Machine Learning (CE 40717)

Fall 2024

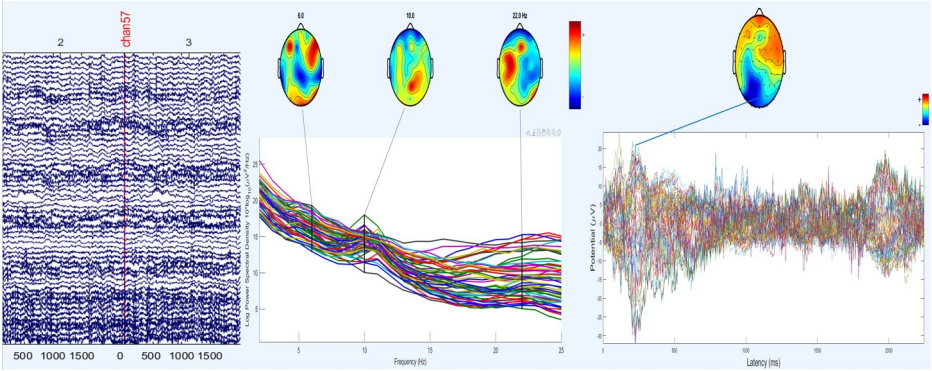
Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

November 7, 2024

High Dimensional Data

- High dimensions has many features.
- EEG signals from the brain, recorded with 56 channels and 3000 time points per trial.



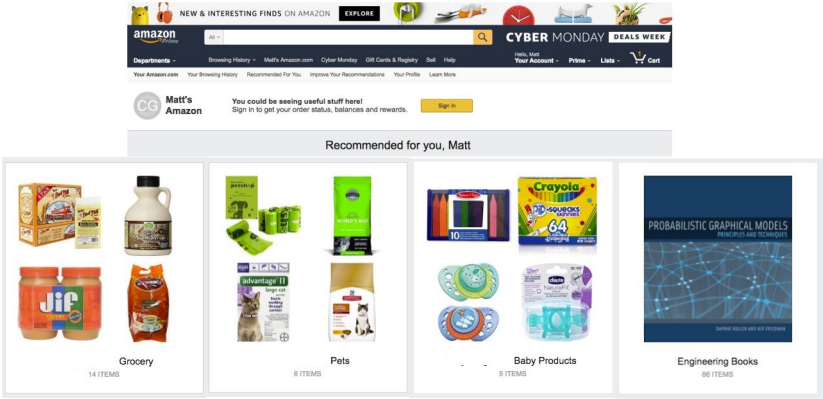
High Dimensional Data

- Social media

Figure 1: Figure reference

High Dimensional Data

- Customer purchase data



Dimensionality Reduction Benefits

- **Visualization**
  - Project high dimensional data into 2D or 3D.
- **Helps avoid overfitting**
  - Reducing noise by reducing features.
  - Improves accuracy by reducing noise.
- **More efficient use of resources**
  - Time, Memory, CPU

Dimensionality Reduction Techniques

- **Feature Selection**
  - Select a subset from a given feature set.
- **Feature Extraction**
  - A linear or non-linear transform from the original feature space to a lower dimension space.

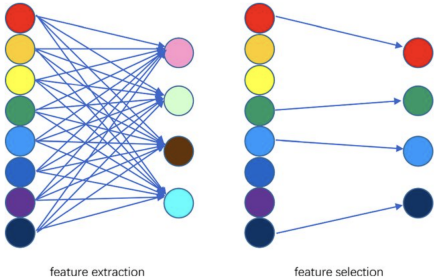


Figure 2: Figure reference

Dimensionality Reduction Purpose

- Maximize retention of **important information** while reducing dimensionality.
- What is **important information**?

Purpose: Variance of Data

- Maximize retention of **important information** while reducing dimensionality.
- **Information:** Variance of projected data

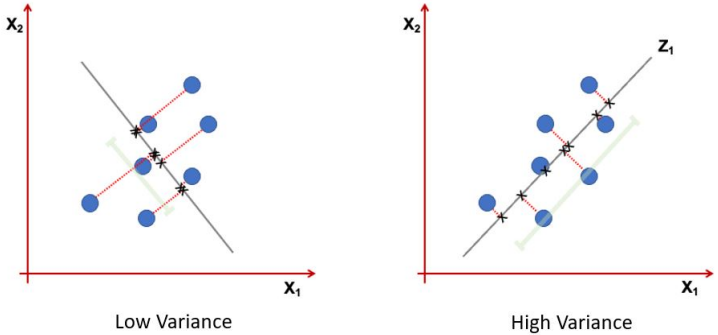


Figure 3: Figure reference

Purpose: Local Geometric Neighborhood

- **Information:** Preserve local geometric neighborhood.

