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 factorized form with sparse factors 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 some application areas to illustrate the efficiency and accuracy of our algorithms.

Key words: low-rank matrix approximation, singular value decomposition, sparse factorization, perturbation analysis

AMS subject classifications. 15A18, 15A23, 65F15, 65F50

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 [5, 14, 17] 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 $\|\cdot\|_F$ [5, Theorem 2.5.3].

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

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

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

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

The minimum is achieved with $\operatorname{best}_k(A) \equiv U_k \operatorname{diag}(\sigma_1, \dots, \sigma_k)V_k^T$, where U_k and V_k are the matrices formed by the first k columns of U and V, respectively.

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, $\operatorname{best}_k(A)$ has the smallest reconstruction error in Frobenius norm among all the rank-k approximations of A. In certain applications, it is desirable to impose further constraints on the low-rank approximation B in addition to requiring that it be of low-rank. Consider the case where, for example, the matrix A is sparse; it is generally not true that

---

* Department of Mathematics, Zhejiang University, Yuquan Campus, 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), the Special Funds for Major State Basic Research Projects of China (project G19990328), and Foundation for University Key Teacher by the Ministry of Education, China. 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.

† 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.

$\text{best}_k(A) = U_k\Sigma_k V_k^T$ or even its associated factors U_k and V_k also will be sparse. Therefore, the storage requirement of $\text{best}_k(A)$ in the factorized form $\text{best}_k(A) = U_k\Sigma_k V_k^T$ can be even greater than that of the original matrix A. To overcome this difficulty, we seek to find low-rank approximations that simultaneously also possess some sparsity properties. One possibility will be to impose 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 difficult to achieve a reasonable reconstruction error (comparing with that obtained from $\text{best}_k(A)$, for example) using a sparse B. Inspired by the work reported in [7, 15], we consider the approach of writing B in a factorized form as $B = XDY^T$, and imposing sparsity requirements on the factors X and Y instead while keeping D in positive 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. One by-product of using the factorized form is that the low-rank constraint on B is trivially satisfied once B is in the factored form, i.e., $\text{rank}(B) \le k$ 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 latent class models for two-way contingency tables [4], probabilistic Latent Semantic Indexing [6] and nonnegative matrix factorization [8], 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 [12]. We also mention that there has been research on solving linear systems and linear least squares problems with sparse solution vectors [3, 10].

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 within 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 compare the low-rank approximations computed by our algorithms with those obtained by SVD and the approaches developed in [15]. In section 6, we summarize our contributions and point out future research directions.

Notation. We use $\sigma_k(A)$ to denote the kth singular value of a matrix A in nonincreasing order. We also replace $\sigma_k(A)$ by σ_k when the matrix in question is unambiguous. By $\|\cdot\|$ we denote the 2-norm of vectors or matrices.

2. Sparse low-rank approximations.

We first review some previous work on computing low-rank approximations with sparse factors. O'Leary and Peleg proposed a method for computing low-rank approximations for image processing [11]. In [7] Kolda and O'Leary called this *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 nonnegative diagonal. Furthermore, they require that the entries of X_k and Y_k belong to three-element 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 a reconstruction error comparable to that of $\text{best}_K(A)$, and therefore the low-rank property of B_k may not hold. Despite this the storage requirement of B_k in the factored form is usually much lower than that of A, and this is certainly the major strength of SDD as is demonstrated in the application in latent semantic indexing. In [15], Stewart proposes to construct low-rank approximations of a *sparse* matrix A by selecting a subset of its columns and rows, 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 error $\|A - A_c M A_r^T\|_F$ once the left and right factors A_c and A_r are chosen. The matrices A_c and A_r are determined by variations of QR algorithms with a certain pivoting strategy. In general, the matrix M will be dense. Due to the denseness of M, the storage requirement of B_k can become rather high as k increases, and also the low-rank approximation will not be sparse if A itself is not sparse. Numerical experiments showed that Stewart's approach is especially effective when A itself is close to high rank-deficiency. The approach we now propose builds on the strength of the above two approaches: we seek an approximation that is of low-rank and at the same time we also want to have greater control of the sparsity properties of the low-rank approximation. To this end, we consider the following general minimization problem.[1]

$$\begin{aligned}&\min \|A - X_k D_k Y_k^T\|_F \\ &\text{subject to } D_k \text{ positive diagonal, } X_k \in \mathcal{R}^{m\times k} \text{ and } Y_k \in \mathcal{R}^{n\times k} \text{ sparse.}\end{aligned} \tag{2.1}$$

The above optimization problem in its present form is not completely specified because the minimum depends on the sparsity constraints: the number of nonzero elements 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 sparsity 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 sparsity patterns for X_k and Y_k?

