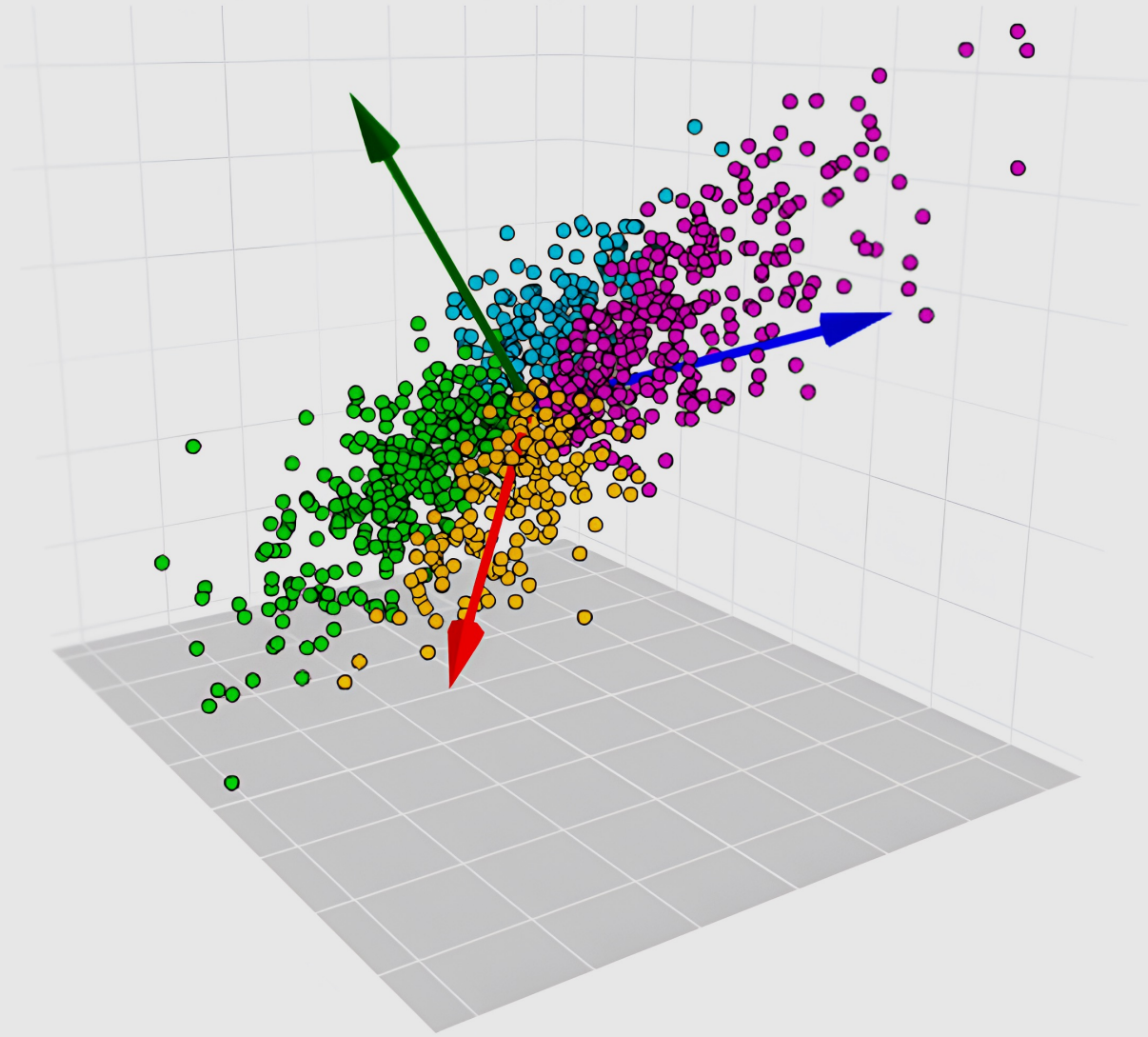
Georgia Tech

Principal Component Analysis



Mohsen Moghaddam, Ph.D.

Gary C. Butler Family Associate Professor
H. Milton Stewart School of Industrial and Systems Engineering
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

SAIL
Symbiotic & Augmented Intelligence Laboratory

Spring 2026

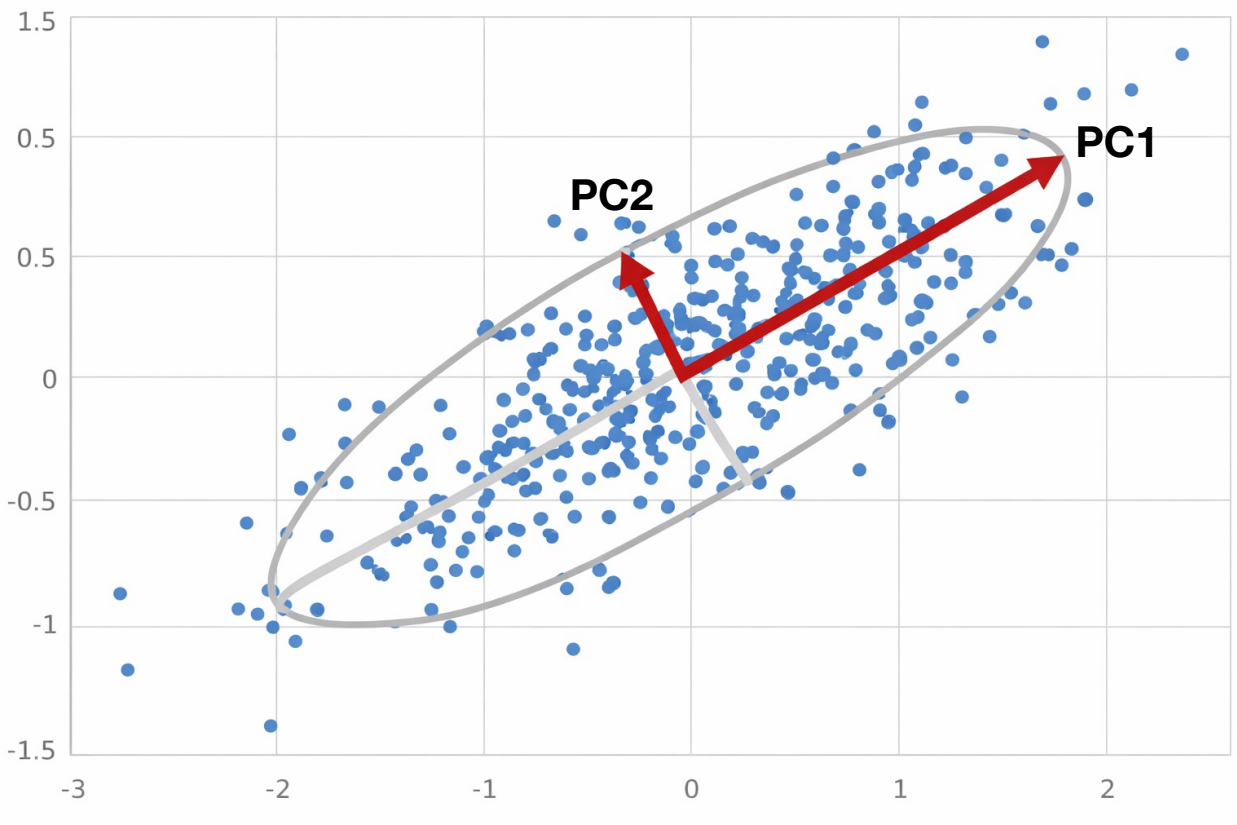
Learning Outcomes

- Explain why correlated features lead to redundancy and instability, motivating dimensionality reduction
- Describe dimensionality reduction as projecting data onto a lower-dimensional space that preserves important structure
- Interpret PCA geometrically as finding directions of maximal variance
- Formulate PCA mathematically using covariance matrices, eigenvalues, and eigenvectors
- Understand how orthogonality of principal components ensures non-redundant representations
- Apply the PCA algorithm step by step and interpret the resulting reduced features
- Recognize the importance of preprocessing (centering, normalization, differencing) for PCA
- Explain the relationship between PCA and Singular Value Decomposition (SVD)
- Identify and interpret common applications of PCA such as visualization, compression, and denoising

Motivation & Intuition

What Is PCA?

- A **dimensionality-reduction technique** that transforms correlated variables into a smaller set of uncorrelated components
- PCA identifies **directions of maximum variance**, known as "principal components," to simplify visualization, reduce noise, and improve models while preserving structure



How much variance in the data is captured by each principal component?

