Numer Algor (2006) **42**: 31–61
DOI 10.1007/s11075-006-9022-x

Using cross-product matrices to compute the SVD

Zhongxiao Jia

Received: 24 December 2004 / Revised: 16 February 2006 /
Accepted: 16 February 2006

Abstract This paper concerns accurate computation of the singular value decomposition (SVD) of an $m \times n$ matrix A. As is well known, cross-product matrix based SVD algorithms compute large singular values accurately but generally deliver poor small singular values. A new novel cross-product matrix based SVD method is proposed: (a) Use a backward stable algorithm to compute the eigenpairs of $A^{\mathrm{T}}A$ and take the square roots of the large eigenvalues of it as the large singular values of A; (b) form the Rayleigh quotient of $A^{\mathrm{T}}A$ with respect to the matrix consisting of the computed eigenvectors associated with the computed small eigenvalues of $A^{\mathrm{T}}A$; (c) compute the eigenvalues of the Rayleigh quotient and take the square roots of them as the small singular values of A. A detailed quantitative error analysis is conducted on the method. It is proved that if small singular values are well separated from the large ones then the method can compute the small ones accurately up to the order of the unit roundoff ϵ. An algorithm is developed that is not only cheaper than the standard Golub–Reinsch and Chan SVD algorithms but also can update or downdate a new SVD by adding or deleting a row and compute certain refined Ritz vectors for large matrix eigenproblems at very low cost. Several variants of the algorithm are proposed that compute some or all parts of the SVD. Typical numerical examples confirm the high accuracy of our algorithm.

Keywords cross-product matrix · eigenvalue · eigenvector · finite precision arithmetic · Rayleigh quotient · singular value · singular vector · SVD

AMS(MOS) Subject Classfications 65F15 · 65F30 · 65G05 · 15A12 · 15A18

Communicated by Claude Brezinski.

Supported in part by the National Science Foundation of China (No. 10471074).

Z. Jia (✉)
Department of Mathematical Sciences, Tsinghua University, Beijing 100084, PR China
e-mail: jiazx@tsinghua.edu.cn

1. Introduction

The SVD of an $m \times n$ matrix A with $m \geq n$ is of great theoretical and practical importance. It plays a key role in many application areas such as least squares solutions, signal and image processing, control theory, pattern recognition and time-series analysis, to name only a few. There are several reliable and efficient algorithms for accurately computing the SVD [3, 4]:

$$A = U \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^{\mathrm{T}} = U_1 \Sigma V^{\mathrm{T}}, \tag{1}$$

where $U = (U_1 \;\; U_2) = (u_1, u_2, \ldots, u_m)$ and $V = (v_1, v_2, \ldots, v_n)$ are $m \times m$ and $n \times n$ orthogonal matrices, Σ is the diagonal matrix with the diagonal entries $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$, u_i and v_i are the left and right singular vectors associated with σ_i, $i = 1, 2, \ldots, n$, respectively, and the superscript T denotes the transpose of a matrix (here we only consider a real A).

Throughout this paper, the norm $\|\cdot\|$ is the Euclidean vector norm and the subordinate spectral matrix norm, and $fl(\cdot)$ denotes a computed quantity in finite precision arithmetic. We assume that singular vectors, eigenvectors and their computed counterparts have been normalized to have norm one. $\sin\angle(x, y)$ and $\sin\angle(\mathcal{X}, \mathcal{Y})$ denote the distances between two nonzero vectors x and y and two subspaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. For more on the properties of sin, see [4, 10].

It follows from (1) that

$$A^{\mathrm{T}} A V = V \Sigma^2. \tag{2}$$

That is, σ_i^2, v_i, $i = 1, 2, \ldots, n$ are the eigenvalues and eigenvectors of the cross-product matrix $A^{\mathrm{T}} A$, and we have $\sigma_i = \|Av_i\| = \sqrt{v_i^{\mathrm{T}} A^{\mathrm{T}} A v_i}$. Therefore, theoretically speaking, we may use a backward stable algorithm to compute the eigenpairs σ_i^2, v_i, $i = 1, 2, \ldots, n$ of $A^{\mathrm{T}} A$ and get Σ, V. However, it is well known that such an algorithm may not be reliable for accurately computing small singular values of A because of a possible loss of useful information when forming $A^{\mathrm{T}} A$ in finite precision arithmetic [2–4, 13]; see an analysis below.

For brevity, we assume that $\|A\| = 1$ throughout the paper. Let ϵ be the unit roundoff. Then a backward stable algorithm applied to $A^{\mathrm{T}} A$ returns a computed decomposition $\hat{V} \hat{\Sigma}^2 \hat{V}^{\mathrm{T}}$, which is nearly the exact eigendecomposition of a matrix $A^{\mathrm{T}} A + E$, i.e., $A^{\mathrm{T}} A + E = (\hat{V} + \delta \hat{V}) \hat{\Sigma}^2 (\hat{V} + \delta \hat{V})^{\mathrm{T}}$ in which $\hat{V} + \delta \hat{V}$ is orthogonal, where $\frac{\|E\|}{\|A^{\mathrm{T}} A\|} = \|E\| \leq \alpha\epsilon$ and $\|\delta V\| \leq \alpha\epsilon$ with α being a modest constant depending on m and n; see, e.g., [1, p. 83]. We now look at a well known result [1, p. 83, 87] followed by an analysis.

Theorem 1.1. Let all the notations as defined above. Then

$$\left|\hat{\sigma}_i^2 - \sigma_i^2\right| \leq \alpha\epsilon. \tag{3}$$

Define $g_i = \min_{\sigma_j \neq \sigma_i} \left|\sigma_i^2 - \sigma_j^2\right|$. Then if $g_i > 4\alpha\epsilon$, we have

$$\sin\angle(\hat{v}_i, v_i) \leq \frac{2\alpha\epsilon}{g_i - 4\alpha\epsilon}. \tag{4}$$

If $\sigma_i \leq \sqrt{\alpha\epsilon}$, we have from (3)

$$\sigma_i^2 - \alpha\epsilon \leq \hat{\sigma}_i^2 \leq \sigma_i^2 + \alpha\epsilon \leq 2\alpha\epsilon,$$

and so $|\hat{\sigma}_i| \leq \sqrt{2\alpha\epsilon}$ (note that $\hat{\sigma}_i^2$ may well be negative since $\sigma_i^2 - \alpha\epsilon$ is nonpositive under assumption). Therefore, we obtain

$$\left|\hat{\sigma}_i - \sigma_i\right| \leq \left|\hat{\sigma}_i\right| + |\sigma_i| \leq (1 + \sqrt{2})\sqrt{\alpha\epsilon} = O(\sqrt{\epsilon}). \tag{5}$$

If $1 \geq \sigma_i > \sqrt{\alpha\epsilon}$, we have from (3)

$$0 \leq \sigma_i^2 - \alpha\epsilon \leq \hat{\sigma}_i^2 \leq \sigma_i^2 + \alpha\epsilon,$$

and so $\hat{\sigma}_i \geq 0$. Therefore, we get

$$\left|\hat{\sigma}_i - \sigma_i\right| \leq \frac{\alpha\epsilon}{\hat{\sigma}_i + \sigma_i} \leq \frac{\alpha\epsilon}{\sigma_i} = \frac{O(\epsilon)}{\sigma_i}, \tag{6}$$

which varies from $\sqrt{\alpha\epsilon} = O(\sqrt{\epsilon})$ to $\alpha\epsilon = O(\epsilon)$ as σ_i increases from $\sqrt{\alpha\epsilon} = O(\sqrt{\epsilon})$ to one. Therefore, if σ_i is $O(1)$, then the absolute error in σ_i is $O(\epsilon)$; however, it deteriorates up to $\sqrt{\alpha\epsilon} = O(\sqrt{\epsilon})$ as σ_i decreases up to $\sqrt{\alpha\epsilon} = O(\sqrt{\epsilon})$. Hence computed small singular values generally have accuracy $O(\sqrt{\epsilon})$. In particular, if $\sigma_n = 0$ and $\epsilon = 10^{-16}$, then $\left|\hat{\sigma}_n\right| = O(10^{-8})$, about 10^8 times larger than $O(10^{-16})$, a computed approximation to $\sigma_n = 0$ by a backward stable algorithm working on A directly!

So a good algorithm applied to $A^{\mathrm{T}}A$ can compute large singular values of A with high absolute accuracy $O(\epsilon)$ but it may fail to return small singular values accurately. As a result, such kind of algorithms is not preferable and seldom used in practice.

On the other hand, however, we observe from (4) a remarkable fact that $\hat{v}_n$ is an approximation to v_n with full accuracy $O(\epsilon)$ if σ_n is well separated from σ_{n-1} no matter how small σ_n is. To see this, let $\sigma_{n-1} = O(1)$ and σ_n be small. Then we have

$$g_n - \alpha\epsilon \approx \sigma_{n-1}^2 - 4\alpha\epsilon \approx \sigma_{n-1}^2 = O(1) \gg 0.$$

So in this case we have

$$\sin\angle(\hat{v}_n, v_n) \leq \frac{2\alpha\epsilon}{g_n - 4\alpha\epsilon} \approx \frac{2\alpha\epsilon}{\sigma_{n-1}^2} = O(\epsilon), \tag{7}$$

that is, although $\hat{\sigma}_n$ may be inaccurate, $\hat{v}_n$ is an approximation to v_n with accuracy $O(\epsilon)$.

In Section 2, assume that σ_{n-1} is not small and is well separated from σ_n. We then take $\|A\hat{v}_n\| = \sqrt{(A\hat{v}_n)^{\mathrm{T}}(A\hat{v}_n)}$ as a new better approximation to σ_n and analyze its accuracy in exact arithmetic. In Section 3 we prove that in finite precision arithmetic $fl(\|A\hat{v}_n\|)$ is an approximation to σ_n with accuracy $O(\epsilon)$. In Section 4, we consider the case that more than one small singular values are present. We have to form a more complicated Rayleigh quotient of the matrix $A^{\mathrm{T}}A$ with respect to the matrix $\hat{V}_2$ whose columns consist of the $\hat{v}$ associated with those small σ (here we drop subscripts). We then take the square roots of eigenvalues of the Rayleigh quotient matrix as new approximations to the small singular values, and we analyze their accuracy in exact arithmetic. In Section 5, we prove that in finite precision arithmetic they have accuracy $O(\epsilon)$ under certain mild conditions and are always much more accurate

than the corresponding $\hat{\sigma}$s obtained by a backward stable algorithm applied to $A^{\mathrm{T}}A$ directly. Based on our theory, we develop an efficient and stable cross-product matrix based SVD algorithm and consider many implementional details in Section 6. We show that our new algorithm are (much) cheaper that the Golub–Reinsch and Chan SVD algorithms in most cases. In Section 7, we address two typical and important applications of our algorithm. The first application is in refined projection methods [7, 8, 12, 14] for a general projection subspace, where our algorithm can compute refined Ritz vectors efficiently and accurately. The second application is that our algorithm can update or downdate a new SVD efficiently and accurately when a row is appended to or deleted from A. In Section 8 we propose several variants of our new algorithm that compute some or all of Σ, U, V, and they are cheaper and can be much cheaper than the Golub–Reinsch and Chan algorithms when the number of small singular values is much smaller than n or $m \ll n$. In Section 9 we report a number of typical numerical examples to illustrate the high accuracy of our algorithm followed by concluding remarks in Section 10.

