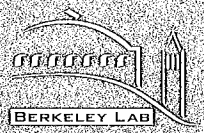
LBNL-44217
Preprint



ERNEST ORLANDO LAWRENCE
BERKELEY NATIONAL LABORATORY

Low-Rank Approximations with Sparse Factors II: Penalized Methods with Discrete Newton-Like Iterations

Zhenyue Zhang, Hongyuan Zha, and Horst Simon

National Energy Research
Scientific Computing Division

September 1999

Submitted to
SIAM Journal of
Matrix Analysis

LBNL-44217

Low-Rank Approximations with Sparse Factors II: Penalized Methods with Discrete Newton-Like Iterations

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

September 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 by the NSFC under Project No. 19771073 and the National Science Foundation under Grant Nos. CCR-9619452 and CCR-9901986.

LOW-RANK APPROXIMATIONS WITH SPARSE FACTORS II: PENALIZED METHODS WITH DISCRETE NEWTON-LIKE ITERATIONS

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

Abstract. In [9], we developed algorithms for computing low-rank approximations of matrices together with a detailed error analysis. The low-rank approximations are constructed in a certain factored form with the sparsity patterns of the factors controlled by some user determined parameters. In this paper, we cast the *sparse* low-rank approximation problem in a penalized optimization framework. We discuss various approximation schemes for the optimization problem that gives arise to formulations of the problem that is more amendable for numerical computations. We develop a globally convergent *discrete* secant method for solving those penalized optimization problems. We also compare the reconstruction errors of the sparse low-rank approximations computed by our new methods with those obtained using the methods in [9]. Numerical examples show that the penalized methods are more robust and produce approximations with lower ranks and more sparse factor.

1. Introduction. Low-rank approximations of matrices have many applications in information retrieval, data mining and solving ill-posed problems, to name a few [5, 8]. The theory of singular value decomposition (SVD) provides the best rank-k approximation $\text{best}_k(A)$ of a given matrix A in terms of its singular values and singular vectors,

$$\text{best}_k(A) \equiv U_k \Sigma_k V_k^T = [u_1, \ldots, u_k] \, \text{diag}(\sigma_1, \ldots, \sigma_k) [v_1, \ldots, v_k]^T,$$

where $\sigma_i, i = 1, \ldots, k$, are the largest k singular values of A, and u_i and v_i are the corresponding left and right singular vectors [1]. Notice that even when A is sparse, there is in general no guarantee that $\text{best}_k(A)$ will be sparse, not even the factors U_k and V_k. To remedy this drawback of the low-rank approximations computed by SVD, it was proposed to write a low-rank approximation B_k of A in a factored form $B_k = X_k D_k Y_k$ [3, 6]. In [9] we further developed this idea and propose to find $B_k = X_k D_k Y_k$ that solves the following optimization problem,

$$\min\{\|A - X_k D_k Y_k^T\|_F \mid D \text{ diagonal}, X_k \in \mathcal{R}^{m \times k} \text{ and } Y_k \in \mathcal{R}^{n \times k} \text{ sparse } \}. \tag{1.1}$$

* 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.

In [9], several algorithms are developed for choosing the sparsity patterns of X_k and Y_k, and a detailed error analysis of our proposed algorithms is given that compares the computed sparse low-rank approximations with those obtained from SVD and some of the previous methods developed in [3, 6]. The basic idea is to use a sequence of rank-one deflation steps to construct the approximation

$$B_k = X_k D_k Y_k^T = \sum_{i=1}^{k} x_i d_i y_i^T.$$

At each deflation step, approximate largest left and right singular vectors u_i and v_i of the deflated matrix $A_{i-1} = A - B_{i-1}$ are used to construct a sparse rank-one approximation $x_i d_i y_i^T$ to matrix A_i. Specifically, the sparse vectors x_i and y_i are obtained by discarding small components of u_i and v_i. We proved that if the norm of the vector consisting of the discarded components is no greater than $\sqrt{2}\epsilon$ at each step, then the computed sparse low-rank approximation B_k has *reconstruction error*, defined as $\|A - B_k\|_F$, no greater than the best rank-k approximation $\text{best}_k(A)$ by a factor $(1 + b_k\epsilon)^{1/2}$, i.e.,

$$\|A - B_k\|_F \leq (1 + b_k\epsilon)^{1/2} \|A - \text{best}_k(A)\|_F,$$

where $b_1 = \sigma_1^2(A)/\left(\sigma_2^2(A) + \cdots \sigma_n^2(A)\right)$ and

$$b_k = \frac{\sum_{i=1}^{k} \sigma_i(A)\sigma_{i+1}(A)}{\sum_{i=k+1}^{n} \sigma_i^2(A)} + O(\epsilon), \quad k \geq 2.$$

The tolerance ϵ determined by the user can balance the tradeoff between sparsity and good reconstruction error of the low-rank approximations. We suggested in [9] that the size of the tolerance used at each deflation step can be a variable determined by

$$\epsilon_i = \frac{\|A_{i-1}\|_F}{\|A\|_F}\epsilon$$

for step i. Numerical results in [9] show that the variable-tolerance scheme works better than the constant-tolerance scheme. In general if we fix the desirable reconstruction error, reducing ϵ will yield a smaller rank k and X_k and Y_k that have poor degree of sparsity while increasing ϵ will cause the rank of the low-rank approximation to increase but the degree of sparsity of the factors is reduced. However, we also observed that the rank and the degree of sparsity computed by the methods in [9] sometimes can be quite sensitive to the choice of ϵ (ϵ_k), i.e., a slight change of ϵ, though does not change the reconstruction error very much, can have a much greater effect on the rank of the low-rank approximation and the degree of sparsity of its factors. This behavior is rather undesirable.

The goal of this paper is to develop more robust methods for sparse low-rank approximations. Our basic idea for the improved algorithms is to use penalty terms to penalize low-rank approximations with factors X_k and Y_k that have large number of nonzeros. In a rather general framework, we can consider the following optimization problem,

$$\min\{\text{nnz}(X_k) + \text{nnz}(Y_k) + \|A - X_k D_k Y_k^T\|_F \mid D \text{ diagonal}\}, \tag{1.2}$$

where $\text{nnz}(\cdot)$ denotes the number of nonzeros. In essence, we want a low-rank approximation $B_k = X_k D_k Y_k^T$ to have a small reconstruction error $\|A - X_k D_k Y_k^T\|_F$ and

at same time we also penalize those B_k the X_k and Y_k factors of which have large number of nonzero elements. We can certainly use some general techniques to solve the optimization problem (1.2). However, the problem itself possesses many useful structures that deserves exploitation. In particular, we will use the deflation technique to reduce the problem to the problem of finding a sequence of sparse rank-one approximations, and build the low-rank approximation one rank at a time [3, 6, 9].

Our ultimate goal is to reduce (1.2) to a simpler form that is easy to solve. We will describe the reduction process in several steps (each of which involves certain approximation) and give the rational behind each steps. When we reach the final simple formulation, we will have a penalized optimization problem that is easy to solve, and at the same time has a solution that is close to the solution of (some variation of) (1.2).

