1
2007

This is to certify that the
dissertation entitled

A Rank-Revealing Method for Low Rank Matrices with
Updating, Downdating, and Applications

presented by

Tsung-Lin Lee

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in Department of Mathematics

Tien-Yien Li

Major Professor's Signature

7/17/07

Date

A Rank-Revealing Method for Low Rank Matrices with Updating, Downdating, and Applications

By

Tsung-Lin Lee

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Mathematics

2007

ABSTRACT

A Rank-Revealing Method for Low Rank Matrices with Updating, Downdating, and Applications

By

Tsung-Lin Lee

As one of the basic problems in matrix computation, rank-revealing has a wide variety of applications in scientific computing. Although singular value decomposition is the standard rank-revealing method, it is costly in both computing time and storage when the the rank or the nullity is low, and it is inefficient in updating and downdating when rows and columns are inserted or deleted. Consequently, alternative methods are in demand in those situations. Following up on a recent rank-revealing algorithm by Li and Zeng in the low nullity case, we present a new rank-revealing algorithm for low rank matrices with efficient and reliable updating/downdating capabilities. A comprehensive computing experiment shows the new method is accurate, robust, and substantially faster than existing rank-revealing algorithms.

To my parents.

ACKNOWLEDGMENTS

With great pleasure, I would like to express my sincere gratitude to Professor Tien-Yien Li, my dissertation advisor, for his guidance and support during my graduate study at Michigan State University. As his student, I have been privileged to see and learn many mathematical insights from him.

I would like to thank Professor Zhonggang Zeng for his inspiring discussions and suggestions.

I would also like to thank Professor Gang Bao, Professor Chichia Chiu, Professor Guowei Wei, and Professor Di Liu for their time and concern.

TABLE OF CONTENTS

LIST OF FIGURES **vii**

LIST OF TABLES **viii**

Introduction **1**

1 A Rank-Revealing Method **4**

1.1 Approximate ranks 4

1.2 The convergence theory 6

1.3 Computing the approxi-range and the approxi-rowspace 10

1.4 The overall algorithm 16

1.5 Numerical experiments 18

1.5.1 Type 1: Low approxi-rank with median noise level 18

1.5.2 Type 2: Increasing approxi-rank, fixed size and median noise level 19

1.5.3 Type 3: Decreasing gap, fixed size and median noise level . . . 20

1.5.4 Type 4: High noise level with increasing size 20

1.5.5 Type 5: Near zero noise level, low approxi-rank, large gap . . 21

2 Updating and Downdating Problems **23**

2.1 The USV-plus decomposition 23

2.2 Updating and downdating 25

2.2.1 Updating 26

2.2.2 Downdating 27

2.3 Numerical results on updating and downdating 32

2.3.1 Row-updating with increasing approxi-ranks 33

2.3.2 Row-updating without changing approxi-ranks 34

2.3.3 Row-updating with a small approxi-rank gap 34

2.3.4 Row-downdating without changing approxi-ranks 35

2.3.5 Row-downdating with decreasing approxi-ranks 35

3 Applications **37**

3.1 Information retrieval: latent semantic indexing 37

3.2 Image processing: saving storage of photographs 43

BIBLIOGRAPHY **46**

LIST OF FIGURES

1.1 Algorithm larank 22

2.1 Algorithm lrowup 28

2.2 Algorithm lrowdown..32

3.1 The photograph formation process: lattice partition and assignment 44

3.2 Rank 21 approximation of Figure 3.1 44

3.3 The original image and three lower rank approximation images 45

LIST OF TABLES

1.1 Results for Type 1 matrices.........19

1.2 Results for Type 2 matrices.........19

1.3 Results for Type 3 matrices 20

1.4 Results for Type 4 matrices.........21

1.5 Results for Type 5 matrices.........21

2.1 Results for row-updating with increasing approxi-ranks 33

2.2 Results for row-updating without changing approxi-ranks 34

2.3 Results for row-updating with a small approxi-rank gap 35

2.4 Results for row-downdating without changing approxi-ranks.........36

2.5 Results for row-downdating with decreasing approxi-ranks 36

3.1 Term-by-document matrix 39

3.2 Comparisons for a 3000 x 1400 term-by-document matrix from CRAN.........41

3.3 The retrieval results for three lower rank databases and the raw databases 42

3.4 Results for updating databases.........42

3.5 Results for downdating databases 43

3.6 Comparison results for a 480 x 640 fingerprint image matrix 43

Introduction

Rank-revealing appears frequently in scientific computing such as signal processing [13, 28, 36], information retrieval [3, 12, 38] and numerical polynomial algebra [10, 37]. While the singular value decomposition (SVD) is undoubtedly the most reliable method for determining the numerical rank of a matrix, it has drawbacks in certain situations. In particular, it is expensive when matrix size becomes large but either the rank or the nullity is low, and it is difficult to update or downdate when rows or columns are inserted or deleted. Alternative methods have been proposed for those situations, such as rank-revealing QR decomposition (RRQR) [5, 6, 7], rank-revealing two-sided orthogonal decompositions (UTV, or URV/ULV) [16, 34, 35], and rank-revealing LU decomposition (RRLU) [21, 26, 29]. In low-nullity cases, a new rank-revealing algorithm has been developed by Li and Zeng [24]. We follow up with a new rank-revealing algorithm for low rank matrices.

For a given $m \times n$ matrix A, our method determines the approximate rank of A by calculating the approximate range of A. We briefly outline the method as follows. With $m \geq n$ and $\text{rank}(A) = k$, let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$ be non-zero singular values of A along with $\mathbf{u}_i$ and $\mathbf{v}_i$ being the corresponding unit left singular vectors and unit right singular vectors associated with σ_i, respectively, for $i = 1, \cdots, k$. Unless otherwise mentioned, we shall always use "singular vector" to represent the *right* singular vector. Since $\mathbf{u}_j^\top A = \sigma_j \mathbf{v}_j^\top$ for $1 \le j \le k$,

$$A = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^\top = \mathbf{u}_1 \mathbf{u}_1^\top A + \cdots + \mathbf{u}_k \mathbf{u}_k^\top A.$$

Clearly,

$$A_1 := A - \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top = A - \mathbf{u}_1 \mathbf{u}_1^\top A$$

has the same set of singular values along with associated singular vectors as those of A except the largest singular value σ_1 of A is replaced with 0 as a singular value of A_1, and the second largest singular value σ_2 of A becomes the largest singular value of A_1. Thus, the rank of A_1 becomes $k-1$. Similarly, if $k \ge 2$, matrix

$$A_2 := A_1 - \mathbf{u}_2 \mathbf{u}_2^\top A = A - \mathbf{u}_1 \mathbf{u}_1^\top A - \mathbf{u}_2 \mathbf{u}_2^\top A$$

has the same set of singular values of A except σ_1 and σ_2 are replaced with 0 and the rank of A_2 is reduced to $k-2$. For the problem of finding the approximate rank, namely the number of singular values larger than a prescribed threshold $\theta > 0$ (see Definition 1 in §1.1), we begin by finding a unit vector $\tilde{\mathbf{u}}_1$ in the *approxi-range*, namely, the subspace spanned by the left singular vectors of A associated with singular values larger than the threshold $\theta > 0$. This task can be accomplished efficiently by applying the power iteration on $AA^\top$ with a proper stopping criterion. We must emphasize here that we do not require $\tilde{\mathbf{u}}_1$ to be any of the left singular vectors of A. It can be shown that (Theorem 4 in §1.3) the rank of the matrix

$$\widetilde{A}_1 := A - \tilde{\mathbf{u}}_1 \tilde{\mathbf{u}}_1^\top A$$

is one less than the rank of A. Similarly, a unit vector $\tilde{\mathbf{u}}_2$ in the approxi-range of $\widetilde{A}_1$ is also in the approxi-range of A, and the rank of the matrix

$$\widetilde{A}_2 \;:=\; A - \tilde{\mathbf{u}}_1 \tilde{\mathbf{u}}_1^\top A - \tilde{\mathbf{u}}_2 \tilde{\mathbf{u}}_2^\top A$$

is reduced by another one, making it two below the rank of A. Moreover, $\tilde{\mathbf{u}}_1$ and $\tilde{\mathbf{u}}_2$ are orthogonal since $\tilde{\mathbf{u}}_1$ is in the left kernel of $\widetilde{A}_2$. This process continues recursively and terminates with the approximate rank identified as well as an orthonormal basis obtained for the approxi-range.

Our method has been implemented as a Matlab package LowRank. Comprehensive numerical results of our code comparing with UTV Tools [16] and Matlab SVD function are exhibited in §1.5. For low rank matrices, our code is consistently faster than UTV Tools and the full SVD by a large margin.

Moreover, row/column updating and downdating in our method are quite simple and straightforward. In §2.3, numerical results on both cases are presented to compare our method with UTV Tools in this respect. While UTV Tools may sometimes return incorrect ranks or inaccurate ranges, our method reliably yields accurate results on all the matrices we tested.

Practical applications of our algorithms on information retrieval and image processing are presented in §3.

CHAPTER 1

A Rank-Revealing Method

1.1 Approximate ranks

The terms rank, nullity, and kernel are used in the *exact* sense as in common linear algebra textbooks. The approximate rank, also known as the numerical rank, has a specific meaning as in Definition 1 below. We use specific terms *approxi-rank*, *approxi-range* and *approxi-rowspace* for them respectively, but $\text{rank}(A)$ is still the exact rank of matrix A.

We shall denote matrices by upper case letters such as A , B, R, and column vectors by lower case boldface letters like $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{y}$. Notation $(\cdot)^{\top}$ stands for the transpose of a matrix or a vector, $(\cdot)^{\perp}$ represents the orthogonal complement of a subspace, and $\|\cdot\|$ denotes the 2-norm of a matrix or a vector. The symbol $\sigma_i(M)$ will denote the i-th largest singular value of matrix M.

Definition 1. [18] *For a given threshold $\theta > 0$, a matrix $A \in \mathbb{R}^{m\times n}$ is of* ***approxi-rank*** *k within θ, denoted by $\text{rank}_\theta(A) = k$, if k is the smallest rank of all matrices within a 2-norm distance θ of A. Namely,*

$$\text{rank}_\theta(A) \;=\; \min_{\|A-B\|\,\le\,\theta}\left\{\text{rank}(B)\right\} \;=\; k. \tag{1.1.1}$$

*Hereby, we also say the **approxi-nullity** of A within θ is $n-k$.*

The exact rank may be regarded as a special case of the approxi-rank since $\text{rank}(A) = \text{rank}_\theta(A)$ for any matrix A within sufficiently small θ.

The minimum in (1.1.1) is attainable [18, 27]: Let the singular value decomposition of A be

$$A \;=\; U\Sigma V^\top \;=\; \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^\top + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^\top \tag{1.1.2}$$

where $U = [\mathbf{u}_1, \cdots, \mathbf{u}_m]$ and $V = [\mathbf{v}_1, \cdots, \mathbf{v}_n]$ are orthogonal matrices along with diagonal matrix $\Sigma = diag\{\sigma_1, \cdots, \sigma_n\}$ formed by singular values $\sigma_1, \cdots, \sigma_n$ satisfying

$$\sigma_1 \;\geq\; \cdots \;\geq\; \sigma_k \;>\; \theta \;\geq\; \sigma_{k+1} \;\geq\; \cdots \;\geq\; \sigma_n \geq 0. \tag{1.1.3}$$

Then $\left\|A - A_k\right\| = \sigma_{k+1}$ is the minimum 2-norm distance from A to any rank k matrix for $A_k = U\Sigma_k V^\top$ with diagonal matrix $\Sigma_k = diag\left\{\sigma_1, \cdots, \sigma_k, 0, \cdots, 0\right\}$. Moreover, $\text{rank}(A_k) = \text{rank}_\theta(A) = k$. In other words, for

$$\breve{\sigma} = \inf\left\{\mu \mid \text{rank}_\mu(A) = k\right\} \text{ and } \widehat{\sigma} = \sup\left\{\eta \mid \text{rank}_\eta(A) = k\right\},$$

we have $\breve{\sigma} = \sigma_{k+1}$ and $\widehat{\sigma} = \sigma_k$. We call the ratio $\gamma = \widehat{\sigma} \,/\, \breve{\sigma}$ the *approxi-rank gap*.

The fundamental subspaces *range* $\mathcal{R}(A)$, *kernel* $\mathcal{K}(A)$, *left kernel* $\mathcal{K}(A^\top)$ and *row space* $\mathcal{R}(A^\top)$ associated with matrix A can be naturally generalized in the approximate sense. In terms of the SVD of A in (1.1.2) with singular values satisfying (1.1.3), the approximate subspaces of A along with their notations are listed as follows.

