




Application of Singular Value Decomposition technique for compressing images

Khadeejah James Audu

Department of Mathematics, Federal University of Technology, Minna, Nigeria.

*Correspondence: k.james@futminna.edu.ng

Abstract	Article History
<p>Image processing is becoming increasingly important as imaging technology has advanced. A storage constraint might occur even when image quality is an influential factor. This means finding a way to reduce the volume of data while still retaining quality, since compactable systems and minimal space are more desirable in the current computing field. An image compression technique that is frequently used is singular value decomposition (SVD). SVD is a challenging and promising way to loosely compress images, given how many people use images now and how many different kinds of media there are. SVD can be employed to compress digital images by approximating the matrices that generate such images, thereby saving memory while quality is affected negligibly. The technique is a great tool for lowering image dimensions. However, SVD on a large dataset might be expensive and time-consuming. The current study focuses on its improvement and implements the proposed technique in a Python environment. We illustrate the concept of SVD, apply its technique to compress an image through the use of an improved SVD process, and further compare it with some existing techniques. The proposed technique was used to test and evaluate the compression of images under various r-terms, and the singular value characteristics were incorporated into image processing. By utilization of the proposed SVD technique, it was possible to compress a large image of dimension 4928 x 3264 pixels into a reduced 342 x 231 pixels with fair quality. The result has led to better image compression in terms of size, processing time, and errors.</p>	<p>Received: 20/06/2022 Accepted: 17/08/2022 Published: 19/08/2022</p> <p>Keywords Image compression; Singular Value Decomposition; Compressed image; Matrix; Role of singular values</p> <p>License: CC BY 4.0*</p>  <p>Open Access Article</p>
<p>How to cite this paper: Audu K.J. (2022). Application of Singular Value Decomposition technique for compressing images. <i>Gadau J Pure Alli Sci, 1(2): 82-94.</i> https://doi.org/10.54117/gjpas.v1i2.21.</p>	

1.0 Introduction

An image is commonly represented as an X-Y matrix, with the value of each pixel corresponding to its intensity. In general, the higher the value, the brighter the pixel, and the lower the value, the darker the pixel. In a color image, separate channels can be used for each color. Images are generally represented as such, but they are actually stored in a completely different manner. Assume we have a 12 mega pixel color image that is 4000 pixels wide and 3000 pixels deep, This means that we have to store 12 million values for each

color channel, bringing the total number of value to 36 million (4000 x 3000 x 3). If these values are assumed to be stored as single byte integers, then a 36MB file is expected, but such large file can be reduced to a lower size by compressing the image. (Lalnunhlima *et al.*, 2021). Several signals, images and videos are captured everyday around the world and they have to be stored, compressed, transferred and processed with all kinds of algorithms that are utilized for such purposes. Techniques related to images compression are of utmost importance in recent developments. The

traditional approach to processing images and signals is based on the Fourier transform/analysis on both continuous and discrete data (Krasnala *et al.*, 2017; Vasanth *et al.*, 2019; Rasheed *et al.*, 2020). Currently, advances in variational image compression have supplemented these techniques as well (Ericsson *et al.*, 2017; Pandey and Singh-Umrao, 2019; Hilles and Shafii, 2019; Compton and Ernstberger, 2020; Przystupa *et al.*, 2021; Zhou *et al.*, 2022). As the volume of experimental and computational data increases rapidly, there is a growing need for creative algorithms to characterize the data such that the essential features can be easily extracted. Large dimension of images can be compressed through the use of SVD into a much smaller space losing very small data along the way (Bonaccorso and Incognito, 2020). SVD is a vital technique used in Data science, Machine learning, Sciences, Statistics, Computer sciences and Engineering. It is actually useful and helps us to accomplish many physical things in the real world. For instance, SVD can be used to solve linear systems of equations $Ax = b$ for non-square matrices, especially in linear regression models. It can also be applied to data science, in the case of a matrix that is potentially very large and using the SVD, one can compress it into a much smaller space losing very small data along the way (Brunton and Kutz, 2019). It can also be applied to solve tensor problems (Qin *et al.*, 2022).

Many applications necessitate the transmission and storage of images. The transmission and storage costs are lower when the image is smaller. As a result, data compression is frequently required in order to reduce the amount of storage space required by an image (Kumar and Parmar 2020; Xu *et al.*, 2021; Intawichai and Chaturantabut, 2022; Li and Wu, 2021; Asnaoui, 2020). Recently, some authors have carried out studies on image compression in different perspective. Sandhu and Singh (2018) discussed how the SVD can be applied to digital image processing through Matlab. The SVD serves as basis for principal component analysis used in most statistical computations. Rani *et al.* (2021) proposed a hybrid SVD technique which was implemented on an X-ray and MRI images and they discovered an improved compression ratio of images. Olajide and Kolawole (2021) compared the performance of the SVD technique with that of QR decomposition and observed that SVD technique is quite effective in tackling least square and linear

system problems than the QR technique. Again, Abdillah *et al.* (2021) conducted some analysis on image compressing through SVD, Haar-wavelet and coiflets techniques and observed that SVD technique gives higher compression ratio compared to the remaining techniques used. Afrose *et al.* (2015) employ an hybridization by combination of different image processing techniques to compress images and Munshi *et al.* (2021) utilized a supervised learning procedure known as the K-means clustering techniques to process and compress images.