The rest of the paper is organized as follows. In section 2, we motivates the introduction of the penalized optimization problem. Specifically, we look at the relation between the singular values of a matrix with those of its submatrices. In the rank-one case, we also give an upper bound of the number of nonzeros of the sparse factors in terms of elements of the largest left and right singular vectors. In section 3, we focus on the rank-one case of the penalized optimization problem and discuss ways to reduce it to a simpler form that is more amenable for numerical computations. In section 4, we propose a *discrete* globally convergent secant method for solving the simplified rank-one penalized optimization problem. Many new computational issues arise in the discrete secant method: We focus on how to compute the secant directions at each iteration step, and how to select the next iterate to guarantee the existence of a bracketing interval. In section 5, we present several numerical examples, and make comparison with previously proposed methods.

2. Motivations. To motivate the introduction of the penalized optimization problem similar to (1.2), we look at several issues concerning the trade-off of degree of sparsity of the low-rank approximations and the their reconstruction errors. First we have the following general result.

THEOREM 2.1. *Let matrix*

$$F = \begin{bmatrix} A & B \\ C^T & D \end{bmatrix},$$

and let (σ_1, u_1, v_1) *be the largest singular triplet of* A. *Then* $\sigma_1 = \sigma_{\max}(F)$ *if and only if*

$$u_1^T B = 0, \quad v_1^T C = 0$$

and $\left(\sigma_1, \begin{bmatrix} u_1 \\ 0 \end{bmatrix}, \begin{bmatrix} v_1 \\ 0 \end{bmatrix}\right)$ *is the largest singular triplet of* F.

Proof. We only need to prove the "if" part. The "only if" part is trivial. We will use induction. First we assume that B and C are column vectors. Let

$$A = U\Sigma V^T = [u_1, \ldots, u_l] \operatorname{diag}(\sigma_1, \ldots, \sigma_l)[v_1, \ldots, v_l]^T$$

be the SVD of A. Denoting $\tilde{B} = U^T B, \tilde{C} = V^T C$, we have

$$FF^T = \operatorname{diag}(U, 1) \begin{bmatrix} \Sigma^2 + \tilde{B}\tilde{B}^T & \Sigma\tilde{C} + D\tilde{B} \\ \tilde{C}^T\Sigma + D\tilde{B}^T & D^2 + \tilde{C}^T\tilde{C} \end{bmatrix} \operatorname{diag}(V^T, 1).$$

By the assumption of the theorem σ_1^2 is the largest eigenvalue of FF^T and Σ^2 which implies that σ_1^2 is also the largest eigenvalue of $\Sigma^2 + \tilde{B}\tilde{B}^T$. It follows that the first element of $\tilde{B}$ must be zero. Apply the same argument to F^TF we can also show that the first element of $\tilde{C}$ must be zero, i.e., $u_1^TB = 0, v_1^TC = 0$. Hence

$$F\begin{bmatrix} v_1 \\ 0 \end{bmatrix} = \sigma_1 \begin{bmatrix} u_1 \\ 0 \end{bmatrix}, \quad F^T\begin{bmatrix} u_1 \\ 0 \end{bmatrix} = \sigma_1 \begin{bmatrix} v_1 \\ 0 \end{bmatrix}.$$

Now we assume that the theorem is true for B with less than k columns and C with one column. We consider the case that B has k columns and C is a column vector. Partition $B = [B_1, b], D = [D_1, d]$, where B_1 and D_1 have $k-1$ columns. Denote

$$\tilde{F} = \begin{bmatrix} A & B_1 \\ C^T & D_1 \end{bmatrix}, \quad \tilde{f} = \begin{bmatrix} b \\ d \end{bmatrix}.$$

By the assumption of the theorem, σ_1 is the largest singular value of $\tilde{F}$. Therefore the induction assumption implies that $u_1^TB_1 = 0$ and $\left(\sigma_1, \begin{bmatrix} u_1 \\ 0 \end{bmatrix}, \begin{bmatrix} v_1 \\ 0 \end{bmatrix}\right)$ is the largest singular triplet of $\tilde{F}$. Since $F = [\tilde{F}, \tilde{f}]$, it follows that $[U_1^T, 0]\tilde{f} = U_1^Tb = 0$. Hence, $u_1^TB = 0$ and $\left(\sigma_1, \begin{bmatrix} u_1 \\ 0 \end{bmatrix}, \begin{bmatrix} v_1 \\ 0 \end{bmatrix}\right)$ is the largest singular triplet of F. We can similarly prove the same for C having more that one columns. □

The result of the above theorem in essentially says that we can not, in general, expect the left and right largest singular vectors of a matrix to have many zero entries. Therefore, in order to find sparse low-rank approximations, we need to relax the requirement on the reconstruction errors. For example, we may try to find low-rank approximations $X_kD_kY_k^T$ with reconstruction error

$$\|A - X_kD_kY_k^T\|_F^2 \leq (1+\tau)\|A - U_k\Sigma_kV_k^T\|_F^2$$

while requiring that X_k and Y_k has as few nonzero entries as possible. (Notice that $\text{best}_k(A) = U_k\Sigma_kV_k^T$.) We now consider the following optimization problem,

$$\min\left\{\text{nnz}(X_k) + \text{nnz}(Y_k) \;\middle|\; \|A - X_kD_kY_k^T\|_F^2 \leq (1+\tau)\|A - U_k\Sigma_kV_k^T\|_F^2\right\}. \tag{2.1}$$

We next further elaborate the above for the case of rank-one approximation. For given vectors x and y, it is easy to verify that

$$\begin{aligned}
\min_d \|A - xdy^T\|_F^2 &= \|A\|_F^2 - \left(\frac{x^TAy}{\|x\|\,\|y\|}\right)^2 \\
&= \|A - u_1\sigma_1v_1^T\|_F^2 + \sigma_1^2 - \left(\frac{x^TAy}{\|x\|\,\|y\|}\right)^2 \\
&= \left(1 + b_1\left(1 - \left(\frac{x^TAy}{\|x\|\,\|y\|\,\sigma_1}\right)^2\right)\right)\|A - u_1\sigma_1v_1^T\|_F^2,
\end{aligned}$$

where $b_1 = \sigma_1^2/(\sigma_2^2 + \cdots + \sigma_n^2)$. Therefore the parameter τ in (2.1) can be written as $\tau = b_1\xi$ and the optimization problem (2.1) becomes

$$\begin{aligned}
N(\xi) &= \min\left\{\text{nnz}(x) + \text{nnz}(y) \;\middle|\; \|A - xdy^T\|_F^2 \leq (1 + b_1\xi)\|A - u_1\sigma_1v_1^T\|_F^2\right\} \\
&= \min\left\{\text{nnz}(x) + \text{nnz}(y) \;\middle|\; \left(\frac{x^TAy}{\|x\|\,\|y\|\,\sigma_1}\right)^2 \geq 1 - \xi\right\}.
\end{aligned}$$

It is easy to see that $N(\xi)$ is the smallest sum of numbers of rows and columns of submatrices of A with 2-norm greater than or equal to $\sqrt{1-\xi}\|A\|$. The following theorem gives an upper bound of $N(\xi)$.

