Dimensionality Reduction: Principal Component Analysis and SVD

CS771: Introduction to Machine Learning

Piyush Rai

Dimensionality Reduction

Can think of W as a linear mapping that transforms low-dim $\boldsymbol{z}_n$ to high-dim $\boldsymbol{x}_n$

Some dim-red techniques assume a nonlinear mapping function f such that $\boldsymbol{x}_n = f(\boldsymbol{z}_n)$

For example, f can be modeled by a kernel or a deep neural net

- A broad class of techniques
- Goal is to compress the original representation of the inputs
- Example: Approximate each input $\boldsymbol{x}_n \in \mathbb{R}^D$, $n = 1,2, \dots, N$ as a linear combination of $K < \min\{D, N\}$ "basis" vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K$, each also $\in \mathbb{R}^D$

Note: These "basis" vectors need not necessarily be linearly independent. But for some dim. red. techniques, e.g., classic principal component analysis (PCA), they are

$$\boldsymbol{x}_n \approx \sum_{k=1}^{K} z_{nk} \boldsymbol{w}_k = \mathbf{W} \boldsymbol{z}_n$$

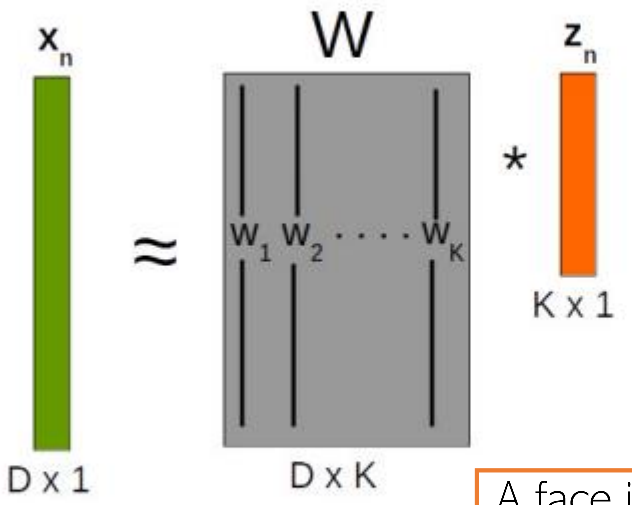
$\mathbf{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K]$ is $D \times K$

$\boldsymbol{z}_n = [z_{n1}, z_{n2}, \dots, z_{nK}]$ is $K \times 1$

- We have represented each $\boldsymbol{x}_n \in \mathbb{R}^D$ by a K-dim vector $\boldsymbol{z}_n$ (a new feat. rep)
- To store N such inputs $\{\boldsymbol{x}_n\}_{n=1}^N$, we need to keep $\mathbf{W}$ and $\{\boldsymbol{z}_n\}_{n=1}^N$
  - Originally we required $N \times D$ storage, now $N \times K + D \times K = (N + D) \times K$ storage
  - If $K \ll \min\{D, N\}$, this yields substantial storage saving, hence good compression

