TAIWANESE JOURNAL OF MATHEMATICS
Vol. xx, No. x, pp. 1–36, xx 20xx
DOI: 10.11650/tjm/240306

A Refinement of Approximate Invariant Subspaces of Matrices Based on SVD in High Dimensionality Reduction and Image Compression

Peter Chang-Yi Weng* and Frederick Kin Hing Phoa

Abstract. The techniques of high dimensionality reduction are important tools in machine learning and data science. The method of singular value decomposition (SVD) is a popular method in dimensionality reduction and image compression. However, it suffers from heavily computational overhead in practice, especially for images with high-resolution. In order to achieve the efficiency and the accuracy, we propose a refinement of approximate invariant subspaces of matrices (REIS) algorithm based on SVD. The theoretical contribution of our paper is threefold. Firstly, we describe the properties of the SVD of the matrices and discuss how to apply SVD to do image compression. Secondly, we introduce the method of REIS based on SVD for image compression in the high-resolution images. The core of REIS is adapted to large and real matrices in $\mathbb{R}^{n\times n}$, through some nonsymmetric algebraic Riccati equations or their associated Sylvester equations via Newton's method. Thirdly, some measurement tools are provided such as compression ratio, mean square error, peak signal to noise ratio and structural similarity index to compare the performance of the compression factors and the quality of the compressed images. Numerical examples for testing some real world image sets are presented to illustrate the feasibility of our proposed algorithm.

1. Introduction

There are many applications about digital images in our daily life such as medicine, ecommerce, astronomy, and remote sensing [15, 20, 26, 28] and it led to some problems about sharing and storage of enormous amounts of digital image data such as transmission slow and storage expensive. Therefore, we need a method to reduce the amount of datum to represent these digital images, image compression, an important area in the digital image processing deals with techniques for reducing the storage of datum required for saving digital images or the bandwidth required for transmitting it [17].

Received September 9, 2023; Accepted March 29, 2024.
Communicated by Eric Chung.
2020 *Mathematics Subject Classification.* 35A01, 65L10, 65L12, 65L20, 65L70.
Key words and phrases. refinement of approximate invariant subspace, singular value decomposition, high dimensionality reduction, high-resolution image compression, nonsymmetric algebraic Riccati equation, Newton's method.
*Corresponding author.

The image compression technique has a lot of applications in various fields such as medical imaging, web applications, facsimile and security industry, satellite imagery and object recognition. The goal of image compression deals with reducing the number of bits in order to represent an image via removing irrelevance and redundancy of the image data, therefore, it can optimize the storage space and increase the transmission rate.

In general, image compression is broadly classified into two categories: lossless and lossy [21, 39]. A lossless compression retains the full information with no loss of data to reconstruct the original image. The lossless compression is useful for exact reconstruction of images and the methods include entropy coding, Huffman coding [29], bit-plane coding and run-length coding. The other compression called lossy compression is another type of image compression technique. Some of information in the image can be discarded without greatly changing the appearance of the image and the size of the image is drastically reduced. The methods for lossy compression are discrete cosine transform [5, 22], discrete wavelet transform, transform coding and fractal compression [38].

For large-sized image data obtained in some applications of remote sensing and multimedia, the storage space and transmission bandwidth become important and are reliant on image compression techniques [30]. The objective of image compression is to reduce number of bits for large-sized image data in order to speed up the communication. One of the efficient method in image compression, singular value decomposition (SVD), a powerful tool widely used in image processing and data hiding. The SVD factorizes a real matrix into three component matrices, called left singular vectors, singular values in diagonal, and right singular vectors [8, 9, 24]. During factorization, some of the singular values are removed for reconstruction and it results in the compression of the image. A brief review of some important contributions in the image compression schemes and their applications in image processing including SVD method and its variations is presented in the related work in Section 2. However, the regular SVD method can not be utilized in large-sized image data, especially high-resolution image.

In this paper, we propose the refinement of an approximate invariant subspace for large-scale matrix in large-sized image data that employs the concept of the SVD for image compression. The estimate of an invariant subspace for large-scale matrix can be refined through the solution of the nonsymmetric algebraic Riccati equation (NARE). We apply Newton's method to the NAREs and their associated sylvester equations are derived. The crux of the method is the inversion of some well-conditioned unstructured matrices via the efficient and stable inversion of the associated structured but near-singular matrices. Furthermore, we provide image compression measures such as compression ratio, mean square error and peak signal to noise ratio to evaluate the quality of a compressed image. Our numerical results show that our algorithm can select an appropriate number

of singular values which guarantees the image quality.

This paper is organized as follows. In Section 2, we briefly introduce a review of previous related work including lossy and lossless compression, especially SVD and its variations. In Section 3, we outline the concept of the SVD including properties of the SVD, image compression using SVD and image compression measures. Section 4 describes the proposed lossy image compression technique, which is the refining estimate of the invariant subspace for large-scale matrix. We apply the efficient method to solve NAREs and get an invariant subspace, then reconstruct the compressed matrix based on the SVD method. Our numerical results in Section 5 demonstrate the effectiveness of the proposed lossy image compression using the refinement of the approximate invariant subspace based on SVD. Finally, some conclusions are drawn in Section 6.

To the best of our knowledge, this is the first work to apply the REIS based on SVD in the applications of the image compression, especially large-sized image data compression and high-resolution image data compression.

2. Related work

In recent year, there are a lot of compression methods and their applications in image processing described in the literature. In this section, some of the developed methods are briefly outlined.

In general, there are two categories of the image compression methods: lossy and lossless. In [29], the authors proposed a new lossless method of image compression by decomposing the tree of Huffman technique (LM-DH) and this method can achieve up to 20% compression ratio more than the classical Huffman coding technique. In [33], the authors introduced fast Fourier transform (FFT), scalar quantization (SQ) and entropy encoding to compress satellite images and grayscale pictures with high compression ratio and return with a better quality. In [11], the authors presented block truncation coding (BTC) and the method uses a two-level nonparametric quantizer and preserves the local sample moments. Later in [23], the authors utilized the absolute moment block truncation coding (AMBTC) and the method preserves the higher mean and lower mean of the blocks. Furthermore, AMBTC provides better image quality than BTC at the same bit rate and is faster compared to BTC.

Generally, SVD is a lossy compression technique which achieves compression by using a smaller rank to approximate the original matrix representing an image. Moreover, lossy compression has good compression ratio comparing with lossless compression while the lossless compression provides good quality of compressed images. In [2], the authors proposed a graph coloring technique in the quantization process based on wavelet-SVD and the compression ratio stands between 50%–60% while the PSNR stands between

40–80 dB. In [32], the authors presented SVD and wavelet difference reduction (WDR) and the method is better than JPEG2000 and WDR techniques respectively. In [1], the authors provided a way of image compression based on two mathematic concepts, SVD and moment preserving quantizer BTC (MPQ-BTC) and the proposed algorithm has a higher compression ratio without compromising much on the quality of the image. Later, the authors [3] suggested two improved algorithms: an improvement of the BTC and a new rank of SVD. The first algorithm can overcome the disadvantages of the classical BTC and the second method shows the new rank to approximate an image using the least amount of information. Other lossy compression including the variations of the SVD are described. In [6], the authors applied the k-SVD algorithm to do facial image compression based on sparse and redundant representations. Moreover, the post-processing addition of image deblocking is introduced and the method can create a more appealing and smooth visual facial images. In [31], the authors proposed shuffled SVD (SSVD) and the method needs about 30% fewer singular values and singular vectors. In [4], the authors proposed block SVD power method that overcomes the disadvantages of Matlab's SVD function and the algorithm can provide different degrees of error resilience which gives a better image compression in a short execution time.

3. Singular value decomposition

In this section we discuss the implementation of the SVD in image compression. Firstly, we introduce the theory of the SVD, move to consider properties of the SVD, then discuss image compression using SVD, finally list some image compression measures. The SVD was first introduced independently by Beltrami and Jordan [35]. SVD is a significant topic in linear algebra and has some practical and theoretical applications such as scientific computing, signal processing and automatic control. It is one of the most powerful tools not only in finding the eigenvalues, but also being implemented in image compression and noise removal. The SVD is a factorization of a real or complex matrix. One special feature of SVD is that it can be proformed on any real matrix.

For any real matrix $A \in \mathbb{R}^{m\times n}$, there exist orthogonal matrices $U = [u_1, u_2, \ldots, u_m]$, $u_i \in \mathbb{R}^m$, $V^\top = [v_1^\top, v_2^\top, \ldots, v_n^\top]$, $v_i \in \mathbb{R}^{1\times n}$ and $D = \mathrm{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_p\} \in \mathbb{R}^{m\times n}$ (see [16, p. 70]) such that

$$A = UDV^\top = \sum_{i=1}^{p} \sigma_i u_i v_i^\top, \tag{3.1}$$

where D is a diagonal matrix in which the entries along the diagonal of D are singular values of A, σ_i are the singular values of A and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$, $p = \min\{m, n\}$; $U \in \mathbb{R}^{m\times m}$ is a matrix containing left singular vectors of A; $V \in \mathbb{R}^{n\times n}$ is a matrix

containing right singular vectors of A; U and V are orthogonal matrices such that $UU^\top = VV^\top = I$.

3.1. Properties of the SVD

In this section, we show how to compute U, D and V in the SVD. Firstly, we try to find orthogonal matrix U. Post-multiplying both sides of the equation (3.1) by $A^\top$ demonstrates

$$AA^\top = UD^2U^\top. \tag{3.2}$$

From equation (3.2), it is clear that the orthogonal matrix U is derived from the eigenvectors of $AA^\top$ and D is obtained from the eigenvalues of $AA^\top$ such that $\sigma_i = \sqrt{\lambda_i}$, where λ_i are the eigenvalues of $AA^\top$, $i = 1, 2, \ldots, p$.