PCA Roadmap

Covariance Matrix Σ

How to mathematically summarize the **total shape of the data**—its spread and inter-feature correlations

The Eigen Equation $\Sigma u = \lambda u$

Why the eigenvectors of the covariance matrix are the exact directions of maximum variance in the data

Principal Components

How the new powerful summary of features in the new coordinate system are constructed from the original high-dimensional features

Example: Housing Price Prediction

To find the **most reasonably priced** apartment, you would consider square ft., distance to campus, # bedrooms, # bathrooms, …

- Some **features can be highly correlated** (e.g., # bedrooms and # bathrooms)
- Correlation = **redundant information**
- Redundancy can cause overfitting, unstable parameter estimates, poor generalization

Rent	# bedroom	# bathroom
$1,800	2	1
$1,300	1	1
$2,450	3	2
$1,950	2	1.5
⋮	⋮	⋮
?	**3**	**1.5**

If two features tell us almost the same thing, why treat them as separate pieces of information?

What's the Issue with Correlated Features?

In a predictive model:

$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n + \epsilon$$

$$X = \begin{bmatrix} x_1^1 & \cdots & x_1^m \\ \vdots & \ddots & \vdots \\ x_n^1 & \cdots & x_n^m \end{bmatrix} \in \mathbb{R}^{n \times m} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m$$

MSE:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \left(y^i - \theta^\top x^i \right)^2 = (XX^\top)^{-1} Xy$$

$XX^\top$ is **not invertible**, because at least two columns of X are linearly dependent—correlated features make it hard to **distinguish individual effects**

Mathematical Explanation of Rank-One

Data: Consider the housing price example with $x_1 \approx 1.5x_2$:

$$X^\top = \begin{bmatrix} x_1^1 & x_2^1 \\ \vdots & \vdots \\ x_1^{10} & x_2^{10} \end{bmatrix} \in \mathbb{R}^{10\times 2}$$

Due to **linear dependence of features**, X is the outer product of two vectors:

$$X^\top \approx \begin{bmatrix} x_2^1 \\ \vdots \\ x_2^{10} \end{bmatrix} c, \qquad c = [1.5 \quad 1] \qquad \Longrightarrow \qquad XX^\top \approx \left(\sum_{i=1}^{10} \left(x_2^i\right)^2 \right) cc^\top$$

Matrix X is rank-one—most variance only exists along the direction c

How to Address This?

Combining features by taking weighted averages is a natural idea:

$$\text{new feature} = 0.5 \times \#\text{ bedroom} + 0.5 \times \#\text{ bathroom}$$

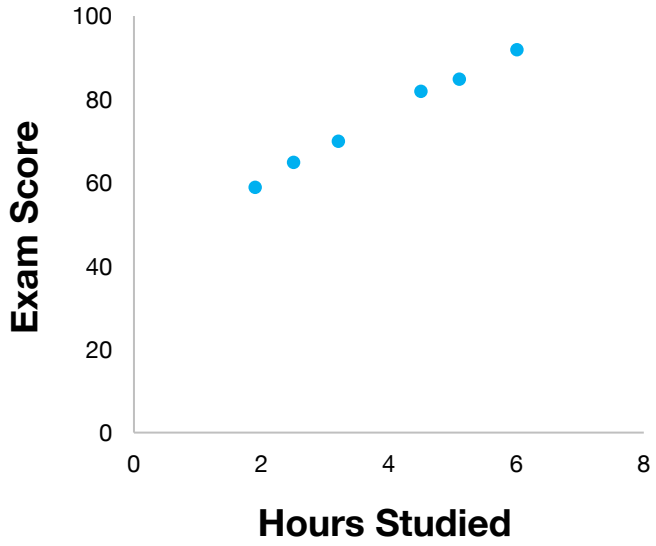
- But is it optimal?
- What features should we combine?
- How should we combine those features?
- If we can use weights, are some weights better than others?
- How do we guarantee maximized information and minimized redundancy?

PCA rigorously determines the best linear combinations—computed from data, not chosen by intuition—that capture maximum information

Another Example: Student Performance

A survey of student records shows the following relationship between two features, **hours studies** and **exam scores**

Student	Hours Studied	Exam Score
A	2.5	65
B	5.1	85
C	3.2	70
D	4.5	82
E	1.9	59
F	6.0	92

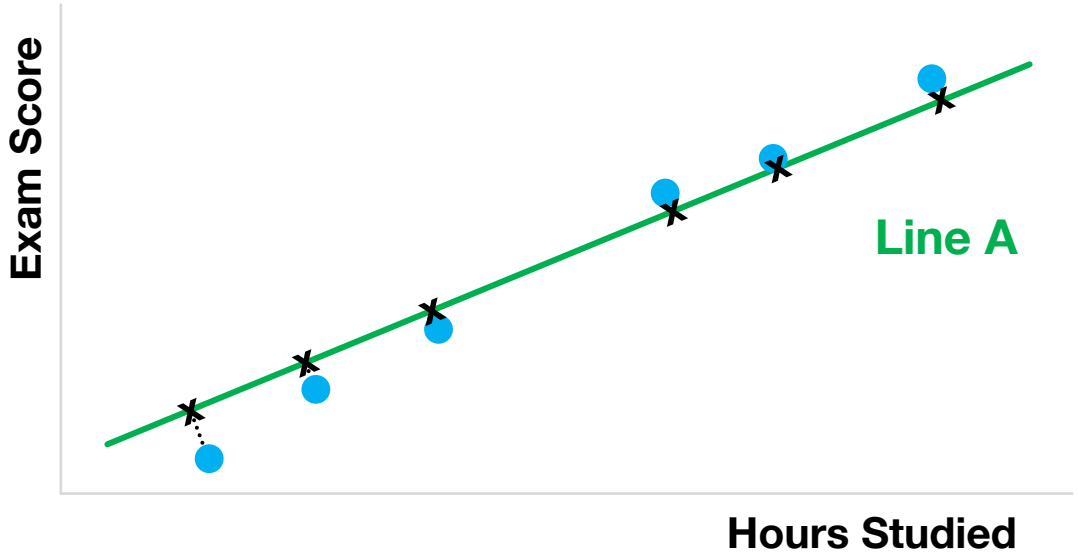


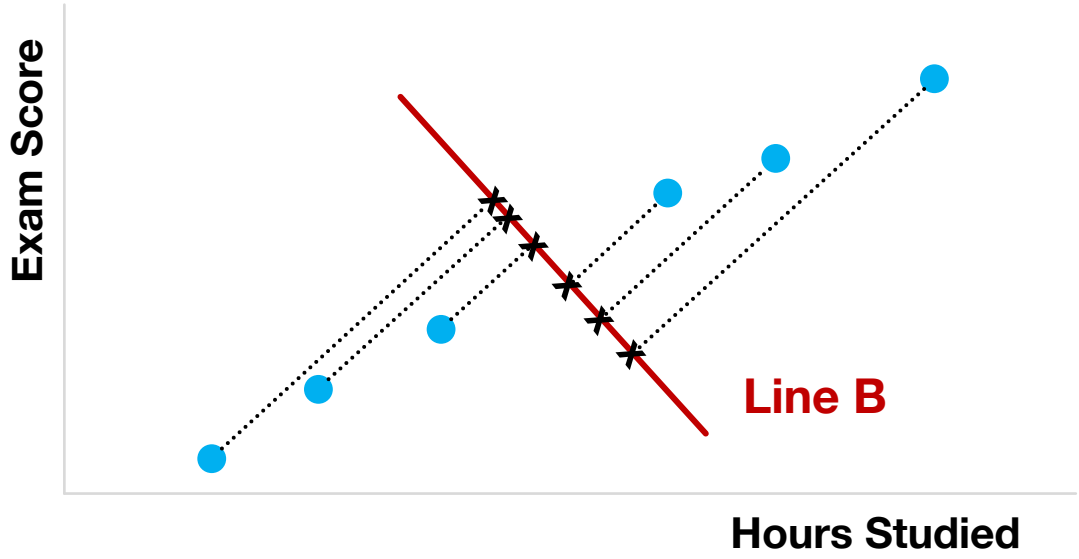
How can we best summarize this data trend using a single line?

Another Example: Student Performance (cont.)

