AN ALTERNATIVE ALGORITHM FOR REFINEMENT OF ULV DECOMPOSITIONS *

JESSE L. BARLOW , HASAN ERBAY †, AND IVAN SLAPNIČAR ‡

Abstract. The ULV decomposition (ULVD) is an important member of a class of rank-revealing two-sided orthogonal decompositions used to approximate the singular value decomposition (SVD). It is useful in the applications of the SVD such as principal components where we are interested in approximating a matrix by one of lower rank. It can be updated and downdated much more quickly than an SVD.

In many instances, the ULVD must be refined to improve the approximation it gives for the important right singular subspaces or to improve the matrix approximation. Present algorithms to perform this refinement require $O(mn)$ operations if the rank of the matrix is k where k is very close to 0 or n, but these algorithms require $O(mn^2)$ operations otherwise. Presented here is an alternative refinement algorithm that requires $O(mn)$ operations no matter what the rank is. Our tests show that this new refinement algorithm produces similar improvement in matrix approximation and in the subspaces. We also propose slight improvements on the error bounds on subspaces and singular values computed by the ULVD.

1. Introduction. The singular value decomposition (SVD) of a matrix $X \in \mathbf{R}^{m\times n}, m \geq n$, is written

$$X = Y\Sigma W^T \tag{1.1}$$

where $W \in \mathbf{R}^{n\times n}$ is orthogonal and $Y \in \mathbf{R}^{m\times n}$ is *left orthogonal*, that is, $Y^TY = I_n$, and

$$\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n) \in \mathbf{R}^{n\times n}, \quad \sigma_1 \geq \ldots \geq \sigma_n. \tag{1.2}$$

It contains useful information about a linear operator including rank, important subspaces, and conditioning. It also gives us the ability to compute low rank approximations.

Some of the most important applications of the SVD include solving ill-conditioned least squares problems [5], total least squares problems [24], subspace tracking, and isolating principal components [17].

In these applications, our interest in the SVD is to write X in the form

$$X = Y_1\Sigma_1 W_1^T + Y_2\Sigma_2 W_2^T \tag{1.3}$$

where

$$\Sigma_1 = \mathrm{diag}(\sigma_1, \ldots, \sigma_k), \quad \Sigma_2 = \mathrm{diag}(\sigma_{k+1}, \ldots, \sigma_n)$$

and Y_1, Y_2 , W_1 and W_2 are the corresponding matrices of left and right singular vectors. The other information in the SVD is not needed. Moreover, accurate representations of Range(W_1) and/or Range(W_2) are often satisfactory. In principal component analysis, we only need a good low rank approximation of X.

We use $\|\cdot\|$ to denote the two norm (matrix and vector), and $\|\cdot\|_F$ to denote the Frobenius norm. The notation $\sigma_k(X)$ denotes the kth largest singular value of X.

*Research fund by the National Science Foundation under contract no. CCR–9732081.

†The Pennsylvania State University, Department of Computer Science and Engineering, University Park, PA 16802-6106 USA (barlow,erbay@cse.psu.edu).

‡Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, R. Boškovića b.b., 21000 Split, Croatia (ivan.slapnicar@fesb.hr)

Complete Orthogonal Decompositions (CODs) can compute a approximation to the dichotomy in (1.3) slightly faster than an SVD. However, if the decomposition must be modified by adding rows (updating) and/or deleting rows (downdating), then the SVD can be computationally expensive [15] whereas CODs may be updated and downdated in $O(mn)$ flops [10, 3, 26, 22, 23]. Our algorithms for construction and refinement lead to a form that can be updated by any of these procedures.

The CODs have the form

$$X = UCV^T \tag{1.4}$$

where C is a triangular matrix partitioned according to

$$\begin{array}{c} \\ C = \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} C_1 & C_2 \end{pmatrix} \end{array}, \quad \|C_2\| \approx \|\Sigma_2\| = \sigma_{k+1}$$

and $U \in \mathbf{R}^{m\times n}$ is left orthogonal, $V \in \mathbf{R}^{n\times n}$ is orthogonal.

The first such decompositions were proposed independently in [11] and in [16]. These used maximal column pivoting [6]. More recent enhancements, such as the URV and ULV decompositions, discussed in [22, 23], made use of condition estimators. This paper uses the ULV decomposition (ULVD) which we define in the form (1.4) where, for a fixed integer k and tolerance ϵ, we have

$$\begin{array}{c} \\ U = \end{array} \begin{array}{c} \begin{array}{cc} k & m-k \end{array} \\ \begin{pmatrix} U_1 & U_2 \end{pmatrix} \end{array}, \quad \begin{array}{c} \\ V = \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} V_1 & V_2 \end{pmatrix} \end{array}, \tag{1.5}$$

$$\begin{array}{c} \\ C = \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{array}{c} k \\ n-k \end{array} \begin{pmatrix} L & 0 \\ F & G \end{pmatrix} \end{array} \tag{1.6}$$

and

$$\|L^{-1}\|^{-1} \le \epsilon, \quad \|\begin{pmatrix} F & G \end{pmatrix}\| \le \Phi(n)\epsilon. \tag{1.7}$$

Here $\Phi(n)$ is modestly growing function of n, say, $\sqrt{n}$. This formulation is a variant of one in [26, 9, 10, 4, 8]. We refer to k are the ϵ-psuedorank of X, corresponding to a definition used by Lawson and Hanson [18, p.77].

The matrix V yields approximations to the two right singular subspaces, Range(W_1) and Range(W_2), the matrix U yields approximations to the two left singular subspaces Range(Y_1) and Range(Y_2). The accuracy of these approximations and approximations to the singular values of C are discussed in §2. The results given there are slight enhancements of those by Mathias and Stewart [19] and Fierro and Bunch [12].

In this work, we give improvements to a refinement procedure for the ULVD in [19]. Often after an update or downdate, the conditions (1.7) are not satisfied, and or $\|\begin{pmatrix} F & G \end{pmatrix}\|_F$ is not small enough. Thus procedures which make $\|\begin{pmatrix} F & G \end{pmatrix}\|_F$ smaller are necessary. Procedures are given in [19, 12], but these require $O(mn^2)$ flops (floating point operations, see [14, pp.18–19]) unless k is very close to 0 or n. Our new procedure (Algorithm 4.2) requires $O(mn)$ flops regardless of the value of k.

