Principle Component Analysis (PCA)

CSE 5334 Data Mining
Spring 2020

Won Hwa Kim

(Slides courtesy of Heng Huang at Pittsburg)

Goals for the lecture

you should understand the following concepts

- Dimensionality Reduction
- Normalization of data
- Covariance Matrix
- Maximum Variance Method
- Measures of Association

Principle Component Analysis (PCA)

- Idea
  - Given: data points in d-dimensional space
  - Project them onto lower dimension space
  - Preserve as much information as possible

- E.g.,
  - Find the best planar approximation to 3D data
  - Find the best 12-D approximation to 100000-D data

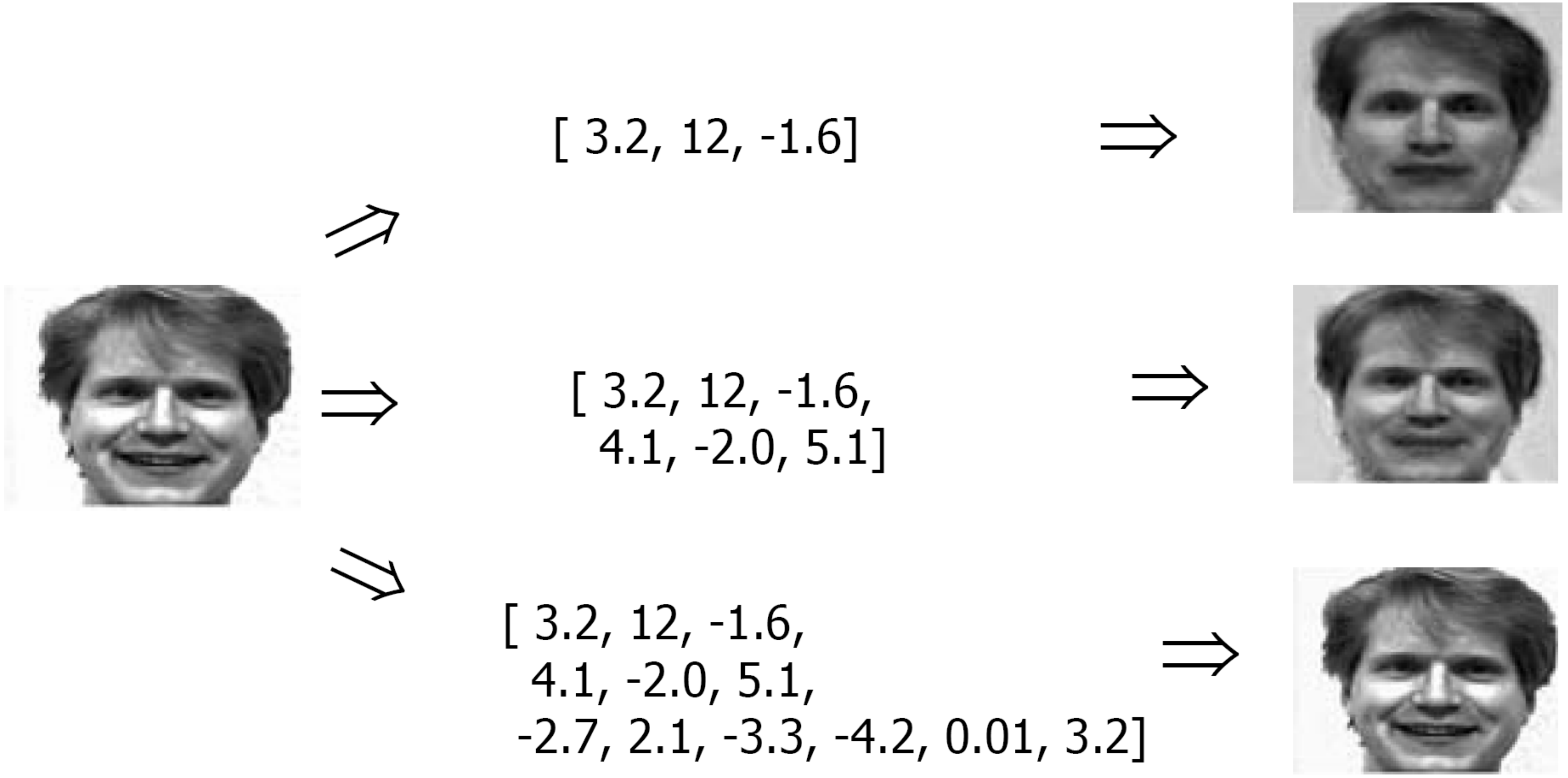
- In particular, choose projection that minimizes squared error in reconstructing the original data

Vision Application: Face Recognition

- Want to identify specific person, based on facial image
- Robust to…
  - Facial hair, glasses,
  - Different lighting

- Cannot use all 256x256 given pixels
- Need another option.