PCA goal is to **maximize data variance**—find the line that captures the most information or spread

→ **Imagine a light source casting shadows on the data points**

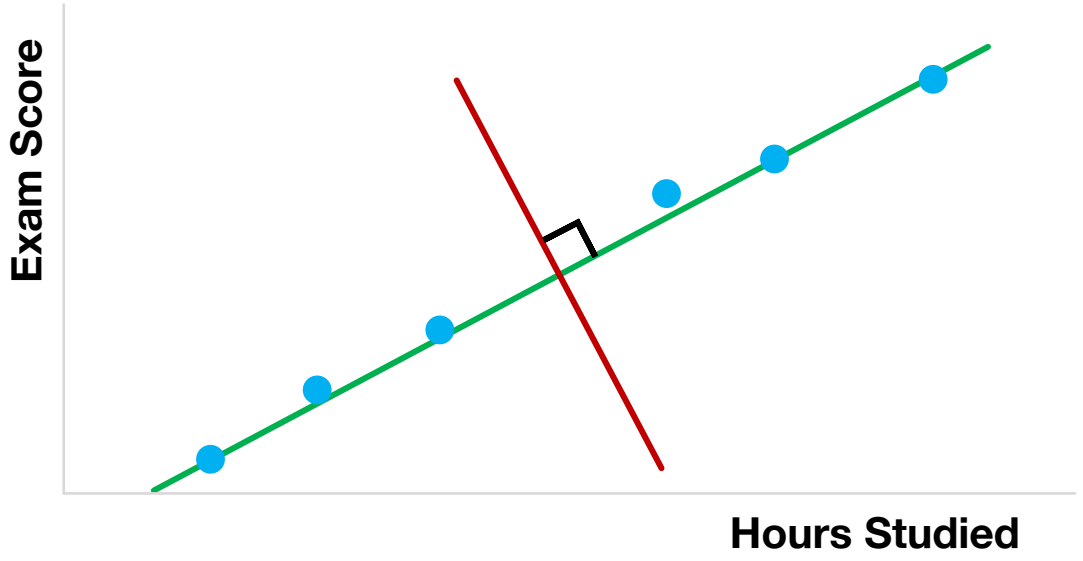




PCA rotates the line until the spread of the shadow is as wide as possible

Another Example: Student Performance (cont.)

- The line that captures the widest spread: the **first principal component**
- The line that captures the leftover variance: the **second principal component**
- And so on…

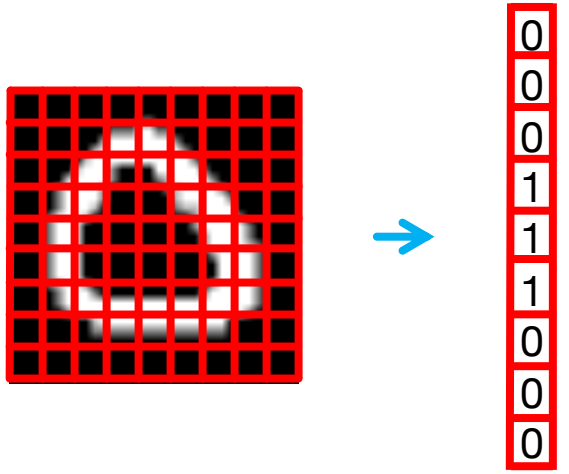


Key rule: the lines (principal components) must be orthogonal to each other

Dimensionality Reduction

What Is Dimensionality Reduction?

- Representing each point using **fewer features** without losing key information
- Reducing the **number of random variables** by combining, transforming or selecting them, or by using linear or nonlinear operations
- PCA performs this using **linear projections**
- The reduced representation is **a new coordinate system** aligned with the data's structure



Original data point

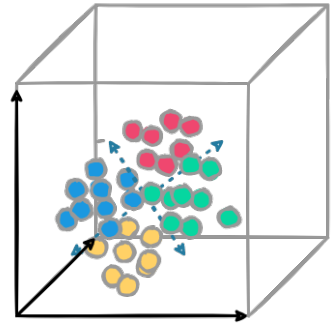
Reduced representation

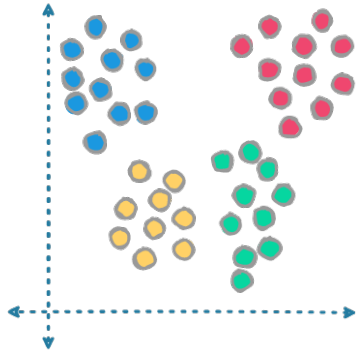
$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad f(x): \mathbb{R}^n \mapsto \mathbb{R}^k \quad z = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \end{pmatrix}$$

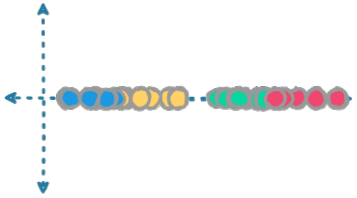
k is the number of "new transformed features" $k \ll n$

Why Dimensionality Reduction?

- Enable easier **visualization**, **exploration**, and **interpretation** of complex datasets
- Highlight meaningful **structure** in the data while reducing **noise** and **weak signals**
- Simplify the dataset by removing **redundant/correlated** features
- Speed up downstream learning tasks
- Enable simpler, more stable models that **generalize** better







What criteria best define an informative lower-dimensional space, and what are the algorithmic steps?

Linear Projection Model

Given m **mean-centered** data points $\{x^1, \ldots, x^m\}$, each with n features, suppose we combine n features into one new feature (i.e., $\mathbb{R}^n \mapsto \mathbb{R}$) using weights:

$$u = [u_1, \ldots, u_n]^\top \in \mathbb{R}^n$$

This means for each sample, the new feature called the **"score"** is given by:

$$\sum_{j=1}^{n} x_j^i \, u_j = u^\top x^i \in \mathbb{R}, \qquad i = 1, \ldots, m$$

$u^\top x^i$ is the **linear projection** of sample x^i onto direction u, so choosing u properly ensures these scores capture the essential variation

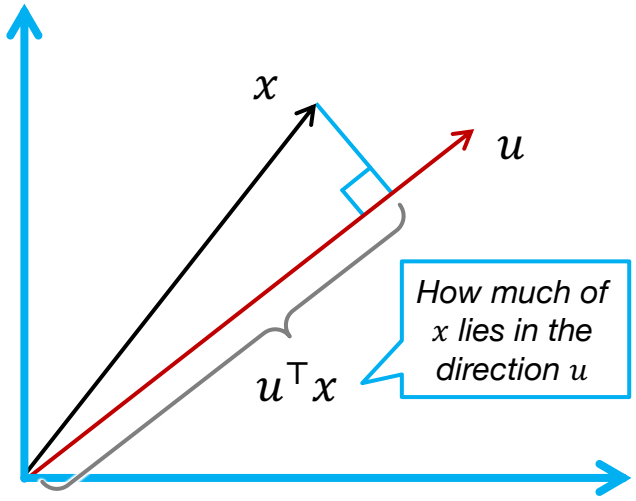
Projection in Vector Space

The score of a data point x along direction u, which separates projection into **direction** and **magnitude**, is:

$$u^\top x = \left(\frac{u}{\|u\|}\right)^\top x \cdot \|u\| = u'^\top x \cdot \|u\|$$

