DOI: 10.1515/auom-2015-0024
An. Şt. Univ. Ovidius Constanţa

VERSITA
Vol. 23(2),2015, 45–58

Block Power Method for SVD Decomposition

A. H. Bentbib and A.Kanber

Abstract

We present in this paper a new method to determine the k largest singular values and their corresponding singular vectors for real rectangular matrices $A \in \mathbf{R}^{n\times m}$. Our approach is based on using a block version of the Power Method to compute an k-block SVD decomposition: $A_k = U_k \Sigma_k V_k^T$, where Σ_k is a diagonal matrix with the k largest non-negative, monotonically decreasing diagonal $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_k$. U_k and V_k are orthogonal matrices whose columns are the left and right singular vectors of the k largest singular values. This approach is more efficient as there is no need of calculation of all singular values. The QR method is also presented to obtain the SVD decomposition.

1 Introduction

The singular value decomposition SVD is a generalization of the eigen-decomposition used to analyse rectangular matrices(see [7]). It is an important useful tool in many applications, including mathematical models in economics, physical and biological processes (see [3]). For example, one way of estimating the eigenvalues of covariance matrix is singular value decomposition (SVD). Covariance matrix is used by many researchers in image processing applications. Singular value analysis has also been applied in data mining applications and by search engines to rank documents in very large databases, including the Web (see [6]). Several numerical methods for calculating eigenvalues of a real matrix is based on the asymptotic behaviour of successive power of this matrix. This is the case, for instance, of the so called power method. Using

Key Words: Eigenvalues, Power Method, Singular, values.
2010 Mathematics Subject Classification: 15A18, 65F15, 65F35
Received: Nov, 2013.
Accepted: June, 2014.

a block version of the power method, we obtain a new algorithm for computing the singular values and corresponding singular vectors for a matrix. The paper is organized as follows. In section 2 we recall the power method to find the largest eigenvalue in magnitude of a square matrix and the corresponding eigenvector (see [4] and [8]). The power method is adapted to compute the largest singular value in section 3. In section 4, a block power method for computing the SVD decomposition for a real matrix is given. In section 5, the very useful QR method (see [2]) is applied to compute the SVD decomposition. The proofs of the presented methods are given and numerical examples are provided to illustrate the effectiveness of the proposed algorithms.

2 Power Method

2.1 Classical Power Method

Computing eigenvalues and eigenvectors of matrices play an important roles in many applications in the physical sciences. For example, they play a prominent role in image processing applications. Measurement of image sharpness can be done using the concept of eigenvalues. The power method is one of the oldest techniques for finding the largest eigenvalue in magnitude and its corresponding eigenvector. We describe below the theory of the method. Briefly, given a square matrix A, one picks a vector v and forms the sequence : $v, Av, A^2v, \ldots$ In order to produce this sequence, it is not necessary to get the powers of A explicitly, since each vector in the sequence can be obtained from the previous one by multiplying it by A. The sequence converges in direction of the dominant eigenvector. The proof of the convergence is usually given if the eigenvalues of A are ordered so that

$$|\lambda_1|>|\lambda_2| \geq \ldots \geq |\lambda_n|.$$

However, the method has some disadvantages such as when the largest eigenvalue is multiple or when we may to compute other eigenvalues. To obtain the smallest eigenvalue in magnitude, one consider powers of A^{-1}, a method which is called the inverse power method or inverse iteration.

2.2 Algorithm

Algorithm 2.2: Power Method

1. **Input :** A square matrix $A \in \mathbf{R}^{n\times n}$ and a vector $u^{(0)} \in \mathbf{R}^n$,

2. **Output :** The largest eigenvalue λ_1 and the associated eigenvector

3. for $k = 1, 2, \cdots$ (repeat until convergence)

$$w^{(k)} = Au^{(k-1)},\ u^{(k)} = \frac{w^{(k)}}{\|w^{(k)}\|},\ \lambda^{(k)} = u^{(k)T}\left(Au^{(k)}\right)$$

2.3 Convergence

Let us examine the convergence of the power iteration in the case when $A \in \mathbf{R}^{n\times n}$ is diagonalizable with p distinct eigenvalues $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_p|$ $(p \le n)$. Let $u^{(0)} \in \mathbf{R}^n$, such that $\|u^{(0)}\| = 1$. Since A is diagonalizable, then $\mathbf{R}^n = E_{\lambda_1} \oplus \cdots \oplus E_{\lambda_p}$ where E_{λ_i} is the eigenspace of A corresponding to the eigenvalue λ_i. We set $u^{(0)} = u_1 + u_2 \ldots + u_p$ where $u_i \in E_{\lambda_i}$. By induction, we obtain

$$\begin{aligned} u^{(k)} &= \frac{1}{\gamma_k}\left(\lambda_1^k u_1 + \sum_{j=2}^{p} \lambda_j^k u_j\right) \textit{ with } \gamma_k = \left\|\lambda_1^k u_1 + \sum_{j=2}^{p} \lambda_j^k u_j\right\| \\ &= \frac{\lambda_1^k}{\gamma_k}\left(u_1 + \sum_{j=2}^{p}\left(\frac{\lambda_j}{\lambda_1}\right)^k u_j\right) \end{aligned}$$