Vision Application: Face Recognition



[3.2, 12, -1.6]

[3.2, 12, -1.6,
4.1, -2.0, 5.1]

[3.2, 12, -1.6,
4.1, -2.0, 5.1,
-2.7, 2.1, -3.3, -4.2, 0.01, 3.2]

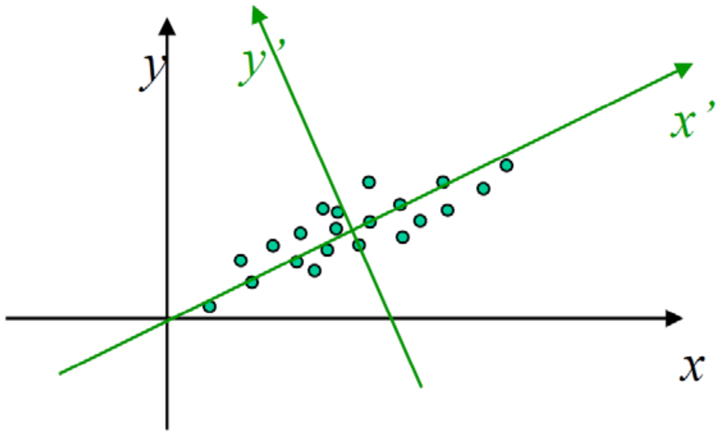
Why do we care

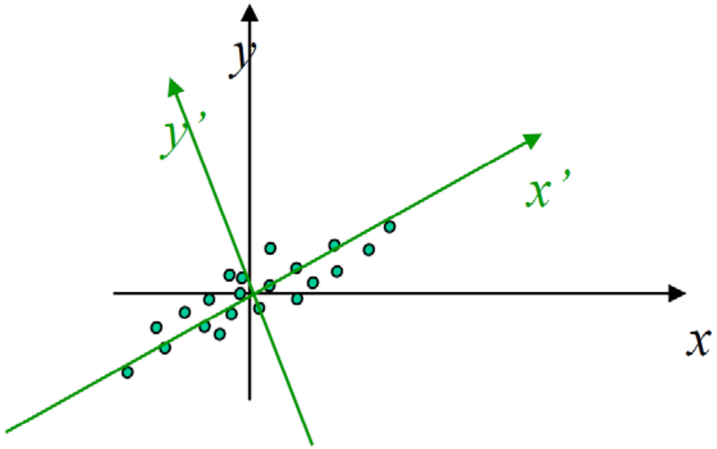
- Orthonormal basis provides trivial projection
- Given basis $\mathrm{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$
- Project any d-dimensional **x** to k values
  - $\alpha_1 = \mathbf{u}_1^{\mathrm{T}}\mathbf{x} \quad \alpha_2 = \mathbf{u}_2^{\mathrm{T}}\mathbf{x} \;\ldots\; \alpha_k = \mathbf{u}_k^{\mathrm{T}}\mathbf{x}$
  - $\alpha = \mathbf{U}^{\mathrm{T}}\mathbf{x}$
  - $\mathbf{x} \approx \sum_{\mathbf{i}} \alpha_i \mathbf{u}_i = \sum_{\mathbf{i}} (\mathbf{u}_i^{\mathrm{T}}\mathbf{x})\,\mathbf{u}_i$ [“=” if all d values]

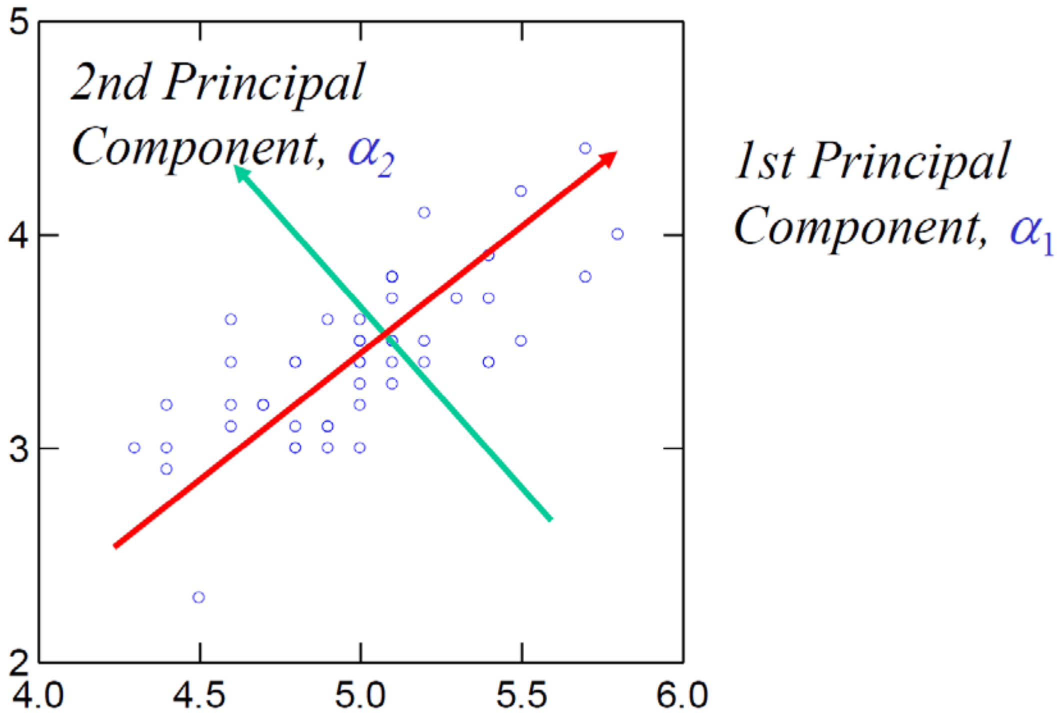
- Use ``centered’’ vectors:

$$\mathbf{x}' = \mathbf{x} - \underline{\mathbf{x}} \quad \text{where} \quad \underline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}^{n} \qquad \boxed{\alpha_i = \mathbf{u}_i^{\mathrm{T}}(\mathbf{x} - \underline{\mathbf{x}})}$$

Principle Component Analysis (PCA)







Minimize reconstruction error

- Assume that data is a set of ND-dimensional vectors $\mathbf{x}^n = \langle x_1^n \ldots x_d^n \rangle$
- Represent each in terms of any d orthogonal basis vectors

$$\mathbf{x}^n = \sum_{i=1}^{d} z_i^n \mathbf{u}_i; \quad \mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}$$

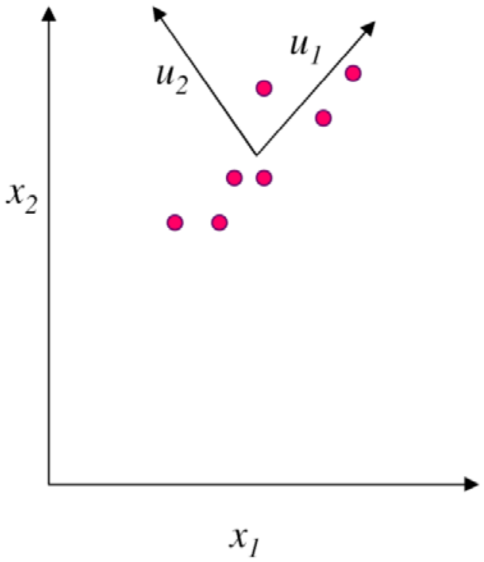
PCA: given k<d. Find { $\mathbf{u}_1$, …, $\mathbf{u}_k$ }

that minimizes $E_k = \sum_{n=1}^{N} \left\| x^n - \hat{x}_k^n \right\|_2^2$

where $\hat{\mathbf{x}}_k^n = \underline{\mathbf{x}} + \sum_{i=1}^{k} \alpha_i^n \mathbf{u}_i$

Mean

$$\underline{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}^n$$



PCA

- Note: $\hat{\mathbf{x}}_d^n = \underline{\mathbf{x}} + \sum_{i=1}^{d} \alpha_i^n \mathbf{u}_i \equiv \mathbf{x}^n$
- So… $\mathbf{x}^n - \hat{\mathbf{x}}_k^n = \sum_{i=k+1}^{d} \alpha_i^n \mathbf{u}_i = \sum_{i=k+1}^{d} \left((\mathbf{x}^n - \underline{\mathbf{x}})^T \mathbf{u}_i \right) \mathbf{u}_i$
- Therefore…

PCA: given k<d. Find { $\mathbf{u}_1$. …. $\mathbf{u}_k$ }

that minimizes $E_k = \sum_{n=1}^{N} \left\| x^n - \hat{x}_k^n \right\|_2^2$

where $\hat{\mathbf{x}}_k^n = \underline{\mathbf{x}} + \sum_{i=1}^{k} \alpha_i^n \mathbf{u}_i$

$$E_k = \sum_{n=1}^{N} \left\| \sum_{i=k+1}^{d} \left((\mathbf{x}^n - \underline{\mathbf{x}})^T \mathbf{u}_i \right) \mathbf{u}_i \right\|^2 = \sum_{n=1}^{N} \sum_{i=k+1}^{d} \left[(\mathbf{x}^n - \underline{\mathbf{x}})^T \mathbf{u}_i \right]^2$$

$$= \sum_{i=k+1}^{d} \sum_{n=1}^{N} \left[\mathbf{u}_i^T (\mathbf{x}^n - \underline{\mathbf{x}}) \right] \left[(\mathbf{x}^n - \underline{\mathbf{x}})^T \mathbf{u}_i \right]$$

$$= \sum_{i=k+1}^{d} \mathbf{u}_i^T \, \Sigma \, \mathbf{u}_i$$

Covariance matrix:

$$\Sigma = \sum_n (\mathrm{x}^n - \bar{\mathrm{x}})(\mathrm{x}^n - \bar{\mathrm{x}})^T$$