In this paper we will not discuss how to impose the sparsity constraints on the factors X_k and Y_k in general, but rather we will first start with an heuristic. In this section, we propose the framework of our sparse low-rank approximation (SLRA) approach based on the idea of deflation. 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 positive 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 $A_i = A_{i-1} - x_i d_i y_i^T$.
```

Algorithm SLRA consists of a sequence of k deflation steps [13] which allows us to build a low-rank approximation one rank at a time. This general approach is

[1] The diagonal elements of D can certainly be constructed to be positive as we will do in the sequel.

also adopted in [7], but the actual deflation step there 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 $D_k = \mathrm{diag}(d_1, \cdots, d_k)$. It is worthwhile to point out that the integer k, the rank of B_k in general, can be determined by the stopping criterion $\|A - X_k D_k Y_k^T\|_F \le \mathtt{tol}$ because the error $\|A - X_k D_k Y_k^T\|_F = \|A_k\|_F$ can be easily calculated by a recurrence relation derived in Section 4.

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

$$\|A - xdy^T\|_F^2 = \min_s \|A - xsy^T\|_F^2 = \|A\|_F^2 - d^2. \tag{2.2}$$

Since the left and right singular vectors u and v will undergo this sparsification process, it is not necessary to compute them to high accuracy (see the remark after Theorem 3.2). The details of **Step 2.1** of SLRA is listed below.

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:

$$x \leftarrow P_1^T \begin{bmatrix} u_+ \\ 0 \end{bmatrix} / \|u_+\|, \quad y \leftarrow P_2^T \begin{bmatrix} v_+ \\ 0 \end{bmatrix} / \|v_+\|.$$

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

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

3. Error analysis.

In this section we will compare the low-rank approximations computed by Algorithm SLRA with those obtained by SVD with respect to the reconstruction errors. One possible 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, and leave the positions of those nonzeros open. 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 the low-rank approximation $B_k = U_k D_k V_k^T$ 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. Use the same notation as in Step 2.1 of Algorithm SLRA, and assume that $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$ with $\epsilon \le 1/\sqrt{3}$. Then*

$$\|A - xdy^T\|_F \le \sqrt{1+\alpha\tau}\|A - u\sigma v^T\|_F, \tag{3.3}$$

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 $\|A - xdy^T\|_F^2 = \|A\|_F^2 - d^2$ as shown in (2.2), we need to derive a lower bound for $|d|$. To this end, partition

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

conformably with $P_1 u$ and $P_2 v$ (see Step 2.1 of Algorithm SLRA). It follows from the choice of d that

$$d = x^T A y = u_+^T A_{11} v_+ / (\|u_+\| \cdot \|v_+\|). \tag{3.4}$$

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

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

Subtracting the sum of the last two equations of the four equations above from the sum of the first two yields

$$u_+^T A_{11} v_+ = u_-^T A_{22} v_- + \sigma(1 - \|u_-\|^2 - \|v_-\|^2). \tag{3.5}$$

Then substituting (3.5) into (3.4) gives

$$\begin{aligned} |d| &= \frac{|u_-^T A_{22} v_- + \sigma(1 - \|u_-\|^2 - \|v_-\|^2)|}{\|u_+\| \cdot \|v_+\|} \\ &\ge \frac{\sigma(1 - \|u_-\|^2 - \|v_-\|^2) - \sigma\|u_-\| \cdot \|v_-\|}{\|u_+\| \cdot \|v_+\|} \\ &\ge \sigma\frac{1 - \frac{3}{2}(\|u_-\|^2 + \|v_-\|^2)}{1 - \frac{1}{2}(\|u_-\|^2 + \|v_-\|^2)} \\ &\ge \sigma\frac{1 - 3\epsilon^2}{1 - \epsilon^2} \\ &= \sigma\left(1 - \frac{2\epsilon^2}{1-\epsilon^2}\right) \ge 0. \end{aligned} \tag{3.6}$$

Here we have used the fact that $\|A_{22}\| \le \|A\| = \sigma$. It follows from $\|A - u\sigma v^T\|_F^2 = \|A\|_F^2 - \sigma^2$ that

$$\|A - xdy^T\|_F^2 \le \|A\|_F^2 - \sigma_1^2 \left(1 - \frac{2\epsilon^2}{1-\epsilon^2}\right)^2 = (1+\alpha\tau)\|A - u\sigma v^T\|_F^2,$$

where

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

completing the proof. □

In practice, 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 discarding 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 the approximate largest left and right singular vectors of A and $\sigma = \sigma_1(A)$. Use the notation of Step 2.1 of Algorithm SLRA and that of Theorem 3.1, and assume that $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$. Then*

$$\|A - xdy^T\|_F \le \sqrt{1+\alpha(\tau+\delta)}\|A - u\sigma v^T\|_F, \tag{3.7}$$

where τ is the same as that defined in Theorem 3.1 and

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

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 (3.5), we have

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

where $r = \big([u_+^T, u_-^T]r_1 + [v_+^T, v_-^T]r_2\big)/2$ with norm $\|r\| \le \eta\sigma$. By (3.6) and the inequality $\|u_+\|\|u_+\| \ge \sqrt{1-2\epsilon^2}$ we obtain