Since $\|u^{(k)}\| = 1$, then

$$\frac{|\lambda_1|^k}{\gamma_k} = \frac{1}{\|u_1 + \sum_{j=2}^{p}\left(\frac{\lambda_j}{\lambda_1}\right)^k u_j\|}$$

that leads us to prove that

$$\lim_{k\to+\infty} \frac{|\lambda_1|^k}{\gamma_k} = \frac{1}{\|u_1\|}$$

and then

$$\lim_{k\to+\infty} = \frac{u_1}{\|u_1\|} \quad \textit{and } \lambda^{(k)} = u^{(k)T}\left(Au^{(k)}\right) \to \lambda_1$$

2.4 Block Power Method

In this section we give a block version of the power method to compute the first s eigenvalues of a square matrix. The proposed algorithm, used the QR factorization at the normalization step. [4] and [5].

Algorithm 2.4: Block Power Method

1. Input : A square matrix $A \in \mathbf{R}^{n\times n}$, and a block of s vectors $V \in \mathbf{R}^{n\times \mathbf{s}}$.

2. Output : A diagonal matrix Λ with the first s eigenvalues

3. . **While** $err > precision$

$B = AV$, $B = QR$ (QR factorization),

$V = Q(:, 1 : \mathbf{s})$ and $\Lambda = R(1 : \mathbf{s}, :)$. (Here Matlab notation is used)

$err = \|AV - V\Lambda\|$;

End

2.5 Numerical Example :

In this example, we tested the numerical block method given in Algorithm 2.4 compared with Matlab function *eig*. The rectangular matrix $A \in \mathbf{R}^{n\times m}$ is defined as $A = Q\Sigma Q^T$ where Q is a random orthogonal matrix. We compute relative error occurred when computing eigenvalues.

$$\Sigma = diag([40, 40, 40, 32, 15, 2, 1.5, 1]),\ n = 80,\ rank(A) = 8$$

eigenvalues	Alg 2.4	Matlab
40	$0.3553e-015$	$0.0533e-014$
40	$0.1776e-015$	$0.1421e-014$
40	$0.1776e-015$	$0.1421e-014$
32	$0.2220e-015$	$0.3331e-014$
15	$0.2368e-015$	$0.0474e-014$
2	$0.2220e-015$	$0.0222e-014$
1.5	$0.2961e-015$	$0.1480e-014$
1	$0.4441e-015$	$0.2440e-014$

3 SVD Power Method

In this section we give an algorithm to compute the SVD decomposition for a real matrix $A \in \mathbf{R}^{n\times m}$. We know that there exists an orthogonal real matrix $U \in \mathbf{R}^{n\times n}$, an orthogonal matrix $V \in \mathbf{R}^{m\times m}$ and a positive diagonal matrix Σ =diag$(\sigma_1, \sigma_2, \ldots, \sigma_r, 0...) \in \mathbf{R}^{m\times n}$ such that $A = U\Sigma V^T$ ($r = rank(A)$). Let us set $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_m]$ where $(u_i)_{1\le i\le n} \in \mathbf{R}^n$ and $(v_j)_{1\le j\le m} \in \mathbf{R}^m$. We obtain $A = \sum_{k=1}^{r} \sigma_k u_k v_k^T$, $Au_k = \sigma_k v_k$ and $A^T v_k = \sigma_k u_k$ for $k = 1, \cdots, r$.

3.1 Algorithm

We present here an algorithm that compute the dominant singular value $\sigma_1 = \sigma_{\max}$ of a rectangular real matrix and its associate right and left singular vector. The convergence proof of the presented algorithm is given below.

Algorithm 3.1: SVD Power Method

Input : A matrix $A \in \mathbf{R}^{n\times m}$, a vector $v^{(0)} \in \mathbf{R}^m$,
Output : The first singular value σ_1 and
the corresponding right and left singular vector: $Av = \sigma_1 u$
for $k = 1, 2, \cdots$ (repeat until convergence)
While $error > \epsilon$ do :
$w^{(k)} = Av^{(k-1)}$, $\alpha_k = \|w^{(k)}\|$, $u^{(k)} = \alpha_k^{-1} w^{(k)}$
$z^{(k)} = A^T u^{(k)}$, $\beta_k = \|z^{(k)}\|$, $v^{(k)} = \beta_k^{-1} z^{(k)}$
$error := \|Av^{(k)} - \beta_k u^{(k)}\|$ and $\sigma_1 := \beta_k$
EndDo

