行政院國家科學委員會補助專題研究計畫

# ■成果報告　□期中進度報告

（計畫名稱） The research and application of fast principle component analysis and singular value decomposition on huge data set

計畫類別：■個別型計畫　□整合型計畫
計畫編號：NSC 98－2115－M－004－006－MY2
執行期間：　98年10月01日 至 100年 9月30日

執行機構及系所：政治大學應用數學系

計畫主持人：曾正男
共同主持人：
計畫參與人員：

成果報告類型(依經費核定清單規定繳交)：□精簡報告 ■完整報告

本計畫除繳交成果報告外，另須繳交以下出國心得報告：
□赴國外出差或研習心得報告
□赴大陸地區出差或研習心得報告
■出席國際學術會議心得報告
□國際合作研究計畫國外研究報告

處理方式：除列管計畫及下列情形者外，得立即公開查詢
　　□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

中　華　民　國　100 年 10　月 27　日

# 國科會補助專題研究計畫成果報告自評表

請就研究內容與原計畫相符程度、達成預期目標情況、研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）、是否適合在學術期刊發表或申請專利、主要發現或其他有關價值等，作一綜合評估。

1. 請就研究內容與原計畫相符程度、達成預期目標情況作一綜合評估

■達成目標

□ 未達成目標（請說明，以100字為限）

□ 實驗失敗

□ 因故實驗中斷

□其他原因

說明:

2. 研究成果在學術期刊發表或申請專利等情形:

論文: ■已發表 □未發表之文稿 ■撰寫中 □無

專利: □已獲得 ■申請中 □無

技轉: □已技轉 □洽談中 ■無

其他: （以100字為限）

3. 請依學術成就、技術創新、社會影響等方面，評估研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）（以500字為限）

學術成就:

我們開發了一套在特定條件下可以快速計算大資料矩陣主成份和奇異值分解的方法，在技術創新的部份是有別於使用統計的方式來估計矩陣行向量的主成份，速度上較現有的統計方法慢卻比傳統方法快許多，在精準度上比統計方法高但比傳統方法低。所以，我們所完成的大資料矩陣奇異值計算法 是一個在速度與精準度上都折衷的方法。

技術創新:

此方法是利用矩陣切割的方式將大矩陣切成列數很少的超長矩陣，藉此我們在一個狹長的矩陣上可以用傳統的奇異值算法來計算這一個部份矩陣的奇異值，再用這些算好的奇異值組合起來，計算原本矩陣該有的奇異值。

社會影響:

此算法我們在三個領域上嘗試應用，但只在互動式網頁的推薦系統上獲得較好的表現。利用此方法，我們可以在網頁所累積的大資料上，作快速又精準的資料探勘。並且此方法很適合用在平行計算領域，對於追求高速運算的時代，此方法提供統計方法之外的一種選擇。

Project title:

# The research and application of fast principle component analysis and singular value decomposition on huge data set

Contents:

Report info:

1. Preface

This project is started in 2009/10/1. Two undergraduate assistants are hired. One is Mr. Wei-Da Lai and the other is Mr. Yuan-Lin Su. Mr. Su is supper in Linux system management. He helped me to purchase our computer equipment and build up our server system. Because the budget of the computer equipment was not fully supported by NSC, we bought a notebook instead of the server. Because the data matrix is too large such that we cannot compare the traditional SVD method with our implement method, the college of science of NCCU support us a powerful server to complete this project. The server is purchased in Aug 2011, so we have to extend this project one two months that we have enough time to set up the new machine, run the relative codes and collect the final results.

The part time assistants of this project are very unstable. Many of them are quit automatically when they want to prepare the master class entrance examination. Only one student, Mr. Yeng-Hong Chen, has not left. He is going straight up to the master degree of our department and will join my next NSC project.

Because this project is “The research and application of fast principle component analysis and singular value decomposition on huge data set”, we try to use NHI dataset to present our result. However, when we obtain the NHI data, the data are full missing value and the data is not permit by the National Health Research Institutes. Hence, we only show our implementation in image classification problem. To overcome the missing value problem, we have model the form as a rotation problem in N-dimensional space. If we know how to operate the N-dimensional rotation, the problem will be solved easily.

Up to data, we have successful develop a fast principle component analysis and singular value decomposition method for growing large data set. This fast method obtains an approximation solution of PCA and SVD result. When the data is going updated, this fast method can update the PCA and SVD result immediately. We can show that in the simulation data for the rank of matrix is much smaller than the matrix size, our method is better than other PCA and SVD methods. However, when the rank is near the matrix size, there is no advantage in our method.

To search the application fields is an important task. We have tried our method to the image recognition problem, the classification in the microarray data and the web recommendation system. In the image recognition problem, if we only use our method to find the first two or three components to represent image data in the 2D or 3D dimension, the classification result is fault. We have to increase the dimension in the low dimensional space for accuracy classification. In the classification of microarray sample problem, PCA will remove much useful information in the array data. Thus, the classification is fault too. In the web recommendation system, our method obtains better result. This might be the web data fit our data requirement, that is the huge matrix with low rank.

## 2. Research purposes

The research purpose of this project is to develop a fast algorithm in singular value decomposition. Because singular value decomposition is cost in computation complexity, when data is increasing, the

traditional SVD method becomes infeasible. So we are looking for a fast algorithm to make SVD is feasible for large dataset.

In our previous study, when the number of independent vectors is fewer then the square root of number of total vectors, we have a fast singular value decomposition method. If the number of independent vectors is close to the number of total vectors and singular values decay rapidly, we also have a fast approximation SVD algorithm if the essential rank is also smaller.