THEOREM 2.2. *Let $\{u, \sigma, v\}$ be the largest singular triplet of A, and let $\tilde{w}$ a vector consisting of the elements of u and v sorted such that*

$$|\tilde{w}_1| \geq |\tilde{w}_2| \geq \cdots \geq |\tilde{w}_{m+n}|.$$

Denote $\epsilon = \xi/(2 + 2\sqrt{1-\xi} + \xi)$. Then for any $0 \leq \xi < 1$,

$$N(\xi) \leq k(\epsilon) \equiv \min\left\{ k \;\middle|\; \tilde{w}_1^2 + \tilde{w}_2^2 \cdots + \tilde{w}_k^2 \geq 2(1-\epsilon) \right\}.$$

Proof. It can be verified that $\xi = 4\epsilon(1-2\epsilon)/(1-\epsilon)^2$, and $\epsilon < 1/3$ for $0 \leq \xi < 1$. Let $k = k(\epsilon)$, and denote $i(k)$ and $j(k)$, the number of the elements of u and v, respectively, in the vector $\tilde{w}(1:k)$. Set

$$x = P_1^T \begin{bmatrix} \tilde{u}(1:i(k)) \\ 0 \end{bmatrix}, \quad y = P_2^T \begin{bmatrix} \tilde{v}(1:j(k)) \\ 0 \end{bmatrix},$$

where P_1 and P_2 are the permutations determined by $\tilde{w} = P[u^T, v^T]^T$ satisfying $\tilde{u} = P_1 u$, $\tilde{v} = P_2 v$, and

$$|\tilde{u}_1| \geq \cdots \geq |\tilde{u}_m| \quad \text{and} \quad |\tilde{v}_1| \geq \cdots \geq |\tilde{v}_n|.$$

By Theorem 3.1 of [9], we obtain that

$$\|A - xdy^T\|_F^2 \leq (1 + b_1\epsilon)\|A - u_1\sigma_1 v_1^T\|_F^2.$$

Therefore $N(\xi) \leq k$, completing the proof. □

It is easy to see that ξ can be considered as a measure of accuracy of the rank-one approximation (as compared with that obtained by the largest singular triplet of A) while $N(\xi)$ a measure of its degree of sparsity. Obviously, $N(\xi)$ is a decreasing function.

3. Penalized optimization problems for sparse rank-one approximations. As is discussed in the previous section, we want to construct a low-rank approximation of A with low degree of sparsity and good (but not necessarily the best) approximation accuracy. A natural way is to consider the following optimal problem of minimizing a penalized cost function for a fixed η

$$\min_{x,y,\xi} \left\{ \eta\big(\mathbf{nnz}(x) + \mathbf{nnz}(y)\big) + \xi \right\} \tag{3.1}$$

with the inequality constrains $\xi \geq 0$ and

$$\|A - xdy^T\|_F^2 \leq (1 + b_1\xi)\|A - u_1\sigma_1 v_1^T\|_F^2,$$

where b_1 is defined in Section 1. However, the above problem is not straightforward to be solved and the goal of this section is to reduce it into a more amendable form.

First we consider how to remove the constrains, let x and y be the vectors that achieve the value of $N(\xi)$. Obviously, denoting

$$\xi^* = 1 - \left(\frac{x^T A y}{\|x\|\,\|y\|\,\sigma_1}\right)^2 \leq \xi$$

gives $N(\xi) = N(\xi^*)$. Therefore the accuracy measure ξ can be replaced by $1 - (x^T Ay/(\|x\| \|y\| \sigma_1))^2$, and the optimization problem (3.1) is reduced to

$$\min \left\{ \eta\big(\mathrm{nnz}(x) + \mathrm{nnz}(y)\big) + \left(1 - \left(\frac{x^T Ay}{\|x\| \|y\| \sigma_1}\right)^2\right)\right\},$$

or equivalently,

$$\min \left\{ \eta\big(\mathrm{nnz}(x) + \mathrm{nnz}(y)\big) - \left(\frac{x^T Ay}{\|x\| \|y\| \sigma_1}\right)^2\right\}.$$

More generally, we can consider the following optimization problem

$$N(\alpha, \beta, p) = \min \left\{ \alpha \mathrm{nnz}(x) + \beta \mathrm{nnz}(y) - \left(\frac{x^T Ay}{\|x\| \|y\| \sigma_1}\right)^p \right\} \tag{3.2}$$

by introducing penalty factors α and β and using an arbitrary p-norm instead of the 2-norm. The parameters α and β can be chosen by the following formula

$$\alpha = \frac{\lambda\mu}{(1-\lambda)m}, \quad \beta = \frac{\lambda(1-\mu)}{(1-\lambda)n}, \quad \lambda,\ \mu \in [0, 1)$$

because with this choice of α and β, the objective function in (3.2) becomes, after multiplying by the constant $(1 - \lambda)$,

$$\lambda \left(\mu \frac{\mathrm{nnz}(x)}{m} + (1-\mu)\frac{\mathrm{nnz}(y)}{n}\right) + (1-\lambda)\xi.$$

and the parameter λ and μ have the following interpretations: λ balances the degree of sparsity and the reconstruction error of the rank-one approximation while μ balances the degrees of sparsity of the left and right factors of the rank-one approximation xdy^T. In general, we choose $\mu = 1/2$ to keep the sparsity structure of the approximation *symmetric* if no other reasons dictate us to do otherwise.

Now the optimization problem (3.2) is still combinatorial in nature although it does not have any constrains. To this end we will derive an upper bound for $N(\alpha, \beta, p)$ which will then lead to an approximate and simplified formulation of (2.1). Let us define a matrix function

$$N_H(\alpha, \beta, p) = \min_{ij} \left\{ \alpha\, i + \beta\, j - \left(\frac{h_{ij}}{\sigma_1}\right)^p \right\}, \tag{3.3}$$

where $H = (h_{ij})$. It is easy to see that if let $S = (s_{ij})$ with s_{ij} the largest singular value of submatrices consisting of the intersection of i rows and j columns of A, then

$$N(\alpha, \beta, p) = N_S(\alpha, \beta, p).$$

Therefore a component-wise lower bound of S can lead to an upper bound of $N(\alpha, \beta, p)$ and then an approximate variation of (3.2). In the next we will derive such lower bounds.

Let $\{u, \sigma, v\}$ be the largest singular triplet of matrix A. Denote by $R = (r_{ij})$ the Rayleigh quotient

$$r_{ij} = \max_{|I|=i,\,|J|=j} \left| \frac{u(I)^T A(I, J) v(J)}{\|u(I)\| \|v(J)\|} \right|.$$