- $\mathcal{R}_\theta(A) = \text{span}\{\mathbf{u}_1, \cdots, \mathbf{u}_k\}$: The *approxi-range* of A within θ.
- $\mathcal{K}_\theta(A) = \text{span}\{\mathbf{v}_{k+1}, \cdots, \mathbf{v}_n\}$: The *approxi-kernel* of A within θ.
- $\mathcal{R}_\theta(A^\top) = \text{span}\{\mathbf{v}_1, \cdots, \mathbf{v}_k\}$: The *approxi-rowspace* of A within θ.
- $\mathcal{K}_\theta(A^\top) = \text{span}\{\mathbf{u}_{k+1}, \cdots, \mathbf{u}_m\}$: The *approxi-leftkernel* of A within θ.

1.2 The convergence theory

For a given $m \times n$ matrix A and a rank threshold $\theta > 0$, we can assume $m \geq n$, $\operatorname{rank}_\theta(A) = k$ and the SVD of A is given in (1.1.2) with

$$\sigma_1 \geq \cdots \geq \sigma_k = \widehat{\sigma} > \theta \geq \breve{\sigma} = \sigma_{k+1} \geq \cdots \geq \sigma_n. \tag{1.2.1}$$

For a vector $\mathbf{z} \neq \mathbf{0}$ and a subspace $\mathcal{W}$ in $\mathbb{R}^l$, the distance between $\mathbf{z}$ and $\mathcal{W}$, denoted by $\operatorname{dist}(\mathbf{z}, \mathcal{W})$, is defined as the distance between supspaces $\operatorname{span}\{\mathbf{z}\}$ and $\mathcal{W}$. Namely,

$$\operatorname{dist}(\mathbf{z}, \mathcal{W}) = \frac{\|\mathbf{z} - WW^\top \mathbf{z}\|}{\|\mathbf{z}\|}$$

if columns of matrix W form an orthonormal basis for subspace $\mathcal{W}$. We say a sequence $\{\mathbf{z}_j\}_{j=1}^{\infty}$ of nonzero vectors converges into subspace $\mathcal{W}$ if $\lim_{j \to \infty} \operatorname{dist}(\mathbf{z}_j, \mathcal{W}) = 0$.

Our strategy of revealing the approxi-rank of A is to construct an orthonormal basis for the approxi-range $\mathcal{R}_\theta(A)$. For this purpose, we use the power iteration on $AA^\top$ as follows: For a randomly generated unit vector $\mathbf{y}_0 \in \mathbb{R}^m$, define sequences $\{\mathbf{x}_j\}$ and $\{\mathbf{y}_j\}$ as

$$\mathbf{x}_j = A^\top \mathbf{y}_{j-1} \big/ \|A^\top \mathbf{y}_{j-1}\|, \ \mathbf{y}_j = A\mathbf{x}_j \big/ \|A\mathbf{x}_j\| \quad \text{for } j = 1, 2, \cdots \tag{1.2.2}$$

that converge into the approxi-rowspace $\mathcal{R}_\theta(A^\top)$ and the approxi-range $\mathcal{R}_\theta(A)$, respectively, at convergence rates given in the following proposition.

Proposition 2. *For $\theta > 0$ and $A \in \mathbb{R}^{m \times n}$ with SVD in* (1.1.2) *and singular values satisfying* (1.2.1), *let $\mathbf{y}_0 \in \mathbb{R}^m$ such that $\mathbf{y}_0 \notin \mathcal{R}_\theta(A)^\perp$, then the sequences $\{\mathbf{x}_j\}$ and $\{\mathbf{y}_j\}$ generated by iteration* (1.2.2) *converge* *into $\mathcal{R}_\theta(A^\top)$ and $\mathcal{R}_\theta(A)$ respectively at linear rate*

$$\operatorname{dist}\big(\mathbf{x}_j, \mathcal{R}_\theta(A^\top)\big) \le \phi_{2j-1} \quad \textit{and} \quad \operatorname{dist}\big(\mathbf{y}_j, \mathcal{R}_\theta(A)\big) \le \phi_{2j}, \; j = 1, 2, \cdots \tag{1.2.3}$$

with

$$\phi_l \equiv \left(\frac{\breve{\sigma}}{\bar{\hat{\sigma}}}\right)^l \frac{\operatorname{dist}\big(\mathbf{y}_0, \mathcal{R}_\theta(A)\big)}{\sqrt{1 - \operatorname{dist}\big(\mathbf{y}_0, \mathcal{R}_\theta(A)\big)^2}} \quad \textit{for } l \in \{1, 2, \cdots\}.$$

Proof. Write $\mathbf{y}_0 = c_1\mathbf{u}_1 + \cdots + c_m\mathbf{u}_m$. Without loss of generality, we assume $m \ge n$. Let matrix $\widehat{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_k]$. From $AA^\top = U\Sigma\Sigma^\top U^\top$ and for some $\eta \in \mathbb{R}$,

$$\begin{aligned} \mathbf{y}_1 &= \eta\left(c_1\sigma_1^2\mathbf{u}_1 + \cdots + c_n\sigma_n^2\mathbf{u}_n\right) \qquad (1.2.4) \\ &= \alpha\left(c_1\frac{\sigma_1^2}{\widehat{\sigma}^2}\mathbf{u}_1 + \cdots + c_k\frac{\sigma_k^2}{\widehat{\sigma}^2}\mathbf{u}_k + c_{k+1}\frac{\sigma_{k+1}^2}{\widehat{\sigma}^2}\mathbf{u}_{k+1} + \cdots + c_n\frac{\sigma_n^2}{\widehat{\sigma}^2}\mathbf{u}_n\right) \end{aligned}$$

for $\alpha = \eta\,\widehat{\sigma}^2$. Then

$$\left\|\widehat{U}\widehat{U}^\top \mathbf{y}_1\right\| = \left\|\alpha\left(c_1\frac{\sigma_1^2}{\widehat{\sigma}^2}\mathbf{u}_1 + \cdots + c_k\frac{\sigma_k^2}{\widehat{\sigma}^2}\mathbf{u}_k\right)\right\| \ge |\alpha|\left\|\widehat{U}\widehat{U}^\top \mathbf{y}_0\right\|$$

and

$$\begin{aligned} \left\|\mathbf{y}_1 - \widehat{U}\widehat{U}^\top \mathbf{y}_1\right\| &= \left\|\alpha\left(c_{k+1}\frac{\sigma_{k+1}^2}{\widehat{\sigma}^2}\mathbf{u}_{k+1} + \cdots + c_n\frac{\sigma_n^2}{\widehat{\sigma}^2}\mathbf{u}_n\right)\right\| \\ &\le |\alpha|\left(\frac{\breve{\sigma}}{\bar{\hat{\sigma}}}\right)^2\left\|\mathbf{y}_0 - \widehat{U}\widehat{U}^\top \mathbf{y}_0\right\|. \end{aligned}$$

Since $\mathbf{y}_0 \notin \mathcal{R}_\theta(A)^\perp$, we have $\widehat{U}\widehat{U}^\top \mathbf{y}_0 \ne \mathbf{0}$ and

$$\operatorname{dist}(\mathbf{y}_1, \mathcal{R}_\theta(A)) \le \frac{\|\mathbf{y}_1 - \widehat{U}\widehat{U}^\top \mathbf{y}_1\|}{\|\widehat{U}\widehat{U}^\top \mathbf{y}_1\|} \le \left(\frac{\breve{\sigma}}{\bar{\hat{\sigma}}}\right)^2\frac{\|\mathbf{y}_0 - \widehat{U}\widehat{U}^\top \mathbf{y}_0\|}{\|\widehat{U}\widehat{U}^\top \mathbf{y}_0\|} = \phi_2$$

and inequality $\operatorname{dist}\big(\mathbf{y}_j, \mathcal{R}_\theta(A)\big) \le \phi_{2j}$ in (1.2.3) follows a straightforward induction. Inequality $\operatorname{dist}\big(\mathbf{x}_j, \mathcal{R}_\theta(A^\top)\big) \le \phi_{2j-1}$ can be proved similarly.

□

Most existing rank-revealing methods for a low rank matrix A begin with calculating the singular vector corresponding to $\sigma_1(A)$. For those methods, the accuracy of computed subspaces and approxi-ranks relies heavily on the quality of singular vector estimation [14]. A distinctive feature of our method is that it only computes vectors in the approxi-range and the approxi-rowspace, and none of those computed vectors needs to be singular vectors.

For finding a unit vector in the approxi-range, we use iteration (1.2.2) with stopping criterion

$$\left(\frac{\theta}{\|A^\top \mathbf{y}_j\|}\right)^{2j} < \epsilon_m \tag{1.2.5}$$

where ϵ_m is chosen on the order of the machine precision. This stopping criterion is established for the following considerations: From equation (1.2.4), and by a simple induction, we have

$$\begin{aligned} \mathbf{y}_j \;=\; \alpha_j \Bigg[& \mathbf{u}_1 + \frac{c_2}{c_1}\left(\frac{\sigma_2}{\sigma_1}\right)^{2j} \mathbf{u}_2 + \cdots + \frac{c_k}{c_1}\left(\frac{\sigma_k}{\sigma_1}\right)^{2j} \mathbf{u}_k \\ & + \frac{c_{k+1}}{c_1}\left(\frac{\sigma_{k+1}}{\sigma_1}\right)^{2j} \mathbf{u}_{k+1} + \cdots + \frac{c_n}{c_1}\left(\frac{\sigma_n}{\sigma_1}\right)^{2j} \mathbf{u}_n \Bigg] \end{aligned} \tag{1.2.6}$$

where α_j is the scalar that normalizes $\mathbf{y}_j$. Obviously, sequence $\{\mathbf{y}_j\}$ approaches $\mathcal{R}_\theta(A)$ first, and will ultimately converge to $\mathbf{u}_1$ if the largest singular value σ_1 is strictly larger than the others. Since $\operatorname{rank}_\theta(A) = k$, $\sigma_{k+1} \le \theta$ and $\eta_j \equiv \|A^\top \mathbf{y}_j\| \le \|A^\top\| = \sigma_1$, hence $(\sigma_{k+1}/\sigma_1)^{2j} \le (\theta/\eta_j)^{2j}$ for $j = 1, 2, \cdots$. Assume $(\theta/\eta_h)^{2h} < \epsilon_m$ after h steps of iteration (1.2.2). Then $(\sigma_i/\sigma_1)^{2h} < \epsilon_m$ for $i = k+1, \cdots, n$. Set $\rho \equiv \max_{k+1 \le i \le n} |c_i/c_1|$. The distance from $\mathbf{y}_h$ to the approxi-range $\mathcal{R}_\theta(A)$ is

$$\begin{aligned}
\operatorname{dist}\big(\mathbf{y}_h, \mathcal{R}_\theta(A)\big) &= \alpha_h \left\| \frac{c_{k+1}}{c_1}\left(\frac{\sigma_{k+1}}{\sigma_1}\right)^{2h} \mathbf{u}_{k+1} + \cdots + \frac{c_n}{c_1}\left(\frac{\sigma_n}{\sigma_1}\right)^{2h} \mathbf{u}_n \right\| \\
&< \left\| \rho\epsilon_m \mathbf{u}_{k+1} + \rho\epsilon_m \mathbf{u}_{k+2} + \cdots + \rho\epsilon_m \mathbf{u}_n \right\| = \sqrt{n-k}\,\rho\epsilon_m.
\end{aligned}$$

Since $\mathbf{y}_0$ is randomly generated, ρ is of order 1. Thus, $\mathbf{y}_h$ can be taken as a unit vector in the approxi-range $\mathcal{R}_\theta(A)$.

Iteration (1.2.2) can also be viewed as a power iteration on $A^\top A$. Similar to (1.2.6), we may obtain

$$\begin{aligned}
\mathbf{x}_j &= \beta_j \left[\mathbf{v}_1 + \frac{c_2}{c_1}\left(\frac{\sigma_2}{\sigma_1}\right)^{2j-1} \mathbf{v}_2 + \cdots + \frac{c_k}{c_1}\left(\frac{\sigma_k}{\sigma_1}\right)^{2j-1} \mathbf{v}_k \right. \\
&\quad \left. + \frac{c_{k+1}}{c_1}\left(\frac{\sigma_{k+1}}{\sigma_1}\right)^{2j-1} \mathbf{v}_{k+1} + \cdots + \frac{c_n}{c_1}\left(\frac{\sigma_n}{\sigma_1}\right)^{2j-1} \mathbf{v}_n \right],
\end{aligned}$$

where $\mathbf{v}_i$ is the (right) singular vector of A associated with σ_i and β_j is the scalar that normalizes the vector $\mathbf{x}_j$. Similarly, $\{\mathbf{x}_j\}$ converges into the approxi-rowspace $\mathcal{R}_\theta(A^\top)$, and as before, condition $\big(\theta/\|A\mathbf{x}_j\|\big)^{2j-1} < \epsilon_m$ can be used as an stopping criterion.

