University
of Colorado
Boulder

UNIVERSITY OF COLORADO BOULDER

Introduction to Singular Value Decomposition

Student :
Ainsley Braunscheidel
Divya Nallawar
Luke Petet
Reza Naiman

Teacher :
Prof. James H. Curry

October 21, 2024

Contents

1 Abstract

The primary purpose of this project is to explore the application of Singular Value Decomposition (SVD) on various real-world data sets and examine how it facilitates dimensionality reduction, feature extraction, and the creation of reduced-order models. Through the analysis of three or four interesting and diverse data sets, we aim to illustrate the practical use of SVD in uncovering latent structures in data, while also discussing the significance of the Eckart-Young Theorem, which forms a theoretical foundation for SVD-based dimensionality reduction. A key focus of this project is to highlight the importance of the Eckart-Young Theorem, which provides the theoretical underpinning for using SVD in data approximation.

2 Introduction to SVD

Consider the linear equation $A\mathbf{x} = \mathbf{B}$, where A is a matrix and $\mathbf{x}$ and $\mathbf{B}$ are vectors. This system of equations can exhibit different types of solutions depending on the properties of matrix A. Specifically, the system may have a unique solution, no solution, or infinitely many solutions. One common approach to finding a solution is through the **least squares** method, which minimizes the residual error when an exact solution is not attainable.

Alternatively, if A is invertible, the system can be solved using the inverse of A, such that $\mathbf{x} = A^{-1}\mathbf{B}$. However, this approach is only feasible when A is a square matrix and its determinant is non-zero, ensuring that A^{-1} exists. For matrices with a zero determinant or those that are non-square, an inverse does not exist, and other methods, such as the pseudoinverse or least squares, must be employed to find an approximate solution. That's when we can use SVD to solve the linear equation $Ax = B$.

Singular Value Decomposition (SVD)[3] is a fundamental mathematical technique used for decomposing or factorizing a matrix into three distinct components: U, Σ, and V^T.[17, 4] The three matrices that make up the SVD equation are related to the base vectors in terms of the column space via the U matrix with its left singular values, the row space via the V^T matrix with its right singular values, and the scalar values via the Σ matrix.

Given a matrix A (of size mxn, where m is the number of rows and n is the number of columns, which can represent various data such as images, text, or other multi-dimensional data sets), the decomposition can be written as:

$$A_{m\times n} = U\Sigma V^T$$

- U: This is a left singular matrix where each column is an orthonormal eigenvector of A^TA.
- Σ : A diagonal matrix containing the singular values of A, sorted in decreasing order.
- V^T: This is a right singular matrix where each row is an orthonormal eigenvector of AA^T.

Let's assume the size of matrix A being $m \times n$. After the SVD, the sizes of each matrices are:

- $U : m \times r$
- $\Sigma : r \times r$
- $v^T : r \times n$

Here m is the number of rows, n is the number of columns and r is the rank of the matrix A. The above decomposition with the size of all the matrices can be written as:

$$A_{m\times n} = U_{m\times r}\Sigma_{r\times r}V^T_{r\times n}$$

For matrix multiplication to be valid, the number of columns in the first matrix must be equal to the number of rows in the second matrix[16]. In the context of Singular Value Decomposition (SVD), the matrices U and V represent orthogonal transformations, while the diagonal matrix Σ contains the singular values that scale the transformation along specific axes. This decomposition allows SVD to capture both the structure and complexity of the original matrix A, making it useful in a wide variety of applications such as dimensionality reduction, noise reduction, and data compression.

2.1 Purpose of SVD

Singular Value Decomposition (SVD) can be applied to any matrix, whether it is square, rectangular, or even complex. This universality distinguishes SVD from other matrix decomposition techniques[17], as it is applicable to any $m \times n$ matrix. In the context of modern data analysis, where high-dimensional and complex data—such as images, text, or large-scale datasets—are prevalent, SVD plays a crucial role. It serves as a powerful tool for both understanding the underlying structure of such complex data and simplifying it, thereby enabling more efficient and insightful analysis.