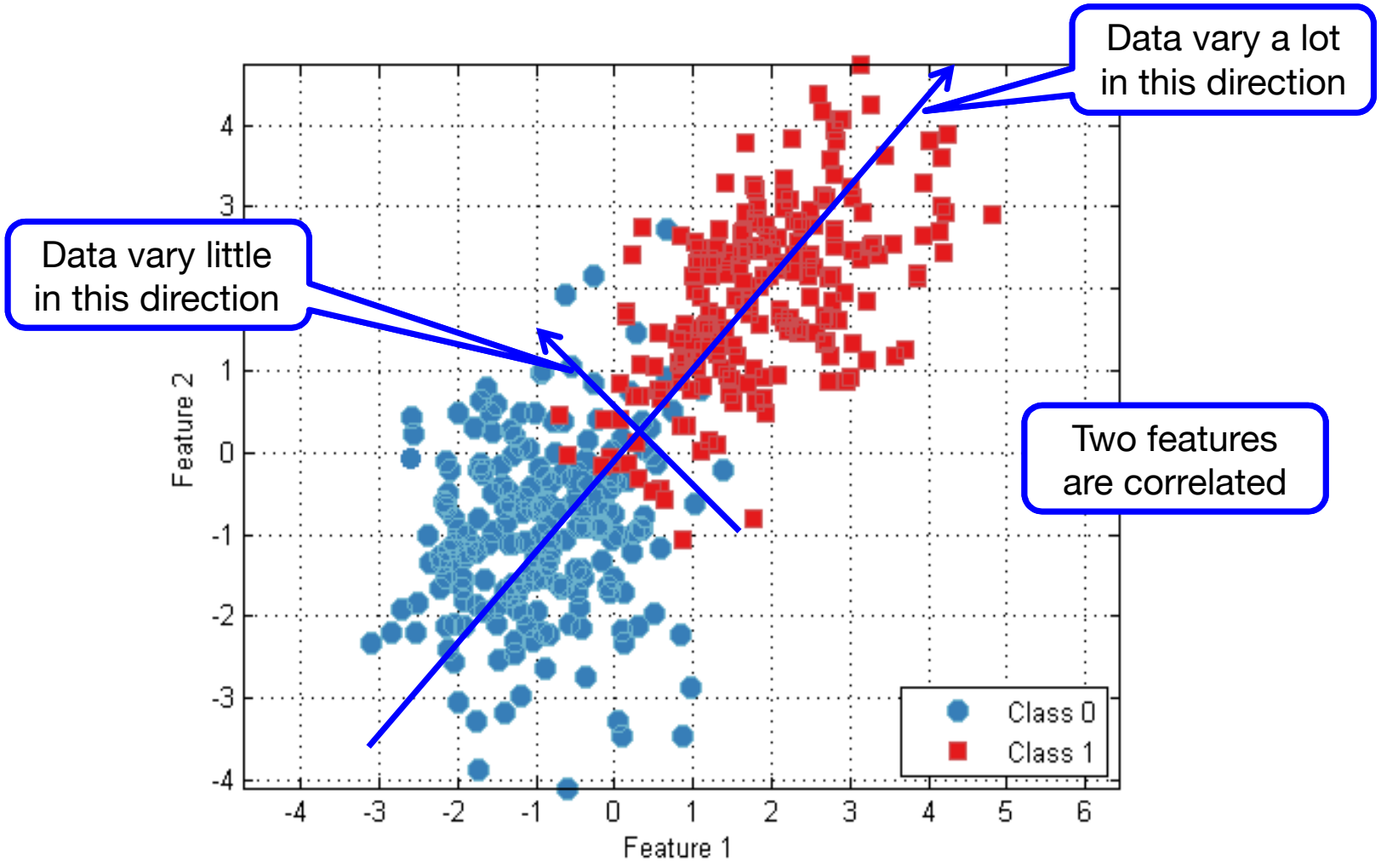
Direction Magnitude

- First, project onto the **unit direction** u', giving the component of x along that direction, i.e., $u'^\top x$
- Then, stretch by the **magnitude** $\|u\|$ to recover the full projection—it is standard in PCA to restrict to directions with $\|u\| = 1$ to focus on orientation rather than arbitrary scaling



Example

- The long arrow shows the direction that captures **most variation**—this is the first principal component
- The short arrow shows a direction with little variation—projecting onto this direction causes **information to collapse**

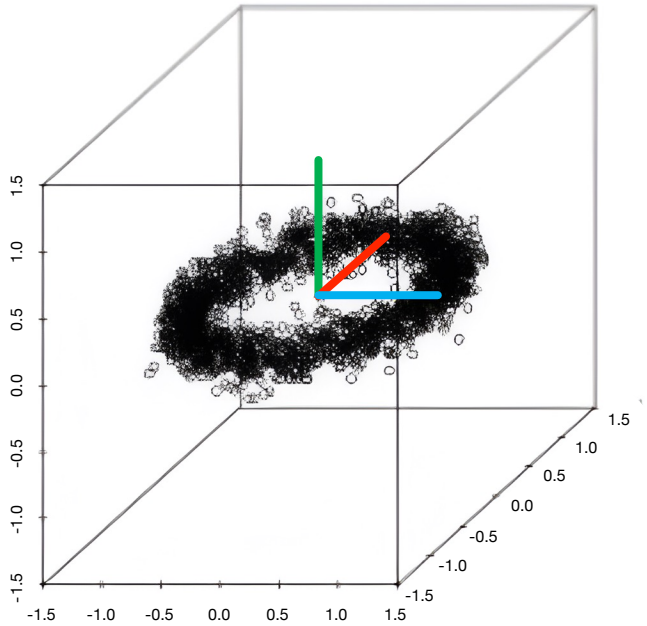


Projection Direction Matters

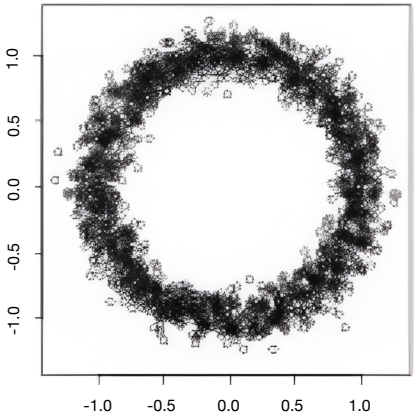
Not all projections preserve structure! How do we choose the projection direction that keeps the most meaningful differences between points?

Example

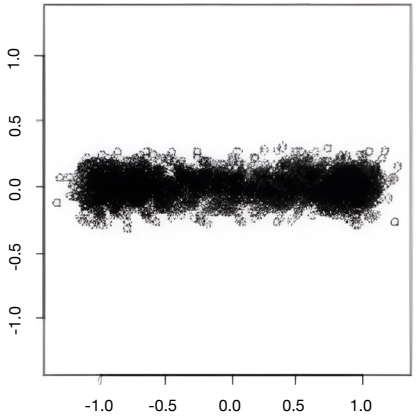
$X \in \mathbb{R}^{1000\times 3}$, and $v_1, v_2, v_3 \in \mathbb{R}^{1000\times 3}$ are unit vectors parallel to the coordinate axes



Scores onto **blue** & **red**



Scores onto **blue** & **green**



Source: R. Tibshrani

PCA selects the direction that yields maximal spread in projected data

Criterion for Reduction

- A natural measure of information is **variance**—high-variance directions capture meaningful differences, while low-variance directions often reflect **noise**
- Many criteria exist for dimensionality reduction (e.g., geometric or information-theoretic)
- Variations serve as signals in the data, but variables should be **normalized** to make their contributions comparable
- This process also reveals **highly correlated dimensions**, which can be combined into stronger, more informative signals, leading to a simpler representation
- PCA therefore chooses **orthogonal directions** that maximize variance in the projected space

PCA Mathematical Formulation

PCA Objective

Suppose the mean is:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^i, \qquad x^1, \ldots, x^m \in \mathbb{R}^n$$

Objective: Maximize the sample variance of the scores $u^\top x^i$, i.e., the variance of the projected data points relative to the projection of their mean:

$$\max_{u:\|u\|=1} \frac{1}{m}\sum_{i=1}^{m}\left(u^\top x^i - u^\top \mu\right)^2$$

Variance can be written in terms of the **covariance matrix**, which transforms the optimization into an eigenvalue problem

Optimization Derivation

Manipulate the objective with linear algebra:

$$\frac{1}{m}\sum_{i=1}^{m}\left(u^\top x^i - u^\top \mu\right)^2 = \frac{1}{m}\sum_{i=1}^{m}\left(u^\top\left(x^i - \mu\right)\right)^2 = \frac{1}{m}\sum_{i=1}^{m} u^\top\left(x^i - \mu\right)\left(x^i - \mu\right)^\top u$$

$$= u^\top\underbrace{\left(\frac{1}{m}\sum_{i=1}^{m}\left(x^i - \mu\right)\left(x^i - \mu\right)^\top\right)}_{\Sigma = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) & \cdots \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}} u = u^\top \Sigma u$$

← Covariance matrix

By imposing the constraint $\|u\| = 1$, the optimization reduces to finding the **top eigenvector of** Σ

Eigenvalue Problem: Formulation

Maximize the variance of $X^\top u$ to find the direction to project data, by solving:

$$\max_{\|u\|=1} u^\top \Sigma u$$

Using a **Lagrange multiplier**:

$$\mathcal{L}(u, \lambda) = u^\top \Sigma u - \lambda(u^\top u - 1)$$

$$\nabla_u \mathcal{L} = 2\Sigma u - 2\lambda u = 0$$

$$\Sigma u = \lambda u \quad \leftarrow \text{Eigen Equation}$$

Let's plug in an eigenvector v into the formula:

$$\text{Variance} = v^\top \Sigma v = v^\top(\lambda v) = \lambda(v^\top v) = \lambda$$

$= 1$ since $\|v\| = 1$

Eigenvalue Problem: Interpretation

Maximize the variance of $X^\top u$ to find the direction to project data, by solving:

$$\max_{\|u\|=1} u^\top \Sigma u$$

- The principal component is associated with the **largest eigenvalue** of Σ—the **direction** explaining the most variance in the data
- **Eigenvectors** define directions that remain unchanged in orientation under the linear transformation Σ—only their lengths are scaled
- In PCA, they reveal dominant patterns in data
- **Eigenvalues** quantify how much variance is captured along each eigenvector, and are ordered $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$ (larger $\lambda_k \implies$ more information)

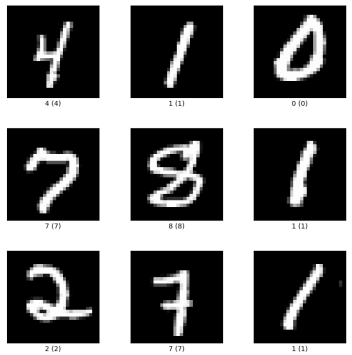
Eigenvalue Problem: Interpretation (cont.)