1.3 Computing the approxi-range and the approxi-rowspace

Iteration (1.2.2) produces a vector $\mathbf{z}_1$ in the approxi-range $\mathcal{R}_\theta(A)$. We shall show in this section that the approxi-rank $\text{rank}_\theta(A - \mathbf{z}_1\mathbf{z}_1^\top A) = \text{rank}_\theta(A) - 1$. Moreover, when the approxi-rank $\text{rank}_\theta(A)$ is higher than one, applying iteration (1.2.2) to $A - \mathbf{z}_1\mathbf{z}_1^\top A$ yields another vector $\mathbf{z}_2 \in \mathcal{R}_\theta(A)$ that is orthogonal to $\mathbf{z}_1$. This deflation process can be continued recursively to produce an orthonormal basis for the approxi-range $\mathcal{R}_\theta(A)$.

Lemma 3. *For matrix $M \in \mathbb{R}^{m \times n}$ with $m \geq n$ and vector $\mathbf{h} \in \mathbb{R}^n$, let $\widehat{M} = \begin{bmatrix} \mathbf{h}^\top \\ M \end{bmatrix}$. If $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ are singular values of M and $\sigma'_1 \geq \sigma'_2 \geq \cdots \geq \sigma'_n$ are singular values of $\widehat{M}$, then $\sigma'_1 \geq \sigma_1 \geq \sigma'_2 \geq \sigma_2 \geq \cdots \geq \sigma'_n \geq \sigma_n$.*

Proof. This interlacing property is an analogical consequence of Theorem 7.3.9 in [20]. □

With the same notations as in the last section, we have the following theorem.

Theorem 4. *For integer $j \in \{1, 2, \cdots, k\}$ and matrix $W = \left[\mathbf{z}_1, \ \mathbf{z}_2, \ \cdots, \ \mathbf{z}_j\right]$ whose columns form an orthonormal basis for a j-dimensional subspace $\mathcal{W}$ of $\mathcal{R}_\theta(A)$, matrix $A - WW^\top A$ has singular values $\{\sigma'_1, \sigma'_2, \cdots, \sigma'_n\}$ satisfying*

$$\begin{aligned} \sigma_1 &\geq \sigma'_1 \geq \sigma'_2 \geq \cdots \geq \sigma'_{k-j} \geq \sigma_k \,, \\ \sigma'_{k-j+1} &= \sigma'_{k-j+2} = \cdots = \sigma'_k = 0, \end{aligned}$$

and $\sigma'_i = \sigma_i$ *for* $i = k+1, \cdots, n$.

Moreover, let $\mathcal{W}' = \text{span}\{\mathbf{u}'_1, \cdots, \mathbf{u}'_{k-j}\}$ where $\mathbf{u}'_i$ is the left singular vector of matrix $A - WW^\top A$ associated with σ'_i for $1 \leq i \leq k - j$. Then $\mathcal{W}' \subset \mathcal{R}_\theta(A) \cap \mathcal{W}^\perp$.

Proof. For $j = 1$ and $B = A - \mathbf{z}_1\mathbf{z}_1^\top A$, let $\{\mathbf{z}_1, \hat{\mathbf{u}}_2, \cdots, \hat{\mathbf{u}}_k\}$ be an orthonormal basis for $\mathcal{R}_\theta(A)$. Also let $\widehat{U} = [\mathbf{z}_1, \hat{\mathbf{u}}_2, \cdots, \hat{\mathbf{u}}_k, U_{k+1}]$, where $U_{k+1} = [\mathbf{u}_{k+1}, \cdots, \mathbf{u}_m]$. Then

$$
\begin{aligned}
\widehat{U}^\top AV &= [\mathbf{z}_1, \hat{\mathbf{u}}_2, \cdots, \hat{\mathbf{u}}_k, U_{k+1}]^\top \begin{bmatrix} A\mathbf{v}_1 & \cdots & A\mathbf{v}_k & A\mathbf{v}_{k+1} & \cdots & A\mathbf{v}_n \end{bmatrix} \\
&= \begin{bmatrix}
\mathbf{z}_1^\top A\mathbf{v}_1 & \cdots & \mathbf{z}_1^\top A\mathbf{v}_k & & & \\
\hat{\mathbf{u}}_2^\top A\mathbf{v}_1 & \cdots & \hat{\mathbf{u}}_2^\top A\mathbf{v}_k & & & \\
\vdots & & \vdots & & & \\
\hat{\mathbf{u}}_k^\top A\mathbf{v}_1 & \cdots & \hat{\mathbf{u}}_k^\top A\mathbf{v}_k & & & \\
 & & & \sigma_{k+1} & & \\
 & & & & \ddots & \\
 & & & & & \sigma_n \\
 & & & & & \\
 & & & & &
\end{bmatrix} \\
&= \begin{bmatrix}
\widehat{M} & & & \\
 & \sigma_{k+1} & & \\
 & & \ddots & \\
 & & & \sigma_n \\
 & & &
\end{bmatrix},
\end{aligned}
$$

where $\widehat{M} = \begin{bmatrix} \mathbf{h}^\top \\ M \end{bmatrix}$ with $\mathbf{h}^\top = [\mathbf{z}_1^\top A\mathbf{v}_1, \cdots, \mathbf{z}_1^\top A\mathbf{v}_k]$

$$
\text{and } M = \begin{bmatrix}
\hat{\mathbf{u}}_2^\top A\mathbf{v}_1 & \cdots & \hat{\mathbf{u}}_2^\top A\mathbf{v}_k \\
\vdots & \ddots & \vdots \\
\hat{\mathbf{u}}_k^\top A\mathbf{v}_1 & \cdots & \hat{\mathbf{u}}_k^\top A\mathbf{v}_k
\end{bmatrix}.
$$

Since $\widehat{U}$ and V are orthogonal matrices, the singular values of $\widehat{M}$ are $\{\sigma_1, \sigma_2, \cdots, \sigma_k\}$.

On the other hand, $\widehat{U}^\top BV = \widehat{U}^\top \left(I - \mathbf{z}_1\mathbf{z}_1^\top\right) AV$

$$
\begin{aligned}
&= \left[\mathbf{z}_1, \hat{\mathbf{u}}_2, \cdots, \hat{\mathbf{u}}_k, U_{k+1}\right]^\top \left(I - \mathbf{z}_1\mathbf{z}_1^\top\right) \\
&\quad \begin{bmatrix} A\mathbf{v}_1 & \cdots & A\mathbf{v}_k & A\mathbf{v}_{k+1} & \cdots & A\mathbf{v}_n \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{0} \\ \hat{\mathbf{u}}_2^\top \\ \vdots \\ \hat{\mathbf{u}}_k^\top \\ U_{k+1}^\top \end{bmatrix} \begin{bmatrix} A\mathbf{v}_1 & \cdots & A\mathbf{v}_k & A\mathbf{v}_{k+1} & \cdots & A\mathbf{v}_n \end{bmatrix} \\
&= \begin{bmatrix} M' & & & \\ & \sigma_{k+1} & & \\ & & \ddots & \\ & & & \sigma_n \\ & & & \\ & & & \end{bmatrix}
\end{aligned}
$$

with $M' = \begin{bmatrix} \mathbf{0} \\ M \end{bmatrix}$. By Lemma 3, singular values $\sigma'_1 \geq \sigma'_2 \geq \cdots \geq \sigma'_k$ of M' satisfy

$$\sigma_1 \geq \sigma'_1 \geq \sigma_2 \geq \sigma'_2 \geq \cdots \geq \sigma'_{k-1} \geq \sigma_k \geq \sigma'_k.$$

Since $\operatorname{rank}(M') = k - 1$, $\sigma'_k = 0$, hence only $k - 1$ singular values of B are larger than θ, and the rest of them satisfy $\sigma'_{k+1} = \sigma_{k+1}, \sigma'_{k+2} = \sigma_{k+2}, \cdots, \sigma'_n = \sigma_n$.

Now, left singular vectors $\mathbf{u}'_1, \mathbf{u}'_2, \cdots, \mathbf{u}'_{k-1}$ of matrix B corresponding to singular values $\sigma'_1 \geq \sigma'_2 \geq \cdots \geq \sigma'_{k-1}$ form a basis for the approxi-range of B within θ and $\mathbf{z}_1$ is in the approxi-leftkernel of $B = \left(I - \mathbf{z}_1\mathbf{z}_1^\top\right) A$, therefore, $\mathbf{z}_1 \in \mathcal{W}'^\perp$ with $\mathcal{W}' = \operatorname{span}\left\{\mathbf{u}'_1, \cdots, \mathbf{u}'_{k-1}\right\}$.

The assertion for general $1 < j \leq k$ follows from a straightforward induction. $\square$

Applying iteration (1.2.2) on matrix $A - \mathbf{z}_1\mathbf{z}_1^\top A$ yields a unit vector $\mathbf{z}_2 \in \mathcal{R}_\theta(A)$ that is orthogonal to $\mathbf{z}_1$. Continuing this process recursively, an orthonormal basis for $R_\theta(A)$ will be obtained. Likewise, an orthonormal basis for the approxi-rowspace can be obtained recursively by finding a sequence of vectors in the approxi-rowspace.

Corollary 5. *For integer $j \in \{1, 2, \cdots, k\}$ and matrix $Y = \left[\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_j\right]$ whose columns form an orthonormal basis for a j-dimensional subspace $\mathcal{Y}$ of the approxi-rowspace $\mathcal{R}_\theta(A^\top)$, matrix $A - AYY^\top$ has singular values $\{\sigma_1', \sigma_2', \cdots, \sigma_n'\}$ satisfying*

$$
\begin{array}{rcl}
\sigma_1 & \geq & \sigma_1' \geq \sigma_2' \geq \cdots \geq \sigma_{k-}' \geq \sigma_k , \\
\sigma_{k-j+1}' & = & \sigma_{k-j+2}' = \cdots = \sigma_k' = 0,
\end{array}
$$

and $\sigma_i' = \sigma_i$ *for* $i = k+1, \cdots, n$.

Moreover, let $\mathcal{Y}' = span\left\{\mathbf{v}_1', \cdots, \mathbf{v}_{k-j}'\right\}$ where $\mathbf{v}_i'$ is the right singular vector of matrix $A - AYY^\top$ associated with σ_i' for $1 \leq i \leq k-j$. Then $\mathcal{Y}' \subset \mathcal{R}_\theta(A^\top) \cap \mathcal{Y}^\perp$.

From Theorem 4, one may deduce the following general rule: When a unit vector $\mathbf{z}$ in the approxi-range of A is obtained, let $B = A - \mathbf{z}\mathbf{z}^\top A$, then

- one of the singular values of A above the rank threshold becomes zero for matrix B;
- the remaining singular values of A above the threshold may shift but stay in the same interval; and
- singular values of A below the threshold stay the same as singular values of B.

Therefore, the approxi-rank gap of the new matrix after each deflation process will not become smaller, yielding an essential ingredient for achieving robustness in our algorithm.

The unit vector produced by iteration (1.2.2) is only *close* to, not exactly in, the approxi-range of A. The following proposition shows that deflation with such a vector, the approxi-rank of the resulting matrix within the same threshold remains the same as long as the approxi-rank gap of A is not too small.

Proposition 6. *Let* $A \in \mathbb{R}^{m \times n}$ *and* $\theta > 0$*. For unit vector* $\mathbf{z} \in \mathbb{R}^m$ *and* $\widehat{\mathbf{z}}$ *being its orthogonal projection on the approxi-leftkernel* $\mathcal{K}_\theta(A^\top)$*. Assume* $\left\| \widehat{\mathbf{z}} \right\| = \epsilon < 1$*. Then* $\mathrm{rank}_\theta(A - \mathbf{z}\mathbf{z}^\top A) = \mathrm{rank}_\theta(A) - 1$ *if* $\min\{\sigma_k - \theta,\ \theta - \sigma_{k+1}\} > \|A\|\,\epsilon$.