The main focus of this research paper is geared towards improvement on the SVD and examining the effect of singular values in application of the proposed SVD technique to image processing. Specifically in the aspect of compression of large and complex images. Also, this technique takes a matrix X and expresses it as a product of three special matrices P , Σ and Q^T that results into $P\Sigma Q^T$, where P and Q^T are unitary matrices. The experiments are performed using different singular values of term r , with expansion of the outer product of matrix image (X) for compression of digital images. The programming of Python was employed to calculate the improved SVD of the desired image matrix and equally obtain the compressed images using the desired low rank approximation.

2.0 Theory Related to SVD

In linear algebra, an eigenvalue decomposition of any matrix is usually represented as

$$M_{m \times m} = U \Lambda U^{-1} \quad (1)$$

which can only be used for diagonalizable square matrices, therefore it is only a kind of small class of matrices but the SVD basically generalizes the idea of eigenvalue decomposition of complex matrices. Hence this generalized decomposition allows researchers to use a lot of matrices with different dimensions (Lay *et al.*, 2016). The SVD of any given $X_{m \times n}$ matrix is decomposed or factored into smaller matrices given by the expression

$$X_{m \times n} = P_{m \times m} \Sigma_{m \times n} Q_{n \times n}^T \quad (2)$$

and illustrated as

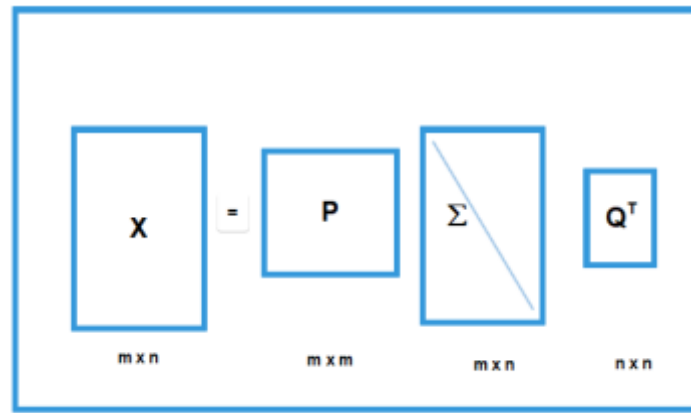


Figure 1: Schematic diagram of SVD

where m is the number of rows and n is the number of columns associated with the matrix X , matrix P is an $m \times m$ orthogonal matrix

$$P = (p_1, p_2, \dots, p_r, p_{r+1}, \dots, p_m) \tag{3}$$

with column vectors p_j , for $j = 1, 2, \dots$ that forms unitary sets such as $P^T P = P P^T = I_{m \times m}$

Also matrix Q^T is an $n \times n$ orthogonal matrix

$$Q = (q_1, q_2, \dots, q_r, q_{r+1}, \dots, q_n) \tag{4}$$

with column vectors q_j , for $j = 1, 2, \dots$ that forms unitary sets such as $Q^T Q = Q Q^T = I_{n \times n}$

It is to note that P and Q^T are orthonormal matrices which means that they contain vectors which are linearly independent to each other. They also contains

singular vectors both left and right of the original matrix X . The sigma matrix (Σ) is a diagonal matrix containing the singular values

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sigma_r & \vdots & \dots & 0 \\ 0 & 0 & 0 & 0 & \sigma_{r+1} & \dots & \vdots \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \vdots & \dots & \sigma_m \\ 0 & 0 & \dots & 0 & \vdots & \dots & 0 \end{pmatrix}$$

for σ_j , for $j = 1, 2, 3, \dots, m$ and σ_j 's are known as the singular values of the main matrix. It can be proved that

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \dots \geq \sigma_r > 0 \text{ and}$$

$$\sigma_{r+1} = \sigma_{r+2} = \sigma_{r+3} = \dots = \sigma_m$$

An interesting part of the SVD technique is finding a good approximation of a given matrix through one of low rank. The SVD can be utilized to obtain low rank

approximation of any particular matrix. Based on this fact, it has so many applications in areas like image compression, voting tendencies, facial recognition,

and so on. Matrix decomposition is very useful for reducing or compressing big data by extracting the most important feature in the data set, that is to say

the dominant or most important feature that are strongly correlated to the given matrix. This decomposition approach can be expressed as;

$$\begin{aligned}
 X &= \begin{pmatrix} \vdots & \vdots & \vdots \\ p_1 & p_2 & \dots & p_m \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \ddots \end{pmatrix} \begin{pmatrix} \dots & q_1^T & \dots \\ q_1^T & & \\ \vdots & & \\ \dots & q_n^T & \dots \end{pmatrix} \\
 &= \begin{pmatrix} \vdots & \vdots & \vdots \\ p_1 & p_2 & \dots & p_m \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \dots & \sigma_1 q_1^T & \dots \\ & \sigma_1 q_2^T & \\ & \vdots & \\ \dots & \sigma_n q_n^T & \dots \end{pmatrix} \\
 &= \sigma_1 p_1 q_1^T + \sigma_2 p_2 q_2^T + \sigma_3 p_3 q_3^T + \dots + \sigma_r p_r q_r^T
 \end{aligned} \tag{5}$$