3.2 Convergence

It is known that there exists orthonormal bases $U = [u_1, \dots, u_n]$ and $V = [v_1, \dots, v_m]$, respectively, of $\mathbf{R}^n$ and $\mathbf{R}^m$, such that $A = \sum_{j=1}^{r} \sigma_j u_j v_j^T$. Let $v^{(0)} \in \mathbf{R}^m$, $v^{(0)} = \sum_{j=1}^{m} y_j v_j$ where $y_j = v_j^T v^{(0)}$. If $w^{(1)} = Av^{(0)}$ and $\alpha_1 = \|w^{(1)}\|^{-1}$, then we set $u^{(1)} = \alpha_1 w^{(1)}$, $z^{(1)} = A^T u^{(1)}$ and $v^{(1)} = \beta_1 z^{(1)}$ where $\beta_1 = \|z^{(1)}\|^{-1}$. We repeat the process until convergence is obtained.
Indeed, since $A = \sum_{j=1}^{r} \sigma_j u_j v_j^T$ and $v^{(0)} = \sum_{j=1}^{m} y_j v_j$, then $w^{(1)} = \sum_{j=1}^{r} \sigma_j y_j u_j$, $u^{(1)} = \alpha_1 \sum_{j=1}^{r} \sigma_j y_j u_j$, $z^{(1)} = A^T u^{(1)} = \alpha_1 \sum_{j=1}^{m} \sigma_j^2 y_j v_j$ and $v^{(1)} = \alpha_1 \beta_1 \sum_{j=1}^{r} \sigma_j^2 y_j v_j$.
By induction we obtain

$$v^{(k)} = \delta_{2k} \sum_{j=1}^{r} \sigma_j^{2k} y_j v_j\text{` } \textit{and } u^{(k)} = \delta_{2k+1} \sum_{j=1}^{r} \sigma_j^{2k+1} y_j u_j$$

Where δ_{2k} and δ_{2k+1} are the corresponding normalization factors (δ_{2k} and δ_{2k+1} are positive). We can easily see that $v^{(k)}$ and $u^{(k)}$ converge to the first, right and left singular vector, respectively.
Since $\|u^{(k)}\|^2 = \delta_{2k+1}^2 \sum_{j=1}^{r} \sigma_j^{4k+2} y_j^2 = 1$ and $\|v^{(k)}\|^2 = \delta_{2k}^2 \sum_{j=1}^{r} \sigma_j^{4k} y_j^2 = 1$, then

$$\frac{\|u^{(k)}\|^2}{\|v^{(k)}\|^2} = 1 = \sigma_1^2 \left(\frac{\delta_{2k+1}^2}{\delta_{2k}^2} \right) \left(\frac{C + \sum_{j=\mu_1+1}^{r} (\frac{\sigma_j}{\sigma_1})^{4k+2} \alpha_j^2}{C + \sum_{j=\mu_1+1}^{r} (\frac{\sigma_j}{\sigma_1})^{4k} \alpha_j^2} \right)$$

Where μ_1 is the multiplicity of the singular value σ_1 and $C = \sum_{j=1}^{\mu_1} y_j^2$. Thus $\frac{\delta_{2k+1}}{\delta_{2k}} \longrightarrow \sigma_1$ and since $Av^{(k)} = \frac{\delta_{2k+1}}{\delta_{2k}} u^{(k)}$, then $\|Av^{(k)} - \sigma_1 u^{(k)}\| \longrightarrow 0$.

4 Block SVD Power Method

The main goal in this section is to give a block iterative algorithm that computes the singular value decomposition. The idea is based on the technique used in the block power method. From a block-vector $V^{(0)} \in \mathbf{R}^{m\times s}$, we construct two block-vector sequences $V^{(k)} \in \mathbf{R}^{m\times s}$ and $U^{(k)} \in \mathbf{R}^{n\times s}$ that converges respectively to the s first right and left singular vectors corresponding to singular values $\sigma_1 \geq \ldots \geq \sigma_s$.

4.1 Algorithm

Algorithm 4.1: Block *SVD* Power Method

Input : A matrix $A \in \mathbf{R}^{n\times m}$, a block-vector $V = V^{(0)} \in \mathbf{R}^{m\times s}$ and a tolerance *tol*
Output : An orthogonal matrices $U = [u_1, \ldots, u_s] \in \mathbf{R}^{n\times s}$, $V = [v_1, \ldots, v_s] \in \mathbf{R}^{m\times s}$ and a positive diagonal matrix $\Sigma_1 = diag\,(\sigma_1, \sigma_2, \ldots, \sigma_s)$ such that : $AV = U\Sigma_1$
While $err > tol$ **do**
  $AV = QR$ (factorization QR), $U \longleftarrow Q(:, 1:s)$ (the s first vector colonne of Q)
  $A^T U = QR$, $V \longleftarrow Q(:, 1:s)$ and $\Sigma_1 \longleftarrow R(1:s, 1:s)$
  $err = \|AV - U\Sigma_1\|$
End

4.2 Convergence

Let s be an integer such that $r = qs$ where r is the rank of A and

$$\sigma_1 \geq \ldots \geq \sigma_s > \sigma_{s+1} \geq \ldots \geq \sigma_{qs} > 0$$

the singular values of A. We can write A as $A = \sum_{i=1}^{q} U_i \Sigma_i V_i^T$ where Σ_i is a diagonal matrix with nonzero, monotonically decreasing diagonal $\sigma_{(i-1)s+1} \geq \sigma_{(i-1)s+2} \geq \ldots \geq \sigma_{is} > 0$. U_i and V_i are the orthogonal matrices whose columns are respectively the corresponding left and right singular vectors.