Proof. Let $\breve{\mathbf{z}}$ be the orthogonal projection of $\mathbf{z}$ on $\mathcal{R}_\theta(A)$. Set $\mathbf{d} = \breve{\mathbf{z}}/\|\breve{\mathbf{z}}\|$, $B = A - \mathbf{z}\mathbf{z}^\top A$, and $\tilde{B} = A - \mathbf{d}\mathbf{d}^\top A$. Write $\mathbf{z} = \widehat{\mathbf{z}} + \breve{\mathbf{z}} = \widehat{\mathbf{z}} + (\mathbf{z}^\top\mathbf{d})\mathbf{d} = \widehat{\mathbf{z}} + \sqrt{1-\epsilon^2}\,\mathbf{d}$. Let $\mathbf{h} = \widehat{\mathbf{z}}/\|\widehat{\mathbf{z}}\|$ and $U = [\mathbf{h},\ \mathbf{d}] \in \mathbb{R}^{m \times 2}$. Then

$$
\begin{aligned}
\mathbf{z}\mathbf{z}^\top - \mathbf{d}\mathbf{d}^\top &= \left(\widehat{\mathbf{z}} + \sqrt{1-\epsilon^2}\,\mathbf{d}\right)\left(\widehat{\mathbf{z}} + \sqrt{1-\epsilon^2}\,\mathbf{d}\right)^\top - \mathbf{d}\mathbf{d}^\top \\
&= \widehat{\mathbf{z}}\widehat{\mathbf{z}}^\top + \sqrt{1-\epsilon^2}\;\widehat{\mathbf{z}}\;\mathbf{d}^\top + \sqrt{1-\epsilon^2}\,\mathbf{d}\;\widehat{\mathbf{z}}^\top - \epsilon^2\,\mathbf{d}\mathbf{d}^\top \\
&= \epsilon^2\,\mathbf{h}\mathbf{h}^\top + \sqrt{1-\epsilon^2}\,\epsilon\,\mathbf{h}\mathbf{d}^\top + \sqrt{1-\epsilon^2}\,\epsilon\,\mathbf{d}\mathbf{h}^\top - \epsilon^2\,\mathbf{d}\mathbf{d}^\top \\
&= U \begin{bmatrix} \epsilon^2 & \epsilon\sqrt{1-\epsilon^2} \\ \epsilon\sqrt{1-\epsilon^2} & -\epsilon^2 \end{bmatrix} U^\top.
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\left\|\tilde{B} - B\right\| &= \left\|\left(\mathbf{z}\mathbf{z}^\top - \mathbf{d}\mathbf{d}^\top\right) A\right\| \\
&\le \left\|\mathbf{z}\mathbf{z}^\top - \mathbf{d}\mathbf{d}^\top\right\| \|A\| \le \left\| \begin{bmatrix} \epsilon^2 & \epsilon\sqrt{1-\epsilon^2} \\ \epsilon\sqrt{1-\epsilon^2} & -\epsilon^2 \end{bmatrix} \right\| \|A\| \\
&= \epsilon \left\| \begin{bmatrix} \epsilon & \sqrt{1-\epsilon^2} \\ \sqrt{1-\epsilon^2} & -\epsilon \end{bmatrix} \right\| \|A\| = \epsilon\,\|A\|
\end{aligned}
$$

since matrix $\begin{bmatrix} \epsilon & \sqrt{1-\epsilon^2} \\ \sqrt{1-\epsilon^2} & -\epsilon \end{bmatrix}$ is orthogonal.

Let $\{\tilde{\sigma}_1, \tilde{\sigma}_2, \cdots, \tilde{\sigma}_n\}$ be singular values of $\tilde{B}$. As shown in Theorem 4, those singular values satisfy $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_{k-1} \geq \sigma_k$, $\tilde{\sigma}_k = 0$, and $\tilde{\sigma}_{k+j} = \sigma_{k+j}$ for $j = 1, 2, \ldots, n-k$. By reindexing, we write $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n$ with $\tilde{\sigma}_n = 0$. Let $\sigma_1' \geq \sigma_2' \geq \cdots \geq \sigma_n'$ be the singular values of B. Then the perturbation theorem for singular values [19] yields $\left|\tilde{\sigma}_i - \sigma_i'\right| \leq \|A\|\,\epsilon$ for $i = 1, 2, \cdots, n$. Consequently,

$$\sigma_{k-1}' \geq \tilde{\sigma}_{k-1} - \|A\|\,\epsilon \geq \sigma_k - \|A\|\,\epsilon > \theta > \sigma_{k+1} + \|A\|\,\epsilon \geq \tilde{\sigma}_k + \|A\|\,\epsilon \geq \sigma_k'.$$

□

When the approxi-rank gap $\gamma = \sigma_k/\sigma_{k+1}$ is significantly larger than one, say $\gamma \approx 10^3$, and the threshold θ is not too close to the boundary of the interval (σ_{k+1}, σ_k), Proposition 6 ensures the deflation process to be safe in our rank-revealing algorithm.

1.4 The overall algorithm

Our algorithm has two main steps. We first find an approxi-range vector by the power iteration on $AA^\top$, followed by implicit deflation via subtracting an outer product of two vectors from matrix A. The power iteration on $AA^\top$ for approximating an approxi-range vector requires $4nm\mu$ flops, where μ is the average number of iterations per deflation step. We must emphasize here that our algorithm needs only a unit vector in the approxi-range instead of a singular vector. From equation (1.2.6), the average number μ of power iteration steps is small for our algorithm. This may help explain the high efficiency of our code.

Notice that if matrix $A \in \mathbb{R}^{m \times n}$ is too "tall" (i.e. $m \gg n$), we shall calculate the QR factorization of A $(= QR)$ first, and apply our algorithm on matrix R.

Our rank-revealing algorithm `larank` for low rank matrices can be outlined as follows. Let matrix $A \in \mathbb{R}^{m \times n}$ be given along with threshold $\theta > 0$.

Step 0. Initialize $A_1 = A$ and $i = 1$.

Step 1. Find unit vectors $\mathbf{x}_i \in \mathcal{R}_\theta(A_i^\top)$ and $\mathbf{y}_i \in \mathcal{R}_\theta(A_i)$ by iteration (1.2.2) on A_i. If $\mathcal{R}_\theta(A_i)$ is empty, that is, $\|A_i^\top \mathbf{y}_i\| \le \theta$, exit the algorithm.

Step 2. Set $A_{i+1} = A_i - \mathbf{y}_i\mathbf{y}_i^\top A$ implicitly.

Step 3. Increase i by one and go to Step 1.

On exit, this process returns the approxi-rank $k = i - 1$, bases $\{\mathbf{y}_1, \cdots, \mathbf{y}_k\}$ and $\{\mathbf{x}_1, \cdots, \mathbf{x}_k\}$ for $\mathcal{R}_\theta(A)$ and $\mathcal{R}_\theta(A^\top)$ respectively.

At Step 2, matrix A_{i+1} is implicitly obtained. It does not need to be constructed or stored. Matrix $A_{i+1} = A - \mathbf{y}_1\mathbf{y}_1^\top A - \cdots - \mathbf{y}_i\mathbf{y}_i^\top A$ is stored as matrix $[A, Y_i]$ where $Y_i = [\mathbf{y}_1, \cdots, \mathbf{y}_i]$. When applying iteration (1.2.2) on A_{i+1}, we use the identities

$$
\begin{aligned}
A_{i+1}^{\mathsf{T}}\mathbf{y} &= A^{\mathsf{T}}(\mathbf{y} - \mathbf{y}_1\mathbf{y}_1^{\mathsf{T}}\mathbf{y} - \cdots - \mathbf{y}_i\mathbf{y}_i^{\mathsf{T}}\mathbf{y}) = A^{\mathsf{T}}\left[\mathbf{y} - Y_i(Y_i^{\mathsf{T}}\mathbf{y})\right], \\
A_{i+1}\mathbf{x} &= A\mathbf{x} - \mathbf{y}_1\mathbf{y}_1^{\mathsf{T}}(A\mathbf{x}) - \cdots - \mathbf{y}_i\mathbf{y}_i^{\mathsf{T}}(A\mathbf{x}) = (A\mathbf{x}) - Y_i\left[Y_i^{\mathsf{T}}(A\mathbf{x})\right]
\end{aligned}
$$

without forming A_{i+1} explicitly. The detailed pseudo-code is given in Figure 1.1.

Vector sets $\{\mathbf{y}_1, \cdots, \mathbf{y}_k\}$ and $\{\mathbf{x}_1, \cdots, \mathbf{x}_k\}$ produced by Algorithm **larank** are almost orthonormal. The modified Gram-Schmidt method safely applies for their re-orthogonalization, yielding orthonormal bases for approxi-range $\mathcal{R}_\theta(A)$ and approxi-rowspace $\mathcal{R}_\theta(A^{\mathsf{T}})$ respectively. The flop counts for the pseudo-code is $(4nm - n - m)\mu(k+1) + (4m-1)k(k+1)\mu$ assuming $\text{rank}_\theta(A) = k$.

1.5 Numerical experiments

Our rank-revealing algorithm for low rank matrices is implemented as a Matlab package LowRank. We shall compare the efficiency, robustness and accuracy of our code `larank` in LowRank with Matlab built-in `svd` function as well as `lurv` and `lulv` in the UTV Tools implemented by Fierro, Hansen and Hansen [16]. All tests are carried out in Matlab 7.0 on a Dell PC with a Pentium D CPU of 3.2 GHz, 1GB of memory, and machine precision $\epsilon_{machine} \approx 2.2 \times 10^{-16}$. The main objective of our code `larank` is to calculate the approxi-rank, the approxi-range, and the approxi-rowspace of a matrix that has a low approxi-rank within a user-specified threshold.

If $A \in \mathbb{R}^{m \times n}$ is of approxi-rank k with threshold $\theta > 0$, then there is a "noise" matrix E with $\|E\| \leq \theta$ where $A - E$ has exact rank k. Relative perturbation $\|E\| \,/\, \|A\|$ is often referred to as noise level. Usually, the magnitude of relative perturbation near machine precision, say 10^{-12}, is taken as a low noise level, relative perturbation near 1, say 10^{-3}, a high noise level, and the median noise level is around 10^{-8}. In general, the threshold $\theta > 0$ one chooses reflects the noise level the matrices may have encountered.

1.5.1 Type 1: Low approxi-rank with median noise level

Matrices for this test are of size $2n \times n$ with approxi-rank fixed at 10 within threshold $\theta = 10^{-8}$. The singular values range from $\epsilon_{machine}$ to $\|A\| = 20$ with approxi-rank gap $\sigma_{10}/\sigma_{11} = 10^3$. Each matrix A is constructed by using those specified singular values to form a diagonal matrix Σ and setting $A = U\Sigma V^\top$ with randomly generated orthogonal matrices U and V with proper sizes. We use this type of matrices to test the efficiency and accuracy of our `larank` compared with `svd`, `lurv`, and `lulv` for increasing n. For approxi-ranks, the outputs of all four algorithms are accurate.

Table 1.1. Results for Type 1 matrices.

	Matrix sizes							
	400×200		800×400		1600×800		3200×1600	
	time	error	time	error	time	error	time	error
SVD	0.31	3e-09	2.19	4e-09	16.6	3e-09	144.	7e-09
lurv	0.66	3e-09	1.52	4e-09	5.97	3e-09	32.5	7e-09
lulv	0.56	4e-09	1.52	6e-09	6.03	5e-09	31.9	5e-09
larank	0.05	3e-09	0.11	4e-09	0.39	3e-09	1.81	4e-09

Table 1.1 only lists the time and subspace errors in executing `svd`, `lurv`, `lulv` and `larank`. The time measures in seconds and the error measures the distance of the computed approxi-range to the exact approxi-range. The results show that our `larank` is more than ten times faster than `lurv` and `lulv` with the same accuracy.

1.5.2 Type 2: Increasing approxi-rank, fixed size and median noise level

Matrices for this test are of size 1000×500. The singular values range from $\epsilon_{machine}$ to $\|A\| = 20$ with approxi-rank gap 10^3, and the approxi-ranks are set to be $10 + 20j$, for $j = 0, 1, \cdots, 5$, within a threshold 10^{-8}. We use this type of matrices to test the efficiency of `larank` compared with `lurv` and `lulv`.

Table 1.2. Results for Type 2 matrices.

	Average	Approximate rank					
Code	subspace error	10	30	50	70	90	110
lurv	5e-9	2.16	5.11	8.09	11.9	15.5	16.4
lulv	6e-9	2.17	5.72	8.22	11.8	15.4	17.5
larank	4e-9	0.14	0.34	0.53	0.75	1.02	1.22

Results in Table 1.2 shows our `larank` is over ten times faster than `lurv` and `lulv` on all cases with the same accuracy.

1.5.3 Type 3: Decreasing gap, fixed size and median noise level

Matrices for this test are of size 1000×500 with approxi-rank fixed at 10 within a threshold 10^{-8}. The singular values stretch from $\epsilon_{machine}$ to $\|A\| = 20$. However, the approxi-rank gaps are set at $12 - 2j$, for $j = 0, 1, \cdots, 5$ respectively. We use this type of matrices to test the accuracy of `larank` compared with `lurv` and `lulv` by comparing the approxi-range error which is the distance of the computed approxi-range to the corresponding approxi-range.

Table 1.3. Results for Type 3 matrices.

	Average	Approximate rank gaps					
Code	time	12.	10.	8.	6.	4.	2.
lurv	2.39	4e-8	3e-8	3e-8	4e-8	1e-7	7e-4
lulv	2.21	4e-8	4e-8	6e-8	6e-8	2e-6	1e-3
larank	0.17	3e-8	3e-8	3e-8	4e-8	5e-8	3e-5

The results show that even when the approxi-rank gaps are as small as 6.0, these three codes can still produce quite accurate approxi-ranges. When the gap is 4.0, all codes become worse while `lurv` and `lulv` have encountered tiny errors. When the gap is 2.0, our `larank` still enjoys a better accuracy.