(Li and Wu, 2021) where r is the rank of A , the σ after this are zeros and thus SVD breaks down any given matrix into sum of r rank-1 matrices. Since σ 's are in decreasing size and p 's and q 's are unit vectors, then the rank-1 matrices can be written in decreasing size too. Now, suppose a low-rank

approximation of a given matrix is needed, then one can just stop at the sum after few terms. The initial term informs us about the single direction that becomes most magnified by the matrix A , the second term tells the direction that becomes magnified second most and so on (Kumar *et al.*, 2019). This is clearly illustrated as

$$\begin{array}{ccccccc}
 \sigma_1 p_1 q_1^T & + & \sigma_2 p_2 q_2^T & + & \sigma_3 p_3 q_3^T & + \dots + & \sigma_r p_r q_r^T \\
 \text{Most important} & & \text{less important} & & \text{important} & & \text{and so on}
 \end{array}$$

2.1 Computation of SVD for Matrices

Before applying the SVD technique to digital image compression, first we shall demonstrate how SVD is computed using a 2×3 small matrix X ;

$$X = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{6}$$

and gradually follow a series of steps to factor matrix X into the different components of $P\Sigma Q^T$

Step One: Construct the matrix $X^T X$

We commenced the procedure by constructing $X^T X$ for the desired matrix and then carry out some matrix multiplication as thus;

$$X^T X = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{7}$$

This process results into a square matrix with 3×3 dimension

Step Two: Obtain the Eigenvalues of matrix $X^T X$

$$\begin{aligned} \det(X^T X - \lambda I) &= \begin{vmatrix} 2 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix} - \lambda \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 2-\lambda & 1 & 1 \\ 1 & 1-\lambda & 0 \\ 1 & 0 & 1-\lambda \end{vmatrix} \\ &= \lambda^3 - 4\lambda^2 + 3\lambda \\ &= (\lambda = 0), (\lambda = 1), (\lambda = 3) \end{aligned} \tag{8}$$

Setting the determinant to equals zero, the characteristic equation is solved for λ to obtain the values of λ to be $\lambda = 0$, $\lambda = 1$ and $\lambda = 3$. Hence, reordering the eigenvalues in decreasing order, we have $\lambda_1 = 3$, $\lambda_2 = 1$ and $\lambda_3 = 0$.

Step Three: Form the matrix Q^T

After obtaining the eigenvalues in step two, we compute the eigenvectors that corresponds to the eigenvalues and then normalize them to form the Q^T matrix. Generally, eigenvectors are computed through the use of $X^T X - \lambda I$ matrix and further simplification of it for every eigenvalues. For instance, taking the eigenvalue $\lambda_1 = 3$, we get;

$$(X^T X - \lambda_1 I) = X^T X - 3I = \begin{pmatrix} 2-3 & 1 & 1 \\ 1 & 1-3 & 0 \\ 1 & 0 & 1-3 \end{pmatrix} = \begin{pmatrix} -1 & 1 & 1 \\ 1 & -2 & 0 \\ 1 & 0 & -2 \end{pmatrix} \tag{9}$$

So, solving the equation $(X^T X - 3I)z_1 = 0$ to get the first eigenvector z_1 , results into the column vector as;

$$z_1 = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} \tag{10}$$

Next the eigenvector is normalized through the process of dividing z_1 by its magnitude to form a new vector as;

$$\bar{q}_1 = \frac{z_1}{\|z_1\|} = \frac{1}{\sqrt{6}} \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{pmatrix} \tag{11}$$

Similarly, we apply the above procedure to each eigenvalues to give a complete set of the eigenvectors that is required to form the Q^T matrix

$$Q = [\bar{q}_1, \bar{q}_2, \bar{q}_3] = \begin{pmatrix} \frac{2}{\sqrt{6}} & 0 & \frac{-1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \end{pmatrix} \tag{12}$$

Consequently, we can easily obtain the transpose of Q , by interchanging the columns of (12) with their corresponding rows. As a result of that, we obtain the matrix

$$Q^T = \begin{pmatrix} \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix} \quad (13)$$

