LBNL-44003
Preprint

Ernest Orlando Lawrence Berkeley National Laboratory

Low-Rank Approximations with Sparse Factors I: Basic Algorithms and Error Analysis

Zhenyue Zhang, Hongyuan Zha, and Horst Simon

National Energy Research Scientific Computing Division

July 1999

Submitted to
SIAM Journal of Matrix Analysis

LBNL-44003

Low-Rank Approximations with Sparse Factors I: Basic Algorithms and Error Analysis

Zhenyue Zhang, Hongyuan Zha, and Horst Simon

National Energy Research Scientific Computing Division
Ernest Orlando Lawrence Berkeley National Laboratory
University of California
Berkeley, California 94720

July 1999

This work was supported by the Director, Office of Science, Office of Laboratory Policy and Infrastructure Management, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. Computing resources were supported by the Director, Office of Advanced Scientific Computing Research, Division of Mathematical, Information, and Computational Sciences of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. Work was also supported in part by NSFC under Project No. 19771073 and National Science Foundation Grant Nos. CCR-9619452 and CCR-9901986.

LOW-RANK APPROXIMATIONS WITH SPARSE FACTORS I: BASIC ALGORITHMS AND ERROR ANALYSIS

ZHENYUE ZHANG*, HONGYUAN ZHA†, AND HORST SIMON‡

Abstract. We consider the problem of computing low-rank approximations of matrices. The novel aspects of our approach are that we require the low-rank approximations be written in a factored form with the factors having certain sparsity patterns and the degree of sparsity of the factors can be traded off for reduced reconstruction error by certain user determined parameters. We give a detailed error analysis of our proposed algorithms and compare the computed sparse low-rank approximations with those obtained from singular value decomposition. We present numerical examples arising from several application areas to illustrate the efficiency and accuracy of our algorithms.

1. Introduction. We consider the problem of computing low-rank approximations of a given matrix $A \in \mathcal{R}^{m\times n}$ which arises in many applications areas, see [9, 12] for a few examples. The theory of singular value decomposition (SVD) provides the following characterization of the best low-rank approximations of A in terms of Frobenius norm $\|\cdot\|_F$ [4].

THEOREM 1.1. *Let the singular value decomposition of* $A \in \mathcal{R}^{m\times n}$ *be* $A = U\Sigma V^T$,

$$\Sigma = \mathrm{diag}(\sigma_1, \dots, \sigma_{\min(m,n)}), \quad \sigma_1 \geq \dots \geq \sigma_{\min(m,n)},$$

and U *and* V *orthogonal. Then for* $1 \leq k \leq \min(m,n)$,

$$\sum_{i=k+1}^{\min(m,n)} \sigma_i^2 = \min\{\, \|A - B\|_F^2 \mid \mathrm{rank}(B) \leq k\}.$$

And the minimum is achieved with $\mathrm{best}_k(A) \equiv U_k \,\mathrm{diag}(\sigma_1, \dots, \sigma_k) V_k^T$, *where* U_k *and* V_k *are the matrices formed by the first* k *columns of* U *and* V, *respectively. Furthermore,* $\mathrm{best}_k(A)$ *is unique if and only if* $\sigma_k > \sigma_{k+1}$.

For any low-rank approximation B of A, we call $\|A - B\|_F$ the *reconstruction error* of using B as an approximation of A. By Theorem 1.1, $\mathrm{best}_k(A)$ has the smallest reconstruction error among all the rank-k approximations of A. In some applications,

* Center for Mathematical Sciences & Department of Applied Mathematics, Zhejiang University, Hangzhou, 310027, P. R. China. zyzhang@math.zju.edu.cn, and National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory, One Cyclotron Road, M/S: 50F, Berkeley, CA 94720, USA. The work of this author was supported in part by NSFC (project 19771073), Zhejiang Provincial Natural Science Foundation of China, and Scientific Research Foundation for Returned Overseas Chinese Scholars, State Education Commission. The work also was supported in part by NSF grants CCR-9619452 and by the Director, Office of Science, Office of Laboratory Policy and Infrastructure Management, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. Computing resources were supported by the Director, Office of Advanced Scientific Computing Research, Division of Mathematical, Information, and Computational Sciences of the U.S. Department of Energy under contract number DE-AC03-76SF00098.

†Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802, zha@cse.psu.edu. The work of this author was supported in part by NSF grants CCR-9619452 and CCR-9901986.

‡National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory, One Cyclotron Road, M/S: 50B, Berkeley, CA 94720, HDSimon@lbl.gov. This work was supported by the Director, Office of Science, Office of Laboratory Policy and Infrastructure Management, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. Computing resources were supported by the Director, Office of Advanced Scientific Computing Research, Division of Mathematical, Information, and Computational Sciences of the U.S. Department of Energy under contract number DE-AC03-76SF00098.

it is desirable to impose further constraints on the low-rank approximation B (besides being low rank). For instance, even if the matrix A is sparse, it is generally not true that $\text{best}_k(A)$ or U_k and V_k will also be sparse. Therefore, the storage requirement of $\text{best}_k(A)$ in the factored form $\text{best}_k(A) = U_k \operatorname{diag}(\sigma_1, \ldots, \sigma_k)V_k^T$ can be even greater than that of the original matrix A. To overcome this difficulty, we seek to find low-rank approximations with sparsity properties. One possibility will be to impose some sparsity requirements directly on the low-rank approximation B itself, i.e., we require that B be sparse. However, this approach is less flexible and it is very hard to achieve a reasonable reconstruction error (as compared with that obtained from $\text{best}_k(A)$, for example) using a sparse B. Besides, it is not straightforward to construct low-rank matrices with a given sparsity patterns. Inspired by the work reported in [6, 10], we consider the approach of writing B in a factored form as $B = XDY^T$, and imposing sparsity requirements on the factors X and Y instead while keeping D in diagonal form. Therefore, even though X and Y are sparse B may be rather dense, and this actually gives the flexibility to achieve smaller reconstruction errors. On the other hand, the low-rank constraint on B is automatically imposed by writing B in the factored form, i.e., $\text{rank}(B) \leq k$ if X has k columns. Although the focus of this paper is on imposing sparsity constraints, we should also mention that other constraints on the low-rank approximations may also be desirable: in probabilistic Latent Semantic Indexing [5], for example, elements of columns X and Y represent conditional probabilities, and therefore are required to be nonnegative. As another example, in the so-called structured total least squares problems, the low-rank approximations need to have certain structures such as Toeplitz or Hankel.

The rest of the paper is organized as follows: In section 2, we cast the problem of computing sparse low-rank approximations in the framework of an optimization problem. We then propose algorithms and heuristics for finding approximate optimal solutions of this optimization problem. In section 3, we give a detailed error analysis of the proposed algorithms and heuristics. Specifically, we prove that the reconstruction errors of the computed sparse low-rank approximations are with a constant factor of those that are obtained by SVD. In section 4, we discuss several computational variations of the basic algorithms proposed in section 2 and in section 5 we conduct several numerical experiments to illustrate the various numerical and efficiency issues of our proposed algorithms. We also compared the low-rank approximations computed by our algorithms with those obtained by SVD and the approaches developed in [10]. In section 6, we summarize our contribution and point out future research directions.

2. Sparse low-rank approximation. Computing low-rank approximations with sparse factors has been considered by several authors before. In [6] Kolda and O'Leary propose the so-called *semi discrete decomposition* (SDD) where they write a low-rank approximation as $B_k = X_k D_k Y_k^T$ with $X_k \in \mathcal{R}^{m\times k}$, $Y_k \in \mathcal{R}^{k\times n}$, and D_k is diagonal. Furthermore, they require that X_k and Y_k contain elements drawn from the set $\{-1, 0, 1\}$. The restriction on the elements of X_k and Y_k usually demands a much larger $k \gg K$ in order for B_k to achieve the comparable reconstruction error as that of $\text{best}_K(A)$, and therefore the low-rank property of B_k may not hold. But usually the storage requirement of B_k in the factored form is much lower than that of A, and this is certainly the major strength of SDD. In [10] Stewart proposes to construct low-rank approximations of a *sparse* matrix A by selecting certain columns and rows of it, i.e., he writes a low-rank approximation as $B_k = A_c M A_r^T$, where A_c and A_r^T are certain k columns and k rows of A, respectively, and M is chosen to minimize the the error $\|A - A_c M A_r^T\|_F$ once the left and right factors A_c and A_r are chosen. A_c and

A_r are determined by variations of QR algorithms with certain pivoting strategy. In general, the matrix M will be dense. Due to the denseness of M, storage requirement of B_k can become dramatically higher as k increases. Numerical experiments showed that Stewart's approach is especially effective when A itself is close to rank-deficient. The approach we now propose can be considered as a compromise of the above two approaches: we want to have a low-rank approximation and at the same time we also want to have greater control of the sparsity properties of the approximation. To this end, we consider the following general minimization problem.

$$\text{(2.1)} \quad \min\{\|A - X_k D_k Y_k^T\|_F \mid D \text{ diagonal}, X_k \in \mathcal{R}^{m\times k} \text{ and } Y_k \in \mathcal{R}^{n\times k} \text{ sparse }\}.$$

The above optimization problem in its present form is ill-defined because the minimum depends on the sparsity constraints: the number of nonzeros of the left and right factors and the positions of those nonzero elements which constitute what we call their *sparse patterns*. So *ideally* the goal is to make the reconstruction error $\|A - X_k D_k Y_k^T\|_F$ as small as possible and keep in mind the following questions:

- How to determine good sparse patterns for the left and right factors?
- How to find the best approximation $B_k = X_k D_k Y_k^T$ with the chosen sparse structures of X_k and Y_k?