1.5.4 Type 4: High noise level with increasing size

The series of matrices in this test are of $2n \times n$ with approxi-rank fixed at 10 within a threshold 10^{-2}. The singular values range from $\epsilon_{machine}$ to $\|A\| = 20$ with approxi-

Table 1.4. Results for Type 4 matrices.

	Matrix sizes							
	400 × 200		800 × 400		1600 × 800		3200 × 1600	
	time	error	time	error	time	error	time	error
lurv	0.50	5e-15	1.42	5e-15	8.27	5e-15	34.9	6e-15
lulv	0.49	5e-15	1.50	6e-15	7.80	7e-15	35.0	1e-14
larank	0.02	4e-15	0.13	4e-15	0.48	4e-15	2.08	5e-15

rank gap $\sigma_{10}/\sigma_{11} = 10^3$. The results show that all codes achieve accurate approxi-ranks and approxi-ranges, while our code has a substantial advantage in efficiency.

1.5.5 Type 5: Near zero noise level, low approxi-rank, large gap

This series of test matrices have singular values in the magnitude of machine precision except ten singular values are in the interval $[1, 2]$. We use threshold 10^{-12} to compute approxi-ranks and approxi-ranges.

Table 1.5. Results for Type 5 matrices.

	Matrix sizes							
	400 × 200		800 × 400		1600 × 800		3200 × 1600	
	time	error	time	error	time	error	time	error
lurv	0.48	2e-15	1.39	2e-15	8.33	4e-15	36.4	6e-15
lulv	0.50	2e-15	1.52	2e-15	8.38	5e-15	35.8	5e-15
larank	0.02	1e-15	0.06	2e-15	0.27	3e-15	1.41	3e-15

The results are similar to the results of Type 4. All codes obtain accurate approxi-ranks and approxi-ranges, while our code maintains the advantage in efficiency.

Algorithm `larank`

`Input`: Matrix $A \in \mathbb{R}^{m \times n}$, approxi-rank threshold $\theta > 0$

- `Initialize` $\epsilon_m = \|A\|_\infty \epsilon_{machine}$ along with empty matrices U and V
- `for` $k = 1, \cdots, n$ `do`
  - `generate` a random unit vector $\mathbf{y}_0$, `set` $\eta_0 = \zeta_0 = 0$
  - `for` $j = 1, 2, \cdots$ `do`
    - `set` $\mathbf{x} = A^\top[\mathbf{y}_{j-1} - U(U^\top \mathbf{y}_{j-1})]$, $\eta_j = \|\mathbf{x}\|$
    - `if` $\left(\frac{\theta}{\eta_j}\right)^{2j-1}$ `or` $\frac{\left|\eta_j - \eta_{j-1}\right|}{\left|\eta_j\right|} < \epsilon_m$ **then break** j-loop, **end** `if`
    - `set` $\mathbf{x}_j = \frac{1}{\eta_j}\mathbf{x}$, $\mathbf{p} = A\mathbf{x}_j$, $\mathbf{y} = \mathbf{p} - U(U^\top \mathbf{p})$, $\zeta_j = \|\mathbf{y}\|$
    - `if` $\left(\frac{\theta}{\zeta_j}\right)^{2j}$ `or` $\frac{\left|\zeta_j - \zeta_{j-1}\right|}{\left|\zeta_j\right|} < \epsilon_m$ **then break** j-loop, **end** `if`
    - $\mathbf{y}_j = \frac{1}{\zeta_j}\mathbf{y}$

    `end do`
  - `if` η_j `or` $\zeta_j \le \theta$ **then break** k-loop, **end** `if`
  - `update` $U = [U,\ \mathbf{y}_j]$ and $V = [V,\ \mathbf{x}_j]$
  - `orthogonalize` U and V by modified Gram-Schmidt method

  `end do`

`Output`: Approxi-rank k, orthonormal bases $\{\mathbf{y}_1, \cdots, \mathbf{y}_k\}$ and $\{\mathbf{x}_1, \cdots, \mathbf{x}_k\}$ for $\mathcal{R}_\theta(A)$ and $\mathcal{R}_\theta(A^\top)$ respectively.

(Operations applied on an empty vector or matrix is defined to be an empty operation.)

Figure 1.1. Algorithm `larank`

CHAPTER 2

Updating and Downdating Problems

2.1 The USV-plus decomposition

For a given threshold $\theta > 0$, we assume the singular values $\{\sigma_i\}$ of matrix $A \in \mathbb{R}^{m \times n}$ satisfy

$$\sigma_1 \geq \cdots \geq \sigma_k > \theta \geq \sigma_{k+1} \geq \cdots \geq \sigma_n.$$

Let $A = U\Sigma V^\top$ be the SVD of A. Write

$$A = A_k + E, \tag{2.1.1}$$

where $A_k = U\Sigma_k V^\top$ with $\Sigma_k = diag\{\sigma_1, \cdots, \sigma_k, 0, \cdots, 0\}$ and $E = U\Sigma_p V^\top$ with $\Sigma_p = diag\{0, \cdots, 0, \sigma_{k+1}, \cdots, \sigma_n\}$. Clearly, $\text{rank}_\theta(A) = \text{rank}(A_k) = k$ and $\|E\| = \sigma_{k+1} \leq \theta$. We shall call A_k the *dominant* part of A and call E the *noise* part of A.

Matrix A_k can be considered a by-product of Algorithm `larank` given in §1.4. While finding the approxi-rank of A, the algorithm produces a matrix $\widetilde{U} = [\tilde{\mathbf{u}}_1, \cdots, \tilde{\mathbf{u}}_k]$ whose columns form an orthonormal basis for the approxi-range $\mathcal{R}_\theta(A)$, and, by Theorem 4,

$$A - \tilde{\mathbf{u}}_1\tilde{\mathbf{u}}_1^\top A - \cdots - \tilde{\mathbf{u}}_k\tilde{\mathbf{u}}_k^\top A \; = \; A - \widetilde{U}\widetilde{U}^\top A \; = \; E.$$

Thus, $\widetilde{U}\widetilde{U}^\top A = A_k$ and $A = \widetilde{U}\widetilde{U}^\top A + E$. Similarly, if columns of matrix $\widetilde{V}$ form an orthonormal basis for the approxi-rowspace $\mathcal{R}_\theta(A^\top)$, then $A = A\widetilde{V}\widetilde{V}^\top + E$. Here, we shall consider several decompositions of A induced by (practical) factorizations of A_k.

Let $LQ^\top$ be the transpose of the "skinny" QR-factorization of $B = \widetilde{U}^\top A$, where $L \in \mathbb{R}^{k \times k}$ is lower triangular and $Q \in \mathbb{R}^{n \times k}$ has orthonormal columns. By a straightforward argument one can easily see that the row space $\mathcal{R}(B^\top)$, spanned by the orthonormal columns of Q, agrees with the approxi-rowspace $\mathcal{R}_\theta(A^\top)$. Now substituting $A_k = \widetilde{U}LQ^\top$ into (2.1.1) yields

$$A \; = \; \widetilde{U}LQ^\top + E, \tag{2.1.2}$$

which we call a "ULV-plus decomposition" of A within θ. If the SVD of L in (2.1.2) is $L = X\widehat{\Sigma}Y^\top$, then

$$A \; = \; \widehat{U}\widehat{\Sigma}\widehat{V}^\top + E,$$

where $\widehat{U} = \widetilde{U}X$ and $\widehat{V} = Y^\top Q^\top$. We call this an "SVD-plus decomposition" of A within θ. Let $L = \widetilde{Q}R$ be the QR-factorization of L where $R \in \mathbb{R}^{k \times k}$ is upper triangular, then (2.1.2) becomes

$$A \; = \; \widehat{U}RQ^\top + E$$

with $\widehat{U} = \widetilde{U}\widetilde{Q}$. We shall call this a "URV-plus decomposition" of A within θ.

Those ULV/URV/SVD-plus decompositions of A defined above are convenient tools when we deal with updating and downdating problems in the next section. The lower-triangular matrix L, the upper-triangular matrix R and diagonal matrix $\widehat{\Sigma}$ are small for low rank matrix A. We further assign a general name for these three types of decomposition as the "USV-plus decomposition" within θ where " S " suggests small size. Of particular importance is that when the approxi-rank k of A is small, the computation cost of those decompositions will be low.

2.2 Updating and downdating

Suppose the approxi-rank of A has been calculated along with orthonormal bases for the approxi-range and the approxi-rowspace. When a row/column is inserted in A, we wish to update all those results by taking all the available information into account. This process is called updating [32, 33]. It is called downdating [30] if a row/column is deleted from A. One of the main reasons for seeking alternatives to SVD is its difficulties in benefitting from the known information when updating and downdating are required. Like SVD, the USV-plus decomposition of A we introduced in the last section also contains, in addition to the approxi-rank of A, orthonormal bases for approxi-range and approxi-rowspace of A. Therefore, in updating/downdating we may update/downdate those results by updating/downdating the USV-plus decomposition of A. Both of these updating and downdating procedures turn out to be straightforward in our rank revealing method. Our extensive computing test shows they are accurate, stable and efficient. While the existing UTV decomposition processes robust updating capabilities, its downdating procedure seems somewhat complicated [16].

2.2.1 Updating

For $A \in \mathbb{R}^{m \times n}$ with $\text{rank}_\theta(A) = k > 0$, suppose one of its USV-plus decomposition is available, say, $A = ULV^\top + E$, where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ whose columns form an orthonormal basis for approxi-range $\mathcal{R}_\theta(A)$ and approxi-rowspace $\mathcal{R}_\theta(A^\top)$ respectively, $L \in \mathbb{R}^{k \times k}$ is lower triangular with $\sigma_k(L) > \theta$, and $E \in \mathbb{R}^{m \times n}$ with $\|E\| \leq \theta$. For $\mathbf{a} \in \mathbb{R}^n$, let $\widehat{A} = \begin{bmatrix} A \\ \mathbf{a}^\top \end{bmatrix}$ and $\alpha = \left\|\mathbf{a} - VV^\top \mathbf{a}\right\|$. If $\alpha \leq \theta$, then $\mathbf{a}$ may be taken as a vector in the approxi-rowspace, making the approxi-rowspaces $\mathcal{R}_\theta(\widehat{A}^\top) = \mathcal{R}_\theta(A^\top)$, so $\text{rank}_\theta(\widehat{A}) = k$. To update USV-plus decomposition of $\widehat{A}$, let $B = \widehat{A}V$. Then, for $\mathbf{b} = \mathbf{a} - VV^\top \mathbf{a}$,

$$BV^\top = \begin{bmatrix} A \\ \mathbf{a}^\top \end{bmatrix} VV^\top = \begin{bmatrix} AVV^\top \\ \mathbf{a}^\top VV^\top \end{bmatrix} = \begin{bmatrix} A - E \\ \mathbf{a}^\top - \mathbf{b}^\top \end{bmatrix} = \widehat{A} - \begin{bmatrix} E \\ \mathbf{b}^\top \end{bmatrix}.$$

Hence, $\widehat{A} = BV^\top + \begin{bmatrix} E \\ \mathbf{b}^\top \end{bmatrix}$, and the "skinny" QR-factorization $B = QR$ provides a URV-plus decomposition of $\widehat{A} = QRV^\top + \begin{bmatrix} E \\ \mathbf{b}^\top \end{bmatrix}$. If $\alpha > \theta$, then $\tilde{\mathbf{v}} = \frac{1}{\alpha}(\mathbf{a} - VV^\top \mathbf{a})$ is a unit vector of the projection of $\mathbf{a}$ on approxi-kernel $\mathcal{K}_\theta(A)$. Now, let $A_e = A - E$. Then

$$\widehat{A} = \begin{bmatrix} A \\ \mathbf{a}^\top \end{bmatrix} = \begin{bmatrix} A_e \\ \mathbf{a}^\top \end{bmatrix} + \begin{bmatrix} E \\ \mathbf{0}^\top \end{bmatrix} = \begin{bmatrix} U & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} L & \mathbf{0} \\ \mathbf{a}^\top V & \alpha \end{bmatrix} \begin{bmatrix} V^\top \\ \tilde{\mathbf{v}}^\top \end{bmatrix} + \begin{bmatrix} E \\ \mathbf{0}^\top \end{bmatrix}$$

is a ULV-plus decomposition of $\widehat{A}$.

When $\text{rank}_\theta(A) = k$ is small, finding SVD of $\begin{bmatrix} L & \mathbf{0} \\ \mathbf{a}^\top V & \alpha \end{bmatrix}$ is inexpensive, and importantly, it provides the singular value decomposition of the dominant part of $\widehat{A}$.

If $\alpha \approx \theta$, one of the singular values of $\widehat{A}$ may become close to the threshold θ which may result in a smaller approxi-rank gap. Consequently, we may lose the accuracy of approxi-range and approxi-rowspace as estimated before. In this case, we apply Algorithm `larank` in §1.4 with input matrix $\widehat{A}$ and use the vector $\tilde{\mathbf{v}}$ and the columns of V as the initial vectors individually for power iterations to obtain a new orthonormal base of approxi-rowspace of $\widehat{A}$. Certainly, this procedure may be used in general when more accurate approxi-rowspace or approxi-range are required.