Matrix Decomposition

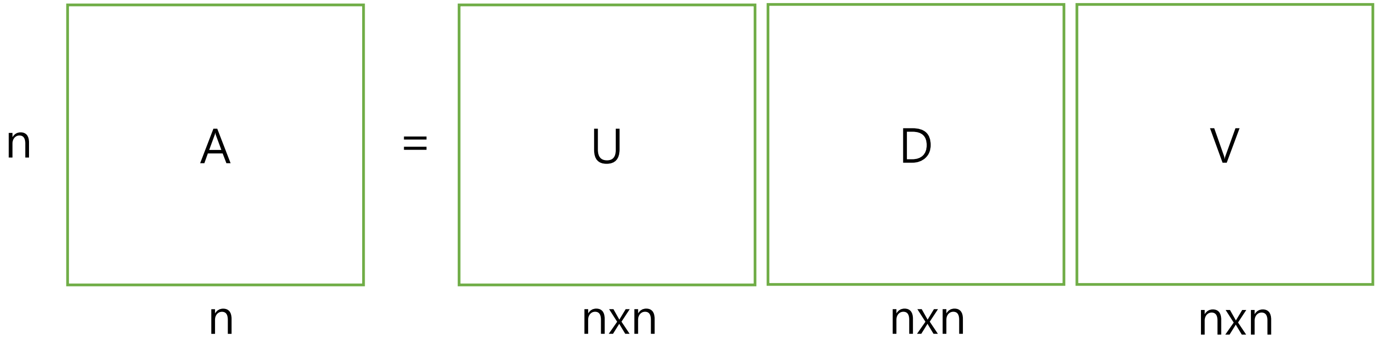
Eigendecomposition

- Matrix decomposition of a **square** matrix $A \in \mathbb{R}^{n \times n}$
- Pairs of eigenvalues and eigenvectors (λ, χ)
- Often called as **diagonalization**

$$A = UDU^{-1}$$

where U is a matrix composed of eigenvectors χ, D is diagonal matrix with non-degenerate eigenvalues λ.

- $A^2 = UD^2U^{-1}$
- $A^n = UD^nU^{-1}$
- $A^{-1} = UD^{-1}U^{-1}$



n

A = U D V

n nxn nxn nxn

Matrix Decomposition

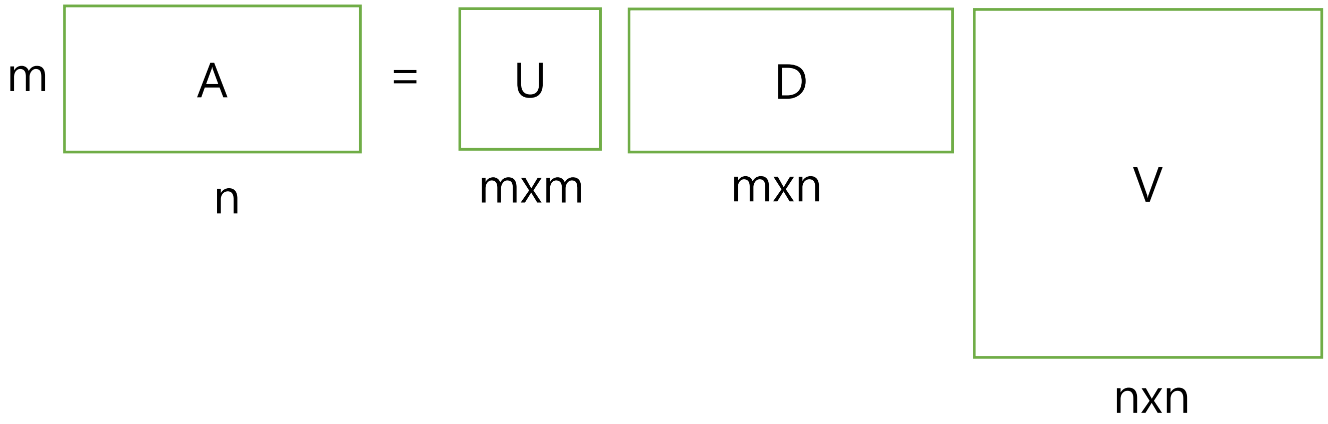
Singular-value Decomposition

- Singular value decomposition (SVD)is a factorization of a matrix
- Generalization of eigendecomposition of a matrix
- Given a matrix $A \in \mathbb{R}^{m \times n}$,

$$A = UDV^*$$

where $U \in \mathbb{R}^{m \times m}$ is a unitary matrix (i.e., $UU^* = I$), $D \in \mathbb{R}^{m \times n}$ is retangular diagonal matrix with non-negative real entries, and $V \in \mathbb{R}^{n \times n}$ is also a unitary matrix.