Step Four: Form the Σ matrix

To form the sigma (Σ) matrix, the singular values which are nonzero are listed s_k where $s_k = \sqrt{\lambda_k}$ in a decreasing order along the main diagonal of sigma. Then additional columns and rows are added to make up the same number of dimension as the original dimension of matrix A in sigma (Σ). That is to say, it is only required to keep the non-zero eigenvalues and thus the sigma (Σ) matrix is formed as;

$$\Sigma = \begin{pmatrix} \sqrt{3} & 0 & 0 \\ 0 & \sqrt{1} & 0 \end{pmatrix} = \begin{pmatrix} \sqrt{3} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad (14)$$

Comparing equations (2) and (14), it is observed that sigma has same dimension as the original matrix.

$$\vec{p}_1 = \frac{1}{\sqrt{s_1}} X \vec{q}_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \text{ and } \vec{p}_2 = \frac{1}{\sqrt{s_1}} X \vec{q}_2 = \frac{1}{\sqrt{1}} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad (16)$$

and combining the above column vectors results into matrix P as

$$P = [\vec{p}_1 \quad \vec{p}_2] = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \quad (17)$$

Step Six: Rewrite the X matrix into $X = P\Sigma Q^T$

Finally, using equation (2), we rewrite matrix X in terms of P, Σ and Q^T to obtain

$$X = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \sqrt{3} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \quad (18)$$

Step Five: Form the P matrix

The left singular vector P can be obtained from the formula $XQ = \Sigma P$ or the main formula in (2), Since X is given and we have obtained matrices Q and Σ , then P matrix can be obtained as thus

$$\begin{aligned} X &= P\Sigma Q^T \\ XQ &= P\Sigma \\ P &= \Sigma^{-1} XQ = \frac{1}{\Sigma} XQ \end{aligned} \quad (15)$$

with respect to the singular values and matrix Q and taking the two singular values computed in this example to form the singular vectors of P results into

3.0 Methodology of SVD for Compressing Images

3.1 SVD Technique for Compressing Images

Digital image compression is mainly focused on problems of data reduction needed in representing images. This compression can be done by getting rid of some basic data redundancies, such as inter-pixel redundancy, which is caused by the correlations between pixels, psych visual redundancy, which is caused by data that is ignored because of how people see it, and coding redundancy, which is caused by data that is ignored because of how it is coded. Considering the property of SVD that the rank of any matrix is equal to the number of singular values (nonzero), it follows that we can compress any given data (matrix) through the elimination of small singular values or that of higher low ranks (Cao, 2006).

A digitized image can be thought of as an array of numbers in a large matrix, it could either be in a gray level scale for images that are black and white or in a color level for colored images. Suppose we have an image with a dimension of 3000 x 4000 pixels. This means that we need twelve million (4000 x 3000) numbers to represent this matrix for its storage. The image can be reduced by a ten-term SVD, which allows us to store ten thousand numbers (10 P's), 10 Sigma (Σ)'s and two thousand numbers (10 Q's). The cost of storage reduces to over thirty thousand (30,000) by a reduction ratio of over 550:1. Therefore, this means that we can represent any given matrix with a fewer numbers using the idea of SVD (Singh and Lairenjam, 2020).

Given an image that is transformed into a matrix X , which can be decomposed by SVD technique into some factored matrices like $X = P\Sigma Q^T$. Then by low rank approximation, we might approximate the matrix X by some low rank product such as

$$X \approx P_r \Sigma_r Q_r^T \quad (19)$$

where P_r is the first r column of the P matrix, Σ_r is the first $r \times r$ matrix of Σ and Q_r^T is the first r columns of the Q matrix. Equivalently, the original matrix can be characterized by the expansion of the outer product (Yu *et al.*, 2018)

$$X \approx \sigma_1 p_1 q_1^T + \sigma_2 p_2 q_2^T + \sigma_3 p_3 q_3^T + \dots + \sigma_r p_r q_r^T \quad (20)$$

3.2 Proposed SVD Algorithm

Higher resolution sensors and big data in greater dimension measurement are increasing high-dimensional data. SVD is a great tool for reducing data dimensions. Although it has been observed that computing the SVD on larger datasets can be expensive and time-consuming. The existing SVD is improved to develop a computationally efficient approach for extracting the dominating key features and rank of the SVD from a big dataset. This enhancement is accomplished by constructing a projection data matrix that randomly samples the column space and spans the original data matrix's subspace. The projection matrix was then subjected to techniques such as oversampling and power iterations in order to get the appropriate low rank approximation that may be utilized to compress images. The proposed improved SVD process is described in the flowchart below.