We are looking for a good update method that when the data is growing continuously. To re-compute the PCA and SVD in the continuously growing data is challenged, we need to find a good method to update an approximated PCA and SVD solution. If there exist fast updated method, we can use this method to the real time application.

Because the PCA and the SVD is a very fundamental technique in linear algebra, we can use fast SVD algorithm to improve computational cost. There should be many applications in this case. We will focus on image classification problem, microarray sample classification and web recommendation system.

## 3. Literature reviews

In calculating the singular values decomposition of a matrix, the traditional fast method is proposed by Golub and Kahan [7]. The approach to compute the SVD is transforming the matrix to an upper bi-diagonal form by householder transformation. Given a matrix $A$ which is an m-by-n matrix. Without lost of general, we assume that $m$ is greater than $n$. The householder transform find two unitary matrix $P$ and $Q$, such that $A=PJQ^*$. After obtain the upper bi-diagonal matrix $J$. There are many method to produce the SVD of tri-diagonal matrix.

One possible method is proposed by Lanczos, they extend the matrix $J$ to $\tilde{J} = \begin{pmatrix} 0 & J \\ J^* & 0 \end{pmatrix}$, then transform, then using the variable transform to transfer this matrix to a real, nonnegative, tri-diagonal matrix. Then we can use Sturm sequences that are proposed by Wilkinson [8] to obtain an accurate SVD result.

Another approach to computing the singular value of $J$ is to compute the eigenvalues of $J^*J$. Note again that $J^*J$ is a tri-diagonal hermitian matrix there exists a diagonal unitary matrix $\Delta$ such that $\Delta$ $J^*J\Delta=K$ is a real, symmetric, positive semi-definite, tri-diagonal matrix. Then the eigenvalue of $K$ can be compute by the Sturm sequence algorithm.

4. Methodology

It is crucial to develop a fast algorithm of Principle component analysis (PCA) and Singular value decomposition. They are fundamental techniques of linear algebra and statistics. There are many modern applications based on these two tools, such as linear discriminate analysis and multidimensional scaling analysis. In recently years, digital information increases rapidly. Many analytic methods based on PCA and SVD are challenged by the computational cost of huge data processing.

MDS is a method to represent the high dimensional data into the low dimensional configuration. When the data configuration is Euclidean, MDS is similar to principle component analysis (PCA), which can remove inherent noise with its compact representation of data. So the classical MDS method has O(n3) complexity. In 2008, we implemented the classical to reduce the computational cost from O(n3) to O(n). We have proved when the data dimension is significantly smaller than the number of data, classical MDS has fast linear algorithm.

The main ideal of fast MDS is using statistical resampling to divide data into overlapping subsets. We perform the classical MDS on each subset and get the configuration. Then we use the overlapping information to combine each configuration of subset to the configuration of whole data.

Assume $X_1$ and $X_2$ are matrices in which the columns are the two coordinates of the overlapping points obtained by applying MDS to two data sets, and $\bar{X}_1$ and $\bar{X}_2$ are the means of columns of $X_1$ and $X_2$, respectively. In order to use the same orthogonal basis to represent these vertices, we apply QR factorization to $X_1 - \bar{X}_1 \mathrm{i}^T$ and $X_2 - \bar{X}_2 \mathrm{i}^T$, so that $X_1 - \bar{X}_1 \mathrm{i}^T = Q_1 R_1$ and $X_2 - \bar{X}_2 \mathrm{i}^T = Q_2 R_2$. Since these two coordinates represent the same points, the triangular matrices $R_1$ and $R_2$ should be identical when there is no noise in $X_1$ and $X_2$. Due to randomness of the sign of columns of $Q_i$ in QR factorization, the sign of columns of $Q_i$ should be adjusted according to the corresponding diagonal element of $R_i$, so that the signs of diagonal elements of $R_1$ and $R_2$ become the same.

After the sign of column of $Q_i$ is modified, we conclude

$$Q_1^T (X_1 - \bar{X}_1 \mathrm{i}^T) = Q_2^T (X_2 - \bar{X}_2 \mathrm{i}^T).$$

Furthermore, we can obtain

$$X_1 = Q_1 Q_2^T X_2 - Q_1 Q_2^T (\bar{X}_2 \mathrm{i}^T) + \bar{X}_1 \mathrm{i}^T. \quad (0.1)$$

That is, the unitary operator is $U = Q_1 Q_2^T$ and the shifting operator is $b = -Q_1 Q_2^T \bar{X}_2 + \bar{X}_1$. Since the key processing of finding this affine mapping is QR operation, the computational cost is O(n3) too. Therefore, the cost O(n3) computation is controlled by the number of samples in each subgroup. The key proof of computational cost is as the following:

Assume that there are N points in a data set, $N_I$ is the number of points in each intersection region, and $N_g$ is the number of points in each group. When we split N points into K overlapping groups, we have

$$KN_g - (K-1)N_I = N,$$

and then we have $K = \dfrac{(N - N_I)}{(N_g - N_I)} \sim O(N)$.