Before proceeding, we should stress that in this paper we only attempt to compute small singular values with high absolute accuracy, this is the same in the spirit of the Golub–Reinsch SVD and the Chan SVD which is in contrast to those SVD algorithms with high relative accuracy [2, 3], such as the Demmel–Kahan SVD algorithm and the variants of the qd algorithms as well as the Jacobi type SVD algorithms and the SVD algorithms based on divide-and-conquer. Throughout the paper, we always assume that overflow or underflow does not occur. Furthermore, we assume that A is real in order to make error analyses simpler. However, we should comment that our analysis is also valid for a complex matrix A, provided that the constants in inner products and matrix–vector products are increased properly; see [5, p. 80], so that our analysis works.

2. Rayleigh quotient and computation of a small σ_n in exact arithmetic

The preceding analysis motivates us to take $\hat{v}_n$ into account when attempting to compute σ_n as accurately as other standard algorithms working on A directly.

We first present the following result on the Rayleigh quotient $\hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n$.

Theorem 2.1. Assume that $\sigma_{n-1} = O(1)$ and $\sigma_n \ll \sigma_{n-1}$, and let $g_n - 4\alpha\epsilon = \tilde{g}_n$. Then we have

$$\sin \angle(\hat{v}_n, v_n) = \epsilon_1 \leq \frac{2\alpha}{\tilde{g}_n}\epsilon = O(\epsilon) \tag{8}$$

and for the Rayleigh quotient $\hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n = \left\| A\hat{v}_n \right\|^2$ we have

$$| \hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n - \sigma_n^2 | \leq \epsilon_1^2 \leq \frac{4\alpha^2}{\tilde{g}_n^2}\epsilon^2. \tag{9}$$

Proof. (8) follows from (7) and assumptions directly.

By (8) we decompose $\hat{v}_n$ into the orthogonal direct sum

$$\hat{v}_n = \sqrt{1 - \epsilon_1^2}\, v_n + \epsilon_1 v_\perp$$

with $v_\perp^{\mathrm{T}} v_n = 0$ and $\|v_\perp\| = 1$. So we get

$$\begin{aligned}\hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n &= \left(\sqrt{1-\epsilon_1^2}\, v_n + \epsilon_1 v_\perp\right)\left(\sqrt{1-\epsilon_1^2}\,\sigma_n^2 v_n + \epsilon_1 A^{\mathrm{T}} A v_\perp\right) \\ &= (1-\epsilon_1^2)\sigma_n^2 + \epsilon_1^2 v_\perp^{\mathrm{T}} A^{\mathrm{T}} A v_\perp \\ &\le (1-\epsilon_1^2)\sigma_n^2 + \epsilon_1^2,\end{aligned}$$

from which it follows that

$$\left|\hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n - \sigma^2\right| \le (1-\sigma_n^2)\epsilon_1^2 \le \frac{4\alpha^2}{(g_n - 4\alpha\epsilon)^2}\epsilon^2 = \frac{4\alpha^2}{\tilde{g}_n^2}\epsilon^2$$

This completes the proof of (9). □

If $\sigma_n \le \frac{2\alpha}{\tilde{g}_n}\epsilon$, we get from (9)

$$\left\|A\hat{v}_n\right\|^2 \le \frac{4\alpha^2}{\tilde{g}_n^2}\epsilon^2 + \sigma_n^2 \le \frac{8\alpha^2}{\tilde{g}_n^2}\epsilon^2,$$

so

$$\left|\left\|A\hat{v}_n\right\| - \sigma_n\right| \le \left\|A\hat{v}_n\right\| \le \frac{2\sqrt{2}\alpha}{\tilde{g}_n}\epsilon = O(\epsilon); \tag{10}$$

if $\sigma_n > \frac{2\alpha}{\tilde{g}_n}\epsilon$, we have

$$\left|\left\|A\hat{v}_n\right\| - \sigma_n\right| \le \frac{4\alpha^2}{\left(\left\|A\hat{v}_n\right\| + \sigma_n\right)\tilde{g}_n^2}\epsilon^2 \le \frac{4\alpha^2}{\sigma_n \tilde{g}_n^2}\epsilon^2, \tag{11}$$

which is no more than $\frac{2\alpha}{\tilde{g}_n}\epsilon = O(\epsilon)$. Therefore, regardless of the size of σ_n, the approximation $\left\|A\hat{v}_n\right\|$ has at most absolute error $O(\epsilon)$. In particular, we have $\left\|A\hat{v}_n\right\| \le \frac{2\sqrt{2}\alpha}{\tilde{g}_n}\epsilon = O(\epsilon)$ if $\sigma_n = 0$. The accuracy is the same as that achieved by a backward stable SVD algorithm working on A directly. Thus, it seems plausible to take $\left\|A\hat{v}_n\right\|$ as a new approximation to σ_n.

3. Computation of a small σ_n in finite precision arithmetic

In finite precision arithmetic, unfortunately, the above Rayleigh quotient theory fails to hold and accurate computation of a small σ_n becomes subtle. So our method must be considered carefully. As it turns out, although (9) does not hold any longer, careful computation guarantees that $fl(\left\|A\hat{v}_n\right\|)$ is still an approximation to σ_n with accuracy $O(\epsilon)$.

First of all, it appears that in finite precision arithmetic we must compute the Rayleigh quotient using

$$\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right) \tag{12}$$

other than using

$$\hat{v}_n^{\mathrm{T}}\left(\left(A^{\mathrm{T}}A\right)\hat{v}_n\right). \tag{13}$$

Relations (12) and (13) are mathematically equivalent. In finite precision arithmetic, however, they behave very differently when $\left\|A\hat{v}_n\right\|$ is small. For example, suppose

that the smallest singular value σ_n of A is exactly zero and v_n is its right singular vector. Then in finite precision arithmetic we will have

$$fl\left(\left(A^{\mathrm{T}}A\right)v_n\right) = O\left(\epsilon\right),$$

which is already at the level of the unit roundoff and thus will compute a garbage by left multiplying v_n^{T}. The square root of it is $O\left(\sqrt{\epsilon}\right)$, a very inaccurate approximation to $\sigma_n = 0$. On the other hand, however, we have $fl\left(Av_n\right) = O\left(\epsilon\right)$ for (12) in finite precision arithmetic and so $fl\left(\left(Av_n\right)^{\mathrm{T}}\left(Av_n\right)\right) = O(\epsilon^2)$. Taking the square root of it, we get an approximation $O(\epsilon)$ to $\sigma_n = 0$.

In what follows we give a detailed quantitative error analysis on our method and prove that it is as accurate as other standard SVD algorithms for the computation of a small σ_n.

Theorem 3.1. Let $fl\left(\left\|A\hat{v}_n\right\|\right)$ *be a computed singular value of* A by computing $\sqrt{\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)}$ in finite precision arithmetic, where $\sin\angle\left(\hat{v}_n, v_n\right) = \epsilon_1 \leq \frac{2\alpha}{\tilde{g}_n}\epsilon$. Then

$$\left|fl\left(\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)\right) - \sigma_n^2\right| \leq |\beta_1|\,\sigma_n\epsilon + |\beta_2|\epsilon^2 \tag{14}$$

with the real constants β_1 and β_2 satisfying

$$|\beta_1| \leq 2.02n + 1.01m\sigma_n, \quad |\beta_2| \leq \frac{4\alpha^2}{\tilde{g}_n^2}\sigma_n^2 + 2.04mn\sigma_n + \frac{4.04n\alpha}{\tilde{g}_n} + \frac{4\alpha^2}{\tilde{g}_n^2} + 1.02n^2. \tag{15}$$

If $\sigma_n = \beta_3\epsilon$ with β_3 ranging from zero to a modest positive constant, then

$$\left|fl\left(\left\|A\hat{v}_n\right\|\right) - \sigma_n\right| \leq \left|\sqrt{\beta_3^2 + \beta_1\beta_3 + \beta_2} - \beta_3\right|\epsilon \tag{16}$$

by dropping the higher order term $\sqrt{\beta_3^2 + \beta_1\beta_3 + \beta_2}\epsilon^2$.

If σ_n is much bigger than ϵ, then

$$\left|fl\left(\left\|A\hat{v}_n\right\|\right) - \sigma_n\right| \leq \left(|\beta_1| + \sigma_n\right)\epsilon. \tag{17}$$

Proof. By assumption, we have the orthogonal direct sum decomposition

$$\hat{v}_n = \sqrt{1 - \epsilon_1^2}v_n + \epsilon_1 v_\perp,$$

with $\epsilon_1 \leq \frac{2\alpha}{\tilde{g}_n}\epsilon$ and $v_\perp^{\mathrm{T}}v_n = 0$, where $v_\perp$ is a unit length vector lying in the subspace spanned by the right singular vectors $v_1, v_2, \ldots, v_{n-1}$ of A. It follows from $Av_i = \sigma_i u_i,\ i = 1, 2, \ldots, n$ that

$$u_n^{\mathrm{T}}Av_\perp = 0 \tag{18}$$

as $Av_\perp$ lies the subspace spanned by the left singular vectors $u_1, u_2, \ldots, u_{n-1}$ of A.

In finite precision arithmetic, we have (see, e.g., [5, p. 76])

$$fl\left(A\hat{v}_n\right) = A\hat{v}_n + e_2,$$

where

$$\|e_2\| = \epsilon_2 \leq 1.01n\epsilon\,\|A\|\left\|\hat{v}_n\right\| = 1.01n\epsilon.$$

Note from [5, p. 75] that for any m dimensional real vectors x and y it holds that

$$\left| fl(x^{\mathrm{T}}y) - x^{\mathrm{T}}y \right| \leq 1.01m\epsilon \, |x|^{\mathrm{T}} |y| \,,$$

provided that $m\epsilon < 0.01$. Therefore, if $y = x$, then we have

$$\left| fl(x^{\mathrm{T}}x) - x^{\mathrm{T}}x \right| \leq 1.01m\epsilon \, |x|^{\mathrm{T}} |x| = 1.01m\epsilon x^{\mathrm{T}}x,$$

from which it follows immediately that

$$fl\left(x^{\mathrm{T}}x\right) = x^{\mathrm{T}}x \, (1 + \epsilon_3) \,, \quad |\epsilon_3| \leq 1.01m\epsilon. \tag{19}$$

This equality says that a floating point inner product of a vector with itself *must be nonnegative*.