Throughout the paper, we use MATLAB notation for submatrices, thus $X(i{:}j,:)$ denotes rows i through j of X, $X(:,k{:}\ell)$ denotes columns k through ℓ, and $X(i{:}j,k{:}\ell)$ denotes the intersection of the two.

In the next section, we give some improvements of singular value and vector bounds in [19, 12]. In §3, we introduce the necessary matrix computational tools to build the algorithms for the ULVD. In §4, give our new refinement procedure (Algorithm 4.2) and some results on its properties. In §5, we give numerical tests of the new algorithm, and give a conclusion in §6.

2. Subspaces and Singular Values. The ULVD (1.4)–(1.5) generates approximations to the right and left singular subspaces associated with the first k and last $n-k$ singular values. In the Davis–Kahan [7] framework the errors in these subspaces are characterized by

$$|\sin\,\Theta_R| = \|W_1^T V_2\|_2 = \|V_1^T W_2\|_2, \tag{2.1}$$

$$|\sin\,\Theta_L| = \|Y_1^T U_2\|_2 = \|U_1^T Y_2\|_2. \tag{2.2}$$

The following proposition is a slight improvement on bounds by Mathias and Stewart [19] and Bunch and Fierro [12]. It characterizes the error in the singular subpspaces and singular values. The long, but elementary proof is in longer technical report version of this paper [2].

PROPOSITION 2.1. *Let X have the ULVD* (1.4)–(1.5) *and let*

$$\rho \stackrel{def}{=} \min\left\{\frac{\sigma_{k+1}(X)}{\sigma_k(L)}, \frac{\sigma_1(G)}{\sigma_k(X)}\right\}, \tag{2.3}$$

$$\gamma_R \stackrel{def}{=} \max\{\sigma_k^2(X) - \sigma_1^2(G), \sigma_k^2\left[\begin{pmatrix} L \\ F \end{pmatrix}\right] - \sigma_{k+1}^2(X)\}, \tag{2.4}$$

$$\gamma_L \stackrel{def}{=} \max\{\frac{\sigma_k^2(X) - \sigma_1^2(G)}{\sigma_k(X)}, \frac{\sigma_k^2(L) - \sigma_{k+1}^2(X)}{\sigma_k(L)}\}. \tag{2.5}$$

Then

$$|\sin\,\Theta_R| \le \frac{\|F^T G\|_2}{\gamma_R}, \quad |\sin\,\Theta_L| \le \frac{\|F\|_2}{\gamma_L}, \tag{2.6}$$

$$|\sin\,\Theta_R| \le \rho|\sin\,\Theta_L|, \tag{2.7}$$

$$|\cos\,\Theta_L| \le \frac{\sigma_j(L)}{\sigma_j(X)}, \frac{\sigma_{i+k}(X)}{\sigma_i(G)} \le 1, \quad \begin{matrix} j = 1, 2\ldots, k \\ i = 1, 2, \ldots, n-k. \end{matrix} \tag{2.8}$$

The bounds in [19] and [12] are the right side of these two inequalities, thus the bounds in (2.6) are sharper.

$$|\sin\,\Theta_R| \le \frac{\|F^T G\|_2}{\gamma_R} \le \frac{\|F\|_2\|G\|_2}{\sigma_k^2(L) - \sigma_1^2(G)},$$
$$|\sin\,\Theta_L| \le \frac{\|F\|_2}{\gamma_L} \le \frac{\|F\|_2\sigma_k(L)}{\sigma_k^2(L) - \sigma_1^2(G)}.$$

Through the use of trigonometric identities (2.8) implies the relative error bound

$$0 \le \frac{\sigma_j(X) - \sigma_j(L)}{\sigma_j(X)}, \frac{\sigma_i(G) - \sigma_{i+k}(X)}{\sigma_{i+k}(X)} \le \frac{\sin^2\,\Theta_L}{1 + |\cos\,\Theta_L|}, \tag{2.9}$$

for $j = 1, 2\ldots, k$ and $i = 1, 2, \ldots, n-k$. Thus as pointed out in [19], the relative error in the singular values is proportional to $\|F\|_2^2$.

3. Matrix Computational Tools for the ULVD.

3.1. Norm and Condition Estimates. For a lower triangular matrix $C \in \mathbf{R}^{n\times n}$, we need to estimate its smallest singular value and associated vectors given by the triple $(\sigma_n, \mathbf{u}_n, \mathbf{v}_n)$. Realistically, we need to compute $\bar{\sigma}_n$, $\bar{\mathbf{u}}_n$, and $\bar{\mathbf{v}}_n$ such that

$$C^{-1}\bar{\mathbf{u}}_n = \bar{\sigma}_n^{-1}\bar{\mathbf{v}}_n, \tag{3.1}$$

$$C^{-T}\bar{\mathbf{v}}_n = \bar{\sigma}_n^{-1}\bar{\mathbf{u}}_n + \mathbf{r}_{inv}, \tag{3.2}$$

$$\mathbf{r}_{inv}^T\bar{\mathbf{u}}_n = 0, \quad \|\mathbf{r}_{inv}\| \le \bar{\sigma}_n^{-1}\delta \tag{3.3}$$

where δ is some prescribed tolerance which should be smaller than ϵ in (1.5). In our tests in §5, we used $\delta = 10^{-3}\epsilon$. In practice, we expect to use only a few Lanczos or power iterations and some back and forward solves with C to perform this computation.

Likewise, to estimate the largest singular value and associated vectors, given by the triple $(\sigma_1, \mathbf{u}_1, \mathbf{v}_1)$, we compute a triple $(\bar{\sigma}_1, \bar{\mathbf{u}}_1, \bar{\mathbf{v}}_1)$ such that

$$C\bar{\mathbf{v}}_1 = \bar{\sigma}_1\bar{\mathbf{u}}_1, \tag{3.4}$$

$$C^T\bar{\mathbf{u}}_1 = \bar{\sigma}_1\bar{\mathbf{v}}_1 + \mathbf{r}, \tag{3.5}$$

$$\mathbf{r}^T\bar{\mathbf{v}}_1 = 0, \quad \|\mathbf{r}\| \le \bar{\sigma}_1\delta. \tag{3.6}$$

Again, this can be achieved with a few Lanczos or power iterations. Here we do not restrict C to be lower triangular or even square. For complexity estimates, we assume a fixed number iterations, c_{iter}, for both of these procedures.