Dimensionality Reduction

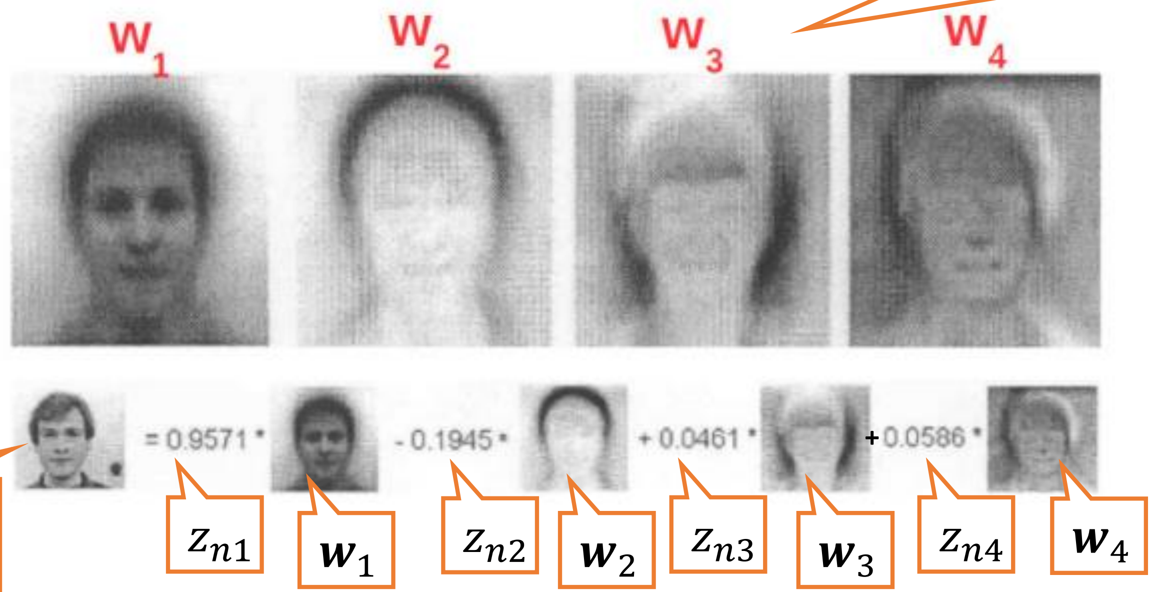
- Dim-red for face images



A face image $\boldsymbol{x}_n \in \mathbb{R}^D$

Each "basis" image is like a "template" that captures the common properties of face images in the dataset

K=4 "basis" face images



- In this example, $\boldsymbol{z}_n \in \mathbb{R}^K$ $(K = 4)$ is a low-dim feature rep. for $\boldsymbol{x}_n \in \mathbb{R}^D$

Like 4 new features

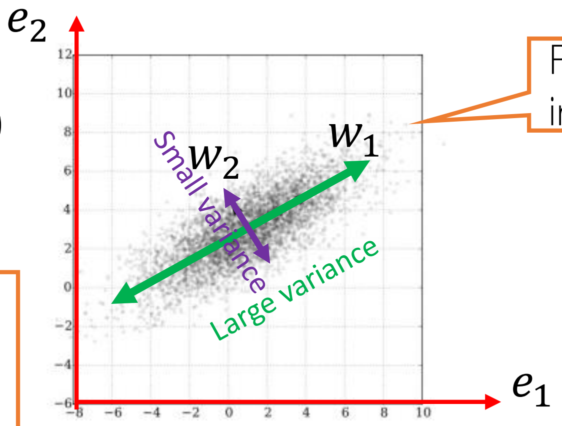
- Essentially, each face image in the dataset now represented by just 4 real numbers ☺
- Different dim-red algos differ in terms of how the basis vectors are defined/learned
  - .. And in general, how the function f in the mapping $\boldsymbol{x}_n = f(\boldsymbol{z}_n)$ is defined

Principal Component Analysis (PCA)

- A classic linear dim. reduction method (Pearson, 1901; Hotelling, 1930)
- Can be seen as
  - Learning directions (co-ordinate axes) that capture maximum variance in data

e_1, e_2: Standard co-ordinate axis ($\boldsymbol{x} = [x_1, x_2]$)

w_1, w_2: New co-ordinate axis ($\boldsymbol{z} = [z_1, z_2]$)



PCA is essentially doing a change of axes in which we are representing the data

Each input will still have 2 co-ordinates, in the new co-ordinate system, equal to the distances measured from the new origin

To reduce dimension, can only keep the co-ordinates of those directions that have largest variances (e.g., in this example, if we want to reduce to one-dim, we can keep the co-ordinate z_1 of each point along w_1 and throw away z_2). We won't lose much information

  - Learning projection directions that result in smallest reconstruction error

$$\underset{\boldsymbol{W},\boldsymbol{Z}}{\operatorname{argmin}} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \boldsymbol{W}\boldsymbol{z}_n\|^2 = \underset{\boldsymbol{W},\boldsymbol{Z}}{\operatorname{argmin}} \|\boldsymbol{X} - \boldsymbol{Z}\boldsymbol{W}\|^2$$

