# Principal Component Analysis (PCA) Refresher

Geometry of Data

October 1, 2024

# Centering a Data Matrix

Data matrix $X$: $n \times d$

$n$ rows (data points)
$d$ columns (dimensions, or features)

# Centering a Data Matrix

Data matrix $X$: $n \times d$

$n$ rows (data points)
$d$ columns (dimensions, or features)

Mean of data (rows):

$$\mu = \frac{1}{n} \sum_{i=1}^{n} X_{i\bullet}$$

# Centering a Data Matrix

Data matrix $X$: $n \times d$

$n$ rows (data points)
$d$ columns (dimensions, or features)

Mean of data (rows):

$$\mu = \frac{1}{n} \sum_{i=1}^{n} X_{i\bullet}$$

Centered data (subtract mean from each row):

$$\tilde{X}_{i\bullet} = X_{i\bullet} - \mu$$

# Covariance Matrix

Sample covariance matrix:

$$\Sigma = \frac{1}{n}\tilde{X}^T\tilde{X}$$

# Covariance Matrix

Sample covariance matrix:

$$\Sigma = \frac{1}{n}\tilde{X}^T\tilde{X}$$

$\Sigma_{ij}$ is the covariance between the $i$th and $j$th dimension (feature)

$$\Sigma_{ij} = \frac{1}{n}\sum_{k=1}^{n}(X_{ki} - \mu_i)(X_{kj} - \mu_j) = \mathrm{cov}(X_{\bullet i}, X_{\bullet j})$$

# Properties

Covariance is **symmetric**: $\Sigma = \Sigma^T$

$$\Sigma_{ij} = \text{cov}(X_{\bullet i}, X_{\bullet j}) = \text{cov}(X_{\bullet j}, X_{\bullet i}) = \Sigma_{ji}$$

Covariance is **positive-semidefinite**:

$$v^T \Sigma v \geq 0$$

# Eigenvectors, Eigenvalues

Square matrix $A$: $d \times d$
Eigenvector $v \in \mathbb{R}^d$ and eigenvalue $\lambda \in \mathbb{R}$:

$$Av = \lambda v$$

# Eigenvectors, Eigenvalues

Square matrix $A$: $d \times d$
Eigenvector $v \in \mathbb{R}^d$ and eigenvalue $\lambda \in \mathbb{R}$:

$$Av = \lambda v$$

**Meaning:** The transformation $A$ is a scaling when applied to $v$

# Eigenanalysis of a Symmetric Matrix

**Fact:** If $A$ is a $d \times d$ symmetric matrix, it has *exactly* $d$ real eigenvalues $\lambda_k \in \mathbb{R}$ (possibly with repeats).

Each eigenvalue $\lambda_k$ has a corresponding eigenvector $v_k \in \mathbb{R}^d$.

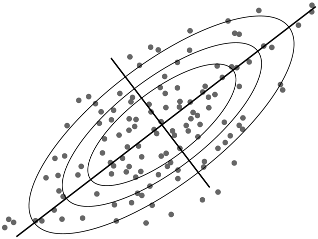
# Eigenanalysis of a Symmetric Matrix

The SVD of a symmetric matrix looks like this:

$$A = VSV^T$$

- The singular values are the eigenvalues: $s_k = \lambda_k$.
- The left and right singular vectors are the *same* and are the eigenvectors, $v_k$.

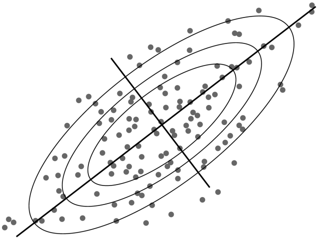
# Principal Component Analysis



PCA is an eigenanalysis of the covariance matrix:

$$\Sigma = V \Lambda V^T$$

# Principal Component Analysis



PCA is an eigenanalysis of the covariance matrix:

$$\Sigma = V \Lambda V^T$$

- Eigenvectors: $v_k = V_{\bullet k}$ are **principal components**
- Eigenvalues: $\lambda_k$ are the **variance** of the data in the $v_k$ direction

