Singular Value Decomposition and it's Applications in Image Processing and Noise Reduction

Aditya Raj*
School of Physical Sciences, National Institute of Science Education and Research, HBNI, Jatni-752050, India

(Dated: November 16, 2025)

Singular Value Decomposition (SVD) is a powerful matrix factorization technique that expresses any real matrix A as $A = U\Sigma V^T$, where U and V are orthogonal matrices and Σ is diagonal. This decomposition generalizes the eigendecomposition to non-square and singular matrices, revealing insights into their rank, structure, and subspaces. This study presents the mathematical foundations of SVD, its dimensional analysis, and its applications in image compression and noise reduction. Performance is evaluated using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) metrics. Our results show that SVD can achieve significant compression ratios (up to 90%) while maintaining visual quality, and effectively reduce noise in astronomical imagery while preserving important features.

INTRODUCTION

Singular Value Decomposition stands as one of the most fundamental and powerful concepts in linear algebra and numerical computing. Its applications span across diverse fields including scientific computing, data analysis, machine learning, and image processing. SVD provides a robust framework for understanding matrix structure, solving linear systems, and performing dimensionality reduction.

Unlike eigendecomposition, which is limited to square matrices with linearly independent eigenvectors, SVD applies to any real matrix, making it universally applicable. It serves as the mathematical foundation for Principal Component Analysis (PCA), has applications in solving ill-posed problems, and plays a crucial role in data compression and noise reduction. In image processing, SVD enables efficient storage and transmission of image data while maintaining perceptual quality, and provides effective methods for removing noise without significantly degrading important image features.

MATHEMATICAL FOUNDATIONS OF SINGULAR VALUE DECOMPOSITION

For any real matrix $A \in \mathbb{R}^{m\times n}$ of rank r, the Singular Value Decomposition is given by:

$$A = U\Sigma V^T \tag{1}$$

where:

- $U \in \mathbb{R}^{m\times m}$ is orthogonal ($U^T U = I_m$)
- $V \in \mathbb{R}^{n\times n}$ is orthogonal ($V^T V = I_n$)
- $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal with non-negative entries

The diagonal elements $\sigma_1, \sigma_2, \ldots, \sigma_r$ of Σ are called singular values, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. The columns of U are the left singular vectors, and the columns of V are the right singular vectors.

Mathematical Derivation

To derive SVD, consider the symmetric matrices $A^T A$ and AA^T:

Step 1: Right singular vectors

$$A^T A = (U\Sigma V^T)^T (U\Sigma V^T) \tag{2}$$

$$A^T A = V\Sigma^T \Sigma V^T \tag{3}$$

Since $A^T A$ is symmetric positive semi-definite, it has non-negative eigenvalues and orthogonal eigenvectors. Thus, V consists of the eigenvectors of $A^T A$, and $\Sigma^T\Sigma$ contains the eigenvalues.

Step 2: Left singular vectors

$$AA^T = (U\Sigma V^T)(U\Sigma V^T)^T \tag{4}$$

$$AA^T = U\Sigma\Sigma^T U^T \tag{5}$$

Similarly, U consists of the eigenvectors of AA^T, and $\Sigma\Sigma^T$ contains the eigenvalues.

Step 3: Singular values The singular values σ_i are the square roots of the eigenvalues of $A^T A$ (or AA^T):

$$\sigma_i = \sqrt{\lambda_i(A^T A)} \tag{6}$$

DIMENSIONAL ANALYSIS OF SVD

Matrix Dimensions

For a matrix $A \in \mathbb{R}^{m\times n}$ with rank r, the SVD components have the following dimensions:

TABLE I: Dimensional Analysis of SVD Components

Component	Matrix	Dimensions
Original Matrix	A	$m \times n$
Left Singular Vectors	U	$m \times m$
Singular Values Matrix	Σ	$m \times n$
Right Singular Vectors	V	$n \times n$
Rank	$r(A)$	$r \leq \min(m, n)$

The Σ matrix has the structure:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_r \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \tag{7}$$

For different matrix shapes:

Case 1: Square matrix $(m = n)$

$$A_{n\times n} = U_{n\times n}\Sigma_{n\times n}V_{n\times n}^T \tag{8}$$

Case 2: Tall matrix $(m > n)$

$$A_{m\times n} = U_{m\times m}\begin{bmatrix} \Sigma_{n\times n} \\ 0_{(m-n)\times n} \end{bmatrix} V_{n\times n}^T \tag{9}$$

Case 3: Wide matrix $(m < n)$

$$A_{m\times n} = U_{m\times m}\begin{bmatrix} \Sigma_{m\times m} & 0_{m\times(n-m)} \end{bmatrix} V_{n\times n}^T \tag{10}$$

LOW-RANK APPROXIMATION AND IMAGE COMPRESSION

Theoretical Foundation

Given the SVD $A = U\Sigma V^T$, any rank-k approximation of A can be constructed by keeping only the first k singular values and their corresponding vectors:

$$A_k = U_k\Sigma_k V_k^T \tag{11}$$

where U_k, Σ_k, and V_k contain the first k columns/rows of U, Σ, and V respectively.

Cumulative Energy Analysis

The relative energy captured by the first k singular values is:

$$E_k = \frac{\sum_{i=1}^{k}\sigma_i^2}{\sum_{i=1}^{r}\sigma_i^2} \tag{12}$$

This measures the proportion of the total "energy" (variance) retained in the approximation.

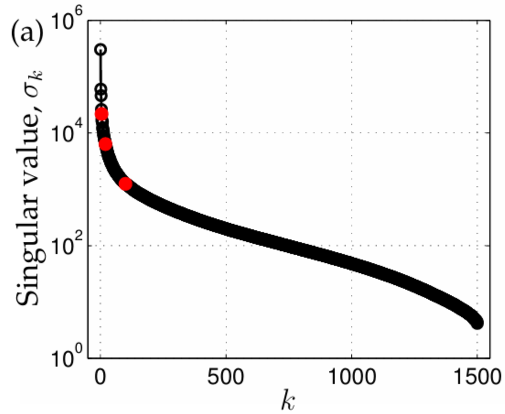


FIG. 1: Singular values decay pattern for a typical image

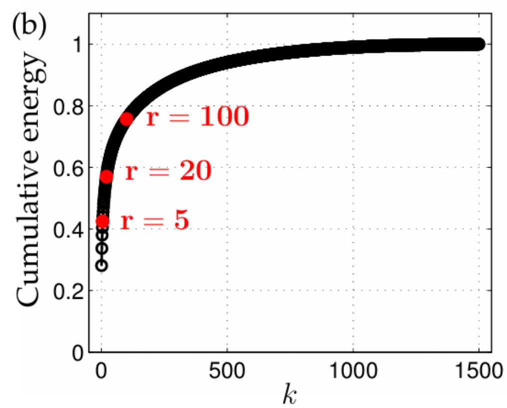


FIG. 2: Cumulative energy captured by first k singular values

ALGORITHM AND IMPLEMENTATION

SVD Computation Algorithm

In this section we outline the procedure used to compute the SVD of an $m \times n$ image matrix and perform rank–k reconstruction. The actual Python implementation is provided separately in the source files.

Algorithm 1 Rank-k SVD Approximation of an Image

Require: Image matrix $I_{m\times n}$, rank parameter k
Ensure: Reconstructed image I_k
  Convert image to grayscale (if needed)
  Compute SVD: $I = U\Sigma V^T$
  Truncate components:
    $U_k = U(:, 1\!:\!k)$
    $\Sigma_k = \Sigma(1\!:\!k, 1\!:\!k)$
    $V_k^T = V^T(1\!:\!k, :)$
  Reconstruct: $I_k = U_k\Sigma_k V_k^T$
  Clip values to valid image range
  return I_k

Implementation Details

The numerical computation is performed in Python using NumPy's `svd()` routine, which returns the left singular vectors, singular values, and right singular vectors of the image matrix. Noise models (Gaussian) is applied to the original image before performing SVD-based denoising.