For each group, we apply CMDS to compute the coordinates of group data, which costs $O(N_g^3)$ computation time. At each overlapping region, we apply QR factorization to compute the affine transform, which costs $O(N_I^3)$ computation time. Since the lower bound of $N_I$ is p+1, we can roughly assume that $N_g = \alpha p$ for some constant $\alpha$. Then the total computation time is about

$$\frac{N-p}{(\alpha-1)p}O(\alpha^3 p^3) + \frac{N-\alpha p}{(\alpha-1)p}O(p^3) \sim O(p^2 N).$$

When p<<N, the computation time $O(p^2 N)$ is smaller than $O(\sqrt{N}N)$, the computation time of the fast MDS method proposed by (Morrison et al., 2002). Because MDS is similar to principle component analysis (PCA) when the data configuration is Euclidean, we can implement SC-MDS method to fast PCA in the constrain p<<N. When fast PCA is implemented, we can develop fast SVD in the same criterion.

To implement SC-MDS method to fast PCA, the first challenge is to make sure the true dimension p is significantly smaller than the number of samples N. When p is given and significantly smaller than N, it is easy to transform SC-MDS to fast PCA by changing the row vector to the column form. So, the first step is to make sure the dimension of data.

The principles of SVD and PCA are very similar. Since the PCA starts from decomposing the covariance matrix of data, it can be considered as adjusting the center of mass of a row vector to zero. Then SVD is performed for the matrix after the tensor product of shifted vectors. If the row data is distributed at the data with a center of mass equal to zero, then the eigenvector of the row vector decomposed by the SVD will be equal to the base decomposed by the PCA. Our question is: if we have the result of PCA, is there a fast algorithm to produce the SVD result without re-computing the eigenvectors of the whole data? The following is the mathematical analysis for this problem.

Let X is a column matrix of data. $\widetilde{X} = X - \overline{X} \cdot \mathbf{1}^T$, where $\overline{X}$ is the average of columns of X and $\mathbf{1}$ is a vector that all of its elements are one. Hence, the row mean of $\widetilde{X}$ is zero. Assume we have the PCA result of X, that is $\widetilde{X}\widetilde{X}^T = SV^2S^T$, where the columns of S are the eigenvectors of $\widetilde{X}\widetilde{X}^T$ and the $rank(\widetilde{X}) = r$ is much smaller than p and n. And $\widetilde{X} = SVD^T$ for some orthogonal matrix D. We observe

that $rank(X) = rank(\widetilde{X})$ or $rank(X) = rank(\widetilde{X}) + 1$, which depends on whether $\overline{X}$ is spanned by S. If $\overline{X}$ is spanned by S, then

$$X = \widetilde{X} + \overline{X} \cdot \dot{1}^T = SVD^T + S \cdot c \cdot \dot{1}^T = S(VD^T + c \cdot \dot{1}^T),$$

Where c is the coefficient that $\overline{X}$ be represented by S, i.e., $\overline{X} = S \cdot c$.

If $(VD^T + c \cdot \dot{1}^T)$ can be represented as $(QV_1D_1^T)$, where Q is a r-by-r unitary matrix, $V_1$ is a r-by-r diagonal matrix and $D_1$ is a r-by-n orthogonal matrix, then we have

$$X = S(QV_1D_1^T) = (SQ)V_1D_1^T = S_1V_1D_1^T.$$

Because Q is an unitary matrix, $S_1 = SQ$ is automatically an orthogonal matrix too. Hence, we have the SVD of X, if we can decompose $(VD^T + c \cdot \dot{1}^T)$ to $(QV_1D_1^T)$.

Checking the matrix size of $(VD^T + c \cdot \dot{1}^T)$, we can see that to compute the SVD of $(VD^T + c \cdot \dot{1}^T)$ is not a big problem. $(VD^T + c \cdot \dot{1}^T)$ is a r-by-n matrix. Under our assumption, r is much smaller than n. So, we can easily compute the SVD of $(VD^T + c \cdot \dot{1}^T)$.

On the other hand, if $\overline{X}$ is not spanned by S, the analysis becomes

$$X = \widetilde{X} + \overline{X} \cdot \dot{1}^T = \begin{bmatrix} S & s \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} D^T \\ 0 \end{bmatrix} + \begin{bmatrix} S & s \end{bmatrix} \cdot c \cdot \dot{1}^T = \begin{bmatrix} S & s \end{bmatrix} \left( \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} D^T \\ 0 \end{bmatrix} + c \cdot \dot{1}^T \right),$$

Where s is a unit vector that is derived by $\overline{X}$ filtered out the components on S. That is

$$s = \frac{\overline{X} - SS^T\overline{X}}{\left\| \overline{X} - SS^T\overline{X} \right\|}.$$

Using the same concept in the case of $\overline{X}$ is spanned by S, we find the SVD of $\left( \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} D^T \\ 0 \end{bmatrix} + c \cdot \dot{1}^T \right) = QV_1D_1^T$.

Then

$X = \begin{bmatrix} S & s \end{bmatrix} Q V_1 D_1^T = S_1 V_1 D_1^T$,

where $S_1 = \begin{bmatrix} S & s \end{bmatrix} Q$ is another orthogonal matrix and the SVD of X is completed. Note that when $\overline{X}$ is not spanned by S, the SVD of X finds r+1 orthogonal column vectors in column space of X.

The most challenge is to implement this method into the approximation fast PCA and SVD when the dimension p is not easy to estimate. How to control the error in an acceptable region is our goal. Moreover, to choose the optimal $N_l$ and $N_g$ in SC-MDS is also the further research.

We look for the solution when the data is updated constantly and we need to compute SVD continuously. Instead of scanning all the data again, we try to use the previous SVD result together with the new updated data to compute the next SVD.