Analogously, we have shown that

$$A^\top A = VD^2V^\top,$$

where V is obtained from the eigenvectors of $A^\top A$. The following are some of the important properties of SVD.

(1) The singular values $\sigma_1, \sigma_2, \ldots, \sigma_p$ are unique, however, the orthogonal matrices U and V are not unique;

(2) Since $AA^\top = UD^2U^\top$, U diagonalizes $AA^\top$ and it follows that u_i is the eigenvector of $AA^\top$;

(3) Since $A^\top A = VD^2V^\top$, V diagonalizes $A^\top A$ and it follows that $v_i^\top$ is the eigenvector of $A^\top A$;

(4) The rank of matrix A is equal to the number of its nonzero singular values [25];

(5) If A has rank of r, then $u_1, u_2, \ldots, u_r$ form an orthonormal basis for range space of A, $R(A)$ and $v_1, v_2, \ldots, v_r$ form an orthonormal basis for range space of $A^\top$, $R(A^\top)$.

3.2. Image compression using SVD

Image compression deals with the problem of reducing the amount of data required to represent a digital image and it aims to reduce the number of bits required to represent an image via removing the redundant or irrelevant information in an image. Redundancies can be one of the following types [18]:

(I) Inter-pixel Redundancy: Prediction of pixel values is based on values of its neighboring pixels. The pixels of 2D intensity arrays are correlated spatially that is each pixel is similar to or independent on neighbouring pixels, information is unnecessarily replicated in the representation of the correlated pixels. Inter-pixed redundancy is sometimes called spatial redundancy, inter frame redundancy, geometric redundancy.

(II) Coding Redundancy: It contains variable length code words selected as to match the statistics of the original source. In the case of digital image processing, it is the image itself or the processed version of its pixel values. Examples of this technique are Huffman coding and Arithmetic coding.

(III) Psycho-visual Redundancy: Human eye is not fine-tuned to process any band of frequencies. Most 2D intensity arrays consist of information that is ignored by the human visual system. Image and video compression techniques aim at eliminating or reducing any amount of data.

Different image compression techniques apply different methods to remove the redundancies in order to represent an image without much compromise in the image quality. When the image matrix A is decomposed into three matrices U, D and V using the SVD and singular values σ_i are arranged in descending order on the diagonal of D, it follows from the fact that the first singular value contains the greatest amount of information and subsequent singular values contain decreasing amounts of information step by step. Therefore, only a few singular values are kept while lower singular values containing negligible or less important information can be discarded and it will not affect the quality of the image.

In order to achieve good amount of compressing, we rewrite the equation (3.1) via the choice of k values

$$A = \sum_{i=1}^{k} \sigma_i u_i v_i^\top = \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \cdots + \sigma_k u_k v_k^\top, \tag{3.3}$$

where $k < r$. The compressed image applying (3.3) will reduce the storage space requirement to $k(m+n+1)$ bytes as against the storage space requirement of mn bytes of the original image. The choices of the difference integer k have a difference corresponding image and storage. In summary, the value of k is chosen such that good amount of compression is achieved while image quality is maintained.

3.3. Image compression measures

To compare the performance of different compression techniques and to measure the degree to which an image is compressed, we can compute the compression factor and the quality of the compressed image. Some performance measures are listed.

(1) Compression Ratio (CR): It is the ratio of the storage space required to store original image to that required to store a compressed image. It computes the extent to which pixels in compressed image and is calculated as

$$\mathrm{CR} = \frac{m \times n}{k \times (m + n + 1)}.$$

(2) Mean Square Error (MSE): It is the measure of deterioration of image quality as compared to the original image when an image is compressed. It is defined as square of the difference between pixel value of original image and the corresponding pixel value of the compressed image averaged over the entire image, i.e.,

$$\mathrm{MSE} = \frac{\|A - \widetilde{A}\|_F^2}{mn},$$

where the original image and the compressed image are denoted by A and $\widetilde{A}$, respectively.

(3) Peak Signal to Noise Ratio (PSNR): It is the ratio of maximum signal power to the noise power that corrupts it. In image compression maximum signal power refers to the original image and noise is introduced to compress it and noise is the deviation of the compressed image from the original one. Hence, it follows that PSNR provides the quality of the compressed image:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\mathrm{MSE}}.$$

(4) Structural Similarity Index (SSIM): Structural similarity provides an alternative and complementary method to the image quality assessment. It is based on a top-down assumption that the human visual system is highly adapted for extracting structural information from the scene, and therefore a measure of structural similarity should be a good approximation of perceived image quality. Let x and y be two discrete nonnegative signals that have been aligned with each other, and let μ_x, σ_x^2 and σ_{xy} be the mean of x, the variance of x and the covariance of x and y, respectively. The SSIM models the image quality distortion as a combination of three different factors: loss of correlation, luminance distortion, and contrast distortion, which are given as follows:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \quad c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \quad s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3},$$

where C_1, C_2 and C_3 are small constants given by

$$C_1 \equiv (K_1 L)^2, \quad C_2 \equiv (K_2 L)^2, \quad C_3 \equiv C_2/2.$$

L is the dynamic range of the pixel values ($L = 255$ for 8 bits/pixel gray scale images), and $K_1 \ll 1$ and $K_2 \ll 1$ are two scalar constants. The general form of the SSIM between signal x and y is defined as

$$\mathrm{SSIM}(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y).$$

Therefore, the SSIM can be given by

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)},$$

which satisfies the following conditions:

(a) symmetry: $\mathrm{SSIM}(x, y) = \mathrm{SSIM}(y, x)$;

(b) boundedness: $\mathrm{SSIM}(x, y) \leq 1$;

(c) unique maximum: $\mathrm{SSIM}(x, y) = 1$ if and only if $x = y$.

4. Proposed lossy image compression technique

The contribution of this paper is the introduction of the concept of refinement of approximate invariant subspaces based on SVD method to image compression, as the main idea of image compression is reducing the redundancy of the image and transferring data in the efficient form. We have to mention that this work is the first one about an application of refinement of approximate invariant subspaces based on SVD method to high-resolution image compression. We are not aware of any work done for high-resolution image compression problems, although SVD and other variations of SVD can be applicable for general image compression. Therefore, our method is efficient, provided that high-resolution image compression is available. It is general and many other computer visions can benefit from using it.

In this section, we propose our contribution in which we integrate the refinement of approximate invariant subspaces and apply it to create an algorithm that compresses an image.

Initially the input image is read by the MATLAB software and stored as an array of integers. If this color image is segregated into RGB or red, green, blue component matrices. The pixel values of these matrices are converted to double data type for high accuracy. Hence, we can obtain one of the following matrices $A \in \mathbb{R}^{m\times n}$ for image processing. When the traditional method called SVD is applied to an image, it is not compressed, but the data take a form in which the first singular value has a great amount of image information. Moreover, we can keep only a few singular values to represent the image with little differences from the original. According to the information of SVD method,

our contribution in this paper is to introduce invariant subspaces for high-resolution image compression that overcomes some difficulties encountered in existing methods. Our efficient method is to concern with the computation of the eigenvalues of largest modulus of a large-scale matrix A and we use the dominant eigenvalues with largest modulus to do image compressing and obtain a good quality of the reconstructed image compared to the original image with little differences. Next, we will introduce how to adopt the invariant subspace corresponding to sought eigenvalues.

4.1. Refinement of approximate invariant subspace of A

In order to represent it simply, we consider the standard eigenvalue problem (SEP) for $A \in \mathbb{R}^{n\times n}$ ($A \in \mathbb{R}^{m\times n}$ treated similarly and consider $A^\top A \in \mathbb{R}^{n\times n}$):

$$Ax = \lambda Ix, \quad x \neq 0,$$

where λ is the eigenvalue and $x \in \mathbb{R}^{n\times 1}$ is the following eigenvector.

From [12], an estimate of an invariant subspace for $A \in \mathbb{R}^{n\times n}$ can be refined through

$$[-R, I_s]P^\top AP[I_{n-s}, R^\top]^\top = 0 \tag{4.1}$$

with $R \in \mathbb{R}^{s\times(n-s)}$ being some correction and $P = [P_1, P_2] \in \mathbb{R}^{n\times n}$ being orthogonal (or $P^{-1} = P^\top$). We assume that P is in Householder factors (see [16, p. 224]), so that vector multiplications by P (from the left and the right) can be computed in $O(n)$ flops. Here, $P_1 \in \mathbb{R}^{n\times s}$ ($s \ll n$) is an accurate estimate to the basis of some invariant subspace associated with A_{11} (see [34]) and the subspectral of the submatrices $A_{11} \in \mathbb{R}^{s\times s}$ and $A_{22} \in \mathbb{R}^{(n-s)\times(n-s)}$ are nonintersecting, providing

$$\sigma(A_{11}) \cap \sigma(A_{22}) = \emptyset,$$

thus the invariant subspace approximated by $\mathrm{span}(P_1)$ is isolated and well-defined. We can rewrite (4.1) into the nonsymmetric algebraic Riccati equation (NARE)

$$\mathcal{N}(R) \equiv A_{22}R - RA_{11} - RA_{12}R + A_{21} = 0 \tag{4.2}$$

with $A_{ij} = P_i^\top AP_j$ ($i, j = 1, 2$). The condition for the solvability of (4.2) is satisfied; see [10,12,13,34], essentially when P_1 is an accurate estimate, and the residual $A_{21} \in \mathbb{R}^{(n-s)\times s}$ or the correction R are small in norm, relative to the gap between A_{11} and A_{22}. Next, we introduce the important theorem.