$$\begin{aligned} |d| &\ge \sigma\left(1 - \frac{2\epsilon^2}{1-\epsilon^2}\right) - \frac{\|r\|}{\|u_+\|\|u_+\|} \\ &\ge \sigma\left(1 - \frac{2\epsilon^2}{1-\epsilon^2} - \frac{\eta}{\sqrt{1-2\epsilon^2}}\right). \end{aligned}$$

The result (3.7) follows immediately from (2.2) and the following inequality

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

completing the proof. □

REMARK. We notice that η defined in Theorem 3.1 measures the accuracy of the approximate left and right singular vectors in a certain relative sense. The results in

Theorem 3.1 say that $\tau = O(\epsilon^2)$. Consequently, 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 estimate the reconstruction error of $\|A - X_k D_k Y_k^T\|$ for the general case with $k > 1$. The basic idea is to estimate $\{\sigma_j^2(A_k)\}$ in terms of $\{\sigma_j^2(A)\}$ (recall that $A_k = A_{k-1} - x_k d_k y_k^T$). The key of our proof is to derive a *tight* upper bound on $\sum_{j=1}^k \sigma_j^2(A - xdy^T)$ in terms of $\sum_{j=2}^{k+1} \sigma_j^2(A)$. Then we will apply the bounds to A_{k-1} and $x_k d_k y_k^T$ step by step, to obtain an upper bound of $\|A_k\| = \|A - X_k D_k Y_k^T\|$ in terms of $\{\sigma_j^2(A_0)\}$ with $A_0 = A$. With the assumptions that the left and right singular vectors are only approximate, the proof becomes rather unwieldy, and the bounds obtained are less transparent. Therefore, in the following we will assume that the left and right singular vectors u and v are computed exactly for each rank-one SVD approximation in the deflation process.

Notice that if $\{x, d, y\}$ is the *exact* largest singular triplet of A, $\sigma_i(A - xdy^T) = \sigma_{i+1}(A)$, 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 $i_1, \ldots, i_k$,

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

Therefore it is reasonable to expect an $O(\epsilon)$ estimation:

$$\sum_{j=1}^k \sigma_{i_j}^2(A - xdy^T) = \sum_{j=1}^k \sigma_{i_j+1}^2(A) + O(\epsilon) \tag{3.8}$$

when the triplet (x, d, y) is an $O(\epsilon)$ approximation of (u, σ, v). We now want to make (3.8) more precise and prove it rigorously. To this end, we first present several technical lemmas.

Lemma 3.3. *Denote $\hat{d} = u_+^T A_{11} v_+ / (\|u_+\| \cdot \|v_+\|)^2$ and $\sigma = \sigma_1(A)$. If $\|u_-\|^2 + \|v_-\|^2 \leq 2\epsilon^2$, assuming $\epsilon^2 \leq 1/\sqrt{5}$, we then have*

$$\left| \frac{\sigma - \hat{d}}{\sigma} \right| \leq c_1 \epsilon^2, \quad c_1 = \frac{1 + \epsilon^2}{(1 - \epsilon)^2} \tag{3.9}$$

Proof. By (3.4) and (3.5) we have

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

Hence

$$|\hat{d} - \sigma| \leq \sigma \frac{\|u_-\| \|v_-\| + \|u_-\|^2 \|v_-\|^2}{\|u_+\|^2 \|v_+\|^2}. \tag{3.10}$$

Writing $\|u_-\| = \sqrt{2} a \cos(\theta)$, $\|v_-\| = \sqrt{2} a \sin(\theta)$ for certain $\theta \in [0, \pi/2]$ and $0 < a \leq \epsilon$, and furthermore, denoting $t = a^2 \sin(2\theta) \in [0, a^2]$, we have

$$\frac{\|u_-\| \|v_-\| + \|u_-\|^2 \|v_-\|^2}{\|u_+\|^2 \|v_+\|^2} = \frac{a^2 \sin(2\theta) + a^4 \sin^2(2\theta)}{1 - 2a^2 + a^4 \sin^2(2\theta)} = \frac{t + t^2}{1 - 2a^2 + t^2}.$$

It can be shown that the function $g(t) \equiv (t+t^2)/(1-2a^2+t^2)$ is monotonically increasing in the interval $[0, a^2]$ if $a^2 \le \epsilon^2 \le 1/\sqrt{5}$. Therefore,

$$\begin{aligned} \frac{\|u_-\|\|v_-\| + \|u_-\|^2\|v_-\|^2}{\|u_+\|^2\|v_+\|^2} &\le g(a^2) = \frac{a^2+a^4}{1-2a^2+a^4} \\ &\le \frac{\epsilon^2+\epsilon^4}{1-2\epsilon^2+\epsilon^4} = c_1\epsilon^2. \end{aligned} \tag{3.11}$$

Equation (3.9) then follows from (3.10). □

LEMMA 3.4. *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*

$$\|E\|_F \le \sigma_1(A)(\sqrt{2}+\epsilon^2)\epsilon \tag{3.12}$$

and

$$|\sigma_j(A - xdy^T) - \sigma_{j+1}(A)| \le \sigma_1(A)(\sqrt{2}+\epsilon^2)\epsilon. \tag{3.13}$$

Proof. Let $\hat{h} = (\sigma - \hat{d})/\sigma$ with $\hat{d}$ defined in Lemma 3.3. Then $\hat{d} = \sigma(1-\hat{h})$ and

$$P_1 E P_2^T = \sigma \begin{bmatrix} \hat{h}\|v_+\|u_+ & \|v_-\|u_+ \\ \|v_+\|u_- & \|v_-\|u_- \end{bmatrix} \begin{bmatrix} v_+/\|v_+\| & 0 \\ 0 & v_-/\|v_-\| \end{bmatrix}^T.$$

Hence, by (3.10) we obtain that

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

Here we have used the following inequality

$$\left(\frac{\|u_-\| + \|v_-\|}{\|u_+\|\|v_+\|} \right)^2 \le \left(\frac{2\epsilon}{1-\epsilon^2} \right)^2,$$

which is valid for $\epsilon^2 \le 1/3$. This inequality can be proved using the same technique as when we prove (3.11). The standard perturbation bounds for singular values [5, Section 8.6.1] now give

$$\begin{aligned} \sigma_j(A - xdy^T) &= \sigma_j(A - u\sigma v^T + E) \\ &\le \sigma_j(A - u\sigma v^T) + \|E\| \\ &= \sigma_{j+1}(A) + \|E\| \\ &\le \sigma_{j+1}(A) + \sigma_1(A)(\sqrt{2}+\epsilon^2)\epsilon, \end{aligned}$$

completing the proof. □

Remark. It can be shown that if $\|u_-\| \leq \epsilon$ and $\|v_-\| \leq \epsilon$, then

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