With (18) and (19), we get

$$\begin{aligned}
fl\left(\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)\right) &= \left(A\hat{v}_n + e_2\right)^{\mathrm{T}}\left(A\hat{v}_n + e_2\right)(1+\epsilon_3) \\
&= \left(\sqrt{1-\epsilon_1^2}\sigma_n u_n + \epsilon_1 A v_\perp + e_2\right)^{\mathrm{T}}\left(\sqrt{1-\epsilon_1^2}\sigma_n u_n + \epsilon_1 A v_\perp + e_2\right)(1+\epsilon_3) \\
&= \Big(\left(\sigma_n^2\left(1-\epsilon_1^2\right) + \sigma_n\sqrt{1-\epsilon_1^2}u_n^{\mathrm{T}}e_2 + \epsilon_1^2\left(Av_\perp\right)^{\mathrm{T}}\left(Av_\perp\right) + \epsilon_1\left(Av_\perp\right)^{\mathrm{T}}e_2\right. \\
&\quad \left. + \sqrt{1-\epsilon_1^2}\sigma_n e_2^{\mathrm{T}}u_n + \epsilon_1 e_2^{\mathrm{T}}Av_\perp + e_2^{\mathrm{T}}e_2\right)(1+\epsilon_3) \\
&= \Big(\sigma_n^2 - \epsilon_1^2\sigma_n^2 + 2\sigma_n\sqrt{1-\epsilon_1^2}e_2^{\mathrm{T}}u_n + 2\epsilon_1 e_2^{\mathrm{T}}Av_\perp + \epsilon_1^2(Av_\perp)^{\mathrm{T}}(Av_\perp) + e_2^{\mathrm{T}}e_2\Big) \\
&\quad \cdot (1+\epsilon_3) \,.
\end{aligned}$$

Note that $(Av_\perp)^{\mathrm{T}}(Av_\perp) \leq 1$ and $1.01^2 \approx 1.02$ as well as the properties of ϵ_1, ϵ_2 and e_2. We then get by manipulation

$$fl\left(\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)\right) = \left(\sigma_n^2 + \beta_1'\sigma_n\epsilon + \beta_2'\epsilon^2\right)(1+\epsilon_3) \,,$$

where

$$\left|\beta_1'\right| \leq 2.02n, \quad \left|\beta_2'\right| \leq \frac{4\alpha^2}{\tilde{g}_n^2}\sigma_n^2 + \frac{4.04n\alpha}{\tilde{g}_n} + \frac{4\alpha^2}{\tilde{g}_n^2} + 1.02n^2.$$

Based on the above results, it can be verified that

$$fl\left(\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)\right) = \sigma_n^2 + \beta_1\sigma_n\epsilon + \beta_2\epsilon^2 \, (\geq 0)$$

with

$$|\beta_1| \leq 2.02n + 1.01m\sigma_n, \quad |\beta_2| \leq \frac{4\alpha^2}{\tilde{g}_n^2}\sigma_n^2 + 2.04mn\sigma_n + \frac{4.04n\alpha}{\tilde{g}_n} + \frac{4\alpha^2}{\tilde{g}_n^2} + 1.02n^2,$$

which proves (14) and (15).

Taking the square root of $fl\left(\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)\right)$ in finite precision arithmetic, we get

$$fl\left(\left\|A\hat{v}_n\right\|\right) = \sqrt{\sigma_n^2 + \beta_1\sigma_n\epsilon + \beta_2\epsilon^2}\,(1+\epsilon_4) \quad \text{with} \quad |\epsilon_4| \leq \epsilon.$$

If $\sigma_n = \beta_3\epsilon$ with β_3 ranging from zero to a modest positive constant, say like $|\beta_2|$, we have

$$fl\left(\left\|A\hat{v}_n\right\|\right) = \sqrt{\beta_3^2 + \beta_1\beta_3 + \beta_2}\epsilon\,(1+\epsilon_4) = \beta_4\epsilon + \beta_5\epsilon^2$$

with

$$\beta_4 = \sqrt{\beta_3^2 + \beta_1\beta_3 + \beta_2}, \quad |\beta_5| \le \beta_4.$$

Therefore, we get

$$\left|fl\left(\left\|A\hat{v}_n\right\|\right) - \sigma_n\right| \le \left|\beta_4 - \beta_3\right|\epsilon + \beta_4\epsilon^2$$

by dropping the higher order term $O\left(\epsilon^2\right)$, which proves (16).

If σ_n is much bigger than ϵ, say, $\sigma_n \gg |\beta_2|\,\epsilon$, then we have

$$\begin{aligned} fl\left(\left\|A\hat{v}_n\right\|\right) &= \sigma_n\sqrt{1+\beta_1\frac{\epsilon}{\sigma_n}+\beta_2\left(\frac{\epsilon}{\sigma_n}\right)^2} + \sigma_n\left(1+\frac{\beta_1}{2}\frac{\epsilon}{\sigma_n}+\left(\left(\frac{\epsilon}{\sigma_n}\right)^2\right)\right)\epsilon_4 \\ &= \sigma_n\sqrt{1+\beta_1\frac{\epsilon}{\sigma_n}+\beta_2\left(\frac{\epsilon}{\sigma_n}\right)^2} + \sigma_n\epsilon_4 + \frac{\beta_1}{2}\epsilon\epsilon_4 + \frac{O\left(\epsilon^3\right)}{\sigma_n}. \end{aligned}$$

Dropping the highest term $O\left(\epsilon^3\right)/\sigma_n$, we obtain

$$\begin{aligned} \left|fl\left(\left\|A\hat{v}_n\right\|\right) - \sigma_n\right| &= \left|\sigma_n\sqrt{1+\beta_1\frac{\epsilon}{\sigma_n}+\beta_2\left(\frac{\epsilon}{\sigma_n}\right)^2} - \sigma_n + \sigma_n\epsilon_4 + \frac{\beta_1}{2}\epsilon\epsilon_4\right| \\ &= \left|\frac{\beta_1\epsilon + \beta_2\frac{\epsilon^2}{\sigma_n}}{\sqrt{1+\beta_1\frac{\epsilon}{\sigma_n}+\beta_2\left(\frac{\epsilon}{\sigma_n}\right)^2}+1} + \sigma_n\epsilon_4 + \frac{\beta_1}{2}\epsilon\epsilon_4\right| \\ &\le |\beta_1|\,\epsilon + |\beta_2|\,\frac{\epsilon^2}{\sigma_n} + \sigma_n\epsilon + \frac{|\beta_1|}{2}\epsilon^2, \end{aligned}$$

which proves (17) by noting that $|\beta_2|\,\frac{\epsilon^2}{\sigma_n} \ll \epsilon$ under the assumption that σ_n is much bigger than ϵ, say $\sigma_n \gg |\beta_2|\epsilon$ and by ignoring the higher order terms $|\beta_2|\,\frac{\epsilon^2}{\sigma_n} + \frac{|\beta_1|}{2}\epsilon^2$. □

It is seen that for a small σ_n, $|\beta_1|$ and $|\beta_2|$ are modest functions of m and n once σ_{n-1} is not small. Compared with (9), we see from (14) that in finite precision arithmetic roundoff errors introduce the error term $\sigma_n O(\epsilon)$ other than a term $O(\epsilon)$ when forming the Rayleigh quotient $\left(A\hat{v}_n\right)^{\mathrm{T}}\left(A\hat{v}_n\right)$. If we instead form $\hat{v}_n^{\mathrm{T}}A^{\mathrm{T}}A\hat{v}_n$ by $\hat{v}_n^{\mathrm{T}}\left(\left(A^{\mathrm{T}}A\right)\hat{v}_n\right)$, it is not hard to show that

$$\left|fl\left(\hat{v}_n^{\mathrm{T}}\left(\left(A^{\mathrm{T}}A\right)\hat{v}_n\right)\right) - \sigma_n^2\right| \le (2.02m + 1.01n)\epsilon + \left(1.02m^2 + 1.02mn + \frac{4\alpha^2}{\tilde{g}_n^2}\right)\epsilon^2. \tag{20}$$

Here the error term $O(\epsilon)$ is much greater than the right-hand side of (14) for $\sigma_n \ll 1$. So in finite precision arithmetic we should use (12) other than (13) to compute the Rayleigh quotient $\hat{v}_n^{\mathrm{T}} A^{\mathrm{T}} A \hat{v}_n$.

Comparing Theorem 3.1 with Theorem 2.1, we see that the computed $\left\| A\hat{v}_n \right\|$ is still an approximation with absolute accuracy $O(\epsilon)$ to σ_n. Therefore, our method can exploit the computed eigenvector $\hat{v}_n$ to recover the lost useful information for a small σ_n when forming $A^{\mathrm{T}} A$.

4. Rayleigh quotient matrix and computation of several small singular values in exact arithmetic

The matrix A may have more than one, say k, small singular values σ. For this case, we see from Theorem 1.1 that each corresponding v is very sensitive to roundoff errors and $\hat{v}$ may be quite inaccurate (here we drop the subscripts). So our method for computing a single small σ works no longer, and we have to treat this case separately.

Partition $V = (V_1 \;\; V_2)$ and $\Sigma = \mathrm{diag}(\Sigma_1, \; \Sigma_2)$ conformally, where the diagonal entries of Σ_2 are the k small singular values $\sigma_{n-k+1}, \; \sigma_{n-k+2}, \; \ldots, \sigma_n$ and the columns of V_2 are the corresponding right singular vectors. Therefore, we have

$$V_2^{\mathrm{T}} A^{\mathrm{T}} A V_2 = \Sigma_2^2.$$

Similarly, we partition $\hat{\Sigma} = \mathrm{diag}\left(\hat{\Sigma}_1, \; \hat{\Sigma}_2\right)$ and $\hat{V} = \left(\hat{V}_1 \;\; \hat{V}_2\right)$ accordingly. Then borrowing a theorem of [4, p. 398] and combining it with a result of [1, p. 87], we have the following result for the computed $\hat{V}_2$.

Theorem 4.1. Assume that $A^{\mathrm{T}} A + E = \left(\hat{V} + \delta\hat{V}\right) \hat{\Sigma}^2 \left(\hat{V} + \delta\hat{V}\right)^{\mathrm{T}}$ is the exact eigendecomposition of $A^{\mathrm{T}} A + E$ with $\|E\| \leq \alpha\epsilon$ and $\left\|\delta\hat{V}\right\| \leq \alpha\epsilon$. Then if $gap = \sigma_{n-k}^2 - \sigma_{n-k+1}^2 \geq 5\alpha\epsilon$, there exists an $(n-k) \times k$ matrix P satisfying

$$\|P\| \leq \frac{4\alpha\epsilon}{gap} \tag{21}$$

such that

$$\hat{V}_2 + \delta\hat{V}_2 = (V_2 + V_1 P)\left(1 + P^{\mathrm{T}} P\right)^{-1/2}. \tag{22}$$

Moreover, let $\mathcal{V}_2$ and $\hat{\mathcal{V}}_2$ denote the eigenspace and the approximate eigenspace spanned by the columns of V_2 and $\hat{V}_2$, respectively. Then if $gap > 4\alpha\epsilon$, we have

$$\sin\angle\left(\hat{\mathcal{V}}_2, \mathcal{V}_2\right) \leq \frac{2\alpha\epsilon}{gap - 4\alpha\epsilon}. \tag{23}$$

This theorem says that once σ_{n-k} is far away from zero and σ_{n-k+1} is small then $\hat{V}_2$ spans an approximate singular subspace of A with accuracy $O(\epsilon)$ though the last k small singular values may be very clustered and the k individual singular vectors may be very sensitive to errors. This suggests that we consider $\hat{V}_2$ as a whole other than its columns individually when attempting to accurately compute the k small singular values. Therefore, as a generalization of the Rayleigh quotient in Section 2,

we naturally introduce the Rayleigh quotient matrix $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$ to see what is going on.

Let μ_{n-k+i}^2, $i = 1, 2, \ldots, k$ be the eigenvalues of $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$ labeled in decreasing order. In exact arithmetic, we have the following result.