- U: left singular vector, V: right singular vector
- Diagonals of D, σ, are the singular values



m

A

n

=

U

mxm

D

mxn

V

nxn

PCA

- Goal
  - Minimize $\mathbf{u}^\mathsf{T} \Sigma \mathbf{u}$
  - Subject to $\mathbf{u}^\mathsf{T} \mathbf{u} = 1$

- Use Lagrange Multipliers to minimize

$$f(\mathbf{u}) = \mathbf{u}^\mathsf{T} \Sigma \mathbf{u} - \lambda[\mathbf{u}^\mathsf{T} \mathbf{u} - 1]$$

- Set its derivative to 0: $\Sigma \mathbf{u} - \lambda \mathbf{u} = 0$

- Definition of eigenvalue λ and eigenvector $\mathbf{u}$!

- If multiple vectors $\mathbf{u}_\mathbf{i}$
  - Minimize the sum of independent terms
  - Each is eigen value/vector

$$\Sigma \mathbf{u}_i = \underset{\substack{\nearrow \\ \text{Eigenvalue}}}{\lambda_i} \underset{\substack{\uparrow \\ \text{Eigenvector}}}{\mathbf{u}_i}$$

PCA

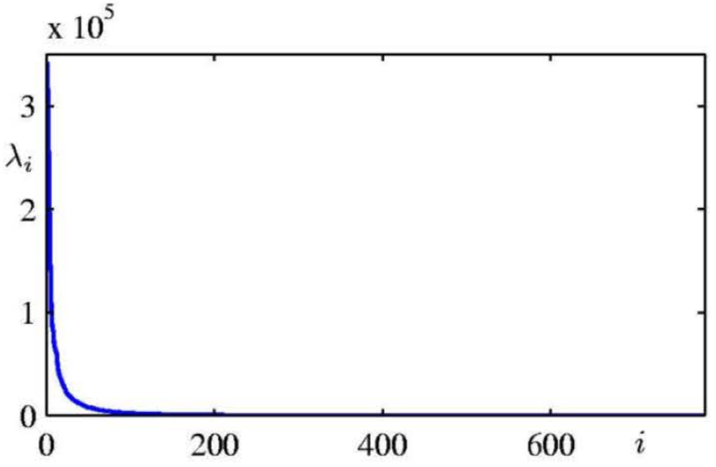
- Minimize $E_k = \sum_{i=k+1}^{d} \mathbf{u}_i^{\mathsf{T}} \, \Sigma \, \mathbf{u}_i$

$$\rightarrow \quad \Sigma \mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Eigenvalue Eigenvector

$$\Rightarrow E_k = \sum_{i=k+1}^{d} \mathbf{u}_i^{\mathsf{T}} \, \Sigma \mathbf{u}_i = \sum_{i=k+1}^{d} \mathbf{u}_i^{\mathsf{T}} \, \lambda_i \, \mathbf{u}_i$$

$$= \sum_{i=k+1}^{d} \lambda_i \, \mathbf{u}_i^{\mathsf{T}} \mathbf{u}_i = \sum_{i=k+1}^{d} \lambda_i$$



(a)