In this paper we will not discuss how to impose the sparsity constraints on the fators X_k and Y_k in general, but rather start with an heuristic. In this section, we propose the framework of our sparse low-rank approximation (SLRA) approach and discuss several of its computational variations in Section 4. As can be seen, the heuristic *dynamically and implicitly* imposes sparsity constraints on X_k and Y_k.

Algorithm SLRA (Sparse low-rank approximation). Given a matrix $A \in \mathcal{R}^{m\times n}$ and an integer $k \le \min\{m, n\}$, this algorithm produces a diagonal matrix D_k, and sparse matrices X_k and Y_k. At the conclusion of the algorithm, $B_k \equiv X_k D_k Y_k^T$ gives a low-rank approximation of A with sparse factors.

1. [Initialize] Set $A_0 = A$.
2. For $i = 1, 2, \cdots, k$
  2.1 [Rank-one approximation] Find a *sparse* rank-one approximation $x_i d_i y_i^T$ to A_{i-1} with sparse unit vectors x_i and y_i.
  2.2 Set $A_i = A_{i-1} - x_i d_i y_i^T$.

The core structure of Algorithm SLRA is a sequence of k deflation steps [8] which allows us to build a low-rank approximation one rank at a time. This general approach is also adopted in [6], but the actual deflation step is very different from ours. After k steps, $A_k = A - X_k D_k Y_k$ with $X_k = [x_1, \cdots, x_k]$, $Y_k = [y_1, \cdots, y_k]$ and $D_k = \text{diag}(d_1, \cdots, d_k)$.

It is worthwhile to point out that the integer k, the rank of B_k in general, can be determined by the stopping criterion $\|A - X_k D_k Y_k^T\|_F \le \texttt{tol}$ because the error $\|A - X_k D_k Y_k^T\|_F = \|A_k\|_F$ can be easily calculated by a recurrence relation, see Section 4 for more details.

The key step of Algorithm SLRA is **Step 2.1**, i.e., computing *sparse* rank-one approximations. By Theorem 1.1 the best rank-one approximation to A is given by $u\sigma v^T$ with $\{u, \sigma, v\}$ the largest singular triplet of A. The triplet $\{u, \sigma, v\}$ can also be used to produce a good sparse rank-one approximation. The basic idea is to sparsify u and v to get sparse vectors x and y, and choose a scalar d such that

$$\|A - xdy^T\|_F = \min_s \|A - xsy^T\|_F.$$

Since u and v will undergo this sparsification process, it is not necessary to compute them to high accuracy. Some less expensive approximation will do and this results in faster algorithms. Now we give more details of the computation of the sparse rank-one approximation, **Step 2.1**.

Step 2.1 of SLRA (Sparse rank-one approximation.) Given a matrix A, this algorithm produces a rank-one matrix xdy^T with sparse vectors x and y.

1. Compute (approximations of) the largest left and right singular vectors u and v of A.

2. Sparsify u and v to get sparse vectors x and y with $\|x\| = \|y\| = 1$.

2.1 [Sort] Sort the entries of u and v in two sections:

$$P_1 u = \begin{bmatrix} u_+ \\ u_- \end{bmatrix}, \quad P_2 v = \begin{bmatrix} v_+ \\ v_- \end{bmatrix},$$

where P_1 and P_2 are the permutation matrices resulted from the sorting process.

2.2 [Sparsify] Discard the second sections u_- and v_- to get sparse vectors x and y:

$$x \leftarrow P_1^T \begin{bmatrix} u_+ \\ 0 \end{bmatrix} / \|u_+\|, \quad y \leftarrow P_1^T \begin{bmatrix} v_+ \\ 0 \end{bmatrix} / \|v_+\|.$$

3. Set $d \equiv x^T A y$ which minimizes

$$\{\|A - xsy^T\|_F \mid s \text{ scalar}\}.$$

Two aspects of Algorithm SLRA are still left to be specified: 1) the determination of k and 2) the partition of the u and v into sections. Since our error analysis of Algorithm SLRA does not depend on these two issues, we will delay their discussion to Section 4. We now present an example to illustrate the low-rank approximations computed by Algorithm SLRA.

EXAMPLE. This example is taken from [2]. We have a list of book titles. Figure 2.1 plots the 16×17 term-document matrix $A = (a_{ij})$ with a_{ij} represents the number of times term[1] i appears in title j. Below is the list of terms.

[1] In this example, a term is just a word since we do not use multi-word phrases.

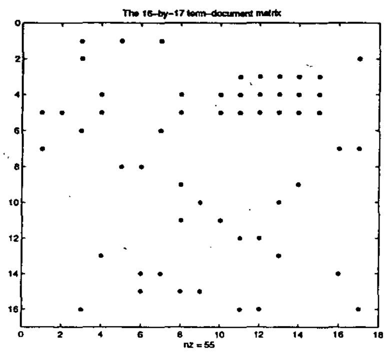

FIG. 2.1. *The* 16×17 *term-document matrix* A.

1	algorithm	9	methods
2	application	10	nonlinear
3	delay	11	ordinary
4	differential	12	oscillation
5	equations	13	partial
6	implementation	14	problem
7	integral	15	systems
8	introduction	16	theory

The index of the terms correspond to the row number of the matrix A. We apply the *Separated* variation of Algorithm SLRA to the term-document matrix A, choosing $k = 2$ and $\epsilon = 0.1$ (see Section 4 for details). In the following we list the nonzero components of $x_i, y_i, i = 1, 2$ arranged in nondecreasing order and their corresponding row indexes.

index	x_1	index	x_2	index	y_1	index	y_2
5	0.7125	14	0.4944	15	0.3671	7	0.4789
4	0.5320	1	0.4825	14	0.3421	3	0.4332
3	0.3315	7	0.3772	11	0.3313	16	0.3958
16	0.1409	6	0.3148	12	0.3313	6	0.3624
13	0.1329	8	0.3009	4	0.3209	5	0.3557
11	0.1250	16	0.2899	13	0.3207	17	0.3445
9	0.1200	2	0.2622	10	0.3190	1	0.1585
12	0.1192	15	0.1824	2	0.2874	9	0.0987
7	0.0881			8	0.2806	15	-0.0488
				1	0.2284	11	0.0485
						12	0.0485

The decomposition we computed above has a very interesting interpretation: the two triplets $\{x_1, d_1, y_1\}$ and $\{x_2, d_2, y_2\}$ divide the 17 book titles into two topics. The first topic is about *differential equations* and the second *algorithms and systems.* The nonzero elements of x_i specify the most influential words for the topic, and nonzero elements of y_i specify those book tiles that belong to this topic. Below we list the influential words for topic *differential equations* and *algorithms and systems*, respectively.

Differential Equations		Algorithms and Systems	
5	equations	14	problem
4	differential	1	algorithm
3	delay	7	integral
16	theory	6	implementation
13	partial	8	introduction
11	ordinary	16	theory
9	methods	2	application
12	oscillation	15	systems
7	integral		

The indexes corresponding to y_1 indicate the book titles which deal with the first topic while the indexes corresponding to y_2 indicate the book titles which deal with the second topic.

3. Error analysis. In this section we will compare the low-rank approximations computed by Algorithm SLRA with those obtained by SVD. One potential alternative is to make the comparison directly with the optimal solutions of (2.1) assuming we have made more specifications on the sparsity of X_k and Y_k, for example, we can impose constraints on the number of nonzeros of X_k and Y_k. This approach at the moment is rather difficult to pursue because we still do not have a good understanding of the structures of the optimal solutions (2.1). Fortunately, $\text{best}_k(A)$ obtained from SVD gives the optimal solutions for (2.1) when there are no sparsity constraints on X_k and Y_k, and the heuristic of Algorithm SLRA takes advantage of this connection. Therefore we choose to compare with $\text{best}_k(A)$ computed by SVD. To proceed, we first consider the rank-one case, assuming we have computed the largest singular triplet *exactly*. Throughout the rest of the paper, we assume that $A \in \mathcal{R}^{m\times n}$.

THEOREM 3.1. *Let $\{u, \sigma, v\}$ be the largest singular triplet of A. Using the same notation as in Step A2.1 of Algorithm SLRA, and assume that $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$ with $\epsilon \le 1/\sqrt{3}$. Then*

$$\|A - xdy^T\|_F \le \sqrt{1+\alpha\tau}\,\|A - u\sigma v^T\|_F,$$

where

$$\alpha = \frac{\sigma_1^2}{\sum_{j=2}^n \sigma_j^2}, \quad \tau = 4\epsilon^2\left(1 - \frac{\epsilon^4}{(1-\epsilon^2)^2}\right) < 4\epsilon^2.$$

Proof. Notice that d is chosen such that $\|A - xdy^T\|_F^2 = \|A\|_F^2 - d^2$, we need to derive a lower bound for $|d|$. To this end, partition

$$P_1 A P_2^T = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

conformally with those of $P_1 u$ and $P_2 v$ (see Step A2.1 of Algorithm SLRA). It follows from the choice of d that

$$d = x^T A y = u_+^T A_{11} v_+ / (\|u_+\| \cdot \|v_+\|).$$

Recalling that $Au = \sigma v$ and $A^T v = \sigma u$, we obtain

$$u_+^T A_{11} v_+ + u_+^T A_{12} v_- = \sigma\|u_+\|^2, \quad u_-^T A_{21} v_+ + u_-^T A_{22} v_- = \sigma\|u_-\|^2,$$

and similarly, we have

$$v_+^T A_{11}^T u_+ + v_+^T A_{21}^T u_- = \sigma \|v_+\|^2, \quad v_-^T A_{12}^T u_+ + v_-^T A_{22}^T u_- = \sigma \|v_-\|^2.$$