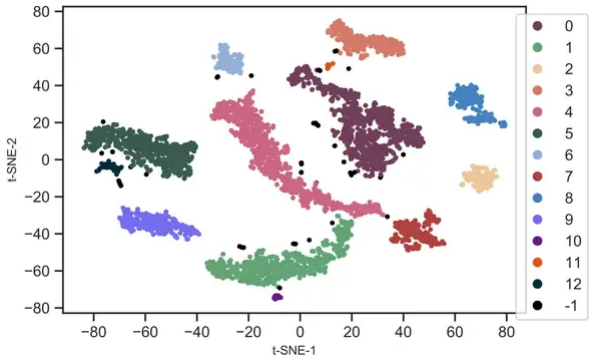


Figure 4: Figure reference

Purpose: Local and Global Geometric Neighborhood

- **Information:** Preserve both local and global geometric neighborhood.

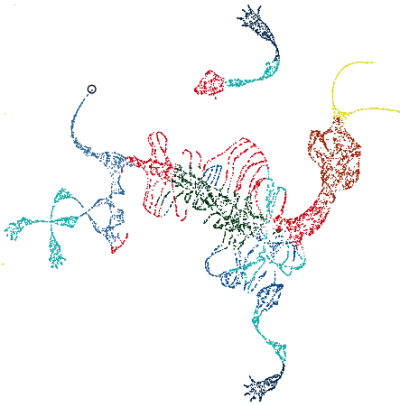


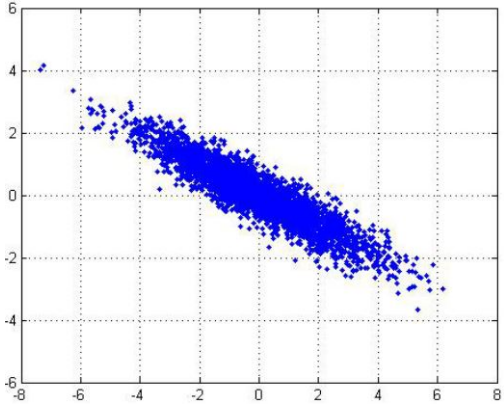
Figure 5: Figure reference

Idea

- Given data points in a d-dimensional space, project them into a lower dimensional space while preserving as much information as possible:
  - Find the best planar approximation of 3D data.
  - Find the best 12-D approximation of 104-D data.
- In particular, choose projection that minimizes the squared error in reconstructing the original data.

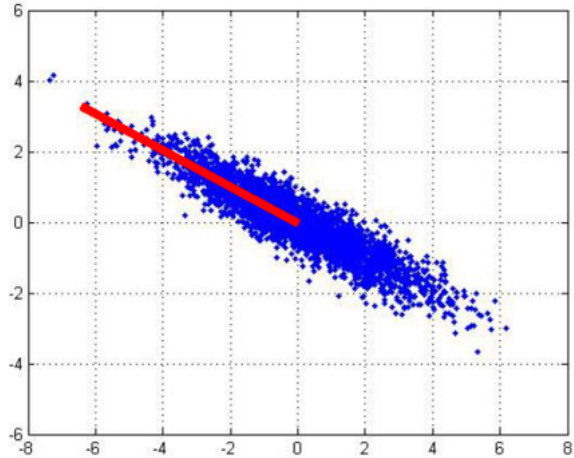
Principal Components Idea

- 2D Gaussian dataset:



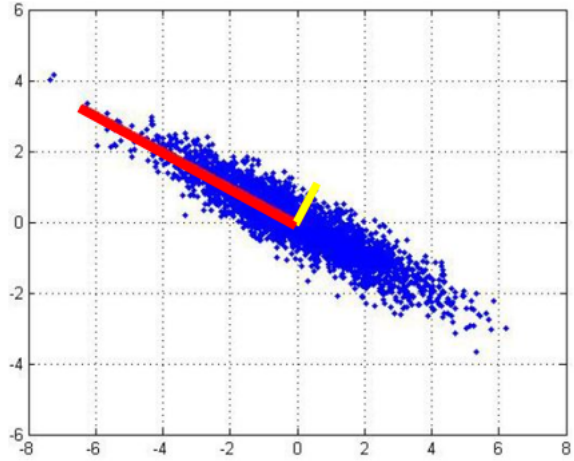
Principal Components Idea

- First PCA axis:



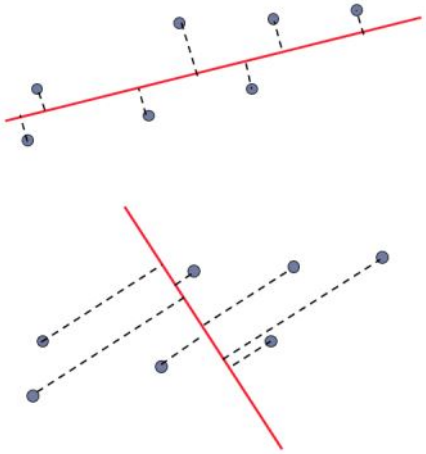
Principal Components Idea

- First and second PCA axes:



Random vs. Principal Projection

- Random direction versus principal component:



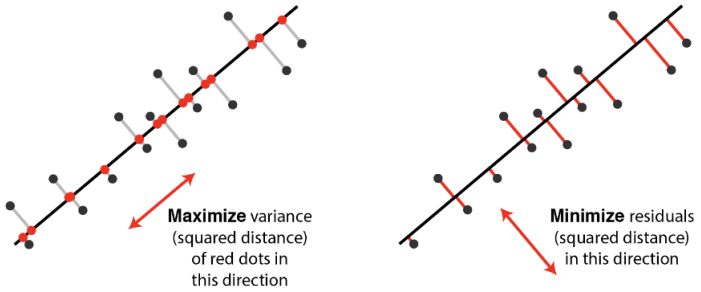
Definition

- **Goal**: reducing the dimensionality of the data while preserving important aspects of the data.
- Suppose $\mathbf{X} = \begin{pmatrix} \mathbf{X}_1^T \\ \vdots \\ \mathbf{X}_N^T \end{pmatrix}_{N \times d} = \begin{pmatrix} F_1 & F_2 & & F_d \\ x_{11} & x_{12} & \dots & x_{1d} \\ x_{21} & x_{22} & \dots & x_{2d} \\ \vdots & & & \\ x_{N1} & x_{N2} & \dots & x_{Nd} \end{pmatrix}$
- $\mathbf{X}_{N \times d} \xrightarrow{\text{PCA}} \tilde{\mathbf{X}}_{N \times k} \quad \text{with} \quad k \leq d$
- **Assumption**: Data is mean-centered, which is: $\mu_x = \frac{1}{N} \sum_{i=1}^{N} X_i = 0_{d \times 1}$

Interpretations

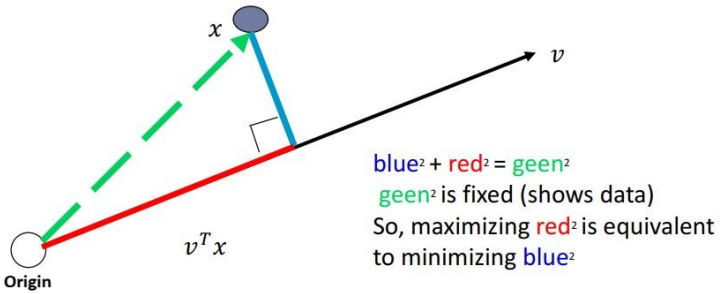
Orthogonal projection of the data onto a **lower-dimensional** linear **subspace** that:

- **Interpretation 1.** Maximizes variance of projected data.
- **Interpretation 2.** Minimizes the sum of squared distances to the subspace.



Equivalence of the Interpretations

- Minimizing the sum of square distances to the subspace is **equivalent** to maximizing the sum of squares of the projections on that subspace.



Equivalence of the Interpretations

Principal Components (PCs): A set of **orthonormal** vectors ($v = [v_1, v_2, ..., v_k]$) (where each v_i is $d \times 1$) generated by PCA, which fulfill both of the interpretations.

Interpretation 1. Maximizes variance of projected data

- Projection of data points on v_1

$$\Pi = \Pi_{v_1}\{X_1, \ldots, X_N\} = \{v_1^T X_1, \ldots, v_1^T X_N\}$$

- Note that $\mathrm{Var}(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$$\mathbb{E}[X] = 0 \rightarrow \mathrm{Var}(\Pi) = \frac{1}{N}\left(\sum_{i=1}^{N} (v_1^T X_i)^2\right)$$

Pre-processing

- **Mean-center the data.**
  - **Zero**ing out the **mean** of each feature.
- **Scaling to normalize each feature to have variance 1 (an arbitrary step).**
  - Might affect results.
  - It helps when unit of measurements of features are different and some features may be ignored without normalization.

Background

- Before jumping to PCA algorithm, we should be familiar with followings:
  - What are eigenvalues and eigenvectors?
  - Sample covariance matrix
  - Lagrangian multiplier

What are Eigenvalues and Eigenvectors?