Theorem 4.2. Under the assumptions of Theorem 4.1, we have

$$\left\| \hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2 - \Sigma_2^2 \right\| \leq \gamma_1 \sigma_{n-k+1}^2 \epsilon + \gamma_2 \epsilon^2 \tag{24}$$

and

$$\left| \mu_{n-k+i}^2 - \sigma_{n-k+i}^2 \right| \leq \gamma_1 \sigma_{n-k+1}^2 \epsilon + \gamma_2 \epsilon^2, \; i = 1, 2, \ldots, k, \tag{25}$$

where

$$\gamma_1 = 2\alpha, \quad \gamma_2 = \alpha^2 + \frac{8\alpha^2}{gap} + \frac{16\alpha^2}{gap^2} + \frac{16\alpha^2}{gap^2} \sigma_{n-k+1}^2. \tag{26}$$

Proof. By the Taylor series expansion and (21), we have

$$\begin{aligned} \left(I + P^{\mathrm{T}} P\right)^{-1/2} &= I - \frac{1}{2} P^{\mathrm{T}} P + O\left(\epsilon^4\right) \\ &= I - \frac{1}{2} P^{\mathrm{T}} P \end{aligned}$$

by ignoring the higher order term $O\left(\epsilon^4\right)$. Therefore, we obtain from (22) that

$$\begin{aligned} \hat{V}_2 &= \left(V_2 + V_1 P\right) \left(I + P^{\mathrm{T}} P\right)^{-1/2} - \delta \hat{V}_2 \\ &= V_2 \left(I - \frac{1}{2} P^{\mathrm{T}} P \right) + V_1 P \left(I - \frac{1}{2} P^{\mathrm{T}} P \right) - \delta \hat{V}_2 \\ &= V_2 - \frac{1}{2} V_2 P^{\mathrm{T}} P + V_1 P - \frac{1}{2} V_1 P P^{\mathrm{T}} P - \delta \hat{V}_2 \\ &= V_2 + \delta V_2. \end{aligned} \tag{27}$$

Note that $A^{\mathrm{T}} A V_2 = V_2 \Sigma_2^2$ and $A^{\mathrm{T}} A V_1 = V_1 \Sigma_1^2$. We substitute $\hat{V}_2$ and δV_2 into $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$. Keeping the terms $O(\epsilon)$ and $O\left(\epsilon^2\right)$ but ignoring $O\left(\epsilon^3\right)$, we then get

$$\begin{aligned} \hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2 &= \left(V_2 + \delta V_2\right)^{\mathrm{T}} A^{\mathrm{T}} A \left(V_2 + \delta V_2\right) \\ &= \Sigma_2^2 - \frac{1}{2} \Sigma_2^2 P^{\mathrm{T}} P - \frac{1}{2} P^{\mathrm{T}} P \Sigma_2^2 - \Sigma_2^2 V_2^{\mathrm{T}} \delta \hat{V}_2 - \left(\delta \hat{V}_2\right)^{\mathrm{T}} V_2 \Sigma_2^2 \\ &\quad + P^{\mathrm{T}} \Sigma_1^2 P - \left(\delta \hat{V}_2\right)^{\mathrm{T}} V_1 \Sigma_1^2 P - P^{\mathrm{T}} \Sigma_1^2 V_1^{\mathrm{T}} \delta \hat{V}_2 + \left(\delta \hat{V}_2\right)^{\mathrm{T}} A^{\mathrm{T}} A \delta \hat{V}_2. \end{aligned}$$

Since $\left\| \delta \hat{V}_2 \right\| \leq \left\| \delta \hat{V} \right\| \leq \alpha \epsilon$, we obtain from (21) and above after manipulation

$$\left\| \hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2 - \Sigma_2^2 \right\| \leq \gamma_1 \sigma_{n-k+1}^2 \epsilon + \gamma_2 \epsilon^2$$

with

$$\gamma_1 = 2\alpha, \quad \gamma_2 = \alpha^2 + \frac{8\alpha^2}{gap} + \frac{16\alpha^2}{gap^2} + \frac{16\alpha^2}{gap^2} \sigma_{n-k+1}^2.$$

This proves (25) and (26).

Consider the matrix $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$ to be a perturbed matrix of the symmetric matrix Σ_2^2. Then standard perturbation theory proves (25). □

It is observed that γ_1 and γ_2 are modest functions of m and n if σ_{n-k} is not small and is well separated from σ_{n-k+1}.

Remark. When $k = 1$, Theorem 4.2 cannot reduce to Theorem 2.1. In this case, however, $\left\| \hat{V}_2 \right\| = 1$ *exactly* in the proof of Theorem 4.2 as $\hat{V}_2$ is a normalized vector, so that $\delta \hat{V}_2 = 0$ exactly. It is then easily checked that in the proof of Theorem 4.2 all the terms involving $\delta \hat{V}_2$ disappear, so the term $\gamma_1 \sigma_{n-k+1}^2 \epsilon$ does not show up in the right-hand sides of (24) and (25). Therefore, we can recover Theorem 2.1 for $k = 1$.

Using Theorem 4.2, we can prove the following result.

Theorem 4.3. 1. If $0 \le \sigma_{n-k+1} \le \epsilon$, then

$$|\mu_{n-k+1} - \sigma_{n-k+1}| \le \sqrt{\gamma_2 + 1}\epsilon. \tag{28}$$

If $\sigma_{n-k+1} > \epsilon$, then

$$|\mu_{n-k+1} - \sigma_{n-k+1}| \le (\gamma_1 + \gamma_2)\epsilon. \tag{29}$$

2. For $i = 2, 3, \ldots, k$, if $\sigma_{n-k+i} \le \sqrt{\gamma_1 \sigma_{n-k+1}^2 \epsilon + \gamma_2 \epsilon^2}$, then

$$|\mu_{n-k+i} - \sigma_{n-k+i}| \le \sqrt{2\gamma_1}\sigma_{n-k+1}\sqrt{\epsilon} + \sqrt{2\gamma_2}\epsilon; \tag{30}$$

if $\sigma_{n-k+i} > \sqrt{\gamma_1 \sigma_{n-k+1}^2 \epsilon + \gamma_2 \epsilon^2}$, then

$$|\mu_{n-k+i} - \sigma_{n-k+i}| \le \gamma_1 \frac{\sigma_{n-k+1}^2}{\sigma_{n-k+i}}\epsilon + \sqrt{\gamma_2}\epsilon \tag{31}$$

$$\le \sqrt{\gamma_1}\sigma_{n-k+1}\sqrt{\epsilon} + \sqrt{\gamma_2}\epsilon. \tag{32}$$

Proof. Part 1. If $0 \le \sigma_{n-k+1} \le \epsilon$, we get from (25) that

$$\mu_{n-k+1}^2 \le (\gamma_2 + 1)\epsilon^2 + \gamma_1 \epsilon^3,$$

which shows that

$$\mu_{n-k+1} \le \sqrt{\gamma_2 + 1}\epsilon$$

by ignoring the higher order term $\gamma_1 \epsilon^3$. Therefore, we have

$$|\mu_{n-k+1} - \sigma_{n-k+1}| \le \mu_{n-k+1} \le \sqrt{\gamma_2 + 1}\epsilon.$$

Hence assertion (28) holds.

If $\sigma_{n-k+1} > \epsilon$, we derive from (25) that

$$\begin{aligned}|\mu_{n-k+1} - \sigma_{n-k+1}| &\le \frac{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}{\mu_{n-k+1} + \sigma_{n-k+1}} \\ &\le \frac{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}{\sigma_{n-k+1}} \\ &\le \gamma_1\sigma_{n-k+1}\epsilon + \gamma_2\frac{\epsilon^2}{\sigma_{n-k+1}} \\ &\le \gamma_1\epsilon + \gamma_2\epsilon.\end{aligned}$$

Thus, assertion (29) holds.

Part 2. If $\sigma_{n-k+i} \le \sqrt{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}$ for $i = 2, 3, \ldots, k$, we get from (25) that

$$\mu_{n-k+i}^2 \le 2\gamma_1\sigma_{n-k+1}^2\epsilon + 2\gamma_2\epsilon^2,$$

yielding

$$\mu_{n-k+i} \le \sqrt{2\gamma_1\sigma_{n-k+1}^2\epsilon + 2\gamma_2\epsilon^2},$$

Therefore, we have

$$\begin{aligned}|\mu_{n-k+i} - \sigma_{n-k+i}| \le \mu_{n-k+i} &\le \sqrt{2\gamma_1\sigma_{n-k+1}^2\epsilon + 2\gamma_2\epsilon^2} \\ &\le \sqrt{2\gamma_1}\sigma_{n-k+1}\sqrt{\epsilon} + \sqrt{2\gamma_2}\epsilon,\end{aligned}$$

which shows (30).

If $\sigma_{n-k+i} > \sqrt{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}$ for $i = 2, 3, \ldots, k$, then we get from (25) that

$$\begin{aligned}|\mu_{n-k+i} - \sigma_{n-k+i}| &\le \frac{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}{\mu_{n-k+i} + \sigma_{n-k+i}} \\ &\le \frac{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}{\sigma_{n-k+i}} \\ &\le \gamma_1\frac{\sigma_{n-k+1}^2}{\sigma_{n-k+i}}\epsilon + \frac{\gamma_2\epsilon^2}{\sqrt{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}} \\ &\le \gamma_1\frac{\sigma_{n-k+1}^2}{\sigma_{n-k+i}}\epsilon + \sqrt{\gamma_2}\epsilon \\ &\le \frac{\gamma_1\sigma_{n-k+1}^2\epsilon}{\sqrt{\gamma_1\sigma_{n-k+1}^2\epsilon + \gamma_2\epsilon^2}} + \sqrt{\gamma_2}\epsilon \\ &\le \sqrt{\gamma_1}\sigma_{n-k+1}\sqrt{\epsilon} + \sqrt{\gamma_2}\epsilon,\end{aligned}$$

which proves (31) and (32). □

Some comments are made in order. First, we can always compute σ_{n-k+1} to full accuracy. Second, (30) and (32) show that μ_{n-k+i}, $i = 2, 3, \ldots, k$ are approximations to σ_{n-k+i} with accuracy $O(\epsilon)$ once σ_{n-k+1} lies between zero and the order of $\sqrt{\epsilon}$.

Third, (31) shows that (a) if the ratios $\sigma_{n-k+1}^2/\sigma_{n-k+i}$, $i = 2, 3, \ldots, k$ are modest then the μ_{n-k+i}'s are approximations to the σ_{n-k+i}'s with accuracy $O(\epsilon)$ and (b) if these ratios are large for some i then, in the worst case, (31) and (32) together indicate that the corresponding μ_{n-k+i} approximate σ_{n-k+i} with accuracy at least $\sigma_{n-k+1} O\left(\sqrt{\epsilon}\right)$, which is much smaller than $O\left(\sqrt{\epsilon}\right)$, the accuracy of $\hat{\sigma}_{n-k+i}$ computed by a backward stable algorithm working on $A^{\mathrm{T}}A$ directly. So, theoretically, this theorem suggests us to use the square roots of eigenvalues of the Rayleigh quotient matrix $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$ as new approximations to the k small singular values.

5. Computation of several small singular values in finite precision arithmetic

In comparison with accurate computation of a small σ_n, the situation now becomes more complicated and subtle in finite precision arithmetic. Similar to (12) and (13), it is critical to form the Rayleigh quotient matrix $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$ by computing

$$\left(A\hat{V}_2\right)^{\mathrm{T}} \left(A\hat{V}_2\right)$$

other than

$$\hat{V}_2^{\mathrm{T}} \left(\left(A^{\mathrm{T}}A\right) \hat{V}_2\right).$$

The information in the latter is less lost than that in the former in finite precision arithmetic.

As we have seen from Theorem 1.1, roundoff errors have a severe effect on the accuracy of a small singular value of A when a backward stable algorithm is used to compute the eigendecomposition of $A^{\mathrm{T}}A$. Yet, now we still suggest to compute the k small σ_{n-k+i}, $i = 1, 2, \ldots, k$ by taking the square roots of eigenvalues of a $k \times k$ cross-product matrix $\hat{V}_2^{\mathrm{T}} A^{\mathrm{T}} A \hat{V}_2$. It seems quite puzzling and sounds strange, and one naturally suspects the rationale and feasibility of this method in finite precision arithmetic. Fortunately, we can prove that roundoff errors affect accurate computation of σ_{n-k+i}, $i = 1, 2, \ldots, k$ in a *right* way, as the following theorem, which is a generalization of Theorem 3.1, reveals.

