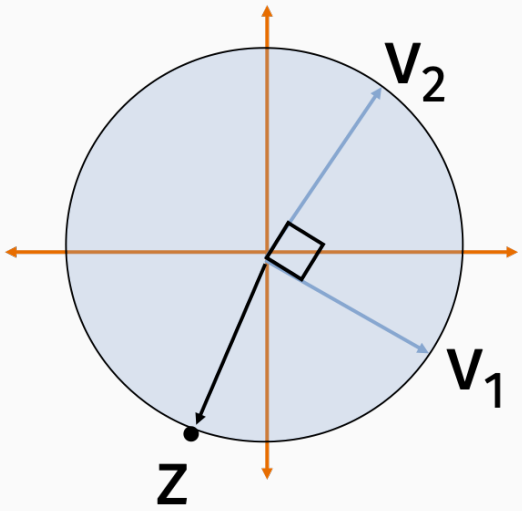
Power method:

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

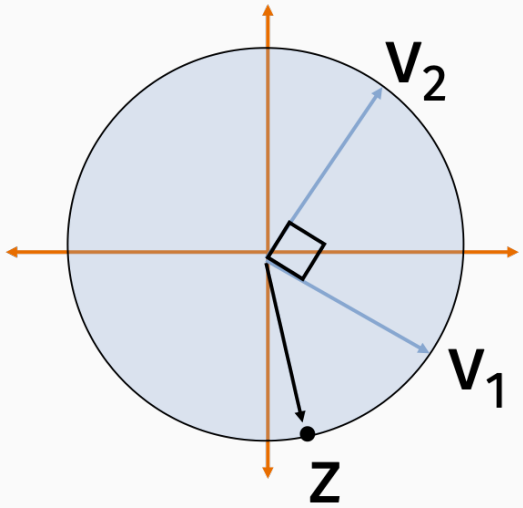
  Return $\mathbf{z}^{(T)}$

POWER METHOD INTUITION

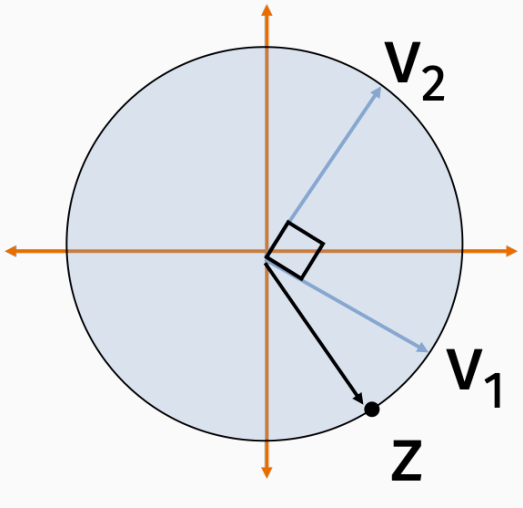
0 iterations



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

Also: $\|\mathbf{c}^{(i)}\|_2^2 = \sum_{j=1}^{d} \left(c_j^{(i)}\right)^2 = 1$.

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

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

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

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 $\left|\frac{\alpha_j}{\alpha_1}\right| \leq \sqrt{\frac{\epsilon}{d}}$ then:

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

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

$$\alpha_1^2 \geq 1 - \epsilon$$

$$|\alpha_1| \geq 1 - \epsilon$$

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

Lets proves that $\left|\frac{\alpha_j}{\alpha_1}\right| \leq \sqrt{\frac{\epsilon}{d}}$ where $\alpha_j = \frac{1}{\prod_{i=1}^{T} n_i} c_j^{(0)} \sigma_j^{2T}$

Assumption: Starting coefficients are all roughly equal.

$$\text{For all } j \qquad O(1/d^{1.5}) \leq \left|c_j^{(0)}\right| \leq 1.$$

This is a very loose bound, but it's all that we will need. We will prove shortly that it holds with probability 99/100.

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

$$\text{Need } T =$$

Need to prove: Starting coefficients are all roughly equal.

$$\text{For all } j \qquad\qquad O(1/d^{1.5}) \leq |c_j^{(0)}| \leq 1$$

with probability 99/100. **Prove using Gaussian (anti)-concentration.**

Right hand side is immediate from fact that $\sum_j (c_j^{(0)})^2 = 1$.

