COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Fall 2019.

Lecture 13

LOGISTICS

- Pass/Fail Deadline is 10/29 for undergraduates and 10/31 for graduates. We will have your Problem Set 2 and midterm grades back before then.
- Will release Problem Set 3 next week due $\sim$ 11/11.

- Pass/Fail Deadline is 10/29 for undergraduates and 10/31 for graduates. We will have your Problem Set 2 and midterm grades back before then.
- Will release Problem Set 3 next week due $\sim$ 11/11.
- MAP Feedback:
    - Going to adjust a bit how I take questions in class.
    - Will try to more clearly identify important information (what will appear on exams or problem sets) v.s. motivating examples.
    - Will try to use iPad more to write out proofs in class.

Last Few Classes: Low-Rank Approximation and PCA

Last Few Classes: Low-Rank Approximation and PCA

- Discussed how to compress a dataset that lies close to a k-dimensional subspace.
- Optimal compression by projecting onto the top k eigenvectors of the covariance matrix X^TX (PCA).
- Saw how to calculate the error of the approximation – interpret the spectrum of X^TX.

Last Few Classes: Low-Rank Approximation and PCA

- Discussed how to compress a dataset that lies close to a k-dimensional subspace.
- Optimal compression by projecting onto the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ (PCA).
- Saw how to calculate the error of the approximation – interpret the spectrum of $\mathbf{X}^T\mathbf{X}$.

This Class: Low-rank approximation and connection to singular value decomposition.

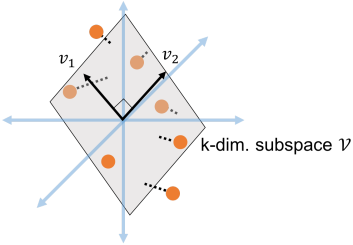
Last Few Classes: Low-Rank Approximation and PCA

- Discussed how to compress a dataset that lies close to a k-dimensional subspace.
- Optimal compression by projecting onto the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ (PCA).
- Saw how to calculate the error of the approximation – interpret the spectrum of $\mathbf{X}^T\mathbf{X}$.

This Class: Low-rank approximation and connection to singular value decomposition.

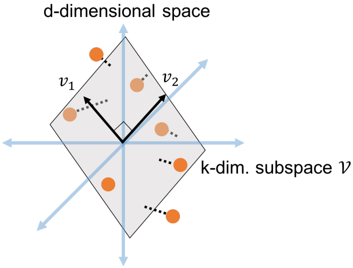
- Show how PCA can be interpreted in terms of the singular value decomposition (SVD) of $\mathbf{X}$.
- Applications to word embeddings, graph embeddings, document classification, recommendation systems.

Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.



$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

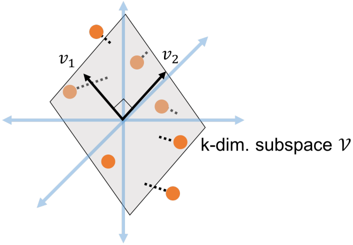
Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.



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

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.

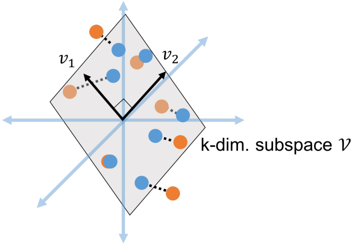


Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

- $\mathbf{V}\mathbf{V}^T \in \mathbb{R}^{d \times d}$ is the projection matrix onto $\mathcal{V}$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.

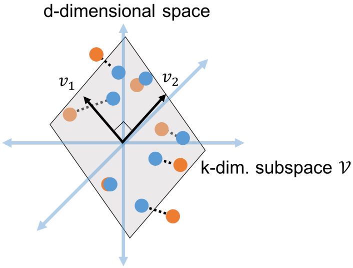


Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

- $\mathbf{V}\mathbf{V}^T \in \mathbb{R}^{d \times d}$ is the projection matrix onto $\mathcal{V}$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.

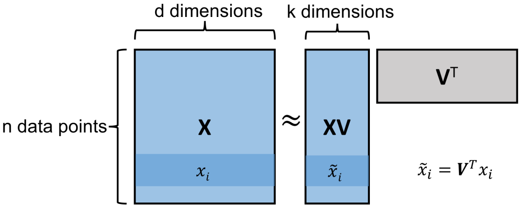


Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

- $\mathbf{V}\mathbf{V}^T \in \mathbb{R}^{d \times d}$ is the projection matrix onto $\mathcal{V}$.
- $\mathbf{X} \approx \mathbf{X}(\mathbf{V}\mathbf{V}^T)$. Gives the closest approximation to $\mathbf{X}$ with rows in $\mathcal{V}$.

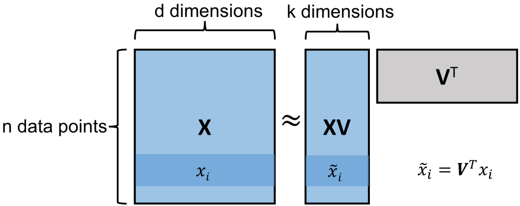
$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Low-Rank Approximation: Approximate $\mathbf{X} \approx \mathbf{X}\mathbf{V}\mathbf{V}^T$.