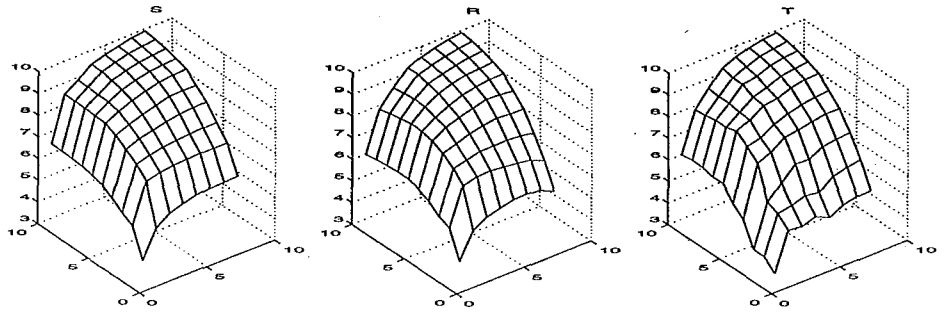


FIG. 3.1. *Plots of the matrices S, R, and T.*

Obviously, $s_{ij} \geq r_{ij}$ and $N_S(\alpha,\beta,p) \leq N_R(\alpha,\beta,p)$. Obviously, (3.3) with $H = R$ is also a combinatorial optimization problem. To circumenvent this, we impose constrains on the vectors x and y as we did in [9]. Let $\tilde{u}$ and $\tilde{v}$ be the sorted versions of u and v, respectively, such that

$$|\tilde{u}_1| \geq \cdots \geq |\tilde{u}_m|, \quad \text{and} \quad |\tilde{v}_1| \geq \cdots \geq |\tilde{v}_n|,$$

and let I and J be the index vectors satisfying $\tilde{u} = u(I)$ and $\tilde{v} = v(J)$. Denote by $T = (t_{ij})$ the Rayleigh quotient of the truncated vectors of $\tilde{u}$ and $\tilde{v}$

$$t_{ij} = \left| \frac{u(I(1:i))^T A(I(1:i), J(1:j)) v(J(1:j))}{\|u(I(1:i))\| \, \|v(J(1:j))\|} \right| \leq r_{ij}.$$

Since $s_{ij} \geq r_{ij} \geq t_{ij}$, we have

$$N_S(\alpha,\beta,p) \leq N_R(\alpha,\beta,p) \leq N_T(\alpha,\beta,p).$$

Of course, problem (3.3) with different H will give different optimal solutions, i.e., the solution of the simple variation (3.3) with $H = T$ may differ from that for $H = S$, the solution of (3.2). However, the difference between the solutions is not large because the matrix T is similar to S. To show this, let us consider a small numerical example. (We have also tested other small matrices and found similar behavior.[1])

EXAMPLE 1. Let $m = 10$, and $n = 8$, and $l = \min(m,n)$. We construct A as (using the notation of MATLAB)

```
[U, r] = qr(rand(m,l));
[V, r] = qr(rand(n,l));
A = U*diag(10*rand(l,1))*V';
```

First, we compare the three matrices S, R, and T. Figure 3.1 plots the matrices S, R, and T. In general, s_{ij}, r_{ij}, and t_{ij} are close to each other if i and$j$ are not small, i.e., large discrepancy in s_{ij}, r_{ij} and t_{ij} may occur only when the indexes i and j are small. Below we list the average values and the maximums of the relative errors between s_{ij}, r_{ij}, and t_{ij}. Note that the maximums occur generally with small indexes i and j.

[1] Only small matrices are used in our examples since computing S and R involves exhaustive search.

	average	max
$(s_{ij} - r_{ij})/s_{ij}$	2.5809e-02	1.0719e-01
$(r_{ij} - t_{ij})/r_{ij}$	4.3807e-02	2.8485e-01

Second, we compute the indexes i_H and j_H of the optimal solution of (3.3) with different choices of H and corresponding values h_{ij}. for $p = 1, 2$, respectively. Below we list the computed results.

	$p=1$			$p=2$		
H	i	j	h_{ij}	i	j	h_{ij}
S	6	3	8.72648	7	6	9.50874
R	5	3	8.35768	8	6	9.52802
T	6	4	8.79437	8	6	9.51918

The example furthermore shows that larger total number of nonzeros generally implies higher accuracy of the approximation determined by the optimal problems.

REMARK. Let (i_S, j_S), (i_R, j_R), and (i_T, j_T) be the integers which achieve the minimums $N_S(\lambda, \mu, p)$, $N_R(\lambda, \mu, p)$, and $N_T(\lambda, \mu, p)$, respectively. It seems that in most cases, we can expect

$$i_S + j_S \leq i_R + j_R \leq i_T + j_T, \quad \text{and} \quad s_{i_S,j_S} \leq r_{i_R,j_R} \leq t_{i_T,j_T}.$$

Unfortunately, the above assertion is difficult to verify in general. However, we can prove the following weaker form

$$s^p_{i_S,j_S} \leq r^p_{i_R,j_R} + c_{SR} s^p_{i_S,j_S} - \sigma_1^p \left(\alpha(i_R - i_S) + \beta(j_R - j_S)\right)$$

provided $i_S + j_S \leq i_R + j_R$, where $c_{SR} = \max_{i,j}(s^p_{ij} - r^p_{ij})/s^p_{ij}$.

It is easy to see that the above follows straightforwardly from the definitions of $N_R(\lambda, \mu, p)$, $N_S(\lambda, \mu, p)$ and the following result. A similar result can also be proved for r_{i_R,j_R} and t_{i_T,j_T}.

PROPOSITION 3.1. *Let $g(x)$ and $h(x)$ satisfy $(1 - c)g(x) \leq h(x) \leq g(x)$, where $g(x) \geq 0$ and $0 \leq c < 1$. Define x_1 and x_2 such that*

$$f(x_1) - g(x_1) = \min\{f(x) - g(x)\}, \quad f(x_2) - h(x_2) = \min\{f(x) - h(x)\}.$$

If $x_1 \leq x_2$, then

$$g(x_1) \leq h(x_2) + cg(x_1) - (f(x_2) - f(x_1)).$$

Proof. Let $m_1 = f(x_1) - g(x_1)$ and $m_2 = f(x_2) - h(x_2)$. By the assumption of the proposition

$$f(x) - h(x) \leq f(x) - (1 - c)g(x) = f(x) - g(x) + cg(x).$$

It follows that

$$m_2 \leq f(x_1) - h(x_1) \leq f(x_1) - g(x_1) + cg(x_1).$$

Therefore, $m_1 - m_2 \leq cg(x_1)$, and

$$g(x_1) - h(x_2) = m_2 - m_1 - (f(x_2) - f(x_1)) \leq cg(x_1) - (f(x_2) - f(x_1)),$$

completing the proof. □

Reduction to 1-D form. Compared with the computations of s_{ij} and r_{ij}, the computation of t_{ij} is much easier. However, the corresponding optimization problem (3.3) with $H = T$ still involves optimization among a 2-D index set $\{i, j\}, i = 1, \ldots, m, j = 1, \ldots, n$. What we do next is to further simplify it into an optimization problem that only involves an 1-D index set. Our strategy is to develop a continuous version of the optimization problem (3.3) with $H = T$ by introducing some smooth interpolating functions. Next we will try to find equivalent formulation of this continuous optimization problem, and then transform it back into discrete format.