Theorem 4.1. (Stewart [34, Theorem 4.11]) *Let $P^\top AP = (A_{ij})$ be as in (4.2) with the orthogonal $P \equiv [P_1, P_2]$. If $\delta = \mathrm{sep}(A_{11}, A_{22}) \equiv \inf_{\|X\|=1} \|A_{11}X - XA_{22}\| > 0$ and $\|A_{21}\| \cdot \|A_{12}\| < \delta^2/4$, then there exists a unique matrix R such that the columns of*

$T = P_1 + P_2R$ span an invariant subspace of A and $A_{11} + A_{12}R$ is the representation of A in span(T). *The matrix R solves the NARE $A_{22}R - RA_{11} - RA_{12}R + A_{21} = 0$ in* (4.2) *with $\|R\| < 2\delta^{-1}\|A_{21}\|$.*

Note that the perturbation analysis of (4.2) has presented in [34]. We assume that s, the dimension of the invariant subspace, is small relative to n, the dimension of A. Throughout the refinement of the invariant subspace method, we can get the matrix $\widetilde{A}_{11} = A_{11} + A_{12}R \in \mathbb{R}^{s\times s}$ and the first s dominant eigenvalues with largest modulus are located in $\widetilde{A}_{11}$, then we can reconstruct a new image with little differences compared to the original one based on SVD method. In the following subsection, we attempt the refinement of an estimate for an invariant subspace for a real matrix A by applying Newton's method to the NARE (4.2).

4.2. Newton's method for solving NARE

Consider the application of Newton's method on NARE (4.2) with $R_0 = 0$, we may solve the refinement of an estimate invariant subspace (REIS) problem via Sylvester equations in the following form: (for $k \geq 0$)

$$\widetilde{A}_{22}R_{k+1} - R_{k+1}\widetilde{A}_{11} = -\widetilde{A}_{21} \tag{4.3}$$

with $\widetilde{A}_{22} \equiv A_{22} - R_kA_{12}$, $\widetilde{A}_{11} \equiv A_{11} + A_{12}R_k$ and $\widetilde{A}_{21} \equiv A_{21} + R_kA_{12}R_k$. For the convergence of Newton's method in (4.2), please confer [10, 12], which are predated by [36, 37]. Under favourable conditions in [10, 36, 37], the iterates R_k will converge to the correction R quadratically. Note that $\widetilde{A}_{22}$ is low-ranked update of A_{22}, we can compute its inversion efficiently, through the Sherman–Morrison–Woodbury formula (SMWF)

$$(\widetilde{B} \pm M\widetilde{C}N)^{-1} = \widetilde{B}^{-1} \mp \widetilde{B}^{-1}M(\widetilde{C}^{-1} \pm N\widetilde{B}^{-1}M)^{-1}N\widetilde{B}^{-1}. \tag{4.4}$$

Furthermore, A_{11}, A_{12} and A_{21}, and thus $\widetilde{A}_{11}$, $\widetilde{A}_{12}$ and $\widetilde{A}_{21}$ can all be calculated efficiently via the products of the matrix A to $P_1^\top$ and P_1, P (in its Householder factors) to vectors, and between vectors. Next, we need to solve the Sylvester equation (4.3). When $s = 1$, (4.3) degenerates to a linear equation in R_{k+1}, requiring the inversion of $\widetilde{A}_{22} - \widetilde{A}_{11}I$, a low-ranked update of $A_{22} - \widetilde{A}_{11}I$. Then, the solution of Sylvester equation (4.3) can be computed the inverse iteration for $\widetilde{A}_{22}$ with the shift $\lambda = \widetilde{A}_{11}$.

When we consider $s > 1$ with $s \ll n$, we have to construct an algorithm to solve the Sylvester equation (4.3), by the inversion of matrices such as $A_{22} - \gamma I$. Consider the Schur decomposition [16, p. 313] on the small $\widetilde{A}_{11} = QZQ^\top \in \mathbb{R}^{s\times s}$, we can transform (4.3) into

$$\widetilde{A}_{22}R_{k+1} - R_{k+1}QZQ^\top = -\widetilde{A}_{21}, \tag{4.5}$$

where Q is unitary, Z is upper triangular with z_j $(j = 1, \ldots, s)$ being the jth diagonal element and $\eta_j \in \mathbb{C}^{j-1}$ $(j = 2, \ldots, s$; degenerate otherwise) contains elements above z_j in Z. We multiply the unitary matrix Q on the right-hand side of (4.5) and get

$$\widetilde{A}_{22}X - XZ = -\widehat{A}_{21} \tag{4.6}$$

with $X = R_{k+1}Q$ and $\widehat{A}_{21} = \widetilde{A}_{21}Q$. For $j = 1, 2, \ldots, s$, we solve the Sylvester equation (4.6), providing

$$(\widetilde{A}_{22} - z_j I)x_j = s_j \equiv -(\widehat{A}_{21})_j + X_{-j}\eta_j, \tag{4.7}$$

where $(\cdot)_j$ denotes the jth column, $x_j = (X)_j$ and $X_{-j} = [x_1, \ldots, x_{j-1}]$ $(j = 2, \ldots, s$; degenerate otherwise). Note that $\widetilde{A}_{22} - z_j I = (A_{22} - z_j I) - R_k A_{12}$, a low-ranked update of $A_{22} - z_j I$, is nonsingular since the spectra of A_{11} and A_{22} are nonintersecting and $\|R_k\|$ is small. As a matter of fact, in many applications, the spectra are far apart and $A_{22} - z_j I$ should be well-conditioned. Next, we solve the linear systems in (4.7) and it can be realized efficiently via the inversion of $A_{22} - z_j I$ discussed in the next section, with the help of the SMWF. The correction $R_{k+1} = XQ^\top$ can be retrieved from X. When we solve the equation (4.7) $(j = 1, \ldots, s)$ we need $\widetilde{A}^{-1}U$ in the SMWF in (4.4) with $\widetilde{A} = A_{22} - z_j I$, $U = R_k$ (degenerate for $k = 0$), $\widetilde{C} = I_s$ and $V = A_{12}$ and the details information can be listed in the following section.

4.3. Linear systems associated with $A_{22} - \gamma I$

The main building block of our algorithm is the inversion of $A_{22} - \gamma I$, for some γ, via the efficient inversion of the near-singular $A - \gamma I$. It is required for the Sylvester equation (4.3) in Newton's method.

Note that near-singular linear systems are modified to well-conditioned ones [7, 19], but we solve a smaller well-conditioned linear system via a slightly bigger ill-conditioned one, because of the inherent structure in A. The error analysis can be discussed in [14, Section 2.3.1]. In order to solve the linear system (4.7) efficiently, we apply the efficient inversion of $A - \gamma I$. To unify the treatment for the refinement of an estimate invariant subspace, we assume

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = P^\top (A - \gamma I)P.$$

It is vital that P is stored in Householder factor, so that the multiplication by P can be carried out in $O(n)$ computational complexity. Let $D_{ij} = P_i^\top (A - \gamma I)^{-1} P_j$ $(i, j = 1, 2)$,

we have $D_{11} = (C_{11} - C_{12}C_{22}^{-1}C_{21})^{-1}$, and

$$P^\top (A-\gamma I)^{-1} P \equiv \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}^{-1} = \left[\begin{array}{c|c} D_{11} & -D_{11}C_{12}C_{22}^{-1} \\ \hline -C_{22}^{-1}C_{21}D_{11} & C_{22}^{-1} + C_{22}^{-1}C_{21}D_{11}C_{12}C_{22}^{-1} \end{array}\right].$$

From $D_{22}v = C_{22}^{-1}v + C_{22}^{-1}C_{21}D_{11}C_{12}C_{22}^{-1}v$, we can obtain

$$C_{22}^{-1}v = D_{22}v - C_{22}^{-1}C_{21}D_{11}C_{12}C_{22}^{-1}v. \tag{4.8}$$

Replacing $C_{22}^{-1}C_{21}D_{11}$ and $C_{12}C_{22}^{-1}$ by D_{21} and $D_{11}^{-1}D_{12}$, respectively in (4.8), for some v, we get

$$C_{22}^{-1}v = D_{22}v - D_{21}D_{11}^{-1}D_{12}v. \tag{4.9}$$

Therefore, the products $D_{22}v$ and $D_{12}v$ can be computed through

$$\begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} 0 \\ v \end{bmatrix} = P^\top (A-\gamma I)^{-1} P \begin{bmatrix} 0 \\ v \end{bmatrix},$$

which is equivalent to the solution of the linear system (e.g., Gaussian elimination with partial pivoting)

$$(A-\gamma I)y = P \begin{bmatrix} 0 \\ v \end{bmatrix} \tag{4.10}$$

with $\begin{bmatrix} D_{12}v \\ D_{22}v \end{bmatrix} = P^\top y$. From (4.9), we also require

$$D_{11} = (C_{11} - C_{12}C_{22}^{-1}C_{21})^{-1} \quad \text{and} \quad D_{21} = -C_{22}^{-1}C_{21}D_{11},$$

which have only s columns and can be calculated efficiently through the linear system (solved by Gaussian elimination with partial pivoting)

$$(A-\gamma I)W = P \begin{bmatrix} I_s \\ 0 \end{bmatrix} \tag{4.11}$$

with $\begin{bmatrix} D_{11} \\ D_{21} \end{bmatrix} = P^\top W$.

