Lecture 24:

– Principal Component Analysis
This week

• Motivation
• PCA algorithms
• Applications
• PCA shortcomings
• Autoencoders
• Kernel PCA
PCA Applications

• Data Visualization
• Data Compression
• Noise Reduction
• Learning
• Anomaly detection
Data Visualization

Example:

- Given 53 blood and urine samples (features) from 65 people.
- How can we visualize the measurements?
Data Visualization

- Matrix format (65x53)

<table>
<thead>
<tr>
<th>Instances</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>H-WBC</td>
</tr>
<tr>
<td>A2</td>
<td>H-RBC</td>
</tr>
<tr>
<td>A3</td>
<td>H-Hgb</td>
</tr>
<tr>
<td>A4</td>
<td>H-Hct</td>
</tr>
<tr>
<td>A5</td>
<td>H-MCV</td>
</tr>
<tr>
<td>A6</td>
<td>H-MCH</td>
</tr>
<tr>
<td>A7</td>
<td>H-MCHC</td>
</tr>
</tbody>
</table>

Difficult to see the correlations between the features...
Data Visualization

• Spectral format (65 curves, one for each person)

Difficult to compare the different patients...
Data Visualization

- Spectral format (53 pictures, one for each feature)

Difficult to see the correlations between the features...
How can we visualize the other variables???

... difficult to see in 4 or higher dimensional spaces...
Data Visualization

• Is there a representation better than the coordinate axes?

• Is it really necessary to show all the 53 dimensions?
  - ... what if there are strong correlations between the features?

• How could we find the smallest subspace of the 53-D space that keeps the most information about the original data?

• A solution: Principal Component Analysis
PCA algorithms
Principal Component Analysis

PCA:

Orthogonal projection of the data onto a lower-dimension linear space that...

• maximizes variance of projected data (purple line)

• minimizes mean squared distance between
  - data point and
  - projections (sum of blue lines)
Principal Component Analysis

Idea:

• Given data points in a d-dimensional space, project them into a lower dimensional space while preserving as much information as possible.
  - Find best planar approximation to 3D data
  - Find best 12-D approximation to $10^4$-D data

• In particular, choose projection that minimizes squared error in reconstructing the original data.
Principal Component Analysis

- **PCA Vectors** originate from the center of mass.

- Principal component #1: points in the direction of the **largest variance**.

- Each subsequent principal component
  - is **orthogonal** to the previous ones, and
  - points in the directions of the **largest variance** of the residual subspace.
2D Gaussian dataset
1\textsuperscript{st} PCA axis
2\textsuperscript{nd} PCA axis
PCA algorithm I (sequential)

Given the centered data \(\{\mathbf{x}_1, ..., \mathbf{x}_m\}\), compute the principal vectors:

\[
\mathbf{w}_1 = \arg \max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \{(\mathbf{w}^T \mathbf{x}_i)^2\} \quad 1^{\text{st}} \text{ PCA vector}
\]

We maximize the variance of projection of \(\mathbf{x}\)

\[
\mathbf{w}_2 = \arg \max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \{[\mathbf{w}^T (\mathbf{x}_i - \mathbf{w}_1 \mathbf{w}_1^T \mathbf{x}_i)]^2\} \quad k^{\text{th}} \text{ PCA vector}
\]

We maximize the variance of the projection in the residual subspace

\[\mathbf{x}' = \mathbf{w}_1 (\mathbf{w}_1^T \mathbf{x})\]
PCA algorithm I (sequential)

Given $\mathbf{w}_1, \ldots, \mathbf{w}_{k-1}$, we calculate $\mathbf{w}_k$ principal vector as before:

Maximize the variance of projection of $\mathbf{x}$

$$
\mathbf{w}_k = \arg \max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \left\{ \mathbf{w}^T (\mathbf{x}_i - \sum_{j=1}^{k-1} \mathbf{w}_j \mathbf{w}_j^T \mathbf{x}_i) \right\}^2
$$

$k^{th}$ PCA vector

$\mathbf{x}'$ PCA reconstruction

We maximize the variance of the projection in the residual subspace

$$
\mathbf{x}' = \mathbf{w}_1(\mathbf{w}_1^T \mathbf{x}) + \mathbf{w}_2(\mathbf{w}_2^T \mathbf{x})
$$
PCA algorithm II
(sample covariance matrix)

• Given data \( \{ \mathbf{x}_1, \ldots, \mathbf{x}_m \} \), compute covariance matrix \( \Sigma \)

\[
\Sigma = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^T
\]

where

\[
\overline{\mathbf{x}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i
\]

• **PCA** basis vectors = the eigenvectors of \( \Sigma \)

• Larger eigenvalue \( \Rightarrow \) more important eigenvectors
PCA algorithm II
(sample covariance matrix)

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

% $X = N \times m$ data matrix,
% ... each data point $x_i =$ column vector, $i=1..m$

- $x = \frac{1}{m} \sum_{i=1}^{m} x_i$
- $X \leftarrow$ subtract mean $x$ from each column vector $x_i$ in $X$
- $\Sigma \leftarrow xx^T$ ... covariance matrix of $X$
- $\{ \lambda_i, u_i \}_{i=1..N} =$ eigenvectors/eigenvalues of $\Sigma$
  ... $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$
- Return $\{ \lambda_i, u_i \}_{i=1..k}$
  % top $k$ PCA components
PCA algorithm III
(SVD of the data matrix)