A simple calculation yields that

$$u_+^T A_{11} v_+ = u_-^T A_{22} v_- + \sigma(1 - \|u_-\|^2 - \|v_-\|^2).$$

Thus,

$$\begin{aligned} |d| &\geq \frac{\sigma(1 - \|u_-\|^2 - \|v_-\|^2) - \sigma \|u_-\| \cdot \|v_-\|}{\|u_+\| \cdot \|v_+\|} \\ &\geq \sigma \frac{1 - \frac{3}{2}(\|u_-\|^2 + \|v_-\|^2)}{1 - \frac{1}{2}(\|u_-\|^2 + \|v_-\|^2)} \\ &\geq \sigma \frac{1 - 3\epsilon^2}{1 - \epsilon^2} \\ &= \sigma \left(1 - \frac{2\epsilon^2}{1 - \epsilon^2}\right) \geq 0. \end{aligned}$$

Here we used that fact that $\|A_{22}\| \leq \|A\| = \sigma$. It follows that

$$\|A - xdy^T\|_F^2 \leq \sum_{j=1}^{\min(m,n)} \sigma_j^2 - \sigma_1^2 \left(1 - \frac{2\epsilon^2}{1 - \epsilon^2}\right)^2 = (1 + \alpha\tau)\|A - u\sigma v^T\|_F^2,$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min m,n}$ are the singular values of A and

$$\tau = 1 - \left(1 - \frac{2\epsilon^2}{1 - \epsilon^2}\right)^2 = 4\epsilon^2 \left(1 - \frac{\epsilon^2}{(1 - \epsilon^2)^2}\right),$$

completing the proof. □

In practical situations the exact largest singular triplet is not available and as we mentioned before it may not be even desirable to have it computed to high accuracy since we will sparsify u and v by throwing away some of their nonzero elements anyway during the sparsification process. Hence, we need to consider the case when we only have approximations of the left and right singular vectors.

THEOREM 3.2. *Let $\{u, v\}$ be approximate largest left and right singular vectors of A and $\sigma = \sigma_1(A)$. Using the notation of Step A.2.1 of Algorithm SLRA, and assume that $\|u_-\|^2 + \|v_-\|^2 \leq 2\epsilon^2$. Then*

$$\|A - xdy^T\|_F \leq \sqrt{1 + \alpha(\tau + \delta)}\|A - u\sigma v^T\|_F,$$

where

$$\delta = \frac{2 - 6\epsilon^2 - \eta}{1 - 2\epsilon^2}\eta, \quad \eta = \frac{\|Av - \sigma u\| + \|A^T u - \sigma v\|}{2\sigma}.$$

Proof. Define $r_1 = P_1(Av - \sigma u)$ and $r_2 = P_2^T(A^T u - \sigma v)$. Similarly as in the proof of Theorem 3.1, we have

$$u_+^T A_{11} v_+ = u_-^T A_{22} v_- + \sigma(1 - \|u_-\|^2 - \|v_-\|^2) + \left([u_+^T, u_-^T]r_1 + [v_+^T, v_-^T]r_2\right)/2,$$

which yields

$$
\begin{aligned}
|d| &\geq \sigma\left(1 - \frac{2\epsilon^2}{1-\epsilon^2}\right) - \frac{\|r_1\| + \|r_2\|}{2\sqrt{1-2\epsilon^2}} \\
&= \sigma\left(1 - \frac{2\epsilon^2}{1-\epsilon^2} - \frac{\eta}{\sqrt{1-2\epsilon^2}}\right).
\end{aligned}
$$

The result follows because

$$
\begin{aligned}
1 - \left(1 - \frac{2\epsilon^2}{1-\epsilon^2} - \frac{\eta}{\sqrt{1-2\epsilon^2}}\right)^2 &= \tau + \left(\frac{2-6\epsilon^2}{1-\epsilon^2} - \frac{\eta}{\sqrt{1-2\epsilon^2}}\right)\frac{\eta}{\sqrt{1-2\epsilon^2}} \\
&\leq \tau + \frac{\eta(2-6\epsilon^2-\eta)}{1-2\epsilon^2} = \tau + \delta,
\end{aligned}
$$

completing the proof. □

REMARK. We notice that η measures the accuracy of the approximate left and right singular vectors in a certain relative sense. From Theorem 3.1, $\tau = O(\epsilon^2)$. If ϵ is fixed, there is no point to compute u and v to higher accuracy than $O(\epsilon^2)$. On the other hand, given approximate u and v and the corresponding η, we should choose ϵ to match their accuracy, i.e., $\epsilon = O(\sqrt{\eta})$.

Now we proceed to prove the general case. With the assumptions that the left and right singular vectors are only approximate, the proof become rather unwieldy, and the bounds obtained are less transparent. Therefore, in the following we will assume that the left and right singular vectors are computed exactly. We first need several technical lemmas.

LEMMA 3.3. *If $s \geq 0$, $t \geq 0$ satisfy $s^2 + t^2 \leq 2\epsilon^2 \leq 5 - \sqrt{17}$, then*

$$
\frac{st(1+st)}{(1-s^2)(1-t^2)} \leq \frac{\frac{1}{2}(s^2+t^2)(1+\frac{1}{2}(s^2+t^2))}{(1-\frac{1}{2}(s^2+t^2))^2} \leq \frac{\epsilon^2(1+\epsilon^2)}{(1-\epsilon^2)^2}.
$$

Proof. It is easy to see that the condition $s^2 + t^2 \leq 2\epsilon^2 \leq 5 - \sqrt{17}$ implies that

$$
\frac{s^2+t^2}{2}\left(1 - \frac{s^2+t^2}{2}\right) \leq 2(1-s^2-t^2), \quad \text{and} \quad (s+t)^2 \leq 2(s^2+t^2) < 2.
$$

It follows that

$$
\begin{aligned}
st(1+st)(s+t)^2 &\leq st(1+st)\left(1 + \frac{(s+t)^2}{2}\right) \\
&\leq \frac{s^2+t^2}{2}\left(1 + \frac{s^2+t^2}{2}\right)\left(1 + st + \frac{s^2+t^2}{2}\right) \\
&\leq 2(1-s^2-t^2)\left(1 + 2st + \frac{(s-t)^2}{2}\right) \\
&\leq 2(1-s^2)(1-t^2)\left(1 + 2st + \frac{(s-t)^2}{2}\right)
\end{aligned}
$$

Multiplying $(s-t)^2/4$ on the two sides of the inequality yields that

$$
\begin{aligned}
st(1+st)\left(\frac{s^2-t^2}{2}\right)^2 &\leq (1-s^2)(1-t^2)\left((1+2st)\frac{(s-t)^2}{2} + \frac{(s-t)^4}{4}\right) \\
&= (1-s^2)(1-t^2)\left(\frac{s^2+t^2}{2}\left(1 + \frac{s^2+t^2}{2}\right) - st(1+st)\right).
\end{aligned}
$$

Therefore we obtain that

$$st(1+st)\left(1-\frac{s^2+t^2}{2}\right)^2 = st(1+st)\left((1-s^2)(1-t^2)+\left(\frac{s^2-t^2}{2}\right)^2\right)$$
$$\leq (1-s^2)(1-t^2)\left(\frac{s^2+t^2}{2}\right)\left(1+\frac{s^2+t^2}{2}\right),$$

completing the proof. □

LEMMA 3.4. *Denote* $\hat{d} = u_+^T A_{11} v_+/(\|u_+\|\cdot\|v_+\|)^2$, *and* $\sigma = \sigma_1(A)$. *If* $\|u_-\|^2 + \|v_-\|^2 \leq 2\epsilon^2 \leq 5-\sqrt{17}$, *Then*

$$\left|\frac{\sigma-\hat{d}}{\sigma}\right| \leq c_1\epsilon^2, \quad c_1 = \frac{1+\epsilon^2}{(1-\epsilon^2)^2},$$

and if $\epsilon^2 < 1/3$,

$$\left|\frac{\sigma-\hat{d}}{\hat{d}}\right| \leq c_2\epsilon^2, \quad c_2 = \frac{1+\epsilon^2}{1-3\epsilon^2}.$$

Proof. Similar as in the proof of Theorem 3.1, we have

$$\hat{d} = \frac{u_-^T A_{22} v_- + \sigma(1-\|u_-\|^2-\|v_-\|^2)}{\|u_+\|^2\|v_+\|^2}.$$

It is easy to verify that

$$\|u_+\|^2\|v_+\|^2 = 1-\|u_-\|^2-\|v_-\|+\|u_-\|^2\|v_-\|^2.$$

Hence,

$$\hat{d} = \sigma + \frac{u_-^T A_{22} v_- - \sigma(\|u_-\|^2\|v_-\|^2)}{\|u_+\|^2\|v_+\|^2}.$$

It follows from Lemma 3.3 that

$$|\hat{d}-\sigma| \leq \sigma\frac{\|u_-\|\|v_-\|+\|u_-\|^2\|v_-\|^2}{\|u_+\|^2\|v_+\|^2} \leq \sigma\frac{\epsilon^2+\epsilon^4}{(1-\epsilon^2)^2} = c_1\epsilon^2\sigma.$$

The second inequality directly follows from the first one. □

Now we prove a key lemma. Notice that if $\{x, d, y\}$ is the *exact* largest singular triplet, $\sigma_i(A - xdy^T) = \sigma_{i+1}(A)$, for $i = 1, \cdots, \min\{m, n\} - 1$, and $\sigma_i(A - xdy^T) = 0$ for $i \geq \min\{m, n\}$, i.e., the 2nd largest singular value of A becomes the largest singular value of $A - xdy^T$, the 3rd largest singular value of A becomes the 2nd largest singular value of $A - xdy^T$, and so on. It is easy to see that for any distinct indexes $i_1, \ldots, i_k$,

$$\sum_{j=1}^{k} \sigma_{i_j}^2(A - xdy^T) \leq \sum_{j=1}^{k} \sigma_{i_j+1}^2(A),$$

The following result shows what happens if $\{x, d, y\}$ is only an approximate largest singular triplet.