# PCA Algorithm Summary

**Input:** Data matrix $X$: $n \times d$

1. Compute centered data $\tilde{X}$
2. Compute covariance matrix:

$$\Sigma = \frac{1}{n}\tilde{X}^T\tilde{X}$$

3. Eigenanalysis of covariance:

$$\Sigma = V\Lambda V^T$$

# PCA Algorithm Summary

**Input:** Data matrix $X$: $n \times d$

1. Compute centered data $\tilde{X}$
2. Compute covariance matrix:

$$\Sigma = \frac{1}{n}\tilde{X}^T\tilde{X}$$

3. Eigenanalysis of covariance:

$$\Sigma = V\Lambda V^T$$

**Hint:** `numpy.linalg.eig` computes an eigenanalysis!

# Dimensionality Reduction

**Goal:** Find a $k$-dimensional subspace, $V_k$, that best fits our data

# Dimensionality Reduction

**Goal:** Find a $k$-dimensional subspace, $V_k$, that best fits our data

Least-squares fit:

$$\arg\min_{V_k} \sum_{i=1}^{n} \mathrm{distance}(V_k, x_i)^2$$

# Dimensionality Reduction

**Goal:** Find a $k$-dimensional subspace, $V_k$, that best fits our data

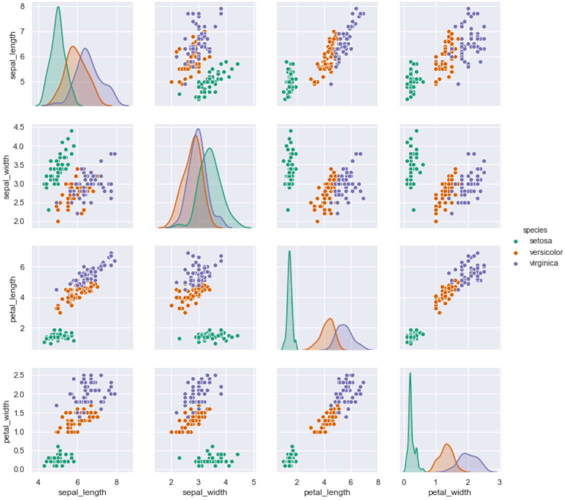
Least-squares fit:

$$\arg\min_{V_k} \sum_{i=1}^{n} \text{distance}(V_k, x_i)^2$$

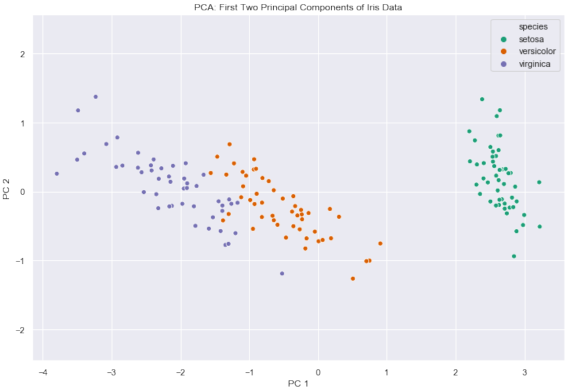
**Solution:** Use first $k$ principal components:

$$V_k = \text{span}(v_1, v_2, \ldots, v_k)$$

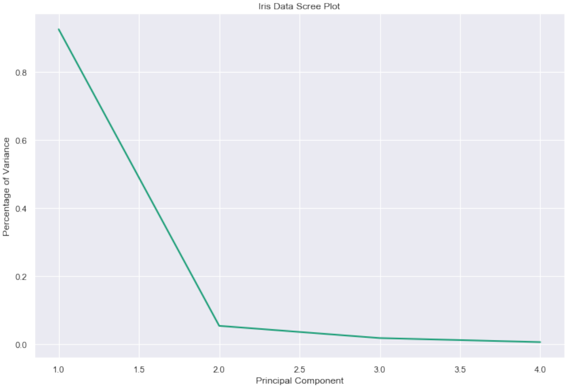
# Example: Iris Data



# Example: Iris Data PCA



# Scree Plot: Eigenvalues (Variance)




Horizontal axis: index $k$

Vertical axis: proportion of variance: $\frac{\lambda_k}{\sum_{j=1}^{d} \lambda}$