4.4. Real Schur form

In Section 4.2, we apply the real Schur form of $\widetilde{A}_{11}$ on (4.3) and discuss real eigenvalues, then we also consider the complex conjugate pair of eigenvalues z_j, $z_{j+1} = \overline{z}_j$ and the corresponding eigenvectors x_j, $x_{j+1} = \overline{x}_j$. We have the real $X_j \in \mathbb{R}^{(n-s)\times 2}$, spanning the same column space as $[x_j, x_{j+1}]$. Corresponding, we have the real 2×2 block Z_j instead of z_j, z_{j+1}, with $\eta_j, \eta_{j+1} \in \mathbb{R}^{j-1}$ above the block Z_j. With $(\cdot)_{j1:j2}$ denoting the $j1$ to $j2$ columns and X_{-j} as previously defined in Section 4.2, (4.7) is changed into

$$\widetilde{A}_{22}X_j - X_jZ_j = -(\widehat{A}_{21})_{j:j+1} + X_{-j}[\eta_j, \eta_{j+1}]. \tag{4.12}$$

Applying the Kronecker product, (4.12) becomes

$$\widetilde{M}_{kj}v(X_j) = r_j$$

with $\widetilde{M}_{kj} \equiv I_2 \otimes \widetilde{A}_{22} - Z_j^\top \otimes I_{n-s}$, $r_j \equiv v[-(\widehat{A}_{21})_{j:j+1} + X_{-j}[\eta_j, \eta_{j+1}]]$ and $v[\cdot]$ stacking columns. The inversion of $\widetilde{M}_{kj}$ is to adopt the similar structures as $\widetilde{A}_{22}$ repeated in the two diagonal subblocks, which can be computed efficiently. One possible method is to apply the diagonal blocks as pivot to eliminate the off-diagonal blocks.

Let $Z_j = \left[\begin{smallmatrix} z_j & \alpha_{j1} \\ \alpha_{j2} & z_{j+1} \end{smallmatrix}\right]$, we have the low-ranked update

$$\widetilde{M}_{kj} = M_{kj} - \begin{bmatrix} R_k & 0 \\ 0 & R_k \end{bmatrix} \begin{bmatrix} A_{12} & 0 \\ 0 & A_{12} \end{bmatrix}$$

with

$$\widetilde{M}_{kj} = \left[\begin{array}{c|c} \widetilde{A}_{22} - z_jI_{n-s} & -\alpha_{j2}I_{n-s} \\ \hline -\alpha_{j1}I_{n-s} & \widetilde{A}_{22} - z_{j+1}I_{n-s} \end{array}\right],$$

$$M_{kj} \equiv \left[\begin{array}{c|c} A_{22} - z_jI_{n-s} & -\alpha_{j2}I_{n-s} \\ \hline -\alpha_{j1}I_{n-s} & A_{22} - z_{j+1}I_{n-s} \end{array}\right]. \tag{4.13}$$

Applying the structure of A on (4.13), we obtain

$$P^\top \begin{bmatrix} A - z_jI_n & -\alpha_{j2}I_n \\ -\alpha_{j1}I_n & A - z_{j+1}I_n \end{bmatrix} P = P^\top N_j P = \begin{bmatrix} * & * \\ * & M_{kj} \end{bmatrix}$$

with

$$N_j \equiv I_2 \otimes A - Z_j^\top \otimes I_n, \quad P = \begin{bmatrix} P & 0 \\ 0 & P \end{bmatrix} \begin{bmatrix} I_s & 0 & 0 & 0 \\ 0 & 0 & I_{n-s} & 0 \\ 0 & I_s & 0 & 0 \\ 0 & 0 & 0 & I_{n-s} \end{bmatrix}.$$

The inversion of $\widetilde{M}_{kj}$ can be utilized via the SMWF through the inversion of M_{kj}, which can in turn be done by inverting N_j as N_j is structured (cf. $A-\gamma I_n$ with $\gamma = z_j$ in Section 4.3), P is a projection and M_{kj} is the lower-right corner block in $P^\top N_j P$.

4.5. Algorithm and operation counts

For high-resolution image compression, we perform the computation in the refinement of an approximate invariant subspace based on SVD method. Firstly, we read the input image in the mathematical software MATLAB and it is stored as an array of integers. If this image is color, we obtain three matrices for red, green and blue components, otherwise, the matrix A is generated for grayscale image.

For color image in the image compression, the method is similar to that in the grayscale image, therefore, we only focus on the image compression for the grayscale image.

When we get the matrix A, we can adapt the refinement of an estimate of the invariant subspace on A. Equations (4.1)–(4.4), the linear systems in (4.8)–(4.9) and the real Schur form in (4.6)–(4.13), constitute our efficient method for image compression. When we get the correction R, we can obtain the following matrix $\widetilde{A}_{11}$ as the dominant eigenvalues (with largest modulus) are located in $\widetilde{A}_{11}$, then we apply $\widetilde{A}_{11}$ to reconstruct the matrix $\widetilde{A}$ and get the compressed image.

4.6. The relationship between REIS and SVD

In summary, the details about the relationship between REIS and SVD is following listed:

(I) We specially apply the REIS based on SVD to do image compression including large-sized image data and high-resolution image data.

(II) We only need to keep the first some singular values with largest modulus and take out remaining small singular values from SVD in order to achieve image compression.

(III) For large-sized images and high-resolution images, we should compute dominant eigenvalues of the large-scale matrices, which is related to singular values with largest modulus. Therefore, we adapt efficiently the refinement of estimates of invariant subspaces for large-scale matrices to get the dominant eigenvalues with largest modulus, through some nonsymmetric algebraic Riccati equations or their associated Sylvester's equations via Newton's method.

(IV) In the applications, which have not been investigated thoroughly for large-sized images and high-resolution images associated with large-scale matrices, are important and interesting, since they form a vital part of any eigen-solver or in the context of the continuation of invariant subspaces.

We summarize the algorithm for the refinement of estimates invariant subspaces in Algorithm 4.1 below, together with the dominant operation counts on the right side. The algorithm with real Schur form described in Section 4.4, is similar and will not be summarized here. We assume sM flops for one linear solve of a linear system associated with $A - \gamma I_n$ with s right-hand-sides, and $c_b n$ (or qn) flops for one multiplication of a vector in $\mathbb{R}^n$ by A or its transpose (or P or its transpose). With s-dimensional invariant subspaces, there are s factors in one of these projections and $q = 4s$. For the computation of the norm of A_{21} in $\mathbb{R}^{(n-s)\times s}$, we assume $2s^2 n$ flops [16]. The count for the solution of (4.7) is summarized as listed:

1. Let each linear system associated with $A_{22} - z_j I_{n-s}$ require $\widetilde{Z}$ flops.

2. For an application of the SMWF (4.4) to (4.7), the rank-s update in $\widetilde{A}_{22} - z_j I_{n-s} = (A_{22} - z_j)I - R_k A_{12}$ requires the quantities $(A_{22} - z_j I)^{-1} s_j$ (from (4.7) and $(A_{22} - z_j I)^{-1} R_k$), totalling $(s+1)\widetilde{Z}$ flops. Putting all the terms together in (4.4) adds another $2s(s+2)n$ flops.

3. Solving (4.10) and (4.11) requires $(s+1)(M+2qn)$ flops, and computing $C_{22}^{-1} v = (A_{22} - z_j I_{n-s})^{-1} v$ in (4.9) requires $(2s^2+1)n$ flops. Summing these counts up implies $\widetilde{Z} = (s+1)M + [2s^2 + 2(s+1)q + 1]n$ flops.

4. In total, the flop count for (4.7) is $s[\widetilde{Z} + s\widetilde{Z} + 2s(s+2)n] = s(s+1)M + s^2(s+1)M + c_a n s$ with
$$c_a \equiv 2s^3 + 4s^2 + 5s + 1 + 2(s+1)^2 q.$$

Algorithm 4.1

Inputs: A, $P = [P_1, P_2]$ in Householder factors, $\epsilon > 0$
Outputs: R, $\widetilde{A} = P_1 \widetilde{A}_{11} P_1^\top$
compute A_{11}, A_{12}, A_{21}; ▷ $2(c_b + q + 3)sn$
Assign $k = 0$, $R_0 = 0$, $r_0 = A_{21}$;
if $\|r_k\| < \epsilon$, then $R = R_k$, exit; ▷ $2s^2 n$
  else compute $\widetilde{A}_{11} = A_{11} + A_{12} R_k$ ▷ $2s^2 n$
    compute Schur decomposition $\widetilde{A}_{11} = QZQ^\top$; ▷ $O(s^3)$
    compute $\widehat{A}_{21} = (A_{21} + R_k A_{12} R_k)Q$, $X_{-j}\eta_j$ $(j = 2, \ldots, s)$; ▷ $(2.5s + 1.5)sn$
    solve (4.7) for X with the help of the SMWF (4.4); ▷ $s(s+1)M + c_a n$
    compute $R_{k+1} = XQ^\top$, $r_{k+1} = \mathcal{N}(R_{k+1})$, $k \leftarrow k+1$; ▷ $(2c_b + q + 3)n$
end

The final operation count for Algorithm 4.1 is $s(s+1)M + \widetilde{c}_a n$ flops per iteration with $\widetilde{c}_a \equiv c_a + 6.5s^2 + 1.5s + 2c_b + q + 3$, in addition to $2(c_b + q + 3)sn$ flops for the initial computation of A_{11}, A_{12} and A_{21}.

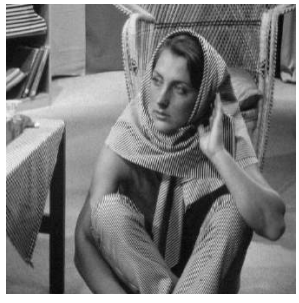
5. Numerical examples

In this section we present the results of five numerical experiments to show the performance of approximate invariant subspaces for large-scale matrices based on SVD in image compression. To validate the proposed algorithm, a number of experiments were conducted on real images. A set of grey-scale images, Lena, Pentagon, Fishing Boat, Barbara and Peppers are shown in Figure 5.1 for testing the performance, of which Lena is a quite smooth figure image; Pentagon and Fishing Boat are outdoor pictures with well-separated background; Barbara is highly textured and Peppers is an image of natural product, consisting of many but well-separated objects.