3 Understanding and application of Eckart-Young Theorem

The Singular Value Decomposition (SVD) of a given dataset results in three matrices: U, Σ, and V^T, with Σ containing the most significant information regarding the dataset. However, comprehending and utilizing the full decomposition can be computationally expensive, particularly for large datasets. This is where Low-Rank Approximation[15] becomes essential. By retaining only the most significant singular values and their corresponding singular vectors, Low-Rank Approximation reduces the complexity of the matrix while preserving its essential features, making computations more efficient without significant loss of information. Low-Rank Approximation is given by the Eckart-Young Theorem.[10]

The theorem states that given a matrix A of rank r, the best rank- k approximation A_k (where $k < r$) is obtained by truncating the Singular Value Decomposition (SVD) of A, keeping only the top k singular values. Mathematically, this can be expressed as:

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

Where:

- σ_i are the singular values of A, sorted in decreasing order.
- u_i and v_i^T are the corresponding left and right singular vectors, respectively.
- k is the number of singular values to retain.

The approximation A_k minimizes the Frobenius norm[2] of the difference between A and its approximation, i.e., $\|A-A_k\|_F$, which represents the sum of the squared differences between the entries of A and A_k.
Formally the theorem guarantees that:

$$\min_{B:\text{rank}(B)\leq k} \|A - B\|_F = \|A - A_k\|_F = \sigma_{k+1} \tag{2}$$

This means that the matrix A_k is the closest rank- k matrix to A in terms of the Frobenius norm, and no other rank-k matrix can approximate A better than A_k.

We understand that the SVD decomposes any matrix into 3 matrices, where the Σ matrix is the heart of the data. Now Eckart-Young theorem can be applied to reduce the rank of the matrix A to k while preserving as much information as possible. This can be done in 2 steps:

- We know the singular values are in descending order. Truncate the Σ values after the $k-th$ largest one. (One way to achieve this is by plotting the singular values and truncate at the point where the decrement is slow).
- Keep only the first k columns of U and the first k rows of V^T

4 Applications of SVD

Singular Value Decomposition (SVD) has a wide range of applications across various fields. Its ability to decompose complex data into simpler components makes it an essential tool in areas like data science, image processing, machine learning, and more.
While there are many factorization theorem in the field of Linear Algebra, SVD are widely used because it provides the most optimal low-rank approximation of a matrix according to theorem .

Theorem (Norms and Orthogonal Transformation) The 2-norm is invariant under orthogonal transformation, for if $Q^TQ = I$ then $\|\mathbf{Qx}\|_2^2 = x^TQ^TQx = x^Tx = \|\mathbf{x}\|_2^2$. The matrix 2-norm and the Frobenius norm are also invariant with respect to orthogonal transformations. In particular, it is easy to show that for all orthogonal Q and Z of appropriate dimension we have $\|\mathbf{QAZ}\|_2 = \|\mathbf{A}\|_2$.

We know that the singular values of the matrix A are arranged in descending order in the diagonal entries of the Σ matrix. This means that the largest singular values are positioned first, and the smallest singular values are placed at the end of the diagonal.

This is crucial in many fields because the largest singular values capture the most significant information in the data. This property gives Singular Value Decomposition (SVD) the power to be applied in different applications[9, 7], such as:

1. ***Dimensionality Reduction/low rank approximation/ Data Compression***
   - In datasets with high dimensionality, many features may be redundant or irrelevant. SVD helps reduce the number of features while preserving the most important information.
   - By retaining only the largest singular values, you can reduce the dimensionality of the dataset while still capturing the essential structure.

2. ***Noise Reduction (Denoising)***[8, 5]
   - SVD helps in denoising data by eliminating smaller singular values that often correspond to noise or insignificant variations.
   - Small singular values, which usually represent noise, are discarded, and only the larger singular values are kept to reconstruct a cleaner version of the data.

3. ***Recommender Systems***
   - SVD is heavily used in collaborative filtering for building recommender systems, such as movie or product recommendation platforms.
   - By decomposing the user-item interaction matrix, SVD identifies latent factors that describe the preferences of users and the characteristics of items.

4. ***Latent Semantic Analysis (LSA)***
   - In natural language processing (NLP), SVD is used in Latent Semantic Analysis (LSA) to analyze relationships between words and documents in a corpus.
   - SVD decomposes the term-document matrix into singular values and vectors, capturing the latent semantic structure of the text data.

5. ***Image Compression and Processing***
   - SVD is widely used in image processing, particularly for compression and denoising.
   - retaining the most significant singular values, images can be compressed without losing much detail. SVD is also used to enhance or clean images by removing noise.