When a new vector is inserted, we may always consider it is inserted in the last row by multiplying a permutation P first. On the other hand, the case of a URV-plus decomposition of $\widehat{A}$ as well as the column updating can be computed in a similar way.

We summarize the row updating algorithm `lrowup` in Figure 2.1 as a pseudo-code.

2.2.2 Downdating

To elaborate our downdating procedure, we need a singular value extracting strategy given below. Suppose $R \in \mathbb{R}^{k \times k}$ is upper triangular and $R\mathbf{v} = \sigma\mathbf{u}$, where σ is a singular value of R along with corresponding unit left/right singular vectors $\mathbf{u}$ and $\mathbf{v}$ respectively. We shall construct orthogonal matrices G and $\widetilde{G}$ as products of Givens rotations such that

$$\widetilde{G}RG = \begin{bmatrix} \widetilde{R} & 0 \\ 0 & \sigma \end{bmatrix}, \tag{2.2.1}$$

where $\widetilde{R} \in \mathbb{R}^{(k-1)\times(k-1)}$ remains upper triangular. Similarly, for a lower triangular matrix $L \in \mathbb{R}^{k \times k}$ with $L\mathbf{v} = \sigma\mathbf{u}$, orthogonal matrices G and $\widetilde{G}$ can be constructed such that $GL\widetilde{G}$ is in the form of $\begin{bmatrix} \widetilde{L} & 0 \\ 0 & \sigma \end{bmatrix}$, where $\widetilde{L} \in \mathbb{R}^{(k-1)\times(k-1)}$ is lower triangular.

The process for constructing G and $\widetilde{G}$ is recursive. Let G_1 be the Givens rotation which eliminates the first entry of $\mathbf{v}$. That is, if we write $G_1 = [\mathbf{g}_1, \cdots, \mathbf{g}_k]^\top$ then

$$G_1\mathbf{v} = \begin{bmatrix} \mathbf{g}_1^\top \\ \mathbf{g}_2^\top \\ \vdots \\ \mathbf{g}_k^\top \end{bmatrix} \mathbf{v} = \begin{bmatrix} 0 \\ \times \\ \vdots \\ \times \end{bmatrix}$$

Algorithm `lrowup`

`Input:` matrix A, approxi-rank k, rank threshold θ, new row $\mathbf{a}^\top$, row index p at which $\mathbf{a}^\top$ to be inserted in A, matrices U, S, V for the USV-plus decomposition

- `set` $\alpha = \left\| \mathbf{a} - VV^\top \mathbf{a} \right\|$, $\tilde{\mathbf{v}} = \frac{1}{\alpha}(\mathbf{a} - VV^\top \mathbf{a})$.
- `if` $\alpha \approx \theta$, `then` apply Algorithm `larank` on $\widehat{A}$ and use $\tilde{\mathbf{v}}$ and the columns of V individually as the initial vectors for the power iterations, `end if`
- `if` $\alpha > \theta$, `then`
  - `update` approxi-rank $k = k + 1$
  - `form` U_p by inserting $\mathbf{e}_{k+1}^\top$ above the p-th row of $\begin{bmatrix} U & \mathbf{0} \end{bmatrix}$.
  - `set` new $S = \begin{bmatrix} S & \mathbf{0} \\ \mathbf{a}^\top V & \alpha \end{bmatrix}$, $V = \begin{bmatrix} V & \tilde{\mathbf{v}} \end{bmatrix}$ and $U = U_p$.

  `else`
  - `form` the new matrix $\widehat{A}$ by inserting $\mathbf{a}^\top$ above the p-th row of A.
  - `set` $W = \widehat{A}V$, `find` the skinny QR factorization $W = QR$.
  - `set` $S = R$, $U = Q$.

  `end if`

`Output:` k , S, U, V

Figure 2.1. Algorithm `lrowup`

and $\mathbf{g}_1^\top \mathbf{v} = 0$. Because $\mathbf{v}$ and $\mathbf{u}$ are left and right singular vectors of $R^\top$ associated with singular value σ respectively, we have $R^\top \mathbf{u} = \sigma \mathbf{v}$ and therefore $(R\mathbf{g}_1)^\top \mathbf{u} = \mathbf{g}_1^\top R^\top \mathbf{u} = \sigma \mathbf{g}_1^\top \mathbf{v} = 0$. Only first two entries of $R\mathbf{g}_1$ can be nonzero since R is upper triangular and all entries of $\mathbf{g}_1$ are zeros except the first two. Let $R\mathbf{g}_1 =$

$[r_1, r_2, 0, \cdots, 0]^\top$ and $\mathbf{u} = [u_1, u_2, \cdots, u_k]^\top$. This yields $r_1 u_1 + r_2 u_2 = 0$ and

$$RG_1^\top = R\,[\mathbf{g}_1, \cdots, \mathbf{g}_k] = \begin{bmatrix} r_1 & \times & \times & \cdots & \times \\ r_2 & \times & \times & \cdots & \times \\ 0 & 0 & \times & \cdots & \times \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \times \end{bmatrix}.$$

Let $\widetilde{G}_1 = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & I_{k-2} \end{bmatrix}$ be the Givens rotation with $c = r_1/\sqrt{r_1^2 + r_2^2}$ and $s = r_2/\sqrt{r_1^2 + r_2^2}$. Clearly, $\widetilde{G}_1 R G_1^\top$ becomes upper triangular and the first entry of $\widetilde{G}_1 \mathbf{u}$ is zero. In summary, $R\mathbf{v} = \sigma \mathbf{u}$ implies $\widetilde{G}_1 R G_1^\top G_1 \mathbf{v} = \sigma \widetilde{G}_1 \mathbf{u}$ and with upper triangular matrix $R_1 = \widetilde{G}_1 R G_1^\top$, we have

$$R_1 \begin{bmatrix} 0 \\ \times \\ \vdots \\ \times \end{bmatrix} = \sigma \begin{bmatrix} 0 \\ \times \\ \vdots \\ \times \end{bmatrix}.$$

Similarly, Givens rotation G_2 for eliminating the second entry of $G_1\mathbf{v}$ yields

$$\left(\widetilde{G}_2 \widetilde{G}_1 R G_1^\top G_2^\top\right)(G_2 G_1 \mathbf{v}) = R_2 \begin{bmatrix} 0 \\ 0 \\ \times \\ \vdots \\ \times \end{bmatrix} = \sigma \begin{bmatrix} 0 \\ 0 \\ \times \\ \vdots \\ \times \end{bmatrix}$$

where $R_2 = \widetilde{G}_2 \widetilde{G}_1 R G_1^\top G_2^\top$ is upper triangular, and ultimately we have

$$\widetilde{G}_{k-1} \cdots \widetilde{G}_1 R G_1^\top \cdots G_{k-1}^\top \mathbf{e}_k = \sigma \mathbf{e}_k.$$

The last column of the upper triangular matrix $R_{k-1} = \widetilde{G}_{k-1} \cdots \widetilde{G}_1 R G_1^\top \cdots G_{k-1}^\top$ is thus $[0, \cdots, 0, \sigma]^\top$ as in (2.2.1). The assertion for the lower triangular case follows the similar argument.

Now, let $\widehat{A}$ be the matrix resulting from deleting the top row $\mathbf{a}^\top$ of $A \in \mathbb{R}^{m \times n}$. The approxi-rank $\text{rank}_\theta(\widehat{A})$ may or may not decrease. If $\text{rank}_\theta(A) = 0$, then $\text{rank}_\theta(\widehat{A})$ remains zero, requiring no further computation. For $\text{rank}_\theta(A) = k > 0$, write $A = URV^\top + E$, where columns of $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ form an orthonormal basis for approxi-range $\mathcal{R}_\theta(A)$ approxi-rowspace $\mathcal{R}_\theta(A^\top)$ respectively, $R \in \mathbb{R}^{k \times k}$ is upper triangular with $\sigma_k(R) > \theta$, and $E \in \mathbb{R}^{m \times n}$ with $\|E\| \leq \theta$. Since $A - AVV^\top = E$, we have

$$\begin{bmatrix} \mathbf{a}^\top \\ \widehat{A} \end{bmatrix} - \begin{bmatrix} \mathbf{a}^\top \\ \widehat{A} \end{bmatrix} VV^\top = E = \begin{bmatrix} \mathbf{a}^\top - \mathbf{a}^\top VV^\top \\ \widehat{E} \end{bmatrix},$$

where $\widehat{E} \in \mathbb{R}^{(m-1) \times n}$ is the matrix resulting from deleting the top row of E and $\left\|\widehat{E}\right\| \leq \|E\| \leq \theta$. Consequently, $\widehat{A} = \widehat{A}VV^\top + \widehat{E}$. Let $\widehat{A}V = Q\widetilde{R}$ be the "skinny" QR-factorization of $\widehat{A}V$, then

$$\widehat{A} = Q\widetilde{R}V^\top + \widehat{E}. \tag{2.2.2}$$

Let $\sigma_{\min}(\widetilde{R})$ be the smallest singular value of $\widetilde{R}$. If $\sigma_{\min}(\widetilde{R}) > \theta$, then $\text{rank}_\theta(\widehat{A}) = k$ and (2.2.2) is a URV-plus decomposition of $\widehat{A}$. If $\sigma_{\min}(\widetilde{R}) \leq \theta$, we shall extract the singular value $\sigma_{\min}(\widetilde{R})$ from $\widetilde{R}$. By (2.2.1), two products of Givens rotations G_1 and G_2 exist such that $G_1 \widetilde{R} G_2 = \begin{bmatrix} \widehat{R} & 0 \\ 0 & \sigma_{\min}(\widetilde{R}) \end{bmatrix}$ for certain upper triangular matrix $\widehat{R}$. It follows that

$$\widehat{A} = QG_1^\top \begin{bmatrix} \widehat{R} & 0 \\ 0 & \sigma_{\min}(\widetilde{R}) \end{bmatrix} G_2^\top V^\top + \widehat{E} = [U_{\mathbf{d}}, \mathbf{d}] \begin{bmatrix} \widehat{R} & 0 \\ 0 & \sigma_{\min}(\widetilde{R}) \end{bmatrix} \begin{bmatrix} V_{\mathbf{w}}^\top \\ \mathbf{w}^\top \end{bmatrix} + \widehat{E},$$

where $[U_{\mathbf{d}}, \mathbf{d}] = QG_1^\top$, $U_{\mathbf{d}} \in \mathbb{R}^{m\times(k-1)}$, $\mathbf{d} \in \mathbb{R}^m$, $\begin{bmatrix} V_{\mathbf{w}}^\top \\ \mathbf{w}^\top \end{bmatrix} = G_2^\top V^\top$, $V_{\mathbf{w}} \in \mathbb{R}^{n\times(k-1)}$, and $\mathbf{w} \in \mathbb{R}^n$. Hence,

$$\widehat{A} = U_{\mathbf{d}}\widehat{R}V_{\mathbf{w}}^\top + F, \text{ where } F = \widehat{E} + \sigma_{\min}(\widetilde{R})\mathbf{d}\mathbf{w}^\top. \tag{2.2.3}$$

Since $\mathbf{w}$ is in the approxi-rowspace of $\widehat{A}$ and $\mathbf{d}$ is in the approxi-range of $\widehat{A}$, we have $\|F\| \leq \theta$. Therefore, $\text{rank}_\theta(\widehat{A}) = k-1$ and (2.2.3) is a URV-plus decomposition of $\widehat{A}$.

Since $\text{rank}_\theta(A) = k$ is small, we find the SVD of $\widehat{R}$ directly which gives the left and right singular vectors of $\widehat{R}$ associated with the smallest singular value. The argument is similar when any other row or column of A is deleted.

Our row downdating algorithm `lrowdown` is summarized in Figure 2.2.

Algorithm `lrowdown`

Input: matrix A, approxi-rank k, rank threshold θ, index p of the row to be deleted, matrix V in the USV-plus decomposition.

- `form` the new matrix $\widehat{A}$ by deleting the p-th row of A, `set` $W = \widehat{A}V$.
- `find` the skinny QR factorization of $W = QR$.
- `find` $\sigma_{\min}(R)$ and the corresponding singular vector $\mathbf{v}_{\min}$ by RankRev [24]
- `if` $\sigma_{\min}(R) > \theta$, `then`
  - `set` $S = R$, $U = Q$

    (The approxi-rank stays at k and V does not change).

  `else`
  - `set` $k = k - 1$ (approxi-rank reduced by one)
  - `get` $U_{\mathbf{d}}$, $\widehat{R}$, and $V_{\mathbf{w}}$ as in (2.2.3) using the singular value extracting strategy
  - `set` $S = \widehat{R}$, $U = U_{\mathbf{d}}$ and $V = V_{\mathbf{w}}$.