(a) Lena

(b) Pentagon

(c) Fishing Boat



(d) Barbara

(e) Peppers

Figure 5.1: Original image.

As the original pixel size of images is not large-scale, we modify Figure 5.1(c)(d)(e) into high-resolution images by using the MATLAB command `imresize.m`. Therefore, the following size of each image is changed into 512×512, 1024×1024, 5120×5120, 10200×10200, 51000×51000 and 8-bit per pixel. In order to present the numerical performance of our algorithm, we list the dimension of the invariant subspaces (k), the compression ratio (CR), the mean square error (MSE), the peak signal to noise ratio (PSNR), the structural similarity index (SSIM), and the CPU time in seconds (CPU) for each compressed image in Tables 5.1–5.5 below. This ratio is often used as the compression degree and the quality measurement between the original and compressed images. Furthermore, we draw three figures for each compressed image to illustrate the relationship between normalize MSE values in dB (NMSE), normalize PSNR values in dB (NPSNR) and SSIM with respect to the selected number of singular values in proposed image compression technique. The normalize MSE and normalize PSNR in the figures in Examples 5.1–5.4 are listed as

$$\text{NMSE} = \frac{\text{MSE}(k)}{\text{MSE}(k=2)} \quad \text{and} \quad \text{NPSNR} = \frac{\text{PSNR}(k)}{\text{PSNR}(k=2)},$$

where $\text{MSE}(k)$, $\text{MSE}(k=2)$ and $\text{PSNR}(k)$, $\text{PSNR}(k=2)$ denote the MSE and PSNR values obtained from k and $k=2$, respectively. In order to see the performance of the proposed image compression techniques, we show the quantitative comparison between the proposed techniques and the discrete cosine transformation (DCT) [22] by use of MSE, PSNR and SSIM for the same compression ratio in Examples 5.1–5.4. For the largest Example 5.4, we only perform our proposed technique as the DCT compression technique can not work. All testing images are downloaded from the image database at the Signal and Image Processing Institute, University of South California (USC-SIPI). All the numerical experiments are carried on a MacBook Pro with a 2.30 GHz Intel Core 8 Duo processor and 64 GB RAM, with machine accuracy eps $= 2.22 \times 10^{-16}$. All the numerical results are obtained from running the MATLAB [27] Version R2022a software.

Example 5.1. Figures 5.2–5.13 show examples of images used for the system tests under different k terms. When the values of k are taken as 2 and 10, the images are blurred which are shown in Figures 5.2–5.3. $k = 2$ and 10 mean that the images are reconstructed considering only the first two and ten singular values of the matrix D. Figures 5.4–5.5 show the reconstructed images with $k = 20$ and 30 which are somewhat less distorted than Figures 5.2–5.3. By observing Figures 5.2 to 5.13, it is clear that as the value of k (i.e., number of singular values used for reconstruction of the compressed image) is increased, the compressed image is close to the original image. This implies that image quality goes on increasing as the value of k is increased. All the above observations can be summarized in the form of Table 5.1 and Figure 5.14 as shown below.

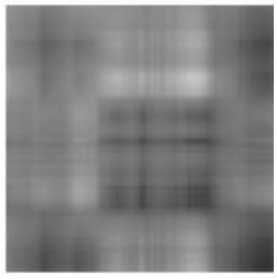


Figure 5.2: $k = 2$

Figure 5.3: $k = 10$

Figure 5.4: $k = 20$

Figure 5.5: $k = 30$

Figure 5.6: $k = 41$

Figure 5.7: $k = 51$

Figure 5.8: $k = 60$

Figure 5.9: $k = 73$

Figure 5.10: $k = 90$

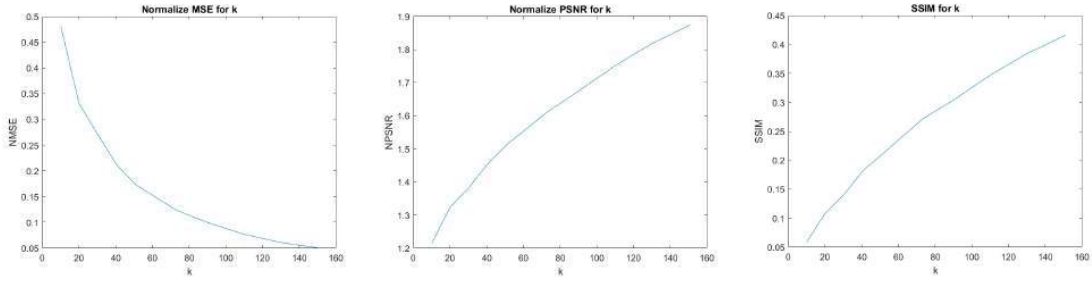
Figure 5.11: $k = 110$

Figure 5.12: $k = 130$

Figure 5.13: $k = 151$

Table 5.1: Summary of the result for image compression in Example 5.1.

k	CR	MSE (dB)	PSNR (dB)	SSIM	CPU (s)
2	127.875	2128.965	14.849	0.016	12.627
10	25.575	1023.328	18.031	0.059	12.193
20	12.788	704.816	19.65	0.108	10.14
30	8.525	577.241	20.517	0.14	12.053
41	6.238	445.753	21.64	0.183	11.997
51	5.015	368.077	22.471	0.21	13.515
60	4.263	327.12	22.984	0.234	80.914
73	3.602	263.101	23.93	0.271	10.714
90	2.842	212.081	24.866	0.304	14.095
110	2.325	162.983	26.009	0.347	27.139
130	1.967	129.718	27.001	0.384	11.816
151	1.694	107.1	27.833	0.416	35.812



(a) NMSE (b) NPSNR (c) SSIM

Figure 5.14: The diagram of curve for the image of "Lena" in the three measurements.

In the above Table 5.1 and Figure 5.14, degree of compression is measured using CR. MSE, PSNR values expressed in dB and SSIM are used as a measure of image quality. Following conclusions can be drawn on the basis of Table 5.1 and Figures 5.2–5.14:

1. Value of k represents the number of singular values used in the reconstruction of the compressed image.

2. Smaller the value of k, the more is the compression ratio (i.e., less storage space is required) but image quality deteriorates (i.e., larger MSE and smaller PSNR values).

3. As the value of k increases, image quality improves (the normalize MSE is diminishing by degrees and the normalize PSNR/SSIM are upward progressively) but more storage space is required to store the compressed image.

4. Therefore, we strike a balance between storage space required and image quality for good image compression. And from the above observations, we can find that optimum compression results are obtained when MSE of the compressed image is less than or equal to 212.081 dB (i.e., MSE $\leq$ 212.081 dB). In our example, it is obtained when the value of k is 90.

5. Generally, the choice of k depends on the application. For example, in some applications, if image quality is important then higher values of k are chosen. However, the storage space is more important than image quality, in the case lower k values are taken.

6. When k is equal to the rank of the image matrix, the reconstructed image is almost same as the original one. As k increases further, it is negligible decrease in MSE and increase in PSNR values. This means that it is negligible improvement in the image quality.

7. As an example, we fix the same compression ratio (CR = 1.694) to show MSE, PSNR, SSIM and their figures, respectively. The MSE value of the proposed compression technique for Lena image is 107.1 lower than that of DCT compression about 17543.782. However, the PSNR and SSIM values for the proposed compression technique are 27.833 and 0.416 higher than that of DCT compression technique about 5.69 and 0.00002, respectively. The figures of our proposed technique and DCT compression are shown in Figure 5.15. From the above information, the proposed technique outperforms DCT compression technique.

(a) Our proposed technique (b) DCT

Figure 5.15: The images of "Lena" when CR = 1.694.

Example 5.2. Figures 5.16–5.27 describe examples of images used for the system tests under different k terms. Figure 5.16 displays the result of the reconstruction image using 2 singular values, and Figure 5.17 shows the result using 11 singular values and so on. The observation on those examples, we found when $k \leq 101$, the images are blurry and with the increase of singular values we have a approach to the original image. Table 5.2 shows a summary of the results obtained with the measure of CR, MSE, PSNR, SSIM and CPU time for the tested images.