Let A be an m-by-n matrix, where m is the number of variables and n is the number of samples. And we assume that both m and n are huge. When new data comes in, we collect these new data to form a column matrix which is denoted by U. Assume that we have the singular value decomposition of A, that is

$$A = Z \Sigma V^T,$$

where $Z \in M_m(\Re), V \in M_n(\Re)$ are orthogonal and $\Sigma$ is a diagonal. Since the data gets updated, the data matrix becomes

$$A_1 = \begin{bmatrix} A \mid B \end{bmatrix}.$$

To compute the singular value decomposition of $A_1$, we need to compute the eigenvalue and eigenvector of $A_1 A_1^T$.

We can represent the column matrix $B$ by $B = ZC$, where $C$ is the coefficient matrix of $B$ with columns of $Z$ as the basis. Since $Z$ is orthogonal, the coefficient matrix $C$ can be computed easily by $C = Z^T B$. Then we have

$$
\begin{aligned}
A_1 A_1^T &= [A \mid ZC][A \mid ZC]^T \\
&= AA^T + ZC(ZC)^T \\
&= Z(\Sigma^2 + CC^T)Z^T \qquad (1) \\
&= ZU\hat{\Sigma}^2 U^T Z^T \\
&= Z_1 \hat{\Sigma}^2 Z_1^T .
\end{aligned}
$$

Note that the matrix $\Sigma^2 + CC^T$ is positive symmetric. Using the spectrum theorem, we can decompose this matrix into $U\hat{\Sigma}^2 U^T$. Because the matrix $U$ is unitary, $Z_1$ is $Z$rotated by $U$.

When the matrix size of $A$ is huge, the computational cost of SVD is high. If the data is constantly growing, it is difficult to compute the singular value decomposition of $A_1$ in real time. Therefore, we look for an approximated solution with fast method.

Let $Z = [z_1, z_2, \cdots, z_m]$. If the new updated data B has only the components in $\{z_1, z_2, \cdots, z_r\}$, where $r << m$, then only $r$-dimensional space will be perturbed by this new data. This is proved as follows.

**Theorem** *Let $A = Z\Sigma V^T$. Assume that $A_1 = [A \mid B]$, where $B$ has no component in $i$-th column of $Z$ for $i > r$. Then the singular value decomposition of $A_1$ has the same spectrum $\sigma_i$ and singular vector $z_i, v_i$ for $i > r$.*

**Proof**: Let

$$
A = \begin{pmatrix} \overset{r}{Z_1} & \overset{p-r}{Z_2} \end{pmatrix} \begin{pmatrix} \overset{r}{\Sigma_1} & \overset{p-r}{0} \\ 0 & \Sigma_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix},
$$

Where $Z_1$ and $V_1$ are the first $r$ columns of $Z$ and $V$. Because $B$ has no component in $Z_2$, $B = Z_1 C$ for some $C$. Then $A_1 A_1^T$ can be written as

$$
\begin{aligned}
A_1 A_1^T &= \left[A \mid B\right]\left[A \mid B\right]^T \\
&= AA^T + UU^T \\
&= \begin{pmatrix} Z_1 & Z_2 \end{pmatrix} \begin{pmatrix} \Sigma_1^2 + CC^T & 0 \\ 0 & \Sigma_2^2 \end{pmatrix} \begin{pmatrix} Z_1^T \\ Z_2^T \end{pmatrix} \\
&= \begin{pmatrix} Z_1 & Z_2 \end{pmatrix} \begin{pmatrix} U\hat{\Sigma}_1 U^T & 0 \\ 0 & \Sigma_2^2 \end{pmatrix} \begin{pmatrix} Z_1^T \\ Z_2^T \end{pmatrix} \qquad (2) \\
&= \begin{pmatrix} Z_1 U & Z_2 \end{pmatrix} \begin{pmatrix} \hat{\Sigma}_1^2 & 0 \\ 0 & \Sigma_2^2 \end{pmatrix} \begin{pmatrix} U^T Z_1^T \\ Z_2^T \end{pmatrix} \\
&= \begin{pmatrix} \hat{Z}_1 & Z_2 \end{pmatrix} \begin{pmatrix} \hat{\Sigma}_1^2 & 0 \\ 0 & \Sigma_2^2 \end{pmatrix} \begin{pmatrix} \hat{Z}_1^T \\ Z_2^T \end{pmatrix}
\end{aligned}
$$

where $U$ is unitary. We can see that the terms $\Sigma_2$ and $Z_2$ do not change if $Z_2^T B = 0$. Thus, the singular value decomposition of $A_1$ has the same spectrum $\sigma_i$ and $z_i$ for $i > r$. Moreover, the $v_i$ for $i > r$ are unchanged too.

In many applications, we are only concerned with the first few spectrums and eigenvectors. For example, in high dimensional data visualization, we only consider the first two or three eigenvectors, that is $r = 2$ or 3. In this case, the new updated data should be have component in the space that spanned by $Z_2$. We are therefore interested in the performance of the approximated solution compared with the true solution when we only retained the first $r$ components of $B$ every time we updated the new data by the previous method. If the performance decays slowly, we can daringly compute only three or four components in many SVD-based methods and the result in low dimensional space is still quite reliable. So, we need to understand in what kind of conditions, the approximated solution is stable.

Now, we analyze the effect of the perturbation of a matrix in its eigenvalues and eigenvectors. Let matrix $A$ be real symmetric and $A = S\Lambda S^T$, where $S$ is unitary, such that $S^{-1} = S^T$. A matrix change $\Delta A$ produces changes in eigenvalues and eigenvectors, which are denoted by $\Delta\Lambda$ and $\Delta S$ respectively. Because $S$ is orthogonal, $AS = S\Lambda$. Similarly, we have

$$
\left(A + \Delta A\right)\left(S + \Delta S\right) = \left(S + \Delta S\right)\left(\Lambda + \Delta\Lambda\right).
$$

The above equation can be represented by

$$A(\Delta S)+(\Delta A)S = S(\Delta\Lambda)+(\Delta S)\Lambda, \qquad (3)$$

when ignoring the small terms $(\Delta A)(\Delta S)$ and $(\Delta S)(\Delta\Lambda)$.

