Image Compression with Using Truncated Singular Value Decomposition

İlke Başak Baydar - 150140709

BLG 202E, Spring 2025, Term Project

1 Introduction

In this project, it is aimed Truncated Singular Value Decomposition (TSVD) implementation from scratch, applied to two images and analyze the effectiveness of this method in image compression by examining the storage requirements and the quality of reconstructed images at different levels.

2 Definitions and Techniques

Singular Value Decomposition (SVD) is a mathematical technique used in linear algebra to factorize a matrix into three components. For any real or complex matrix A of size m × n, SVD can be written as:

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

where:

- U is an $m \times m$ orthogonal matrix (left singular vectors),
- Σ is an $m \times n$ diagonal matrix containing the singular values of A,
- V^T is the transpose of an $n \times n$ orthogonal matrix (right singular vectors).

Truncated SVD is an approximation of the full SVD where only the top k singular values and their corresponding vectors are retained. This results in a reduced-rank approximation of the original matrix A:

$$A_k = U_k \Sigma_k V_k^T$$

where:

- U_k is the matrix with the first k columns of U,

- Σ_k is a $k \times k$ diagonal matrix with the top k singular values,
- V_k^T is the transpose of the first k columns of V.

Given an input matrix $A \in R^{m \times n}$ and a desired rank k such that $k < \min(m, n)$, the Truncated SVD algorithm proceeds as follows:

1. Compute the full Singular Value Decomposition (SVD) of A:

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

   where $U \in R^{m \times m}$, $\Sigma \in R^{m \times n}$, and $V \in R^{n \times n}$.

2. Extract the top k components:
   - U_k: the first k columns of U ($U_k \in R^{m \times k}$)
   - Σ_k: the top-left $k \times k$ diagonal block of Σ
   - V_k: the first k columns of V ($V_k \in R^{n \times k}$)

3. Construct the rank-k approximation of A:

$$A_k = U_k \Sigma_k V_k^T$$

   This matrix A_k is the best rank-k approximation of A in terms of the Frobenius norm.

To implement this algorithm conveniently, I used Power Method. Power iteration is a simple and efficient algorithm used to estimate the largest eigenvalue and corresponding eigenvector of a matrix. It forms the basis for estimating singular values and vectors in our custom implementation of Truncated SVD.

Given a matrix $A \in R^{n \times n}$:

1. Start with a random initial vector $b_0 \in R^n$ (not equal to the zero vector).

2. For $k = 1, 2, \ldots$, compute the next vector as:

$$b_k = \frac{Ab_{k-1}}{\|Ab_{k-1}\|}$$

3. Repeat the iteration until convergence (i.e., b_k stabilizes).

4. The vector b_k approximates the dominant eigenvector of A.

5. The corresponding eigenvalue can be approximated as:

$$\lambda_{max} \approx b_k^T A b_k$$

3 Image Compression

As the images are represented in RGB format, Truncated SVD is applied separately to each of the three color channels (Red, Green, Blue). After processing each channel independently, the results are combined to reconstruct the final image. This procedure is repeated for various compression levels by applying Truncated SVD at ranks r = 2, 4, 8, 16, 32, 64.
Original and compressed images corresponding ranks are given below with their storage values and errors.

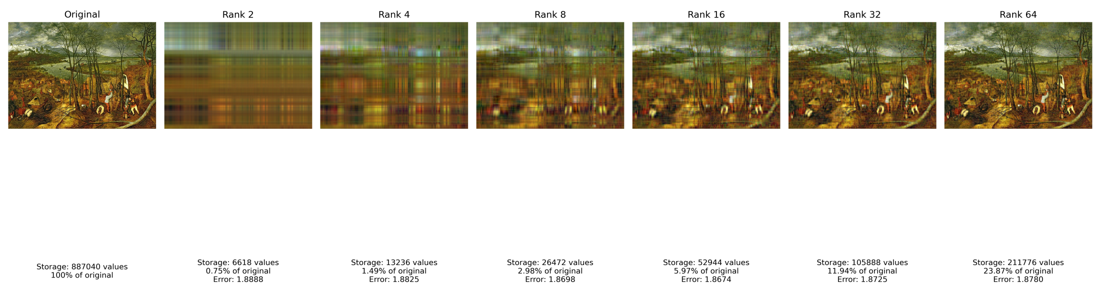

Figure 1: Brueghel Image Results

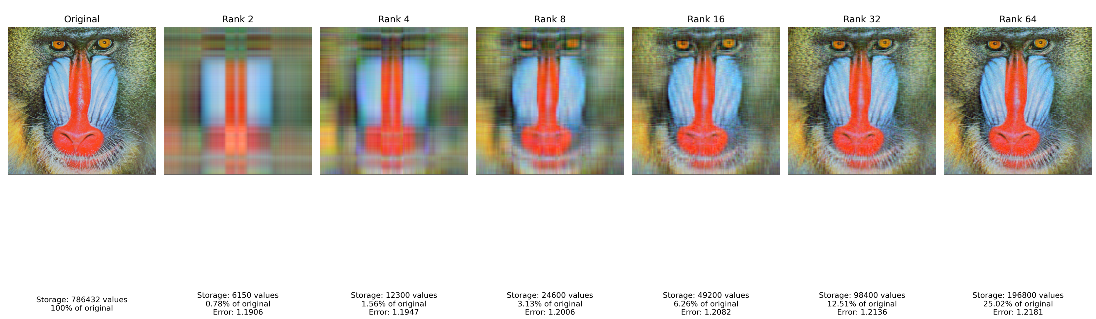

Figure 2: Mandrill Image Results

4 Graphical Results

Error vs rank graphs are given below.