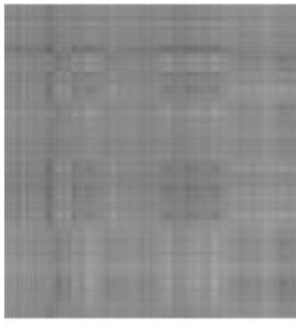


Figure 5.16: $k = 2$

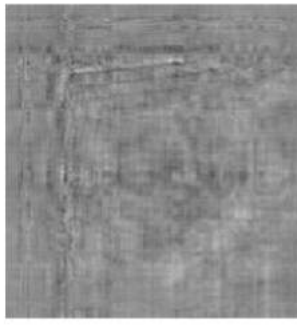


Figure 5.17: $k = 11$

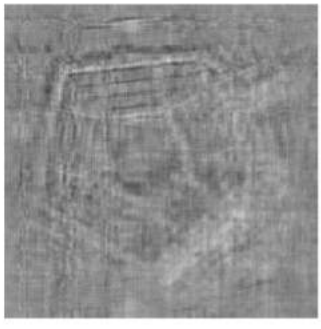


Figure 5.18: $k = 21$

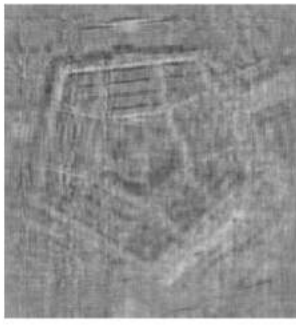


Figure 5.19: $k = 30$

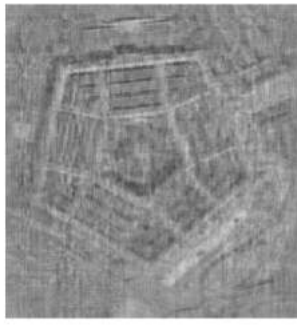


Figure 5.20: $k = 41$

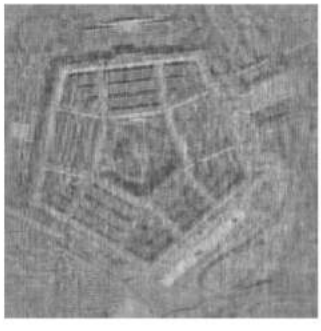


Figure 5.21: $k = 50$

Figure 5.22: $k = 101$

Figure 5.23: $k = 150$

Figure 5.24: $k = 200$

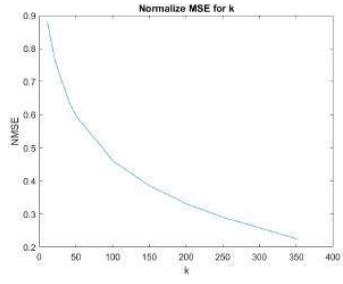
Figure 5.25: $k = 251$

Figure 5.26: $k = 301$

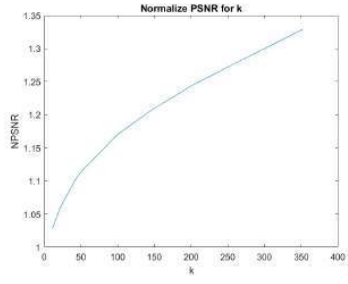
Figure 5.27: $k = 352$

Table 5.2: Summary of the result for image compression in Example 5.2.

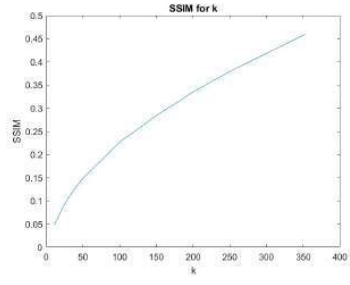
k	CR	MSE (dB)	PSNR (dB)	SSIM	CPU (s)
2	255.875	691.932	19.73	0.013	11.644
11	46.523	608.641	20.287	0.048	12.59
21	24.369	530.326	20.885	0.082	12.261
30	17.058	487.121	21.254	0.106	11.229
41	12.482	440.676	21.69	0.131	23.697
50	10.235	412.473	21.977	0.149	45.308
101	5.067	317.4	23.115	0.228	23.573
150	3.412	266.637	23.872	0.284	171.583
200	2.559	228.956	24.533	0.335	96.372
251	2.039	200.427	25.111	0.38	40.957
301	1.7	176.482	25.664	0.42	43.791
352	1.462	155.218	26.221	0.459	37.623



(a) NMSE



(b) NPSNR

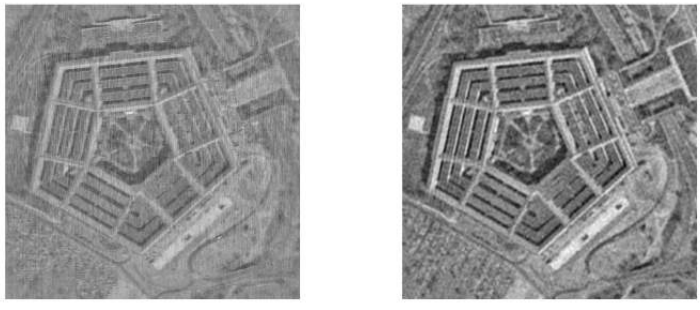


(c) SSIM

Figure 5.28: The diagram of curve for the image of "Pentagon" in the three measurements.

With the results from Table 5.2 and Figure 5.28, we have a couple of the observations:

1. Using less singular value (smaller k), the better compression ratio is achieved.
2. But, the more singular value is used (larger k), the quality measurement MSE is smaller (the NMSE becomes reducing) and PSNR is larger (the NPSNR and SSIM are rising stage by stage for better image quality), then the reconstructed images are more equal to the original one.
3. For the testing image, the acceptable image quality is about with $k = 150$, and the compression ratio is 3.412.
4. The reconstructed image is close to original one when $k = 352$. At this point, the CR = 1.462, and MSE = 155.218.
5. As an example, we fix the same compression ratio (CR = 3.412) to show MSE, PSNR, SSIM and their figures, respectively. The MSE value of the proposed compression technique for Pentagon image is 266.637 lower than that of DCT compression about 19393.149. However, the PSNR and SSIM values for the proposed compression technique are 23.872 and 0.284 higher than that of DCT compression technique about 5.254 and 0.00002, respectively. The figures of our proposed technique and DCT compression are shown in Figure 5.29. Overall, the proposed technique overcomes DCT compression technique.



(a) Our proposed technique (b) DCT

Figure 5.29: The images of "Pentago" when CR = 3.412.

Example 5.3. Figures 5.30–5.41 represent examples of images used for the system tests under different k terms. Since the pixel size of Fishing Boat is large-scale with 5120×5120, the first seven reconstruction images in figures are still blurry. When $k \geq 151$, the reconstruction images are visible in Figure 5.37. When we take $k = 500$, the compressed image is close to the original one and it shows the efficiency of our proposed algorithm.

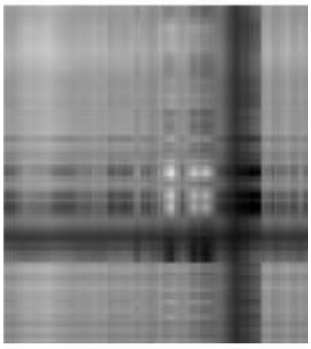


Figure 5.30: $k = 2$

Figure 5.31: $k = 11$

Figure 5.32: $k = 21$

Figure 5.33: $k = 40$

Figure 5.34: $k = 60$

Figure 5.35: $k = 82$

Figure 5.36: $k = 100$

Figure 5.37: $k = 151$

Figure 5.38: $k = 200$

Figure 5.39: $k = 250$

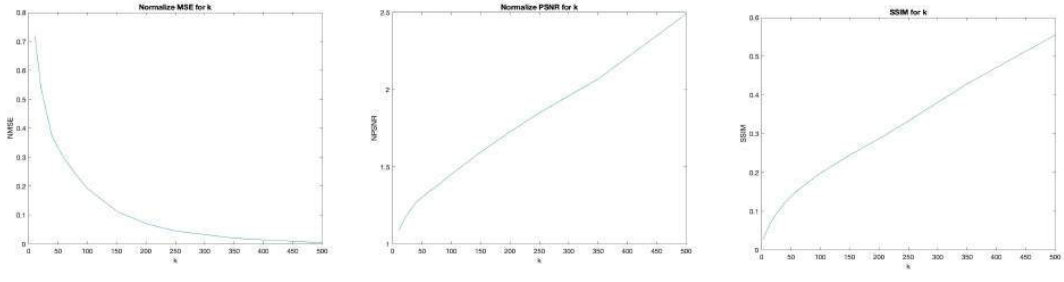
Figure 5.40: $k = 351$

Figure 5.41: $k = 500$

Table 5.3 presents a summary of the results obtained with the measure of k, CR, MSE, PSNR, SSIM and CPU for the tested image. Moreover, we plot some figures to show the trends between NMSE, NPSNR, SSIM and the selected number of singular values k.

Table 5.3: Summary of the result for image compression in Example 5.3.

k	CR	MSE (dB)	PSNR (dB)	SSIM	CPU (s)
2	1279.875	1696.532	15.835	0.026	10.435
11	232.705	1219.155	17.27	0.057	17.78
21	121.893	915.422	18.515	0.085	18.976
40	63.994	630.717	20.132	0.123	30.388
60	42.663	504.351	21.104	0.152	124.052
82	31.216	402.450	22.084	0.178	138.578
100	25.598	326.188	22.996	0.198	192.216
151	16.952	190.797	25.325	0.246	214.307
200	12.799	119.67	27.351	0.287	253.806
250	10.239	76.449	29.297	0.333	158.331
351	7.293	34.082	32.806	0.43	390.688
500	5.12	7.361	39.461	0.555	49.525



(a) NMSE (b) NPSNR (c) SSIM

Figure 5.42: The diagram of curve for the image of "Fishing Boat" in the three measurements.

Based on the results from Table 5.3 and Figure 5.42, we can get some conclusions:

1. Smaller the value of k, the larger is the compression ratio and the mean square error, but otherwise the smaller is the peak signal to noise ratio and the structural similarity index.

2. Since the value of k is increasing, the normalize MSE is gradually decreasing, but the normalize PSNR and the SSIM are raising step by step.

3. For the testing image, the acceptable image quality is about with $k = 200$, and the CR, MSE, PSNR and SSIM are 12.799, 119.67, 27.351 and 0.287, respectively.

4. The compressed image approaches original one when $k = 500$. At this point, the CR = 5.12, PSNR = 39.461, SSIM = 0.555, and CPU = 49.525.

5. In the example, we fix the same compression ratio (CR = 25.598) to show MSE, PSNR, SSIM and their figures, respectively. The MSE value of the proposed compression technique for Fishing Boat image is 326.188 lower than that of DCT compression about 1462.233. However, the PSNR and SSIM values for the proposed compression technique are 22.996 and 0.198 higher than that of DCT compression technique about 16.481 and 0.002, respectively. The figures of our proposed technique and DCT compression are shown in Figure 5.43. Overall, the proposed technique overcomes DCT compression technique.

(a) Our proposed technique (b) DCT

Figure 5.43: The images of "Fishing Boat" when CR = 25.598.