Given a symmetric matrix $\Sigma \in \mathbb{R}^{n \times n}$, find a vector $u \in \mathbb{R}^n$ and $\|u\| = 1$ such that:

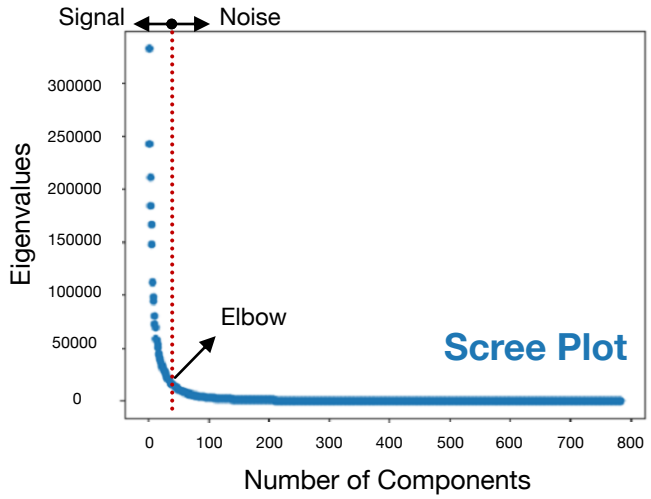
$$\Sigma u = \lambda u$$

- **Eigenvectors** are independent directions $(u^1, u^2, \ldots u^n)$ that are orthonormal $(u^{i\top} u^i = 1, u^{i\top} u^j = 0)$, ensuring that the directions are uncorrelated
- **Eigenvalues** are different $\lambda_1, \lambda_2, \ldots \lambda_n$ values (aka, spectrum) that describe how the data's total variance is distributed across directions

Example: MNIST Dataset



$n = 784$



PCA reveals that handwritten digits lie near a low-dimensional subspace despite living in a very high-dimensional pixel space

Eigendecomposition

Given a symmetric matrix $\Sigma \in \mathbb{R}^{n \times n}$, eigendecomposition:

$$\Sigma = U \Lambda U^{\top}$$

- $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n) \rightarrow$ diagonal matrix of eigenvalues, found by solving the **characteristic equation** $\det(\Sigma - \lambda I) = 0$
- $U = (u^1, \ldots, u^n) \rightarrow$ matrix of eigenvectors (columns), where each u^i is found by solving $(\Sigma - \lambda_i I)u^i = 0$

Student Scores Example Revisited:

$$X_{\text{centered}} = \begin{bmatrix} -1.37 & -10.5 \\ 1.23 & 9.5 \\ -0.67 & -5.5 \\ 0.63 & 6.5 \\ -1.97 & -16.5 \\ 2.13 & 16.5 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 2.11 & 16.92 \\ 16.92 & 136.25 \end{bmatrix}$$

$$U = \begin{bmatrix} -0.99 & 0.12 \\ 0.12 & 0.99 \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} 0.0084 & 0 \\ 0 & 138.35 \end{bmatrix}$$

PCA Algorithm

Principal Component Analysis

Step 1: Centering the Data. Given m data points $\{x^1, x^2, \ldots, x^m\} \in \mathbb{R}^n$, estimate the mean and covariance matrix from data:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^i \qquad \Sigma = \frac{1}{m}\sum_{i=1}^{m}\left(x^i - \mu\right)\left(x^i - \mu\right)^\top$$

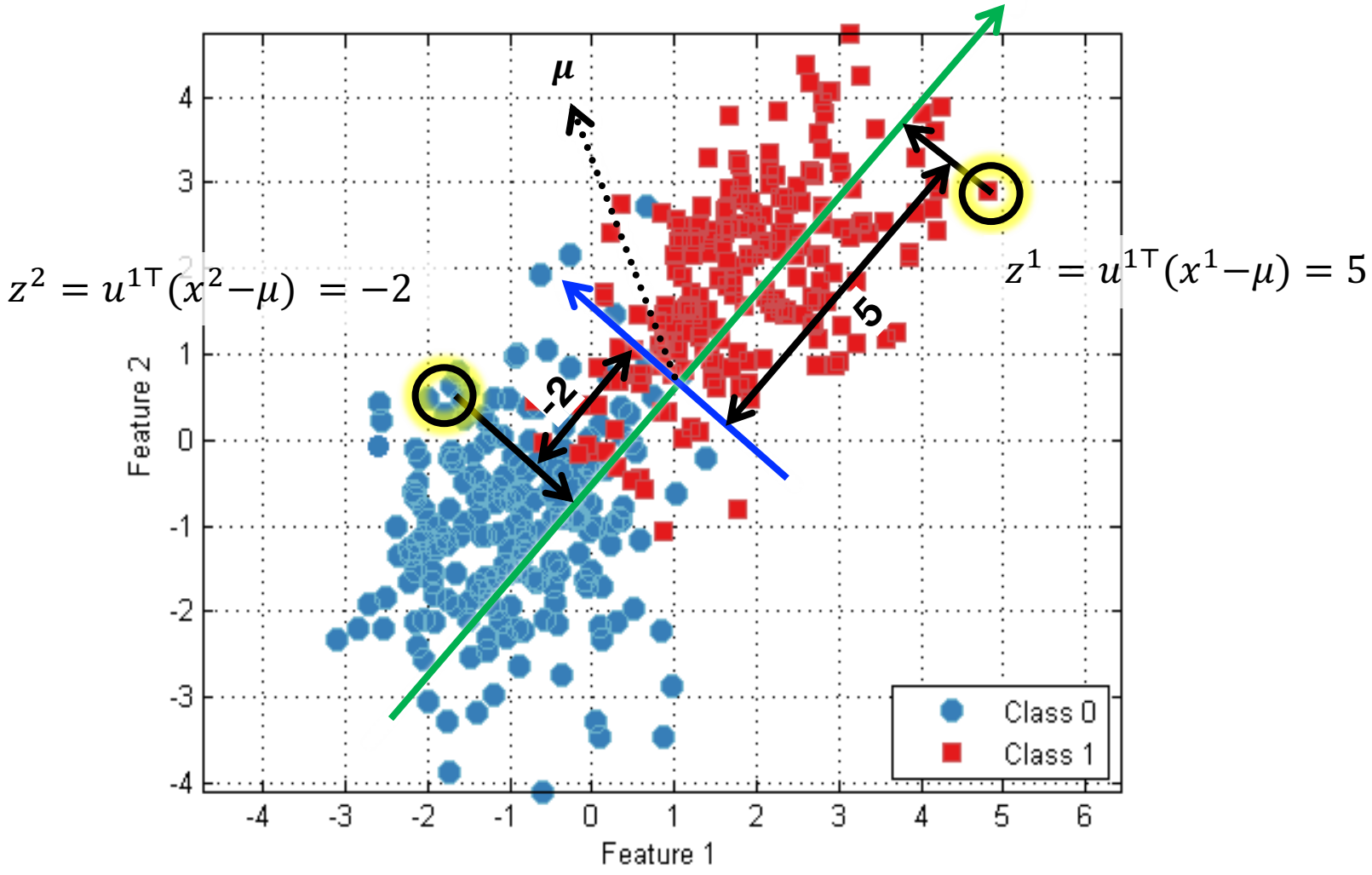
Step 2: Eigendecomposition. Take the eigenvectors $u^1, u^2, \ldots$ of Σ for the largest eigenvalue λ_1, the second largest eigenvalue λ_2, and so forth

Step 3: Projection onto Principal Components. Compute reduced representations:

$$z^i = \begin{pmatrix} u^{1\top}\left(x^i - \mu\right) \\ u^{2\top}\left(x^i - \mu\right) \\ \vdots \end{pmatrix}$$

Example: PCA Scores

The z-score represents how far a point lies along the principal axis relative to the mean, scaled by variance—larger magnitude means stronger deviation along the principal pattern



Revisiting the Housing Example

Assume the features have been standardized:

$$\mu = 0\text{: } \mu_1 = \sum_{i=1}^{m} x_1^i = 0, \;\; \mu_2 = \sum_{i=1}^{m} x_2^i = 0$$

Rent	# bedroom	# bathroom
$1,800	–0.24	–0.96
$1,300	–1.44	–0.96
$2,450	0.96	1.44
$1,950	–0.24	0.24
⋮	⋮	⋮
?	**0.96**	**0.24**

Then for $c^\top = [1.5 \quad 1]$:

$$\Sigma = XX^\top = \left(\sum_{i=1}^{10} \left(x_2^i\right)^2\right) cc^\top$$

$$= \left(\sum_{i=1}^{10} \left(x_2^i\right)^2\right) \begin{bmatrix} 1.5^2 & 1.5 \\ 1.5 & 1 \end{bmatrix}$$

$$x_1 = 1.5x_2$$

bedrooms # bathrooms

Revisiting the Housing Example (cont.)