- Therefore, to minimize E_k, take the smallest eigenvalues $\{ \lambda_i \}$

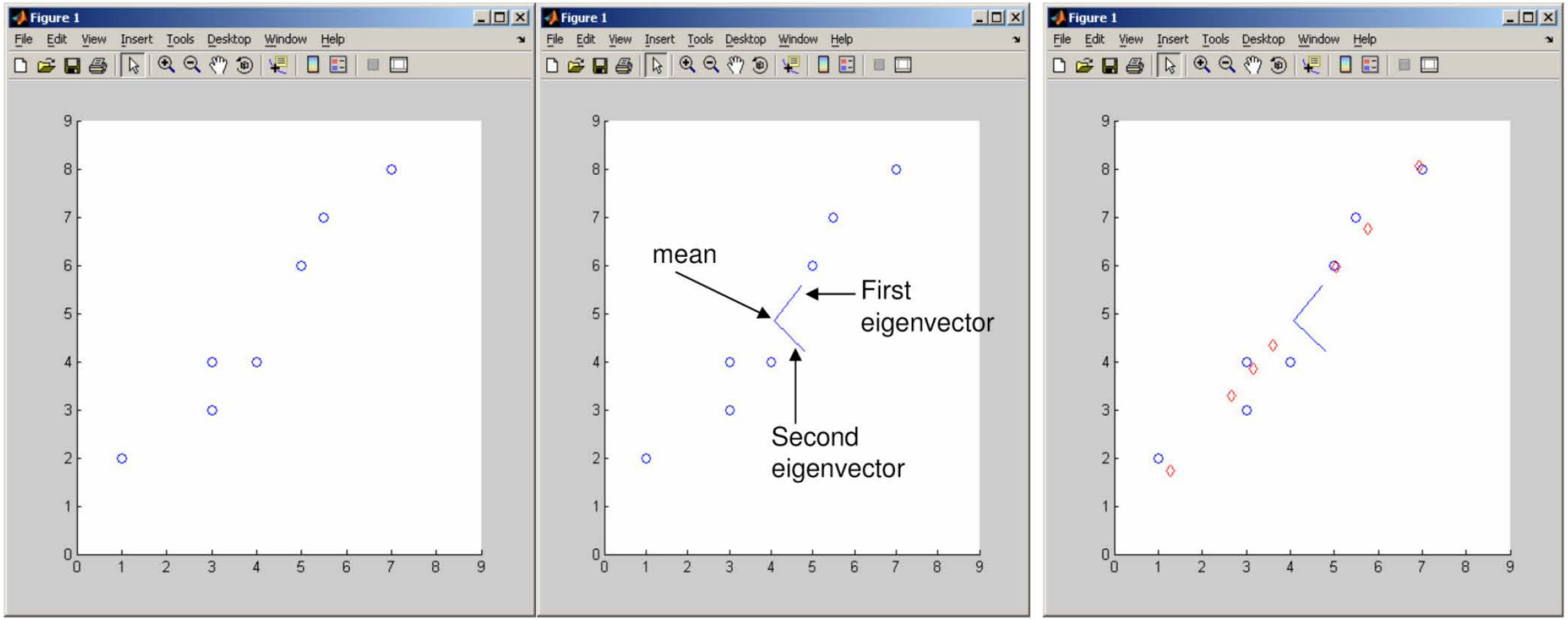
PCA

PCA algorithm(X, *k*): top *k* eigenvalues/eigenvectors

% X = d × N data matrix,
% … each data point x^n = column vector

- $\underline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}^{n}$
- A ← subtract mean $\underline{x}$ from each column vector x^n in X
- $\Sigma \leftarrow A\,A^T$ … covariance matrix of A
- $\{ \lambda_i, \mathbf{u}_i \}_{i=1..d}$ = eigenvectors/eigenvalues of Σ
  … $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- Return $\{ \lambda_i, \mathbf{u}_i \}_{i=1..k}$
  % top *k* principle components

PCA



Reconstructed data using only first eigenvector (k=1)

PCA and SVD

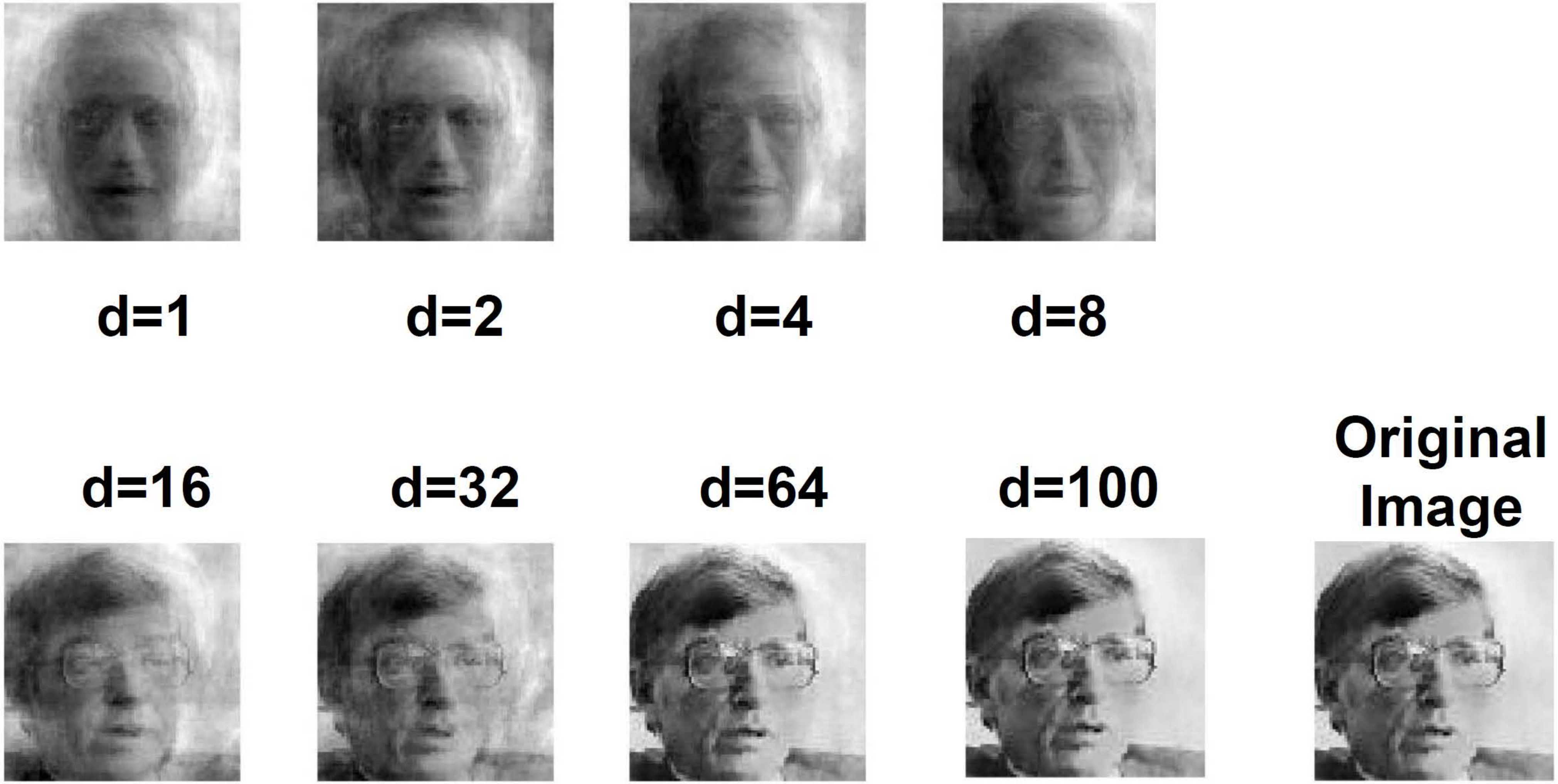
- Compute the principal components by SVD of X:

$$\underline{X = U\Sigma V^T}$$

$$\begin{aligned} XX^T &= U\Sigma V^T (U\Sigma V^T)^T = \\ &= U\Sigma V^T V \Sigma^T U^T = \underline{U\tilde{\Sigma}^2 U^T} \end{aligned}$$

- Thus, the left singular vectors of X are the principal components!
- Sort them by the size of the singular values of X

PCA for image compression



d=1

d=2

d=4

d=8

d=16

d=32

d=64

d=100

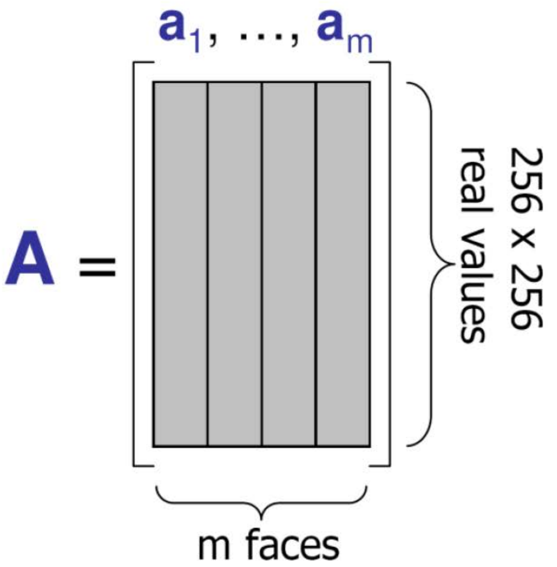
Original Image

Eigenface

- Example dataset: image of faces
  - Famous Eigenface approchaes [Turh & Pentland], [Sirovich & Kirby]

- Each face **a** is
  - 256x256 values (luminance at location)
  - **a** in $\mathbb{R}^{256\times 256}$ as a vector

- Form $A = [\, \mathbf{a}_1 , \ldots, \mathbf{a}_m \,]$
- Compute $\Sigma = AA^T$
- Problem: Σ is Hugggggggeeeee! 64k x 64k



Computational complexity

- Suppose m instances, each of size d
    - Eigenfaces: m=500 faces, each of size d = 64k

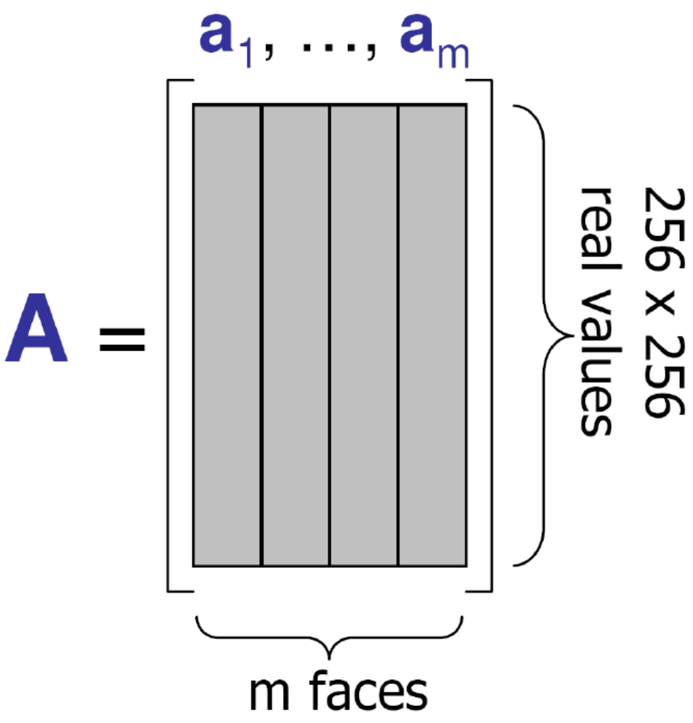
- Given a d x d covariance matrix Σ , we can compute
    - all d eigenvalues / eigenvectors in O(d^3)
    - First k eigenvalues / eigenvectors in O(kd^2)