$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

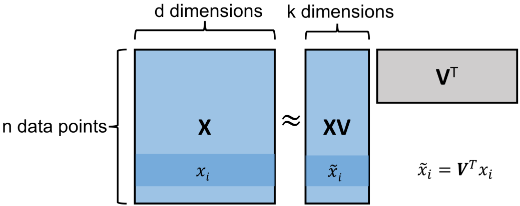
Low-Rank Approximation: Approximate $\mathbf{X} \approx \mathbf{X}\mathbf{V}\mathbf{V}^T$.



- $\mathbf{X}\mathbf{V}\mathbf{V}^T$ is a rank-k matrix – all its rows fall in $\mathcal{V}$.

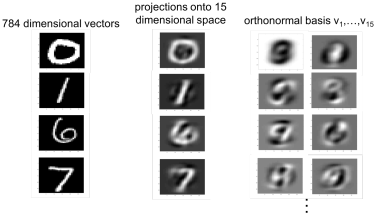
$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Low-Rank Approximation: Approximate $\mathbf{X} \approx \mathbf{X}\mathbf{V}\mathbf{V}^T$.



- $\mathbf{X}\mathbf{V}\mathbf{V}^T$ is a rank-k matrix – all its rows fall in $\mathcal{V}$.
- $\mathbf{X}$'s rows are approximately spanned by the columns of $\mathbf{V}$.
- $\mathbf{X}$'s columns are approximately spanned by the columns of $\mathbf{X}\mathbf{V}$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.



Row (data point) compression

Column (feature) compression

10000* bathrooms+ 10* (sq. ft.) ≈ list price

	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

Given $\vec{x}_1, \ldots, \vec{x}_n$ (the rows of $\mathbf{X}$) we want to find an orthonormal span $\mathbf{V} \in \mathbb{R}^{d \times k}$ (spanning a k-dimensional subspace $\mathcal{V}$).

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Given $\vec{x}_1, \ldots, \vec{x}_n$ (the rows of $\mathbf{X}$) we want to find an orthonormal span $\mathbf{V} \in \mathbb{R}^{d \times k}$ (spanning a k-dimensional subspace $\mathcal{V}$).

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

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Given $\vec{x}_1, \ldots, \vec{x}_n$ (the rows of $\mathbf{X}$) we want to find an orthonormal span $\mathbf{V} \in \mathbb{R}^{d \times k}$ (spanning a k-dimensional subspace $\mathcal{V}$).

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

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Given $\vec{x}_1, \ldots, \vec{x}_n$ (the rows of $\mathbf{X}$) we want to find an orthonormal span $\mathbf{V} \in \mathbb{R}^{d \times k}$ (spanning a k-dimensional subspace $\mathcal{V}$).

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\min} \|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{n} \|\mathbf{V}\mathbf{V}^T \vec{x}_i\|_2^2$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

SOLUTION VIA EIGENDECOMPOSITION

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

SOLUTION VIA EIGENDECOMPOSITION

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

Surprisingly, can find the columns of $\mathbf{V}$, $\vec{v}_1, \ldots, \vec{v}_k$ greedily.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

SOLUTION VIA EIGENDECOMPOSITION

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

Surprisingly, can find the columns of $\mathbf{V}$, $\vec{v}_1, \ldots, \vec{v}_k$ greedily.

$$\vec{v}_1 = \underset{\vec{v} \text{ with } \|v\|_2 = 1}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

Surprisingly, can find the columns of $\mathbf{V}$, $\vec{v}_1, \ldots, \vec{v}_k$ greedily.

$$\vec{v}_1 = \underset{\vec{v} \text{ with } \|v\|_2 = 1}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$$\vec{v}_2 = \underset{\vec{v} \text{ with } \|v\|_2 = 1,\ \langle \vec{v}, \vec{v}_1 \rangle = 0}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

Surprisingly, can find the columns of $\mathbf{V}$, $\vec{v}_1, \ldots, \vec{v}_k$ greedily.

$$\vec{v}_1 = \underset{\vec{v} \text{ with } \|v\|_2 = 1}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$$\vec{v}_2 = \underset{\vec{v} \text{ with } \|v\|_2 = 1,\ \langle \vec{v}, \vec{v}_1 \rangle = 0}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$$\cdots$$

$$\vec{v}_k = \underset{\vec{v} \text{ with } \|v\|_2 = 1,\ \langle \vec{v}, \vec{v}_j \rangle = 0\ \forall j < k}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

SOLUTION VIA EIGENDECOMPOSITION

$\mathbf{V}$ minimizing the error $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2 = \sum_{i=1}^{k} \vec{v}_i^T \mathbf{X}^T \mathbf{X} \vec{v}_i$$