LEMMA 3.5. *Using the notation of Step 2.1 of Algorithm SLRA, and assume that* $\|u_-\|^2 + \|v_-\|^2 \leq 2\epsilon^2$ *with* $\epsilon^2 < 1/3$. *Then for any distinct indexes* $i_1, \ldots, i_k$,

$$\sum_{j=1}^{k} \sigma_{i_j}^2(A - xdy^T) \leq \sum_{j=1}^{k} \sigma_{i_j+1}^2(A) + \sigma_1(A)\sigma_2(A)\epsilon + c\sigma_1^2(A)\epsilon^2,$$

where c depends on k,

$$c = \begin{cases} 2(1+c_1\epsilon^2)^2(c_3+\sqrt{2}c_2\epsilon) + c_1\epsilon, & k = 1 \\ 4(1+c_1\epsilon^2)^2(c_3+\sqrt{2}c_2\epsilon) + c_1\epsilon, & k > 1 \end{cases}$$

with $c_3 = 2 + \max\{c_2(4+c_2)\epsilon^2, 1/(1+c_1\epsilon^2)\} \leq 3$, *where* c_1 *and* c_2 *are defined in Lemma 3.4.*

Proof. Let the SVD of A be $A = U\Sigma V^T$. To simplify the notation, we assume that U and V have the first column u and v, respectively, and Σ has σ on its $(1,1)$ position. Partition

$$P_1U = \left[\begin{bmatrix} u_+ \\ u_- \end{bmatrix}, U_2\right], \quad P_2V = \left[\begin{bmatrix} v_+ \\ v_- \end{bmatrix}, V_2\right], \quad \Sigma = \begin{bmatrix} \sigma & 0 \\ 0 & \Sigma_2 \end{bmatrix}.$$

Denoting $B = U^T(A - xdy^T)V$, we can write

$$B = \begin{bmatrix} 0 & 0 \\ 0 & \Sigma_2 \end{bmatrix} + \hat{d}[e_1, w_1]\begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix}[e_1, w_1]^T = \Sigma - \hat{d}(e_1 - w_1)(e_1 - w_2)^T,$$

where $\hat{d} = d/(\|u_+\| \cdot \|v_+\|)$, $h = (\sigma - \hat{d})/\hat{d}$,

$$w_1 = \begin{bmatrix} w_{11} \\ w_{21} \end{bmatrix} = (P_1U)^T \begin{bmatrix} 0 \\ u_- \end{bmatrix} = \begin{bmatrix} \|u_-\|^2 \\ U_2^T \begin{bmatrix} 0 \\ u_- \end{bmatrix} \end{bmatrix},$$

$$w_2 = \begin{bmatrix} w_{12} \\ w_{22} \end{bmatrix} = (P_2V)^T \begin{bmatrix} 0 \\ v_- \end{bmatrix} = \begin{bmatrix} \|v_-\|^2 \\ V_2^T \begin{bmatrix} 0 \\ v_- \end{bmatrix} \end{bmatrix}.$$

It can be verified that

$$w_{11} = \|u_-\|^2 = \|w_1\|^2, \quad w_{12} = \|v_-\|^2 = \|w_2\|^2.$$

Notice that B is a rank-2 modification of $\operatorname{diag}(0, \Sigma_2)$. We now show that B^TB is a rank-3 modification of $\operatorname{diag}(0, \Sigma_2^2)$. To this end, let

$$\tilde{w}_1 = \begin{bmatrix} 0 \\ \Sigma_2 w_{21} \end{bmatrix}, \quad \Delta_2 = \begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 & w_{11} \\ w_{11} & w_{11} \end{bmatrix}\begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix}^2.$$

Then it can be verified that

$$B^TB = \operatorname{diag}(0, \Sigma_2^2) + \hat{d}[e_1, w_2, \tilde{w}_1]\Delta_3[e_1, w_2, \tilde{w}_1]^T \equiv \operatorname{diag}(0, \Sigma_2^2) + \hat{d}\Delta,$$

where

$$\Delta_3 = \begin{bmatrix} \hat{d}(h + w_{11})^2 & \hat{d}h(1 - w_{11}) & 1 \\ \hat{d}h(1 - w_{11}) & \hat{d}(1 - w_{11}) & -1 \\ 1 & -1 & 0 \end{bmatrix}.$$

Therefore, it follows from [11, Page 202] that for distinct indexes $i_1, \ldots, i_k$,

$$\sum_{j=1}^{k} \lambda_{i_j}(B^T B) \leq \sum_{j=1}^{k} \lambda_{i_j}(\operatorname{diag}(0, \Sigma_2^2)) + \hat{d} \sum_{j=1}^{k} \lambda_j(\Delta),$$

i.e.,

$$\sum_{j=1}^{k} \sigma_{i_j}^2 (A - xdy^T) \leq \sum_{j=1}^{k} \sigma_{i_j+1}^2(A) + \hat{d} \sum_{j=1}^{k} \lambda_j(\Delta).$$

(We have used $\lambda_j(\cdot)$ to denote the j-th *largest* eigenvalue of a symmetric matrix.) Since $\operatorname{rank}(\Delta) \leq 3$, we have $\lambda_j(\Delta) = 0$ for $j > 3$. We now show that

$$\lambda_1(\Delta) > 0, \quad \lambda_2(\Delta) > 0, \quad \lambda_3(\Delta) \leq 0.$$

First, $\operatorname{rank}([e_1, w_2, \tilde{w}_1]) \geq 2$ because e_1 is orthogonal to $\tilde{w}_1$. Without loss of generality, we assume that $\operatorname{rank}([e_1, w_2, \tilde{w}_1]) = 3$. (The case when $\operatorname{rank}([e_1, w_2, \tilde{w}_1]) = 2$ is easier to handle.) Thus by Inertia Theorem, the number of positive eigenvalues of Δ is equal to the number of positive eigenvalues of Δ_3.

Secondly, at least one principal minor of Δ_3 is negative. It implies that Δ_3 has at least one negative eigenvalue. Also Δ_3 is not negative definite since it has a positive diagonal element.

Finally, it can be shown that $\det(\Delta_3) = -\hat{d}(1+h)^2 < 0$. Therefore, Δ_3 has exactly two positive eigenvalues, and so does Δ. Hence we have

$$\sum_{j=1}^{k} \sigma_{i_j}^2 (A - xdy^T) \leq \sum_{j=1}^{k} \sigma_{i_j+1}^2(A) + \hat{d} \sum_{j=1}^{\min\{k,2\}} \lambda_j(\Delta).$$

Now to bound $\lambda_1(\Delta)$ and $\lambda_2(\Delta)$, we write

$$\Delta = [e_1, \tilde{w}_1] \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} [e_1, \tilde{w}_1]^T +$$

$$+ [e_1, w_2, \tilde{w}_1] \begin{bmatrix} d\delta_2 & [0, -1]^T \\ [0, -1] & 0 \end{bmatrix} [e_1, w_2, \tilde{w}_1]^T \equiv H + \tilde{\Delta}.$$

It is easy to see that $\lambda(H) = \{\|\tilde{w}_1\|, 0, \ldots, 0, -\|\tilde{w}_1\|\}$. To estimate $\|\tilde{\Delta}\|$, we write $\hat{w}_2 = w_2/\|w_2\|$ and $\hat{w}_1 = \tilde{w}_2/\|\tilde{w}_2\|$. Obviously,

$$\|[e_1, \hat{w}_2, \hat{w}_1]\| \leq \sqrt{2}, \quad \|\tilde{\Delta}\| \leq 2\|\hat{\Delta}\|,$$

where

$$\hat{\Delta} = \begin{bmatrix} \hat{d}(h + w_{11})^2 & \hat{d}h(1 - w_{11})\|w_2\| & 0 \\ \hat{d}h(1 - w_{11})\|w_2\| & \hat{d}(1 - w_{11})\|w_2\|^2 & -\|w_2\| \cdot \|\tilde{w}_1\| \\ 0 & -\|w_2\| \cdot \|\tilde{w}_1\| & 0 \end{bmatrix}.$$

Now by Lemma 5.2 of [12] and Lemma 3.5,

$$\begin{aligned}
\|\hat{\Delta}\| &\le \max\left\{|\hat{d}|(h+w_{11})^2, \left\|\begin{bmatrix} \hat{d}(1-w_{11})\|w_2\|^2 & -\|w_2\|\,\|\tilde{w}_1\| \\ -\|w_2\|\,\|\tilde{w}_1\| & 0 \end{bmatrix}\right\|\right\} \\
&\quad + |\hat{d}h(1-w_{11})|\|w_2\| \\
&\le \max\left\{|\hat{d}|(h+w_{11})^2, |\hat{d}|(1-w_{11})\|w_2\|^2 + \|w_2\|\|\tilde{w}_1\|\right\} + |\hat{d}h(1-w_{11})|\|w_2\| \\
&\le \sigma_1(A)(1+c_1\epsilon^2)\big(\max\{2+c_2(4+c_2)\epsilon^2, 2+1/(1+c_1\epsilon^2)\} + \sqrt{2}c_2\epsilon\big)\epsilon^2 \\
&= \sigma_1(A)(1+c_1\epsilon^2)(c_3+\sqrt{2}c_2\epsilon)\epsilon^2 \equiv \hat{c}\sigma_1(A)\epsilon^2.
\end{aligned}$$

Therefore,

$$\lambda_1(\Delta) \le \|\tilde{w}_1\| + 2\hat{c}\sigma_1(A)\epsilon^2 \le \sigma_2(A)\epsilon + 2\hat{c}\sigma_1(A)\epsilon^2, \quad \lambda_2(\Delta) \le 2\hat{c}\sigma_1(A)\epsilon^2.$$