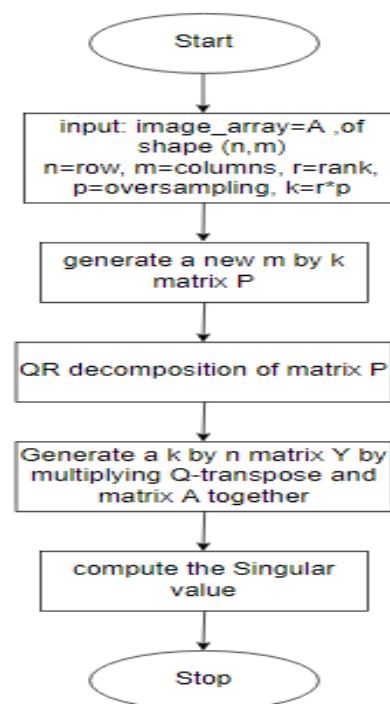


Figure 2: Improved SVD Flowchart

In this paper, we shall apply the above flowchart to compress digital images, test for different r and present the outcome in section four.

3.3 Measures of Image Compression

The SVD performance on image processing can be measured through the computations of quality of the processed image and compression ratio of the desired

image. The compressed factor is obtained through the use of the compression ratio, designated as

$$C_R = \frac{n * m}{r(n + m + 1)} \quad (13)$$

While the mean square error [MSE], employed to determine the quality comparison of the uncompressed image (X) and compressed image (X_r). The MSE is computed by using the equation (Swathi *et al.*, 2017).

$$MSE = \frac{1}{nm} \sum_{y=1}^n \sum_{x=1}^m [f_x(x, y) - f_{x_r}(x, y)]^2 \quad (14)$$

4.0 Experiment and Discussion of Results

Suppose X is a matrix of 3D image of a lion in Figure 1, then it is possible to approximate the image as the product of a few columns of r and a few term of rows. This would result in a compression because instead of storing all the n x m rows and columns, one would just need to store r x r columns and rows, thereby reducing the dimension storage and obtaining the desired result. Figure 3 displays the original photo of a Lion with dimension 4928 x 3264 pixels, utilized in performing the image compression experiment.



Figure 3: A sample image utilized for testing image compression

The sample image in Figure 3 is compressed via SVD, proposed improved SVD and K-means clustering techniques and produces the following compressed images in Figures 4, 7 and 8:

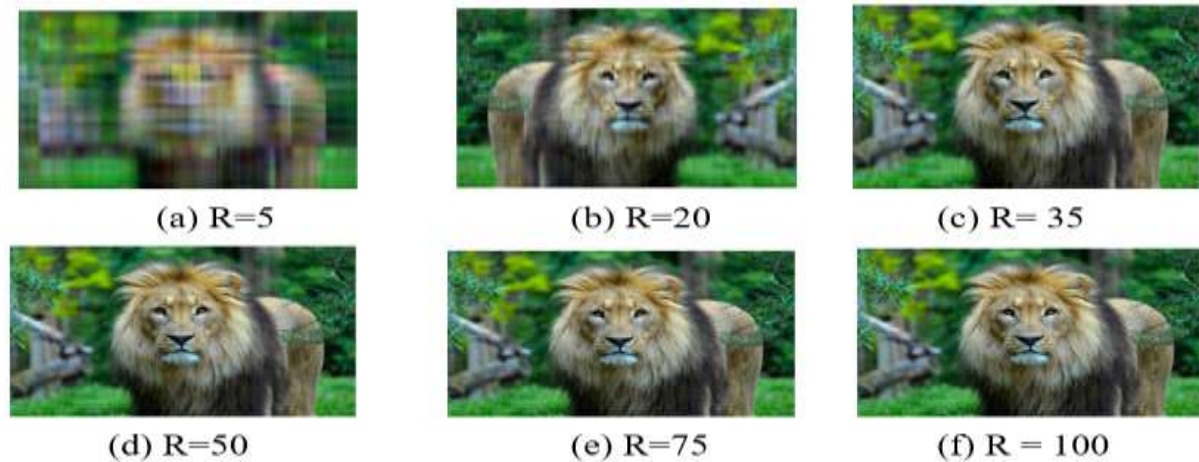


Figure 4: Samples of the proposed SVD compressed images with different R

Figure 4 illustrates some of the images utilized in the system tests for various R terms, where Figure (a) displays the outcomes of the image reconstruction utilizing five singular values, (b) presents the compressed image with twenty singular values, (c) demonstrates that of thirty-five singular values, and so

forth. In such examples, the observation is as follows: It was discovered that when the rank is less than 35,

the pictures turn out to be blurred and as the number of singular values rises, we have a more thorough understanding of the original image. At rank 75, the image quality noticeably improves, and at rank 100 of dimension 342 x 231 pixels, the image quality is essentially the same as the reference image, that is, it appears almost identical to the original. A summary of the findings is shown in Table 1, with the storage space measurement for the images that were tested.

Table 1: Results of Image Compression for the Proposed SVD

Singular values (r)	Storage space (kb)	Compression Ratio (%)	Mean Square Error
5	100	0.25468	734.93
20	126	1.01871	280.97
35	138	1.78275	181.64
50	145	2.54679	134.37
75	151	3.82078	92.30
100	154	5.09357	68.18
3000(original)	2.26M	1	

The compression result displayed in Table 1, reveals that a higher compression ratio can be achieved by using a smaller singular value (r). For improved image quality, a smaller MSE measurement is used, and the recreated images are more comparable to the originals, despite requiring a larger amount of storage space. Also, Figures 3 and 4 show how the singular values from the computed proposed enhanced SVD of the image matrix have a big effect on the compression output.