Surprisingly, can find the columns of $\mathbf{V}$, $\vec{v}_1, \ldots, \vec{v}_k$ greedily.

$$\vec{v}_1 = \underset{\vec{v} \text{ with } \|v\|_2 = 1}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

$$\vec{v}_2 = \underset{\vec{v} \text{ with } \|v\|_2 = 1,\ \langle \vec{v}, \vec{v}_1 \rangle = 0}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

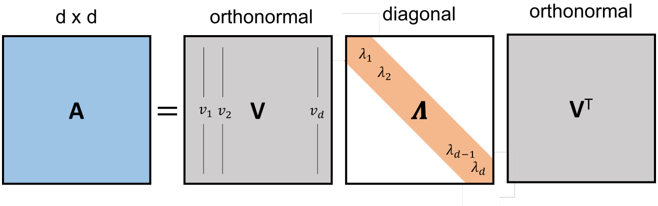
$$\cdots$$

$$\vec{v}_k = \underset{\vec{v} \text{ with } \|v\|_2 = 1,\ \langle \vec{v}, \vec{v}_j \rangle = 0\ \forall j < k}{\arg\max} \vec{v}^T \mathbf{X}^T \mathbf{X} \vec{v}.$$

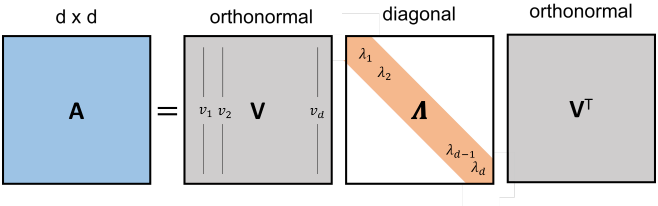
The top k eigenvectors of $\mathbf{X}^T\mathbf{X}$ by the Courant-Fischer Principal.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Any symmetric matrix $\mathbf{A}$ can be decomposed as $\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, where the columns $\mathbf{V}$ are d orthonormal eigenvectors $\vec{v}_1, \ldots, \vec{v}_d$.

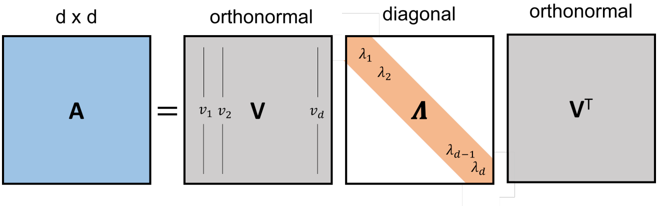


Any symmetric matrix $\mathbf{A}$ can be decomposed as $\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, where the columns $\mathbf{V}$ are d orthonormal eigenvectors $\vec{v}_1, \ldots, \vec{v}_d$.



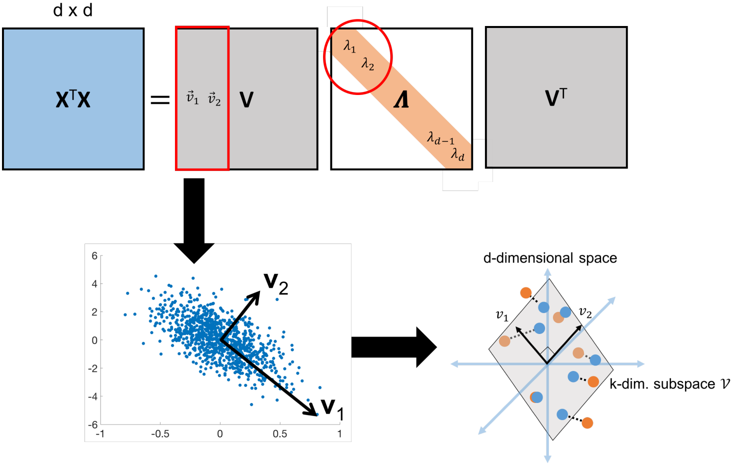
Typically order the eigenvalues in decreasing order: $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_d$.

Any symmetric matrix $\mathbf{A}$ can be decomposed as $\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, where the columns $\mathbf{V}$ are d orthonormal eigenvectors $\vec{v}_1, \ldots, \vec{v}_d$.



Typically order the eigenvalues in decreasing order: $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_d$.

When $\mathbf{A} = \mathbf{X}^T\mathbf{X}$ all eigenvalues are ≥ 0. Why?



$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $X \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of X^TX, $V_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Upshot: Letting $\mathbf{V}_k$ have columns $\vec{v}_1, \ldots, \vec{v}_k$ corresponding to the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k$ is the orthogonal basis minimizing

$$\|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2,$$

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Upshot: Letting $\mathbf{V}_k$ have columns $\vec{v}_1, \ldots, \vec{v}_k$ corresponding to the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k$ is the orthogonal basis minimizing

$$\|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2,$$

This is principal component analysis (PCA).

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

Upshot: Letting $\mathbf{V}_k$ have columns $\vec{v}_1, \ldots, \vec{v}_k$ corresponding to the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k$ is the orthogonal basis minimizing

$$\|\mathbf{X} - \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T\|_F^2,$$

This is principal component analysis (PCA).