- **Eigenvector:** A non-zero vector that multiplies only by a scalar factor when a linear transformation is applied.
- **Eigenvalue:** The scalar factor by which the eigenvector is scaled.
- **Equation** for a $n \times n$ matrix:

$$Av = \lambda v$$

- Where
  - A: A square matrix
  - v: Eigenvector
  - λ: Eigenvalue

Geometrical Interpretation

- Eigenvectors point in the same direction (or opposite) after the transformation.
  - Eigenvectors do not change direction under a transformation.
- Eigenvalues represent how much the vector is stretched or compressed.
  - Eigenvalues tell us how much the vector is scaled.

$$A = \begin{pmatrix} 1 & \frac{1}{3} \\ \frac{4}{3} & 1 \end{pmatrix}$$

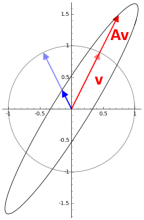


Figure 6: Figure reference

How to Find Eigenvalues and Eigenvectors?

- We know that

$$Av = \lambda v$$

- So

$$Av - \lambda v = 0$$

$$(A - \lambda I)v = 0$$

- v can not be zero, so:

$$det(A - \lambda I) = 0$$

- Solve for λ
- Substitute λ back into the equation $Av = \lambda v$ to find v.

Numerical Example

- Assume $A = \begin{pmatrix} 4 & -5 \\ 2 & -3 \end{pmatrix}$
- $A - \lambda I = ?$

Numerical Example (Continued)

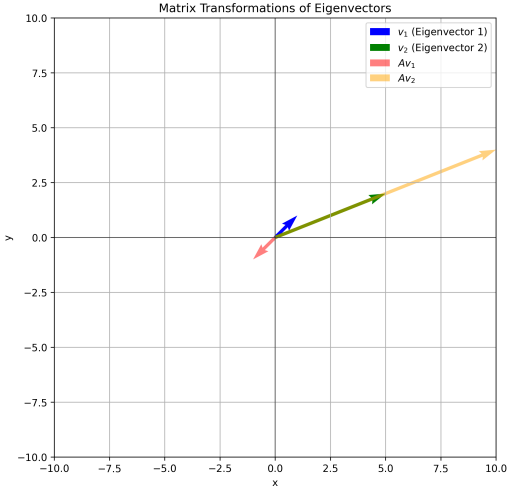
- Assume $A = \begin{pmatrix} 4 & -5 \\ 2 & -3 \end{pmatrix}$
- $A - \lambda I = \begin{pmatrix} 4-\lambda & -5 \\ 2 & -3-\lambda \end{pmatrix}$
- Determinant $(A - \lambda I) = (4-\lambda)(-3-\lambda) + 10 = \lambda^2 - \lambda - 2$

Numerical Example (Continued)

- Assume $A = \begin{pmatrix} 4 & -5 \\ 2 & -3 \end{pmatrix}$
- $A - \lambda I = \begin{pmatrix} 4-\lambda & -5 \\ 2 & -3-\lambda \end{pmatrix}$
- Determinant $(A - \lambda I) = (4-\lambda)(-3-\lambda) + 10 = \lambda^2 - \lambda - 2$
- $\lambda = -1$ or $\lambda = 2$
- $\lambda_1 = -1: \quad (A - \lambda_1 I)v_1 = \begin{bmatrix} 5 & -5 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \rightarrow v_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
- $\lambda_2 = 2: \quad (A - \lambda_2 I)v_2 = \begin{bmatrix} 2 & -5 \\ 2 & -5 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \rightarrow v_2 = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$

Visualization

- $Av = \lambda v$



What is Covariance?

- Covariance is a measure of how much two random features vary together.
- $\text{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[(Y - \mathbb{E}[Y])(X - \mathbb{E}[X])] = \text{Cov}(Y, X)$
- So covariance is symmetric.
- Such as heights and weights of individuals.

What is a Covariance Matrix?

- A **covariance matrix** generalizes the concept of covariance to multiple features.
- For a random vector $\mathbf{F} = [F_1, F_2, \ldots, F_d]$:

$$\Sigma = \begin{pmatrix} \text{Var}(F_1) & \text{Cov}(F_1, F_2) & \cdots & \text{Cov}(F_1, F_d) \\ \text{Cov}(F_2, F_1) & \text{Var}(F_2) & \cdots & \text{Cov}(F_2, F_d) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(F_d, F_1) & \text{Cov}(F_d, F_2) & \cdots & \text{Var}(F_d) \end{pmatrix}$$

- The diagonal elements are the variances, and off-diagonal elements are covariances.

