Optimization Techniques for Machine Learning

AMLZC326 · #12 PCA II

Anshid Aboobacker

Motivation & Perspectives on PCA

- In Lecture 11, we saw that PCA reduces to finding eigenvectors of the covariance matrix.
- But what happens when the data dimension D is very large (say $D = 10{,}000$)?
  - Can we still compute eigenvectors directly?
- PCA can be understood in three equivalent ways:
  - **Variance maximization:** directions of maximum variance
  - **Eigenvector view:** eigenvectors of covariance matrix
  - **SVD view:** low-rank approximation of data
- In this lecture, we focus on:
  - Efficient computation (Power Method)
  - SVD as a computational tool
  - PCA in high dimensions

Learning Objectives

By the end of this lecture you should be able to:

- Connect PCA to SVD and derive principal components from the right singular vectors of the data matrix
- Apply the Eckart-Young theorem: the best rank-M approximation uses the top M singular values/vectors
- Use the power method to efficiently compute the dominant eigenvector of a matrix
- Implement PCA end-to-end: centring the data, computing eigenvectors, projecting, and reconstructing

Revision of PCA Setup

- PCA constructs a lower-dimensional representation using a matrix **B**.
- The covariance matrix **S** determines **B**.
- Key relationships:

$$\mathbf{z} = \mathbf{B}^T\mathbf{x}, \quad \tilde{\mathbf{x}} = \mathbf{B}\mathbf{z}$$

- $\mathbf{z}$: low-dimensional representation
  $\tilde{\mathbf{x}}$: reconstruction in original space

Table of Contents

Eigenvector Computation and SVD Connection

- In PCA, the principal subspace is spanned by eigenvectors of the covariance matrix: $S = \frac{1}{N}\sum_{i=1}^{N} x_i x_i^T = \frac{1}{N} X X^T$ where $X = [x_1, \ldots, x_N] \in \mathbb{R}^{D \times N}$
- To compute eigenvectors of S, we have two approaches:
  - Direct eigen-decomposition of S
  - Singular Value Decomposition (SVD) of X
- Let $X = U \Sigma V^T$ (SVD). Then: $S = \frac{1}{N} X X^T = \frac{1}{N} U \Sigma \Sigma^T U^T$
- **Key insight:**
  - Columns of U are eigenvectors of S
  - Eigenvalues: $\lambda_d = \frac{\sigma_d^2}{N}$

Example: Understanding the Data

Consider the dataset (2 features, 9 samples):

Feature 1	Feature 2
1.11	10
1.21	12
1.36	13
1.49	15
1.63	16
1.68	17
1.83	18
1.88	19
1.95	20

- Each column = a point in $\mathbb{R}^2$
- Matrix form: $X \in \mathbb{R}^{2\times 9}$
- Goal: find main direction

Key observation:

- Feature 2 $\approx$ linear in Feature 1
- Data lies near a line in $\mathbb{R}^2$

Example: What SVD Reveals

We compute the SVD: $X = U\Sigma V^T$

$$U = \begin{bmatrix} 0.0883 & 0.9961 \\ 0.9961 & -0.0883 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 9.535 & 0 & \cdots \\ 0 & 0.084 & \cdots \end{bmatrix}$$

- Columns of U = principal directions
  - First column: direction of maximum variance
  - Second column: orthogonal residual direction
- Singular values (Σ): $9.535 \gg 0.084$
  - Almost all variation lies in one direction

Conclusion:

- Data is approximately **1-dimensional**
- PCA will project onto the first column of U

Why SVD is Useful for PCA

- Instead of computing eigenvectors of $S = \frac{1}{N} XX^T$, we compute the SVD of X: $X = U\Sigma V^T$
- **Interpretation:**
  - Singular values determine variance along principal directions
  - Connects variance maximization view with SVD
- **Why this matters:**
  - More efficient and numerically stable
  - Avoids explicit covariance computation
  - **PCA via covariance = PCA via SVD**

Table of Contents

Best Rank-M Approximation

- In PCA, we seek a low-dimensional representation that captures maximum variance.
- This is equivalent to finding the best rank-M approximation of X: $\tilde{X}_M = \underset{\text{rank}(A) \leq M}{\arg\min} \|X - A\|_F^2$
- **Eckart–Young Theorem:**
  - Let $X = U\Sigma V^T$ (SVD)
  - The best rank-M approximation is: $\tilde{X}_M = U_M \Sigma_M V_M^T$
- **Interpretation:**
  - Keep the top-M singular values and corresponding directions
  - This minimizes reconstruction error in Frobenius norm
  - PCA = optimal low-rank approximation of data