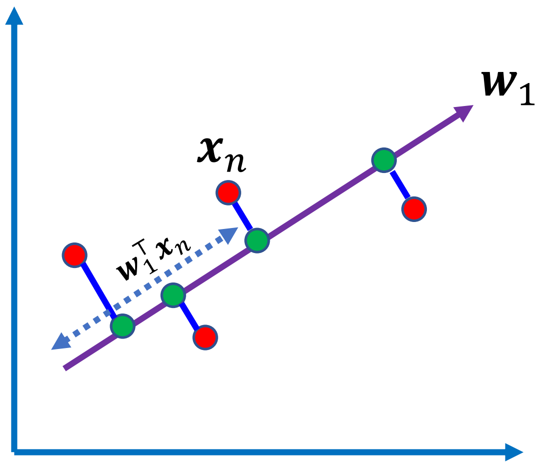
Subject to orthonormality constraints: $\boldsymbol{w}_i^\top \boldsymbol{w}_j = 0$ for $i \neq j$ and $\|\boldsymbol{w}_i\|^2 = 1$

- PCA also assumes that the projection directions are orthonormal

PCA: From the variance perspective

Solving PCA by Finding Max. Variance Directions

- Consider projecting an input $\boldsymbol{x}_n \in \mathbb{R}^D$ along a direction $\boldsymbol{w}_1 \in \mathbb{R}^D$
- Projection/embedding of $\boldsymbol{x}_n$ (red points below) will be $\boldsymbol{w}_1^\top \boldsymbol{x}_n$ (green pts below)



Mean of projections of all inputs:

$$\frac{1}{N}\sum_{n=1}^N \boldsymbol{w}_1^\top \boldsymbol{x}_n = \boldsymbol{w}_1^\top \left(\frac{1}{N}\sum_{n=1}^N \boldsymbol{x}_n\right) = \boldsymbol{w}_1^\top \boldsymbol{\mu}$$

Variance of the projections:

$$\frac{1}{N}\sum_{n=1}^N (\boldsymbol{w}_1^\top \boldsymbol{x}_n - \boldsymbol{w}_1^\top \boldsymbol{\mu})^2 = \frac{1}{N}\sum_{n=1}^N \{\boldsymbol{w}_1^\top (\boldsymbol{x}_n - \boldsymbol{\mu})\}^2 = \boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1$$

$\boldsymbol{S}$ is the $D \times D$ cov matrix of the data:

$$\boldsymbol{S} = \frac{1}{N}\sum_{n=1}^N (\boldsymbol{x}_n - \boldsymbol{\mu})(\boldsymbol{x}_n - \boldsymbol{\mu})^\top$$

For already centered data, $\boldsymbol{\mu} = \boldsymbol{0}$ and $\boldsymbol{S} = \frac{1}{N}\sum_{n=1}^N \boldsymbol{x}_n \boldsymbol{x}_n^\top = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\top$

- Want $\boldsymbol{w}_1$ such that variance $\boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1$ is maximized

$$\underset{\boldsymbol{w}_1}{\text{argmax}}\ \boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1 \quad \text{s.t.} \quad \boldsymbol{w}_1^\top \boldsymbol{w}_1 = 1$$

Need this constraint otherwise the objective's max will be infinity

Max. Variance Direction

Variance along the direction $\boldsymbol{w}_1$

Note: Total variance of the data is equal to the sum of eigenvalues of $\boldsymbol{S}$, i.e., $\sum_{d=1}^{D} \lambda_d$

PCA would keep the top $K < D$ such directions of largest variances

- Our objective function was $\underset{\boldsymbol{w}_1}{\operatorname{argmax}}\ \boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1$ s.t. $\boldsymbol{w}_1^\top \boldsymbol{w}_1 = 1$
- Can construct a Lagrangian for this problem

$$\underset{\boldsymbol{w}_1}{\operatorname{argmax}}\ \boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1 + \lambda_1 (1\text{-}\boldsymbol{w}_1^\top \boldsymbol{w}_1)$$

- Taking derivative w.r.t. $\boldsymbol{w}_1$ and setting to zero gives $\boldsymbol{S}\boldsymbol{w}_1 = \lambda_1 \boldsymbol{w}_1$
- Therefore $\boldsymbol{w}_1$ is an eigenvector of the cov matrix $\boldsymbol{S}$ with eigenvalue λ_1