We multiply equation (3) by $S^T$, then we have

$$\begin{aligned}\Delta\Lambda &= S^T(\Delta A)S + S^T A(\Delta S) - S^T(\Delta S)\Lambda \\ &= S^T(\Delta A)S + \Lambda S^T(\Delta S) - S^T(\Delta S)\Lambda. \end{aligned} \qquad (4)$$

Because the diagonal terms of $\Lambda S^T(\Delta S)$ and $S^T(\Delta S)\Lambda$ are the same, the diagonal part of $S^T(\Delta A)S$ is what we are looking for. Applying this concept to matrix $\Sigma^2 + CC^T$, $CC^T$ can be considered as $S^T(\Delta A)S$. We can conclude that if the maximal element of the absolute value of $CC^T$ is smaller than the difference between $\sigma_i - \sigma_{i+1}$ for $i = 1,\cdots,r$ then the order of columns of $S$ will not change. The first $r$ columns of $S + \Delta S$ can be approximated stably by the first $r$ columns of $S$. If $CC^T$ is too large such that the new spectrum $\hat{\sigma}_{r+1} > \hat{\sigma}_r$, the approximation solution that only use first $r$ components to update the new spectrum and singular vectors will fault by using $\hat{z}_{r+1}$ to replace $\hat{z}_r$. This conclusion will be demonstrated in the experimental result.

## 5. Experimental result

In this section, we show that our fast PCA and SVD method works well for big sized matrix with small rank. The simulated matrix is created by the product of two slender matrices. The size of the first matrix is p-by-r, and the second matrix is r-by-n. Then the product of these two matrixes is of size p-by-n and its rank is smaller than r. When p and n are large and r is much smaller than p and n, the simulated matrix satisfies our SCSVD condition. We pick p = 4000, n = 4000 and r = 50 as our fist example. The elements of the simulated matrix is generated from the normal distribution $N(0,1)$.

The average elapsed time of SCSVD is 3.98 seconds, while the economical SVD takes 16.14 seconds, If we increase the matrix to p = 20000, n = 20000 and the same rank r = 50, the elapsed time of economical SVD is 1209.92 seconds, but SCSVD is only 195.85 seconds. We observe that our SCSVD method demonstrates significant improvement.

Note that when the estimated rank used in SCSVD is greater than the real rank of data matrix, there is almost no error (except rounding error) between economic SVD and SCSVD. Figure 1 shows the

speed comparison between economical SVD (solid line) and SCSVD (dashed line) with square matrix size from 500 to 4000 by fixed rank 50. We also use fixed parameter $N_I = 51$ and $N_g = 2N_I$ in each simulation test. We can see that the computational cost of SVD follows the order 3 increase, compared with linear increase of SCSVD. The error between economical SVD and economical SVD, and that between SVD and SCSVD are shown in Figure 2. Because the results between economical SVD and SVD are very similar, we use solid line to represent the value of economical SVD and circle plot to represent SCSVD. The values in both Figure 1 and Figure 2 are the mean of the results from 100 repeated simulated matrices. The errors between SVD and economic SVD, and that between SVD and SCSVD are all under the $10^{-4}$ level. Thus, when the estimated rank of SCSVD is greater than the true rank, the accuracy of SCSVD is pretty much the same as SVD in the case of small rank matrix.

The purpose of the second simulation experiment is to observe the approximation performance of applying SCPCA to big full rank matrix. We generate random matrix with fixed number of columns and rows, say 1000. The square matrix is created by the form, $A_{p\times r}\cdot B_{r\times n}+\alpha E_{p\times n}$, where $r$ is the essential rank, $E$ is the perturbation and $\alpha$ is a small coefficient for adjusting the influence to the previous matrix. Such matrix can be considered as a big sized matrix with small rank added by a full rank perturbation matrix. We will show that our method works well for this type of matrices.

Figure 3 shows the error vs. estimated rank, where the error is computed by the difference between the original matrix and the composition of three matrices from SCSVD.

All the elements of matrices $A$, $B$ and $E$ are randomly generated from the normal distribution $N(0,1)$, where $\alpha = 0.01$ and the essential rank $r = 50$. We can see that when the estimated rank increases, the composition error decreases. Especially when the estimated rank is greater than the essential rank $r$, the composition error decays rapidly. Thus, it is important to make sure that the estimated rank is greater than the essential rank. In other words, when the estimated rank of SCSVD is smaller than the essential rank, our SCSVD result can be used as the approximated solution of SVD.

In the last experimental result, we will show that we can set the estimated rank $r = 3$, starting from the

SCSVD result and using the previous updating method to continuously update the new SVD. We will show that the performance of the first three components decays very slowly. Thus, many SVD-based modern techniques, for example, Fisher linear discrimination, Latent semantic analysis [5], eigentaste recommendation system [6], dimensional reduction, etc, become feasible even when dealing with huge data set.

We produce a series square random matrices $A$ with size n-by-n for n between 1000 and 3000. Then we decompose $A$ by SVD to obtain $A = Z\Sigma V^T$. We reset the diagonal terms of $\Sigma$ to be exponential decay, so that the data can simulate the meaningful data in the real world. The maximal spectrum is set to be $10^4$. Then we compose $A$ by the new diagonal matrix V. We use SCSVD with estimated rank 3, and the parameter $N_1 = \frac{n}{10}, N_g = 2N_1$.