Theorem 5.1. In finite precision arithmetic, the Rayleigh quotient matrix $\left(A\hat{V}_2\right)^{\mathrm{T}} \left(A\hat{V}_2\right)$ satisfies

$$\left\| fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}} \left(A\hat{V}_2\right)\right) - \Sigma_2^2 \right\| \leq \eta_1' \sigma_{n-k+1} \epsilon + \eta_2' \epsilon^2, \tag{33}$$

where

$$\eta_1' = 2.02n + 1.01m\sigma_{n-k+1} + 2\sigma_{n-k+1}\left(\alpha + \frac{4\alpha}{gap}\right), \tag{34}$$

$$\eta_2' = 1.02n^2 + 2.04mn\sigma_{n-k+1} + 2.02(m\sigma_{n-k+1} + n)\left(\alpha + \frac{4\alpha}{gap}\right) + \left(\alpha + \frac{4\alpha}{gap}\right)^2. \tag{35}$$

Let $\hat{\mu}^2_{n-k+i}$, $i = 1, 2, \ldots, k$ be the computed eigenvalues of $fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)\right)$ by a backward stable algorithm. Then for $i = 1, 2, \ldots, k$ we have

$$| \hat{\mu}^2_{n-k+i} - \sigma^2_{n-k+i} | \leq \eta_1 \sigma_{n-k+1}\epsilon + \eta_2 \epsilon^2, \tag{36}$$

where

$$\eta_1 = \eta_1' + \xi\sigma_{n-k+1}, \quad \eta_2 = \eta_2' + \xi\eta_1' \tag{37}$$

with ξ being a modestly increasing function of m, n.

Proof. We get from (27) that $\hat{V}_2 = V_2 + \delta V_2$. It then follows from (21) and $\left\|\delta\hat{V}_2\right\| \leq \alpha\epsilon$ that

$$\begin{aligned} \|\delta V_2\| \| &\leq \|P\| + \left\|\delta\hat{V}_2\right\| + \frac{1}{2}\|P\|^2 + \frac{1}{2}\|P\|^3 \\ &\leq \left(\alpha + \frac{4\alpha}{gap}\right)\epsilon \end{aligned}$$

by ignoring the higher order terms $\|P\|^2$ and $\|P\|^3$.

In finite precision arithmetic, we have

$$fl\left(A\hat{V}_2\right) = A\hat{V}_2 + E_1, \tag{38}$$

where

$$\begin{aligned} \|E_1\| &\leq 1.01n\|A\|\left\|\hat{V}_2\right\|\epsilon \\ &= 1.01n\left\|\hat{V}_2 + \delta\hat{V}_2 - \delta\hat{V}_2\right\|\epsilon \\ &\leq 1.01n\left(1 + \left\|\delta\hat{V}_2\right\|\right)\epsilon \\ &\leq 1.01n(1 + \alpha\epsilon)\epsilon \\ &= 1.01n\epsilon \end{aligned} \tag{39}$$

by dropping the higher order term $1.01n\alpha\epsilon^2$.

In finite precision arithmetic, we have

$$fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)\right) = \left(A\hat{V}_2 + E_1\right)^{\mathrm{T}}\left(A\hat{V}_2 + E_1\right) + E_2, \tag{40}$$

where

$$\begin{aligned}
\|E_2\| &\le 1.01m\left\|A\hat{V}_2 + E_1\right\|^2\epsilon \\
&\le 1.01m\left(\left\|A\hat{V}_2\right\| + \|E_1\|\right)^2\epsilon \\
&= 1.01m\left(\|A(V_2+\delta V_2)\|^2 + 2\|A(V_2+\delta V_2)\|\|E_1\| + \|E_1\|^2\right)\epsilon \\
&\le 1.01m\left((\sigma_{n-k+1} + \|\delta V_2\|)^2 + 2(\sigma_{n-k+1} + \|\delta V_2\|)\|E_1\| + \|E_1\|^2\right)\epsilon \\
&= 1.01m\left(\sigma_{n-k+1}^2 + 2\sigma_{n-k+1}\|\delta V_2\| + \|\delta V_2\|^2 + 2\sigma_{n-k+1}\|E_1\| + 2\|\delta V_2\|\|E_1\| + \|E_1\|^2\right)\epsilon \\
&\le 1.01m\sigma_{n-k+1}^2\epsilon + 2.02m\sigma_{n-k+1}\left(\alpha + \frac{4\alpha}{gap}\right)\epsilon^2 + 2.04mn\sigma_{n-k+1}\epsilon^2 \qquad (41)
\end{aligned}$$

by dropping the term $O\left(\epsilon^3\right)$.

Note that

$$\begin{aligned}
\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right) &= \left(A(V_2+\delta V_2)\right)^{\mathrm{T}}\left(A(V_2+\delta V_2)\right) \\
&= \sum\nolimits_2^2 + \sum\nolimits_2^2 V_2^{\mathrm{T}}\delta V_2 + (\delta V_2)^{\mathrm{T}}V_2\sum\nolimits_2^2 + (\delta V_2)^{\mathrm{T}}A^{\mathrm{T}}A\delta V_2.
\end{aligned}$$

Therefore, we obtain from (40) that

$$\begin{aligned}
fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)\right) &= \left(A\hat{V}_2 + E_1\right)^{\mathrm{T}}\left(A\hat{V}_2 + E_1\right) + E_2 \\
&= \sum\nolimits_2^2 + \sum\nolimits_2^2 V_2^{\mathrm{T}}\delta V_2 + (\delta V_2)^{\mathrm{T}}V_2\sum\nolimits_2^2 + (\delta V_2)^{\mathrm{T}}A^{\mathrm{T}}A\delta V_2 \\
&\quad + \left(A\hat{V}_2\right)^{\mathrm{T}}E_1 + E_1^{\mathrm{T}}A\hat{V}_2 + E_1^{\mathrm{T}}E_1 + E_2 \\
&= \sum\nolimits_2^2 + E_3,
\end{aligned}$$

from which it follows that

$$\left\| fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)\right) - \Sigma_2^2\right\| \le \|E_3\|.$$

We now estimate $\|E_3\|$. First we have

$$\begin{aligned}
\left\|\left(A\hat{V}_2\right)^{\mathrm{T}}E_1\right\| &\le \left\|A\hat{V}_2\right\|\|E_1\| = \|A\,(V_2+\delta V_2)\|\,\|E_1\| \\
&\le (\sigma_{n-k+1} + \|\delta V_2\|)\,\|E_1\| \\
&\le 1.01n\sigma_{n-k+1}\epsilon + 1.01n\left(\alpha + \frac{4\alpha}{gap}\right)\epsilon^2.
\end{aligned}$$

So we get fromz (41) that

$$\begin{aligned}\|E_3\| &\le 2\sigma_{n-k+1}^2\left(\alpha+\frac{4\alpha}{gap}\right)\epsilon+\left(\alpha+\frac{4\alpha}{gap}\right)\epsilon^2+2.02n\sigma_{n-k+1}\epsilon+2.02n\left(\alpha+\frac{4\alpha}{gap}\right)\epsilon^2\\ &\quad+1.02n^2\epsilon^2+1.01m\sigma_{n-k+1}^2\epsilon+2.04mn\sigma_{n-k+1}\epsilon^2+2.02m\sigma_{n-k+1}\left(\alpha+\frac{4\alpha}{gap}\right)\epsilon^2\\ &=\left(2.02n+1.01m\sigma_{n-k+1}+2\sigma_{n-k+1}\left(\alpha+\frac{4\alpha}{gap}\right)\right)\sigma_{n-k+1}\epsilon\\ &\quad+\left(2.02m\sigma_{n-k+1}\left(\alpha+\frac{4\alpha}{gap}\right)+2.04mn\sigma_{n-k+1}+1.02n^2+2.02n\left(\alpha+\frac{4\alpha}{gap}\right)\right.\\ &\quad\left.+\left(\alpha+\frac{4\alpha}{gap}\right)^2\right)\epsilon^2.\end{aligned}$$

This completes the proof of (33), (34) and (35).

In finite precision arithmetic, the $\hat{\mu}_{n-k+i}^2$, $i=1,2,\ldots,k$ obtained by a backward stable algorithm are the exact eigenvalues of a perturbed matrix

$$fl\left(\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)\right)+E_4=\sum\nolimits_2^2+E_3+E_4,$$

where

$$\|E_4\|\le\xi\|\sum\nolimits_2^2+E_3\|\epsilon \tag{42}$$

$$\le\xi\sigma_{n-k+1}^2\epsilon+\xi\|E_3\|\epsilon \tag{43}$$

with ξ being a modest constant depending on m and n.

Since we have

$$\left|\hat{\mu}_{n-k+i}^2-\sigma_{n-k+i}^2\right|\le\|E_3\|+\|E_4\|,$$

Equations (36) and (37) follow immediately based on this relation and the estimates on $\|E_3\|$ and $\|E_4\|$. □

Remark. It is seen that η_1 and η_2 are very modestly increasing functions of m and n if gap is not near zero.

With this theorem at hand, we can present the following important theorem.

Theorem 5.2. The following results hold:

1. If $0 \le \sigma_{n-k+1} \le \frac{\eta_1+\sqrt{\eta_1^2+4\eta_2}}{2}\epsilon$, then

$$\left|\hat{\mu}_{n-k+1} - \sigma_{n-k+1}\right| \le \frac{\sqrt{3}+1}{2}\left(\eta_1 + \sqrt{\eta_1^2+4\eta_2}\right)\epsilon. \tag{44}$$

If $\sigma_{n-k+1} > \frac{\eta_1+\sqrt{\eta_1^2+4\eta_2}}{2}\epsilon$, then

$$\left|\hat{\mu}_{n-k+1} - \sigma_{n-k+1}\right| \le \left(\eta_1 + \sqrt{\eta_2}\right)\epsilon. \tag{45}$$

2. If $\sigma_{n-k+i} \le \sqrt{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}$, then for $i = 2, 3, \ldots, k$

$$\left|\hat{\mu}_{n-k+i} - \sigma_{n-k+i}\right| \le \left(1+\sqrt{2}\right)\left(\sqrt{\eta_1}\sqrt{\sigma_{n-k+1}\epsilon} + \sqrt{\eta_2}\epsilon\right). \tag{46}$$

If $\sigma_{n-k+i} > \sqrt{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}$, then for $i = 2, 3, \ldots, k$

$$\mid \hat{\mu}_{n-k+i} - \sigma_{n-k+i} \mid \le \eta_1\frac{\sigma_{n-k+1}}{\sigma_{n-k+i}}\epsilon + \sqrt{\eta_2}\epsilon \tag{47}$$

$$\le \sqrt{\eta_1}\sqrt{\sigma_{n-k+1}\epsilon} + \sqrt{\eta_2}\epsilon. \tag{48}$$

Proof. Part 1. By the assumption that $\sigma_{n-k+1}^2 \le \eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2$, an elementary analysis shows that

$$\sigma_{n-k+1} \le \frac{\eta_1 + \sqrt{\eta_1^2+4\eta_2}}{2}\epsilon.$$

Note that

$$\eta_1,\ \sqrt{\eta_2} \le \frac{\eta_1 + \sqrt{\eta_1^2+4\eta_2}}{2}.$$

So we get

$$\begin{aligned}\left|\hat{\mu}_{n-k+1}\right|^2 &\le \eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2 + \sigma_{n-k+1}^2 \\ &\le \frac{3}{4}\left(\eta_1 + \sqrt{\eta_1^2+4\eta_2}\right)^2\epsilon^2,\end{aligned}$$

yielding

$$\left|\hat{\mu}_{n-k+1}\right| \le \frac{\sqrt{3}}{2}\left(\eta_1 + \sqrt{\eta_1^2+4\eta_2}\right)\epsilon.$$

Therefore, we obtain

$$\begin{aligned}\left|\hat{\mu}_{n-k+1}-\sigma_{n-k+1}\right| &\leq \left|\hat{\mu}_{n-k+1}\right|+\sigma_{n-k+1}\\ &\leq \frac{\sqrt{3}+1}{2}\left(\eta_1+\sqrt{\eta_1^2+4\eta_2}\right)\epsilon,\end{aligned}$$

which proves (44).