For $k = 1$, we have

$$\begin{aligned}
\sigma_i^2(A - xdy^T) &\le \sigma_{i+1}^2(A) + \sigma_1(A)(1+c_1\epsilon^2)(\sigma_2(A) + 2\hat{c}\sigma_1(A)\epsilon)\epsilon \\
&= \sigma_{i+1}^2(A) + \sigma_1(A)\left(\sigma_2(A) + c_1\sigma_2(A)\epsilon^2 + 2\hat{c}\sigma_1(A)(1+c_1\epsilon^2)\epsilon\right)\epsilon \\
&\le \sigma_{i+1}^2(A) + \sigma_1(A)\sigma_2(A)\epsilon + c\sigma_1(A)^2\epsilon^2,
\end{aligned}$$

and for $k > 1$, we have

$$\begin{aligned}
\sum_{j=1}^{k}\sigma_{i_j}^2(A - xdy^T) &\le \sigma_{i+1}^2(A) + \sigma_1(A)(1+c_1\epsilon^2)\big(\sigma_2(A) + 4\hat{c}\sigma_1(A)\epsilon\big)\epsilon \\
&\le \sum_{j=1}^{k}\sigma_{i_j+1}^2(A) + \sigma_1(A)\sigma_2(A)\epsilon + \sigma_1(A)^2\big(c_1\epsilon + 4\hat{c}(1+c_1\epsilon^2)\big)\epsilon^2,
\end{aligned}$$

completing the proof. □

We still need one more result before we can prove our main theorem.

LEMMA 3.6. *Let $\{u, \sigma = \sigma_1(A), v\}$ be the largest singular triplet of A. Denote $E = u\sigma v^T - xdy^T$. If $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$, assuming $\epsilon^2 < 1/3$, then*

$$\|E\|_F \le \sigma_1(A)\left(\sqrt{2} + \frac{1+\epsilon^2}{1-\epsilon^2}\epsilon\right)\epsilon$$

and

$$\sigma_j(A - xdy^T) \le \sigma_{j+1}(A) + c_4\sigma_1(A)\epsilon,$$

where $c_4 = \sqrt{2} + \epsilon(1+\epsilon^2)/(1-\epsilon^2)$.

Proof. Let $\hat{h} = (\sigma - \hat{d})/\sigma$, where $\hat{d}$ is defined in Lemma 3.4. Then $\hat{d} = \sigma(1-\hat{h})$, and

$$P_1EP_2^T = \sigma\begin{bmatrix} \hat{h}\|v_+\|u_+ & \|v_-\|u_+ \\ \|v_+\|u_- & \|v_-\|u_- \end{bmatrix}\begin{bmatrix} v_+/\|v_+\| & 0 \\ 0 & v_-/\|v_-\| \end{bmatrix}^T.$$

By Lemma 3.4, $|\hat{h}| < 1$. Hence,

$$\|E\|_F^2 = \sigma^2((\hat{h}^2\|u_+\|^2 + \|u_-\|^2)\|v_+\|^2 + \|v_-\|^2)$$

$$
\begin{aligned}
&= \sigma^2 \left(\hat{h}^2 + (1-\hat{h}^2)(\|u_-\|^2 + \|v_-\|^2 - \|u_-\|^2\|v_-\|^2)\right) \\
&\le \sigma^2 \left(\hat{h}^2 + (1-\hat{h}^2)\cdot 2\epsilon^2\right) \\
&\le \sigma^2 \left((c_1^2\epsilon^4(1-2\epsilon^2) + 2\epsilon^2\right) \\
&\le \sigma^2 \left(2 + \left(\frac{\epsilon(1+\epsilon^2)}{(1-\epsilon^2)}\right)^2\right)\epsilon^2 \\
&\le \sigma^2 \left(\sqrt{2} + \frac{1+\epsilon^2}{1-\epsilon^2}\epsilon\right)^2 \epsilon^2 = (\sigma c_4 \epsilon)^2.
\end{aligned}
$$

By the well-known perturbation theorem about singular values, we have

$$
\begin{aligned}
\sigma_j(A - xdy^T) &= \sigma_j(A - u\sigma v^T + E) \\
&\le \sigma_j(A - u\sigma v^T) + \|E\| \\
&= \sigma_{j+1}(A) + \|E\| \\
&\le \sigma_{j+1}(A) + c_4\sigma_1(A)\epsilon,
\end{aligned}
$$

completing the proof. □

REMARK. It can be shown that If $\|u_-\| \le \epsilon$ and $\|v_-\| \le \epsilon$, then

$$
\sigma_j(A - xdy^T) \le \sigma_{j+1}(A) + \sigma_1(A)\left(1 + \frac{2\epsilon}{\sqrt{1-\epsilon^2}}\right)\epsilon.
$$

THEOREM 3.7. *Use the notation in Step 2.1 of Algorithm SLRA, and assume that* $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$ *with* $\epsilon^2 < 1/3$. *Then*

$$
\|A - X_k D_k Y_k^T\|_F \le \sqrt{1 + b_k\epsilon}\,\|A - U_k\Sigma_k V_k^T\|_F,
$$

where

$$
b_k = \frac{\sum_{j=1}^{k}\sigma_j(A)\sigma_{j+1}(A)}{\sum_{j=k+1}^{n}\sigma_j^2(A)} + O(\epsilon).
$$

Proof. Let $A_k = A - X_k D_k Y_k^T$ with $A_0 = A$, and

$$
X_k = [x_1, \dots, x_k], \quad D_k = \operatorname{diag}(d_1, \dots, d_k), \quad Y_k = [y_1, \dots, y_k].
$$

Then $A_k = A_{k-1} - x_k d_k y_k^T$, where x_k and y_k are the sparsified version of the largest left and right singular vectors $u^{(k-1)}$ and $v^{(k-1)}$ of A_{k-1}, respectively. Specifically, we choose permutation matrices $P_1^{(k-1)}$ and $P_2^{(k-1)}$ such that

$$
P_1^{(k-1)}u^{(k-1)} = \begin{bmatrix} u_+^{(k-1)} \\ u_-^{(k-1)} \end{bmatrix}, \quad P_2^{(k-1)}v^{(k-1)} = \begin{bmatrix} v_+^{(k-1)} \\ v_-^{(k-1)} \end{bmatrix}
$$

with $\|u_-^{(k-1)}\|^2 + \|v_-^{(k-1)}\|^2 \le 2\epsilon^2$. Then

$$
x_k = (P_1^{(k-1)})^T = \begin{bmatrix} u_+^{(k-1)} \\ 0 \end{bmatrix} / \|u_+^{(k-1)}\|, \quad y_k = (P_2^{(k-1)})^T = \begin{bmatrix} v_+^{(k-1)} \\ 0 \end{bmatrix} / \|v_+^{(k-1)}\|.
$$

By Lemma 3.5 with $A_k = A_{k-1} - x_k d_k y_k^T$, we have

$$\begin{aligned}\|A - X_k D_k Y_k^T\|_F^2 &= \sum_{j=1}^{n} \sigma_j^2(A_k) \\ &\le \sum_{j=2}^{n} \sigma_j^2(A_{k-1}) + \sigma_1(A_{k-1})\sigma_2(A_{k-1})\epsilon + c\sigma_1^2(A_{k-1})\epsilon^2 \\ &\le \sum_{j=k+1}^{n} \sigma_j^2(A) + \sum_{j=0}^{k-1}\sigma_1(A_j)\sigma_2(A_j)\epsilon + c\sum_{j=0}^{k-1}\sigma_1^2(A_j)\epsilon^2.\end{aligned}$$

On the other hand, by Lemma 3.6, we have

$$\begin{aligned}\sigma_1(A_j) &\le \sigma_2(A_{j-1}) + c_4\sigma_1(A_{j-1})\epsilon \\ &\le \sigma_3(A_{j-2}) + c_4(\sigma_1(A_{j-2}) + \sigma_1(A_{j-1}))\epsilon \\ &\le \cdots \\ &\le \sigma_{j+1}(A) + c_4 \textstyle\sum_{i=0}^{j-1} \sigma_1(A_i)\epsilon.\end{aligned}$$

Let $s_j = \sum_{i=0}^{j-1} \sigma_1(A_i)$. Then,

$$\begin{aligned}s_j &= \sigma_1(A_{j-1}) + s_{j-1} \le \sigma_j(A) + (1 + c_4\epsilon)s_{j-1} \\ &\le \sigma_j(A) + (1 + c_4\epsilon)(\sigma_{j-1}(A) + (1 + c_4\epsilon)s_{j-2}) \\ &\le \cdots \\ &\le \textstyle\sum_{i=1}^{j}(1 + c_4\epsilon)^{j-i}\sigma_i(A)\end{aligned}$$

which gives

$$\sigma_1(A_j) \le \sigma_{j+1}(A) + c_4\sum_{i=1}^{j}(1 + c_4\epsilon)^{j-i}\sigma_i(A)\epsilon \equiv \sigma_{j+1}(A) + \phi_j\epsilon,$$

where $\phi_j = c_4\sum_{i=1}^{j}(1 + c_4\epsilon)^{j-i}\sigma_i(A)$. Similarly, we have

$$\sigma_2(A_j) \le \sigma_{j+2}(A) + \phi_j\epsilon.$$

Therefore,

$$\sum_{j=0}^{k-1}\sigma_1(A_j)\sigma_2(A_j) \le \sum_{j=1}^{k}\left(\sigma_j(A)\sigma_{j+1}(A) + (\sigma_j(A) + \sigma_{j+1}(A) + \phi_{j-1}\epsilon)\phi_{j-1}\epsilon\right),$$

and

$$\sum_{j=0}^{k-1}\sigma_1^2(A_j) \le \sum_{j=1}^{k}\left(\sigma_j^2(A) + 2\sigma_j(A)\phi_{j-1}\epsilon + \phi_{j-1}^2\epsilon^2\right).$$