  `end if`

`Output` k, U, S, V.

Figure 2.2. Algorithm `lrowdown`

2.3 Numerical results on updating and downdating

Our updating and downdating algorithms have been extensively tested for cases of inserting/deleting rows or columns. Since UTV Tools [16] contains only row-updating and row-downdating modules, we shall restrict our comparison with UTV Tools to those situations only. The results of our method for column updating and downdating are quite similar.

The two modules in UTV Tools for updating are **urv_up** and **ulv_up** accounting for inserting a row at the bottom and two modules for downdating are **urv_dw** and **ulv_dw** applying to deleting the top row.

2.3.1 Row-updating with increasing approxi-ranks

The test matrix is initially a 1000×500 matrix having an approxi-rank 10 with threshold 10^{-8}. The approxi-rank gap is $\gamma = 10^3$. After executing **larank** in our LowRank package and modules **lurv** and **lulv** in UTV Tools on this matrix separately for rank-revealing, a random vector is inserted at the bottom at each updating step. Therefore, each update will increase the approxi-rank by one. Our tests show that our **lrowup** code and two counterparts **ulv_up/urv_up** in UTV Tools are all accurate in identifying the increasing approxi-ranks. Table 2.1 shows the execution time, subspace errors and the computed ranks after inserting 30 random rows. Our **lrowup** appears to be considerably faster than modules in UTV Tools, while **urv_up** achieves better accuracy in the updated approxi-range. For better accuracy of our code we add one refinement step in each updating step which helps our code **lrowup** achieve leading accuracy. Nonetheless, it is still faster than **ulv_up** and **urv_up**.

Table 2.1. Results for row-updating with increasing approxi-ranks

	time (seconds)	range error	computed rank
ulv_up	7.08	7e-5	40
urv_up	8.92	2e-8	40
lrowup	0.33	7e-5	40
lrowup_1	4.64	2e-9	40

lrowup_1 *is* **lrowup** *with one refinement step.*

2.3.2 Row-updating without changing approxi-ranks

When approxi-rank does not change in row updating, module **urv_up** in UTV Tools seems to have difficulties in identifying the approxi-ranks during the recursive updating. In contrast, our code `lrowup` always outputs accurate approxi-ranks in all occasions and the speed is about twice as fast on a typical example shown in Table 2.2. The initial matrix has the same features as the example in §2.3.1 except the approxi-rank is set at 130. A sequence of rows consisting of linear combinations of the existing rows are inserted at the bottom one at a time. The approxi-rank stays at 130. However, after certain steps in the recursive updating, **urv_up** outputs inaccurate approxi-ranks.

Table 2.2. Results for row-updating without changing approxi-ranks

		Number of linearly dependent rows inserted						
		1	2	⋯	5	6	7	⋯	10
Time (seconds)	ulv_up	0.42	0.23	⋯	0.30	0.25	0.28	⋯	0.24
	urv_up	0.47	0.30	⋯	0.38	0.23	0.33	⋯	0.28
	lrowup	0.14	0.14	⋯	0.13	0.14	0.14	⋯	0.16
Approxi-range error	ulv_up	3e-8	4e-8	⋯	5e-8	5e-8	5e-8	⋯	5e-8
	urv_up	2e-7	3e-7	⋯	2e-7	(0.65)	(0.85)	⋯	(0.99)
	lrowup	3e-8	4e-8	⋯	6e-8	5e-8	5e-8	⋯	6e-8
Approxi-rank output	ulv_up	130	130	⋯	130	130	130	⋯	130
	urv_up	130	130	⋯	130	(131)	(131)	⋯	(131)
	lrowup	130	130	⋯	130	130	130	⋯	130

Data in parentheses indicate inaccurate computation.

2.3.3 Row-updating with a small approxi-rank gap

This experiment compares the updating performance of our `lrowup` and UTV Tools when the updated matrix has a small approxi-rank gap. The initial matrix is the same as the one in §2.3.1. The inserted vector is a linear combination of existing rows plus a random vector with a norm close to the threshold 10^{-8}. As shown in Table 2.3, all codes produce correct approxi-ranks, while our code `lrowup` takes less execution time and obtains more accurate approxi-range.

Table 2.3. Results for row-updating with a small approxi-rank gap

The updated matrix has approxi-rank 11 with gap 53.8			
	time (seconds)	range error	computed rank
ulv_up	0.25	3e-3	11
urv_up	0.34	5e-7	11
lrowup	0.14	6e-8	11

2.3.4 Row-downdating without changing approxi-ranks

For the case where the approxi-rank remains invariant when a row is deleted, we construct an initial matrix $A \in R^{1000 \times 500}$ with approxi-rank 30 within threshold 10^{-8}. The approxi-rank gap is $\gamma = 10^3$. Then 30 rows consisting of linear combinations of the existing rows of A are generated and stacked on top of A. Deleting those rows one-by-one does not change the approxi-rank. Table 2.4 shows the results of downdating these 30 rows recursively. The results of our code `lrowdown` and its counterparts `ulv_dw` and `urv_dw` in UTV Tools are quite similar in both robustness and accuracy, while our code `lrowdown` runs more than five times as fast as `ulv_dw` and `urv_dw`.

2.3.5 Row-downdating with decreasing approxi-ranks

As mentioned in [16], UTV decomposition may have difficulties in downdating especially when applied to the cases where the approxi-ranks are reduced. This

Table 2.4. Results for row-downdating without changing approxi-ranks

	time (seconds)	range error	computed rank
ulv_dw	14.6	1e-8	30
urv_dw	10.2	2e-9	30
lrowdown	1.80	2e-9	30

phenomenon does occur in the experiment shown below. We downdate a matrix of 1010×500 obtained by stacking 10 random rows at the top of matrix A of size 1000×500 with approxi-rank 50 within threshold 10^{-8}. The approxi-rank gap is set at 10^3. During the test, the 10 random rows are deleted one-by-one. The approxi-rank should decrease by one at every downdating step.

Table 2.5 shows that in step 1 to 4, downdating of the approxi-ranks were all accurate. While all codes exhibit similar accuracy, our code `lrowdown` runs more than twice as fast as `ulv_dw` and `urv_dw`. At step 5, `ulv_dw` miscalculates the approxi-rank by one and this error is carried on to remaining downdating steps. Whereas, our code `lrowdown` always produces the correct approxi-ranks.

Table 2.5. Results for row-downdating with decreasing approxi-ranks

		Number of linearly dependent rows deleted							
		1	2	3	4	5	6	···	10
Time (seconds)	ulv_dw	0.48	0.33	0.33	0.33	0.28	0.30	···	0.44
	urv_dw	0.27	0.20	0.23	0.22	0.19	0.22	···	0.30
	lrowdown	0.11	0.09	0.08	0.08	0.06	0.09	···	0.08
Approxi-range error	ulv_dw	1e-8	2e-8	4e-8	4e-5	(1.0)	(1.0)	···	(1.0)
	urv_dw	1e-8	8e-9	8e-9	1e-8	1e-8	1e-8	···	1e-8
	lrowdown	8e-9	4e-9	6e-9	4e-9	4e-9	4e-9	···	3e-9
Approxi-rank output	ulv_dw	59	58	57	56	(56)	(56)	···	(56)
	urv_dw	59	58	57	56	55	54	···	50
	lrowdown	59	58	57	56	55	54	···	50

Data in parentheses indicate inaccurate computation.

CHAPTER 3

Applications

There are many scientific computing problems where only the dominant part of a matrix is needed. Those problems include information retrieval and image storage to be presented in this section. For matrix $A \in \mathbb{R}^{m \times n}$, write its USV-plus decomposition with approxi-rank k as

$$A \;=\; USV^{\top} + E \text{ where } U \in \mathbb{R}^{m \times k},\; S \in \mathbb{R}^{k \times k},\; V \in \mathbb{R}^{n \times k}.$$

When $k \ll n$, using the dominant part $U\,S\,V^{\top}$ as a low rank approximation to A may reduce the memory cost by an order of magnitude and substantially cut the subsequential computing time, for instance, matrix-vector product $\mathbf{y} \;=\; A\mathbf{x}$ can be approximated by $U[S(V^{\top}\mathbf{x})]$ with $O(n)$ flops instead of $O(n^2)$ if $k = O(1)$.

3.1 Information retrieval: latent semantic indexing

A novel method called *latent semantic indexing*, which uses key words to find relevant documents from a library database, relies critically on the computation of low rank approximation for large matrices [4, 38]. This method can also be applied to webpage search engines [3]. Assume there are m terms $T_1, \cdots, T_m$ extracted from n documents $D_1, \cdots, D_n$. The database can be stored by a term-by-document matrix $A = \left(a_{ij}\right) \in \mathbb{R}^{m \times n}$, where

$$a_{ij} \;=\; \text{the number of times term } T_i \text{ occurs in document } D_j.$$

In other words, the j-th column of A represents the document D_j and the i-th element of the column is the frequency of the term T_i appears in the document. Hence, we call the j-th column of A the *document vector* associated with D_j. For more sophisticated techniques, weighted frequency strategies may be imposed on a_{ij} [2, 12].

The matrix A is usually contaminated with a certain level of noise caused by the presentation style, ambiguity in the use of vocabulary [25], etc. In such situations, using the dominant part $USV^{\top}$ of A will achieve almost the same objective as using A itself, evidenced by our numerical test given below. In practice, k of A_k is much smaller than $\min\{m, n\}$. When a set of key words is submitted, a *query vector* $\mathbf{q}^{\top} = (q_1, \cdots, q_n)$ is formed by letting

$$q_i \;=\; \begin{cases} 1, & \text{if } T_i \text{ appears in the set of key words,} \\ 0, & \text{otherwise.} \end{cases} \tag{3.1.1}$$

Table 3.1. Term-by-document matrix

	The title of article
A1	Updating and downdating an upper trapezoidal sparse orthogonal factorization
A2	A rank revealing method with updating, downdating and applications
A3	A homotopy for solving polynomial systems
A4	UTV tools: MATLAB templates for rank revealing UTV decompositions
A5	Discrete orthogonal polynomials: polynomial modification of a classical functional
A6	Regularity results for solving systems of polynomials by homotopy method
A7	The polynomial rank of a commutative ring
A8	Orthogonal polynomials: applications and computation

The underlined terms are extracted to form the following 12×8 term-by-document matrix

	Document							
Term	A1	A2	A3	A4	A5	A6	A7	A8
application	0	1	0	0	0	0	0	1
decomposition	0	0	0	1	0	0	0	0
downdating	1	1	0	0	0	0	0	0
factorization	1	0	0	0	0	0	0	0
homotopy	0	0	1	0	0	1	0	0
method	0	1	0	0	0	1	0	0
orthogonal	1	0	0	0	1	0	0	1
polynomial	0	0	1	0	2	1	1	1
rank	0	1	0	1	0	0	1	0
revealing	0	1	0	1	0	0	0	0
system	0	0	1	0	0	1	0	0
updating	1	1	0	0	0	0	0	0

The query $\mathbf{q}$ is compared with document D_j, or the document vector $A_k\mathbf{e}_j$, by measuring the magnitude of

$$\cos\theta_j = \frac{\mathbf{q}^\top A_k \mathbf{e}_j}{\|\mathbf{q}\| \, \|A_k \mathbf{e}_j\|}. \tag{3.1.2}$$

The larger this magnitude is the more relevant the document D_j relates to the query $\mathbf{q}$.

Table 3.1 demonstrates a small size database consists of 12 terms chosen from titles of 8 articles and the corresponding term-by-document matrix. When a user submits a set of key words: *rank*, *revealing*, *updating*, *downdating*, *application*, the associated query vector is

$$\mathbf{q} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}^\top.$$

By using rank $k = 3$ approximation to the term-by-document matrix, the first three most relevant articles will be A2 (0.9136), A4 (0.7844), and A1 (0.5917), where the number in each parenthesis indicates the cosine of the angle of the query vector and the corresponding document vector.

From our rank-revealing method, the decomposition of $A_k = USV^\top$, where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ have orthonormal columns and $S \in \mathbb{R}^{k \times k}$, reduces the storage of database as well as the amount of the computations of the magnitude in (3.1.2) as shown below. Set $W = SV^\top \in \mathbb{R}^{k \times n}$ and rewrite (3.1.2) by

$$\cos\theta_j = \frac{\mathbf{q}^\top USV^\top \mathbf{e}_j}{\|\mathbf{q}\| \, \|USV^\top \mathbf{e}_j\|} = \frac{\mathbf{q}^\top U \left(SV^\top \mathbf{e}_j\right)}{\|\mathbf{q}\| \, \|SV^\top \mathbf{e}_j\|} = \frac{\mathbf{q}^\top U \left(W \mathbf{e}_j\right)}{\|\mathbf{q}\| \, \|W \mathbf{e}_j\|}. \tag{3.1.3}$$

Consequently, the storage of database is reduced from mn to $(m+n)k$ by saving matrices U and W instead of saving the whole matrix A_k. For computing $\cos\theta_j$ for all documents with respect to a given query, we normalize all columns of W which requires less computation on normalizing all columns of A_k. In addition, when a term or a document is added in or removed from the database, our updating and downdating methods can be applied.