Eigendecomposition:

$$\Sigma = \begin{bmatrix} u^1 & u^2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u^{1\top} \\ u^{2\top} \end{bmatrix} = U \Lambda U^\top$$

$$u^1 = \frac{1}{\sqrt{13}} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \qquad u^2 = \frac{1}{\sqrt{13}} \begin{bmatrix} 2 \\ -3 \end{bmatrix} \qquad \lambda_1 = \frac{13}{4} \left(\sum_{i=1}^{10} \left(x_2^i \right)^2 \right) \qquad \lambda_2 = 0$$

PC1: $z_1^i = u^{1\top} x^i = \frac{1}{\sqrt{13}} \left(3x_1^i + 2x_2^i \right)$ **PC2:** $z_2^i = u^{2\top} x^i = \frac{1}{\sqrt{13}} \left(2x_1^i - 3x_2^i \right)$

- The **first eigenvector** assigns positive weights to both bedroom and bathroom counts, confirming that the main variation comes from overall house size
- The **second eigenvector** reflects noise because the features are correlated

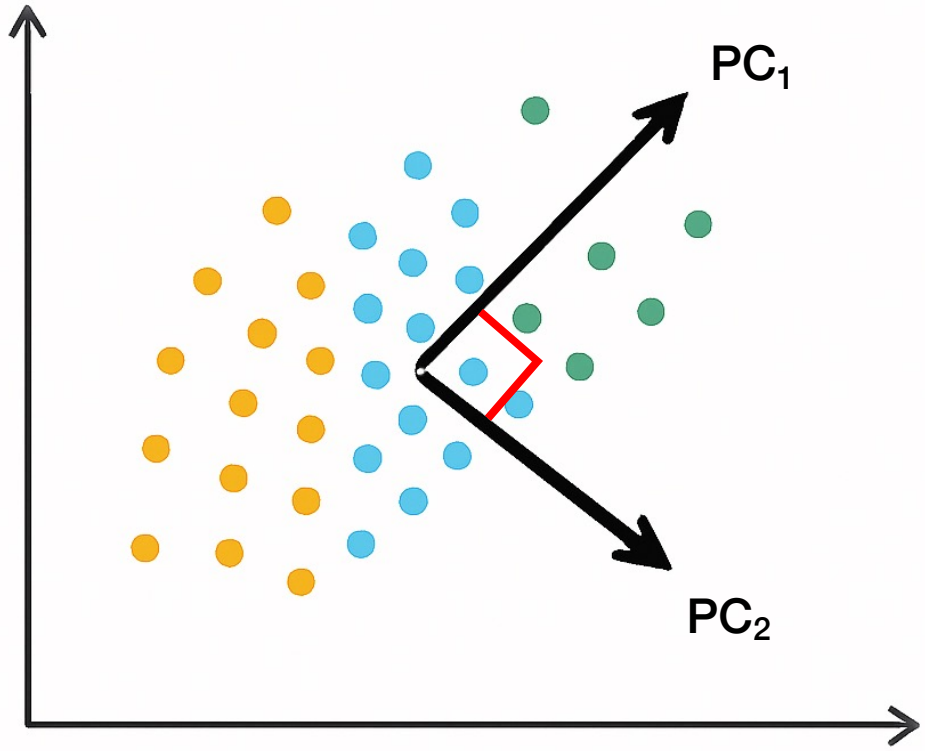
Multiple Principal Directions

After finding the first principal direction, PCA finds another **orthogonal direction** capturing remaining variance—and continues until all directions are identified

- $u^1, u^2, \ldots$ which have the largest variances orthogonal to each other: $(u^1)^\top u^2 = 0$
- Take the eigenvectors $u^1, u^2, \ldots$ of Σ corresponding to the largest eigenvalue λ_1, the second largest eigenvalue λ_2, and so on:

$$u^2 = \arg\max_{w} \; u^\top \Sigma u$$

$$\text{s.t.} \qquad \|u\| = 1, \qquad u^\top u^1 = 0$$



PCA Implementation Examples

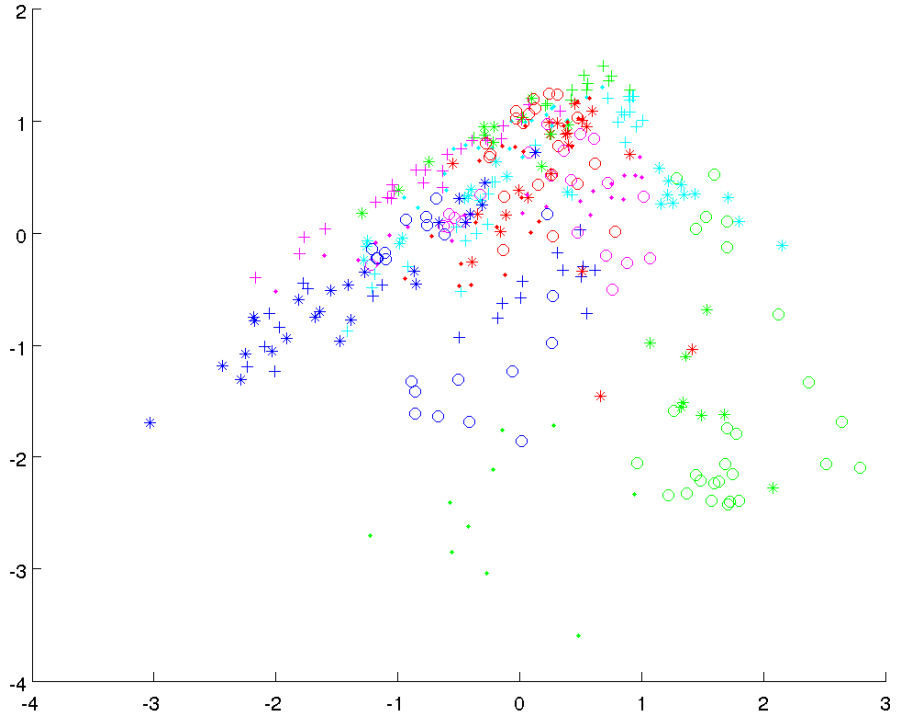
PCA on Leaf Images

Each leaf is represented by 14 features —PCA identifies which combinations of features best separate leaf types

What happens if one feature has a much larger scale than all the others?

PCA on Leaf Images (cont.)

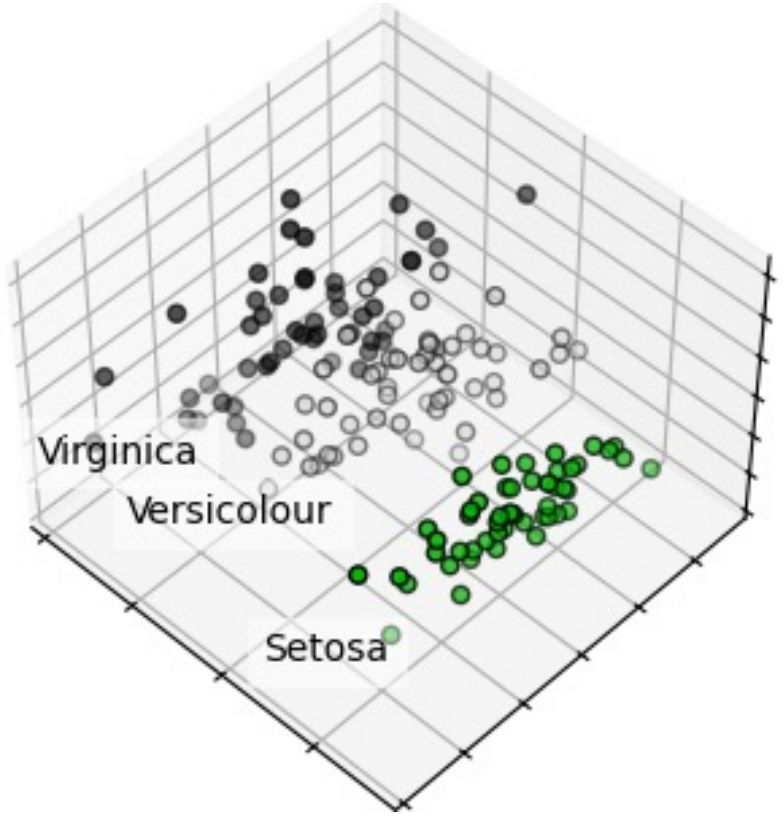
The weights in W show how each original feature contributes to each principal component—shape features dominate the first component; texture features influence the second



	u^1	u^2
	0.0938	0.1924
	0.1902	0.0253
	0.2266	-0.1800
Shape features	-0.1850	0.4084
	-0.1600	0.3825
	-0.2063	0.3488
	0.1940	-0.4037
	0.2150	-0.3566
Texture features	-0.3723	-0.2001
	-0.3657	-0.1974
	-0.3602	-0.2037
	-0.3175	-0.1886
	-0.3056	-0.1243
	-0.3482	-0.1829