It follows that

$$\begin{aligned}\|A - X_k D_k Y_k^T\|_F &\le \sum_{j=k+1}^{n} \sigma_j^2(A) + \sum_{j=1}^{k}\sigma_j(A)\sigma_{j+1}(A)\epsilon + \tilde{b}_k\epsilon^2 \\ &= (1 + b_k\epsilon)\|A - U_k\Sigma_k V_k^T\|_F^2,\end{aligned}$$

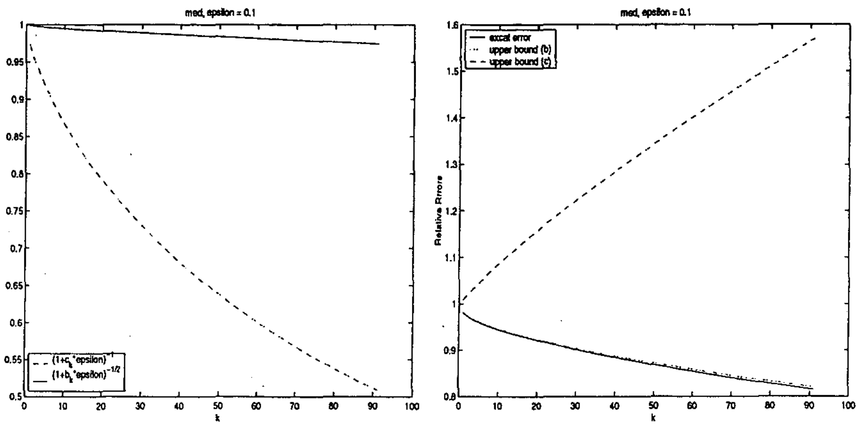

FIG. 3.1. $(1 + c_k * \epsilon)^{-1}$ *and* $(1 + b_k * \epsilon)^{-1/2}$ *(left) and the relative errors (right).*

where

$$\tilde{b}_k = \sum_{j=1}^{k} \left\{ c\sigma_j^2(A) + ((1+2c\epsilon)\sigma_j(A) + \sigma_{j+1}(A) + (1+c\epsilon)\phi_{j-1}\epsilon)\, \phi_{j-1} \right\},$$

completing the proof. □

REMARK. Using the well-known Wielandt-Hofmann Theorem and Lemma 3.6, one can prove that

$$\left(\sum_{i=k}^{n} \sigma_j^2(A - xdy^T) \right)^{1/2} \le \left(\sum_{i=k+1}^{n} \sigma_j^2(A) \right)^{1/2} + \sigma_1(A)\Big(\sqrt{2} + \frac{2\epsilon^2}{\sqrt{1-\epsilon^2}}\Big).$$

Therefore it is not hard to prove that

$$\|A - X_k D_k Y_k^T\|_F \le (1 + c_k\epsilon)\|A - U_k \Sigma_k V_k^T\|_F,$$

with $c_k = \sqrt{2}\sum_{i=1}^{k} \sigma_i(A)/(\sum_{i=k+1}^{n} \sigma_j^2(A))^{1/2} + O(\epsilon)$. However, the coefficient c_k seems to give a less tight bound. Figure 3.1 plots the curves of $(1 + c_k\epsilon)^{-1}$ and $(1 + b_k\epsilon)^{-1/2}$ with the $O(\epsilon)$ terms omitted for the matrix med (cf Section 5) on the left and the relative error

$$\mathrm{err}_{\mathrm{best}}(k) = \frac{\|A - \mathrm{best}_k(A)\|_F}{\|A\|_F},$$

and the upper bounds

$$(1 + c_k\epsilon)\mathrm{err}_{\mathrm{best}}(k) \quad \text{and} \quad (1 + b_k\epsilon)^{1/2}\mathrm{err}_{\mathrm{best}}(k),$$

on the right. It is easy to see that the bound using b_k is much tighter.

4. Computational Variations. In this section, we first discuss several computational variations of Algorithms SLRA, in particular we will discuss two approaches for sparsifying vectors in the sparse rank-one approximation step of Algorithm SLRA. We first briefly discuss how to find approximate solutions to the largest singular triplet of a matrix.

Approximation to Largest Singular Vectors. As we mentioned in Section 2, the largest singular triplet $\{u, \sigma, v\}$ need not be computed to high accuracy because a sparsification process that follows will introduce errors by discarding certain nonzero elements of u and v. There are several approaches for approximating the largest singular triplets such as the power method and Lanczos bidiagonalization process [4, 8]. Using the power method, we suggest to perform several steps of power iteration as follows,

$$\begin{aligned} v &\leftarrow (A^T A)^\alpha v_0, \\ v &\leftarrow v/\|v\|_2, \\ u &\leftarrow Av/\|Av\|_2. \end{aligned}$$

where v_0 is an initial guess, for example, $v_0 = (1, \cdots, 1)^T$, α is a small integer, for example, $\alpha = 3$.

For Lanczos bidiagonalization, we run several iterations to generate a pair of orthogonal basis $\{u_1, \cdots, u_\beta\}$ and $\{v_1, \cdots, v_\beta\}$, and a lower bidiagonal matrix B_β satisfying

$$\begin{aligned} A[v_1, \cdots, v_\beta] &= [u_1, \cdots, u_\beta]B_\beta + b_\beta u_{\beta+1}, \\ A^T[u_1, \cdots, u_\beta] &= [v_1, \cdots, v_\beta]B_\beta^T. \end{aligned}$$

The largest singular vectors a and b of B_β will be used to obtain approximations u and v:

$$v = [v_1, \cdots, v_\beta]a, \quad u = [u_1, \cdots, u_\beta]b.$$

Sorting and Sparsification. This corresponds to how to partition the computed approximate singular vectors u and v for later sparsification process. By Theorems 3.1 and 3.2 the reconstruction error $\|A - xdy^T\|_F$ of the sparse rank-one approximation depends on the size of the discarded sections $\|u_-\|_2$ and $\|v_-\|_2$. Therefore it makes sense to sort vectors u and v in decreasing order by their absolute values so that the number of the discarded elements is largest under the constraints $\|u_-\|_2 \le \epsilon$ and $\|v_-\|_2 \le \epsilon$, or $\|u_-\|_2^2 + \|v_-\|_2^2 \le 2\epsilon^2$. In particular, we find permutations P_1 and P_2 such that $\tilde{u} \equiv P_1 u = \begin{bmatrix} u_+ \\ u_- \end{bmatrix}$, $\tilde{v} \equiv P_2 v = \begin{bmatrix} v_+ \\ v_- \end{bmatrix}$ and

$$|\tilde{u}_1| \ge |\tilde{u}_2| \ge \cdots \ge |\tilde{u}_m|, \quad |\tilde{v}_1| \ge |\tilde{v}_2| \ge \cdots \ge |\tilde{v}_n|.$$

Let k_u and k_v be the lengths of sections u_+ and v_+, respectively. Thus $u_+ = \tilde{u}(1:\mathrm{ku})$ and $v_+ = \tilde{v}(1:\mathrm{kv})$. We then choose

$$x = P_1^T \begin{bmatrix} \tilde{u}(1:\mathrm{ku}) \\ 0 \end{bmatrix} / \|\tilde{u}(1:\mathrm{ku})\|, \quad y = P_2^T \begin{bmatrix} \tilde{v}(1:\mathrm{kv}) \\ 0 \end{bmatrix} / \|\tilde{v}(1:\mathrm{kv})\|.$$

The integers k_u and k_v can be determined by the following two different schemes.

- SEPARATED. In this approach, we sort the elements of u and v *separately* and k_u and k_v are defined by

$$k_u = \min\left\{k \,\Big|\, \sum_{j=1}^{k} \tilde{u}_j^2 \ge 1 - \epsilon^2\right\}, \quad k_v = \min\left\{k \,\Big|\, \sum_{j=1}^{k} \tilde{v}_j^2 \ge 1 - \epsilon^2\right\}$$

  for a given tolerance ϵ.

- MIXED. Another approach is to set $w = [u^T, v^T]^T$ and find a permutation P such that $Pw = \tilde{w}$, $|\tilde{w}_1| \geq |\tilde{w}_2| \geq \cdots \geq |\tilde{w}_{m+n}|$. We determine k_w such that

$$k_w = \min\left\{k \,\Big|\, \sum_{j=1}^{k} \tilde{w}_j^2 \geq 2 - 2\epsilon^2\right\}.$$

Obviously, the order of the u-components of vector $\tilde{w}$ implies the permutation P_1. So does the order of the v-components for P_2. Therefore the main section $\tilde{w}(1 : \mathtt{kw})$ also determine $\tilde{u}(1 : \mathrm{ku})$ and $\tilde{v}(1 : \mathrm{kv})$, where k_u and k_v are, respectively, the numbers of u-components and v-components of $\tilde{w}(1 : \mathtt{kw})$.

REMARK. In general, our experiments show that the mixed scheme performs better than the separated scheme.

Choice of tolerance ϵ. At each iteration step of the Algorithm SLRA, the tolerance ϵ can be chosen to be constant or variable. We will use, for variable tolerance, at the k-th step

$$\epsilon_k = \frac{\|A_{k-1}\|_F}{\|A\|_F}\epsilon,$$

which depends on the approximation computed by previous iterations.