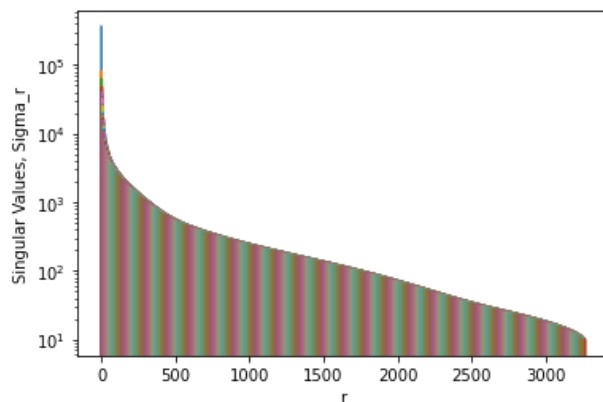


Figure 5: The effect of Singular values for the Image Compression

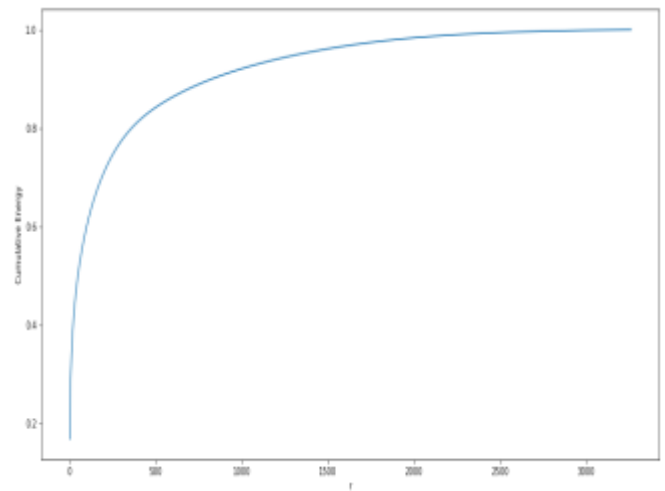


Figure 6: Cumulative Sum of the Singular Values

Figure 5 reveals that the first few modes contain a significant amount of the total energy in the system, that captures a large fraction of the compression energy. The lower singular values are much more effective in compressing the reference image than the remaining ones, and what this means is that one can keep only the first few singular values and neglect all of the remaining single values while truncating. Since some information is neglected during image decoding, the decoded image will not be identical to the original one. However, the compressed image should look nearly as good as the sample (original) image when a reasonable compression rate is used. Similarly, the cumulative plot depicted in Figure 6 reveals that retaining only the first few singular values captures approximately 30% of the energy, which then increases rapidly to 70% or 80% with the first 70 or 100 modes (r). So, even though the sample image is of a high resolution, it is not necessary to retain all the information. It can be compressed using improved SVD.