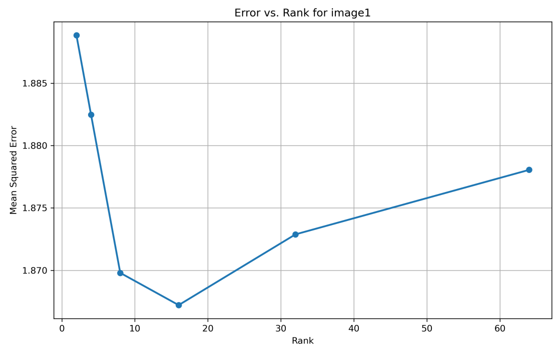

Figure 3: Error vs rank for Brueghel Image

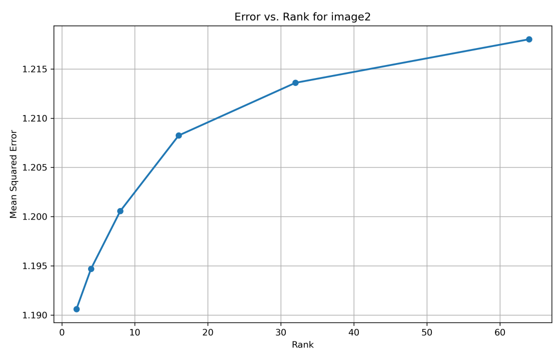

Figure 4: Error vs rank for Mandrill Image

Mean square error initially decreases as the rank increases, reaches a minimum, and then starts increasing again for Brueghel image. Increasing the rank beyond this point likely introduces overfitting or noise. The reason for this situation may be that the colors in the first image are close to each other for the RGB format, and when TSVD is applied separately and combined again, close values in terms of colors are achieved.
On the other hand, trend of mean square error is monotonically increases with increasing rank for Mandrill image. Mandrill image contains information that is better captured at very low-rank approximations, and adding more rank introduces unnecessary components or noise.

Storage comparison plots are given below. Storage trends for different ranks are similar for both images.

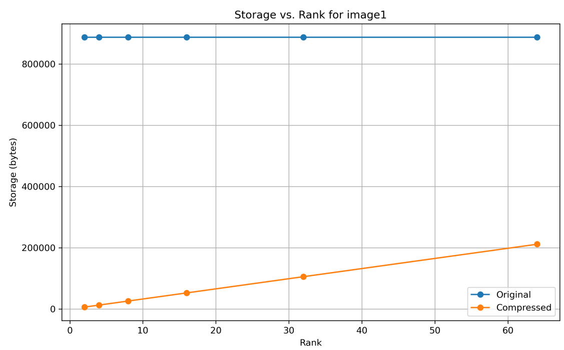

Figure 5: Plot of Storage Brueghel Image

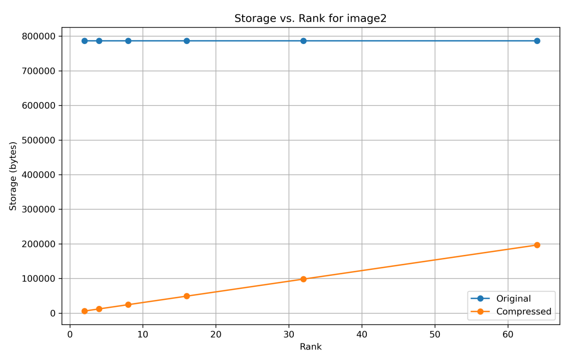

Figure 6: Plot of Storage Mandrill Image

5 Conclusion

In this project, two images with different color characteristics were compressed using truncated SVD. Brueghel image, which contained smoother and more similar RGB tones, showed a U-shaped relationship between reconstruction error and rank. The mean squared error initially decreased as the rank increased, reached a minimum around rank 16, and then increased, indicating that mid-range ranks provided the best balance between compression and fidelity.

In contrast, Mandrill image, which included more distinct and saturated colors, exhibited a monotonically increasing error trend. The lowest MSE was observed at the smallest rank, suggesting that even low-rank approximations captured the essential structure of the image effectively.

Despite these error trends, Mandrill image remained more visually distinguishable at low ranks due to its sharp and vivid color transitions. In comparison, Brueghel image became harder to interpret visually at low ranks because of its smoother and more blended color gradients. This highlights the limitation of relying solely on quantitative error metrics like MSE, as perceptual quality can vary significantly depending on the image content.

Finally, both images resulted in nearly identical compressed sizes for corresponding ranks, indicating that storage requirements were largely unaffected by the nature of the image content. Overall, these findings emphasize the importance of considering both objective error and perceptual quality when applying truncated SVD for image compression.

References

[1] Strang, Gilbert. Introduction to Linear Algebra. 5th ed. Wellesley-Cambridge Press, 2016.

[2] Truncated SVD in Python. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html.

[3] https://matplotlib.org/stable/api/$_a s_g en/matplotlib.pyplot.plot.html$