Example 5.4. Figures 5.44–5.55 represent examples of images used for the system tests under different k terms. When we take the value of $k \leq 201$, the reconstruction images are blurry in Figures 5.44–5.50. With the more singular values used, the compressed images become more and more visible. When the value of k is taken 600, the reconstruction image is close to the original one with the pixel size 10200×10200 and it shows the efficiency of our proposed algorithm. We list Table 5.4 and three figures to describe a summary of the results obtained with the measure of k, CR, MSE, PSNR, SSIM, CPU and the trends between NMSE, NPSNR, SSIM and k.

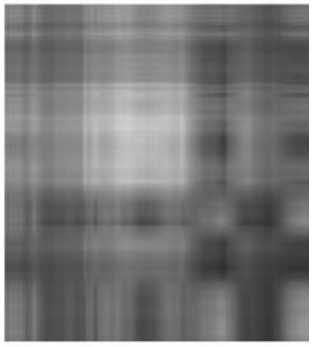


Figure 5.44: $k = 2$

Figure 5.45: $k = 10$

Figure 5.46: $k = 20$

Figure 5.47: $k = 51$

Figure 5.48: $k = 101$

Figure 5.49: $k = 150$

Figure 5.50: $k = 201$

Figure 5.51: $k = 250$

Figure 5.52: $k = 301$

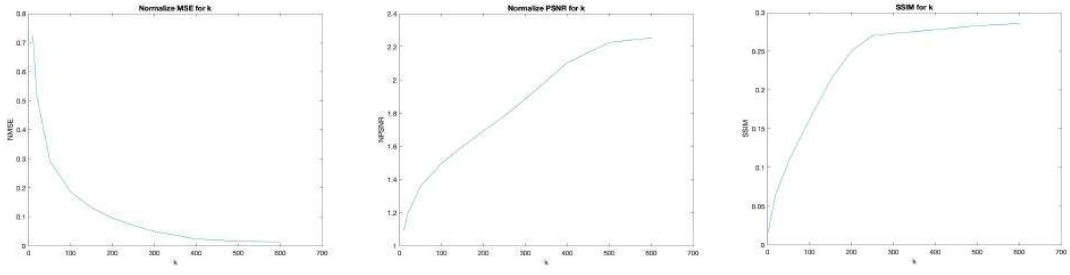
Figure 5.53: $k = 400$

Figure 5.54: $k = 500$

Figure 5.55: $k = 601$

Table 5.4: Summary of the result for image compression in Example 5.4.

k	CR	MSE (dB)	PSNR (dB)	SSIM	CPU (s)
2	2549.875	2217.745	14.672	0.017	26.574
10	509.975	1606.138	16.073	0.042	40.776
20	254.988	1144.203	17.546	0.066	71.595
51	99.995	646.817	20.023	0.109	233.444
101	50.493	410.688	21.996	0.163	506.02
150	33.998	292.339	23.472	0.213	640.837
201	25.372	212.334	24.861	0.251	1096.584
250	20.399	156.979	26.172	0.27	291.706
301	16.943	110.515	27.697	0.273	574.588
400	12.749	53.086	30.881	0.278	2791.985
500	10.2	35.332	32.649	0.283	1329.791
601	8.485	32.065	33.07	0.286	16723.542



(a) NMSE (b) NPSNR (c) SSIM

Figure 5.56: The diagram of curve for the image of "Fishing Boat" in the three measurements.

With the results from Table 5.4 and Figure 5.56, we can have some observations:

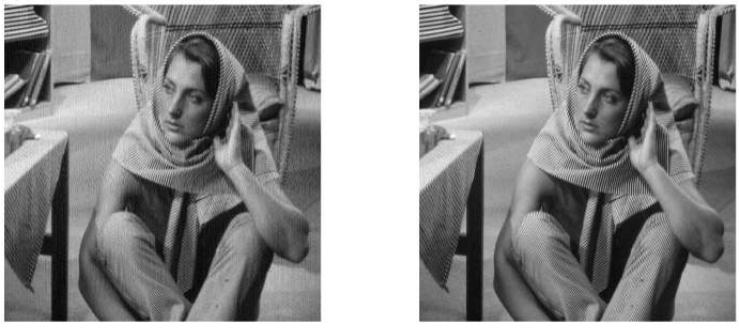
1. Using less singular value, the better compression ratio and less storage space are achieved but the quality of reconstruction image is blurry when $k = 2$.

2. With the increase of value k, the NPSNR and the SSIM are gradually enlarging; however, the NMSE is declining step by step.

3. For the testing image, the acceptable image quality is about with $k = 250$, and the CR, MSE, PSNR, SSIM and CPU are 20.399, 156.979, 26.172, 0.27 and 291.706, respectively.

4. The reconstruction image is close to original one when $k = 601$. At this point, the CR = 8.485, PSNR = 33.07, SSIM = 0.286, and CPU = 16723.542.

5. In the example, we fix the same compression ratio (CR = 20.399) to show MSE, PSNR, SSIM and their figures, respectively. The MSE value of the proposed compression technique for Barbara image is 156.979 lower than that of DCT compression about 15847.982. However, the PSNR and SSIM values for the proposed compression technique are 26.172 and 0.27 higher than that of DCT compression technique about 6.131 and 0.0002, respectively. The figures of our proposed technique and DCT compression are shown in Figure 5.57. From the above information, the proposed technique outperforms DCT compression technique.



(a) Our proposed technique (b) DCT

Figure 5.57: The images of "Barbara" when CR = 20.399.

Example 5.5. Figures 5.58–5.69 denote examples of images used for the system tests under different k terms. Due to the high-resolution image of Peppers with 51000×51000, the compressed images in Figure 5.64 are blurry when $k \leq 200$. When the value of k is taken 700, the reconstruction image is close to the original one and it expresses the efficiency of our proposed algorithm. Table 5.5 and three figures represent a summary of the results obtained with the measure of k, CR, MSE, PSNR, SSIM, CPU and the tendencies between NMSE, NPSNR, SSIM and k.

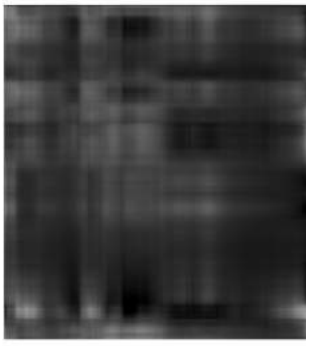


Figure 5.58: $k = 3$

Figure 5.59: $k = 10$

Figure 5.60: $k = 21$

Figure 5.61: $k = 51$

Figure 5.62: $k = 102$

Figure 5.63: $k = 152$

Figure 5.64: $k = 200$

Figure 5.65: $k = 301$

Figure 5.66: $k = 401$

Figure 5.67: $k = 503$

Figure 5.68: $k = 600$

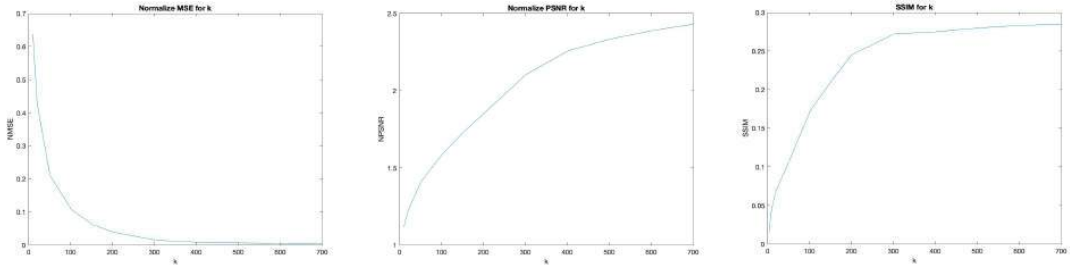
Figure 5.69: $k = 700$

Table 5.5: Summary of the result for image compression in Example 5.5.

k	CR	MSE (dB)	PSNR (dB)	SSIM	CPU (s)
3	8499.917	1467.028	16.466	0.015	632.377
10	2549.975	934.195	18.426	0.047	591.191
21	1214.274	621.809	20.194	0.07	456.953
51	499.995	308.784	23.234	0.107	2514.909
102	249.998	157.904	26.147	0.173	1915.459
152	167.762	91.877	28.499	0.211	9991.022
200	127.499	58.351	30.47	0.245	13245.557
301	84.717	22.343	34.639	0.272	15601.574
401	63.59	12.478	37.169	0.275	43232.312
503	50.695	9.309	38.442	0.28	41294.957
600	42.5	7.594	39.326	0.283	57485.781
700	36.428	6.475	40.019	0.285	45241.042

Since we begin the smallest dimension of the invariant subspace $k = 3$, we do the normalize MSE and normalize PSNR as follows:

$$\text{NMSE} = \frac{\text{MSE}(k)}{\text{MSE}(k=3)} \quad \text{and} \quad \text{NPSNR} = \frac{\text{PSNR}(k)}{\text{PSNR}(k=3)}.$$



(a) NMSE (b) NPSNR (c) SSIM

Figure 5.70: The diagram of curve for the image of "Peppers" in the three measurements.

Based on the results from Table 5.5 and Figure 5.70, we can conclude some observations:

1. For the testing image, the acceptable image quality is with $k = 301$, and the CR,

MSE, PSNR, SSIM and CPU are 84.717, 22.343, 34.639, 0.272 and 15601.574, respectively.

2. The compressed image is close to original one when $k = 700$. At this point, the CR = 36.428, MSE = 6.475, PSNR = 40.019, SSIM = 0.285, and CPU = 45241.042.