To this end, let $\tilde{u}$ and $\tilde{v}$ be written as $\tilde{u} = P_1 u$ and $\tilde{v} = P_2 v$. Motivated by Theorem 2.2, we partition

$$\tilde{u} = \begin{bmatrix} \tilde{u}(1 : k_u(\epsilon)) \\ \tilde{u}(k_u(\epsilon) : m) \end{bmatrix}, \quad \tilde{v} = \begin{bmatrix} \tilde{v}(1 : k_v(\epsilon)) \\ \tilde{v}(k_v(\epsilon) : n) \end{bmatrix},$$

$$k(\epsilon) = \min \left\{ k \;\middle|\; \sum_{i=1}^{k} \tilde{w}_i^2 \geq 2(1 - \epsilon^2) \right\},$$

where $k_u(\epsilon)$ is the number of elements of u, in $\tilde{w}(1 : k(\epsilon))$ and $k_v(\epsilon)$ is the number of elements of v in $\tilde{w}(1 : k(\epsilon))$. Now

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

Thus, problem (3.3) with $H = T$ and $p = 1$ can be rewritten equivalently as

$$\min_{\epsilon} \Big\{ \alpha k_u(\epsilon) + \beta k_v(\epsilon) - \big|h(k_u(\epsilon), k_v(\epsilon))\big| \Big\},$$

where $h(k_u(\epsilon), k_v(\epsilon)) = x^T A y / (\|x\| \, \|y\| \, \sigma_1)$. We obtain an approximate continuous optimization problem

$$F(\alpha, \beta) = \min_{\epsilon} \Big\{ \alpha\phi(\epsilon) + \beta\psi(\epsilon) - \omega(\epsilon) \Big\}, \tag{3.4}$$

if we require that the function $\phi(\epsilon)$, $\psi(\epsilon)$, and $\omega(\epsilon)$ be approximations of the piece-constant functions $k_u(\epsilon)$, $k_v(\epsilon)$, and $|h(k_u(\epsilon), k_v(\epsilon))|$, respectively,

$$\phi(\epsilon) \approx k_u(\epsilon), \quad \psi(\epsilon) \approx k_v(\epsilon), \quad \omega(\epsilon) \approx \big|h(k_u(\epsilon), k_v(\epsilon))\big|.$$

It is easy to verify that the optimal ϵ satisfies

$$\alpha\phi'(\epsilon) + \beta\psi'(\epsilon) - \omega'(\epsilon) = 0.$$

if ϕ, ψ, and ω are differentiable.

Now we find *smooth* functions that interpolate $k_u(\epsilon)$ and $k_v(\epsilon)$. Specifically, we choose functions $\phi(\epsilon)$ and $\psi(\epsilon)$ defined by the following integral equations,

$$\int_0^{\phi(\epsilon)} x(t)dt = 1 - \epsilon, \quad \int_0^{\psi(\epsilon)} y(t)dt = 1 - \epsilon,$$

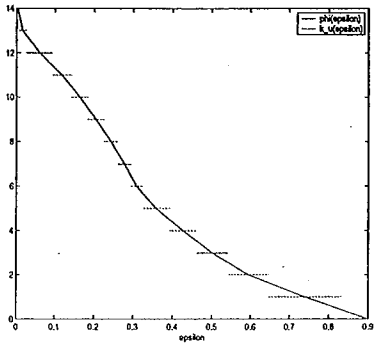


FIG. 3.2. *Plot for the piece-constant function $k_u(\epsilon)$ and its interpolation $\phi(\epsilon)$.*

where $x(t)$ and $y(t)$ interpolate $\{\tilde{u}_i^2\}$ and $\{\tilde{v}_j^2\}$, respectively, see Figure 3.2 for an illustration. Because of the above definitions,

$$\frac{1}{2} \geq |h(k_u(\epsilon), k_v(\epsilon))| \geq \frac{1}{2} - \frac{\epsilon}{1-\epsilon},$$

it makes sense to choose

$$\omega(\epsilon) = \frac{1}{2} - \frac{c\epsilon}{1-\epsilon}, \tag{3.5}$$

where $c \in (0,\ 1]$ is a constant to be determined.

If $x(t)$ and $y(t)$ are continuous, then $\phi(\epsilon)$ and $\psi(\epsilon)$ are differentiable and

$$\phi'(\epsilon) = -\frac{1}{x(\phi(\epsilon))} \approx -\frac{1}{\tilde{u}^2_{k_u(\epsilon)}}, \quad \psi'(\epsilon) = -\frac{1}{y(\psi(\epsilon))} \approx -\frac{1}{\tilde{v}^2_{k_v(\epsilon)}},$$

since $x(\phi(\epsilon)) \approx \tilde{u}^2_{k_u(\epsilon)}$ and $x(\psi(\epsilon)) \approx \tilde{v}^2_{k_v(\epsilon)}$. On the other hand,

$$\omega'(\epsilon) = -\frac{c}{(1-\epsilon)^2},$$

the optimal ϵ approximately satisfies

$$\frac{\alpha}{\tilde{u}^2_{k_u(\epsilon)}} + \frac{\beta}{v^2_{k_v(\epsilon)}} - \frac{c}{(1-\epsilon)^2} = 0,$$

or equivalently,

$$1 - \epsilon = \left(\frac{\alpha}{c} \tilde{u}^{-2}_{k_u(\epsilon)} + \frac{\beta}{c} \tilde{v}^{-2}_{k_v(\epsilon)} \right)^{-1/2}.$$

Note that

$$\frac{1}{2} \left(\sum_{i=1}^{k_u(\epsilon)} \tilde{u}_i^2 + \sum_{j=1}^{k_v(\epsilon)} \tilde{v}_j^2 \right) = 1 - \epsilon^2,$$

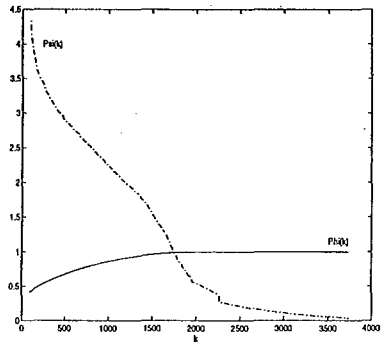


FIG. 3.3. *Plot for the discrete curves $\Phi(k)$ (solid line) and $\Psi(k)$ (dashdot line).*

we conclude that the optimal ϵ approximately satisfies

$$\frac{1}{2}\left(\sum_{i=1}^{k_u(\epsilon)}\tilde{u}_i^2+\sum_{j=1}^{k_v(\epsilon)}\tilde{v}_j^2\right)=\left(\frac{\alpha}{c}\tilde{u}_{i(k)}^{-2}+\frac{\beta}{c}\tilde{v}_{j(k)}^{-2}\right)^{-1/2}.$$

Now we can transform back to the following discrete optimization problem,

$$\min_k\left|\frac{1}{2}\left(\sum_{l=1}^{i(k)}\tilde{u}_l^2+\sum_{l=1}^{j(k)}\tilde{v}_l^2\right)-\left(\frac{\alpha}{c}\tilde{u}_{i(k)}^{-2}+\frac{\beta}{c}\tilde{v}_{j(k)}^{-2}\right)^{-1/2}\right|, \tag{3.6}$$

where the integers $i(k)$ and $j(k)$ are determined by k and satisfy the following

$$\begin{cases} i(k)+j(k) &= k \\ \min\left\{\tilde{u}_{i(k)}^2,\tilde{v}_{j(k)}^2\right\} &= \tilde{w}_k^2 \end{cases} \tag{3.7}$$

i.e., $i(k)=k_u(\epsilon)$, $j(k)=k_v(\epsilon)$, provided $k=k(\epsilon)$. Differing from the discrete problem (3.3), (3.6) is a 1-D problem and is easy to solve.