Using the well-known Wielandt-Hofmann Theorem [5, Section 8.6.1] and Lemma 3.4, one can prove that

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

Therefore it is not difficult to show that

$$\|A - X_k D_k Y_k^T\|_F \leq (1 + c_k\epsilon)\|A - U_k\Sigma_k V_k^T\|_F, \tag{3.14}$$

with

$$c_k = \sqrt{2}\sum_{i=1}^{k}\sigma_i(A)/\Big(\sum_{i=k+1}^{n}\sigma_j^2(A)\Big)^{1/2} + O(\epsilon). \tag{3.15}$$

However, the coefficient c_k seems to give a less tight bound. To derive a much tighter bound for $\|A - X_k D_k Y_k^T\|_F$, we need the following key lemma.

Lemma 3.5. *Use 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 $i_1, \ldots, i_k$,*

$$\sum_{j=1}^{k}\sigma_{i_j}^2(A - xdy^T) \leq \sum_{j=1}^{k}\sigma_{i_j+1}^2(A) + \sigma_1(A)\sigma_2(A)\epsilon + c\sigma_1^2(A)\epsilon^2, \tag{3.16}$$

where $c = c_2$ for $k = 1$ and $c = 2c_2$ for $k > 1$, and $c_2 = (1 + c_1\epsilon^2)^2(3 + \sqrt{2}c_1\epsilon)$ with c_1 defined in Lemma 3.3.

Proof. Let the SVD of A be $A = U\Sigma V^T$ with $\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_n)$. To simplify the notation, denote $u = u_1$, $v = v_1$, $\sigma = \sigma_1$, and $\Sigma_2 = \operatorname{diag}(\sigma_2, \cdots, \sigma_n)$. Denote $B = U^T(A - xdy^T)V$. We also assume that $\|u_-\| \leq \|v_-\|$ which implies $\|u_-\| \leq \epsilon$. (Otherwise we can consider BB^T instead of B^TB in what follows.) The proof of this lemma consists of the following three parts.

1) We first show that the matrix B^TB is a rank-3 modification of $\operatorname{diag}(0, \Sigma_2^2)$, i.e.,

$$B^TB = \operatorname{diag}(0, \Sigma_2^2) + F, \quad \operatorname{rank}(F) \leq 3. \tag{3.17}$$

Thus it follows from [16, Page 202] that for distinct indexes $i_1, \ldots, i_k$,

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

with the notation $\lambda_j(\cdot)$ denoting the j-th *largest* eigenvalue of a symmetric matrix. To write the above in another way, we have

$$\sum_{j=1}^{k}\sigma_{i_j}^2(A - xdy^T) \leq \sum_{j=1}^{k}\sigma_{i_j+1}^2(A) + \sum_{j=1}^{k}\lambda_j(F). \tag{3.18}$$

To this end, partition

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

(See **Step 2.1 of SLRA** for the definition of the permutation matrices P_1 and P_2.) It can be verified that

$$U^T x d y^T V = \hat{d}\Big(e_1e_1^T - e_1w_2^T - w_1e_1^T + w_1e_2^T\Big),$$

where

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

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

Furthermore, we have

$$w_{11} = \|u_-\|^2 = \|w_1\|^2 \le \epsilon^2, \quad w_{12} = \|v_-\|^2 = \|w_2\|^2 \le 2\epsilon^2. \tag{3.19}$$

Therefore, we can write

$$\begin{aligned} B &\equiv U^T(A - xdy^T)V \\ &= \operatorname{diag}(0, \Sigma_2) + \hat{d}\Big(he_1e_1^T + e_1w_2^T + w_1e_1^T - w_1e_2^T\Big) \\ &= \operatorname{diag}(0, \Sigma_2) + \hat{d}[e_1, w_1] \begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix} [e_1, w_2]^T, \end{aligned}$$

i.e., B is a rank-2 modification of $\operatorname{diag}(0, \Sigma_2)$. Here

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

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

$$w_3 = \begin{bmatrix} 0 \\ \Sigma_2 w_{21} \end{bmatrix}, \quad \Delta_1 = \begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & w_{11} \\ w_{11} & w_{11} \end{bmatrix} \begin{bmatrix} h & 1 \\ 1 & -1 \end{bmatrix}.$$

Then it can be verified that

$$B^TB = \operatorname{diag}(0, \Sigma_2^2) + [e_1, w_2, w_3]\Delta[e_1, w_2, w_3]^T \equiv \operatorname{diag}(0, \Sigma_2^2) + F,$$

where

$$\Delta = \hat{d} \begin{bmatrix} \hat{d}\Delta_1 & \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ \begin{bmatrix} 1 & -1 \end{bmatrix} & 0 \end{bmatrix} = \hat{d} \begin{bmatrix} \hat{d}(h^2 + 2hw_{11} + w_{11}) & \hat{d}h(1 - w_{11}) & 1 \\ \hat{d}h(1 - w_{11}) & \hat{d}(1 - w_{11}) & -1 \\ 1 & -1 & 0 \end{bmatrix}.$$

Therefore, (3.17) holds.

2) We now prove that the matrix F has a negative eigenvalue, which implies that the last term of (3.18) is a sum of at most two largest eigenvalues of F. First $\mathrm{rank}([e_1, w_2, w_3]) \geq 2$ since e_1 is orthogonal to w_3. Without loss of generality, we assume that $\mathrm{rank}([e_1, w_2, w_3]) = 3$. (The case when $\mathrm{rank}([e_1, w_2, w_3]) = 2$ is simpler and can be similarly handled.) Thus by Sylvester's Law of Inertia [5, Theorem 8.1.17], the number of positive eigenvalues of F is equal to the number of positive eigenvalues of Δ. Therefore it is enough to show that Δ has only two positive eigenvalues. Clearly, Δ has at least one positive eigenvalue since it has a positive diagonal element. It can be shown that the determinant of Δ is negative: $\det(\Delta) = -\hat{d}^2(1+h)^2 < 0$. It implies that Δ has one and only negative eigenvalue because Δ is obviously not negative definite. Therefore, Δ has exactly two positive eigenvalues, and so does F. Hence we can write (3.18) as

$$\sum_{j=1}^{k} \sigma_{i_j}^2(A - xdy^T) \leq \sum_{j=1}^{k} \sigma_{i_j+1}^2(A) + \sum_{j=1}^{\min\{k,2\}} \lambda_j(F). \tag{3.20}$$