Last Time: Saw how to determine accuracy by looking at the eigenvalues (the 'spectrum') of $\mathbf{X}^T\mathbf{X}$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

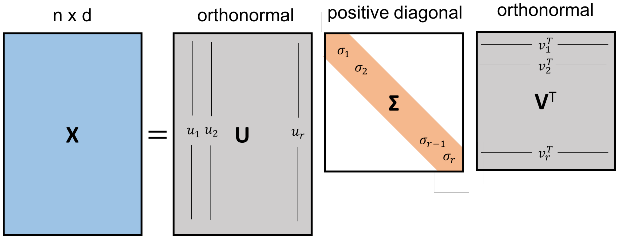
The Singular Value Decomposition (SVD) generalizes the eigendecomposition to asymmetric (even rectangular) matrices.

The Singular Value Decomposition (SVD) generalizes the eigendecomposition to asymmetric (even rectangular) matrices. Any matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $\text{rank}(\mathbf{X}) = r$ can be written as $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$.

- $\mathbf{U}$ has orthonormal columns $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ (left singular vectors).
- $\mathbf{V}$ has orthonormal columns $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ (right singular vectors).
- $\boldsymbol{\Sigma}$ is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ (singular values).

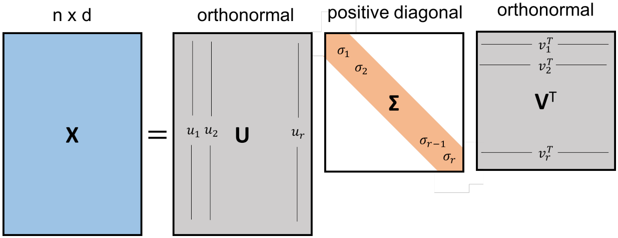
The Singular Value Decomposition (SVD) generalizes the eigendecomposition to asymmetric (even rectangular) matrices. Any matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $\text{rank}(\mathbf{X}) = r$ can be written as $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$.

- $\mathbf{U}$ has orthonormal columns $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ (left singular vectors).
- $\mathbf{V}$ has orthonormal columns $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ (right singular vectors).
- $\boldsymbol{\Sigma}$ is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ (singular values).



The Singular Value Decomposition (SVD) generalizes the eigendecomposition to asymmetric (even rectangular) matrices. Any matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $\text{rank}(\mathbf{X}) = r$ can be written as $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.

- $\mathbf{U}$ has orthonormal columns $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ (left singular vectors).
- $\mathbf{V}$ has orthonormal columns $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ (right singular vectors).
- $\mathbf{\Sigma}$ is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ (singular values).



The 'swiss army knife' of linear algebra.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} =$$

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\mathbf{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

CONNECTION OF THE SVD TO EIGENDECOMPOSITION

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T$$

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\mathbf{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{X}\mathbf{X}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{X}\mathbf{X}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.

The left and right singular vectors are the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ and the gram matrix $\mathbf{X}\mathbf{X}^T$ respectively.

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{X}\mathbf{X}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T$.

The left and right singular vectors are the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ and the gram matrix $\mathbf{X}\mathbf{X}^T$ respectively.

So, letting $\mathbf{V}_k \in \mathbb{R}^{d \times k}$ have columns equal to $\vec{v}_1, \ldots, \vec{v}_k$, we have that $\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T$ is the best rank-k approximation to $\mathbf{X}$ (given by PCA approximation).

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\mathbf{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U\Sigma V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V\Sigma U}^T\mathbf{U\Sigma V}^T = \mathbf{V\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{XX}^T = \mathbf{U\Sigma V}^T\mathbf{V\Sigma U}^T = \mathbf{U\Sigma}^2\mathbf{U}^T$.

The left and right singular vectors are the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ and the gram matrix $\mathbf{XX}^T$ respectively.

So, letting $\mathbf{V}_k \in \mathbb{R}^{d \times k}$ have columns equal to $\vec{v}_1, \ldots, \vec{v}_k$, we have that $\mathbf{XV}_k\mathbf{V}_k^T$ is the best rank-k approximation to $\mathbf{X}$ (given by PCA approximation).

What about $\mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$ where $\mathbf{U}_k \in \mathbb{R}^{n \times k}$ has columns equal to $\vec{u}_1, \ldots, \vec{u}_k$?

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\mathbf{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

CONNECTION OF THE SVD TO EIGENDECOMPOSITION

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{X}\mathbf{X}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.

The left and right singular vectors are the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ and the gram matrix $\mathbf{X}\mathbf{X}^T$ respectively.

So, letting $\mathbf{V}_k \in \mathbb{R}^{d \times k}$ have columns equal to $\vec{v}_1, \ldots, \vec{v}_k$, we have that $\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T$ is the best rank-k approximation to $\mathbf{X}$ (given by PCA approximation).