6. ***Facial Recognition***
   - SVD can be used for face recognition tasks by reducing the dimensionality of the data while keeping essential features for identification.
   - In facial recognition, each face is represented as a matrix, and SVD is applied to extract the most critical features of each face. These features are then used to

compare and match faces.

7. ***Anomaly Detection***
   - SVD is applied in anomaly detection [18, 6], particularly for identifying outliers in high-dimensional data.
   - Anomalies are typically points that cannot be well approximated by a low-rank model. SVD helps identify these outliers by showing which points do not conform to the patterns in the data.

5 Datasets

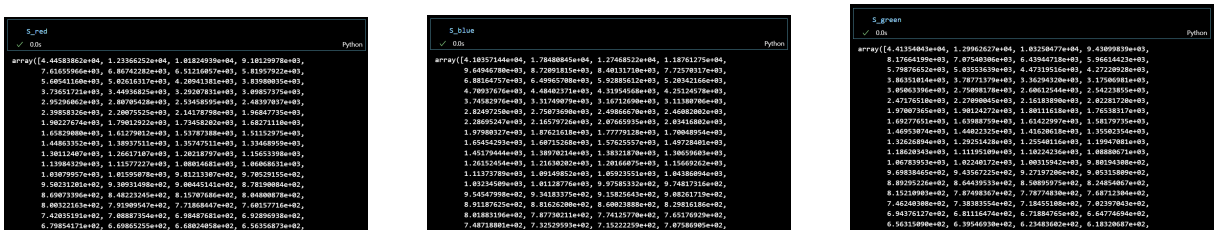
Although we know there are many possible applications of the SVD, (as mentioned in Applications of SVD section)[9], we will deep dive into the image compression and processing application for this report.

In a world of Internet of Things sharing and accessing data have become the norm. One of the many forms of data we use in our daily lives are images or pictures. We use images to capture special moments in our lives, so we could share on social media or store them for memories. For computers images are like a matrix of pixels, which lets the computer understand and store the image in memory. However, images sometimes can get big to store or transmit over the internet. Therefore, we need to reduce the image size or compress the image to reduce irrelevant information of the image to transmit it in an effective way.

To understand this application, we will be using an album cover image which we refer to as the "J-Cole image". [13]. J. Cole is an American rapper and record producer, born in Germany and raised in North Carolina. He has been releasing music since his debut album 2007 with his latest album being released this year. The image we chose for this project is the cover for his album titled 2014 Forest Hills Drive which was released in 2014. We selected this image because it was suggested by Reza as J. Cole is one of his favorite artists.

The image, consisting of 600×600 pixels, was first decomposed into its three distinct color channels: red, green, and blue. Singular Value Decomposition (SVD) was then performed independently on each of these color channels. In addition, SVD was applied to the gray scale version of the image for comparison. The resulting singular values, plotted against their corresponding indices, are presented in the graphs below, illustrating the distribution of singular values across both the color channels and the gray scale image.

Figure 1: A 600x600 image of J. Cole's Album



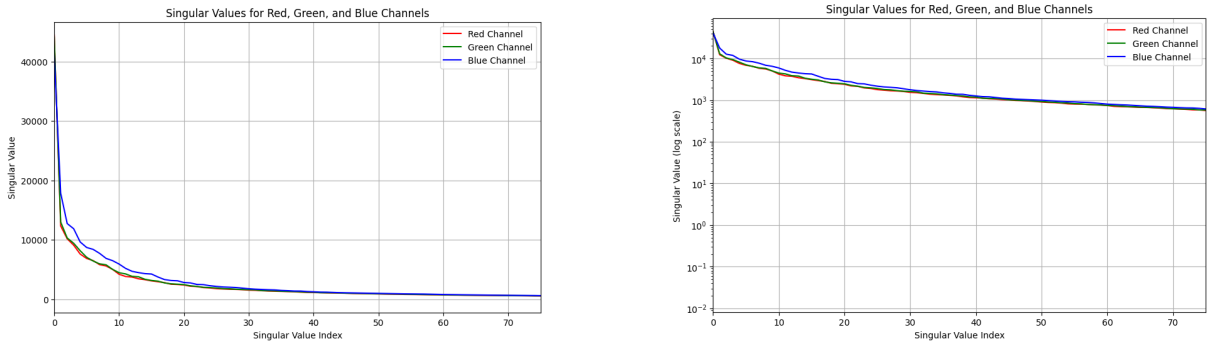
(a) Red

(b) Blue

(c) Green

Figure 2: Singular Values for different color channels

The Figure 2 has 3 sets singular values obtained for 3 different colour channel(Red, blue and green).