Choice of k. Notice that the norm of error A_k reads

$$\|A_k\|_F^2 = \|A - X_k D_k Y_k\|_F^2 = \|A\|_F^2 - \sum_{j=1}^{k} d_j^2.$$

In fact, we have

$$\|A_k\|_F^2 = \|A_{k-1}\|_F^2 - d_k^2.$$

It is quite convenient to use this recurrence as a stopping criterion of Algorithm SLRA:

$$\|A_k\|_F \leq \mathtt{tol}$$

for the given user-specified tolerance `tol`.

Self-correcting Mechanism. This is certainly an area that deserves further research, and in the following we can only touch the tip of the iceberg. When we use a rank-one matrix $u\sigma v^T$ that is constructed from the exact largest singular triplet $\{u, \sigma, v\}$ of A, the difference $A - u\sigma v^T$ will not have any components in the two one-dimensional subspaces spanned by u and v, respectively. Notice that $\|A - u\sigma v^T\|_F^2 = \|A\|_F^2 - \sigma^2$, and the amount of reduction in the F-norm is the largest possible by a rank-one modification. Now when we use an inaccurate rank-one approximation xdy^T, in general, it is true that $\hat{A} \equiv A - xdy^T$ will have some components left in the directions of u and v. Also $\|\hat{A}\|_F^2 = \|A\|_F^2 - d^2$, and the reduction in F-norm will be smaller. The question now is if we compute the rank-one approximation $\hat{x}\hat{d}\hat{y}^T$ for $\hat{A}$, will $\hat{x}\hat{d}\hat{y}^T$ pick up some of the components in u and v that are left by the previous rank-one approximation xdy^T? The answer seems to be yes even though we do not have a formal proof. This indicates that Algorithm SLRA has a self-correcting mechanism: errors made in early deflation steps can be corrected by later deflation steps. We now give an example that illustrate this phenomenon. Table 4 lists the first 10 diagonals $\{d_j\}$ and the singular values $\{\sigma_j\}$ of matrix A, respectively. In this example, those steps j for which $d_j > \sigma_j$ show the self-correcting process at work.

TABLE 4.1
Self-correction

j	d_j	σ_j
1	4.5595e+05	4.5808e+05
2	3.8998e+05	4.5762e+05
3	4.5482e+05	4.5761e+05
4	3.7309e+05	3.9093e+05
5	4.4721e+05	3.9050e+05
6	3.5648e+05	3.9049e+05
7	2.2148e+05	2.2090e+05
8	1.8609e+05	2.2046e+05
9	2.3341e+05	2.2044e+05
10	2.2075e+05	1.1472e+05

A combinatorial optimization problem. Now we reexamine the optimization problem (2.1) for $k = 1$. We can impose the following constraints: $\text{nnz}(x) = n_x, \text{nnz}(y) = n_y$, where $n_x \le m$ and $n_y \le n$ are fixed. Let $i_1, \ldots, i_{n_x}$ and $j_1, \ldots j_{n_y}$ be the indexes of the nonzero elements of x and y, respectively. Then it is easy to see that the optimization problem (2.1) is reduced to

$$\min_{\hat{x}\in\mathcal{R}^{n_x}, \hat{y}\in\mathcal{R}^{n_y}} \|A([i_1, \ldots, i_{n_x}], [j_1, \ldots, j_{n_y}]) - \hat{x}d\hat{y}^T\|_F,$$

where $\tilde{A} \equiv A([i_1, \ldots, i_{n_x}], [j_1, \ldots j_{n_y}])$ is the submatrix of A consists of the intersection of rows $i_1, \ldots, i_{n_x}$ and columns $j_1, \ldots j_{n_y}$. Therefore, by Theorem 1.1 we need to find the largest singular triplet of $\tilde{A}$. Hence, the optimization problem (2.1) for $k = 1$ is equivalent to the following problem: Find n_x rows and n_y columns of A such that the largest singular value of the resulted $\tilde{A}$ is maximized. This is a *combinatorial* optimization problem, and we do not know any good, i.e., polynomial-time, solution method for it. Step 2.1 of Algorithm SLRA does seem to provide an heuristic for its solution. Now we give an example to illustrate this point.

EXAMPLE. In this example, we take a matrix A from [1] with the change $A(4,3) = 1$ so that the largest left singular vector u has different elements to keep the decreasing order to be unique. Below is the matrix A.

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

The goal is to compare the computed sparse low-rank approximation with the *optimal* solution of the combinatorial optimization problem computed by exhaustive search.

We first compute the sparse approximation $X_k D_k Y_k^T$ for $k = 2$ using Algorithm SLRA with $\epsilon = 0.3$ and $\beta = 4$ for the bidiagonalization. Then we compute the best rank-one approximation $u_1 s_1 v_1^T$ to A with $\text{nnz}(x_1) = 5$ nonzeros of u_1 and $\text{nnz}(y_1) = 4$ nonzeros of v_1, and the best rank-one approximation $u_2 s_2 v_2^T$ to matrix $A - u_1 s_1 v_1^T$ with $\text{nnz}(x_2) = 3$ nonzeros of u_2 and $\text{nnz}(y_2) = 3$ nonzeros of v_2. Below we list the computed components of vectors x_i, y_i, u_i, and v_i. The two approximations give the same sparsity patterns.

x_1	x_2	u_1	u_2
0.4058	0.3245	0.4111	0.3118
0.6146	0	0.6362	0
0.4058	0.3245	0.4111	0.3118
0.3583	0	0.3587	0
0.4058	−0.8885	0.3587	−0.8975
0	0	0	0

y_1	y_2	v_1	v_2
0.4508	0.5423	0.4905	0.5066
0	−0.6170	0	−0.6322
0.3075	0	0.3346	0
0.7734	0	0.7318	0
0.3226	−0.5702	0.3346	−0.5863

5. Numerical Experiments. In this section, we discuss several numerical issues associated with Algorithm SLRA and illustrate its effectiveness and efficiency. We will also compare its performance with truncated SVD and the approach proposed in [10]. For our numerical experiments, we made a collection of test matrices which are listed below together with some statistics about the matrices: matrices 3, 4, 5 and 6 are term-document matrices from SMART information retrieval System, and the rest of the matrices are selected from Matrix Market [3, 7]. We do not claim that the collection is comprehensive that covers all possibilities.

	Matrix	m	n	nnz(A)	Density(%)	Accuracy	Rank-SVD
1	ash958	958	292	19196	0.68	8.84e-1	33
2	illc1033	1033	320	4732	1.43	1.65e-2	42
3	cisi	5081	1469	66241	0.89	8.12e-1	68
4	cacm	3510	3204	70339	0.63	7.70e-1	63
5	med	5504	1033	51096	0.90	8.16e-1	79
6	npl	4322	11429	224918	0.46	8.48e-1	41
7	watson4	467	468	2836	1.30	3.04e-2	94
8	orsirr2	886	886	5970	0.76	2.22e-1	248
9	e20r1000	4241	4241	131430	0.73	8.83e-1	159

Some explanation of the notation we used is in order here: m and n represent the row and column dimensions, respectively, of the the given matrix. $\mathrm{nnz}(A)$ gives the number of nonzeros of the matrix A. Density is computed as $\mathrm{nnz}(A)/(mn)$. The seventh column of the table lists the reconstruction errors of the approximations using 100 columns/rows of the corresponding matrix A if $\min(m,n) \le 500$ or 300 columns/rows otherwise, determined by Stewart's sparse pivoted QR approximations (SPQR) [10]. It should be pointed that the SPQR needs *a priori* to determine the integer k, the number of columns/rows of A which are used to construct the approximation $B_k = A_c M A_r^T$.[2] We also use the reconstruction errors of SPQR approximation

[2]Notice that if we need to compute a B_k such that $\|A - B_k\|_F \le \mathtt{tol}$ for a given user specified `tol`, there is no easy way to determine k *a priori* to satisfy this constraint.

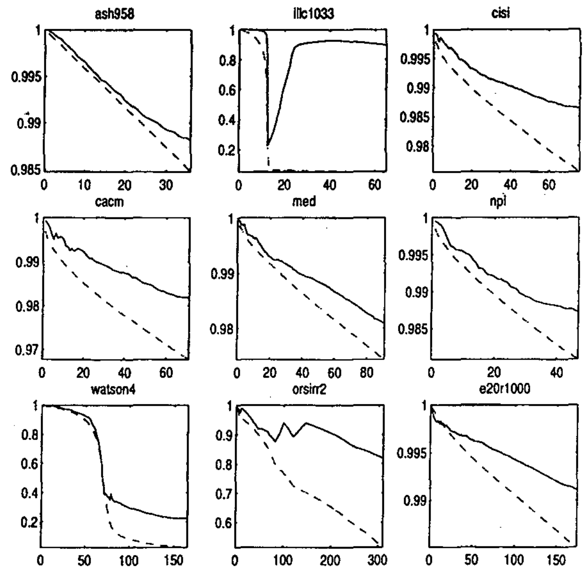

FIG. 5.1. *Plots for merits computed by SLRA with constant tolernce* $\epsilon = 0.1$ *and separated sorting scheme (solid lines) and the lower bounds* $(1 + b_k\epsilon)^{-1/2}$ *in Theorem 3.7 (dashed lines).*

as the accuracy for the numerical experiments. The last column is for the ranks of the best approximation obtained by truncated SVD (TSVD), i.e., $\text{best}_k(A)$, which achieve the same accuracy. For Algorithm SLRA, we run a few steps $\beta = 4$ or 6 of the Lanczos bidiagonalization to compute the approximate largest singular vectors, because bidiagonalization is more efficient than the power method.