What about $\mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$ where $\mathbf{U}_k \in \mathbb{R}^{n \times k}$ has columns equal to $\vec{u}_1, \ldots, \vec{u}_k$?

Gives exactly the same approximation!

> $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T$$

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B}\in\mathbb{R}^{n\times d}} \|\mathbf{X}-\mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

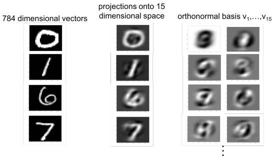
The best low-rank approximation to $\mathbf{X}$:

$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$

Row (data point) compression



Column (feature) compression

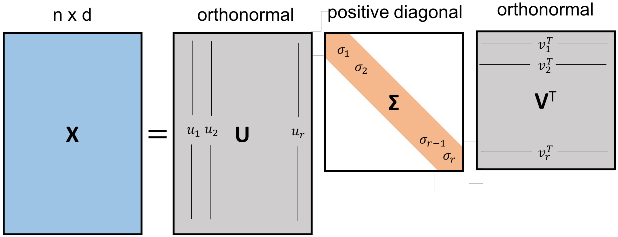
10000* bathrooms+ 10* (sq. ft.) ≈ list price

	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

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$

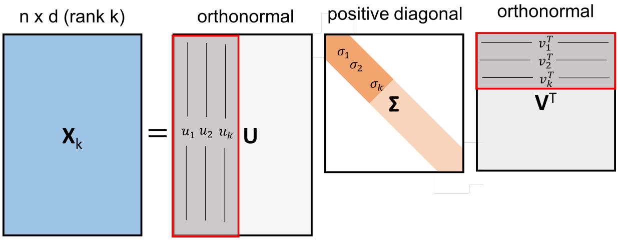


The best low-rank approximation to $\mathbf{X}$:

$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

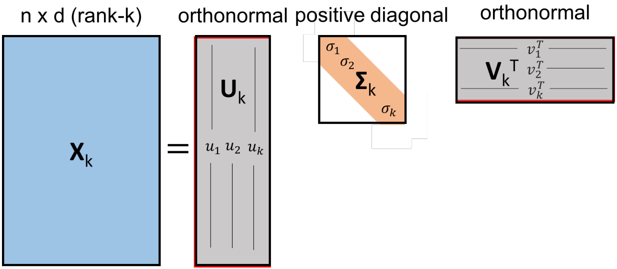
Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$



The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X} = \mathbf{U}_k\boldsymbol{\Sigma}_k\mathbf{V}_k^T$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$



The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B} \in \mathbb{R}^{n \times d}} \|\mathbf{X} - \mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X} = \mathbf{U}_k\boldsymbol{\Sigma}_k\mathbf{V}_k^T$$

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\mathbf{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

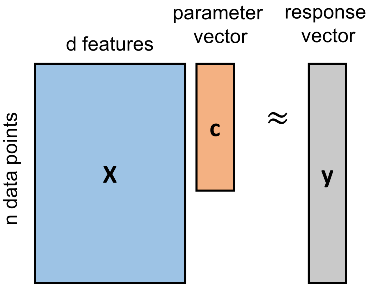
SVD is a ‘swiss army knife’.

SVD is a ‘swiss army knife’.

Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters), and response vector $\vec{y} \in \mathbb{R}^d$, want to find $\vec{c} \in \mathbb{R}^d$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.

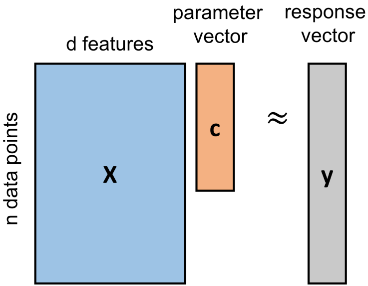
SVD is a ‘swiss army knife’.

Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters), and response vector $\vec{y} \in \mathbb{R}^d$, want to find $\vec{c} \in \mathbb{R}^d$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.



SVD is a 'swiss army knife'.

Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters), and response vector $\vec{y} \in \mathbb{R}^d$, want to find $\vec{c} \in \mathbb{R}^d$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.



E.g., $c_1 \cdot (\#\ \textit{baths}) + c_2 \cdot (\textit{sq.ft.}) + c_3 \cdot (\#\ \textit{floors}) + \ldots \approx \textit{home price}$

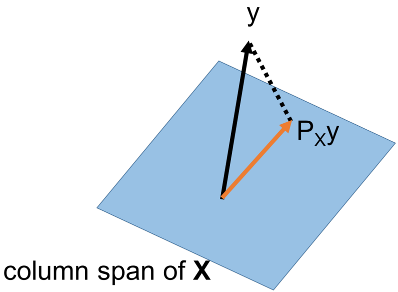
Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters) and response vector $\vec{y} \in \mathbb{R}^{d}$, want to find $\vec{c} \in \mathbb{R}^{d}$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.

Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters) and response vector $\vec{y} \in \mathbb{R}^d$, want to find $\vec{c} \in \mathbb{R}^d$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.