When these operations are counted in complexity estimates, we count (3.1)–(3.3) as $c_{iter}n^2$ flops and for $C \in \mathbf{R}^{m\times n}$ we count (3.4)–(3.6) as $2c_{iter}mn$ flops.

When using these procedures in the analysis of other algorithms, we ignore the errors $\mathbf{r}$ and $\mathbf{r}_{inv}$, though they affect the accuracy.

3.2. Chasing Algorithms. Once we have computed the approximate singular vectors in §3.1, we need the two chasing procedures described in this section. Such algorithms are discussed in [23, 12, 1, 4], so we just summarize them here.

First, suppose that we have computed $(\bar{\sigma}_n, \bar{\mathbf{u}}_n, \bar{\mathbf{v}}_n)$ from (3.1)–(3.3). To "deflate" this smallest singular value, we find an orthogonal matrix $\bar{U}$ such that

$$\bar{U}^T\bar{\mathbf{u}}_n = \pm\mathbf{e}_n \tag{3.7}$$

and an orthogonal matrix $\bar{V}$ such that

$$\bar{U}^T C\bar{V} = \hat{C} \tag{3.8}$$

is lower triangular. The matrices $\bar{U}$ and $\bar{V}$ are each the products of $n-1$ Givens rotations.

The algorithm (3.7)–(3.8) has interesting consequences for $\hat{C}$. If

$$\hat{\mathbf{v}}_n = \bar{V}^T\bar{\mathbf{v}}_n, \quad \hat{\mathbf{r}}_{inv} = U^T\mathbf{r}_{inv}, \tag{3.9}$$

then (3.1)–(3.3) becomes

$$\hat{C}\hat{\mathbf{v}}_n = \pm\bar{\sigma}_n\mathbf{e}_n, \tag{3.10}$$

$$\pm\hat{C}^T\mathbf{e}_n = \bar{\sigma}_n\hat{\mathbf{v}}_n + \bar{\sigma}_n\hat{C}^T\hat{\mathbf{r}}_{inv}, \quad \|\hat{\mathbf{r}}_{inv}\| \le \delta\bar{\sigma}_n^{-1}. \tag{3.11}$$

It is easily concluded from (3.10) that

$$\hat{\mathbf{v}}_n = \pm\mathbf{e}_n, \quad \hat{c}_{nn} = \bar{\sigma}_n,$$

$$\|\hat{C}^T\mathbf{e}_n\|^2 = \bar{\sigma}_n^2 + \bar{\sigma}_n^2\|\hat{C}^T\hat{\mathbf{r}}_{inv}\|^2 \le \bar{\sigma}_n^2 + \|\hat{C}\|^2\delta^2,$$

and that

$$\begin{pmatrix} 0 & \hat{C}(n, 2{:}n) \end{pmatrix} = \mp\bar{\sigma}_n\hat{\mathbf{r}}_{inv}^T\hat{C}.$$

Thus

$$\|\hat{C}(n, 2{:}n)\| \le \delta\|C\|.$$

Likewise, if C is lower trapezoidal with more columns than rows, we can develop a chasing procedure similar to that above that starts with $\bar{\mathbf{u}}_1$ in (3.4)–(3.6) and produces orthogonal matrices $\bar{U}$ and $\bar{V}$ such that

$$\bar{U}^T\bar{\mathbf{u}}_1 = \pm\mathbf{e}_1, \tag{3.12}$$

and $\bar{V}$ such that $\hat{C}$ in (3.8) remains lower trapezoidal.

If $\hat{\mathbf{v}}_1 = V^T\bar{\mathbf{v}}_1$ and $\hat{\mathbf{r}} = V^T\mathbf{r}$, then

$$\hat{C}\hat{\mathbf{v}}_1 = \pm\bar{\sigma}_1\mathbf{e}_1,$$

$$\pm\hat{C}^T\mathbf{e}_1 = \bar{\sigma}_1\hat{\mathbf{v}}_1 + \hat{\mathbf{r}}, \quad \|\hat{\mathbf{r}}\| \le \delta\bar{\sigma}_1.$$

Since $\hat{\mathbf{r}}^T\hat{\mathbf{v}}_1 = 0$, we have

$$\|\hat{C}^T\mathbf{e}_1\|^2 = \bar{\sigma}_1^2 + \|\hat{\mathbf{r}}\|^2, \tag{3.13}$$

thus

$$\bar{\sigma}_1 \le \|\hat{C}^T\mathbf{e}_1\| \le \bar{\sigma}_1(1+\delta^2)^{1/2}. \tag{3.14}$$

4. Refining a ULVD. Refinement is a process of improving the accuracy of the ULVD. We discuss how to measure that improvement below.

The notion of refinement was popularized in [19] where the following was presented. It is equivalent to the QR algorithm without shifts.

ALGORITHM 4.1 (Mathias–Stewart Refinement Procedure).

Step 1. *Find orthogonal* $\tilde{U} \in \mathbf{R}^{n\times n}$ *such that*

$$\tilde{U}^TC = \tilde{U}^T\begin{pmatrix} L & 0 \\ F & G \end{pmatrix} = \begin{pmatrix} L^{(1)} & F^{(1)} \\ 0 & G^{(1)} \end{pmatrix} = C^{(1)}.$$

$$U \leftarrow U\tilde{U}.$$

Step 2. *Find orthogonal* $\tilde{V} \in \mathbf{R}^{n\times n}$ *such that*

$$C^{(1)}\tilde{V} = \begin{pmatrix} L^{(1)} & F^{(1)} \\ 0 & G^{(1)} \end{pmatrix}\tilde{V} = \begin{pmatrix} \bar{L} & 0 \\ \bar{F} & \bar{G} \end{pmatrix}$$

$$V \leftarrow V\tilde{V}.$$

We note three important properties of Algorithm 4.1. Other properties were shown in [19].