We make 16 updates to the data, and each time we add 10% samples of original data. The new data is simulated from the normal distribution $N(0,1)$. We use our updating method to compute the first three new columns of $Z$ and compare it with the true SVD result. Let $a^{(t)}, b^{(t)}$ be the maximal and minimal element of the absolute values of $\hat{Z}_3^{(t)T} Z_3^{(t)}$ , respectively, where $\hat{Z}_3^{(t)}$ is the t-th updated $Z$ by our updating method taking only the first three columns, and $Z^{(t)}$ is the t-th updated $Z$ by normal SVD. If $a^{(t)}$ and $b^{(t)}$ are close to 1, the updated $Z$ derived by our updating method is very close to the true $Z$.

In Figure 4, we can see that both $a^{(t)}$ and $b^{(t)}$ are close to 1, and they decay very slowly as the matrix size increases. In Figure 4, every point is the average value of 32 repeating simulations.

In the application of our fast SVD method, we have try four types of data. The first is NHI data. However we have no permission to use this data for research. The second data is image classification data. We collect 10000 images with the size of 200*200. Then the dimension of every image is 40000. We transform each image to the DCT coefficient, and arrange the DCT coefficient as the vector shape. Then our data matrix is with the size of 40000-by-10000. We compute the SVD of this big matrix, find

the first three components and project each image to the low dimensional subspace that generated by these three components. The classification result is not well because the three dimensions only are not enough. If we increase the dimension of the subspace more than to 25, the classification result will become better.

We have try the third data that is microarray sample classification data. However, the data sample is too small and the class level is too much. We have to develop some new method to overcome this kind of data. That is the dimension is large but sample is very few.

At last we try our method to the web recommendation system. It obtains a good result in this application field. Then we try to apply our method in the patent of web recommendation technique. Because the application is filled, we did not present the detail in this report.

## 6. Conclusion

We proposed fast PCA and SVD methods derived from the technique of SCMDS method. The new PCA and SVD have the same accuracy as the traditional PCA and SVD ones when the rank of a matrix is much smaller than its matrix size. The results of applying SCPCA and SCSVD to a full rank matrix are also quite reliable when the essential rank of the matrix is much smaller than its matrix size.

In most information technology applications, the essential rank of a matrix is usually much smaller than its matrix size. In such cases, utilizing SCPCA or SCSVD in huge data applications will render good approximated results. Since the concept of split-and-combine is very similar to that of parallel computing, this SC-series methods (Splitand-combine series) can be easily implemented via parallel computing. Using our updating method for the growing data, we show that the approximated solution is very close to the actual solution, even when the estimated rank is as small as $r = 3$.

For the future work, we will focus on the cases when the data contains missing values. Our intuitive speculation is that the processing of splitting data should be somehow related to the locations where

the missing values occur. We believe that it would be an interesting topic worth further exploration.

## 7. References


[1] D. J. Hand, ”Discrimination and classification”, *Wiley Series in Probability and Mathematical Statistics, Chichester: Wiley*, 1981

[2] M. Cox, T. Cox ”Multidimensional scaling”, *Handbook of data visualization, Springer*, 2008

[3] A. Morrison, G. Ross and M. Chalmers, ”Fast multidimensional scaling through sampling, springs and interpolation”, *Information Visualization, Vol. 2 , Issue 1, pp*. 68 - 77, 2003

[4] J. Tzeng, H. Lu and W. Li, ”Multidimensional scaling for large genomic data sets”, *BMC Bioinformatics*, 9:179, 2008

[5] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R.A. Harshman, L. A. Streeter and K. E. Lochbaum, ”Information retrieval using a singular value decomposition model of latent semantic structure”, *Annual ACM conference on Research and Development in Information Retrieval*, pp. 465-480, 1988

[6] K. Goldberg, T. Roeder, D. Gupta and C. Perkins, ”Eigentaste: A constant time collaborative filtering method”, *Information Retrieval*, Springer, 2001

[7] G. Golub and W. Kahan, “Calculating the singular values and pseudo-inverse of a matrix”, *J. SIAM Numer. Anal. Ser. B, Vol.2, No. 2,* 1965

[8] Wilkinson, “Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection”, *Numer. Math*., 4 (1962), pp. 362-367.