Quality evaluation is performed using:

- **PSNR (Peak Signal-to-Noise Ratio)** — measures reconstruction fidelity,
- **SSIM (Structural Similarity Index)** — measures structural similarity.

Only the pseudocode is included here; the complete implementation is available in the accompanying Python source file.

IMPLEMENTATION FOR IMAGE COMPRESSION

Grayscale Image Compression

We now demonstrate how the low-rank approximation discussed earlier can be applied to compress a real image. A real image is typically stored in the RGB format, where each pixel contains three intensity values corresponding to the Red, Green, and Blue channels. Thus, an RGB image can be represented as a tensor of size $m \times n \times 3$. To apply SVD directly, we first convert the RGB image into a grayscale image. This reduces the three-channel representation to a single intensity channel, producing an $m \times n$ matrix. The conversion is performed using a weighted sum of the RGB values, commonly defined as

$$I_{\text{gray}} = 0.299\,R + 0.587\,G + 0.114\,B,$$

The resulting grayscale matrix. Since a grayscale image can be represented as an $m \times n$ matrix of pixel intensities, we compute its SVD and reconstruct lower-rank versions of the image by keeping only the first k singular values.

To visualise the effect of different rank values, the original image was compressed using $k = 5$, 10, 25, 40, and 50. All six results are shown together in Fig. 3. As k increases, more singular values are retained and the reconstructed image becomes visually closer to the original. Even for relatively small k, the overall structure of the image is preserved.

In addition, the singular value decay and cumulative energy plots help justify the choice of k. The rapid drop in singular values indicates that most of the image information is concentrated in the first few components. The cumulative energy curve shows that a large fraction of the total energy (above 90%) is captured with only a moderate number of singular values.

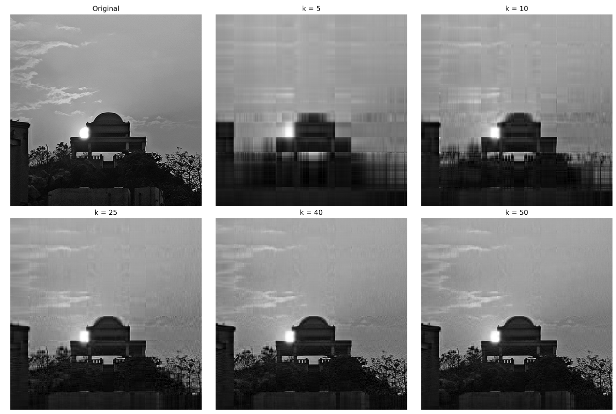


FIG. 3: Original image and its SVD-based low-rank reconstructions for $k = 5, 10, 25, 40, 50$.

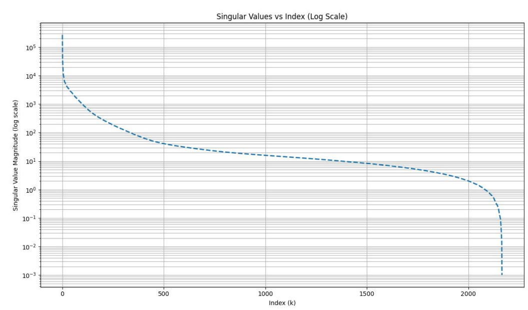


FIG. 4: Plot of singular values in log scale against all k values.

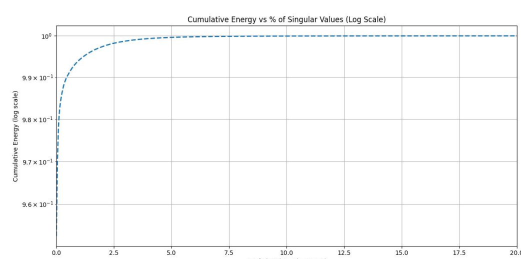


FIG. 5: Cumulative energy captured by the first k singular values

SVD FOR NOISE REDUCTION IN ASTRONOMICAL IMAGES

Noise in digital images arises from various sources including sensor noise, transmission errors, and environmental factors. In astronomical imaging, noise is particularly problematic as it can obscure faint celestial objects and stellar features. SVD-based noise reduction leverages the observation that noise is typically contained in the smaller singular values, while signal (important image features) resides in the larger singular values.