Optimal solution is to chose $\vec{c}$ so that $\mathbf{X}\vec{c} = \mathbf{P}_X\vec{y}$ – the projection of $\vec{y}$ onto the column span of $\mathbf{X}$.

Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters) and response vector $\vec{y} \in \mathbb{R}^{d}$, want to find $\vec{c} \in \mathbb{R}^{d}$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.

Optimal solution is to chose $\vec{c}$ so that $\mathbf{X}\vec{c} = \mathbf{P}_X \vec{y}$ – the projection of $\vec{y}$ onto the column span of $\mathbf{X}$.



Classic Linear Regression: Given $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $n > d$ (we have more data points than parameters) and response vector $\vec{y} \in \mathbb{R}^d$, want to find $\vec{c} \in \mathbb{R}^d$ minimizing $\|\mathbf{X}\vec{c} - \vec{y}\|_2$.

Optimal solution is to chose $\vec{c}$ so that $\mathbf{X}\vec{c} = \mathbf{P}_X\vec{y}$ – the projection of $\vec{y}$ onto the column span of $\mathbf{X}$.

Writing the SVD $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ we have:

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

THE SVD AND LINEAR REGRESSION

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

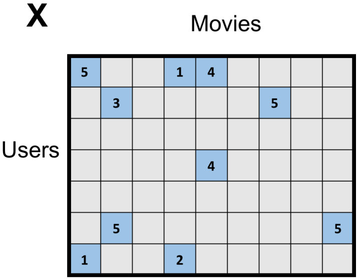
Rest of Class: Examples of how low-rank approximation is applied in a variety of data science applications.

Rest of Class: Examples of how low-rank approximation is applied in a variety of data science applications.

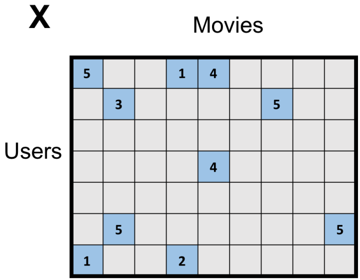
- Used for many reasons other than dimensionality reduction/data compression.

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix).

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



Consider a matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



Solve: $Y = \underset{\text{rank }-k\ \mathbf{B}}{\arg\min} \sum_{\text{observed } (j,k)} \left[\mathbf{X}_{j,k} - \mathbf{B}_{j,k}\right]^2$

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.

Y

Movies

Users

4.9	3.1	3	**1.1**	**3.8**	4.1	4.1	3.4	4.6
3.6	**3**	3	1.2	3.8	4.2	**5**	3.4	4.8
2.8	3	3	2.3	3	3	3	3	3.2
3.4	3	3	4	**4.1**	4.1	4.2	3	3
2.8	3	3	2.3	3	3	3	3	3.4
2.2	**5**	3	4	4.2	3.9	4.4	4	**5.3**
1	3.3	3	**2.2**	3.1	2.9	3.2	1.5	1.8

Solve: $Y = \underset{\text{rank}-k\ \mathbf{B}}{\arg\min} \sum_{\text{observed } (j,k)} \left[\mathbf{X}_{j,k} - \mathbf{B}_{j,k}\right]^2$

Under certain assumptions, can show that $\mathbf{Y}$ well approximates $\mathbf{X}$ on both the observed and (most importantly) unobserved entries.

Dimensionality reduction embeds d-dimensional vectors into d' dimensions. But what about when you want to embed objects other than vectors?

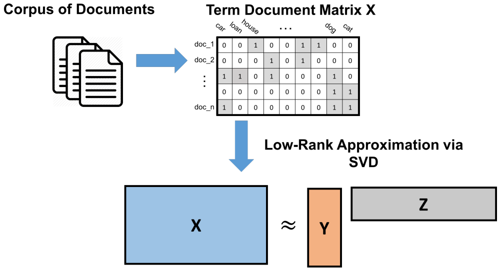
Dimensionality reduction embeds d-dimensional vectors into d' dimensions. But what about when you want to embed objects other than vectors?

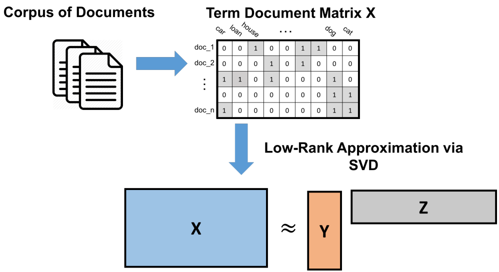
- Documents (for topic-based search and classification)
- Words (to identify synonyms, translations, etc.)
- Nodes in a social network

Dimensionality reduction embeds d-dimensional vectors into d' dimensions. But what about when you want to embed objects other than vectors?