3) We finally derive upper bounds for $\lambda_1(F)$ and $\lambda_2(F)$ which lead to the inequality (3.16). To this end, we write

$$\begin{aligned} F &= \hat{d}[e_1, w_3] \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} [e_1, w_3]^T + \\ &\quad + [e_1, w_2, w_3] \begin{bmatrix} \hat{d}^2\Delta_2 & [0, -\hat{d}]^T \\ [0, -\hat{d}] & 0 \end{bmatrix} [e_1, w_2, w_3]^T \equiv H + \tilde{F}. \end{aligned}$$

It is easy to see that $\lambda(H) = \{\hat{d}\|w_3\|, 0, \ldots, 0, -\hat{d}\|w_3\|\}$. By (3.9) and the inequality $\|w_3\| \leq \sigma_2\epsilon$, we thus have

$$\begin{aligned} \lambda_1(F) &\leq \hat{d}\|w_3\| + \|\tilde{F}\| \leq \sigma_1\sigma_2\epsilon(1 + c_1\epsilon^2) + \|\tilde{F}\|, \qquad (3.21) \\ \lambda_2(F) &\leq \|\tilde{F}\|. \qquad (3.22) \end{aligned}$$

To estimate $\|\tilde{F}\|$, we normalize w_2 and w_3, and let $\hat{w}_2 = w_2/\|w_2\|$ and $\hat{w}_3 = w_3/\|w_3\|$. It is easy to see that

$$\|[e_1, \hat{w}_2, \hat{w}_3]\| \leq \sqrt{2}, \quad \|\tilde{F}\| \leq 2\|\hat{F}\|$$

with $\hat{d}h = \sigma - \hat{d}$, and

$$\begin{aligned} \hat{F} &\equiv (\hat{f}_{ij})_{i,j=1}^3 \\ &= \begin{bmatrix} (\sigma - \hat{d})^2 + \hat{d}(2\sigma - \hat{d})w_{11} & \hat{d}(\sigma - \hat{d})(1 - w_{11})\|w_2\| & 0 \\ \hat{d}(\sigma - \hat{d})(1 - w_{11})\|w_2\| & \hat{d}^2(1 - w_{11})\|w_2\|^2 & -\hat{d}\|w_2\| \cdot \|w_3\| \\ 0 & -\hat{d}\|w_2\| \cdot \|w_3\| & 0 \end{bmatrix}. \end{aligned}$$

By Lemma 5.2 of [17] and the fact that $\hat{f}_{13} = \hat{f}_{31} = \hat{f}_{33} = 0$, we have

$$\begin{aligned} \|\hat{F}\| &\leq \max\left\{|\hat{f}_{11}|, \left\| \begin{bmatrix} \hat{f}_{22} & \hat{f}_{23} \\ \hat{f}_{32} & \hat{f}_{33} \end{bmatrix} \right\| \right\} + |\hat{f}_{12}| \\ &\leq \max\left\{|\hat{f}_{11}|, |\hat{f}_{22}| + |\hat{f}_{23}|\right\} + |\hat{f}_{12}|. \end{aligned}$$

By (3.9) and (3.19), it is easy to see that $\|\hat{F}\| = O(\epsilon^2)$. Furthermore, it can be verified that

$$|\hat{f}_{11}| \le |\hat{f}_{22}| + |\hat{f}_{23}| \le 3\sigma^2(1+c_1\epsilon^2)^2\epsilon^2, \quad |\hat{f}_{12}| \le \sqrt{2}c_1\sigma^2(1+c_1\epsilon^2)\epsilon^3,$$

which leads to

$$\|\hat{F}\| \le (1+c_1\epsilon^2)^2(3+\sqrt{2}c_1\epsilon)\sigma^2\epsilon^2 \equiv c_2\sigma^2\epsilon^2,$$

and

$$\lambda_1(F) \le \sigma_1\sigma_2\epsilon + c_2\sigma^2\epsilon^2, \quad \lambda_2(F) = c_2\sigma^2\epsilon^2.$$

Combining the above bounds with (3.20) yields the result (3.16). ◻

Now we are ready to prove our main theorem.

THEOREM 3.6. *Use the notation in Step 2.1 of Algorithm SLRA, and assume in each iteration of Step 2.1* $\|u_-\|^2 + \|v_-\|^2 \le 2\epsilon^2$ *with* $\epsilon^2 < 1/3$. *Then*

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

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

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

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

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

Since $A_k = A - X_k D_k Y_k^T$, applying Lemma 3.5 to $A_k = A_{k-1} - x_k d_k y_k^T$, we have

$$\begin{aligned} (3.23)\ \|A - X_k D_k Y_k^T\|_F^2 &= \sum_{j=1}^n \sigma_j^2(A_k) \\ &\le \sum_{j=2}^n \sigma_j^2(A_{k-1}) + \sigma_1(A_{k-1})\sigma_2(A_{k-1})\epsilon + c\sigma_1^2(A_{k-1})\epsilon^2 \\ &\le \sum_{j=k+1}^n \sigma_j^2(A) + \sum_{j=0}^{k-1}\sigma_1(A_j)\sigma_2(A_j)\epsilon + c\sum_{j=0}^{k-1}\sigma_1^2(A_j)\epsilon^2. \end{aligned}$$

On the other hand, by Lemma 3.4, we have with $c_3 = \sqrt{2} + \epsilon^2$

$$\begin{aligned} \sigma_1(A_j) &\le \sigma_2(A_{j-1}) + c_3\sigma_1(A_{j-1})\epsilon \\ &\le \sigma_3(A_{j-2}) + c_3(\sigma_1(A_{j-2}) + \sigma_1(A_{j-1}))\epsilon \\ &\le \cdots \\ &\le \sigma_{j+1}(A) + c_3\sum_{i=0}^{j-1}\sigma_1(A_i)\epsilon. \end{aligned} \tag{3.24}$$

Let $s_j = \sum_{i=0}^{j-1}\sigma_1(A_i)$. Then by (3.24)

$$\begin{aligned} s_j &= \sigma_1(A_{j-1}) + s_{j-1} \le \sigma_j(A) + (1 + c_3\epsilon)s_{j-1} \\ &\le \sigma_j(A) + (1 + c_3\epsilon)(\sigma_{j-1}(A) + (1 + c_3\epsilon)s_{j-2}) \\ &\le \cdots \\ &\le \sum_{i=1}^{j}(1 + c_3\epsilon)^{j-i}\sigma_i(A). \end{aligned} \tag{3.25}$$

Substituting (3.25) into (3.24) gives

$$\sigma_1(A_j) \le \sigma_{j+1}(A) + c_3\sum_{i=1}^{j}(1 + c_3\epsilon)^{j-i}\sigma_i(A)\epsilon \equiv \sigma_{j+1}(A) + \phi_j\epsilon,$$

where $\phi_j = c_3\sum_{i=1}^{j}(1 + c_3\epsilon)^{j-i}\sigma_i(A)$. Similarly, we have

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

Therefore,

$$\begin{aligned} &\sum_{j=0}^{k-1}\sigma_1(A_j)\sigma_2(A_j) \\ \le\ &\sum_{j=1}^{k}\big(\sigma_j(A)\sigma_{j+1}(A) + (\sigma_j(A) + \sigma_{j+1}(A) + \phi_{j-1}\epsilon)\phi_{j-1}\epsilon\big), \end{aligned} \tag{3.26}$$

and

$$\sum_{j=0}^{k-1}\sigma_1^2(A_j) \le \sum_{j=1}^{k}\big(\sigma_j^2(A) + 2\sigma_j(A)\phi_{j-1}\epsilon + \phi_{j-1}^2\epsilon^2\big). \tag{3.27}$$

Combining (3.23), (3.26) and (3.27) we obtain that

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

where

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

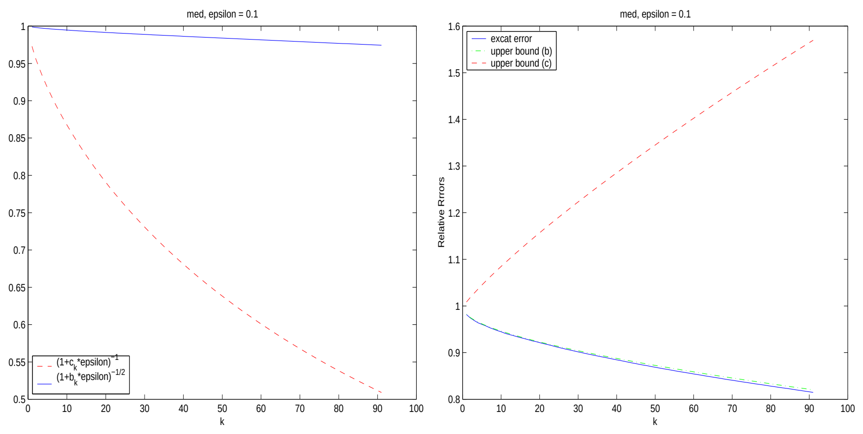