Let $V^{(0)} \in \mathbf{R}^{m\times s}$, $V^{(0)} = \sum_{i=1}^{q} V_i X_i + V^{(0)*}$, where
$span\left(V^{(0)*}\right) \subseteq span\left\{v_{r+1}, v_{r+2}, \cdots, v_m\right\} = \ker\{A\}$. We have

$$W^{(0)} = AV^{(0)} = U_1\Sigma_1 X_1 + \sum_{\mathbf{i=2}}^{q} U_i\Sigma_i X_i.$$

Suppose that the component $X_1 = I_s$, then

$$\begin{aligned}
&AV^{(0)} = U^1\mathbf{R}_1 \left(\mathbf{QR}\ \textit{factorization}\right)\\
&= U_1\Sigma_1 + \sum_{\mathbf{i=2}}^{q} U_i\Sigma_i X_i\\
&U_1^T U^{(1)}\mathbf{R}_1 = \Sigma_1\ \textit{that prove}\ \mathbf{R}_1\ \textit{is non singular and then}\\
&U^{(1)} = U_1\Sigma_1\mathbf{R}_1^{-1} + \sum_{\mathbf{i=2}}^{q} U_i\Sigma_i X_i\mathbf{R}_1^{-1}
\end{aligned}$$

and

$$\begin{aligned}
&A^T U^{(1)} = V^{(1)}\mathbf{R}_2(\mathbf{QR}\ \textit{factorization})\\
&= V_1\Sigma_1^2\mathbf{R}_1^{-1} + \sum_{\mathbf{i=2}}^{q} V_i\Sigma_i^2 X_i\mathbf{R}_1^{-1}\\
&V_1^T V^{(1)}\mathbf{R}_2 = \Sigma_1^2\mathbf{R}_1^{-1},\ \mathbf{R}_2\ \textit{is non singular}\\
&V^{(1)} = V_1\Sigma_1^2\mathbf{R}_1^{-1}\mathbf{R}_2^{-1} + \sum_{\mathbf{i=2}}^{q} V_i\Sigma_i^2 X_i\mathbf{R}_1^{-1}\mathbf{R}_2^{-1}
\end{aligned}$$

and so on, if we note $\mathbf{N}_t = \mathbf{R}_1^{-1}\mathbf{R}_2^{-1}\cdots\mathbf{R}_t^{-1}$, at step k we have

$$\begin{aligned}
AV^{(k-1)} &= U^{(k)}\mathbf{R}_{2k-1}(\mathbf{QR}\ \textit{factorization})\\
&= U_1\Sigma_1^{2k-1}\mathbf{N}_{2(k-1)} + \sum_{\mathbf{i=2}}^{q} U_i\Sigma_i^{2k-1} X_i\mathbf{N}_{2(k-1)}\\
U^{(k)} &= U_1\Sigma_1^{2k-1}\mathbf{N}_{2k-1} + \sum_{\mathbf{i=2}}^{q} U_i\Sigma_i^{2k-1} X_i\mathbf{N}_{2k-1}
\end{aligned}$$

and

$$\begin{aligned}
A^T U^{(k)} &= V^{(k)}\mathbf{R}_{2k}(\mathbf{QR}\ \textit{factorization})\\
&= V_1\Sigma_1^{2k}\mathbf{N}_{2k-1} + \sum_{\mathbf{i=2}}^{q} V_i\Sigma_i^{2k} X_i\mathbf{N}_{2k-1}\\
V^{(k)} &= V_1\Sigma_1^{2k}\mathbf{N}_{2k} + \sum_{\mathbf{i=2}}^{q} V_i\Sigma_i^{2k} X_i\mathbf{N}_{2k}
\end{aligned}$$

$U^{(k)}$ and $V^{(k)}$ are orthogonal matrices, then

$$I_s = \left(U^{(k)}\right)^T U^{(k)} = \mathbf{N}_{2k-1}^T \Sigma_1^{4k-2} \mathbf{N}_{2k-1} + \sum_{\mathbf{i=2}}^{q} \mathbf{N}_{2k-1}^T X_i^T \Sigma_i^{4k-2} X_i \mathbf{N}_{2k-1}$$
$$I_s = \left(V^{(k)}\right)^T V^{(k)} = \mathbf{N}_{2k}^T \Sigma_1^{4k} \mathbf{N}_{2k} + \sum_{\mathbf{i=2}}^{q} \mathbf{N}_{2k}^T X_i^T \Sigma_i^{4k} X_i \mathbf{N}_{2k}$$

by left and right-factoring, we obtain

$$I_s = \mathbf{N}_{2k-1}^T \Sigma_1^{2k-1} \left(I_s + \sum_{\mathbf{i=2}}^{q} \Sigma_1^{-2k+1} X_i^T \Sigma_i^{4k-2} X_i \Sigma_1^{-2k+1} \right) \Sigma_1^{2k-1} \mathbf{N}_{2k-1}$$
$$I_s = \mathbf{N}_{2k}^T \Sigma_1^{2k} \left(I_s + \sum_{\mathbf{i=2}}^{q} \Sigma_1^{-2k} X_i^T \Sigma_i^{4k} X_i \Sigma_1^{-2k} \right) \Sigma_1^{2k} \mathbf{N}_{2k}$$