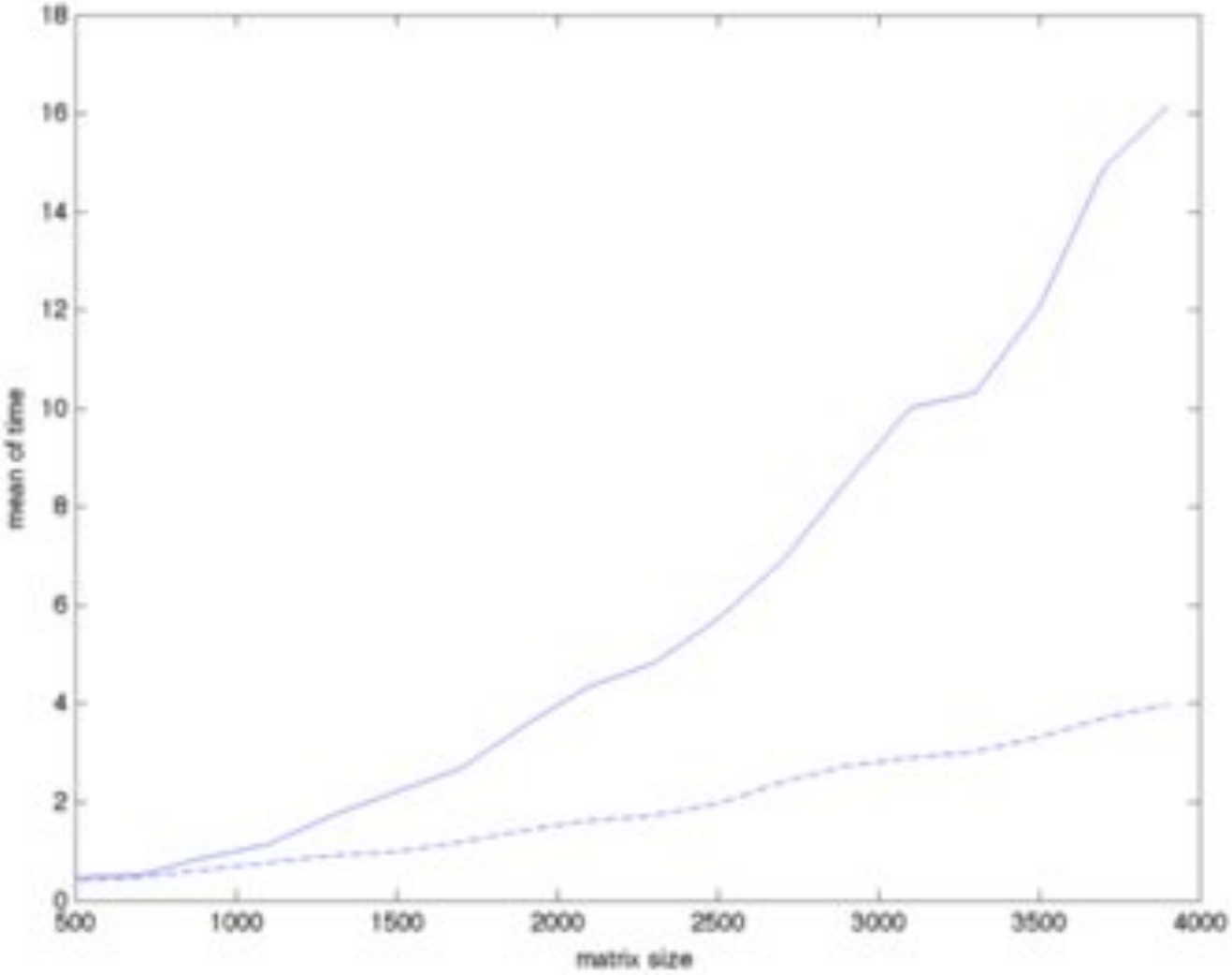


Figure 1. Comparison of the elapsed time between economical SVD (the solid line) and SCSVD (the dashed line).