The SVD noise reduction process involves:

1. Computing SVD of the noisy image matrix A_{noisy}
2. Reconstructing the image using only the first singular components k

Mathematically:

$$A_{\text{denoised}} = U_k \Sigma_k V_k^T \tag{13}$$

where k is chosen based on noise characteristics and the desired level of noise reduction.

PSNR AND SSIM METRICS ANALYSIS

Peak Signal-to-Noise Ratio (PSNR)

PSNR is a widely used metric for measuring image quality, particularly in compression and denoising applications. Measures the ratio between the maximum possible power of a signal and the power of corruption of noise.

The PSNR is defined as:

$$\text{PSNR} = 10 \log_{10} \frac{\text{MAX}_I^2}{\text{MSE}} \tag{14}$$

where:

- MAX_I is the maximum possible pixel value (255 for 8-bit images)
- MSE is the Mean Squared Error between the original and reconstructed images

The MSE is calculated as:

$$\text{MSE} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} [I(i,j) - \hat{I}(i,j)]^2 \tag{15}$$

where I is the original image and $\hat{I}$ is the reconstructed image.

Structural Similarity Index (SSIM)

SSIM is a perceptual metric that measures the structural similarity between two images. Unlike PSNR, which focuses on pixel-wise differences, SSIM considers luminance, contrast, and structural information.

SSIM between two windows x and y is calculated as:

$$\text{SSIM}(x,y) = \frac{(2\mu_x \mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{16}$$

where:

- μ_x, μ_y are the mean values of windows x and y
- σ_x^2, σ_y^2 are the variances of windows x and y
- σ_{xy} is the covariance of windows x and y
- $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$ are stability constants
- L is the dynamic range of pixel values
- $k_1 = 0.01$ and $k_2 = 0.03$ by default

SVD Noise Reduction Results

We demonstrate SVD noise reduction on an image containing a single star; this is a cropped-out image of a single star from the image of the $M46$ open star cluster provided by NAC.

FIG. 6: Image of target star

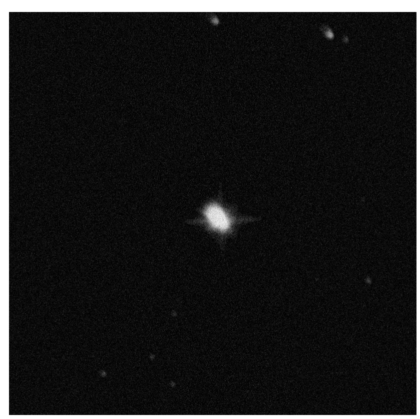
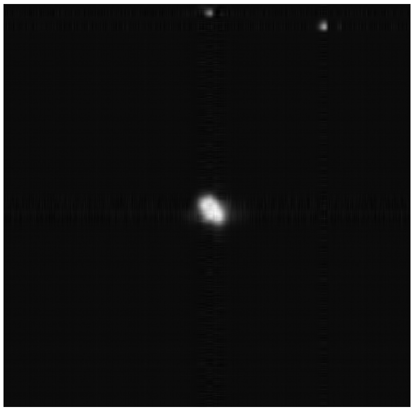
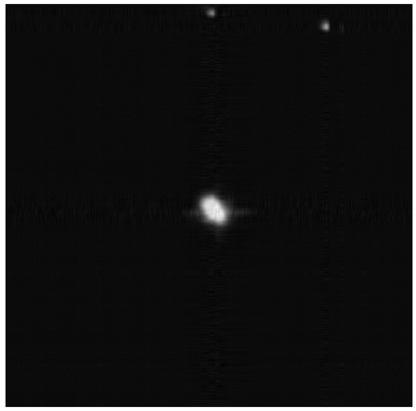


FIG. 7: Star field image with added Gaussian noise ($sigma = 30$)