Covariance Matrix Example

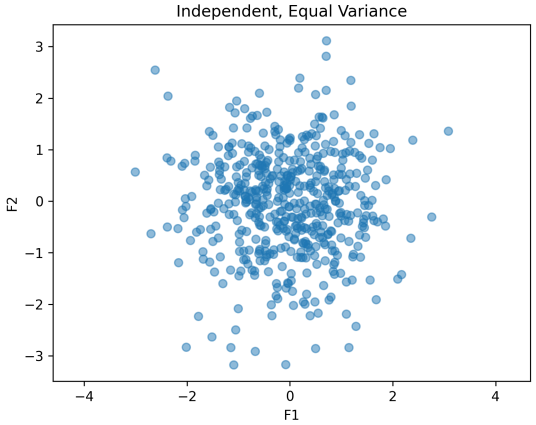
- Suppose there is two feature covariance matrix:

$$\Sigma = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & b \\ b & d \end{pmatrix}$$

- Why $b = c$?
- What is the relation between a, b, and d?

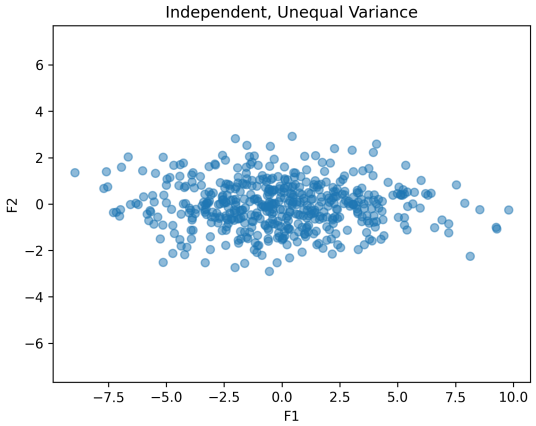
Covariance Matrix Example

- If $\Sigma = \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix}$, then:



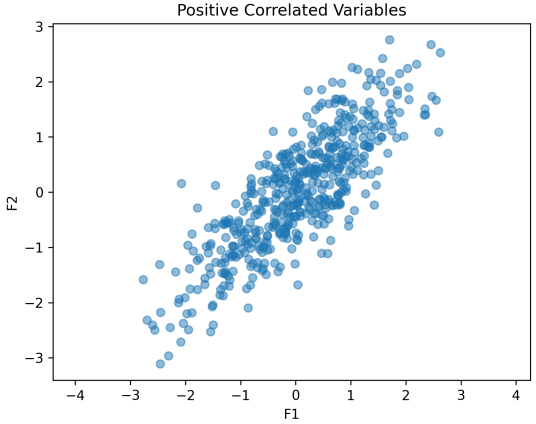
Covariance Matrix Example

- If $\Sigma = \begin{pmatrix} a & 0 \\ 0 & d \end{pmatrix}$ and $a > d$, then:



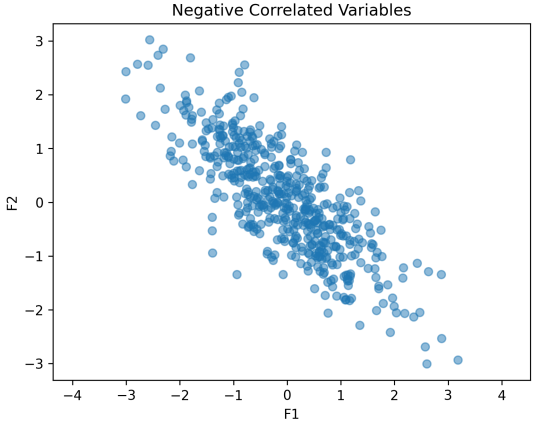
Covariance Matrix Example

- If $\Sigma = \begin{pmatrix} a & b \\ b & d \end{pmatrix}$, $a > d$, and $b > 0$, then:



Covariance Matrix Example

- If $\Sigma = \begin{pmatrix} a & b \\ b & d \end{pmatrix}$, $a > d$, and $b < 0$, then:



Sample Covariance Matrix

- In practice, we estimate covariance from sample data.
- **Sample Covariance Matrix**: Given N samples of d features, the sample covariance matrix Σ is:

$$\Sigma_{d\times d} = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X})(X_i - \bar{X})^T$$

- Where X_i is the i-th sample, and $\bar{X}_{d\times 1}$ is the mean of the samples.

Example Calculation of Sample Covariance Matrix

- Suppose we have the following three samples each one having two features F_1 and F_2:

Sample	$\mathbf{F}_1$	$\mathbf{F}_2$
X_1	3	3
X_2	4	7
X_3	5	8
$\bar{X}$	4	6

$$\Sigma = \frac{1}{N-1}\sum_{i=1}^{N}(X_i-\bar{X})(X_i-\bar{X})^T = \frac{1}{2}(\begin{pmatrix}1 & 3\\ 3 & 9\end{pmatrix}+\begin{pmatrix}0 & 0\\ 0 & 1\end{pmatrix}+\begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix}) = \begin{pmatrix}1 & 2.5\\ 2.5 & 7\end{pmatrix}$$

Lagrange Multiplier: Geometrical Interpretation

- We want to maximize $f(x)$ subject to $g(x) = 0$.
- The optimal point occurs where the gradient of $f(x)$ is proportional to the gradient of $g(x)$ (i.e., they are aligned or in opposite directions).