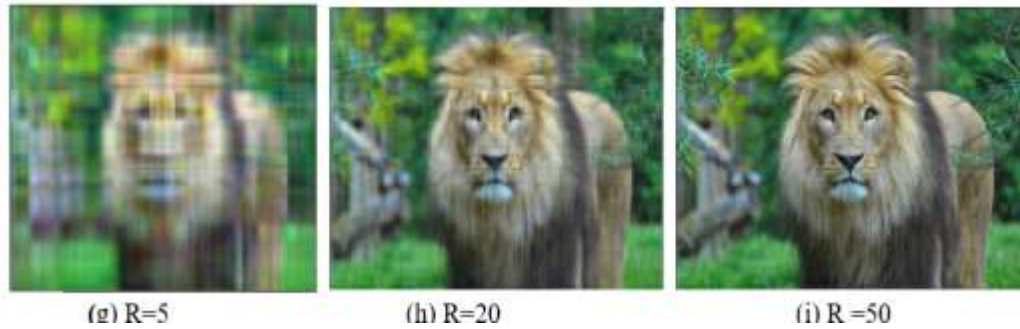


Figure 7: Samples of the SVD compressed images with different R

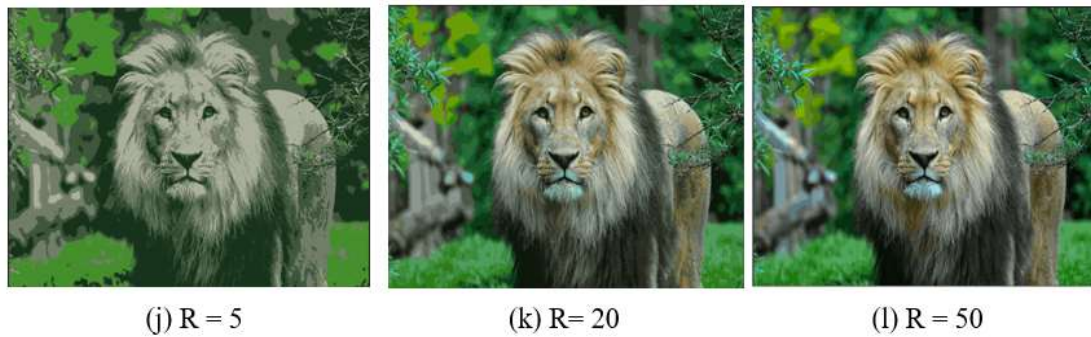


Figure 8: Samples of K-means Clustering compressed images

Table 2: Comparison Results for Compressed Images

Singular/K values	Storage space (kb)	Compression Ratio (%)	Mean Square Error	CPU Time
SVD Technique				
5	424	0.25468	741.20	3min 11sec
20	513	1.01871	297.23	4min 5sec
50	602	2.54679	139.12	4min 27sec
K-Means Clustering Technique				
5	334	0.7640	451.19	3min 2sec
20	484	1.5468	139.76	21min 33sec
50	528	2.8564	66.57	1h 7min 10sec
Proposed Technique				
5	100	0.1654	734.93	6.12sec
20	126	1.01871	280.97	7.82sec
50	145	2.54679	734.93	13.2sec
3000 (original)	2.26MB	1		

Table 2 shows the comparison between the proposed SVD and some existing techniques (SVD and K-means clustering). In terms of the compression ratio, it can be observed that at $k=5, 20,$ and $50,$ the ratios of SVD and K-means are higher than that of the proposed technique. For the mean square error, the proposed technique has a lower error compared to SVD. However, the K-mean has the lowest error among the techniques. With the computational time, the proposed technique compresses faster than both K-means and SVD techniques and has the lowest size of the compressed images.

5.0 Conclusion

This research has been able to achieve better compression of images in terms of size, computational time and less error as shown in Table 2. We presents an analysis of digital image compression through the use of a SVD technique with the aid of Python programming. By utilizing the improved SVD technique on such images, a greater rate with respect to the compression was obtained while maintaining the quality of the reference image. During the process of compression, some of the initial image's information is basically neglected by the role of the singular values which was introduced in this study. The image quality degrades in direct proportion to the amount of information that is neglected. When it comes to file size, there is a trade-off between file size and the image quality. However, compressing images through the use of SVD is a profitable trade because it reduces the size of the original file while maintaining the quality of the image as it is perceived. The primary reason for this is that the components that are discarded are intended to be the components that are less obvious. Results from the experiment tell us that compression design becomes more easily visible as the compression rate increases and it is hard to distinguish an uncompressed image compared to a compressed one. Therefore, some of the compressed images are not distinguishable from the reference (original) image because they utilize a reduced percentage of space storage. The role of the singular vectors (unitary matrices) in compressing images as well as the relationship between the SVD and correlation in the image matrix will be explored in future work.

Declarations

Ethics approval and consent to participate

Not Applicable

Availability of data and material

Not Applicable.

Competing interests

The author affirms that she is not aware of any potential conflicts of interest that may have been a factor in the work that was reported in this paper.

Funding

There was no funding for the current report.

References

- Abdillah, M. Z., Yudhana, A. and Fadlil, A. (2021). Compression analysis using coiflets, haar wavelet and SVD methods. *Journal of Informatica*, 9(1), 43-48.
- Afrose, S., Jahan, S., and Chowdhury, A. (2015). A hybrid SVD-DWT-DCT based method for image compression and quality measurement of the compressed image. *International Conference on Electrical Engineering and Information Communication Technology*, 1-4. doi: 10.1109/ICEEICT.2015.7307442.
- Asnaoui, K. E. (2020). Image compression based on block SVD power method. *Journal of Intelligent Systems*, 29(1), 1345-1359. <https://doi.org/10.1515/jisys-2018-0034>.
- Bonaccorso, K. and Incognito, A. (2020). SVD. *Honor Theses*, 1-18. <https://digitalcommons.coastal.edu/honors-theses/353>. Accessed 09/02/2022.
- Brunton, S. L. and Kutz, J. N. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, Cambridge University Press, New York, USA.
- Cao, Lijie. (2006). SVD applied to digital image processing. Division of Computing Studies, Arizona State University Polytechnic Campus, Mesa, (2006), 1-15.
- Compton, E. A. and Ernstberger, S. L. (2020). SVD: applications to image processing. *Journal of Undergraduate Research*, 17, 99-105.
- Ericsson, N. B., Brunton, S. L., and Kutz, J. N. (2017). Compressed Singular value decomposition for image and video processing. *International Conference on Computer Vision and Workshops (ICCVW)*, 1880- 18888. doi.101109/ICCVW.2017.222.