Since $\left\|\Sigma_1^{-1}\right\| = \frac{1}{\sigma_s}$ and $\|\Sigma_i\| = \sigma_{(i-1)s+1}$ then,

$$\begin{array}{rcl} \left\|\Sigma_1^{-p} X_i^T \Sigma_i^{2p} X_i \Sigma_1^{-p}\right\| & \leq & \|\Sigma_i\|^{2p} \left\|\Sigma_1^{-1}\right\|^{2p} \|X_i\|^2 \\ & \leq & \left(\frac{\sigma_{(i-1)s+1}}{\sigma_s}\right)^{2p} \|X_i\|^2 \longrightarrow_{p\to\infty} 0 \end{array}$$

Thus

$$\lim_{p\longrightarrow\infty} \left(\mathbf{N}_p^T \Sigma_1^p\right)\left(\Sigma_1^p \mathbf{N}_p\right) = \lim_p \longrightarrow \infty \left(\Sigma_1^p \mathbf{N}_p\right)^T \left(\Sigma_1^p \mathbf{N}_p\right) = I_s.$$

Moreover, the matrix $\Sigma_1^p \mathbf{N}_p$ is triangular with positive diagonal entries, then $\lim_{p\longrightarrow\infty} \Sigma_1^p \mathbf{N}_p = \lim_{p\longrightarrow\infty} \mathbf{N}_p^{-1} \Sigma_1^{-p} = I_s$. Otherwise

$$\begin{array}{rcl} A^T U^{(k)} \left(\mathbf{N}_{2k-1}^{-1} \Sigma_1^{-(2k-1)}\right) \Sigma_1^{-1} & = & A^T U^{(k)} \mathbf{R}_{2k}^{-1} \left(\mathbf{N}_{2k}^{-1} \Sigma_1^{-2k}\right) \\ & = & V^{(k)} \left(\mathbf{N}_{2k}^{-1} \Sigma_1^{-2k}\right) \\ & = & V_1 + \sum_{\mathbf{i=2}}^{q} V_i \Sigma_i^{2k} X_i \Sigma_1^{-2k} \longrightarrow_{k\to\infty} V_1 \end{array}$$

$$\begin{array}{rcl} A V^{(k)} \left(\mathbf{N}_{2k}^{-1} \Sigma_1^{-2k}\right) \Sigma_1^{-1} & = & A V^{(k)} \mathbf{R}_{2k+1}^{-1} \left(\mathbf{N}_{2k+1}^{-1} \Sigma_1^{-(2k+1)}\right) \\ & = & U^{(k+1)} \left(\mathbf{N}_{2k+1}^{-1} \Sigma_1^{-(2k+1)}\right) \\ & = & U_1 + \sum_{\mathbf{i=2}}^{q} U_i \Sigma_i^{2k+1} X_i \Sigma_1^{-(2k+1)} \longrightarrow_{k\to\infty} U_1 \end{array}$$

That implies that $\lim_{k\to\infty} V^{(k)} = V_1$, $\lim_{k\to\infty} U^{(k)} = U_1$ and $\lim_{k\to\infty} \mathbf{R}_{2k} = \lim_{k\to\infty} \mathbf{R}_{2k+1} = \Sigma_1$.

5 The QR Method for SVD

Our main goal in this section is to give an iterative algorithm that compute the singular value decomposition. The idea is based on the QR method.

5.1 Algorithm

Algorithm 5.1: The QR Method for SVD

> **Input :** A matrix $A \in \mathbf{R}^{n\times m}$
> **Output :** The Singular Value Decomposition
> Initialization $T_0 = A$ and $S_0 = A^T$
> **For** $k = 1, 2, \cdots$ (repeat until convergence)
> $T_{k-1} = U_k R_k$, $S_{k-1} = V_k Z_k$ (QR Factorization)
> $T_k = R_k V_k$ and $S_k = Z_k U_k$

The algorithm given above is nothing but the QR method applying to the symmetric matrix $M = \begin{pmatrix} 0_n & A \\ A^T & 0_m \end{pmatrix}$ to compute eigenvalues of M which are nothing but the singular values of A. In deed, by setting $T_0 = A$, $S_0 = A^T$ and $M_0 = \begin{pmatrix} 0_n & T_0 \\ S_0 & 0_m \end{pmatrix}$, we have