1. Since this is the QR algorithm without shifts, it follows that asymptotically
$$\|\bar{F}\| \approx \|F\| \frac{\sigma_{k+1}^2(C)}{\sigma_k^2(C)}.$$
2. One can show that
$$\|(\ \bar{F} \quad \bar{G} \ )\| \le \|G\|, \quad \|(\ \bar{F} \quad \bar{G} \ )\|_F \le \|G\|_F.$$
3. Repeated application of this algorithm forces $\|(\ \bar{F} \quad \bar{G} \ )\|$ to converge to $\sigma_{k+1}(C)$, but at about the rate of the shiftless QR algorithm.

This algorithm requires $O((m+n)(n-k)k)$ flops where the constant in $O(\cdot)$ depends upon the implementation. Thus, if $\min\{k, n-k\} = O(1)$, then this is an $O(mn)$ algorithm, but if $\min\{k, n-k\} = O(n)$, this is an $O(mn^2)$ algorithm. In our tests, we used a chasing procedure that combines steps 1 and 2.

An alternative to Algorithm 4.1, proposed in [12], just computes a ULVD of C, and thereby produce a better ULVD of X. This costs about the same as Algorithm 4.1, it is $O(mn(n-k))$ flops, making it $O(mn)$ flops if $n-k = O(1)$ and $O(mn^2)$ flops if $n-k = O(n)$.

Below we give an algorithm that computes a refinement in $O(mn)$ flops no matter what the value of k is. For our statement of this refinement procedure, C has the form
$$C = \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} \hat{L} & 0 \\ \hat{F} & \hat{G} \end{pmatrix} \end{array}.$$

ALGORITHM 4.2 (Alternative Refinement Algorithm).

Step 1. *Using procedures such as those in §3.1, find unit vectors $\bar{\mathbf{u}}_1, \bar{\mathbf{v}}_1$ such that*
$$(\ \hat{F} \quad \hat{G} \ ) \bar{\mathbf{v}}_1 = \bar{\sigma}_1 \bar{\mathbf{u}}_1,$$
where $\bar{\sigma}_1$ is the largest singular value of $(\ \bar{F} \quad \bar{G} \ )$.

Step 2. *Construct orthogonal matrices $U_{(1)}$ and $V_{(1)}$ such that*
$$[U_{(1)}]^T \bar{\mathbf{u}}_1 = \pm \mathbf{e}_1,$$
and
$$G_{(1)} = [U_{(1)}]^T \bar{G} V_{(1)}$$
remains lower triangular. Also compute
$$F_{(1)} = [U_{(1)}]^T \hat{F},$$
$$U(:, k+1:n) \leftarrow U(:, k+1:n) U_{(1)}, \quad V(:, k+1:n) \leftarrow V(:, k+1:n) V_{(1)}.$$
Note that
$$(\ F_{(1)} \quad G_{(1)} \ ) = \begin{array}{c} \\ 1 \\ n-k-1 \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} [\mathbf{f}_1^{(1)}]^T & g_{11}^{(1)} \mathbf{e}_1^T \\ \tilde{F}^{(1)} & \tilde{G}^{(1)} \end{pmatrix} \end{array}$$

where

$$\|(\ \mathbf{f}_1^{(1)^T} \quad g_{11}^{(1)}\mathbf{e}_1^T\ )\| = \bar{\sigma}_1[(\ \hat{F} \quad \hat{G}\ )], \quad \|(\ \tilde{F}_{(1)} \quad \tilde{G}_{(1)}\ )\| = \sigma_2[(\ \hat{F} \quad \hat{G}\ )].$$

Step 3. *Let*

$$S = \begin{pmatrix} \hat{L} & 0 \\ \mathbf{f}_1^{(1)^T} & g_{11}^{(1)} \end{pmatrix}$$

Find unit vectors $\bar{\mathbf{u}}_{k+1}, \bar{\mathbf{v}}_{k+1}$ *such that*

$$S\bar{\mathbf{v}}_{k+1} = \bar{\sigma}_{k+1}\bar{\mathbf{u}}_{k+1}$$

where $\bar{\sigma}_{k+1}$ *is the smallest singular value of* S.

Step 4. *Construct orthogonal matrices* $U_{(2)}, V_{(2)} \in \mathbf{R}^{(k+1)\times(k+1)}$ *such that*

$$[U_{(2)}]^T\bar{\mathbf{u}}_{k+1} = \pm\mathbf{e}_{k+1}$$

$$S^{(1)} = \begin{pmatrix} L_{(1)} & 0 \\ 0 & \bar{\sigma}_{k+1} \end{pmatrix} = [U_{(2)}]^T S V_{(2)}$$

where $L_{(1)}$ *is lower triangular.*

$$(\ \tilde{F}_{(2)} \quad \tilde{G}_{(2)}(:,1)\ ) = (\ \tilde{F}_{(1)} \quad \tilde{G}_{(1)}(:,1)\ )\, V_{(2)}$$

$$U(:,1{:}k+1) \leftarrow U(:,1{:}k+1)U_{(2)}, \quad V(:,1{:}k+1) \leftarrow V(:,1{:}k+1)V_{(2)}.$$

If $\bar{\sigma}_{k+1} \geq \epsilon$, *let*

$$\bar{L} = S^{(1)}, \quad \bar{F} = (\ \tilde{F}^{(2)} \quad \tilde{G}(:,1)\ ), \quad \bar{G} = \tilde{G}^{(2)}(:,2{:}n-k)$$

else

$$\bar{L} = L^{(1)}, \quad \bar{F} = \begin{pmatrix} 0 \\ \tilde{F}^{(2)} \end{pmatrix}, \quad \bar{G} = \begin{pmatrix} \bar{\sigma}_{k+1}\mathbf{e}_1^T \\ \tilde{G}^{(2)} \end{pmatrix}.$$

The refinement algorithm requires $6(m+n)n + 12nk + c_{iter}n^2$ flops. Thus, this is an $O(mn)$ algorithm for all k.

It does not have a property similar to property (1) of Algorithm 4.1. Fortunately, it does have two properties similar to properties (2) and (3). These are:

1.
$$\|(\ \bar{F} \quad \bar{G}\ )\|_F \leq \|(\ F \quad G\ )\|_F$$
where equality occurs if and only if $\|(\ F \quad G\ )\| = \sigma_{k+1}(C)$.

2. Repeated application forces $\|(\ \bar{F} \quad \bar{G}\ )\|$ to converge to $\sigma_{k+1}(C)$. Under reasonable assumptions, a good approximation of $\sigma_{k+1}(C)$ is produced after one iteration.