FIG. 1. $(1 + c_k * \epsilon)^{-1}$ *and* $(1 + b_k * \epsilon)^{-1/2}$ *(left) and the relative errors (right).*

completing the proof. □

The bound proved in the above theorem usually is much tighter than the bound in (3.14). In Figure 1, for various k, we plot the quantities $(1+c_k\epsilon)^{-1}$ 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.

4. Computational Variations. In this section, we first discuss several computational variations of Algorithms SLRA, in particular we look at two approaches for sparsifying vectors in Step 2.1 of Algorithm SLRA. We first briefly discuss how to find approximations to the largest singular triplet of a matrix.

Computing Largest Singular Triplets. As we mentioned in Section 2, the largest singular triplet $\{u, \sigma, v\}$ does not need to 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 [5, 13]. Using the power method, we suggest performing several steps of power iteration as follows,

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

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 can run several Lanczos iterations to generate a pair of orthogonal bases $\{u_1, \cdots, u_\beta\}$ 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 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 $\tilde{u} \equiv P_1 u = \begin{bmatrix} u_+ \\ u_- \end{bmatrix}$, $\tilde{v} \equiv P_2 v = \begin{bmatrix} v_+ \\ v_- \end{bmatrix}$ with

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

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

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

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

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

$$k_w = \min \left\{ k \ge k_0 \,\Big|\, \sum_{j=1}^{k} \tilde{w}_j^2 \ge 2\epsilon^2 \right\},$$

  where k_0 is the smallest integer such that the section $w(\mathtt{1:k_0})$ contains both u-components and v-components. 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 $\tilde{w}(\mathtt{1:k_w})$ also determine $\tilde{u}(\mathtt{1:k_u})$ and $\tilde{v}(\mathtt{1:k_v})$, where k_u and k_v are, respectively, the numbers of u-components and v-components of $\tilde{w}(\mathtt{1:k_w})$.

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 Algorithm SLRA, the tolerance ϵ can be a pre-determined constant or be chosen dynamically during the iteration process. We will use, for variable tolerance, at the k-th iteration

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

which depends on the approximation computed by previous iterations.

Choice of k. Notice that the norm of error matrix A_k at step k can be written as

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

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 for Algorithm SLRA:

$$\|A_k\|_F \le \texttt{tol}$$

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 Frobenius 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 Frobenius norm will be smaller. The question now is the following: 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.

A combinatorial optimization problem. Now we reexamine the optimization problem (2.1) for $k = 1$. We can impose the following constraints on the number of nonzeros of x and y: $\texttt{nnz(x)} = \texttt{n}_\texttt{x}, \texttt{nnz(y)} = \texttt{n}_\texttt{y}$, 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

Table 1
Self-correction phenomenon

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

$$\min_{\hat{x}\in\mathcal{R}^{n_x},\hat{y}\in\mathcal{R}^{n_y}} \|A([i_1,\ldots,i_{n_x}],[j_1,\ldots,j_{n_y}]) - \hat{x}d\hat{y}^T\|_F, \tag{4.28}$$

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 $\tilde{A}$ 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. Consider the following matrix

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

The goal is to compare the computed sparse low-rank approximation with the *optimal* solution of the combinatorial optimization problem (4.28) computed by exhaustive search.

We first compute the sparse approximation $X_kD_kY_k^T$ for $k=2$ using Algorithm SLRA with $\epsilon = 0.3$ and $\beta = 4$ for computing the approximate largest singular triplet using Lanczos bidiagonalization. The computed vectors x_i and y_i have the numbers of nonzeros listed below.

$$\texttt{nnz(x}_1\texttt{)} = 5, \quad \texttt{nnz(y}_1\texttt{)} = 4, \quad \texttt{nnz(x}_2\texttt{)} = 3, \quad \texttt{nnz(y}_2\texttt{)} = 3.$$

Next we compute the best rank-one approximation $u_1s_1v_1^T$ to A with the constraints $\texttt{nnz(u}_1\texttt{)} = \texttt{nnz(x}_1\texttt{)}$ and $\texttt{nnz(v}_1\texttt{)} = \texttt{nnz(y}_1\texttt{)}$, and then the best rank-one approximation $u_2s_2v_2^T$ to matrix $A - u_1s_1v_1^T$ with the constraints $\texttt{nnz(u}_2\texttt{)} = \texttt{nnz(x}_2\texttt{)}$ and $\texttt{nnz(v}_2\texttt{)} =$

nnz(y_2). The above two steps for computing u_i and v_i are carried out using exhaustive search. 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, i.e., wherever x_i (or y_i) has a zero element, u_i (or v_i) also has a zero element in the same position, and vice versa. However, notice that the values of nonzero elements are different but very close.