6. Conclusions

In this article, we propose the REIS algorithm with applications to do image compression and high dimensionality reduction especially high-resolution images, by knitting the singular value decomposition (SVD) together with the REIS algorithm. Our refinement procedures will be useful as part of any eigen-solver. In particular, they will be useful for the CIS problem, or when subspace methods are used to compute the initial invariant subspace. Theoretical results and numerical experiments are established to show the validity and rationality of our proposed algorithm. Our algorithm is much cheaper with $O(n)$ computational complexity per iteration and show some performance measures including compression ratio, mean square error, peak signal to noise ratio and structural similarity index.

The main contribution of this paper is to show why SVD works for the REIS algorithm from a theoretical point of view. To do this, we first consider the decaying property of singular values of the matrices during iterations of the REIS algorithm, and determine the target rank k in the SVD process. In the high-resolution image, we have presented the algorithm for the refinement of invariant subspace for large-scale matrix A. Our technique solves the NARE (4.2) by the quadratically convergent Newton's method, and we provide a sufficient condition for the convergence of the REIS algorithm. Furthermore, we also discuss the operation counts of the REIS algorithm to show $O(n)$ computational complexity per iteration. Numerical experiments on some real-world data sets demonstrate the numerical behavior of the proposed algorithm, and show the effectiveness of our theoretical results.

In the era of big data, the size of the data is usually so huge that the data collections cannot be stored in main memory. Our proposed REIS algorithm based on the SVD provides low-rank approximations of large-scale matrices which is a powerful tool in order to save memory requirement and keep good quality of high-resolution images.

Acknowledgments

This work was supported by Academia Sinica (Taiwan) Thematic Project grant number AS-TP-109-M07, and the Ministry of Science and Technology (Taiwan) grant numbers 107-2118-M-001-011-MY3, 109-2321-B-001-013, and 112-2115-M-415-002.

References

[1] N. K. El Abbadi, A. A. Rammahi, D. S. Redha and M. Abdul-Hameed, *Image compression based on SVD and MPQ-BTC*, J. Comput. Sci. **10** (2014), no. 10, 2095–2104.

[2] M. M. Adiwijaya, B. K. Dewi, F. A. Yulianto and B. Purnama, *Digital image compression using graph coloring quantization based on wavelet-SVD*, J. Phys. Conf. Ser. **423** (2013), 012019, 1–7.

[3] K. EL Asnaoui and Y. Chawki, *Two new methods for image compression*, Int. J. Imag. Robot. **15** (2015), no. 4, 79–96.

[4] K. EL Asnaoui, M. Ouhda, B. Aksasse and M. Ouanan, *An application of linear algebra to image compression*, in: *Homological and Combinatorial Methods in Algebra*, 41–54, Springer Proc. Math. Stat. **228**, Springer, Cham, 2018.

[5] A. Bilgin, M. W. Marcellin and M. I. Altbach, *Compression of electrocardiogram signals using JPEG2000*, IEEE Trans. Consum. Electron. **49** (2003), no. 4, 833–840.

[6] O. Bryt and M. Elad, *Compression of facial images using the K-SVD algorithm*, J. Vis. Commun. Image Represent. **19** (2008), no. 4, 270–282.

[7] T. F. Chan and D. C. Resasco, *Generalized deflated block-elimination*, SIAM J. Numer. Anal. **23** (1986), no. 5, 913–924.

[8] C.-C. Chang, Y.-S. Hu and C.-C. Lin, *A digital watermarking scheme based on singular value decomposition*, in: *Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, 82–93, Springer-Verlag, Berlin, 2007.

[9] C.-C. Chang, P. Tsai and C.-C. Lin, *SVD-based digital image watermarking scheme*, Pattern Recognit. Lett. **26** (2005), no. 10, 1577–1586.

[10] F. Chatelin, *Simultaneous Newton's iteration for the eigenproblem*, in: *Defect Correction Methods (Oberwolfach, 1983)*, 67–74, Comput. Suppl. **5**, Springer, Vienna, 1984.

[11] E. J. Delp and O. Mitchell, *Image compression ssing block truncation coding*, IEEE Trans. Commun. **27** (1979), no. 9, 1335–1342.

[12] J. W. Demmel, *Three methods for refining estimates of invariant subspaces*, Computing **38** (1987), no. 1, 43–57.

[13] J. J. Dongarra, C. B. Moler and J. H. Wilkinson, *Improving the accuracy of computed eigenvalues and eigenvectors*, SIAM J. Num. Anal. **20** (1983), no. 1, 23–45.

[14] H.-Y. Fan, P. C.-Y. Weng and E. K.-w. Chu, *Refining estimates of invariant and deflating subspaces for large and sparse matrices and pencils*, BIT **54** (2014), no. 1, 147–169.

[15] T. Glatard, C. Lartizien, B. Gibaud, R. Ferreira da Silva, G. Forestier, F. Cervenansky, M. Alessandrini, H. Benoit-Cattin, O. Bernard, S. Camarasu-Pop, N. Cerezo, P. Clarysse, A. Gaignard, P. Hugonnard, H. Liebgott, S. Marache, A. Marion, J. Montagnat, J. Tabary and D. Friboulet, *A virtual imaging platform for multi-modality medical image simulation*, IEEE Trans. Med. Imaging **32** (2013), no. 1, 110–118.

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Third edition, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, 1996.

[17] R. C. Gonzalez and R. E. Woods, *Digital image processing*, Second edition, Prentice Hall, New Jersey, 2001.

[18] R. C. Gonzalez, R. E. Woods and S. L. Eddins, *Digital Image Processing Using Matlab*, Pearson Prentice Hall, 2003.

[19] W. Govaerts and J. D. Pryce, *Mixed block elimination for linear systems with wider borders*, IMA J. Numer. Anal. **13** (1993), no. 2, 161–180.

[20] H. Gu, G. Zhao and J. Qiu, *Online metric learning for relevance feedback in e-commerce image retrieval applications*, Tsinghua Sci. Technol. **16** (2011), no. 4, 377–385.

[21] M. A. Joshi, *Digital Image Processing: An algorithmic approach*, PHI Learning, New Delhi, 2006.

[22] A. Kapoor and R. Dhir, *Image compression using fast 2-D DCT technique*, Int. J. Comput. Sci. Eng. **3** (2011), no. 6, 2415–2419.

[23] S. K. Khasim, P. V. Subbaiah and L. H. Prasad, *Image compression using absolute moment block truncation code*, Int. J. Electronics Commun. Technol. **3** (2012), 2415–2419.

[24] C.-C. Lai, *A digital watermarking scheme based on singular value decomposition and tiny genetic algorithm*, Digit. Signal Process. **21** (2011), no. 4, 522–527.

[25] S. J. Leon, *Linear Algebra with Applications*, MacMillan Publishing Company, New York, 1996.

[26] P. Li, X.-x. Tao, J.-q. Zhang and X.-z. Wang, *Retrieval module to choose satellite images by considering the demand of disaster mitigation*, in: *2011 International Conference on Remote Sensing, Environment and Transportation Engineering*, 685–688, IEEE, Nanjing, China, 2011.

[27] MathWorks, *MATLAB Compiler: User's Guide*, 2002.

[28] N. Mou, B. Liu, X. Lu, L. Zhang and W. Liu, *Semantic-based remote sensing images intelligent service on grid environment*, in: *2009 First International Workshop on Database Technology and Applications*, 291–294, IEEE, Wuhan, China, 2009.

[29] M. A. Otair and M. A. Alkhalayleh, *A new lossless method of image compression by decomposing the tree of Huffman technique*, Int. J. Imag. Robot. **15** (2015), no. 2, 79–96.

[30] A. Plaza, J. M. Bioucas-Dias, A. Simic and W. J. Blackwell, *Foreword to the special issue on hyperspectral image and signal processing*, in: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **5** (2012), no. 2, 347–353.

[31] A. Ranade, S. S. Mahabalarao and S. Kale, *A variation on SVD based image compression*, Image Vis. Comput. **25** (2007), no. 6, 771–777.

[32] A. M. Rufai, G. Anbarjafari and H. Demirel, *Lossy image compression using singular value decomposition and wavelet difference reduction*, Digit. Signal Process. **24** (2014), 117–123.

[33] K. Sahnoun and N. Benabadji, *Satellite image compression technique using noise bit removal and discrete wavelet transform*, Int. J. Imag. Robot. **15** (2015), no. 3, 57–64.

[34] G. W. Stewart, *Error and perturbation bounds for subspaces associated with certain eigenvalue problems*, SIAM Rev. **15** (1973), 727–764.

[35] ———, *On the early history of the singular value decomposition*, SIAM Rev. **35** (1993), no. 4, 551–566.

[36] J. G. Sun, *Invariant subspaces and generalized invariant subspaces I: Existence and uniqueness theorems*, Math. Numer. Sinica **2** (1980), no. 1, 1–13.

[37] ———, *Invariant subspaces and generalized invariant subspaces II: Perturbation theorems*, Math. Numer. Sinica **2** (1980), no. 2, 113–123.

[38] J. Wang, N. Zheng, Y. Liu and G. Zhou, *Parameter analysis of fractal image compression and its applications in image sharpening and smoothing*, Signal Process. Image Commun. **28** (2013), no. 6, 681–687.

[39] M. J. Weinberger, G. Seroussi and G. Sapiro, *The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS*, IEEE Trans. Image Process. **9** (2000), no. 8, 1309–1324.

Peter Chang-Yi Weng
Department of Applied Mathematics, National Chiayi University, No. 300 Syuefu Rd., Chiayi City 600355, Taiwan
E-mail address: pweng1206@gmail.com

Frederick Kin Hing Phoa
Institute of Statistical Science, Academia Sinica, 128 Academia Road, Section 2, Nankang, Taipei 11529, Taiwan
E-mail address: fredphoa@stat.sinica.edu.tw