The following lemma exactly quantifies the decrease in $\|(\ F \quad G\ )\|_F$.

LEMMA 4.1. *Assume the terminology of Algorithm* 4.2. *If* $\bar{\sigma}_{k+1} < \epsilon$ *then*

$$\|(\bar{F}\ \bar{G})\|_F^2 = \|(F\ G)\|_F^2 - \sigma_1^2[(F\ G)] + \sigma_{k+1}^2(S) \tag{4.1}$$

$$= \|(F\ G)\|_F^2 - \|([\mathbf{f}_1^{(1)}]^T, g_{11}^{(1)})\|^2 + \bar{\sigma}_{k+1}^2. \tag{4.2}$$

If $\bar{\sigma}_{k+1} \geq \epsilon$ *then*

$$\|(\bar{F}\ \bar{G})\|_F^2 = \|(F\ G)\|_F^2 - \sigma_1^2[(F\ G)] \tag{4.3}$$

and

$$\|(\ \bar{F} \quad \bar{G}\ )\| = \sigma_2(\ F \quad G\ ) \leq \|(\ F \quad G\ )\|.$$

Proof. First, due to orthogonal invariance of Frobenius norm we first observe that

$$\|(\ F_1 \quad G_1\ )\|_F = \|(\ F \quad G\ )\|_F \tag{4.4}$$

where $\|(\ F_1 \quad G_1\ )^T \mathbf{e}_1\|$ is the largest singular value of the matrix $(F\ G)$. Second, we observe that

$$\| \begin{pmatrix} \bar{L} & 0 \\ 0 & \bar{\sigma}_{k+1} \end{pmatrix} \|_F = \| \begin{pmatrix} L & 0 \\ \mathbf{f}_1^{(1)^T} & g_{11}^{(1)} \end{pmatrix} \|_F \tag{4.5}$$

where $\bar{\sigma}_{k+1}$ is the smallest singular value of the matrix $\begin{pmatrix} L & 0 \\ \mathbf{f}_1^{(1)^T} & g_{11}^{(1)} \end{pmatrix}$. Thus equation (4.1) follows from combining (4.4)–(4.5). If $\bar{\sigma}_{k+1} > \epsilon$, then the last row of $S^{(1)}$ is not included in $(\ \bar{F} \quad \bar{G}\ )$, thus (4.3).

Since $\|(\ \mathbf{f}_1^{(1)^T} \quad g_{11}^{(1)}\ )^T\| = \sigma_1[(\ F \quad G\ )]$ and $\mathbf{e}_1$ is the left singular vector associated with this singular value, it follows that $\|(\ \tilde{F}^{(1)} \quad \tilde{G}^{(2)}\ )\| = \sigma_2[(\ F \quad G\ )]$ □

The computation of equation (4.2) requires $O(n)$ flops.

The following theorem shows that repeated application of Algorithm 4.2 forces $\|(\ F \quad G\ )\|$ to converge to $\sigma_{k+1}(C)$.

THEOREM 4.2. *Suppose we apply Algorithm* 4.2 *to* $C_0 = \begin{pmatrix} L_0 & 0 \\ F_0 & G_0 \end{pmatrix}$ *t times to obtain the matrix* $C_t = \begin{pmatrix} L_t & 0 \\ F_t & G_t \end{pmatrix}$. *Then*

$$\lim_{t \to \infty} \|(\ F_t \quad G_t\ )\| = \sigma_{k+1}(C_0). \tag{4.6}$$

Proof. By using Lemma 4.1 we have

$$\|(F_{t+1}\ G_{t+1})\|_F^2 - \|(F_t\ G_t)\|_F^2 = \|(F_t\ G_t)\|^2 - \sigma_{k+1}\left[\begin{pmatrix} L_{t+1} & 0 \\ \mathbf{f}_1^{(t+1)^T} & g_{11}^{(t+1)} \end{pmatrix}\right]^2 \tag{4.7}$$

where $\mathbf{f}_1^{(t+1)^T} = F_{t+1}^T \mathbf{e}_1$, $g_{11}^{(t+1)} = \mathbf{e}_1^T G_{t+1} \mathbf{e}_1$.

By the Cauchy interlace theorem [25, pp.103–104], we have

$$\sigma_{k+1}\left[\begin{pmatrix} L_{t+1} & 0 \\ \mathbf{f}_1^{(t+1)^T} & g_{11}^{(t+1)} \end{pmatrix}\right] \leq \sigma_{k+1}(C_0) \leq \|\begin{pmatrix} F & G \end{pmatrix}\|. \tag{4.8}$$

Thus from (4.7),

$$\|(F_{t+1}\ G_{t+1})\|_F \leq \|(F_t\ G_t)\|_F \tag{4.9}$$

unless $\|(F_t\ G_t)\| = \sigma_{k+1}(C_0)$. Furthermore, we have

$$\sigma_{k+1}(C_0) \leq \|(F_t\ G_t)\|_F \leq \|\begin{pmatrix} F_0 & G_0 \end{pmatrix}\|_F. \tag{4.10}$$

Thus by (4.9) and (4.10), $\{\|(F_t\ G_t)\|_F\}$ defines a monotone, bounded sequence which, in turn, has a limit [20, pp. 47–55]. So we have

$$\lim_{t\to\infty} \|(F_t\ G_t)\|_F = \psi \tag{4.11}$$

for some $\psi \in \mathbf{R}$.

By equation (4.7), since the sequence $\{\|(F_t\ G_t)\|_F\}$ is a Cauchy sequence, and using (4.8), we have

$$\lim_{t\to\infty} \|(F_t\ G_t)\| = \lim_{t\to\infty} \sigma_{k+1}\left[\begin{pmatrix} L_t & 0 \\ \mathbf{f}_1^{(t)^T} & g_{11}^{(t)} \end{pmatrix}\right] = \sigma_{k+1}(C_0).$$

□

The next theorem shows that just one step of the refinement procedure can yield a value of $\bar{\sigma}_{k+1}$ that is a very good approximation of the $(k+1)st$ singular value of C.