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 present several numerical experiments to illustrate the effectiveness and efficiency of our approach for computing sparse low-rank approximations. We will compare the performance of Algorithm SLRA with that of SVD and the approach proposed in [15] with respect to the following two issues:

1) the reconstruction errors; and
2) the computational complexity and storage required.

For the numerical experiments, we generate a collection of test matrices which are listed below together with some relevant statistics: 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 [2, 9]. We do not claim that the collection is comprehensive.

	Matrix	m	n	nnz(A)	Density(%)
1	ash958	958	292	19196	0.68
2	illc1033	1033	320	4732	1.43
3	cisi	5081	1469	66241	0.89
4	cacm	3510	3204	70339	0.63
5	med	5504	1033	51096	0.90
6	npl	4322	11429	224918	0.46
7	watson4	467	468	2836	1.30
8	orsirr2	886	886	5970	0.76
9	e20r1000	4241	4241	131430	0.73

Some explanation of the notation we used is in order here: m and n represent the row and column dimensions, respectively, of the given matrix. As used before, nnz(A)

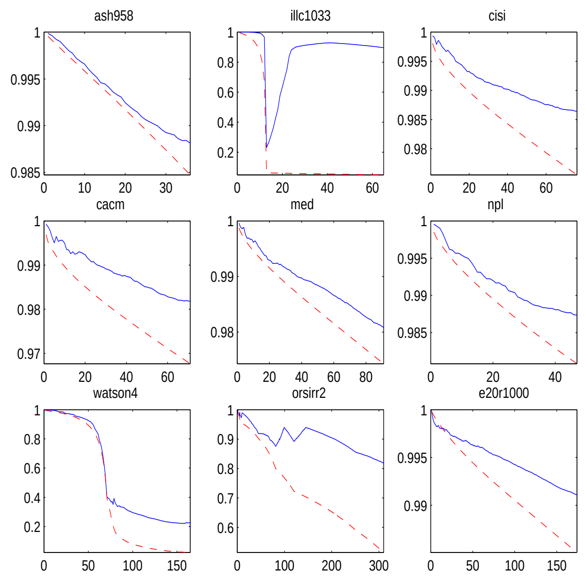

FIG. 2. *The computed* $\mathrm{er}(k)$ *(solid lines) and the lower bounds* $(1+b_k\epsilon)^{-1/2}$ *(dashed lines).*

denotes the number of nonzero elements of A. Density is computed as `nnz(A)/(mn)`, the percentage of nonzero elements of a matrix.

In order to compare our algorithm with SPQR in [15], for each matrix A, we first use SPQR to compute a rank-k approximation $B = A_c M A_r^T$. We use $k = 300$ if $\min(m,n) > 500$, otherwise we use $k = 100$. Then we let $\mathrm{tol}(A) = \|A - B\|_F$, we seek to find a low-rank approximation using SLRA such that

$$\|A - X_k D_k Y_k^T\|_F \le \mathrm{tol}(A).$$

TEST 1. We compare the low-rank approximations computed by Algorithm SLRA with constant tolerance $\epsilon = 0.1$ and those computed by SVD. The dimension used for Lanczos bidiagonalization for computing the approximate largest singular vectors is $\beta = 4$ (See the definition of β in the previous section). To illustrate the reconstruction error $\|A - X_k D_k Y_k^T\|_F$, we use the error ratio $\mathrm{er}(k)$ defined by

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

to measure the effectiveness of Algorithm SLRA. It is easy to see that $0 \le \mathrm{er}(k) \le 1$. The larger the error ratio is, the more effective SLRA is. Below we list the error ratios of SLRA with constant tolerance $\epsilon = 0.1$ using the *separated sorting scheme*. The rank k is chosen to be $5 \sim 20\%$ of the size $l = \min(m,n)$ of a given matrix A. We also computed the average error ratio defined as

$$\mathrm{Average} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{er}(i),$$

where k is the smallest integer satisfying $\|A - X_k D_k Y_k^T\|_F \le \epsilon$.

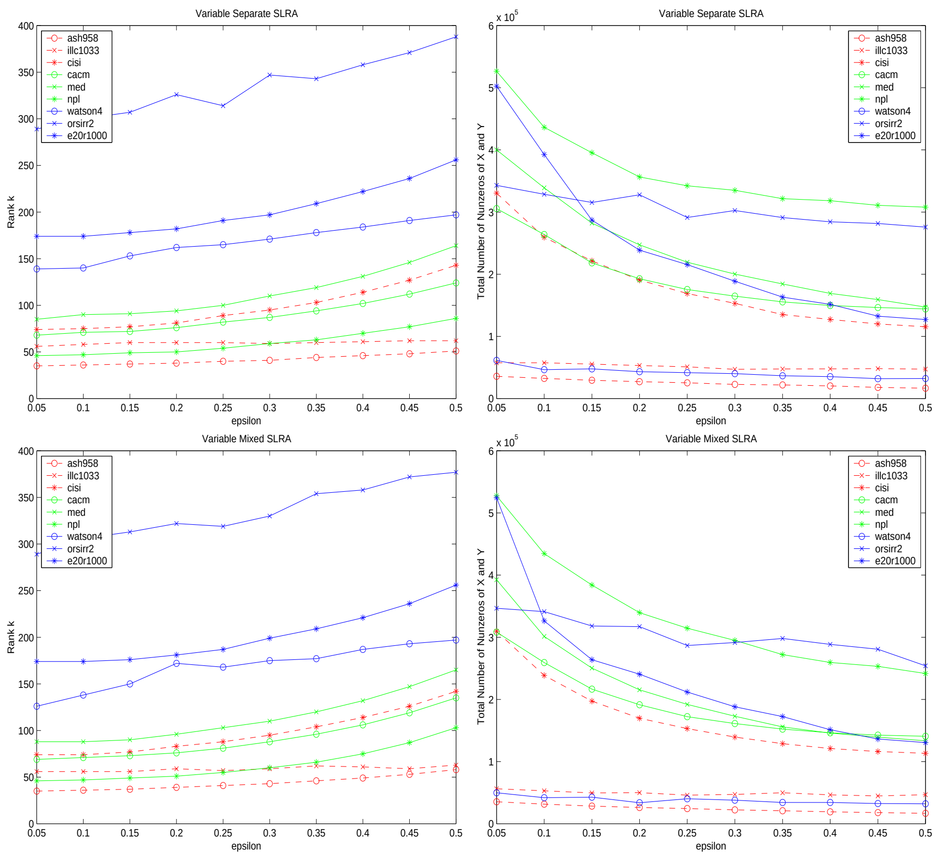

FIG. 3. *Plots for ranks (left) and numbers of nonzero elements of X_k and Y_k (right) vs starting epsilon for the variable tolerance, separated (top) and mixed (bottom) sorting approaches.*

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

For these matrices, Theorem 3.6 gives tight bounds for the ratios. Figure 2 plots, with respect to k, the lower bounds $(1+b_k\epsilon)^{-1/2}$ (dashed lines) given in Theorem 3.6 and the ratio quantities $\mathrm{er}(k)$ (solid line) computed by the separated sorting SLRA with $\epsilon = 0.1$ for all the nine matrices. These examples show that SLRA has very high error ratios for most of the test matrices, especially for the term-document matrices.

TEST 2. In general, the *mixed sorting scheme* gives a smaller number of nonzero elements, for the sparse factors X_k and Y_k, i.e., less storage required, than the *separated sorting scheme* if we use the same tolerance sequence while the rank k of the low-rank approximations computed by the different schemes are about the same. We computed the low-rank approximations using Algorithm SLRA with the same variable tolerance scheme for both the separated and mixed sorting schemes. Different starting tolerances ϵ `= 0.05:0.05:0.5` are used for each test matrix. In Figure 3 we plot the ranks (left) and the total number of nonzero elements of X_k and Y_k (right) computed by SLRA with separated (top) and mixed (bottom) sorting schemes. For each test matrix, the ranks computed by the two sorting schemes are about the same while mixed sorting scheme gives smaller number of nonzero elements, this is especially the case for the starting tolerances around $\epsilon = 0.15$.