(a) Original (b) Scaled(log)

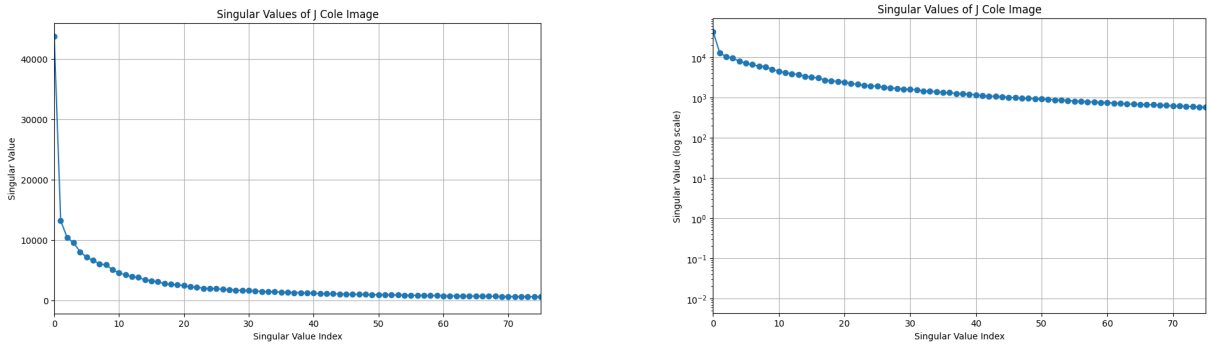
Figure 3: Singular Values plots for the three different color channels

As shown in Figure 3, both subfigures (a) and (b) demonstrate that the blue channel exhibits higher singular values compared to the other color channels. This is supported by our calculations, which indicate that the blue channel accounts for 36.27% of the total singular values, followed by the green channel at 32.18%, and the red channel at 31.55%. Subfigure (a) presents the singular values in their original form, while subfigure (b) provides a transformed representation of the same singular values for comparison.

```
s
✓ 0.0s                                                    Python
array([4.37552985e+04, 1.31860537e+04, 1.03754070e+04, 9.51291554e+03,
       8.03668321e+03, 7.10927195e+03, 6.60129525e+03, 6.02782446e+03,
       5.86965135e+03, 5.02696975e+03, 4.53759320e+03, 4.20951733e+03,
       3.89837827e+03, 3.79170925e+03, 3.36544966e+03, 3.20608318e+03,
       3.08309085e+03, 2.73278343e+03, 2.63319613e+03, 2.53052930e+03,
       2.45702439e+03, 2.22118403e+03, 2.15833918e+03, 1.99623065e+03,
       1.96787228e+03, 1.91609505e+03, 1.80487045e+03, 1.75418356e+03,
       1.68629987e+03, 1.63223766e+03, 1.60673183e+03, 1.55492444e+03,
       1.47023589e+03, 1.43027365e+03, 1.40906463e+03, 1.33873501e+03,
       1.32348512e+03, 1.27312405e+03, 1.25676517e+03, 1.20194901e+03,
       1.17216713e+03, 1.11132880e+03, 1.09005375e+03, 1.08541761e+03,
       1.05941137e+03, 1.01581638e+03, 9.98327221e+02, 9.71154679e+02,
       9.62619176e+02, 9.43150033e+02, 9.20629535e+02, 9.06813609e+02,
       8.77784231e+02, 8.68320594e+02, 8.50584087e+02, 8.20722167e+02,
       8.15526759e+02, 7.92013506e+02, 7.78902430e+02, 7.63922491e+02,
       7.50670498e+02, 7.27375488e+02, 7.13798829e+02, 6.95847825e+02,
       6.87128463e+02, 6.81861570e+02, 6.70277674e+02, 6.63055669e+02,
       6.59297053e+02, 6.37252994e+02, 6.24001956e+02, 6.15072172e+02,
       6.08370136e+02, 6.00415328e+02, 5.84950363e+02, 5.80447089e+02,
       5.65651066e+02, 5.57043964e+02, 5.49924760e+02, 5.47028842e+02,
       5.30514415e+02, 5.26298850e+02, 5.13269552e+02, 5.11235731e+02,
       4.97958078e+02, 4.96065964e+02, 4.89038033e+02, 4.76016955e+02,
       4.69073024e+02, 4.65302739e+02, 4.61827490e+02, 4.42019480e+02,
```

Figure 4: Singular Values for gray scaled image

The Figure 4 shows the different singular values obtained for SVD on gray scaled image.