THEOREM 4.3. *For a given matrix C of the form*

$$C = \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} L & 0 \\ F & G \end{pmatrix} \end{array}, \tag{4.12}$$

Algorithm 4.2 *produces a matrix of the form*

$$\bar{C} = \begin{array}{c} \\ k \\ 1 \\ n-k-1 \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} \bar{L} & 0 \\ 0 & \bar{\sigma}_{k+1}\mathbf{e}_1^T \\ \tilde{F} & \tilde{G} \end{pmatrix} \end{array} \tag{4.13}$$

such that

$$0 \leq \sigma_{k+1}(C) - \bar{\sigma}_{k+1} \leq \frac{\sigma_2^2[\begin{pmatrix} F & G \end{pmatrix}]}{\sigma_{k+1}(C) + \bar{\sigma}_{k+1}}. \tag{4.14}$$

Proof. By orthogonal equivalence, $\bar{C}$ and C have the same singular values. First, we note that

$$\bar{C}\bar{C}^T = \begin{array}{c} \\ k \\ 1 \\ n-k-1 \end{array} \begin{array}{c} \begin{array}{ccc} k & 1 & n-k-1 \end{array} \\ \begin{pmatrix} \bar{L}\bar{L}^T & 0 & \bar{L}\tilde{F}^T \\ 0 & \bar{\sigma}_{k+1}^2 & \bar{\sigma}_{k+1}\mathbf{e}_1^T\tilde{G}^T \\ \tilde{F}\bar{L}^T & \bar{\sigma}_{k+1}\tilde{G}\mathbf{e}_1 & \tilde{F}\tilde{F}^T + \tilde{G}\tilde{G}^T \end{pmatrix} \end{array}.$$

The Cauchy interlace theorem applied to $\bar{C}\bar{C}^T$ gives us that

$$\bar{\sigma}_{k+1}^2 \leq \sigma_{k+1}^2(\bar{C}) = \sigma_{k+1}^2(C)$$

thereby yielding the upper bound in (4.14).

If we note that

$$\bar{C}^T\bar{C} = \begin{pmatrix} \bar{L}^T\bar{L} & 0 \\ 0 & \bar{\sigma}_{k+1}^2 \mathbf{e}_1\mathbf{e}_1^T \end{pmatrix} + \begin{pmatrix} \tilde{F} & \tilde{G} \end{pmatrix}^T \begin{pmatrix} \tilde{F} & \tilde{G} \end{pmatrix},$$

then Weyl's monotonicity theorem applied to the $(k+1)st$ eigenvalue of $\bar{C}\bar{C}^T$ is

$$\sigma_{k+1}^2(C) \leq \sigma_{k+1}^2 \left[\begin{pmatrix} \bar{L} & 0 \\ 0 & \bar{\sigma}_{k+1} \end{pmatrix} \right] + \sigma_1^2[\begin{pmatrix} \tilde{F} & \tilde{G} \end{pmatrix}].$$

Steps 1–2 of Algorithm 4.2 give us that

$$\sigma_1[\begin{pmatrix} \tilde{F} & \tilde{G} \end{pmatrix}] = \sigma_2[\begin{pmatrix} F & G \end{pmatrix}].$$

Steps 3–4 give us that

$$\sigma_{k+1} \left[\begin{pmatrix} \bar{L} & 0 \\ 0 & \bar{\sigma}_{k+1} \end{pmatrix} \right] = \bar{\sigma}_{k+1}.$$

Thus

$$\bar{\sigma}_{k+1}^2 \leq \sigma_{k+1}^2(C) \leq \bar{\sigma}_{k+1}^2 + \sigma_2^2[\begin{pmatrix} F & G \end{pmatrix}].$$

Solving for $\sigma_{k+1}(C) - \bar{\sigma}_{k+1}$ yields the upper bound. □

In the next section, we give some numerical tests that confirm the stated properties of these alternative algorithms.

5. Numerical Tests. The following tests were done using MATLAB 6.

Test Set Let $\omega = 1.1\epsilon$, for a fixed value of k, let $\zeta_k = \ln\, \omega/k$. Then let

$$D_k = \mathrm{diag}(e^{\zeta_k}, e^{2\zeta_k}, ..., e^{k\zeta_k}),$$

noting that $\omega = e^{k\zeta_k}$. For $p = \lfloor 100/k \rfloor$, we let

$$\Sigma_k = \mathrm{diag}(D_k, \xi * D_k, \ldots, \xi^p D_k, \xi^{p+1} D_k(1 : 100 - p * k)), \quad \xi = 0.9\epsilon.$$

Using a technique for generating random orthogonal matrices given by Stewart [21], we generated $U, V \in \mathbf{R}^{100\times 100}$ and let

$$X_k = U\Sigma_k V^T.$$

We then computed the Q–R decomposition

$$X_k = QC_k^T$$

and used the lower triangular matrix C_k. This is matrix with exactly k singular values greater than ϵ, but with the singular values clustered.

For each value of $k = 10, 20, \ldots, 90$, we generated 10 such matrices. We used values of $\epsilon = 10^{-\ell}, \ell = 2, \ldots, 8$. The results were similar for all values of k, so we give the results for $\ell = 3$ only.

We computed and plotted the following:

- MATLAB run times, using the **tic** and **toc** commands. (In Figure 5.1)
- Relative error in the kth (in Figure 5.3) and $(k+1)st$ (in Figure 5.2) singular values. That is, we compute the $\log_{10}$ of

$$\frac{|\sigma_k(C) - \sigma_k(L)|}{\sigma_k(C)}, \quad \frac{|\sigma_{k+1}(C) - \sigma_1(G)|}{\sigma_{k+1}(C)}.$$

- Error in the subspaces. We computed the $\log_{10}$ of

$$|\sin\Theta_R| = \|W_1^T V_2\|_2$$

  in Figure 5.1 and the $\log_{10}$ of

$$|\sin\Theta_L| = \|Y_1^T U_2\|_2$$

  in Figure 5.3.

The unrefined ULVD was computed using the m-file **hulv_a** from the UTV tools package in [13]. Except for rare cases resulting from innaccuracy of the condition estimators, the ϵ-psuedoranks of these matrices were computed correctly by all of the procedures.

We applied each of Algorithm 4.1 and Algorihtm 4.2 after a decomposition computed by **hulv_a**. In Figure 5.1, we plotted the flop counts of the two refinement procedures and the subspaces errors including those from computing a ULVD using **hulv_a** with no refinement at all. In Figure 5.2 we graphed the error in the $(k+1)st$ singular value and the rank error using **hulv_a**. In Figure 5.3, we graphed the error in the kth singular value and the left subspaces.