Note: In general, $\boldsymbol{S}$ will have D eigvecs

- Claim: $\boldsymbol{w}_1$ is the eigenvector of $\boldsymbol{S}$ with largest eigenvalue λ_1. Note that

$$\boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1 = \lambda_1 \boldsymbol{w}_1^\top \boldsymbol{w}_1 = \lambda_1$$

- Thus variance $\boldsymbol{w}_1^\top \boldsymbol{S} \boldsymbol{w}_1$ will be max. if λ_1 is the largest eigenvalue (and $\boldsymbol{w}_1$ is the corresponding top eigenvector; also known as the first Principal Component)
- Other large variance directions can also be found likewise (with each being orthogonal to all others) using the eigendecomposition of cov matrix $\boldsymbol{S}$ (this is PCA)

PCA: From the reconstruction perspective

Alternate Basis and Reconstruction

- Representing a data point $\boldsymbol{x}_n = [x_{n1}, x_{n2}, \dots, x_{nD}]^\top$ in the standard orthonormal basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_D\}$

$$\boldsymbol{x}_n = \sum_{d=1}^{D} x_{nd}\boldsymbol{e}_d$$

$\boldsymbol{e}_d$ is a vector of all zeros except a single 1 at the d^{th} position. Also, $\boldsymbol{e}_d{}^\top \boldsymbol{e}_{d'} = 0$ for $d \neq d'$

- Let's represent the same data point in a new orthonormal basis $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_D\}$

z_{nd} is the projection of $\boldsymbol{x}_n$ along the direction $\boldsymbol{w}_d$ since $z_{nd} = \boldsymbol{w}_d^\top \boldsymbol{x}_n = \boldsymbol{x}_n^\top \boldsymbol{w}_d$ (verify)

$$\boldsymbol{x}_n = \sum_{d=1}^{D} z_{nd}\boldsymbol{w}_d$$

$\boldsymbol{z}_n = [z_{n1}, z_{n2}, \dots, z_{nD}]^\top$ denotes the co-ordinates of $\boldsymbol{x}_n$ in the new basis

- Ignoring directions along which projection z_{nd} is small, we can approximate $\boldsymbol{x}_n$ as

$$\boldsymbol{x}_n \approx \hat{\boldsymbol{x}}_n = \sum_{d=1}^{K} z_{nd}\boldsymbol{w}_d = \sum_{d=1}^{K} (\boldsymbol{x}_n^\top \boldsymbol{w}_d)\boldsymbol{w}_d = \sum_{d=1}^{K} (\boldsymbol{w}_d \boldsymbol{w}_d^\top)\boldsymbol{x}_n$$

Note that $\|\boldsymbol{x}_n - \sum_{d=1}^{K} (\boldsymbol{w}_d \boldsymbol{w}_d^\top)\boldsymbol{x}_n\|^2$ is the reconstruction error on $\boldsymbol{x}_n$. Would like it to minimize w.r.t. $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K$

- Now $\boldsymbol{x}_n$ is represented by $K < D$ dim. rep. $\boldsymbol{z}_n = [z_{n1}, z_{n2}, \dots, z_{nK}]$ and (verify)

$$\boldsymbol{z}_n \approx \mathbf{W}_K^\top \boldsymbol{x}_n$$

Also, $\boldsymbol{x}_n \approx \mathbf{W}_K \boldsymbol{z}_n$

$\mathbf{W}_K = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K]$ is the "projection matrix" of size $D \times K$

Minimizing Reconstruction Error

- We plan to use only K directions $[\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K]$ so would like them to be such that the total reconstruction error is minimized

Constant; doesn't depend on the $\boldsymbol{w}_d$'s

$$\mathcal{L}(\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K) = \sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 = \sum_{n=1}^{N} \left\| \boldsymbol{x}_n - \sum_{d=1}^{K} (\boldsymbol{w}_d \boldsymbol{w}_d^\top) \boldsymbol{x}_n \right\|^2 = C - \sum_{d=1}^{K} \boldsymbol{w}_d^\top \mathbf{S} \boldsymbol{w}_d \text{ (verify)}$$