(a) Original (b) Scaled(log)

Figure 5: Singular values for the gray scale image

Figure 5 depicts the singular values for the gray scale image. The rapid decrease in singular values at the outset indicates that a substantial portion of the image's energy or information is concentrated in the first few singular values. The presence of a large number of smaller singular values, represented by the elongated tail, suggests that the remaining singular values contribute less significantly to the overall structure of the image, but still capture finer details. Part(a) is the non-scaled version of the singular values, part(b) represent the scaled(log) version of the graph for better understanding.

6 Analysis and insights

A useful plot when performing SVD is the "Cumulative Energy Plot" which visualizes the amount of energy captured by the first few singular values of a matrix. This energy is typically measured as a percentage of the total energy contained in the matrix. As we move along the x-axis, the cumulative energy[12] for each channel increases. The rate at which the curves rise is indicative of how quickly the energy is captured by the initial singular values. [3] To define the cumulative energy plot, we graph the formula

$$\frac{\sum_{i=1}^{k} \sigma_i}{\sum_{i=1}^{n} \sigma_i}, \quad \text{for } k = 1, 2, \ldots, n$$

Where,

- σ_i is each i-th singular value, or diagonal element of Σ matrix
- k is the index for each index in the cumulative sum being graphed
- n is the total number of singular values

Therefore, the graph shows the percent form of cumulative energy, or the cumulative sum of a portion from the set of singular values to the k-th value over the total sum of the singular values.

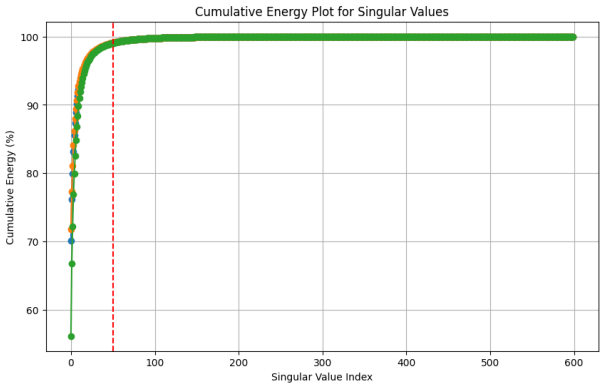


Figure 6: Cumulative Energy Plot of the RGB Scale Channels

As seen in Figure 6, only using rank 50, keeps more than 90 % of the data. This is significant to us because this plot shows us that by using SVD we can truncate a matrix and still keep 90 % of its data.

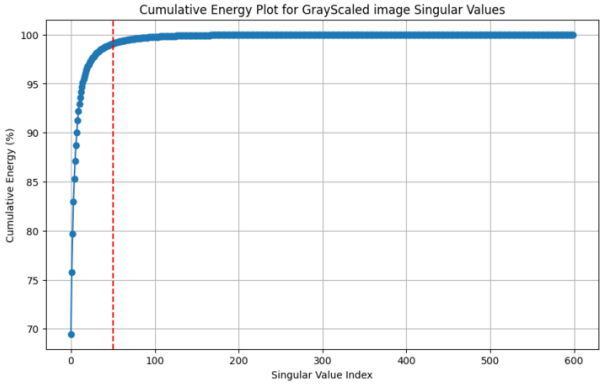
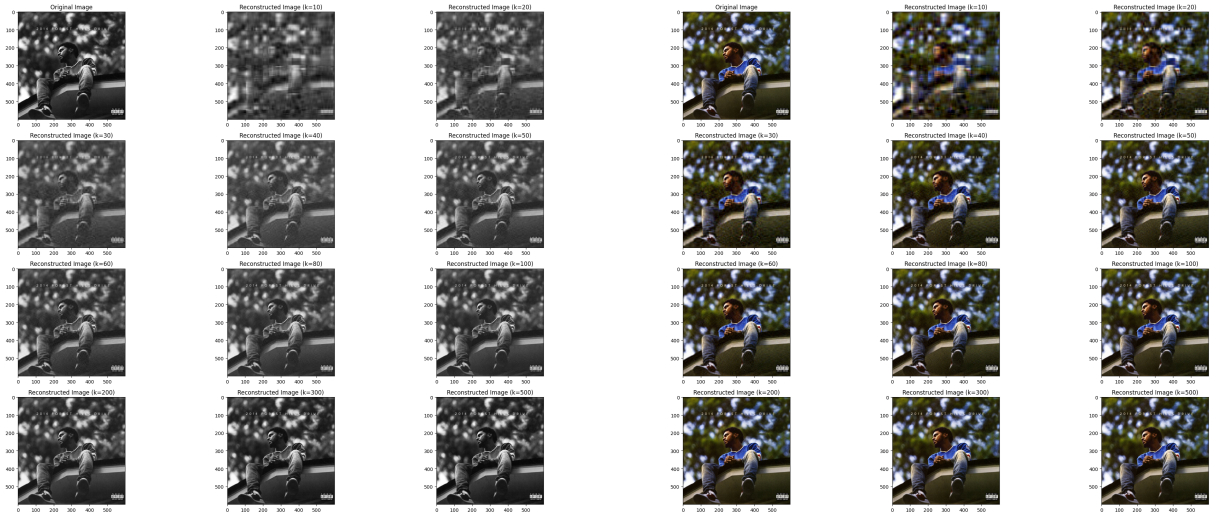