- Hilles, S. M. S., and Shafii, M. S. (2019). Image compression and encryption technique. *International Journal of Data Science and Research*, 1(2), 0-6.
- Intawichai, S. and Chaturantabut, S. (2022). A missing data reconstruction method using an accelerated least-squares approximation with randomized SVD. *Algorithms*, 15(190), 1-16. <https://doi.org/10.3390/a15060190>.
- Krasmala, R., Budimansyah, A., and Lenggana, U. T. (2017). Image compression by combining Discrete Cosine Transform (DCT) method and huffman algorithm. *Journal Online Informationa.*, 2(1), 1- 10. doi: 10.15575/join.v2i1.79.
- Kumar, P., and Parmar, A. (2020). Versatile approaches for medical image compression. *Procedia Computational Sciences*, 167, 1380-1389. doi: 10.1016/j.procs.2020.03.349.
- Kumar, R., Patbhaje, U., and Kumar, A. (2019). An efficient technique for image compression and quality retrieval using matrix completion, *Journal of King Saud Univ. - Computational Information and Sciences*, doi: 10.1016/j.jksuci.2019.08.002.
- Lalnunhlina, J., Johnson T. D., and Lalruatzuala. (2021). Digital image compression using DCT and SVD”, *International Journal of Recent Advances in Multidisciplinary Topics*, 2(6), 88-92.
- Lay, D. C., Steven, R. L., and McDonald, J. (2016). *Linear Algebra and Its Applications*. Boston, Pearson. 5th edition and published by Pearson Education Press, Boston, USA.
- Li, K. and Wu, G. (2021). A randomized generalized low rank approximations of matrices algorithm for high dimensionality reduction and image compression. *Numerical Linear Algebra with Applications*, 1-24. doi: 10.1002/nla.2338.
- Munshi, A., Alshehri, A., Alharbi, B., AlGhamdi, E., Banajjar, E., Albogami, M., and Alshanbari, H. S. (2021). Image compression using K-mean clustering algorithm. *International Journal of Computer Science and Network Security*, 21(9), 275-280. <https://doi.org/10.22937/IJCSNS.2021.21.9.36>
- Olajide, I. A. and Kolawole, M. O. (2021). Examination of QR decomposition and the SVD methods, *Journal of Multidisciplines Engineering Science Studies*, 7(4), 3834-3839.
- Pandey, J. P., and Singh-Umrao, L. (2019). Digital image processing using singular value decomposition. *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE)*.
- Przystupa, K., Beshley, M., Hordiichuk-Bublivska, O., Kyryk, M., Beshley, H., Pyrih, J., and Selech, J. (2021). Distributed SVD method for fast data processing in recommendation systems. *Energies*, 14(2284). 1-24.
- Qin, Z., Ming, Z., and Zhang, L. (2022). SVD of third order quaternion tensors. *Applied Mathematics Letters*, 123(107597)
- Rani, M. L. P., Lavanya, V. and Sasibhushana Rao, G. (2021). Performance analysis of a hybrid method for medical image compression. *Turkish Journal of Computer and Mathematics Education*, 12(14): 4478- 4485.
- Rasheed, M. H., Salih, O. M., Siddeq, M. M., and Rodrigues, M. A. (2020). Image compression based on 2D Discrete Fourier transform and matrix minimization algorithm, *Array*, 6, 100024. <https://doi.org/10.1016/j.array.2020.100024>.
- Sandhu, K. and Singh, E. M. (2018). Image Compression Using SVD. *Journal of Latest Research in Science and Technology*, 7(11), 5-8.
- Singh, Y. S. and Lairenjam, B. (2020). SVD. *Journal of Emerging Technologies and Innovative Research*, 7(11), 180-182.
- Swathi, H. R., Shah-Sohini, S., and Gopichand, G. (2017). Image compression using singular value decomposition. *IOP Conference Series: Materials Science and Engineering*, 263, 042082 doi:10.1088/1757-899X/263/4/042082.
- Vasanth, P., Rajan, S. and Fred, A. L. (2019). An efficient compound image compression using optimal discrete wavelet transform and run length encoding techniques. *Journal of Intelligent Systems*. 28(1), 87-101.
- Xu, S., Zhang, J., Bo, L., Li, H., Zhang, H., Zhong, Z., and Yuan, D. (2021). Singular vector sparse reconstruction for image compression.

Computers and Electrical Engineering,
91(2021), 107069.
<https://doi.org/10.1016/j.compeleceng.2021.107069>.

Yu, W., Gu, Y., and Li, Y. (2018). Efficient randomized algorithms for the fixed-precision low-rank matrix approximation. *Siam Journal of Matrix Analysis Application*, 39, 1339–1359.

Zhou, Y., Wang, S., Wu, T., Feng, L., Wu, W., Luo, J., Zhang, X., and Yan, N. (2022). For-backward LSTM-based missing data reconstruction for time-series Landsat images. *Gisci. Remote Sensing*. 59, 410–430.
<https://doi.org/10.1080/15481603.2022.2031549>.