To make our following discussions concrete, we introduce the following functions,

$$\Phi(k)=\frac{1}{2}\left(\sum_{l=1}^{i}\tilde{u}_l^2+\sum_{l=1}^{j}\tilde{v}_l^2\right),\quad \Psi(k)=\left(\frac{\alpha}{c}\tilde{u}_i^{-2}+\frac{\beta}{c}\tilde{v}_j^{-2}\right)^{-1/2}.$$

with i, j, and k satisfying (3.7). It should be pointed out that the indexes $i=i(k)$ and $j=j(k)$ may not be uniquely determined by k if $|\tilde{w}_k|=|\tilde{w}_{k+1}|$. We will have a detailed discussion about this in the next section. Ignoring, for the moment, the possibility of being multi-valued, we can easily see that the discrete function $\Psi(k)$ is decreasing while $\Phi(k)$ is increasing. Figure 3.3 plots for the graphs of $\Psi(k)$ and $\Phi(k)$ for a matrix of order 2331×1398. In the next section, we will propose a discrete globally convergent method to solve the minimization problem (3.6).

4. Discrete secant iteration. Based upon the monotonicity of the discrete functions $\Phi(k)$ and $\Psi(k)$, we use the following secant iteration for the optimization problem (3.6).

$$k_{l+1}=k_l+\left\lfloor\frac{\Psi_l-\Phi_l}{\delta\Phi_l-\delta\Psi_l}\right\rfloor,$$

where $\delta\Phi_l$ and $\delta\Psi_l$ are the current secants for Φ and Ψ, respectively, and $\lfloor f \rfloor$ is the floor function giving the largest integer no greater than f. Obviously, it is guaranteed that the solution is unique. However, there are some computational issues that need to be discussed before we present our discrete secant method based on the above iteration. We first need to investigate whether the cost function in (3.6) is well-defined or not.

If $|\tilde{w}_k| > |\tilde{w}_{k+1}|$, $i(k)$ and $j(k)$ are uniquely determined by k; $i(k)$ is the number of the u-components of subvector $\tilde{w}(1:k)$ and $j(k)$ is the number of the v-components of $\tilde{w}(1:k)$. In this case, both $\Phi(k)$ and $\Psi(k)$ are well-defined. If $|\tilde{w}_k| = |\tilde{w}_{k+1}|$, $i(k)$ and $j(k)$ are ill-defined. Therefore $\Psi(k)$ may have three different values depending on the choices of $i(k)$ and $j(k)$, see Figure 4.1 for a detailed illustration. However, $\Phi(k)$ is unique regardless whether $|\tilde{w}_k|$ and $|\tilde{w}_{k+1}|$ are equal or not. We now discuss several important computation details.

w-CONSTANT INTERVALS. We call $(a,\, b]$ a w-constant interval if a and b satisfy

$$|\tilde{w}_a| > |\tilde{w}_{a+1}| = \cdots = |\tilde{w}_b| > |\tilde{w}_{b+1}|.$$

For any integer $k \in (a,\, b]$, $i(k)$ and $j(k)$ can be any integers in the u-constant interval $(i(a),\, i(b)]$ and v-constant interval $(j(a),\, j(b)]$, respectively,

$$i(k) \in (i(a),\, i(b)], \quad j(k) \in (j(a),\, j(b)].$$

However, we do not need to pay attention to the w-constant interval if it does not contain the optimal solution k^*.

BRACKETING INTERVALS. A w-constant interval is called a bracketing interval if the optimal $k^* \in [a,\, b]$. Since $\Psi(k)$ is ill-defined in $(a,\, b)$, the optimal k^* should satisfy that

$$\min_{i(a)\le i\le i(b), j(a)\le j\le j(b)} \left|\Phi(i+j) - \Psi(i+j)\right|.$$

It is easy to see that $\Phi(k)$ is uniquely defined and is a linear function in the interval $(a,\, b)$ while $\Psi(k)$ may have multiple values depending on i and j. Figure 4.1 plots for the graph of $\Phi(k)$ many possible graphs for $\Psi(k)$ in a bracketing interval.

The existence of bracketing intervals make the problem (3.6) more complicated. Fortunately, bracketing intervals seldom occur and the length of the occurred bracketing interval is general very small.

CHOOSING THE SECANTS $\delta\Phi_l$ AND $\delta\Psi_l$. There are many ways to choose the secants $\delta\Phi_l$ and $\delta\Psi_l$ which are to be used in the next iteration. For the initial $\delta\Phi_0$ and $\delta\Psi_0$, we set

$$\delta\Phi_0 = (\Phi(k_0+d) - \Phi(k_0))/d, \quad \text{and} \quad \delta\Psi_0 = (\Psi(k_0+d) - \Psi(k_0))/d.$$

with $d \le \frac{1}{2}(m+n-k_0)$, for example, $d = \min(100, [\frac{1}{2}(m+n-k_0)])$. In general, one can choose $\delta\Phi_l = (\Phi_l - \Phi_{l-1})/(k_l - k_{l-1})$ and $\delta\Psi_l = (\Psi_l - \Phi_{l-1})/(k_l - k_{l-1})$. However a better way is to compute $\delta\Phi_l$ and $\delta\Psi_l$ by

$$\delta\Phi_l = \frac{\Phi_{\max} - \Phi_{\min}}{k_{\max} - k_{\min}} \quad \text{and} \quad \delta\Psi_l = \frac{\Psi_{\max} - \Phi_{\min}}{k_{\max} - k_{\min}},$$

if we know that $k^* \in [k_{\min},\, k_{\max}]$, where

$$\Phi_{\min} = \Phi(k_{\min}), \quad \Phi_{\max} = \Phi(k_{\max}),$$
$$\Psi_{\min} = \Psi(k_{\min}), \quad \Psi_{\max} = \Psi(k_{\max}).$$

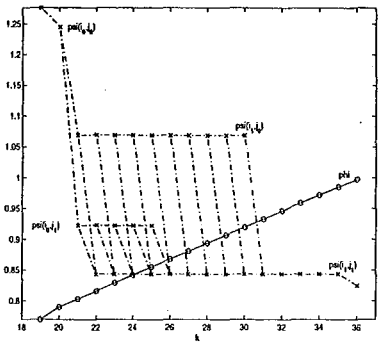


FIG. 4.1. *The curves $\Phi(k)$ (solid line) and the possible curve $\Psi(k)$ (deshed lines) with $\lambda = \mu = 0.5$.*

Initially, we set $k_{\min} = k_0$.

AVOIDING INFINITE LOOP. To avoid infinite loop of the secant iteration, we need to slightly modify k_{l+1} at the l-th iteration so that

$$\begin{array}{rcccll} k_l + 1 & \leq & k_{l+1} & \leq & k_{\max} - 1, & \text{if } k_l \leq k^*, \quad \text{or} \\ k_{\min} + 1 & \leq & k_{l+1} & \leq & k_l - 1, & \text{if } k_l \geq k^*. \end{array}$$

Now we are ready to present an algorithm for solving the discrete optimization problem (3.6).

Algorithm DSI (DISCRETE SECANT ITERATION).

1. [Initialization]

1.1 Sort $\tilde{w} = Pw$, $\tilde{w}_k \leftarrow \tilde{w}_k^2$, $\tilde{u}_i \leftarrow \tilde{u}_i^2$, $\tilde{v}_j \leftarrow \tilde{v}_j^2$.