Figure 7: Cumulative Energy Plot of the Gray Scale Channels

As shown in Figure 7, using rank 50 for the gray scale image still keeps more than 95% of the data. By using these plot, we can see how significant and important SVD is in the image reduction.

The figure 8 has 2 sets of images which shows the reconstruction of the image both in gray scale and colour image for different ranks denoted by k. As the value of k increases, the reconstructed images become more similar to the original. This is because singular values capture more of the image's energy. For lower values of k, the reconstructed images appear blurry or distorted. This is due to the loss of detail associated with the discarded singular values. As k increases, finer details are gradually recovered, leading to a more accurate representation of the original image.

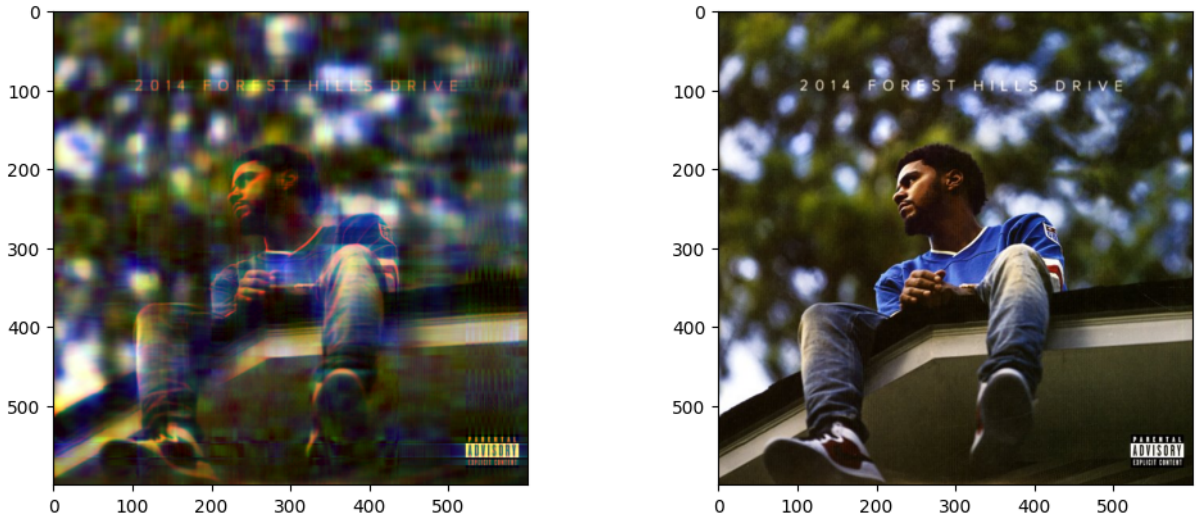


(a) Gray image reconstruction with different ranks

(b) Color image reconstruction with different ranks

Figure 8: Low rank approximation for different ranks.

This reconstruction suggests that with rank=50 most of the image is getting reconstructed for both colour image and gray scale image.