Practical Aspects of PCA

- Computing eigenvalues and eigenvectors is central to PCA and many ML methods.
- **Theoretical limitation:**
  - Eigenvalues are roots of the characteristic polynomial
  - For matrices of size > 4, no general algebraic solution exists
- **Practical approach:**
  - Use iterative numerical methods to compute eigenvalues/ singular values
  - Implemented in all modern linear algebra libraries
- **Key observation:**
  - In PCA, we only need a few top eigenvectors
  - Computing full decomposition is wasteful
- **Implication:**
  - Targeting leading eigenvectors improves efficiency.

Power Method for Eigenvector Computation

- Direct computation is expensive for large matrices.
- The **Power Method** is a simple iterative approach.
- Start with a random vector x_0:

$$x_{k+1} = \frac{Sx_k}{\|Sx_k\|}$$

- Repeated multiplication amplifies the dominant direction.
- Converges to the eigenvector corresponding to the largest eigenvalue.

PCA in High Dimensions: Problem and Idea

- In D dimensions, the covariance matrix $S \in \mathbb{R}^{D \times D}$.
- Eigen-decomposition scales as $\mathcal{O}(D^3) \rightarrow$ infeasible when D is large.
- This is common in practice: $D \gg N$ (high dimension, few samples).
- **Key idea:**
  - Instead of working with $S = \frac{1}{N} X X^T \in \mathbb{R}^{D \times D}$
  - Work with $X^T X \in \mathbb{R}^{N \times N}$
- This reduces computation from $D \times D$ to $N \times N$ when $N \ll D$.

PCA in High Dimensions: Computational Trick

- Assume centered data $X \in \mathbb{R}^{D \times N}$, with $S = \frac{1}{N} X X^T$
- Eigenvalue equation: $S b_m = \lambda_m b_m$
- Substitute: $\frac{1}{N} X X^T b_m = \lambda_m b_m$
- Multiply by X^T and define $c_m = X^T b_m$: $\frac{1}{N} X^T X c_m = \lambda_m c_m$
- **Key result:**
  - Solve eigenproblem for $X^T X$ ($N \times N$)
  - Recover eigenvectors of S via: $b_m = X c_m$

Choosing Number of Components

- How many principal components should we keep?
- Use explained variance:

$$\text{Explained Variance Ratio} = \frac{\lambda_i}{\sum_j \lambda_j}$$

- Common approach:
  - Keep components that explain 90%–95% variance
- Scree plot helps visualize this trade-off.

Table of Contents

PCA Pipeline (Overview)

- Goal: Project data to a lower-dimensional subspace.
- Example: 2D data $\rightarrow$ 1D projection
- Steps:
  1. Preprocessing (centering & scaling)
  2. Compute covariance and eigenvectors
  3. Select principal components
  4. Project data

PCA Step 1: Preprocessing

- **Center the data:**

$$x^{(d)} \leftarrow x^{(d)} - \mu_d$$

- **Standardize (optional but common):**

$$x^{(d)} \leftarrow \frac{x^{(d)}}{\sigma_d}$$

- **Why?**
  - Removes bias due to mean
  - Ensures comparable scaling across features

PCA Step 2: Covariance and Eigenvectors

- Compute covariance matrix:

$$S = \frac{1}{N} X X^T$$

- Compute eigenvalues and eigenvectors of S
- **Interpretation:**
  - Eigenvectors: principal directions
  - Eigenvalues: variance along those directions
- Select top-M eigenvectors (largest eigenvalues)

PCA Step 3: Projection

- Let B contain top-M eigenvectors.
- Project a data point x_*:

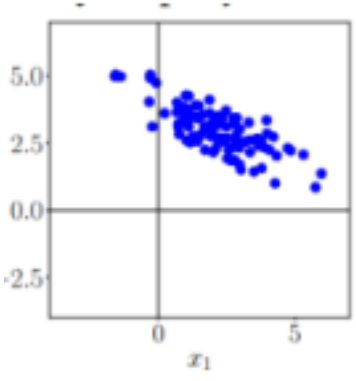
$$z_* = B^T x_*$$

- Reconstruction:

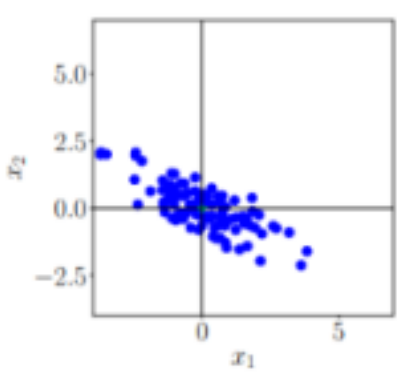
$$\tilde{x} = BB^T x_*$$

- **Interpretation:**
  - z_*: low-dimensional representation
  - $\tilde{x}$: approximation in original space

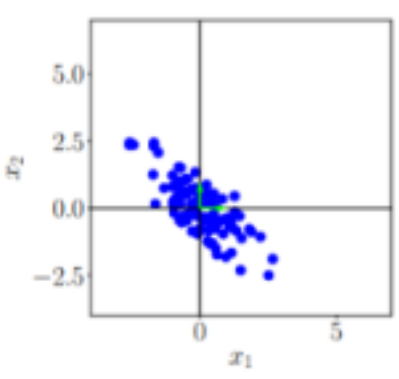
PCA in Practice



(a) Original dataset.



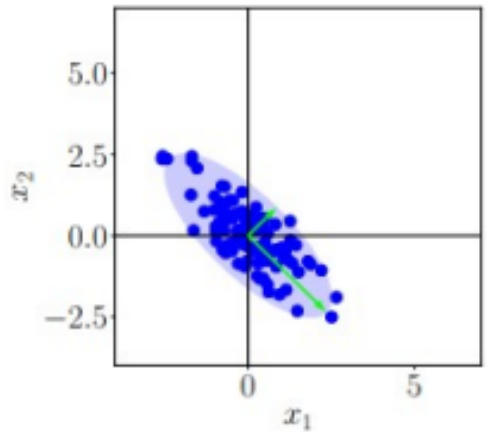
(b) Step 1: Centering by subtracting the mean from each data point.



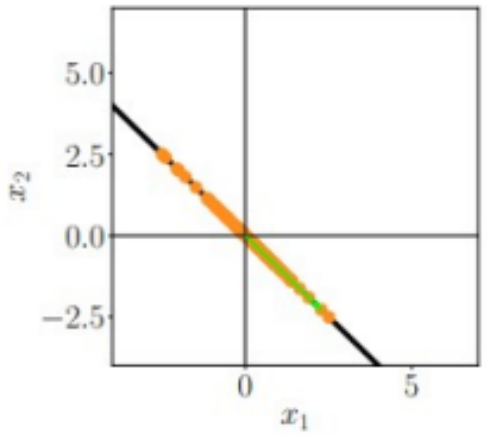
(c) Step 2: Dividing by the standard deviation to make the data unit free. Data has variance 1 along each axis.

Figure 10.11 Steps of PCA. (a) Original dataset; (b) centering; (c) divide by standard deviation; (d) eigendecomposition; (e) projection; (f) mapping back to original data space.

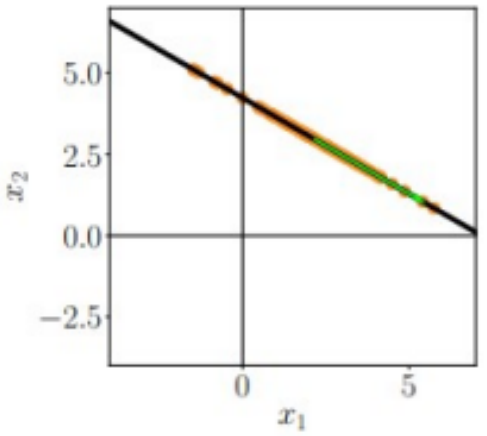
PCA in Practice



(d) Step 3: Compute eigenvalues and eigenvectors (arrows) of the data covariance matrix (ellipse).



(e) Step 4: Project data onto the principal subspace.



(f) Undo the standardization and move projected data back into the original data space from (a).

Summary

- PCA can be viewed in multiple ways:
  - Variance maximization
  - Eigenvector computation
  - Low-rank approximation via SVD
- In practice:
  - Use SVD instead of direct eigendecomposition
  - Use iterative methods like Power Method
- PCA finds the most informative directions in data while enabling efficient computation.

Thank you :)