As one would predict, Algorithm 4.2 is much faster than Algorithm 4.1. Algorithm 4.2 significantly improves the estimate of $\sigma_{k+1}(C)$ given by $\sigma_1(G)$ and slightly improves the estimate of $\sigma_k(C)$ given by $\sigma_k(L)$. It does both better than Algorithm 4.1. It also makes the angles Θ_R and Θ_L with the right and left singular subspaces small and does so as well or better than Algorithm 4.1.

6. Conclusion. The ULVD has already been shown to be a good substitute for the SVD in circumstances where frequent updates and downdates are done.

We propose a refinement procedure that requires $O(mn)$ flops that is just as effective as the $O(mn^2)$ flops procedures proposed in [19]. Our tests above show that this procedure successfully improves both the singular value and subspace estimates.

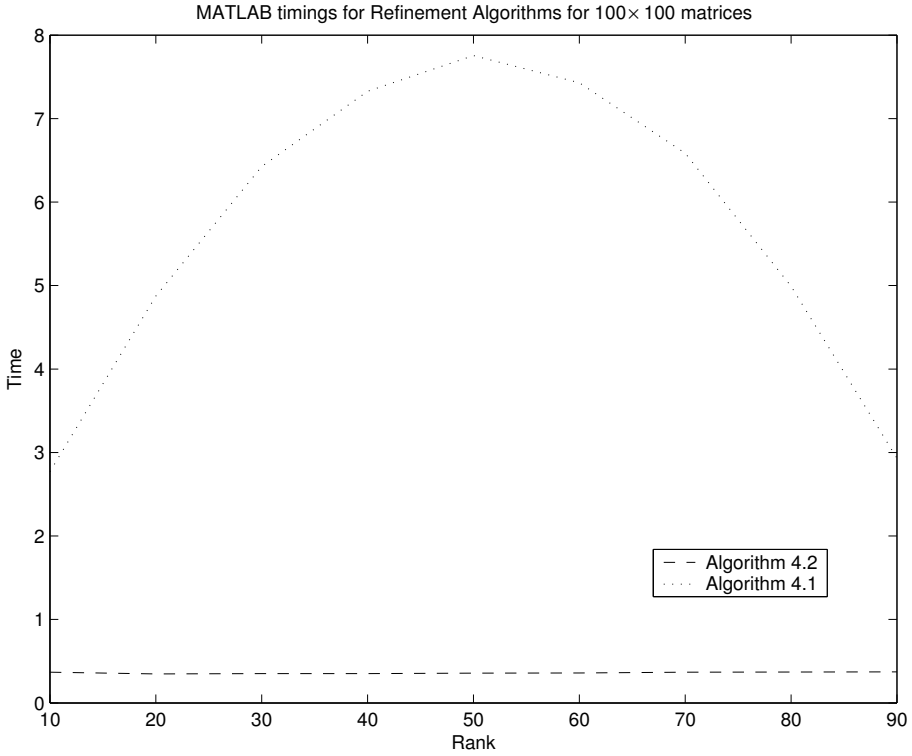

FIG. 5.1. *Flops Counts of Refinement Algorithms,* $\epsilon = 10^{-3}$

REFERENCES

[1] J.L. Barlow. Modification and maintainence of ULV decompositions. to appear, Applied Mathematics and Computing 2001, Z. Drmač et al, eds., 2002.

[2] J.L. Barlow, H. Erbay, and I. Slapničar. An alternative algorithm for refinement of ULV decompositions. Technical report, Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, March 2004. http://www.cse.psu.edu/∼barlow/ULV_Hasan3.ps.

[3] J.L. Barlow and P.A. Yoon. Solving recursive TLS problems using rank-revealing ULV decompositions. In S. Van Huffel, editor, *Recent Advances in Total Least Squares Techniques and Errors–In–Variables Modeling*, pages 117–126, Philadelphia, PA, 1997. SIAM Publications.

[4] J.L. Barlow, P.A. Yoon, and H. Zha. An algorithm and a stability theory for downdating the ULV decomposition. *BIT*, 36:14–40, 1996.

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

[6] P.A. Businger and G.H. Golub. Linear least squares solutions by Householder transformations. *Numerische Mathematik*, 7:269–278, 1965.

[7] C. Davis and W.M. Kahan. The rotation of eigenvectors by a perturbation III. *SIAM J. Num. Anal.*, 7:1–46, 1970.

[8] H. Erbay and J.L. Barlow. Recursive ULV decomposition and an alternative refinement algorithm. In F.T. Luk, editor, *Advanced Signal Processing Algorithms, Architectures and Implementations X*, volume 4116 of *45th SPIE Symposium*, pages 157–166, Bellingham, WA, 2000.

[9] H. Erbay, J.L. Barlow, and I. Slapničar. Recursive ulv decomposition and an alternative refinement algorithm. unpublished manuscript, 2000.

[10] H. Erbay, J.L. Barlow, and Z. Zhang. A modified Gram–Schmidt–based downdating technique for ULV decompostions with applications to recursive TLS problems. *Computational*