If $\sigma_{n-k+1} > \eta_1\epsilon+\eta_2\epsilon^2$, then on the one hand we have

$$0<\sigma_{n-k+1}^2-\left(\eta_1\epsilon+\eta_2\epsilon^2\right)\leq \hat{\mu}_{n-k+1}^2,$$

so that $\hat{\mu}_{n-k+1} > 0$; on the other hand, an elementary analysis shows that

$$\sigma_{n-k+1} > \frac{\eta_1+\sqrt{\eta_1^2+4\eta_2}}{2}\epsilon.$$

Therefore, we obtain

$$\begin{aligned}|\hat{\mu}_{n-k+1}-\sigma_{n-k+1}| &\leq \frac{\eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2}{\mu_{n-k+1}+\sigma_{n-k+1}}\\ &\leq \frac{\eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2}{\sigma_{n-k+1}}\\ &\leq \eta_1\epsilon+\frac{2\eta_2\epsilon}{\eta_1+\sqrt{\eta_1^2+4\eta_2}}\\ &\leq \left(\eta_1+\sqrt{\eta_2}\right)\epsilon,\end{aligned}$$

which proves (45).

Part 2. If $0\leq\sigma_{n-k+i}\leq\sqrt{\eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2}$, then

$$\begin{aligned}\left|\hat{\mu}_{n-k+i}\right|^2 &\leq \eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2+\sigma_{n-k+i}^2\\ &\leq 2\eta_1\sigma_{n-k+1}\epsilon+2\eta_2\epsilon^2,\end{aligned}$$

giving

$$\left|\hat{\mu}_{n-k+1}\right|\leq\sqrt{2\eta_1}\sqrt{\sigma_{n-k+1}\epsilon}+\sqrt{2\eta_2}\epsilon.$$

So we get

$$\begin{aligned}|\hat{\mu}_{n-k+i}-\sigma_{n-k+i}| &\leq |\hat{\mu}_{n-k+i}|+\sigma_{n-k+i}\\ &\leq \left(1+\sqrt{2}\right)\sqrt{\eta_1}\left(\sqrt{\sigma_{n-k+1}\epsilon}+\sqrt{\eta_2}\epsilon\right),\end{aligned}$$

which proves (46).

If $\sigma_{n-k+i} > \sqrt{\eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2}$, then

$$0<\sigma_{n-k+i}^2-\left(\eta_1\sigma_{n-k+1}\epsilon+\eta_2\epsilon^2\right)\leq\hat{\mu}_{n-k+i}^2,$$

showing that $\mu_{n-k+i} > 0$. We thus obtain

$$\begin{aligned}
|\hat{\mu}_{n-k+i} - \sigma_{n-k+i}| &\leq \frac{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}{\mu_{n-k+i} + \sigma_{n-k+i}} \\
&\leq \frac{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}{\sigma_{n-k+i}} \\
&\leq \eta_1\frac{\sigma_{n-k+1}}{\sigma_{n-k+i}}\epsilon + \frac{\eta_2\epsilon^2}{\sigma_{n-k+i}} \\
&\leq \eta_1\frac{\sigma_{n-k+1}}{\sigma_{n-k+i}}\epsilon + \frac{\eta_2\epsilon^2}{\sqrt{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}} \\
&\leq \eta_1\frac{\sigma_{n-k+1}}{\sigma_{n-k+i}}\epsilon + \sqrt{\eta_2}\epsilon \\
&\leq \frac{\eta_1\sigma_{n-k+1}\epsilon}{\sqrt{\eta_1\sigma_{n-k+1}\epsilon + \eta_2\epsilon^2}} + \sqrt{\eta_2}\epsilon \\
&\leq \sqrt{\eta_1}\sqrt{\sigma_{n-k+1}\epsilon} + \sqrt{\eta_2}\epsilon,
\end{aligned}$$

which proves (47) and (48). □

We make some remarks in order. First, we can always compute σ_{n-k+1} to full accuracy. Second, (46) and (48) indicate that $\hat{\mu}_{n-k+i}$, $i = 2, 3, \ldots, k$ are approximations to σ_{n-k+i} with accuracy $O(\epsilon)$ once σ_{n-k+1} is of order ϵ. Third, (47) shows that if the ratios $\sigma_{n-k+1}/\sigma_{n-k+i}$ are modest, then $\hat{\mu}_{n-k+i}$, $i = 2, 3, \ldots, k$ are approximations to σ_{n-k+i} with accuracy $O(\epsilon)$; if these ratios are large for some i, then, in the worst case, the corresponding $\hat{\mu}_{n-k+i}$, $i = 2, 3, \ldots, k$ approximate σ_{n-k+i} with accuracy at least $\sqrt{\sigma_{n-k+1}\epsilon}$, much smaller than $O(\sqrt{\epsilon})$, which is the accuracy for small σ computed by a backward stable algorithm applied to $A^{\mathrm{T}}A$. Very surprisingly, however, a lot of numerical examples later will demonstrate that for small singular values that bound (47) should be applied, our method can compute all of them with full accuracy $O(\epsilon)$. This means that bound (47) may be gross estimates and can possibly be refined.

6. Algorithm and implementation

We now give more details on our method. It may happen that a computed $\hat{\mu}^2_{n-k+i}$ is negative. As is seen from Theorem 5.2, such a case may only occur when σ_{n-k+i} is at the level of ϵ, which means that A is numerically column rank deficient. At this moment, our $|\hat{\mu}_{n-k+i}|$ is also at the level of ϵ, so it is safe to treat it as zero. In implementation, based on the preceding analysis and relations (6) and (23), we first use a backward stable algorithm on $A^{\mathrm{T}}A$ to seek k satisfying

$$\left|\frac{\hat{\sigma}_{n-k}}{\hat{\sigma}_1}\right| \geq tol_1 \tag{49}$$

and

$$\left|\frac{\hat{\sigma}_{n-k+1}}{\hat{\sigma}_1}\right| \leq tol_2 \tag{50}$$

and then use our method to compute the k small singular values.

It follows from (23) that

$$gap \approx \hat{\sigma}_{n-k}^2 - \hat{\sigma}_{n-k+1}^2 \geq \left(tol_1^2 - tol_2^2\right)\hat{\sigma}_1^2. \tag{51}$$

We must require that $tol_1^2 - tol_2^2$ be far from zero, i.e., tol_1 is not far from one and tol_2 not far from zero in order to ensure that gap is comparable to σ_1^2. We may reasonably take $tol_1 = 10^{-2}$ and $tol_2 = 10^{-3}$ or 10^{-4}. From (6), condition (50) means that the computed small singular value $\hat{\sigma}_{n-k+1}$ by a backward stable algorithm on $A^{\mathrm{T}}A$ would have lost three or four digits absolute accuracy, which may be considered unacceptable in one's usual opinion. Conditions (49) and (50) together imply that small singular values should be well separated from those large ones, so that our method can work well because by Theorem 4.1 the subspace spanned by $\hat{V}_2$ is an approximately invariant right singular subspace of A with accuracy at least $4\alpha\epsilon/gap$. For a general matrix A and reasonable choices tol_1 and tol_2, however, one should be aware that these conditions may not be met simultaneously. If (49) is met but (50) is not, then A has no small singular values. So a backward stable algorithm on $A^{\mathrm{T}}A$ can compute all the singular values accurately. If (50) is met but (49) is not, our method may fail to deliver accurate small singular values as there is no essential gap between large singular values and small ones. So throughout the paper we always assume that there exists k for which (49) is fulfilled whenever (50) is met.

Now we are in a position to present our algorithm.

Algorithm. The cross-product matrix based SVD algorithm

- Step 1. Form $C = A^{\mathrm{T}}A$.
- Step 2. If both Σ and V are desired, use a backward stable algorithm (here suppose the QR algorithm is used) to compute all the eigenpairs of C. If only Σ is desired, use the QR algorithm to compute all the eigenvalues of C. Let $\hat{\Sigma}^2$ and $\hat{V}$ (if any) be the computed quantities with $\hat{\sigma}_i^2$ labeled in decreasing order.
- Step 3. Given tol_1 and tol_2, decide if there exists k satisfying both (49) and (50). If (50) is not met and either Σ and V or only Σ is desired, take the square roots of the computed eigenvalues and/or the associated eigenvectors of C as the singular values and/or the right singular vectors of A. Otherwise, if k is small and only Σ is desired, use inverse iterations to compute the k eigenvectors associated with the k small eigenvalues of C and then reorthogonalize them; if only Σ is desired but k is relatively big, simply use the QR algorithm second time to compute all the eigenvectors of C. Then use the Rayleigh quotient matrix to compute approximations $\hat{\mu}_{n-k+i}$ to the k small singular values σ_{n-k+i}, $i = 1, 2, \ldots, k$.

Let us describe practical implementations on the proposed algorithm step by step.

Step 1. Suppose that A is real. Since C is symmetric, we only need to form its upper triangular part in about mn^2 flops.

Step 2. If both Σ and V are desired, the QR algorithm costs about $8\frac{2}{3}n^3$; if only Σ is desired, the QR algorithm finds all the eigenvalues of $A^{\mathrm{T}}A$ using $\frac{4}{3}n^3$ flops [3, p. 211].

Step 3. Suppose only Σ is desired. As Step 3 says, we first use the QR algorithm to compute all the eigenvalues of C and determine k in about $\frac{4}{3}n^3$ flops. Then we suggest the following: When k is small, say $k < \frac{n}{4}$ (e.g., see [2, p. 89]), we use inverse iterations

for computing the k eigenvectors and then reorthogonalize them in $O(kn^2)$ flops. Demmel [3, p. 231] warned that for clustered eigenvalues, even though we use inverse iterations to compute the corresponding eigenvectors and then reorthogonalize them, "there is no guarantee that the computed eigenvectors are accurate or orthogonal" and "the trouble is that after reorthogonalizating a set of nearly dependent $\hat{q}_k$ (i.e., the computed eigenvectors), cancellation may mean (that) some computed eigenvectors consist of little more than roundoff errors". Fortunately, in contrast to this superficial analysis, Stewart [11] has recently proved that this is not true and instead the orthogonalization does not cause the quality of the computed eigenvectors to deteriorate, i.e., residual of each of the reorthogonalized eigenvectors is small to working precision $O(\epsilon)$. Write the reorthogonalized approximate eigenvectors as $\hat{V}_2$. Define

$$R = C\hat{V}_2 - \hat{V}_2\hat{\Sigma}_2^2.$$

Then Stewart's result tell us that $\|R\| = \|C\|O(\epsilon) = O(\epsilon)$ and $\hat{V}_2^{\mathrm{T}}\hat{V}_2 = I + O(\epsilon)$.