- Documents (for topic-based search and classification)
- Words (to identify synonyms, translations, etc.)
- Nodes in a social network

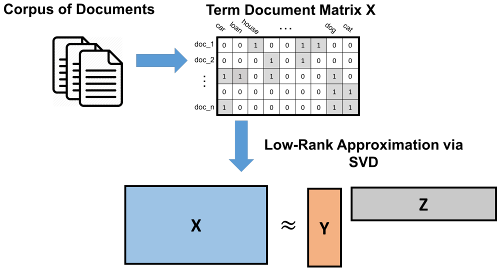
Classical approach is to convert each item into a high-dimensional feature vector and then apply low-rank approximation


Corpus of Documents
Term Document Matrix X
car loan house ... dog cat
doc_1 0 0 1 0 0 1 1 0 0
doc_2 0 0 0 1 0 1 0 0 0
⋮ 1 1 0 1 0 0 0 1 0
0 0 0 0 0 0 0 1 1
doc_n 1 0 0 0 0 0 0 1 1
Low-Rank Approximation via SVD
X
≈
Y
Z

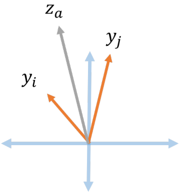


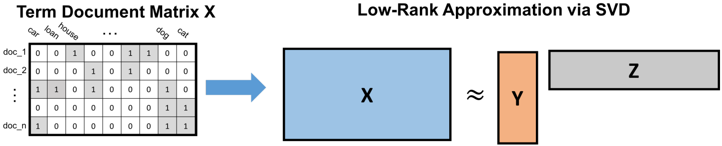
- $\langle \vec{y}_i, \vec{z}_a \rangle \approx 1$ when doc_i contains $word_a$.
- If doc_i and doc_i both contain $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle = 1$.

EXAMPLE: LATENT SEMANTIC ANALYSIS

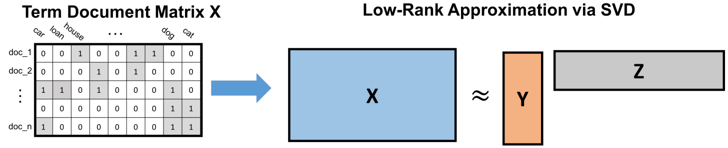


- $\langle \vec{y}_i, \vec{z}_a \rangle \approx 1$ when doc_i contains $word_a$.
- If doc_i and doc_j both contain $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle = 1$.



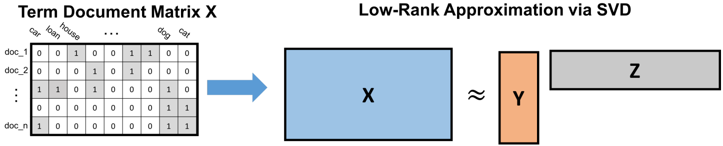

Term Document Matrix X
car loan house ... dog cat
doc_1 0 0 1 0 0 1 1 0 0
doc_2 0 0 0 1 0 1 0 0 0
1 1 0 1 0 0 0 1 0
0 0 0 0 0 0 0 1 1
doc_n 1 0 0 0 0 0 0 1 1
Low-Rank Approximation via SVD
X
≈
Y
Z

EXAMPLE: LATENT SEMANTIC ANALYSIS

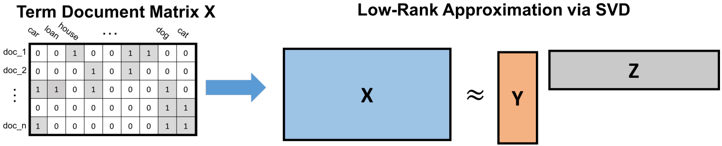


- The columns $\vec{z}_1, \vec{z}_2, \ldots$ give representations of words, with $\vec{z}_i$ and $\vec{z}_j$ tending to have high dot product if $word_i$ and $word_j$ appear in many of the same documents.
- $\mathbf{Z}$ corresponds to the top k right singular vectors: the eigenvectors of $\mathbf{X}\mathbf{X}^T$.

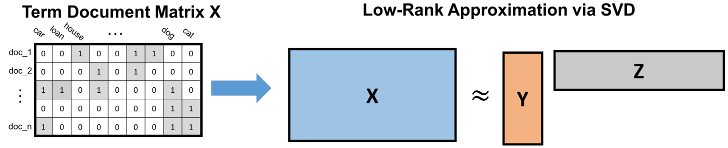
EXAMPLE: LATENT SEMANTIC ANALYSIS



- The columns $\vec{z}_1, \vec{z}_2, \ldots$ give representations of words, with $\vec{z}_i$ and $\vec{z}_j$ tending to have high dot product if $word_i$ and $word_j$ appear in many of the same documents.
- $\mathbf{Z}$ corresponds to the top k right singular vectors: the eigenvectors of $\mathbf{X}\mathbf{X}^T$. Intuitively, what is $\mathbf{X}\mathbf{X}^T$?