TEST 3. In this test we compare, respectively, the ranks of the low-rank approximations, the computation cost in flops and storage required for SVD, SPQR, and SLRA using variable tolerance and mixed sorting scheme. For SLRA, we use $\epsilon = 0.1$ as the starting tolerance and $\beta = 6$ iterations for Lanczos bidiagonalization. The low-rank approximations computed by the three approaches have the same reconstruction errors for each test matrix. In general, as we mentioned before, SVD produces dense factors even when A is sparse. Therefore the low-rank approximation computed by SVD requires at least $(m + n + 1)k$ storage for its associated factors. For SPQR, the rank k of the low-rank approximation $B_k = A_c M A_r^T$, is usually quite large compared with the rank of the optimal low-rank approximation generated by SVD. Since the matrix M is generally dense, the storage required is dominated by M resulting in larger than k^2 storage requirement. In contrast, SLRA can produce low-rank approximations with small rank k and good degree of sparsity of the factors X_k and Y_k. (The number of nonzeros can be reduced by increasing the starting tolerance ϵ, which also increases the flops and ranks.) We list below the comparison for the term-document matrices in the test collection.

atrix		rank	total nnz	flops
cisi	TSVD	68	449412	6925863163
	SLRA	72	217401	523406959
	SPQR	300	129720	568382817
cacm	TSVD	63	426951	5390479001
	SLRA	67	216982	478032905
	SPQR	300	133784	463854304
med	TSVD	79	522664	9598485598
	SLRA	84	278456	658852943
	SPQR	300	120444	469695010
npl	TSVD	41	647472	6208537332
	SLRA	44	384118	616205165
	SPQR	300	227567	588513394

However, we should mention that the performance of SLRA is not as good as SPQR when the matrix A is close to a highly rank-deficient matrix. For example, let A be the matrix `illc1033` in the test collection. We compute, using SPQR, a rank-100 approximation $B = A_c M A_r^T$. The storage required (the number of nonzeros) for the computed low-rank approximation is about 20% of that for the best approximation B^* computed by SVD that achieves the same reconstruction error. SLAP with $\epsilon = 0.1$ gives an approximation B_k that has the same reconstruction error as that of SPQR and the storage required is 85% of that for B^* though the rank of B_k is close to the optimal rank and much smaller than the rank of B. SPQR also requires less flops for computing the low-rank approximation. In general, SPQR is very effective for sparse matrices that are close to highly rank-deficient and the rank of the low-rank approximation can be predetermined. However, SPQR is not convenient to use if the user just impose an upper bound on reconstruction error.

6. Concluding Remarks. We have presented algorithms for computing matrix low-rank approximations with sparse factors. We also gave a detailed error analysis comparing the reconstruction errors for the low-rank approximations computed by SVD and the low-rank approximations computed by our sparse low-rank 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 investigating their relationships 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 $\|\cdot\|_F$.

Acknowledgment. Part of this work was done while the first two authors were visiting National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory. The authors also want to thank the anonymous referees for their comments and suggestions that greatly improved the presentation of the paper.

REFERENCES

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

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

[3] C. Couvreur and Y. Bresler. On the optimality of backward greedy algorithm for the subset selection method. *SIAM J. Matrix Analysis and Applications*, 21:797–808, 2000.

[4] M. Evans, Z. Gilula and I. Guttman. Latent class analysis of twy-way contingency tables by Bayesian methods. *Biometrika*, 76:557–563, 1989.

[5] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland, 3nd edition, 1996.

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

[7] 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.

[8] D. Lee and S. Seung. Learning the parts of objects by non-negative matrix factorization. Nature, 401:788–791, 1999.

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

[10] B. Natarajan. Sparse approximate solutions to linear systems. SIAM J. Computing, 24:227–234, 1995.

[11] D. P. O'Leary and Shmuel Peleg. Digital Image compression by outer product expansion. IEEE Transactions on Communications, 31:441–444, 1983.

[12] Haesun Park, Lei Zhang, and J. Ben Rosen. Low Rank Approximation of a Hankel Matrix by Structured Total Least Norm. TR 97-043, Department of Computer Science, University of Minnesota, 1997.

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

[14] H. Simon and H. Zha. Low-rank matrix approximation using the Lanczos bidiagonalization process. *SIAM Journal of Scientific Computing,* 21:2257–2274, 2000.

[15] 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.

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

[17] H. Zha and Z. Zhang. Matrices with low-rank-plus-shift structure: partial SVD and latent semantic indexing. *SIAM Journal on Matrix Analysis and Applications*, 21:522–536, 1999.