Denote by $\sigma_{\min}(X)$ the smallest singular value of a matrix X. It then follows Theorem 3.8 of [10, p. 252] and Theorem 1.1 that

$$\begin{aligned}
\sin\angle\left(\hat{\mathcal{V}}_2, \mathcal{V}_2\right) &\le \frac{\|R\|}{\left|\sigma_{n-k}^2 - \hat{\sigma}_{n-k+1}^2\right| \sigma_{\min}\left(\hat{V}_2\right)} \\
&\le \frac{\|R\|}{\left(\sigma_{n-k}^2 - \sigma_{n-k+1}^2 - \alpha\epsilon\right) \sigma_{\min}\left(\hat{V}_2\right)} \\
&= \frac{O(\epsilon)}{(gap - \alpha\epsilon)(1 + O(\epsilon))} \\
&= \frac{O(\epsilon)}{gap - \alpha\epsilon}.
\end{aligned}$$

This indicates that although $\hat{v}_{n-k+i}$, $i = 1, 2, \ldots, k$ may be inaccurate approximations to the v_{n-k+i}'s, the subspace spanned by $\hat{V}_2$ is an approximation to $\mathcal{V}_2$ with working precision $O(\epsilon)$ once the separation gap is not small, i.e., the k small singular values are well separated from the large ones of A. Hence it is safe to use inverse iterations to numerically compute an orthonormal basis $\hat{V}_2$ of $\hat{\mathcal{V}}_2$, which spans an approximately invariant subspace with accuracy $O(\epsilon)$.

When only Σ is desired but k is relatively big, say $k \ge \frac{n}{4}$, Step 3 uses the QR algorithm second time to compute all the eigenpairs of $A^{\mathrm{T}}A$ in total $8\frac{2}{3}n^3$ flops [3]. In any event, we then first form $A\hat{V}_2$ in $2mnk$ flops and $\left(A\hat{V}_2\right)^{\mathrm{T}}\left(A\hat{V}_2\right)$ in mk^2 flops and compute its eigenvalues in $\frac{4}{3}k^3$ flops. Table 1 lists costs of our algorithm, the standard Golub–Reinsch SVD algorithm and the Chan SVD algorithm [4, p. 254], assuming that the QR algorithm is used in our algorithm.

Some comments are in order for table 1 when standard Givens transformations are used. First, when $k \ll n$, say $k < \frac{n}{4}$, and $m \approx n$, our algorithm is a little cheaper than the Golub–Reinsch algorithm but much cheaper than the Chan algorithm. Second, when $k \ll n$ and $m \gg n$, our algorithm is much cheaper than the Golub–Reinsch and Chan algorithms. Third, when $k \ge \frac{n}{4}$ and $m \approx n$, our algorithm is more expensive than the other two algorithms. Fourth, when $k \ge \frac{n}{4}$ and $m \gg n$, our algorithm uses fewer flops than the other two algorithms unless $k \approx n$. In summary, if $k \ll n$, then our algorithm is and can be much cheaper than the other two algorithms.

Table 1 A comparison of flops. Note the upper bound of $O\left(nk^2\right) \ll n^3$ if $k \leq \frac{n}{4}$ and $O\left(nk^2\right) = 8\frac{2}{3}n^3$ if $k > \frac{n}{4}$.

Desired	Our algorithm	Golub–Reinsch SVD	Chan SVD
Σ	$mn^2 + 2mnk + mk^2 + \frac{4}{3}n^3 + \frac{4}{3}k^3 + O(nk^2)$	$4mn^2 - 4n^3/3$	$2mn^2 + 2n^3$
$\Sigma,\ V$	$mn^2 + 2mnk + mk^2 + 8\frac{2}{3}n^3 + \frac{4}{3}k^3$	$4mn^2 + 8n^3$	$2mn^2 + 11n^3$

7. Two important applications

7.1. Efficient and accurate computation of refined Ritz vectors

In practice, k is frequently very small and may often equal one, so that our algorithm can be greatly advantageous. A typical application of our algorithm arises from accurate and efficient computation of refined Ritz vectors where a number of SVDs of the $m \times n$ matrices $(A - \theta I)W$ are required for several different shifts θ [6–8, 11, 14] and only one (if any) small singular value is present for each matrix. Here W is an orthonormal basis of a given projection subspace, n is the dimension of the subspace, and the θ are certain approximate eigenvalues obtained by a projection method. In practice, m is much larger than n, and we may well have $m \geq 100n$. What we only need is the smallest singular value and the corresponding right singular vector of each $(A - \theta I)W$ [7, 8, 12, 14], which returns the refined Ritz vector by premultiplying W. So we can compute the refined Ritz vectors accurately and decide when a refined projection algorithm converges by working on a number of cross-product matrices. For this kind of problems, since AW is available, we only need to form all the cross-product matrices $W^{\mathrm{T}}(A - \theta I)^{\mathrm{T}}(A - \theta I)W$ in roughly total mn^2 flops. As a result, our algorithm is enormously cheaper than any other standard SVD algorithms for several θ.

7.2. Accurate computation of the SVD by appending or deleting a row

As is well known and seen from [2], when some small singular values are present, it appears hard to accurately update the other two algorithms for computing the small singular values of a new matrix by appending to or deleting a row from A at low cost. A great advantage of our algorithm is that it very nicely fits into this purpose, as illustrated below. We only describe the case that appends a row to A. The case deleting a row from A is treated similarly.

Let $B^{\mathrm{T}} = \left(A^{\mathrm{T}}\ a\right)$. Then we have

$$\begin{aligned} B^{\mathrm{T}}B &= A^{\mathrm{T}}A + aa^{\mathrm{T}} \\ &= V\Sigma^2 V^{\mathrm{T}} + aa^{\mathrm{T}} \\ &= V\left(\Sigma^2 + cc^{\mathrm{T}}\right)V^{\mathrm{T}} \ \text{ with } \ c = V^{\mathrm{T}}a \\ &= \tilde{V}\tilde{\Sigma}^2\tilde{V}^{\mathrm{T}}, \end{aligned}$$

where $\tilde{V} = VX = V(X_1\ \ X_2)$ with X being the matrix of eigenvectors of $\Sigma^2 + cc^{\mathrm{T}}$ and X_2 the eigenvectors associated with the k small singular values of B. So, formally, we only need to solve the eigenproblem $\Sigma^2 + cc^{\mathrm{T}}$ of rank one modification. A reliable

and efficient divide and conquer algorithm can compute all the eigenpairs of the modified matrix in only $O(n^2)$ flops [3].

A most commonly used algorithm for computing the SVD of B from that of A is to compute the eigenpairs of $\Sigma^2 + cc^{\mathrm{T}}$ using the divide and conquer algorithm and then return $\tilde{\Sigma}$ and $\tilde{V}$; see [2]. This updating scheme is very cheap. As we have seen, however, it cannot compute small singular values of B accurately. Let $\hat{X}_2$ be the computed eigenvectors associated with the k small singular values. In terms of our theory, instead of this usual algorithm, we use our new algorithm to compute small eigenvalues of $\Sigma^2 + cc^{\mathrm{T}}$ and return accurate small singular values of B. The formation of a $k \times k$ Rayleigh quotient matrix $\hat{X}_2^{\mathrm{T}} \left(\Sigma^2 + cc^{\mathrm{T}}\right) \hat{X}_2$ needs $3nk^2$ flops and computation of its all eigenvalues uses $\frac{4}{3}k^3$ flops. Therefore, our algorithm accurately computes the singular values of B from those of A in $O\left(n^2\right) + 3nk^2 + \frac{4}{3}k^{P3}$ flops. If $\tilde{V} = VX$ is also desired, then we need additional $2n^3$ flops. Therefore, we only use about $2n^3$ flops for $k \ll n$ and $6\frac{1}{3}n^3$ flops for $k \approx n - 1$ to get $\tilde{\Sigma},\ \tilde{V}$ from $\Sigma,\ V$. This is cheaper than other updating algorithms [2].

8. Variants of the cross-product matrix based algorithm

We have discussed how the proposed algorithm can be used to accurately compute singular values and right singular vectors, but we have not touched the issue of how to obtain left singular vectors accurately and efficiently. Of course, one may compute U_1 by normalizing the columns of the computed $A\hat{V}$. However, it is easily verified (and also well known) that such an approach is unstable when A has small singular values since the computed left singular vectors this way can be very inaccurate and far from orthonormal. Alternatively, Golub and Van Loan's book [4, p. 452] suggests to use the QR decomposition with column pivoting on AV to get the left singular vectors U:

$$Q^{\mathrm{T}}(AV)\Pi = R.$$

Mathematically, if the columns of V are exact right singular vectors, then R is diagonal as AV is orthogonal, so that $Q = U$ are the left singular vectors. Numerically, however, there are two reasons we do not prefer this approach: First, R would be far from diagonal if V were replaced by the computed $\hat{V}$ and A had some small singular values, so that Q, though numerically orthonormal, would be far from U. Second, if we are only required to compute U_1 other than U, then the flops of this algorithm can be much higher than the Golub–Reinsch and Chan algorithms.

A natural choice is to form another cross-product matrix AA^{T} and compute its eigendecomposition to return U. Although this approach can compute U accurately, it is too expensive when $m \gg n$. Also, it cannot get U_1 without knowing U. So generally we must abandon this choice.

Fortunately, depending on which parts of the SVD are required, we can compute them efficiently and accurately, as described below.

As a first step, depending on what we want, we first compute the full QR decomposition

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} = (Q_1 \quad Q_2) \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{52}$$

or the thin QR decomposition

$$A = Q_1 R. \tag{53}$$

Householder transformations are used to compute (52), and the decomposition requires $2n^2(m - n/3)$ flops if Q is not formed explicitly and $4\left(m^2n - mn^2 + n^3/3\right)$ flops if Q is formed explicitly [4, p. 225]. The modified Gram–Schmit orthogonalization method with iterative refinement is used to compute (53), and the decomposition requires $2mn^2$ flops. (In extreme case, it requires $4mn^2$ flops if one reorthogonalization is used at each step). If Householder transformations are used to compute (53), the cost is $4mn^2 - \frac{4}{3}n^3$ flops [4, p. 232]. In the sequel, we always suppose the cost of computing (53) is $2mn^2$flops.

Let

$$R = U_R \Sigma_R V_R^{\mathrm{T}} \tag{54}$$

be the SVD of the $n \times n$ upper triangular matrix R. Then we have from (52)

$$A = (Q_1 U_R \;\; Q_2) \begin{pmatrix} \Sigma_R \\ 0 \end{pmatrix} V_R^{\mathrm{T}},$$

from which and 1) it follows that

$$U_1 = Q_1 U_R, \;\; U = (Q_1 V_R \;\; Q_2), \;\; \Sigma = \Sigma_R, \;\; V = V_R.$$

Therefore, we can get the SVD of A by computing that of R, which, in contrast to the Golub–Reinsch algorithm, is also the very first step of the Chan algorithm. Since two decompositions (52) and (53) are numerically backward stable, i.e., the computed R is the exact upper triangular matrix of the QR decomposition of a slightly perturbed matrix $A + E$ with $\|E\| \leq \mu\epsilon$ and μ a modest constant depending on m and n. So by standard perturbation theory, the differences between the singular values of the computed R and those of A are bounded above by $\mu\epsilon$. Therefore, we can get an accurate SVD of A by accurately computing the SVD of the computed R.

Now, based on the previous theory and the proposed algorithm, we can form the $n \times n$ cross-product matrices $R^{\mathrm{T}}R$ and RR^{T} and present several variants of the algorithm to compute some or all of Σ, U, V. There are altogether six cases, which are considered one by one below.