To show the left hand side we 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, every entry of $\frac{\mathbf{g}}{\|\mathbf{g}\|_2} \geq c \cdot \frac{1}{d^{1.5}}$.

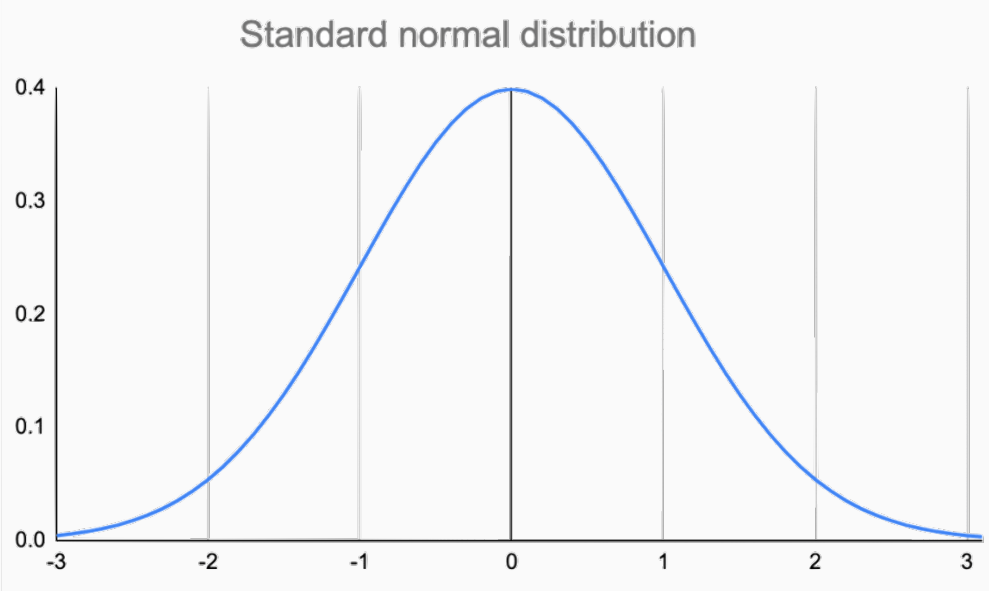
Part 1: With probablility 999/100,

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

Need to show that with high probability, the magnitude of every entry of $\mathbf{g} \geq c \cdot \frac{1}{d}$.

Part 2: With probablility $1 - c/d$,

$$\min_i |g_i| \geq O\left(\frac{c}{d}\right).$$



Applying union bound completes the result.

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

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

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

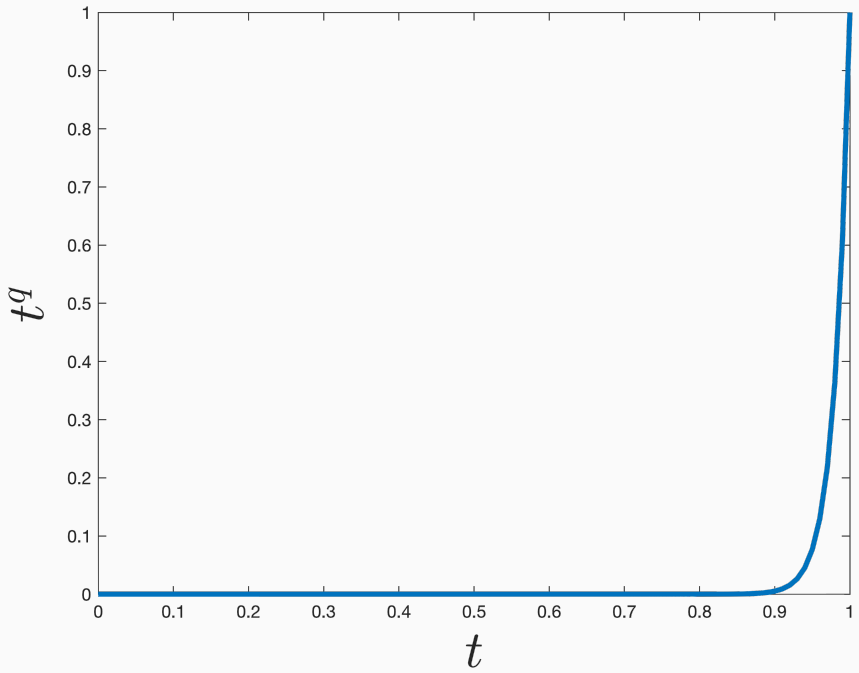
Possible to “accelerate” these methods.