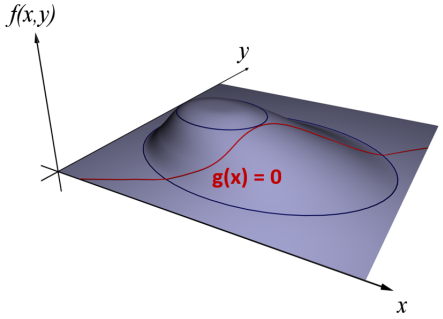


Figure 7: Figure reference

Lagrange Multiplier Method

- Combine the objective function and the constraint using the Lagrange function:

$$\mathscr{L}(x, \lambda) = f(x) + \lambda g(x)$$

- Solve the system of equations:

$$\nabla \mathscr{L}(x, \lambda) = 0$$

- Where:
  - $\mathscr{L}$ is lagrangian
  - λ is lagrange multiplier
  - $g(x)$ is equality constraint
  - $f(x)$ is function

Example Problem

- Maximize:

$$f(x_1, x_2) = x_1 + x_2$$

- Subject to:

$$x_1^2 + x_2^2 = 1$$

- Lagrangian:

$$L(x_1, x_2, \lambda) = x_1 + x_2 - \lambda(x_1^2 + x_2^2 - 1)$$

- Partial derivatives:

$$\begin{cases} \frac{\partial L}{\partial x_1} = 1 - 2\lambda x_1 = 0 \\ \frac{\partial L}{\partial x_2} = 1 - 2\lambda x_2 = 0 \\ \frac{\partial L}{\partial \lambda} = -(x_1^2 + x_2^2 - 1) = 0 \end{cases}$$

Example Problem (Continued)

- Solving:

$$\begin{cases} \lambda x_1 = \frac{1}{2} \\ \lambda x_2 = \frac{1}{2} \\ x_1^2 + x_2^2 = 1 \end{cases}$$

- Since $\lambda x_1 = \lambda x_2$, then $x_1 = x_2$.
- Substitute $x_1 = x_2$ into the constraint:

$$2x_1^2 = 1 \implies x_1 = \pm \frac{1}{\sqrt{2}}$$

- Optimal solution:

$$x_1 = x_2 = \frac{1}{\sqrt{2}}, \quad f_{\max} = x_1 + x_2 = \sqrt{2}$$

Generalization to Multiple Constraints

- The Lagrange multiplier method can be extended to cases with more than one constraint.
- For constraints $g_1(x) = 0, g_2(x) = 0, \ldots, g_m(x) = 0$, the Lagrangian becomes:

$$\mathscr{L}(x, \lambda_1, \lambda_2, \ldots) = f(x) - \lambda_1 g_1(x) - \lambda_2 g_2(x) - \ldots$$

Step 1: Expression for Variance

- The variance of the projected data onto the direction v is:

$$\text{Var}(Xv) = \frac{1}{n}\sum_{i=1}^{n}(x_i^\top v)^2$$

- This can be rewritten as:

$$\text{Var}(Xv) = \frac{1}{n}\|Xv\|^2 = \frac{1}{n}v^\top X^\top Xv = v^\top \Sigma v$$

Step 2: Maximization Problem

- We aim to maximize the variance $v^\top \Sigma v$ under the constraint that $\|v\| = 1$.
- This leads to the following optimization problem:

$$\max_{v} \; v^\top \Sigma v \quad \text{subject to} \quad \|v\| = 1$$

Step 3: Use of Lagrange Multipliers

- We introduce a Lagrange multiplier λ and define the Lagrangian:

$$L(v, \lambda) = v^\top \Sigma v - \lambda(v^\top v - 1)$$

- Taking the derivative with respect to v and setting it to 0:

$$\frac{\partial L}{\partial v} = 2\Sigma v - 2\lambda v = 0$$

- This simplifies to:

$$\Sigma v = \lambda v$$

- We find all $(v_1, \lambda_1), (v_2, \lambda_2), \cdots, (v_k, \lambda_k)$ as the k eigenvectors of Σ having largest eigenvalues: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$

Step 4: Interpretation

- The variance $v^\top \Sigma v$ is maximized when v is the eigenvector corresponding to the largest eigenvalue of Σ.
- The eigenvalue λ represents the variance in the direction of the eigenvector v.
- Conclusion: Eigenvectors of the covariance matrix maximize the variance of the projected data.