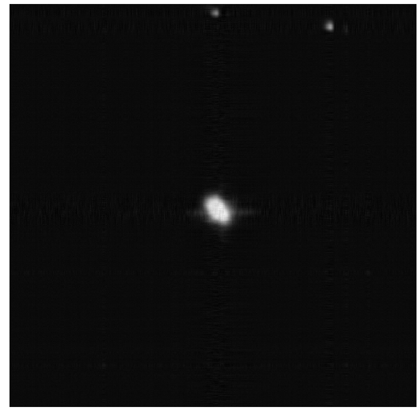
Comparing clean vs noisy image. We have, PSNR = 21.67dB and SSIM = 0.0201



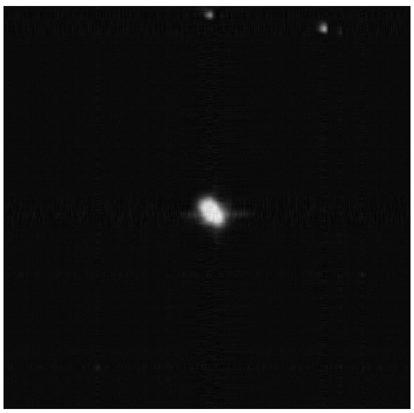
(a) k=5 (PSNR: 27.06 dB, SSIM: 0.0918)



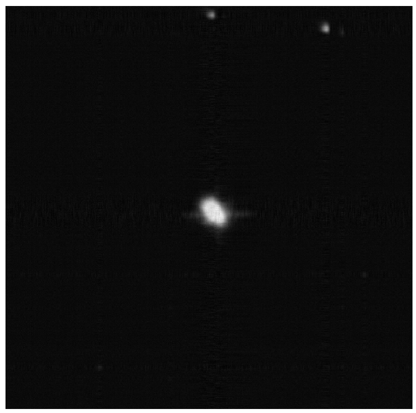
(b) k=10 (PSNR: 26.93 dB, SSIM: 0.0860)



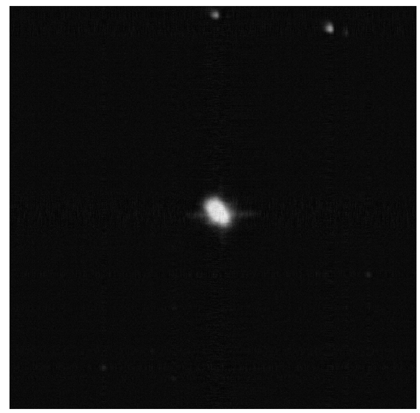
(c) k=15 (PSNR: 26.79 dB, SSIM: 0.0811)



(d) k=20 (PSNR: 26.65 dB, SSIM: 0.0769)



(e) k=30 (PSNR: 26.40 dB, SSIM: 0.0701)



(f) k=50 (PSNR: 25.97 dB, SSIM: 0.0605)

FIG. 8: Low-rank approximations of the image for different values of k.

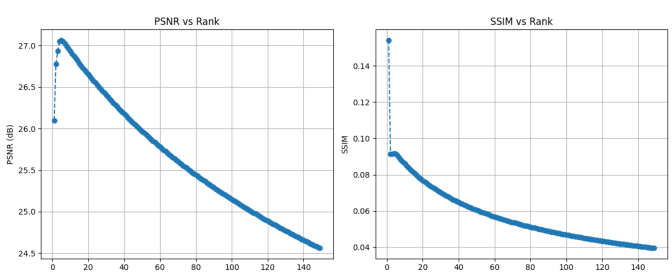


FIG. 9: PSNR and SSIM vs. number of singular values retained

Analysis of Results

The results obtained from low-rank image reconstruction and noise reduction highlight several important trends:

1. **Optimal k Selection**: Very small values of k (e.g., $k = 5$ or $k = 10$) produce highly blurred images, since only a small fraction of the image energy is retained. Visual quality improves as k increases, but with diminishing returns beyond $k \approx 40$.