Table 3.2. Comparisons for a 3000 × 1400 term-by-document matrix from CRAN.

threshold θ	approxi-rank k	compression ratio $\frac{mn}{(m+n)k}:1$	running time (seconds)			
			larank	lurv	lulv	SVD
$12\% \times \|A\|$	56	17.0 : 1	7.92	75.0	66.3	61.0
$10\% \times \|A\|$	70	13.6 : 1	11.6	67.4	65.8	
$8\% \times \|A\|$	90	10.6 : 1	16.5	85.6	80.3	

We use a standard document collection CRAN [9, 31] to be our test sample. The collection provides about 30000 terms selected from 1400 documents. We choose first 3000 terms from CRAN to form a term-by-document matrix $A \in \mathbb{R}^{3000 \times 1400}$ and execute our algorithm **larank**, Matlab built-in **svd** function as well as two codes **lurv** and **lulv** in UTV tools for three different prescribed thresholds. The approxi-rank k's, the compression ratios and the running time of computing the decomposition of A_k's are shown in Table 3.2, which illustrates the considerable efficiency of our algorithm **larank**.

Using those three databases calculated by our algorithm **larank** and the raw database A, we submit a query with seven key words: *thick*, *ring*, *part*, *slight*, *downstream*, *yaw*, and *clamp* to compare the retrieval results. Table 3.3 lists the indices of the first eight relevant documents for each database. It shows that three retrieval results from low rank databases have at least six same documents as the retrieval result from the raw database. However, as shown in Table 3.2, the decomposition of low rank databases requires much less storage. Table 3.4 compares the running time of

Table 3.3. The retrieval results for three lower rank databases and the raw database.

database	The first 8 relevant documents							
	1st	2nd	3rd	4th	5th	6th	7th	8th
A_k, $k = 56$	**766**	**1031**	364	**512**	**680**	**857**	**733**	26
	(.5507)	(.5026)	(.4706)	(.4696)	(.4541)	(.4469)	(.4363)	(.4340)
A_k, $k = 70$	**766**	**1031**	**512**	**733**	**680**	**857**	**943**	513
	(.5495)	(.4908)	(.4779)	(.4585)	(.4580)	(.4365)	(.4325)	(.4309)
A_k, $k = 90$	**766**	**1031**	**512**	**680**	926	**733**	**943**	**857**
	(.5536)	(.4845)	(.4780)	(.4566)	(.4359)	(.4348)	(.4344)	(.4315)
A	766	1031	512	680	943	201	733	857
	(.4880)	(.4725)	(.4558)	(.4558)	(.4364)	(.4226)	(.4193)	(.4193)

The number above the parenthesis is the index j of document D_j. The number in parenthesis represents $\cos\theta_j$ with respect to the query and the corresponding document D_j. Boldfaced numbers are the indices of the common documents in the retrieval result from the raw database A.

adding 10 rows (terms) on the bottom of database A_k with $k = 56$ by using our updating algorithm `lrowup` and two updating codes **urv_up** and **ulv_up** in UTV tools. The comparisons for removing top 5 rows (terms) from database A_k with $k = 56$ are shown in Table 3.5.

Table 3.4. Results for updating database

code	lrowup	urv_up	ulv_up
time (seconds)	1.95	14.2	12.8

Comparison results for updating 10 rows on the bottom of the database A_k with $k = 56$.

Table 3.5. Results for downdating database

code	lrowdown	urv_dw	ulv_dw
time (seconds)	3.00	6.59	7.09

Comparison results for downdating 5 *top rows from the database* A_k *with* $k = 56$.

3.2 Image processing: saving storage of photographs

An image can be stored in a matrix whose entries correspond to the levels of color intensity [1] at pixels. In certain situations, a huge number of images need to be archived while high resolution is not essential, like fingerprints. Our USV-plus decomposition can greatly reduce the storage while maintaining an acceptable quality of the images.

With a certain color map which associates a number with a level of color intensity built in a photograph formation device such as cameras and scanners, the device partitions an image by an $m \times n$ lattice and fits a number for each cell (pixel) according to the color map, resulting in an $m \times n$ matrix [11, 22]. Figure 3.1 demonstrates that an image of a baseball is partitioned by a 480×640 lattice and each cell corresponds to a gray level which ranges from 0 to 255 to form a matrix $A \in \mathbb{R}^{480 \times 640}$.

Table 3.6. Comparison results for a 480×640 fingerprint image matrix.

threshold	approxi-rank	compression ratio	running time (seconds)			
θ	k	$\frac{mn}{(m+n)k} : 1$	larank	lurv	lulv	SVD
$2.1\% \times \|A\|$	18	15.2 : 1	0.17	2.78	2.78	
$1.3\% \times \|A\|$	34	8.07 : 1	0.34	5.31	5.28	1.97
$0.8\% \times \|A\|$	51	5.38 : 1	0.67	7.66	7.61	

Figure 3.1. The photograph formation process: lattice partition and assignment

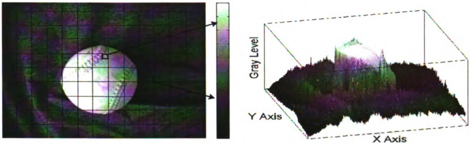
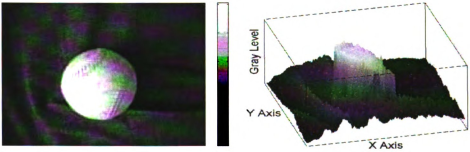


Figure 3.2. Rank 21 approximation of Figure 3.1



Generally, most of the singular values of photograph matrices are relatively small [23]. When we truncate those terms with small singular values $\sigma_{k+1}, \cdots, \sigma_n$ from the SVD of $A = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^\top$, the image from the resulting matrix A_k still maintains the main feature of the image from A. Because, by writing $A_k = A - E$, the 2-norm of E is relatively small as shown in §1.1, thus $A_k \approx A$. Figure 3.2 shows a lower rank approximation image of the picture in Figure 3.1 by truncating those singular values less than 1.3% of the largest singular value σ_1. We can see that the main objects and contours still can be recognizable.

Using the dominant part of matrix A reduces the storage space from mn to $(m+n)k$ by saving $\mathbf{u}_j$ and $\sigma_j \mathbf{v}_j$ for $j = 1, \cdots, k$ instead of the whole $m \times n$ matrix.

Figure 3.3. The original image and three lower rank approximation images

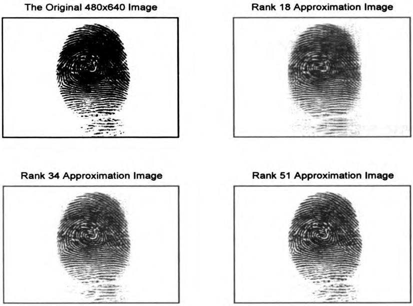


Figure 3.3 illustrates a 480×640 fingerprint photograph from FVC2004 [17] along with three lower rank approximation images. The color map is the same as the one used in Figure 3.1. Table 3.6 shows the compression ratio for each image and the comparisons of the running time for computing the decomposition of matrices by using `larank`, `lurv`, `lulv`, and `svd`. Again, the efficiency of our algorithm `larank` seems to dominate the existing codes.

BIBLIOGRAPHY

[1] T. Acharya and A. K. Ray, *Image Processing Principles and Applications*, Wiley, Hoboken, NJ, 2005.

[2] M. W. Berry, *Using linear algebra for intelligent information retrieval*, SIAM Rev., 37(1995), pp. 573–595.

[3] M. W. Berry and M. Browne, *Understanding Search Engines: Mathematical Modeling and Text Retrieval*, SIAM, Philadelphia, 1999.

[4] M. W. Berry and R. D. Fierro, *Low-rank orthogonal decompositions for information retrieval applications*, Numerical Linear Algebra with Applications. 3(1996), pp. 301–328.

[5] C. H. Bischof and G. Quintana-Orti, *Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices*, ACM Trans. Math. Software, 24(1998), pp. 254–257.

[6] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[7] T. R. Chan, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.

[8] T. R. Chan and P. C. Hansen, *Low-rank revealing QR factorizations*, Numerical Linear Algebra with Applications, 1(1994), pp. 33–44.

[9] Cornell SMART system, *ftp://ftp.cs.cornell.edu/pub/smart.*

[10] B. H. Dayton and Z. Zeng, *Computing the multiplicity struture in solving polynomial systems*, Proceedings of ISSAC '05, ACM Press, pp. 116–123, 2005.

[11] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[12] S. DEERWESTER, S. T. DUMAIS, G. W. FURNAS, T. K. LANDAUER AND R. HARSHMAN, *Indexing by latent semantic analysis*, J. Amer. Soc. Inform. Sci., 41(1990), pp. 391–407.

[13] F. DEPRETTERE, *SVD and Signal Processing, Algorithms, Applications, and Architectures*, North–Holland, Amsterdam, 1988.

[14] R. D. FIERRO, AND J. R. BUNCH, *Bounding the subspaces from rank revealing two-sided orthogonal decompositions*, SIAM J. Martix Analysis and Applicatons, 16(3), pp. 743–759, 1995.

[15] R. D. FIERRO, AND P. C. HANSEN, *Low-rank revealing UTV decompositions*, Numer. Algorithms, 15(1997), pp. 37–55.

[16] R. D. FIERRO, P. C. HANSEN, AND P. S. K. HANSEN, *UTV tools: MATLAB templates for rank-revealing UTV decompositions*, Numer. Algorithms, 20(1999), pp. 165–194.

[17] FVC 2004 DATABASE, *http://biometrics.cse.msu.edu/fvc04db.*

[18] G. H. GOLUB, V. KLEMA, AND G. W. STEWART, *Rank Degeneracy and Least Squares Problems*, Tech. rep. TR 456, University of Maryland, Baltimore, MD, 1976.

[19] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.

[20] R. A. HORN, AND C. R. JOHNSON, *Matrix Analysis*, Cambridge University Press, 1985.

[21] T.-M. HWANG, W.-W. LIN AND E.K. YANG, *Rank-revealing LU Factorization*, Linear Algebra and its Applications, 175(1992), pp. 115–141.

[22] A. K. JAIN, *Fundamentals of Diginal Image Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[23] S. J. LEON, *Linear Algebra with Applications*, Pearson Prentice Hall, NJ, 2006.

[24] T. Y. LI AND Z. ZENG, *A rank-revealing method with updating, downdating, and applications*, SIAM J. Martix Analysis and Applicatons, 26(2005), pp. 918–946.

[25] C. D. MEYER, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2000.

[26] L. MIRANIAN AND M. GU, *Strong rank revealing LU factorization*, Linear Algebra and its Applications, 367(2003), pp 1–16.

[27] L. MIRSKY, *Symmetric gauge functions and unitarily invariant norms*, Quart. J. Math. Oxford Ser. 11(1960), pp. 50–59.

[28] M. MOONEN AND B. DE MOOR, *SVD and Signal Processing, III, Algorithms, Applications, and Architectures*, Elsevier, Amsterdam, 1995.

[29] C.-T. PAN, *On the existence and computation of rank-revealing LU factorizations*, Linear Algebra and its Applications, 316(2000), pp. 199–222.

[30] H. PARK AND L. ELDEN, *Downdating the Rank Revealing URV Decomposition*, SIAM J. Martix Analysis and Applicatons, 16, pp. 138–155, 1995.

[31] G. SALTON AND M. MCGILL, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983.

[32] G. W. STEWART, *An Updating Algorithm for Subspace Tracking*, IEEE Trans. Signal Processing, 40, pp. 1535–1541, 1992.

[33] G. W. STEWART, *Updating a Rank-Revealing ULV Decompositions*, SIAM J. Martix Analysis and Applicatons, 14, pp. 494–499, 1993.

[34] G. W. STEWART, *UTV decompositions*, in Numerical Analysis 1993, D. F. Griffith and G. A. Watson, eds., Pitman Res. Notes Math. Ser. 303, Longman, Harlow, UK 1994, pp. 225–236.

[35] G. W. STEWART, *Matrix Algorithms: Basic Decompositions*, SIAM, Philadelphia, 1998.

[36] R. VACCARO, *SVD and Signal Processing, II, Algorithms, Applications, and Architectures*, Elsevier, Amsterdam, 1991.

[37] Z. ZENG, *Computing multiple roots of inexact polynomials*, Mathematics of Computation, 74, pp. 869–903, 2005.

[38] H. ZHA, AND H. D. SIMON, *Timely communication on updating problems in latent semantic indexing*, SIAM J. Sci. Comput., 21(2), pp. 782–791, 1999.