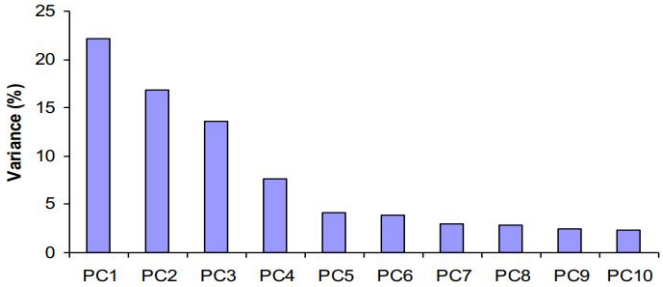
Sample Covariance Matrix

Algorithm 1 Sample Covariance Matrix

1: **Input:** $X \in \mathbb{R}^{N \times d}$ (data matrix with N data points and d dimensions)
2: Compute the mean of each feature: $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$
3: Subtract the mean from each data point (center the data): $\tilde{X} \leftarrow X - \bar{x}^T$
4: Compute the covariance matrix: $\Sigma = \frac{1}{N-1} \tilde{X}^T \tilde{X}$
5: Compute the eigenvalues and eigenvectors of Σ: $[\lambda_1, \lambda_2, \ldots, \lambda_d], [v_1, v_2, \ldots, v_d] = \text{eig}(\Sigma)$
6: Select the top k eigenvectors corresponding to the largest eigenvalues: $A \leftarrow [v_1, v_2, \ldots, v_k]$
7: Transform the data into the new subspace: $X' \leftarrow X \cdot A$
8: **Output:** $X' \in \mathbb{R}^{N \times k}$ (transformed data with reduced dimensions)

Number of Principal Components

- For d original dimensions, the sample covariance matrix is $d \times d$, and has up to d eigenvectors. So we can have up to d principal components.
- Can ignore the components of lesser significance.



- We lose some information, but if the eigenvalues are small, we don't lose much.

Number of Principal Components

- Select the desired variance ratio and select the principal components.

$$\min_k \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{d} \lambda_i} >= 0.9$$

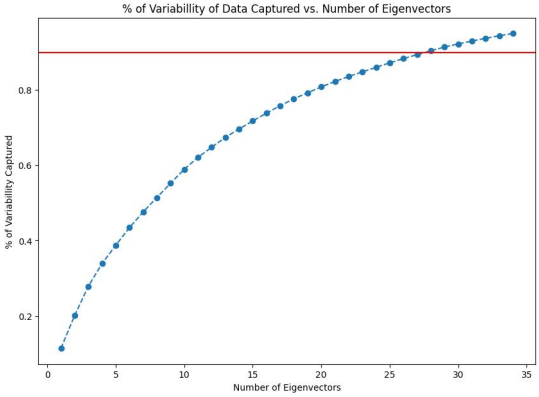


Image Compression

- Divide the original 372×492 image into patches.
  - Each patch is an instance containing 12×12 pixels on a grid.
- Consider each as a 144-D vector.

Image Compression

- 144D to 60D

Image Compression

- 144D to 16D

Image Compression

- The 16 most important eigenvectors

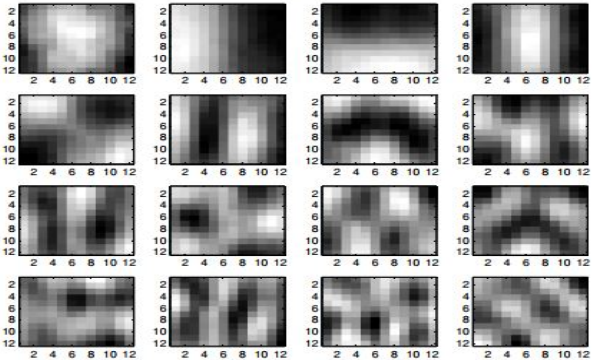
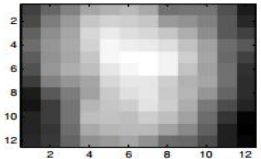


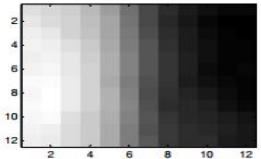
Image Compression

- 144D to 3D

Image Compression

- The 3 most important eigenvectors





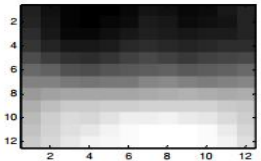
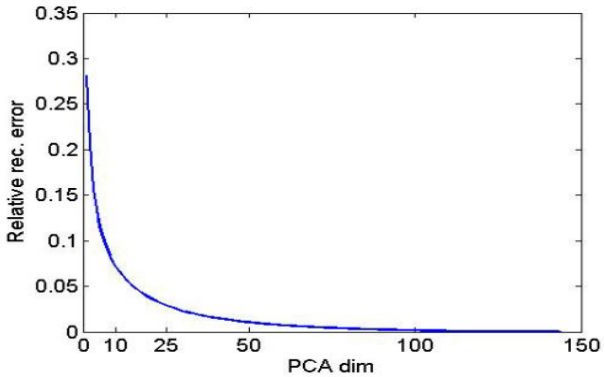