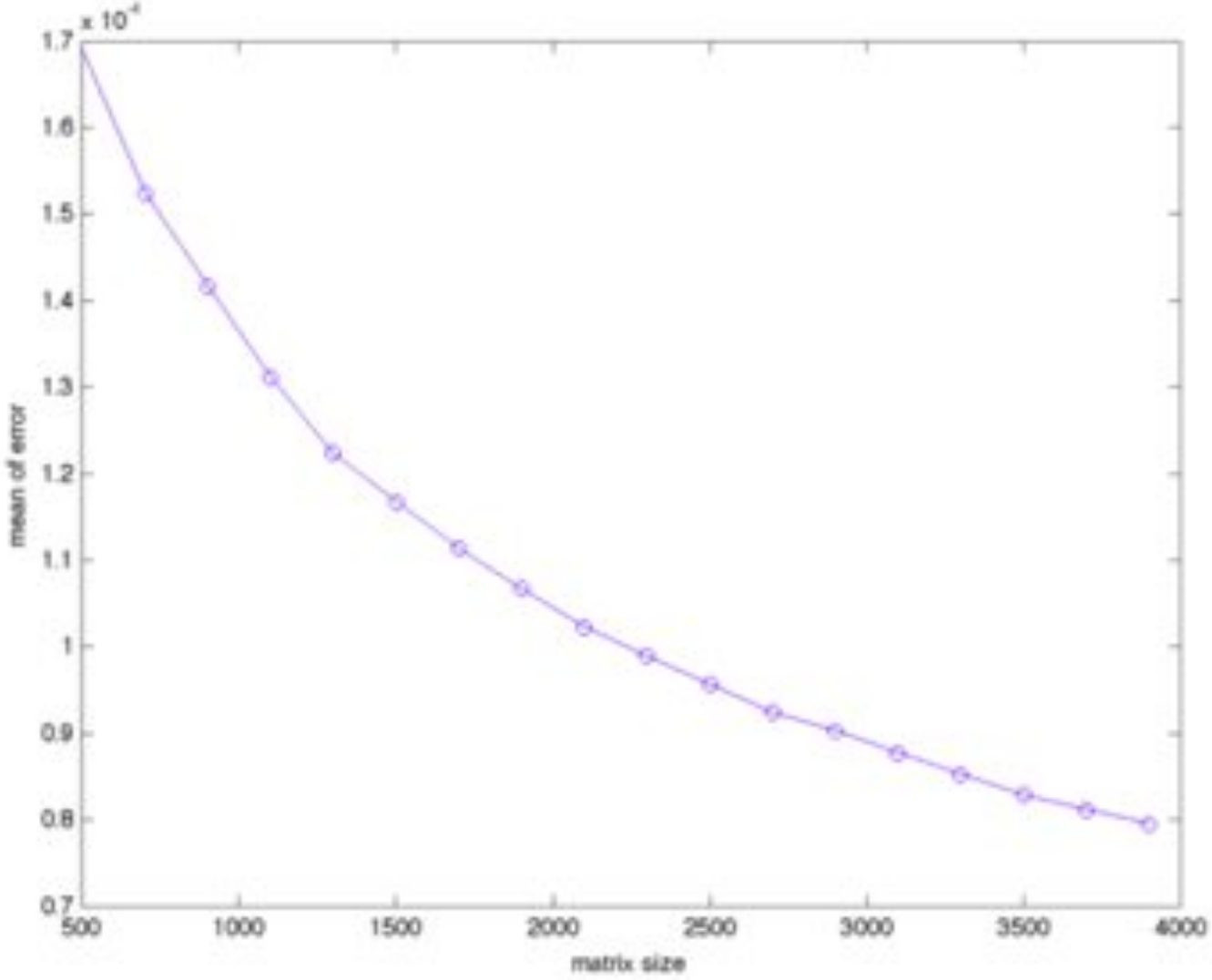


Figure 2. Comparison of the composition errors between economical SVD (the solid line) and SCSVD (the circle plot).


Figure 4. The orthogonality between approximated SVD and true SVD. The solid line is $a^{(t)}$ and the dashed line is $b^{(t)}$

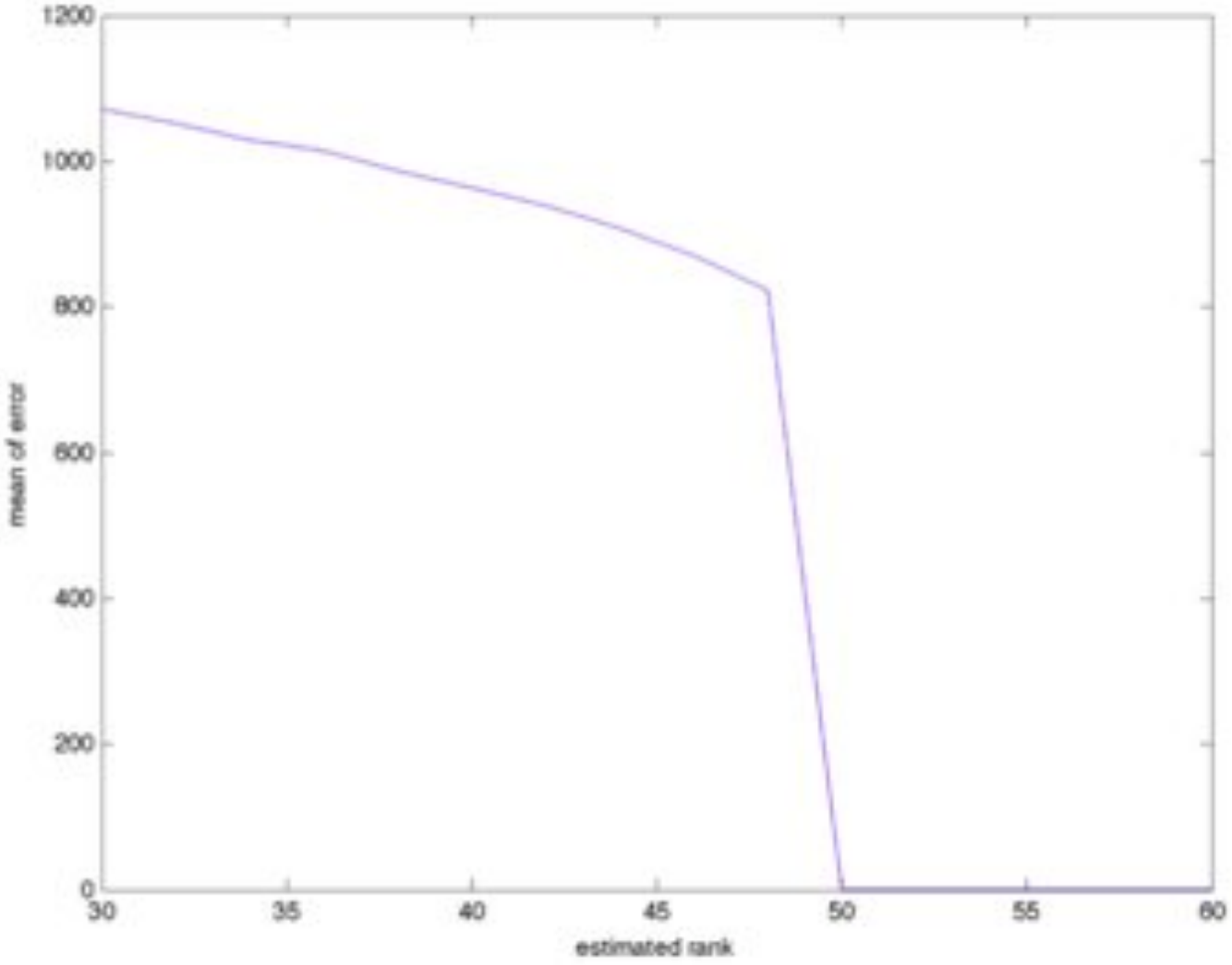


Figure 3. The effect of estimated rank to the composition error. The matrix size is 1000-by-1000 and its essential rank is 50 ($\alpha = 0.01$). When the estimated rank is greater than 50, there is almost no composition error