For $k = 1, 2, \cdots$

$$M_{k-1} = \begin{pmatrix} 0_n & T_{k-1} \\ S_{k-1} & 0_m \end{pmatrix} = \begin{pmatrix} U_k & 0 \\ 0 & V_k \end{pmatrix} \begin{pmatrix} 0_n & R_k \\ Z_k & 0_m \end{pmatrix} \text{ (}QR\text{ Factorization)}$$

$$M_k = \begin{pmatrix} 0_n & T_k \\ S_k & 0_m \end{pmatrix} = \begin{pmatrix} 0_n & R_k \\ Z_k & 0_m \end{pmatrix} \begin{pmatrix} U_k & 0 \\ 0 & V_k \end{pmatrix}$$

5.2 Numerical examples

We compared and tested the numerical results obtained by Algorithm 4.1 with Matlab *svd* function. Let $A \in \mathbf{R}^{n\times m}$ be a rectangular matrix defined as : $A = Q\Sigma U^T$ where Q and U are random orthogonal matrices. We give below relative errors occurred when computing the singular values. We also compare the CPU time. The started block-vector in Algorithm 4.1 is given by $V = V^{(0)} = eye(m, s)$ (Matlab notation). The results are given from Algorithm 4.1 after only at most $k = 2$ iterations. We stopped the algorithm 4.1 whenever the error of the reduction $err = \|AV - U\Sigma\|$ is smaller than that achieved by Matlab *svd* function.

Example 1:

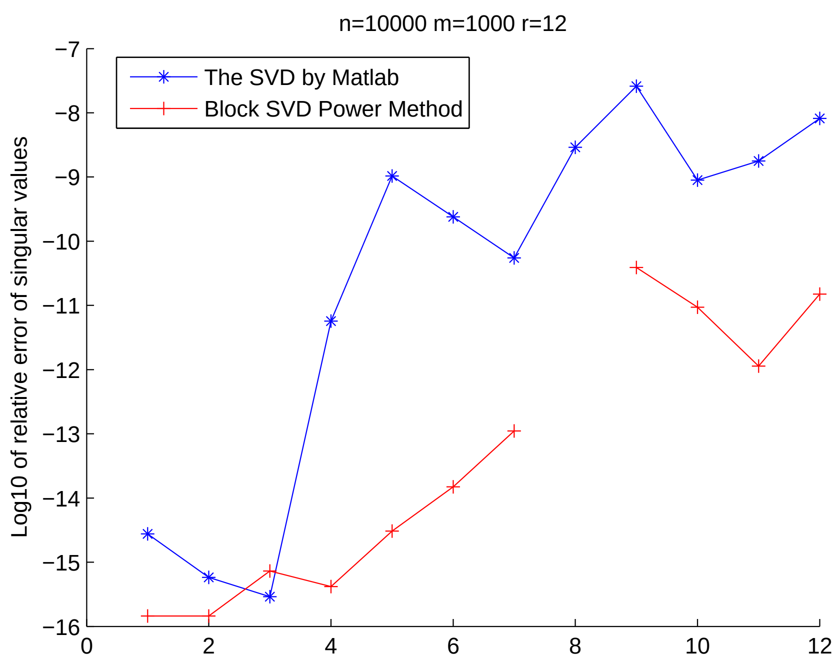
$$\Sigma = diag(10^5, 10^5, 10^5, 10^{-1}, 10^{-1}, 10^{-3}, 10^{-3}, 10^{-3}, 10^{-5}, 10^{-5}, 10^{-5}, 10^{-5})$$
$$n = 10000, \quad m = 1000, \quad s = rank(A) = 12,$$

In this example, the error $\|AV - U\Sigma\|$ obtained using Matlab *svd* function is equal to $6.0570e - 011$. After $k = 2$ iterations of algorithm 4.1 we obtain $\|AV - U\Sigma\| = 5.5582e - 011$.

	Alg 4.1	Matlab *svd*
CPU time	22.9491	55.0144

Relative errors occurred when computing the singular values:

Singular values	Alg 4.1	Matlab *svd*
10^{-5}	$9.6055e - 12$	$1.3281e - 07$
10^{-3}	$2.5977e - 13$	$3.4005e - 07$
10^{-1}	$9.7145e - 16$	$5.7468e - 12$
10^{5}	$1.4552e - 16$	$4.3656e - 16$



Example 2:

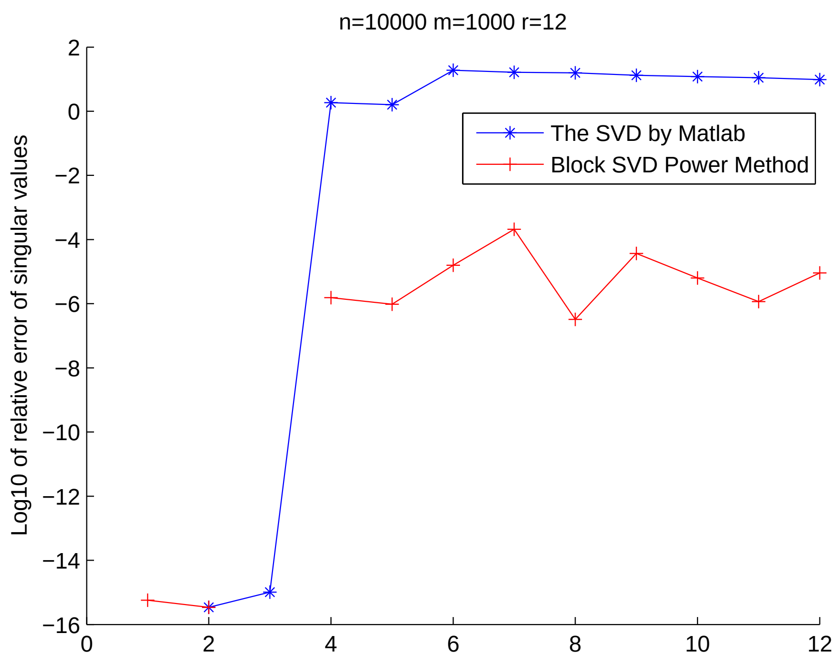
$$\Sigma = diag(10^3, 10^3, 10^3, 10^{-12}, 10^{-12}, 10^{-13}, 10^{-13}, 10^{-13}, 10^{-13}, 10^{-13}, 10^{-13}, 10^{-13})$$
$$n = 10000, \quad m = 1000, \quad s = rank(A) = 12,$$

Here, the error $\|AV - U\Sigma\|$ obtained using Matlab *svd* function is equal to $2.8961e - 012$. After only $k = 1$ iterations of algorithm 4.1 we obtain $\|AV - U\Sigma\| = 1.1372e - 012$.

	Alg 4.1	Matlab *svd*
CPU time	3.1021	53.4363

Relative errors occurred when computing the singular values:

Singular values	Alg 4.1	Matlab *svd*
10^{-13}	$2.6894e - 06$	12.6631
10^{-12}	$5.6916e - 07$	3.4664
10^3	$3.4106e - 16$	$9.0949e - 16$



Example 3:

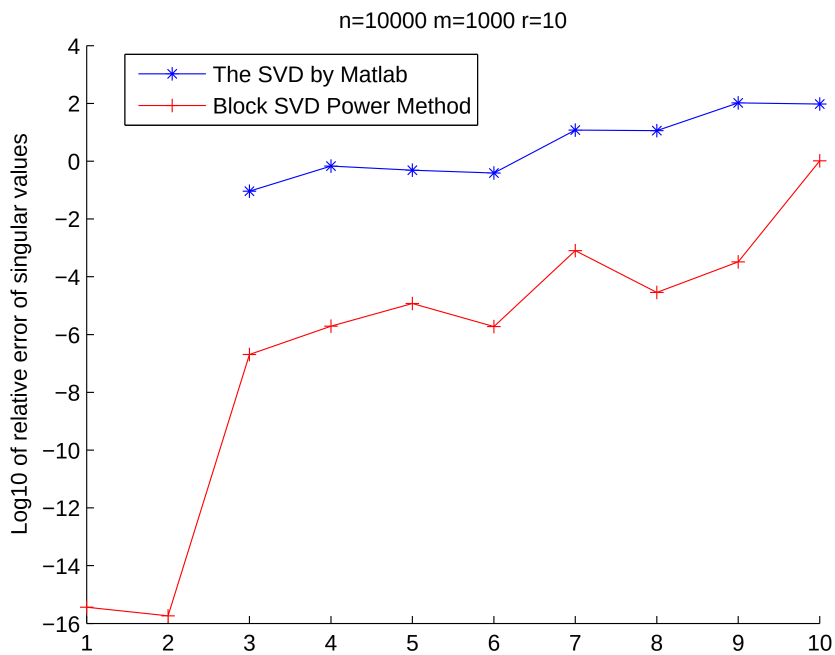
$$\Sigma = diag(10^4, 10^4, 10^{-11}, 10^{-11}, 10^{-12}, 10^{-12}, 10^{-13}, 10^{-13}, 10^{-14}, 10^{-14})$$
$$n = 10000, \quad m = 1000, \quad s = rank(A) = 10,$$

Here, the error $\|AV - U\Sigma\|$ obtained using Matlab *svd* function is equal to $1.6384e-011$. After $k = 2$ iterations of algorithm 4.1 we obtain $\|AV - U\Sigma\| = 1.3313e-011$.