(a) Reconstructed

(b) Original image

Figure 9: Image reconstruction with different rank for different colour channels

For Figure 9, the part (a) showcase the reconstructed image for different rank for different channel. For this, we did rank 100 approximation for red channel, rank 15 approximation for green channel and rank 5 approximation for blue channel and reconstructed the image. Even with the minimum approximation for the blue and green channel, we can observe that most of the features are captured, indicating the majority of the color information in the image is concentrated within the green and blue channels, particularly in their lower-order components, while the red channel requires a significantly higher rank

to represent a similar level of detail. Part(b) is the original image.

7 Results

We can find the compression ratio and the compression percentage from the following formulas. Here, compressed size is the size of the image after compression has occurred from SVD. The Compression ratio is the ratio of the original size of the image to the size of he image after compression has occurred, showing how much compression has occurred. In our calculations, we found that for every 6 pixels of the original image, we have compressed them down to 1 pixel, on average.

$$\text{Compressed Size} = k(m+n)$$

$$\text{Compression Ratio} = \frac{\text{Original Size}}{\text{Compressed Size}} = \frac{mn}{k(m+n)}$$

$$\begin{aligned}\text{Compression Percentage} &= \left(1 - \frac{\text{Compressed Size}}{\text{Original Size}}\right) \cdot 100\% \\ &= \left(1 - \frac{k(m+n)}{mn}\right) \cdot 100\%\end{aligned}$$

Where,

- k columns from U (each of length m)
- k singular values in Σ
- k rows from V^T (each of length n)

We decided to make our rank, k, equal to 50 with m and n being equal to 600,

$$\begin{aligned}\text{Compressed Size} &= 50(600+600) \\ &= 60,000\end{aligned}$$

$$\begin{aligned}\text{Compression Ratio} &= \frac{600 * 600}{50(600+600)} \\ &= 6:1\end{aligned}$$

$$\begin{aligned}\text{Compression Percentage} &= \left(1 - \frac{50(600+600)}{600 * 600}\right) \cdot 100\% \\ &= 98.33\%\end{aligned}$$

To conclude, rank approximations can be used to compress images while preserving varying levels of detail. The choice of k depends on the desired balance between compression and image quality. We can also use elbow method to decide on rank.

Rank approximation via SVD is a powerful technique for image compression. It offers a good balance between compression ratio[11, 14] and image quality, and can be used to extract meaningful features from images.

However, it is important to consider the limitations of this approach, particularly for applications where lossless compression or high noise tolerance is required.

8 Limitations

Some of the limitations of SVD[1] are listed below

- Interpretability: The resulting singular values might not have the direct interpretation
- Sensitive to outliers: Outliers in the data can impact the quality of the decomposition, potentially leading to distorted results. (In case of an image, An unexpected data point which is significantly different from all other points.

9 Future Work

For future work on singular value decomposition within our group, we would be interested to see how image compression applies to particularly noisy images. As noted in the results, SVD is a poor method of image compression when applied to noisy images - notably due to the inability to discern the wanted content of the image from the unwanted noise of grain or blur. Therefore, a study of alternate methods of compression would allow for a more robust set of solutions towards image compression. Additionally, we would also be interested in developing a process for singular value decomposition within R. The current method for singular value decomposition compression is written for Python. Python, as an interpreted language, carries a reputation for being slower and less efficient than other languages. While this is less noticeable when performing compression on a singular image, a dataset containing the frames of an entire video would take exponentially longer in Python than in more efficient languages. Adding to this inefficiency, Python has no inherent image import or processing capabilities, instead relying on libraries to perform the compression described in this paper. As R features built-in matrix manipulation (including a built-in function for performing SVD) we believe that R would be a useful language for the processes described within our next paper. Specifically, we plan on creating a described matrix transformation within both languages, followed by a comparison of the differences between Python and R and a conclusion on which language was more efficient for the process.

References

[1] ActiveLoop, ed. *Singular Value Decomposition (SVD)*. [Online; accessed 17-October-2024]. 2024. URL: https://www.activeloop.ai/resources/glossary/singular-value-decomposition-svd/#:~:text=Some%20limitations%20of%20SVD%20include,time%20or%20resource%2Dconstrained%20settings..

[2] Albrecht Böttcher and David Wenzel. "The Frobenius norm and the commutator". In: *Linear algebra and its applications* 429.8-9 (2008), pp. 1864–1885.