Singular Value Decomposition of the centered data matrix $X$.

$$X = [x_1, \ldots, x_m] \in \mathbb{R}^{N \times m},$$

$m$: number of instances,
$N$: dimension

$$X_{\text{features} \times \text{samples}} = USV^T$$

$$X = \begin{bmatrix} \text{samples} \\ \hline \text{significant} \& \text{noise} \end{bmatrix}, \quad S = \begin{bmatrix} \text{sig.} \& \text{noise} \end{bmatrix}, \quad V^T = \begin{bmatrix} \text{significant} \\ \hline \text{noise} \end{bmatrix}$$
PCA algorithm III

- **Columns of U**
  - the principal vectors, \( \{ u^{(1)}, ..., u^{(k)} \} \)
  - orthogonal and has unit norm – so \( U^T U = I \)
  - Can reconstruct the data using linear combinations of \( \{ u^{(1)}, ..., u^{(k)} \} \)

- **Matrix S**
  - Diagonal
  - Shows importance of each eigenvector

- **Columns of \( V^T \)**
  - The coefficients for reconstructing the samples
Applications
Face Recognition
Face Recognition

- Want to identify specific person, based on facial image
- Robust to glasses, lighting, ...
  - Can’t just use the given 256 x 256 pixels
Applying PCA: Eigenfaces

**Method A:** Build a PCA subspace for each person and check which subspace can reconstruct the test image the best.

**Method B:** Build one PCA database for the whole dataset and then classify based on the weights.

- **Example data set:** Images of faces
  - Famous Eigenface approach
    - [Turk & Pentland], [Sirovich & Kirby]
  - Each face $x$ is ...
    - $256 \times 256$ values (luminance at location)
    - $x$ in $\mathbb{R}^{256 \times 256}$ (view as 64K dim vector)

- Form $X = [x_1, \ldots, x_m]$ centered data mtx
- Compute $\Sigma = XX^T$
- Problem: $\Sigma$ is 64K $\times$ 64K ... HUGE!!!
Computational Complexity

• Suppose \( m \) instances, each of size \( N \)
  • Eigenfaces: \( m=500 \) faces, each of size \( N=64K \)
• Given \( N \times N \) covariance matrix \( \Sigma \), can compute
  • all \( N \) eigenvectors/eigenvalues in \( O(N^3) \)
  • first \( k \) eigenvectors/eigenvalues in \( O(k \ N^2) \)

• But if \( N=64K \), EXPENSIVE!
A Clever Workaround

- Note that \( m << 64K \)
- Use \( L = XX^T \) instead of \( \Sigma = XX^T \)
- If \( v \) is eigenvector of \( L \) then \( Xv \) is eigenvector of \( \Sigma \)

Proof:

\[
\begin{align*}
L \ v &= \gamma \ v \\
X^T X \ v &= \gamma \ v \\
X (X^T X \ v) &= X(\gamma \ v) = \gamma \ Xv \\
(XX^T)X \ v &= \gamma \ (Xv) \\
\Sigma \ (Xv) &= \gamma \ (Xv)
\end{align*}
\]
Principle Components (Method B)
Principle Components (Method B)

• … faster if train with …
  - only people w/out glasses
  - same lighting conditions
Shortcomings

• Requires carefully controlled data:
  - All faces centered in frame
  - Same size
  - Some sensitivity to angle

• Method is completely knowledge free
  - (sometimes this is good!)
  - Doesn’t know that faces are wrapped around 3D objects (heads)
  - Makes no effort to preserve class distinctions
Happiness subspace (method A)
Disgust subspace (method A)
Facial Expression Recognition

Movies

Surprise
Facial Expression Recognition

Movies
Facial Expression Recognition
Movies

Disgust
Image Compression
• Divide the original 372x492 image into patches:
  - Each patch is an instance
• View each as a 144-D vector
$L_2$ error and PCA dim

![Graph showing the relationship between $L_2$ error and PCA dimensionality.](image)
PCA compression: 144D => 60D
PCA compression: 144D => 16D
16 most important eigenvectors

slide by Barnabás Póczos and Aarti Singh
PCA compression: 144D => 6D
6 most important eigenvectors
PCA compression: 144D => 3D
3 most important eigenvectors
PCA compression: 144D => 1D
60 most important eigenvectors

- Looks like the discrete cosine bases of JPG!...
2D Discrete Cosine Basis

Noise Filtering
Noise Filtering

\[ x \] \rightarrow_U \rightarrow \rightarrow \rightarrow Ux \rightarrow \rightarrow \rightarrow \rightarrow x'
Noisy image
Denoised image using 15 PCA components
PCA Shortcomings
Problematic Data Set for PCA

- PCA doesn’t know labels!
PCA vs. Fisher Linear Discriminant

**Principal Component Analysis**
- higher variance
- bad for discriminability

**Fisher Linear Discriminant**
- smaller variance
- good discriminability
Problematic Data Set for PCA

- PCA cannot capture NON-LINEAR structure!
PCA Conclusions

• PCA
  - Finds orthonormal basis for data
  - Sorts dimensions in order of “importance”
  - Discard low significance dimensions

• Uses:
  - Get compact description
  - Ignore noise
  - Improve classification (hopefully)

• Not magic:
  - Doesn’t know class labels
  - Can only capture linear variations

• One of many tricks to reduce dimensionality!