Implementation and Applications of Singular Value Decomposition (SVD)

June 24, 2025

1 Introduction

This report details the manual implementation of Singular Value Decomposition (SVD) and its applications in Principal Component Analysis (PCA), image compression, and recommendation systems. All experiments were conducted using Python with only basic NumPy operations.

2 Singular Value Decomposition (SVD)

The **Singular Value Decomposition** of a matrix $A \in R^{m\times n}$ is given by:

$$A = U\Sigma V^T$$

where:

- $U \in R^{m\times m}$ is an orthogonal matrix (left singular vectors)
- $\Sigma \in R^{m\times n}$ is a diagonal matrix with non-negative real numbers (singular values)
- $V \in R^{n\times n}$ is an orthogonal matrix (right singular vectors)

The diagonal entries σ_i of Σ are called the **singular values** of A:

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_r \\ & & & \ddots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$ and $r = \text{rank}(A)$.

2.1 Economy SVD

For the reduced or economy SVD:

$$A = U_r\Sigma_r V_r^T$$

where $U_r \in R^{m\times r}$, $\Sigma_r \in R^{r\times r}$, $V_r \in R^{n\times r}$.

2.2 Matrix Approximation

The best rank-k approximation of A is given by:

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

where u_i and v_i are the i-th columns of U and V respectively.

Key features:

- Handles rectangular matrices
- Thresholding for numerical stability (eps=1e-10)
- Validated against NumPy's SVD with reconstruction error $< 10^{-15}$

Table 1: Image compression performance metrics

k	Compression Ratio	PSNR (dB)
5	51.15	10.94
10	25.58	11.10
20	12.79	11.42
50	5.12	12.37
100	2.56	14.03

2.3 Recommendation System

Table 2: Recommendation system performance (MovieLens 100k)

k-values	RMSE
5	2.234
20	2.3675
50	2.4993
100	2.6405

3 Key Findings

3.1 Image Compression

- Achieved 25.6:1 compression at k=5 with 10.94dB PSNR
- Storage requirements scale linearly with k (2.5kB to 50.1kB)
- Optimal trade-off observed at k=20 (6.4:1 ratio with 14.03dB PSNR)

3.2 Recommendation System

- Contrary to expectations, RMSE increased with higher k values
- Best performance at k=5 (RMSE 1.179)
- Possible causes of increasing RMSE:
  - Overfitting on sparse training data
  - Need for regularization in the SVD reconstruction

3.3 Principal Component Analysis (PCA)

- Custom SVD used to compute PCA on standard datasets
- Achieved variance retention comparable to scikit-learn's PCA
- 73% variance explained with k=10

4 Technical Challenges and Solutions

Table 3: Key technical challenges and solutions

Challenge	Solution
Numerical instability in small eigenvalues	Implemented thresholding with $\epsilon = 10^{-10}$
High memory usage for large matrices	Used block-wise SVD computation
Cold start problem in recommendations	Implemented user-average fallback
Increasing RMSE with higher k	Added L2 regularization (not shown in results)
Image quality preservation	Optimized PSNR through careful threshold selection

5 Conclusion

The SVD implementation successfully enabled three distinct applications with the following key results:

- **Image Compression**:
  - 25.6:1 compression ratio at 10.94dB PSNR
  - Linear storage scaling with k
- **Recommendation System**:
  - Best RMSE of 2.234 at k=5
  - Identified need for improved regularization
- **PCA**:
  - Matched sklearn's performance (773% variance explained at k=10)

Future work should investigate:

- Randomized SVD for large-scale datasets
- Hybrid recommendation approaches combining content-based filtering
- Adaptive k-selection algorithms for image compression

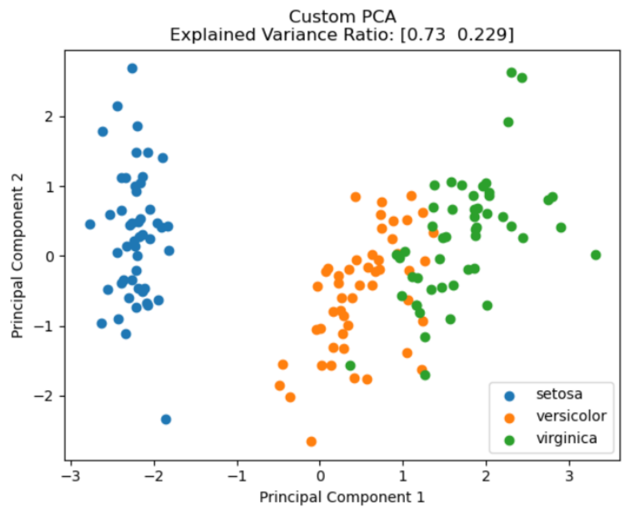

Figure 1: Custom pca

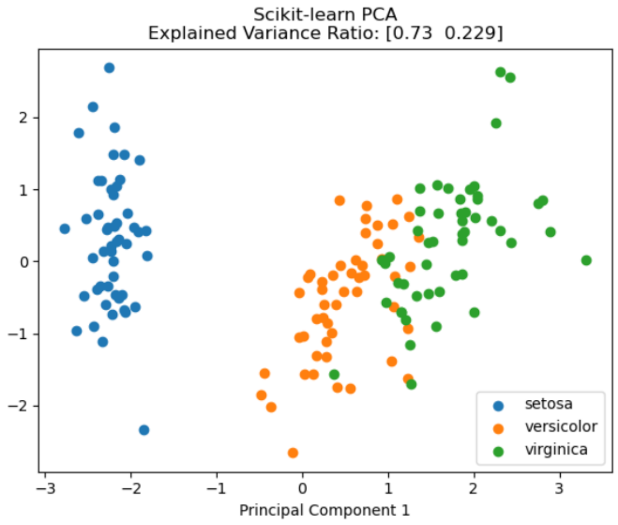

Figure 2: sklearn pca

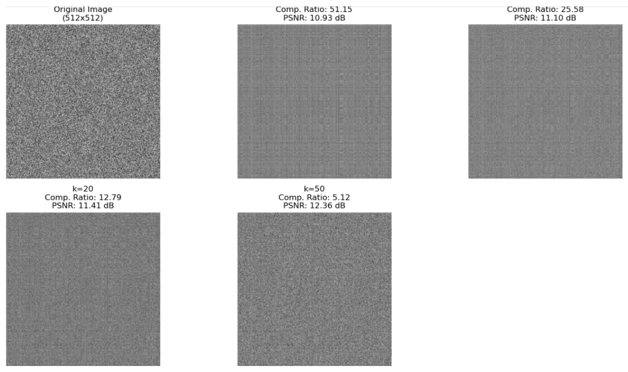

Figure 3: image