Variance along $\boldsymbol{w}_d$

- Each optimal $\boldsymbol{w}_d$ can be found by solving

$$\underset{\boldsymbol{w}_d}{\operatorname{argmin}}\ \mathcal{L}(\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K) = \underset{\boldsymbol{w}_d}{\operatorname{argmax}}\ \boldsymbol{w}_d^\top \mathbf{S} \boldsymbol{w}_d$$

- Thus minimizing the reconstruction error is equivalent to maximizing variance

- The K directions can be found by solving the eigendecomposition of $\boldsymbol{S}$

- Note: $\sum_{d=1}^{K} \boldsymbol{w}_d^\top \mathbf{S} \boldsymbol{w}_d = \operatorname{trace}(\mathbf{W}_K^\top \mathbf{S} \mathbf{W}_K)$
  - Thus $\operatorname{argmax}_{\mathbf{W}_K} \operatorname{trace}(\boldsymbol{W}_K^\top \mathbf{S} \mathbf{W}_K)$ s.t. orthonormality on columns of $\mathbf{W}_k$ is the same as solving the eigendec. of $\boldsymbol{S}$ (recall that Spectral Clustering also required solving this)

Principal Component Analysis

- Center the data (subtract the mean $\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n$ from each data point)
- Compute the $D \times D$ covariance matrix $\mathbf{S}$ using the centered data matrix $\mathbf{X}$ as

$$\mathbf{S} = \frac{1}{N}\mathbf{X}^\top\mathbf{X}$$

(Assuming $\mathbf{X}$ is arranged as $N \times D$)

- Do an eigendecomposition of the covariance matrix $\mathbf{S}$ (many methods exist)
- Take top $K < D$ leading eigvectors $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K\}$ with eigvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_K\}$
- The K-dimensional projection/embedding of each input is

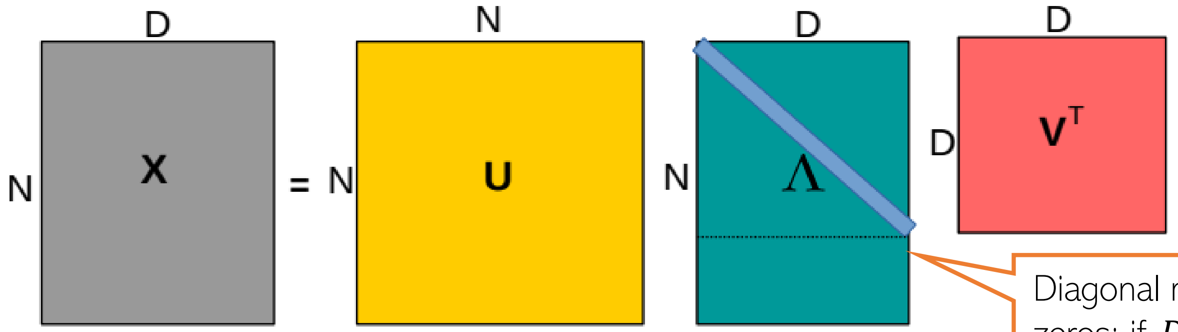
$$\boldsymbol{z}_n \approx \mathbf{W}_K^\top \boldsymbol{x}_n$$

$\mathbf{W}_K = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K]$ is the "projection matrix" of size $D \times K$

Note: Can decide how many eigvecs to use based on how much variance we want to campure (recall that each λ_k gives the variance in the k^{th} direction (and their sum is the total variance)

Singular Value Decomposition (SVD)

- Any matrix $\mathbf{X}$ of size $N \times D$ can be represented as the following decomposition