PCA on Iris Images

Projecting the Iris flower dataset onto the first few principal components reveals natural cluster structures



https://scikit-learn.org/1.5/auto_examples/decomposition/plot_pca_iris.html

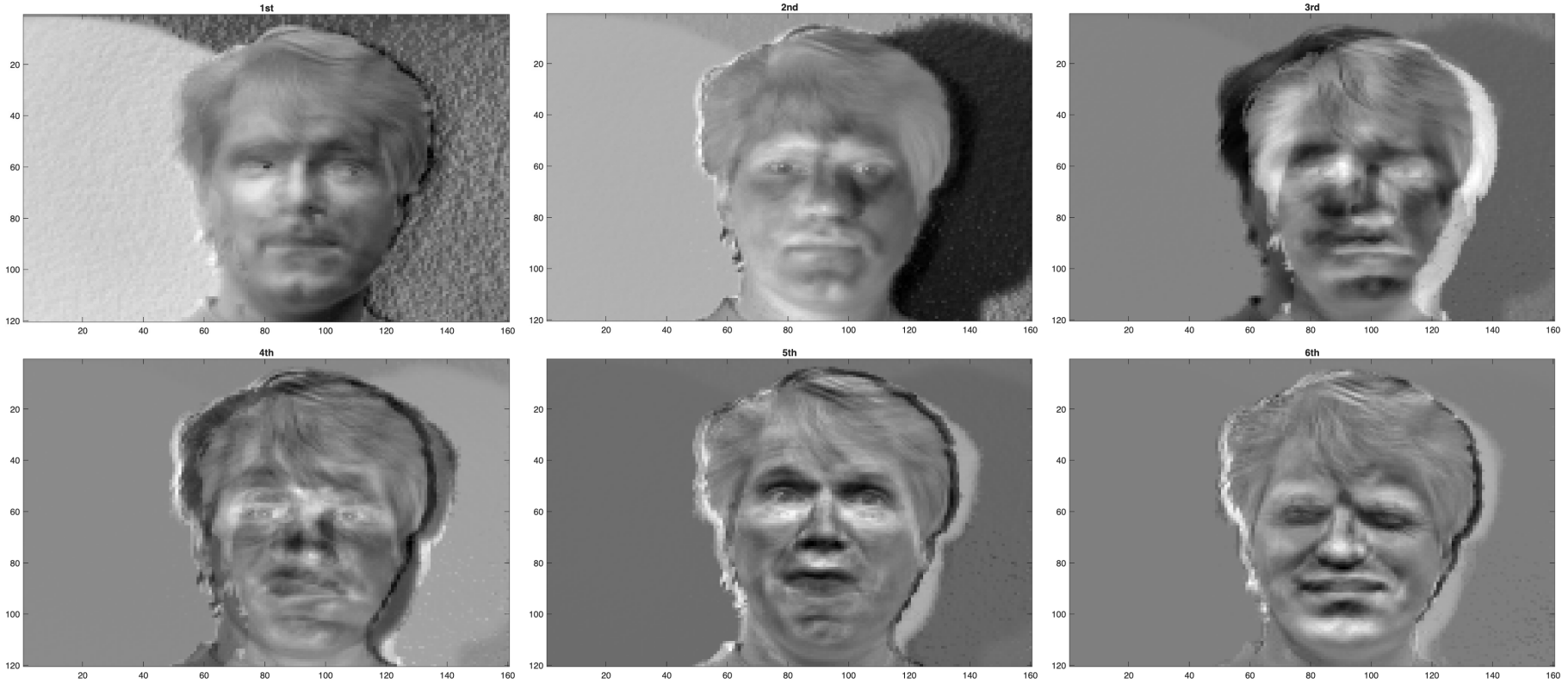
Iris Versicolor

Iris Setosa

Iris Vriginica

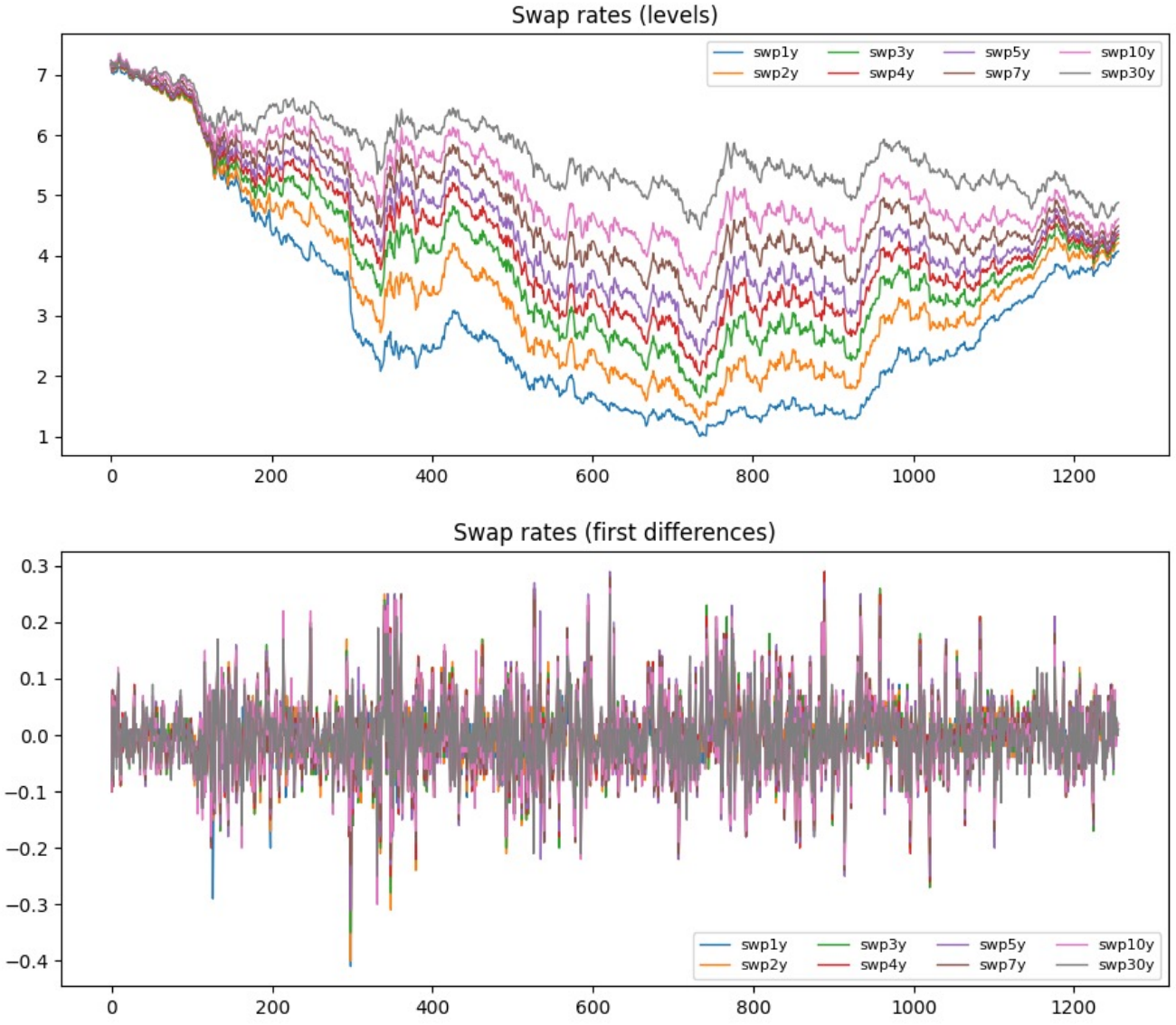
PCA on Face Images

Eigenfaces represent dominant modes of variation in facial images by capturing lighting, orientation, and general face structure