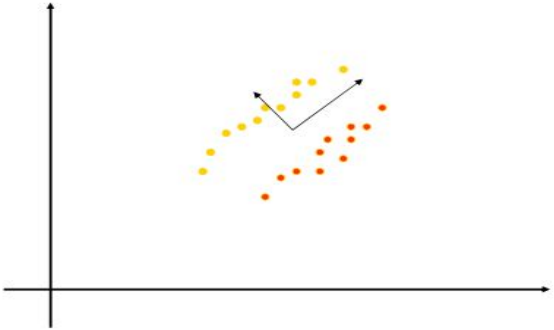
Image Compression

- L^2 error and PCA dimensions



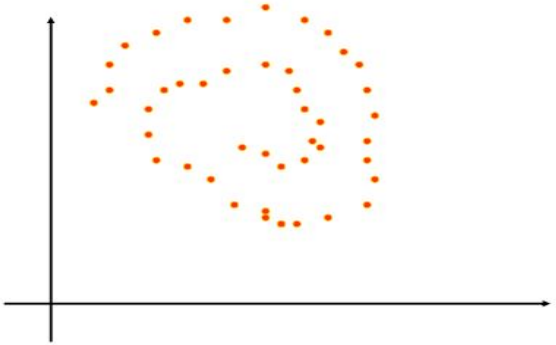
Class Labels

- PCA doesn't recognize about class labels.
- Alternative solution: Linear Discriminant Analysis (LDA)



Non-Linear

- PCA cannot capture non-Linear structure.
- Alternative solution: Kernel PCA



Other Methods

- **t-SNE**:
  - Non-linear method focusing on preserving local structure.
  - Often shows well-separated clusters, making it useful for visualization.
  - Computationally intensive; slower on large datasets.
- **UMAP:**
  - Non-linear method that preserves both local and some global structure.
  - Generally faster than t-SNE and scales better to large datasets.
  - Can capture more complex structures in the data.

PCA vs t-SNE vs UMAP

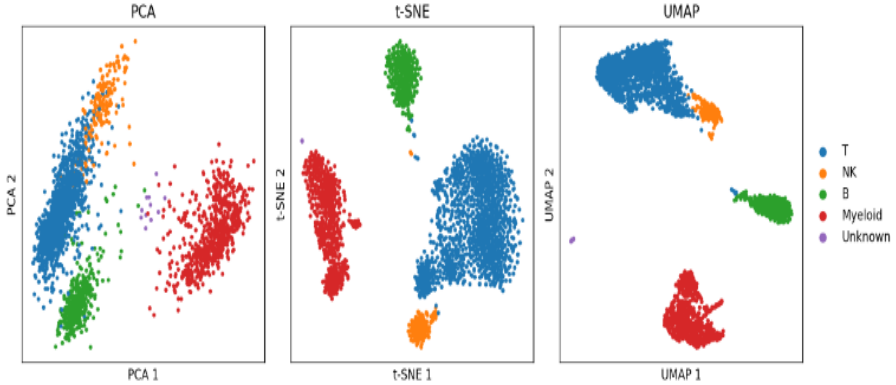


Figure 8: Figure reference

Conclusion

- PCA
  - Finds orthonormal basis for data
  - Sorts principal components in order of importance
  - Discards low significance principal components
- Applications
  - Get compact description
  - Remove noise
  - Improve classification (hopefully)
  - More efficient use of resources
  - Statistical
- Not magic
  - Doesn't recognize class labels
  - Can only capture linear variation

Contributions

- **This slide has been prepared thanks to:**
  - Mohammad Mowlavi

[1] M. Soleymani Baghshah, “Machine learning.” Lecture slides.

[2] B. Póczos, “Advanced introduction to machine learning.” Lecture slides. CMU-10715.

[3] M. Gormley, “Introduction to machine learning.” Lecture slides. 10-701.

[4] M. Gormley, “Introduction to machine learning.” Lecture slides. 10-301/10-601.

[5] F. Seyyedsalehi, “Machine learning and theory of machine learning.” Lecture slides. CE-477/CS-828.

[6] G. Strang, “Linear algebra and its applications,” 2000.