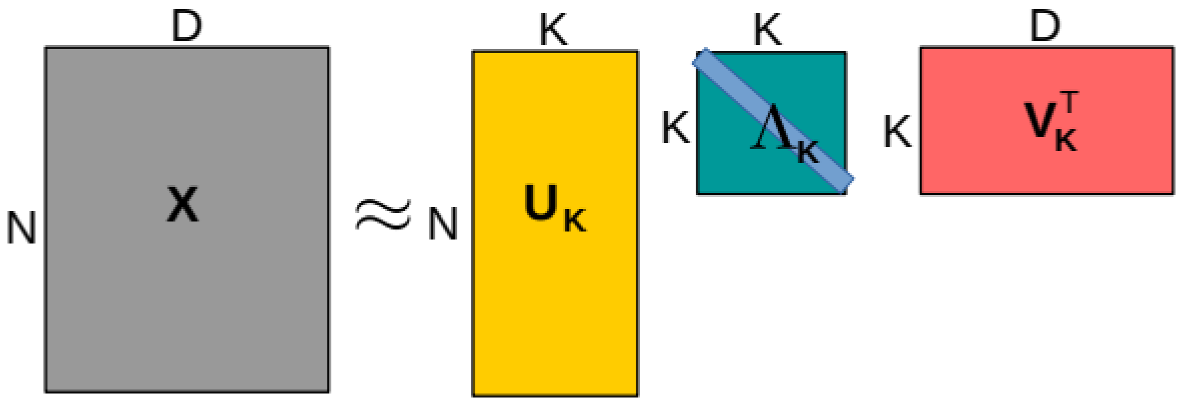
$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top = \sum_{k=1}^{\min\{N,D\}} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k^\top$$

Diagonal matrix. If $N > D$, last $D - N$ rows are all zeros; if $D > N$, last $D - N$ columns are all zeros

- $\mathbf{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_N]$ is $N \times N$ matrix of left singular vectors, each $\boldsymbol{u}_n \in \mathbb{R}^N$
  - $\mathbf{U}$ is also orthonormal
- $\mathbf{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_N]$ is $D \times D$ matrix of right singular vectors, each $\boldsymbol{v}_d \in \mathbb{R}^D$
  - $\mathbf{V}$ is also orthonormal
- Λ is $N \times D$ with only $\mathbf{min}(N, D)$ diagonal entries - singular values
- Note: If $\mathbf{X}$ is symmetric then it is known as eigenvalue decomposition ($\mathbf{U} = \mathbf{V}$)

Low-Rank Approximation via SVD

- If we just use the top $K < \min\{N, D\}$ singular values, we get a rank-K SVD

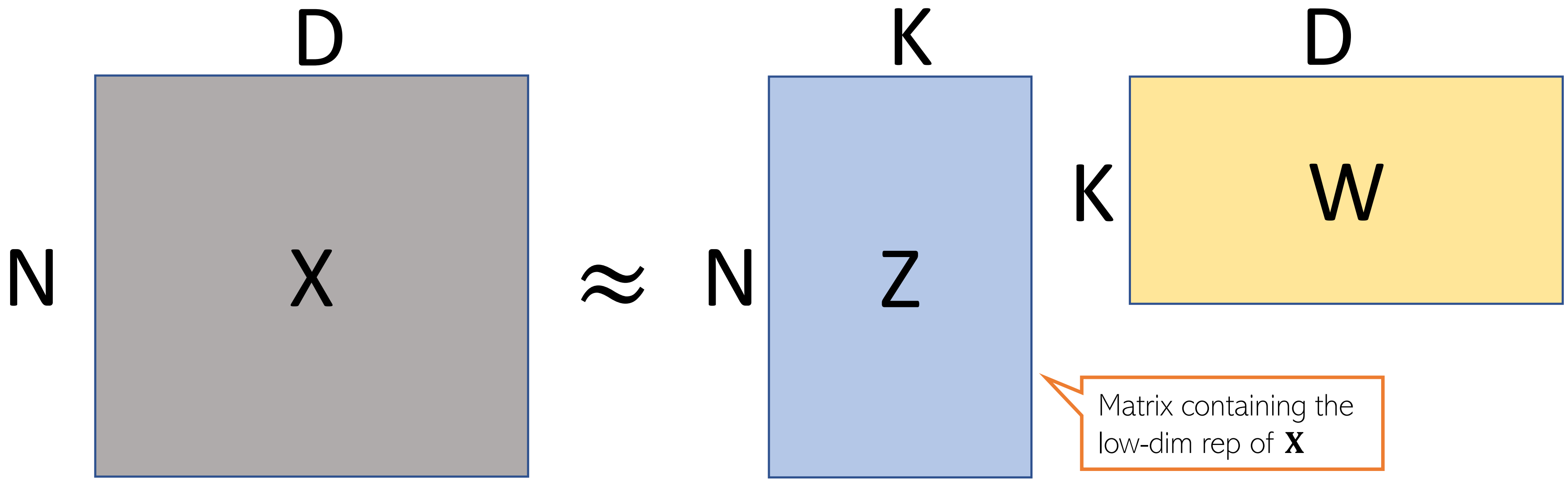


$$\mathbf{X} \approx \hat{\mathbf{X}} = \sum_{k=1}^{K} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k^\top = \mathbf{U}_K \Lambda_K \mathbf{V}_K^\top$$