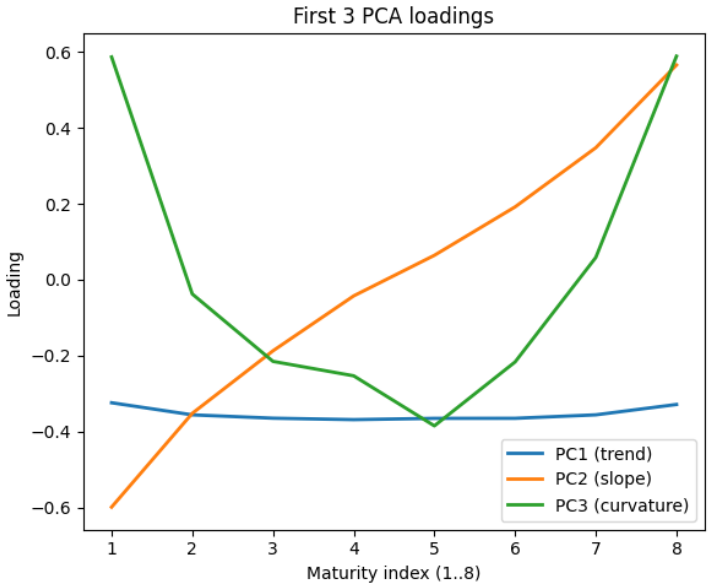
PCA for Time Series

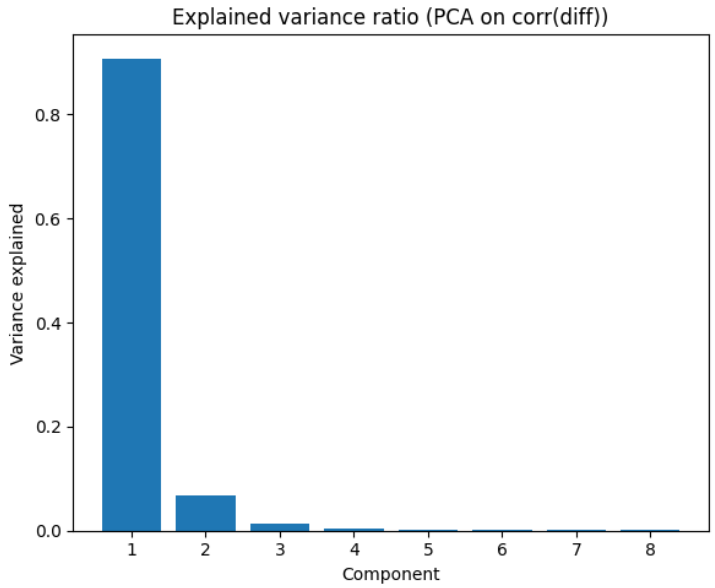
PCA of yield curves typically reveals three major components—**level**, **slope**, and **curvature**—key factors in finance

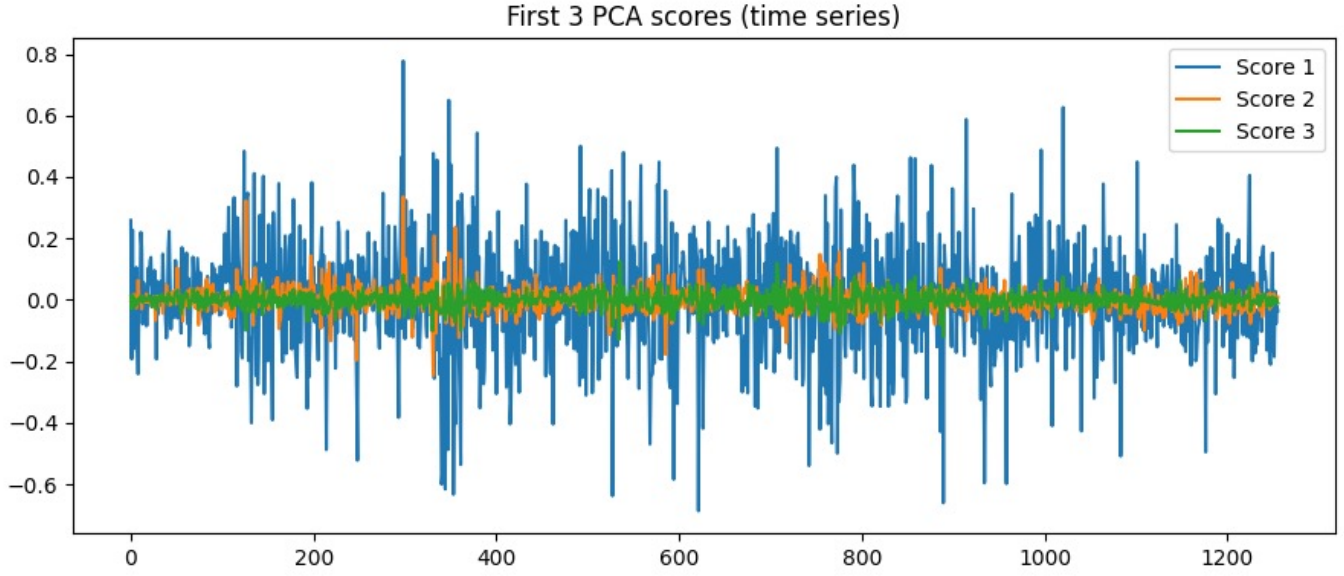


PCA for Time Series (cont.)

First-order differencing ($y_t = x_t - x_{t-1}$) removes trends and makes data stationary, enabling PCA to identify meaningful temporal components



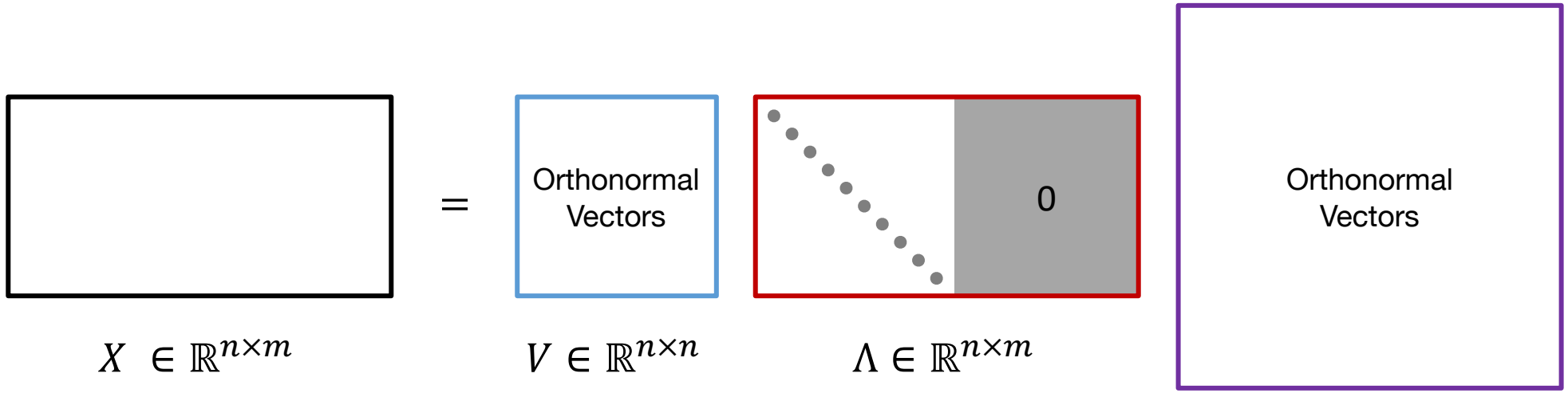




PCA & SVD

Singular Value Decomposition (SVD)

SVD is a factorization of a real matrix, providing a numerically stable and general method for PCA, especially when the covariance matrix is large or ill-conditioned



SVD decomposes a matrix into rotations, scalings, and rotations

Singular Value Decomposition (SVD)

SVD is a factorization of a real matrix, providing a numerically stable and general method for PCA, especially when the covariance matrix is large or ill-conditioned

For a matrix $X \in \mathbb{R}^{n \times m}$ $(n \leq m)$:

$$X = U\Lambda V^\top$$

$$= [u_1 \quad u_2 \quad \cdots \quad u_n] \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & \lambda_n & 0 \end{bmatrix} [v_1 \quad v_2 \quad \cdots \quad v_m]^\top$$

Left Singular Vectors

$U \in \mathbb{R}^{n \times n}$

(orthonormal: $u^\top u = I$)

Singular Values

$\Lambda \in \mathbb{R}^{n \times m}$

$(\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n)$

Right Singular Vectors

$V \in \mathbb{R}^{m \times m}$

(orthonormal: $v^\top v = I$)

Key Fact: SVD always exists—even when eigendecomposition does not

SVD & Eigendecomposition

PCA can be solved directly using SVD **without forming the covariance matrix**:

$$X = U\Lambda V^\top$$

$$\Sigma \coloneqq XX^\top = U\Lambda \underbrace{V^\top V}_{I} \Lambda^\top U^\top = U\boxed{\Lambda\Lambda^\top}U^\top$$

$$\begin{bmatrix} \lambda_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n^2 \end{bmatrix}$$

Implication for PCA

- The eigenvectors of $\Sigma \coloneqq XX^\top$ is U (the left singular vectors of X)
- The eigenvalues of Σ is λ_i^2 (squared singular values of X)

Key Takeaways

What We Learned This Week

- Correlated features introduce redundancy and can destabilize learning algorithms
- Dimensionality reduction addresses this by representing data with fewer, more informative features
- PCA identifies orthogonal directions of maximal variance that capture dominant patterns in the data
- Principal components are given by the eigenvectors of the covariance matrix, with eigenvalues measuring explained variance
- PCA can be interpreted geometrically as projection onto an optimal low-dimensional subspace
- Proper preprocessing (centering, normalization, differencing) is essential for meaningful PCA results
- PCA is closely related to SVD, which provides a stable computational approach