- But if d=64k, still very expensive

Clever workaround

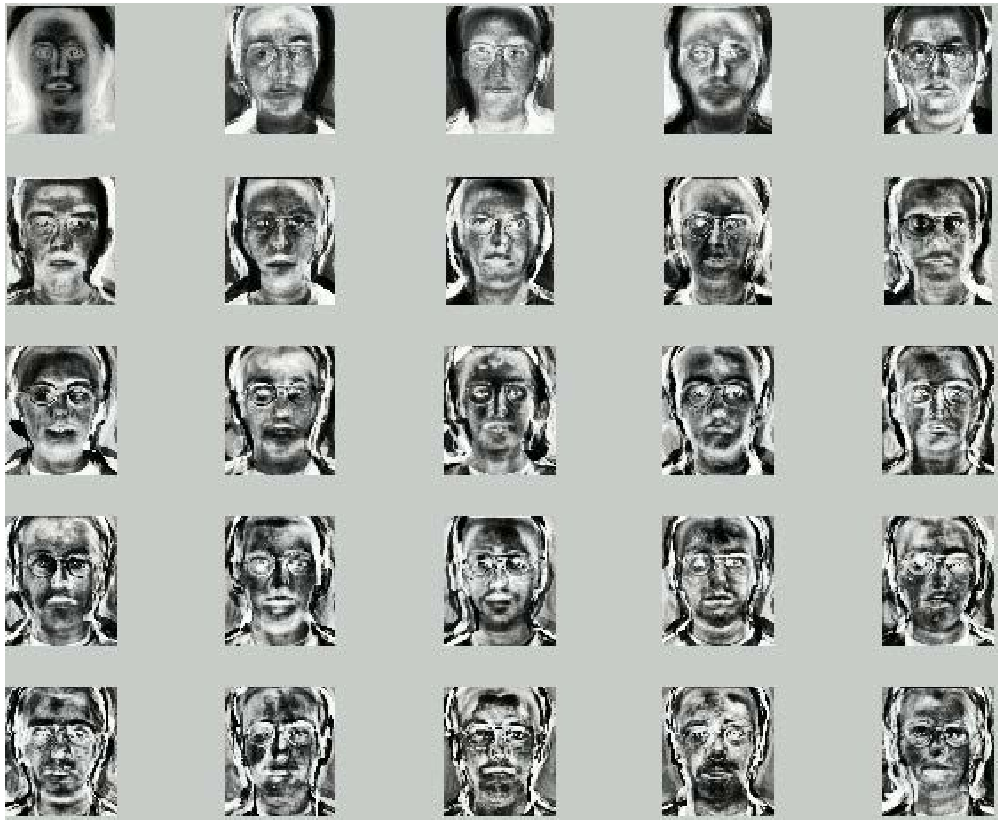
- Note that m << 64k
- Use $L = A^T A$ instead of $\Sigma = AA^T$
- If $\mathbf{v}$ is a eigenvector of L, then $A\mathbf{v}$ is an eigenvector of Σ

Proof: $$\begin{aligned} L\ \mathbf{v} &= \gamma\, \mathbf{v} \\ A^T A\ \mathbf{v} &= \gamma\, \mathbf{v} \\ A\,(A^T A\ \mathbf{v}) &= A(\gamma\, \mathbf{v}) = \gamma\, A\mathbf{v} \\ (A\ A^T)A\ \mathbf{v} &= \gamma\,(A\mathbf{v}) \\ \Sigma\,(A\mathbf{v}) &= \gamma\,(A\mathbf{v}) \end{aligned}$$



Dimensionality reduction

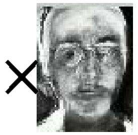
We can represent a face using all of the pixels in a given image (# features = # pixels)



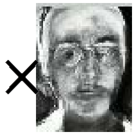
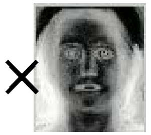
More effective method: represent each face as a linear combination of *eigenfaces* (# features = 20)

Dimensionality reduction example

represent each face as a linear combination of *eigenfaces*

$= \alpha_1^{(1)} \times$  $+ \alpha_2^{(1)} \times$ $+ \cdots + \alpha_{20}^{(1)} \times$

$$\boldsymbol{x}^{(1)} = \left\langle \alpha_1^{(1)},\ \alpha_2^{(1)},\ \cdots,\ \alpha_{20}^{(1)} \right\rangle$$

$= \alpha_1^{(2)} \times$  $+ \alpha_2^{(2)} \times$  $+ \cdots + \alpha_{20}^{(2)} \times$

$$\boldsymbol{x}^{(2)} = \left\langle \alpha_1^{(2)},\ \alpha_2^{(2)},\ \cdots,\ \alpha_{20}^{(2)} \right\rangle$$

# of features is now 20 instead of # of pixels in images

Comments…

- PCA performs dimensionality reduction via linear projection of the high dimensional data into a lower dimensional subspace.

- It accommodates the maximum variance in the data.

- The first principal component has the largest possible variance, the second principal component has the next largest and so on.

- The principal components are the eigenvectors of the co-variance matrix and hence also orthogonal.

- A disadvantage of PCA is that the transformed data loses semantics present in the original features.