- The columns $\vec{z}_1, \vec{z}_2, \ldots$ give representations of words, with $\vec{z}_i$ and $\vec{z}_j$ tending to have high dot product if $word_i$ and $word_j$ appear in many of the same documents.
- $\mathbf{Z}$ corresponds to the top k right singular vectors: the eigenvectors of $\mathbf{X}\mathbf{X}^T$. Intuitively, what is $\mathbf{X}\mathbf{X}^T$?
- $(\mathbf{X}\mathbf{X}^T)_{i,j} = \#$ documents that $word_i$ and $word_j$ co-occur in.



- The columns $\vec{z}_1, \vec{z}_2, \ldots$ give representations of words, with $\vec{z}_i$ and $\vec{z}_j$ tending to have high dot product if $word_i$ and $word_j$ appear in many of the same documents.
- $\mathbf{Z}$ corresponds to the top k right singular vectors: the eigenvectors of $\mathbf{X}\mathbf{X}^T$. Intuitively, what is $\mathbf{X}\mathbf{X}^T$?
- $(\mathbf{X}\mathbf{X}^T)_{i,j} = \#$ documents that $word_i$ and $word_j$ co-occur in.
- A document based similarity matrix.

Not obvious how to convert a word into a feature vector that captures the meaning of that word.

- In LSA, feature vector is the set of documents that word appears in.
- SVD of term-document matrix $\mathbf{X}$ corresponds to eigendecomposition of document based similarity matrix $\mathbf{X}\mathbf{X}^T$.

Not obvious how to convert a word into a feature vector that captures the meaning of that word.

- In LSA, feature vector is the set of documents that word appears in.
- SVD of term-document matrix $\mathbf{X}$ corresponds to eigendecomposition of document based similarity matrix $\mathbf{X}\mathbf{X}^T$.
- Many alternative similarities: how often do $word_i, word_j$ appear in the same sentence, in the same window of w words, in similar positions of documents in different languages, etc.

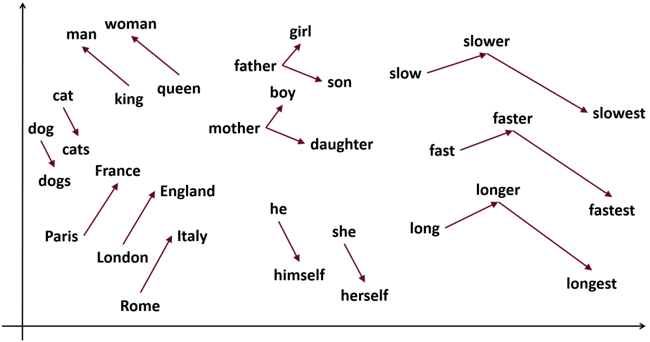
Not obvious how to convert a word into a feature vector that captures the meaning of that word.

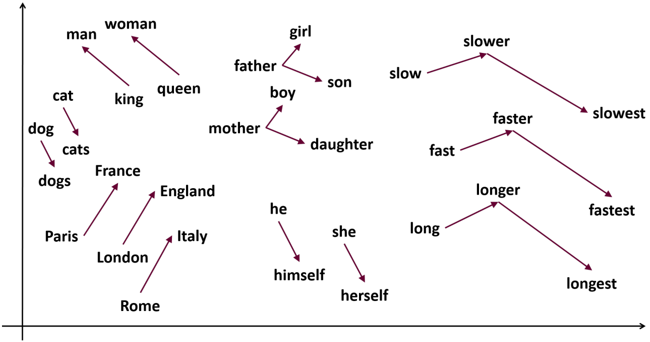
- In LSA, feature vector is the set of documents that word appears in.
- SVD of term-document matrix $\mathbf{X}$ corresponds to eigendecomposition of document based similarity matrix $\mathbf{X}\mathbf{X}^T$.
- Many alternative similarities: how often do $word_i, word_j$ appear in the same sentence, in the same window of w words, in similar positions of documents in different languages, etc.
- Replacing $\mathbf{X}\mathbf{X}^T$ with these different metrics (sometimes appropriately transformed) leads to popular word embedding algorithms: word2vec, GloVe, fastTest, etc.

Not obvious how to convert a word into a feature vector that captures the meaning of that word.

- In LSA, feature vector is the set of documents that word appears in.
- SVD of term-document matrix $\mathbf{X}$ corresponds to eigendecomposition of document based similarity matrix $\mathbf{X}\mathbf{X}^T$.
- Many alternative similarities: how often do $word_i, word_j$ appear in the same sentence, in the same window of w words, in similar positions of documents in different languages, etc.
- Replacing $\mathbf{X}\mathbf{X}^T$ with these different metrics (sometimes appropriately transformed) leads to popular word embedding algorithms: word2vec, GloVe, fastTest, etc.
- Perform low-rank approximation of similarity matrix directly.

EXAMPLE: WORD EMBEDDING





word2vec was originally described as a neural-network method, but Levy and Goldberg show that it is simply low-rank approximation of a specific similarity matrix. *Neural word embedding as implicit matrix factorization.*

Next Time: Build on the idea of low-rank approximation of similarity matrix low-rank approximation to perform non-linear dimensionality reduction for data that is not close to a low-dimensional linear subspace.

Questions?