- Above SVD approx. can be shown to minimize the reconstruction error $\|\boldsymbol{X} - \widehat{\boldsymbol{X}}\|$
  - Fact: SVD gives the best rank-K approximation of a matrix
- PCA is done by doing SVD on the covariance matrix $\mathbf{S}$ (left and right singular vectors are the same and become eigenvectors, singular values become eigenvalues)

Dim-Red as Matrix Factorization

- If we don't care about the orthonormality constraints, then dim-red can also be achieved by solving a matrix factorization problem on the data matrix $\mathbf{X}$



Matrix containing the low-dim rep of $\mathbf{X}$

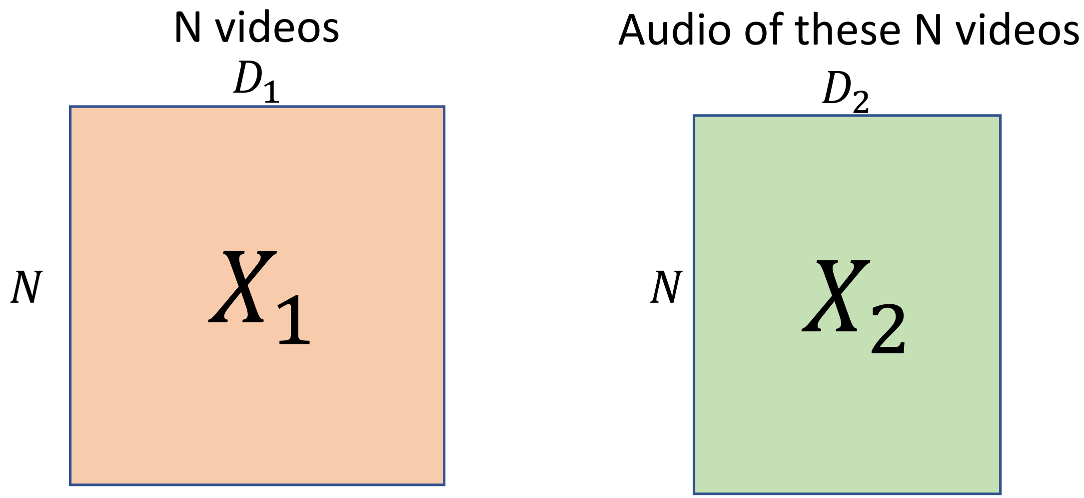
$$\{\hat{\mathbf{Z}}, \hat{\mathbf{W}}\} = \text{argmin}_{\mathbf{Z},\mathbf{W}} ||\mathbf{X} - \mathbf{Z}\mathbf{W}||^2$$

If $K < \min\{D, N\}$, such a factorization gives a low-rank approximation of the data matrix X

- Can solve such problems using ALT-OPT
- Can impose various constraints on $\mathbf{Z}$ and $\mathbf{W}$ (e.g., sparsity, non-negativity, etc)

Joint Dim-Red

- Often we have two or more data sources with 1-1 correspondence between inputs



- Sometimes, we may want to perform a common dim-red for both sources to get a common feature rep which captures properties of both sources (or fused their info)
- This can be done by doing a joint dim-red of both sources. Many methods exists, e.g.,
  - Canonical Correlational Analysis (CCA): looks at cross-covar rather than variances
  - Joint Matrix Factorization

$$\text{argmin}_{\mathbf{Z},\mathbf{W}_1,\mathbf{W}_2} \|\mathbf{X}_1 - \mathbf{Z}\mathbf{W}_1\|^2 + \|\mathbf{X}_2 - \mathbf{Z}\mathbf{W}_2\|^2$$

Coming up next

- Some methods for computing eigenvectors
- Supervised dimensionality reduction
- Nonlinear dimensionality reduction
  - Kernel PCA
  - Manifold Learning