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

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

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

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



$$\mathbf{z}^{(q)} = c \cdot \left[c_1 \cdot \sigma_1^{2q}\mathbf{v}_1 + c_2 \cdot \sigma_2^{2q}\mathbf{v}_2 + \ldots + c_n \cdot \sigma_n^{2q}\mathbf{v}_n\right]$$

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

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 $\left(\mathbf{X}^T\mathbf{X}\right)^q \cdot \mathbf{g}$. What you're using when you run `svds` or `eigs` in MATLAB or Python.

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.

LANCZOS METHOD ANALYSIS

For a degree t polynomial p, let $\mathbf{v}_p = \frac{p(\mathbf{X}^T\mathbf{X})\mathbf{g}}{\|p(\mathbf{X}^T\mathbf{X})\mathbf{g}\|_2}$.

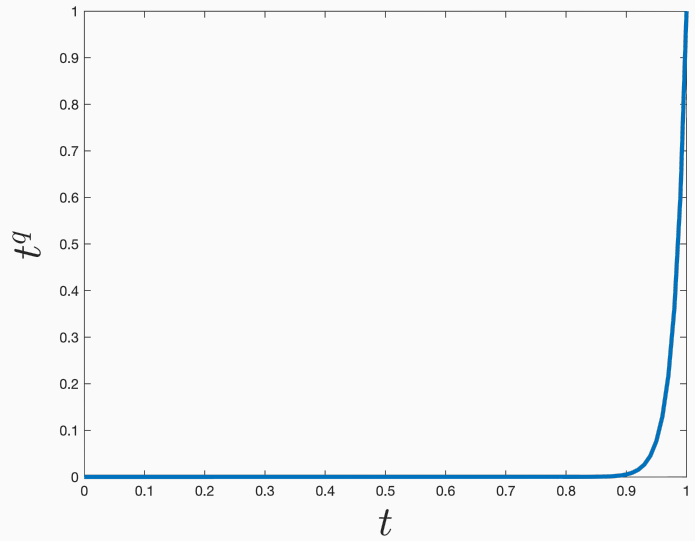
Power method returns:

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

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 $\mathbf{x}^q$ up to error $\Delta\sigma_1^2$ 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: $O\left(\frac{\log(d/\epsilon)}{\sqrt{\gamma}} \cdot \text{nnz}(\mathbf{X})\right)$ vs. $O\left(\frac{\log(d/\epsilon)}{\gamma} \cdot \text{nnz}(\mathbf{X})\right)$

POWER METHOD – NO GAP DEPENDENCE

Again convergence is slow when $\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$ is small. $\mathbf{z}^{(q)}$ has large components of both $\mathbf{v}_1$ and $\mathbf{v}_2$. But in this case:

$$\|\mathbf{X} - \mathbf{X}\mathbf{v}_1\mathbf{v}_1^T\|_F^2 = \sum_{i \neq 1} \sigma_i^2 \approx \sum_{i \neq 2} = \sigma_i^2 \|\mathbf{X} - \mathbf{X}\mathbf{v}_2\mathbf{v}_2^T\|_F^2.$$

So we don't care! Either $\mathbf{v}_1$ or $\mathbf{v}_2$ give good rank-1 approximations.

Claim: To achieve

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

we need $O\left(\frac{\log(d/\epsilon)}{\epsilon}\right)$ power method iterations or $O\left(\frac{\log(d/\epsilon)}{\sqrt{\epsilon}}\right)$ Lanczos iterations.

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