1.2 Determine the smallest k_0 such that $i(k_0) \geq 1$, $j(k_0) \geq 1$, and compute $\Phi_0 = \Phi(k_0)$ and $\Psi_0 = \Psi(k_0)$.

1.3 Check convergence. If $\Psi_0 \leq \Phi_0$, stop, otherwise determine the secant $\delta\Phi_0$ and $\delta\Psi_0$ for next secant iteration.

2. For $l = 1, 2, \cdots$, until convergence,

2.1 One Newton-like iteration. $k = k_l + \left\lceil \frac{\Psi_l - \Phi_l}{\delta\Phi_l - \delta\Psi_l} \right\rceil$.

2.2 Determine the w-constant interval $(a, b]$ of which covers k, and compute

$$\Psi_a = \Psi(a),\ \Psi_b = \Psi(b),\ \Phi_a = \Phi(a),\ \Phi_b = \Phi(b).$$

2.3 If $[a, b]$ does not contain the optimal k^*, compute k_{l+1} by

$$k_{l+1} = \begin{cases} a & \text{if } \Psi_a \leq \Phi_a \\ b & \text{if } \Psi_b > \Phi_b \end{cases},$$

and determine the secant $\delta\Phi_{l+1}$ and $\delta\Psi_{l+1}$ for next secant iteration, otherwise turn to **step 3**.

3. Compute $k^* \in [a, b]$.

TABLE 5.1
Average of the discrete secant iterations.

γ	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
ash958	9.1	8.1	8.2	8.0	8.1	7.4	7.6	7.2	7.2	6.8
illc1033	9.2	9.2	8.3	8.1	7.6	7.1	7.4	7.2	7.1	6.5
cisi	12.7	10.4	10.0	9.9	9.4	9.3	9.1	8.8	8.6	8.8
cacm	10.8	10.5	9.8	10.0	9.5	9.4	9.2	9.1	9.0	8.9
med	13.1	11.6	10.8	10.4	9.9	9.6	9.5	9.2	9.2	8.8
npl	12.8	11.8	11.5	11.6	11.1	11.8	10.9	10.7	10.8	10.8
orsirr2	9.7	9.3	8.8	8.6	8.3	8.1	7.7	7.5	7.7	7.4
e20r1000	11.8	11.6	10.8	10.7	10.5	10.2	10.1	9.7	9.4	9.5

After several iterations, if l is the first integer such that if $\Psi_l < \Phi_l$, we then set $k_{\max} = k_l$ and the interval $[k_{\min}, k_{\max}]$ which contains the optimal k^* can be made smaller and smaller. Otherwise the sequence $\{k_l\}$ converges monotonically to k^*. Therefore the above algorithm is guaranteed to converge.

5. Numerical Experiments. In this section, we will present several numerical experiments to illustrate the discrete secant method we proposed. The test matrices used are the same as in [9]. (The reader is referred to [9] for detailed descriptions.) In all of the numerical tests we use 4 Lanczos bidiagonalization iterations to compute the approximate largest singular vectors of the deflated matrices $A_i = A - X_i D_i Y_i^T$ at each iteration step. The penalty factors α and β in (3.6) are chosen as follows,

$$\alpha = \frac{\gamma}{m}, \quad \beta = \frac{\gamma}{n}, \quad \text{and} \quad \gamma = \frac{\lambda}{2(1-\lambda)},$$

i.e., $\mu = 1/2$. The parameter λ is chosen as following.

$$\lambda = \texttt{(0.1:0.1:1.0)./(1.1:0.1:2.0)}$$
$$\approx \texttt{[.09 .17 .23 .26 .33 .38 .41 .44 .47 .50]},$$

which gives

$$\gamma = \texttt{0.05:0.05:0.5}.$$

For simplicity, we choose $c = 1$ in (3.5).

First we look at the speed of convergence of the Discrete Secant Iteration (DSI) algorithm. DSI is globally convergent. In general, it needs about 10 iterations to obtain the optimal k^*. In Table 5.1 we list the average of the number of iterations for each matrix and γ.

Although it is possible that the optimal integer k^* is in a w-constant interval which will lead to a little bit complex case, bracketing intervals seldom occur and the length $b - a$ of the occurred bracketing intervals are small in general. Among all the eight tested matrices, bracketing intervals never occurred for matrices `illc1033`, `npl`, `orsirr2`, and `e20r1000` and all the choice of γ = `0.05:0.05:0.50` while for the other four matrices the largest length of bracketing intervals are generally 2. See the table listed below, where the integer p in the form $p(q)$ is the maximal length of the q bracketing intervals occurred.

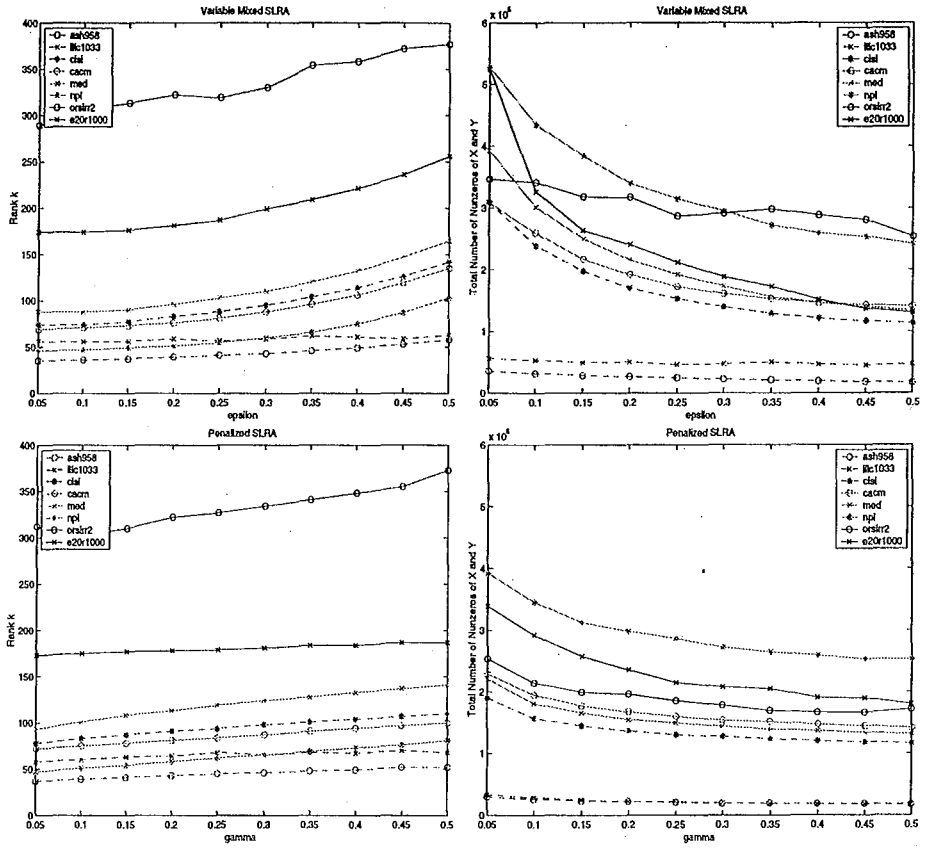


FIG. 5.1. *Plots for ranks (left) and numbers of nonzeros of X_k and Y_k (right) vs starting epsilon for the variable tolerance, mixed sorting approach (top) and the penalized approach (bottom).*