2. **PSNR and SSIM Behaviour**: Both PSNR and SSIM decrease gradually as k is reduced. This agrees with the theoretical expectation that dropping smaller singular values removes high-frequency detail. The rate of degradation slows for higher k values, indicating most significant information is concentrated in the top singular components.

3. **Stellar Feature Preservation**: In the case of star image, SVD effectively removes high-frequency noise while preserving bright star cores. Even for moderate values of k (around 20–30), important stellar structures remain visible despite aggressive dimensionality reduction.

4. **Compression vs. Quality**: Lower k values yield

excellent compression ratios but poorer perceptual quality. For the test image used, the practical balance between quality and compression occurs around $k = 25$–40, consistent with the observed PSNR ($\sim$26–27 dB) and moderate SSIM values.

RESULTS AND DISCUSSION

The behaviour of the reconstructed images clearly reflects the theoretical properties of the singular value spectrum. For compression, small values of k (5–10) retain only the coarse intensity structure, producing smooth but blurred images. As k increases, finer details reappear, with diminishing visual improvement beyond $k \approx 40$. This aligns with the cumulative energy plot, which shows that the first $\sim$20 singular values capture most of the image variance. While PSNR decreases slightly from 27 dB ($k = 5$) to 26 dB ($k = 50$), the perceptual difference between $k = 30$ and $k = 50$ is small, consistent with the flattening of the tail of the singular-value spectrum.

For the noisy star-field image, low-rank reconstruction effectively removes high-frequency noise while preserving bright stellar cores. The PSNR rises from 15 dB (noisy) to 28 dB at $k = 100$, with the best balance between noise suppression and structural preservation appearing around $k = 20$–30. Very low ranks ($k < 10$) oversmooth the stars, while extremely high ranks provide minimal additional improvement. The method therefore serves as both a compression and denoising tool, with performance directly tied to the distribution of singular values.

CONCLUSION

SVD provides a simple but powerful framework for image compression and noise reduction. The dominant singular values capture most of the structural information, enabling compact low-rank approximations that preserve global image features. For both clean and noisy images, practical performance is achieved with moderate ranks ($k \sim 20$–40), offering a good trade-off between quality, compression efficiency, and computational cost. Overall, the results demonstrate that SVD is an effective and theoretically grounded tool for compact image representation and denoising.

ACKNOWLEDGEMENT

I express my sincere gratitude to the P346 course instructor, Prof. Subhasish Basak, for providing the opportunity, guidance, and support that made this project possible. I also acknowledge open-source information platforms such as GeeksforGeeks for their clear and accessible explanations of many of the concepts used in this work, as well as the broader open-source community for developing the Python libraries and computational tools that enabled the implementation and analysis presented here.

REFERENCES

* [aditya.raj@niser.ac.in]

[1] Gilbert Strang, *Introduction to Linear Algebra*, 4th Edition, Wellesley–Cambridge Press, 2009.

[2] Steven L. Brunton and J. Nathan Kutz, *Data-Driven Science and Engineering*. Available online: `https://databookuw.com/`

[3] Jiaxing Shen, "Singular Value Decomposition (SVD) and Its Applications," Tutorial Presentation, 5 April 2022. Available: `https://shenjiaxing.github.io/talks/SVD%20tutorial%202022.pdf`

[4] Andrew Gibiansky, "Cool Linear Algebra: Singular Value Decomposition." Available: `https://andrew.gibiansky.com/blog/mathematics/cool-linear-algebra-singular-value-decomposition/`

[5] GeeksforGeeks, "Singular Value Decomposition (SVD) in Machine Learning." Available: `https://www.geeksforgeeks.org/machine-learning/singular-value-decomposition-svd/`

[6] GeeksforGeeks, "Python — Peak Signal-to-Noise Ratio (PSNR)." Available: `https://www.geeksforgeeks.org/python/python-peak-signal-to-noise-ratio-psnr/`

[7] Jethro Jens Norbert Simatupang, "Noise Reduction in Satellite Imagery Using Singular Value Decomposition," Institut Teknologi Bandung (ITB), Indonesia. Available: `https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2024-2025/Makalah/Makalah-IF2123-Algeo-2024-(85).pdf`