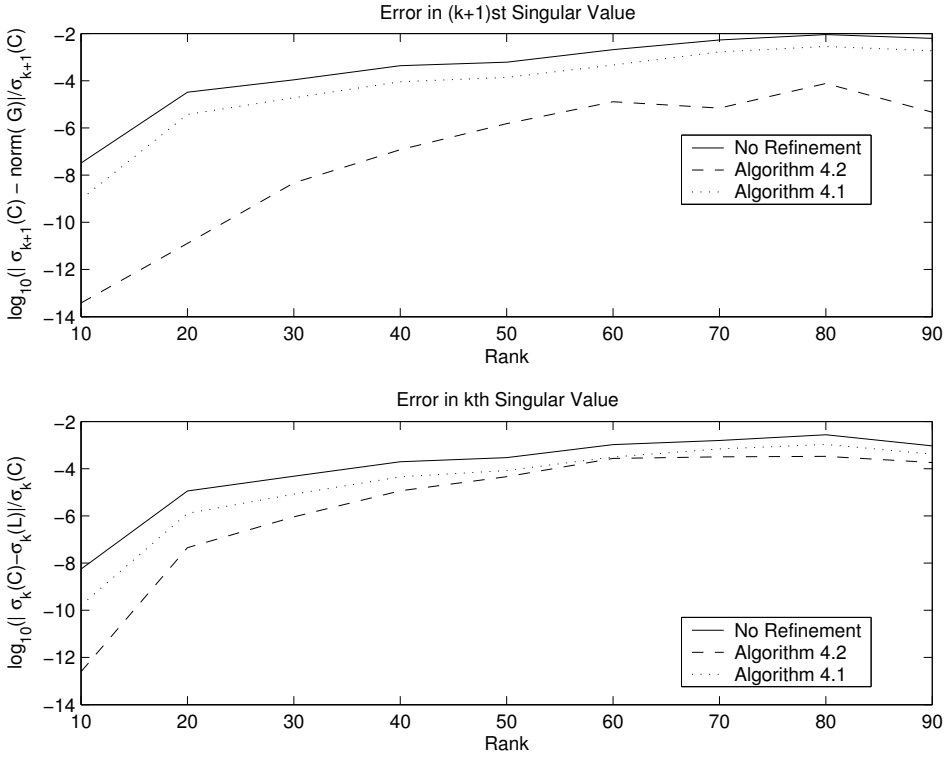

FIG. 5.2. *Error in* $(k+1)$*st and* k*th singular values of Refinement Algorithms,* $\epsilon = 10^{-3}$

Statistics and Data Analysis, 41(1):195–211, 2003.

[11] D.K. Faddeev, V.N. Kublanovskaya, and V.N. Faddeeva. *Sur les Systémes Linéires Algébriques de Matrices Rectangularies et Mal-Conditionnées*, volume VII, pages 161–170. Editions Centre Nat. Recherche Sci., Paris, 1968.

[12] R. Fierro and J. Bunch. Bounding the subspaces from rank revealing two-sided orthogonal decompositions. *SIAM J. Matrix Anal. Appl.*, 16:743–759, 1995.

[13] R. Fierro, P.C. Hansen, and P.S. Kirk. UTV tools: Matlab templates for rank–revealing UTV decompositions. *Numerical Algorithms*, 20(2–3):165–194, 1999.

[14] G.H. Golub and C.F. Van Loan. *Matrix Computations, Third Edition*. The Johns Hopkins Press, Baltimore,MD, 1996.

[15] M. Gu and S. Eisenstat. Downdating the singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 16:793–810, 1995.

[16] R.J. Hanson and C.L. Lawson. Extensions and applications of the Householder algorithm for solving linear least squares problems. *Math. Comp.*, 23:787–812, 1969.

[17] I.T. Joliffe. *Principal Component Analysis*. Springer–Verlag, New York, NY, 1986.

[18] C.L. Lawson and R.J. Hanson. *Solving Least Squares Problems*. Prentice-Hall, Englewood Cliff, NJ, 1974.

[19] R. Mathias and G.W. Stewart. A block QR algorithm and the singular value decomposition. *Linear Algebra and Its Applications*, 182:91–100, 1993.

[20] W. Rudin. *Principals of Mathematical Analysis: Third Edition*. McGraw-Hill, New York, 1976.

[21] G.W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM J. Num. Anal.*, 17:403–409, 1980.

[22] G.W. Stewart. An updating algorithm for subspace tracking. *IEEE Transactions on Signal Processing*, 40:1535–1541, 1992.

[23] G.W. Stewart. Updating a rank-revealing ULV decomposition. *SIAM J. Matrix Anal. Appl.*, 14:494–499, 1993.

[24] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects*

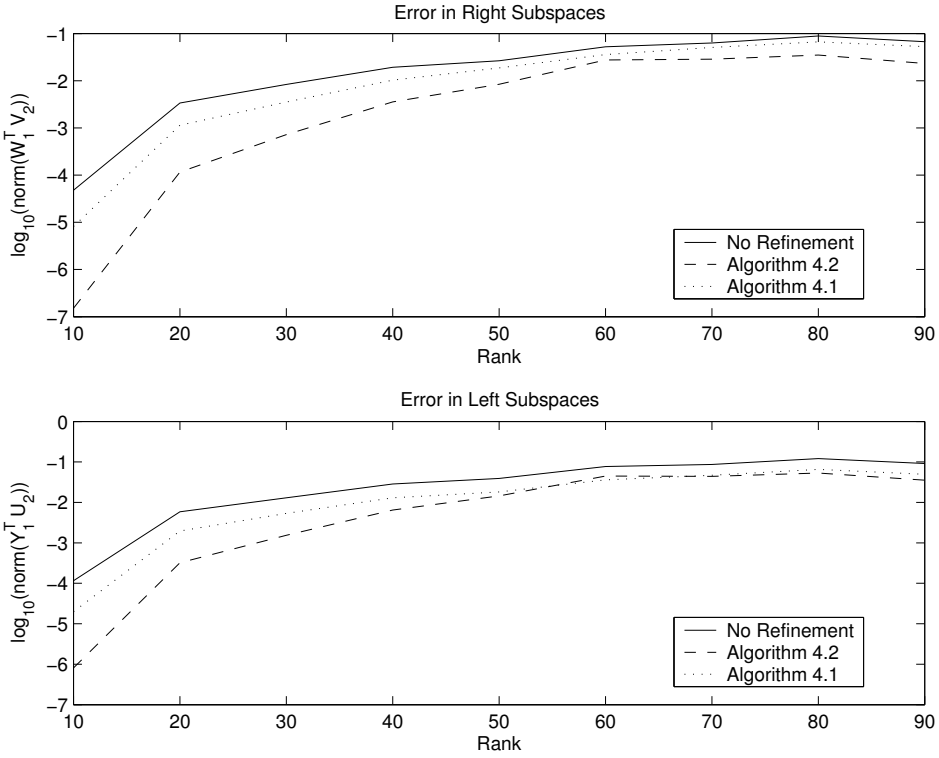

FIG. 5.3. *Error in Right and Left Subspaces of Refinement Algorithms,* $\epsilon = 10^{-3}$

and Analysis. SIAM Publications, Philadelphia, 1991.

[25] J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University Press, London, 1965.

[26] P.A. Yoon and J.L. Barlow. An efficient rank detection procedure for modifying the ULV decomposition. *BIT*, 38:781–801, 1998.