TEST 1. We compare the low-rank approximations computed by Algorithm SLRA with constant tolerance $\epsilon = 0.1$ and those computed by truncated SVD. The dimension for Lanczos bidiagonalization for computing the approximate largest singular vectors is $\beta = 4$. We use the merit defined by

$$\text{merit}(k) = \frac{\|A - \text{best}_k(A)\|_F}{\|A - X_k D_k Y_k^T\|_F}$$

to measure the effectiveness of Algorithm SLRA. It is easy to see that $0 \le \text{merit}(k) \le 1$. The larger the merit is, the more effective SLRA is. Below we list the merits of SLRA with constant tolerance $\epsilon = 0.1$ and *separated sorting scheme*. The rank k is chosen to be $5 \sim 20\%$ of the size $l = \min(m, n)$ of matrix A.

Matrix	k= 5%	10%	15%	20%	Average
ash958	0.9946	0.9896	0.9876	0.9845	0.9908
illc1033	0.3622	0.9160	0.9226	0.8984	0.8595
cisi	0.9866	0.9771	0.9690	0.9612	0.9778
cacm	0.9774	0.9625	0.9427	0.9221	0.9596
med	0.9882	0.9790	0.9699	0.9617	0.9790
watson4	0.9784	0.9374	0.4833	0.3166	0.7809
orsirr2	0.9217	0.8942	0.9136	0.9206	0.9274

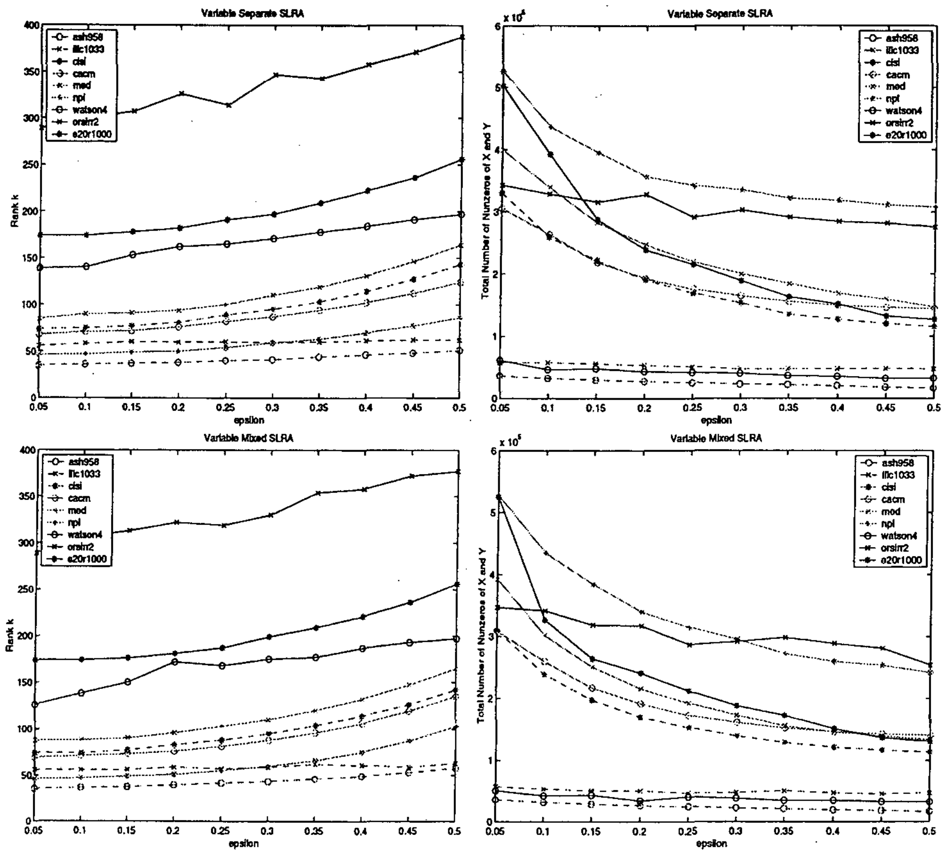

FIG. 5.2. *Plots for ranks (left) and numbers of nonzeros of X_k and Y_k (right) vs starting epsilon for the variable tolerance, separated (top) and mixed (bottom) sorting approaches.*

Figure 5.1 plots the merit quantities computed by SLRA with for all the nine matrices until $B_k = X_k D_k Y_k^T$ achieves the accuracy listed in the test matrix table. These examples show that SLRA has very high merits for most of the test matrices, specially for the term-document matrices.

TEST 2. In general, the *mixed sorting scheme* gives a smaller number of nonzeros for the sparse factors X_k and Y_k than the *separated sorting scheme* if we use the same tolerance sequence and there are no appreciable changes in rank k. Figure 5.2 plots the ranks (left) and the total number of nonzeros of X_k and Y_k (right) computed by SLRA with separated (top) and mixed (bottom) sorting schemes. We use variable tolerance with different starting values ϵ = `0.05:0.05:0.5`.

TEST 3. In this test we compare TSVD, SPQR and SLRA (using variable tolerance and mixed sorting scheme) with $\epsilon = 0.1, 0.5$ and $\beta = 6$ for Lanczos bidiagonalization. The approximations computed by the three approaches have the same reconstruction errors for each test matrix. The numerical results show that SLRA can give very small rank k and acceptable sparse structures of the factors X_k and Y_k. Note that the last matrix `bcsstk02` from Matrix Market [7] is a dense matrix. SLRA can also give sparse approximations to dense matrices. On the other hand, SPQR are very effective for sparse matrices that are close to rank-deficient, for example, watson4 and orsirr2.

atrix		ϵ	rank	total nnz	flops
ash958	TSVD		33	42339	474566713
	SLRA	0.10	34	28196	20696012
		0.50	50	13451	40396443
	SPQR		100	11036	5668695
illc1033	TSVD		42	58590	1024293779
	SLRA	0.10	55	50343	58649189
		0.50	57	43276	62056281
	SPQR		100	11808	8909800
cisi	TSVD		68	449412	6925863163
	SLRA	0.10	72	217401	523406959
		0.50	136	102124	1587083864
	SPQR		300	129720	568382817
cacm	TSVD		63	426951	5390479001
	SLRA	0.10	67	216982	478032905
		0.50	129	121950	1488980629
	SPQR		300	133784	463854304
med	TSVD		79	522664	9598485598
	SLRA	0.10	84	278456	658852943
		0.50	157	118615	2018489359
	SPQR		300	120444	469695010
npl	TSVD		41	647472	6208537332
	SLRA	0.10	44	384118	616205165
		0.50	98	217025	2249873520
	SPQR		300	227567	588513394
watson4	TSVD		94	96726	1305367466
	SLRA	0.10	136	39832	213502126
		0.50	175	24040	341078084
	SPQR		100	11352	7792938
orsirr2	TSVD		248	500960	39404030518
	SLRA	0.10	274	259894	1615591558
		0.50	347	210951	2524914675
	SPQR		300	94127	110865410
e20r1000	TSVD		159	1373919	55819003349
	SLRA	0.10	168	245692	3246302118
		0.50	247	83545	6533202985
	SPQR		300	102695	315702973
bcsstk02	TSVD		40	6880	73220467
	SLRA	0.10	42	4350	7555338
		0.50	57	3846	11306462
	RRQR		44	7744	4266443

6. Concluding Remarks. We have presented algorithms for computing matrix low-rank approximations with sparse factors. We also gave a detailed error analysis that compared the reconstruction errors for the low-rank approximations that are computed by SVD and the low-rank approximations that are computed by by our sparse algorithms. Our algorithms are flexible in the sense that users can balance the tradeoff of high sparsity level of the computed low-rank factors and the reduced reconstruction error. Several issues deserve further investigation: 1) we need to develop better ways for computing sparse rank-one approximations. As we mentioned, for example, if we fix the number of nonzero elements in x and y, say p and q, then $\min \|A - xdy^T\|_F$ is equivalent to the following *combinatorial optimization* problem: find p rows and q columns of A such that the largest singular value of their intersection is maximized. We are in the process of finding heuristics for solving this problem and investing the their relations to the sorting approach of Algorithm SLRA. 2) Once a low-rank approximation A_k is computed, a certain refinement procedure needs to be developed to reduce its reconstruction error and/or the number of nonzeros of its sparse factors. 3) It will also be of great interest to consider reconstruction errors in norms other than $\|\cdot\|_F$.

REFERENCES

[1] M.W. Berry, Z. Drmač and E.R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41:335-362,1999.

[2] M.W. Berry, S.T. Dumais and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573-595, 1995.

[3] Cornell SMART System, `ftp://ftp.cs.cornell.edu/pub/smart`.

[4] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland, 2nd edition, 1989.

[5] T. Hofmann. Probabilistic Latent Semantic Indexing. Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR'99), 1999.

[6] T. Kolda and D. O'Leary. A semidiscrete matrix decomposition for latent semantic indexing in information retrieval. ACM Trans. Information Systems, 16:322-346, 1998.

[7] Matrix Market. `http://math.nist.gov/MatrixMarket/`.

[8] B.N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM Press, Philadelphia, 1998.

[9] H. Simon and H. Zha. Low-rank matrix approximation using the Lanczos bidiagonalization process. CSE Tech. Report CSE-97-008, 1997. (Also LBNL Tech. Report LBNL-40767-UC-405.)

[10] G.W. Stewart. Four algorithms for the efficient computation of truncated pivoted QR approximation to a sparse matrix. CS report, TR-98-12, University of Maryland, 1998.

[11] G.W. Stewart and J. G. Sun. Matrix Perturbation Theory. Academic Press, 1990.

[12] H. Zha and Z. Zhang. Matrices with low-rank-plus-shift structure: partial SVD and latent semantic indexing. To appear in *SIAM Journal on Matrix Analysis and Applications*, 1999.

Ernest Orlando Lawrence Berkeley National Laboratory
One Cyclotron Road | Berkeley, California 94720

Prepared for the U.S. Department of Energy under Contract No. DE-AC03-76SF00098