γ	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
ash958				2(1)		2(1)	2(1)		2(1)	
cisi		2(1)								
cacm	2(2)		2(2)	4(3)	2(3)	2(2)	2(4)	2(1)	2(1)	2(2)
med	2(1)								2(1)	

We now compare the ranks and the numbers of the nonzeros of the factors X_k and Y_k computed by the mixed sorting approach SLRA with variable tolerance discussed in [9] and the penalized method. The initial parameter ϵ is, as we did in [9], that

$$\epsilon = 0.05 : 0.05 : 0.5.$$

Figure 5.1 plots the ranks (left) and the total number of nonzeros of X_k and Y_k (right) computed by the mixed sorting approach SLRA with variable tolerance ϵ (top) and the penalized schemes (bottom). The numerical results show that the penalized method is more "robust" than the mixed sorting SLRA because of that the ranks and the numbers of nonzeros of the factors X_k and Y_k computed by the penalized method are not sensitive depending on the choice of the parameter λ.

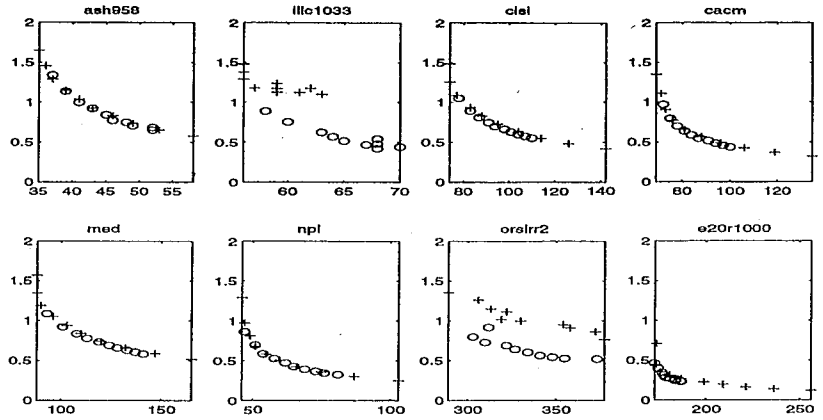


FIG. 5.2. *Plots for the relative density of the computed X_k and Y_k by SLRA with variable tolerance and mixed sorting approach (+-dots) and the penalized method (o-dots).*

Though we choose λ such that the corresponding parameter γ are the same as ϵ, the computed ranks and the numbers of nonzeros of X_k and Y_k are not comparable because the ranks and/or the numbers of nonzeros may not be equal, respectively, even if $\gamma = \epsilon$. To show the efficiency of the penalized method, let us define the relative density function f by

$$f(k) = \frac{\mathrm{nnz}(X_k)}{k * m} + \frac{\mathrm{nnz}(Y_k)}{k * n}.$$

In Figure 5.2, we plot, respectively, the relative density function corresponding the mixed sorting SLRA (+-dots) and the penalized method (o-dots). In general, the function f corresponding to the penalized method locate left and blow that corresponding to the mixed sorting SLRA. That means penalized method generally produce an approximation with lower rank and more sparse factors than the mixed sorting SLRA.

Finally, we point out that for those matrices which are close to rank-deficient the penalized method may produce approximations with large change in rank for sometime special choice of γ. See, for example, the boxed numbers in the following table for the test matrix `watson4`. We list, in the table, the ranks of the computed approximations $B_k = X_k D_k Y_k^T$ using the penalized SLRA with different choice of γ and 4 Lanczos bidiagonalization iterations for computing approximately the largest singular vectors at each step. All the approximations achieve the same reconstruction error.

γ	.05	.10	.15	.20	.25	.30	.35	.40	.45	.50
k	138	191	159	224	172	169	177	182	188	417

The reason is that the computed singular vectors u_i and v_i of $A_{i-1} = A - X_{i-1} D_{i-1} Y_{i-1}^T$ at some iteration step of SLRA are far away from the exact ones, which leads to that the computed d_i may be much smaller than the largest singular value of the deflated matrix A_i and the rank k is increased in order to achieve the same accuracy because

$$\|A - X_i D_i Y_i^T\|_F^2 = \|A\|_F^2 - (d_1^2 + \cdots + d_i^2).$$

Fortunately, increasing the iteration number a little bit more used for the Lanczos bidiagonalization will reduce the sensitiveness. For this test matrix `watson4`, if we use

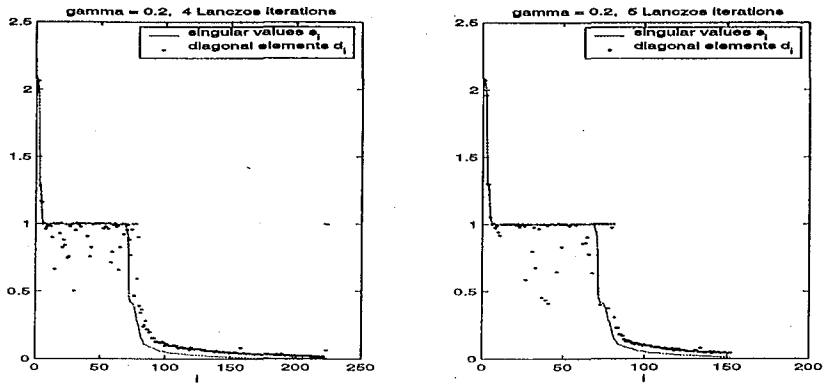


FIG. 5.3. *Plots for the singular values (solid line) of the matrix* `watson4` *and the computed diagonal elements of* D_k *(dot line) with* $\gamma = 0.2$*, using 4 Lanczos iterations (left) or 5 Lanczos iterations (right) for computing the largest singular vectors.*

5 Lanczos bidiagonalizations, the ranks are reduced to 150, 152, and 178 corresponding to $\gamma = 0.1$, 0.2, and 0.5, respectively. Figure 5.3 plots the singular values (solid line) of the matrix `watson4` and the computed diagonal elements of D_k with $\gamma = 0.2$.

6. Concluding Remarks. Computing low-rank approximations of matrices is a very important matrix computation problem that has many applications in information retrieval and data mining. The large sizes and sparsity properties of the matrices arising from these applications entail that we find low-rank approximations that themselves also possess some sparsity properties. We continue our research on this problem following the general framework proposed in [9]: we formulate the sparse low-rank approximation problem as a penalized optimization problem, and derive simpler form of the optimization problem that is more amendable for numerical computations. In particular, we manage to avoid exhaustive combinatorial search to solve the penalized optimization problem. Numerical experiments show that the penalized methods are more robust and produce approximations with lower ranks and more sparse factor.

REFERENCES

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

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

[3] 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.

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

[5] 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.) To appear in *SIAM Journal on Scientific Computing*, 1999.

[6] 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.

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

[8] 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.

[9] Z. Zhang, H. Zha, and H. Simon. Low-rank Approximations with Sparse Factors I: Basic Algorithms and Error Analysis. CSE Tech. Report CSE-99-009, Department of Computer

Science and Engineering, Pennsylvania State University, 1999. (Also LBNL Tech. Report LBNL-44003.)