[3] L. Brunton and J. N. Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, 2022.

[4] Mathematics Stack Exchange, ed. *The SVD solution to linear least squares linear system of equations*. 2016. URL: https://math.stackexchange.com/questions/1816364/the-svd-solution-to-linear-least-squares-linear-system-of-equations.

[5] "Image denoising". In: *IEEE Xplore* (2015). URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7067415.

[6] "Intrusion detection". In: *ScienceDirect* (). URL: https://pdf.sciencedirectassets.com/272990/1-s2.0-S1571066105X02237/1-s2.0-S15710661050522%5B...%5D29t&ua=0f15570653555c065c5851&rr=8d432c373c7e79a4&cc=us.

[7] Zecheng Kuang. "Honors Thesis". University of California, San Diego, 2018. URL: https://math.ucsd.edu/sites/math.ucsd.edu/files/undergrad/honors-program/honors-theses/2017-2018/Zecheng_Kuang_Honors_Thesis.pdf.

[8] Zhipeng Lei, Feiyu Wang, and Chuanyang Li. "A denoising method of partial discharge signal based on improved SVD-VMD". In: *IEEE Transactions on Dielectrics and Electrical Insulation* 30.5 (2023), pp. 2107–2116.

[9] *Listing applications of the SVD*. https://mathoverflow.net/questions/408504/listing-applications-of-the-svd.

[10] lzhangstat. *Proof of Eckart-Young Theorem*. Ed. by medium.com. [Online; posted 09-June-2024]. June 2024. URL: https://lzhangstat.medium.com/the-proof-of-eckhart-young-theorem-8aad76e5f4be.

[11] D. Miczek. *SVD Image Compression, Explained*. Ed. by dmicz devblog. [Online; posted 17-December-2023]. Dec. 2023. URL: https://dmicz.github.io/machine-learning/svd-image-compression/.

[12] Zeinab Sharifi, Mansoor Rezghi, and Mahdi Nasiri. "New algorithm for recommender systems based on singular value decomposition method". In: *ICCKE 2013*. IEEE. 2013, pp. 86–91.

[13] Wikipedia, ed. *2014 Forest Hills Drive*. [Online; posted 20-September-2024]. Sept. 2024. URL: https://en.wikipedia.org/wiki/2014_Forest_Hills_Drive.

[14] Wikipedia, ed. *Data compression ratio*. [Online; posted 25-April-2024]. Apr. 2024. URL: https://en.wikipedia.org/wiki/Data_compression_ratio.

[15] Wikipedia, ed. *Low-rank approximation*. [Online; accessed 17-October-2024]. 2024. URL: https://en.wikipedia.org/wiki/Low-rank_approximation.

[16] Wikipedia, ed. *Matrix (mathematics)*. 2024. URL: https://en.wikipedia.org/wiki/Matrix_(mathematics).

[17] Wikipedia, ed. *Singular value decomposition*. [Online; accessed 19 September 2024]. 2024. URL: https://en.wikipedia.org/wiki/Singular_value_decomposition.

[18] Xinglin Zhang et al. "Anomaly detection of complex magnetic measurements using structured Hankel low-rank modeling and singular value decomposition". In: *Review of Scientific Instruments* 93.4 (2022).

10 Biography

- **Ainsley Braunscheidel** is a Statistics and Data Science undergraduate student at University of Colorado, Boulder. She has experience in coding languages such as R and Python, as well as various techniques for data analysis.
- **Divya Nallawar** is a Data Science graduate student at University of Colorado, Boulder. She has experience in software development, data analysis, and machine learning through different roles at Amdocs, StateSimplify LLC, and the University of Colorado Boulder.
- **Luke Petet** is a Data Science undergraduate student at the University of Colorado, Boulder. He has experience in data analysis with an emphasis in the application of data analysis on audio engineering.
- **Reza Naiman** was born in Kabul, Afghanistan, and moved to Englewood, Colorado, in 2016. After graduating from Englewood High School, he continued his education at the University of Colorado Boulder. He is a senior in the Statistics and Data Science program with a minor in Computer Science. He has completed two internships with Frontier Technologies Defense as a data analyst and software developer. His interest is in continuing to work in the defense industry or government agencies by applying the skills he learned from his education at the University of Colorado Boulder.