	Alg 4.1	Matlab *svd*
CPU time	6.1170	49.7370

Relative errors occurred when computing the singular values:

Singular values	Alg 4.1	Matlab *svd*
10^{-14}	$6.8008e-04$	$3.8380e+01$
10^{-13}	$3.8362e-05$	$6.7545e+00$
10^{-12}	$6.8116e-07$	$1.1270e-01$



Example 4:

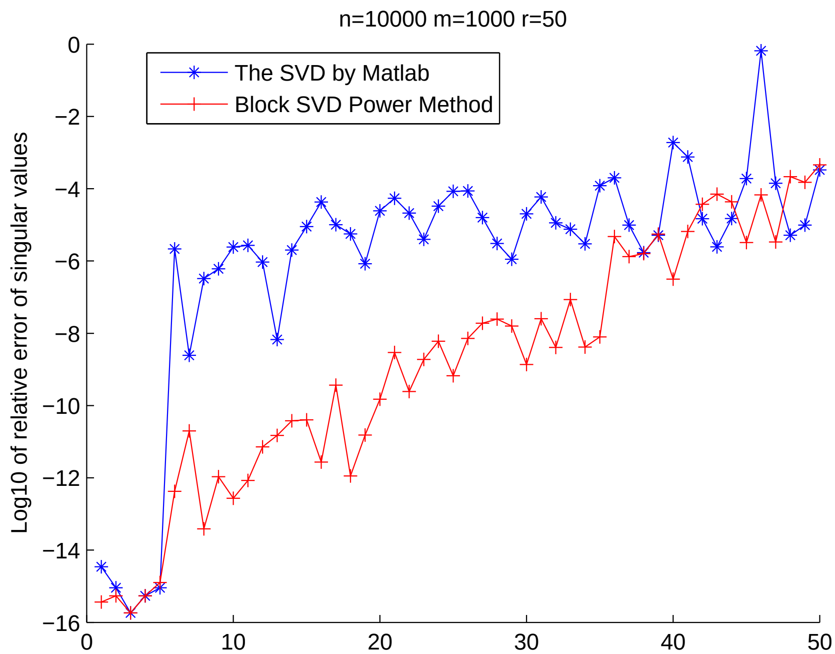
$$\begin{aligned}\Sigma &= diag\,(\sigma_1, \sigma_2, \ldots, \sigma_{50}) \;\; such\ that \\ \sigma_1 &= \sigma_2 = \cdots = \sigma_5 = 10^4, \\ \sigma_{5i+1} &= \sigma_{5i+2} = \cdots = \sigma_{5(i+1)} = 10^{-(4+i)},\ for\ i = 1 \ldots 9\end{aligned}$$

And in this example, the error $\|AV - U\Sigma\|$ obtained using Matlab *svd* function is equal to $1.5080e - 010$. After $k = 2$ iterations of algorithm 4.1 we obtain $\|AV - U\Sigma\| = 8.1825e - 011$.

	Alg 4.1	Matlab *svd*
CPU time	22.3978	54.3242

Relative errors occurred when computing the singular values:

Singular values	Alg 4.1	Matlab *svd*
10^{-13}	$2.4255e - 03$	$8.9669e + 00$
10^{-12}	$9.0965e - 06$	$2.5287e + 00$
10^{-11}	$2.1635e - 06$	$2.2190e - 03$



6 Conclusion

A new approach using block version of the power method is used for the estimation of singular values. The proposed method is very simple and effective for computing all singular values. The numerical examples show the effectiveness of the presented method. The computational time and relative errors corresponding to the computed singular values are considerably reduced.

References

[1] A. G. Akritas , G. I. Malaschonok , P. S. Vigklas *The SVD-Fundamental Theorem of Linear Algebra* , Nonlinear Analysis: Modelling and Control, 2006, Vol. 11, No. 2, 123–136.

[2] J.G.F. Francis *The QR Transformation - a unitary analogue to the LR transformation*, Computer journal. Volume 4, 1961. Part 1 pages 265-271, part II pages 332-345.

[3] H. Gaidhane,V. Hote , V.Singh *A New Approach for Estimation of Eigenvalues of Images* , International Journal of Computer Applications (0975 – 8887) Volume 2 6 – No. 9 , Ju ly 2011 .

[4] H. Golub, A. v.d. Vorst, *Eigenvalue computation in the 20th century* , Journal of Computational and Applied Mathematics 123 (2000) 35–65.

[5] J. Higham, *QR factorization with complete pivoting and accurate computation of the SVD* , Linear Algebra and its Applications 309 (2000) 153–174.

[6] V.Kobayashi , G.Dupret , O.King, H.Samukawa *Estimation of Singular Values of Very Large Matrices Using Random Sampling* , Computers and Mathematics with Applications 42 (2001) 1331-1352.

[7] R.Mathias, G.W.Stewart *A block QR Algorithm an the Sinular value Decomposition* ,UMIACS-TR-91-38 CS-TR 2626 (1992).

[8] D. Stewart *A New Algorithm for the SVD of a long product matrices and the stability of products* Electronic Translation on Numeracul analysis Volume 5 pp 29-47, June 1997 .

[9] G.Strang, *Introduction to applied mathematics* , Wellesly-Cambridge press (1986).

A. H. Bentbib and A. Kanber,
Department of Mathematics,
Laboratoire LAMAI Facultés des Sciences et Techniques-Guéliz,
BP 549 Marrakech, Morocco.
Email: a.bentbib@uca.ma, ahbentbib@gmail.com, kanber@uca.ma