Case 1. Computation of Σ.
We first compute the full QR decomposition (52) without forming Q explicitly in $2mn^2 - \frac{2}{3}n^3$ flops. Then replacing $A^{\mathrm{T}}A$ by $R^{\mathrm{T}}R$ and A by R accordingly, we use the proposed algorithm to compute Σ, which now requires $1\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3 + O\left(nk^2\right)$ flops. Total flops are

$2mn^2 + 1\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3 + O\left(nk^2\right)$. Thus, the flops are approximately equal to $2mn^2 + 1\frac{2}{3}n^3$ if $k \ll n$ and no more than $2mn^2 + 14\frac{2}{3}n^3$ if $k \approx n$.

Recall that in this case the Golub–Reinsch algorithm requires $4mn^2 - \frac{4}{3}n^3$ flops and the Chan algorithm requires $2mn^2 + 2n^3$ flops [4, p. 254]. So if $k \ll n$, our variant is a little cheaper than the Chan algorithm and can be much cheaper than the Golub–Reinsch algorithm when $m \gg n$.

Case 2. Computation of Σ and V.

We first compute the full QR decomposition (52) without forming Q explicitly in $2mn^2 - \frac{2}{3}n^3$ flops. Then replacing $A^{\mathrm{T}}A$ by $R^{\mathrm{T}}R$ and A by R accordingly, we can use the proposed algorithm to compute Σ and V in $9\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$ flops. So total flops are $2mn^2 + 9n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$, which are approximately equal to $2mn^2 + 9n^3$ if $k \ll n$ and no more than $2mn^2 + 13n^3$ if $k \approx n$.

Recall that in this case the Golub–Reinsch algorithm requires $4mn^2 + 8n^3$ flops and the Chan algorithm requires $2mn^2 + 11n^3$ flops [4, p. 254]. So if $k \ll n$, our variant is much cheaper than the Golub–Reinsch algorithm and a little cheaper than the Chan algorithm when $m \gg n$ and it is a little cheaper than the other two algorithms when $m \approx n$. If $k \approx n$, our variant is much cheaper than the Golub–Reinsch algorithm and a little more flops than the Chan algorithm when $m \gg n$, and it is a little more expensive than the other two algorithms when $m \approx n$.

Case 3. Computation of Σ and U_1.

We first compute the thin QR decomposition $A = Q_1 R$ in $2mn^2$ flops. Then replacing $A^{\mathrm{T}}A$ by RR^{T} and A by R^{T} accordingly, we can use the proposed algorithm to compute Σ and U_R in $9\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$. Finally, we form $U_1 = Q_1 U_R$ in $2mn^2$ flops. So total flops are $4mn^2 + 9\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$, which are approximately equal to $4mn^2 + 9\frac{2}{3}n^3$ if $k \ll n$ and no more than $4mn^2 + 14n^3$.

Recall that in this case the Golub–Reinsch algorithm uses $14mn^2 - 2n^3$ flops and the Chan algorithm uses $6mn^2 + 11n^3$ flops [4, p. 254]. So for any $k < n$, our algorithm is much cheaper than the other two algorithm for $m \gg n$. When $m \approx n$, our variant is more expensive than the Golub–Reinsch algorithm, and it is considerably cheaper than the Chan algorithm if $k \ll n$ and a very little more expensive than the Chan algorithm if $k \approx n$.

Case 4. Computation of Σ and U.

We first compute the full QR decomposition (52) in $4\left(m^2n - mn^2 + \frac{n^3}{3}\right)$. Then replacing $A^{\mathrm{T}}A$ by RR^{T} and A by R^{T} accordingly, we can use the proposed algorithm to compute Σ and U_R in $9\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$. Finally, we form $U_1 = Q_1 U_R$ in $2mn^2$ flops to get $U = (U_1 \;\; Q_2)$. Thus, total flops are $4m^2n - 2mn^2 + 10n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$, which are approximately equal to $4m^2n - 2mn^2 + 10n^3$ if $k \ll n$ and no more than $4m^2n - 2mn^2 + 14\frac{1}{3}n^3$ if $k \approx n$.

Recall that in this case the Golub–Reinsch algorithm requires $4m^2n + 8mn^2$ flops and the Chan algorithm requires $4m^2n + 13n^3$ flops [4, p. 254]. So our variant is always cheaper than them for any $k < n$.

Case 5. Computation of Σ, U_1, V.

There are a few steps to be done. First, we compute the thin QR decomposition $A = Q_1 R$ in $2mn^2$ flops. Second, replacing $A^{\mathrm{T}}A$ by $R^{\mathrm{T}}R$ and A by R accordingly, we can use the proposed algorithm to compute Σ and V in $9\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$. Third, we form RR^{T} in n^3 flops and compute only the eigenvectors U_R by the QR algorithm in $8\frac{2}{3}n^3$ flops. Fourth, we form $U_1 = Q_1 U_R$ in $2mn^2$ flops. Total flops are thus $4mn^2 + 19\frac{1}{3}n^3 + 2nk^2 + nk^2 + \frac{4}{3}k^3$, which are approximately equal to $4mn^2 + 19\frac{1}{3}n^3$ if $k \ll n$ and no more than $4mn^2 + 23\frac{2}{3}n^3$ flops.

Recall that in this case the Golub–Reinsch algorithm uses $14mn^2 + 8n^3$ flops and the Chan algorithm uses $6mn^2 + 20n^3$ flops [4, p. 254]. So our variant is much cheaper than them when $m \gg n$. Otherwise, our variant uses roughly the same flops as them.

Case 6. Computation of Σ, U, V.

The only difference with Case 5 is the first step, which is now replaced by the full QR decomposition (52). The cost of this step is $4\left(m^2n - mn^2 + \frac{n^3}{3}\right)$ flops. So total flops are now $4m^2n - 2mn^2 + 18\frac{2}{3}n^3 + 2n^2k + nk^2 + \frac{4}{3}k^3$, which are approximately equal to $4m^2n - 2mn^2 + 18\frac{2}{3}n^3$ if $k \ll n$ and no more than $4m^2n - 2mn^2 + 23n^3$.

Recall that in this case the Golub–Reinsch algorithm requires $4m^2n + 8mn^2 + 9n^3$ flops and the Chan algorithm requires $4m^2n + 22n^3$ flops [4, p. 254]. So our variant is cheaper than these two algorithms for any $k < n$.

In summary, our variants are and can be much cheaper than the Golub–Reinsch and Chan algorithms when $k \ll n$ or $m \gg n$.

9. Numerical examples

We have tested a number of typical examples to confirm the accuracy of our algorithm on an Intel Pentium 450HMZ using MATLAB 5.0 with $\epsilon = 1.11 \times 10^{-16}$. Since the function `svd.m` in MATLAB computes singular values with full absolute accuracy $O(\epsilon)$, as a reference, we assume that it returns "exact" singular values σ_i, $i = 1, 2, \ldots, n$ of A. Let $\hat{\sigma}_i$ and $\hat{\mu}_i$, $i = 1, 2, \ldots, n$ denote the computed singular values obtained by the function `eig.m` and by our algorithm, respectively.

Example 1. We tested a very small but typical matrix taken from [3, p. 241]. The matrix is

$$A = \begin{pmatrix} 1 & 1 \\ 0 & \sqrt{\eta} \end{pmatrix},$$

which has singular values $\sqrt{2}$ and $\sqrt{\eta/2}$. Take $\eta = \epsilon/2$. Then $fl(1 + \eta) = 1$. So

$$A^{\mathrm{T}}A = \begin{pmatrix} 1 & 1 \\ 1 & 1+\eta \end{pmatrix} \text{ is rounded to } \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},$$

an exactly singular matrix in finite precision arithmetic! We got the smaller singular value $\hat{\sigma}_2 = 0$, a very inaccurate approximation to the exact $\sigma_2 = 7.450580596923828e - 9$. However, in finite precision arithmetic, our algorithm computed exactly $fl\left(\|A\hat{v}_2\|\right) = \sigma_2$.

Example 2. We consider the following famous Kahan matrix from [4, p. 260], which makes the QR decomposition with column pivoting fail to detect the smallest singular value σ_n:

$$A_n = \begin{pmatrix} 1 & -c & -c & \dots & \dots & -c \\ 0 & s & -cs & \dots & \dots & -cs \\ & & s^2 & \dots & \dots & -cs^2 \\ & & & \ddots & & \vdots \\ & & & & s^{n-2} & -cs^{n-2} \\ & & & & & s^{n-1} \end{pmatrix}$$

with $c = 0.2$ and $s = \sqrt{1 - c^2}$. We tested our algorithm and the `eig.m` for $n = 50,\ 100,\ 120,\ 150$ and 200.

For $n = 50$, the `svd.m` gave $\sigma_{50} = 9.287521172380022e - 5$. We got

$$\hat{\sigma}_{50} = 9.287521133803280e - 5, \quad fl\left(\left\|A\hat{v}_{50}\right\|\right) = 9.287521172381082e - 5.$$

and

$$\left|\hat{\sigma}_{50} - \sigma_{50}\right| = 3.86 \times 10^{-13}, \quad \left|fl\left(\left\|A\hat{v}_{50}\right\|\right) - \sigma_{50}\right| = 1.1 \times 10^{-17} = O(\epsilon).$$

We see $\hat{\sigma}_{50}$, though quite satisfactory, is less accurate and has the absolute error considerably bigger than ϵ, while our algorithm computed σ_{50} accurately within ϵ.

For $n = 100$, the `svd.m` gave $\sigma_{100} = 3.678056462652092e - 9$. We got

$$\hat{\sigma}_{100} = 6.220666056667578e - 9, \quad fl\left(\left\|A\hat{v}_{100}\right\|\right) = 3.678056463159903e - 9.$$

and

$$\left|\hat{\sigma}_{100} - \sigma_{100}\right| = 2.54 \times 10^{-8} = O\left(\sqrt{\epsilon}\right), \quad \left|fl\left(\left\|A\hat{v}_{100}\right\|\right) - \sigma_{100}\right| = 5.1 \times 10^{-19} = O(\epsilon).$$

We see $\hat{\sigma}_{100}$ only had accuracy $O(\sqrt{\epsilon})$ and was quite inaccurate but our algorithm computed σ_{100} perfectly.

For $n = 120$, the `svd.m` gave $\sigma_{120} = 6.37831268980334e - 11$. We got

$$\hat{\sigma}_{120} = (5.060711839317277e - 9)i, \quad fl\left(\left\|A\hat{v}_{120}\right\|\right) = 6.378312620126201255 75e - 11.$$

and

$$\left|\hat{\sigma}_{120} - \sigma_{120}\right| = 5.06 \times 10^{-9} = O\left(\sqrt{\epsilon}\right), \quad \left|fl\left(\left\|A\hat{v}_{120}\right\|\right) - \sigma_{120}\right| = 3.6 \times 10^{-19} = O(\epsilon).$$

We see $\hat{\sigma}_{120}$ is even a small purely imaginary number while our algorithm computed σ_{100} with accuracy ϵ.

For $n = 150$, the `svd.m` gave $\sigma_{150} = 1.456584578269958e - 13$. Since $\sigma_1 = \|A_{150}\| \approx 10.57$ and $\sigma_{150} = \gamma\sigma_1\epsilon$ with $\gamma \leq n$, A_{150} is singular in finite precision arithmetic. We got

$$\hat{\sigma}_{150} = 4.424639119368757e - 9, \quad fl\left(\left\|A\hat{v}_{150}\right\|\right) = 1.456807544552100e - 13.$$

and

$$\left|\hat{\sigma}_{150} - \sigma_{150}\right| = 4.42 \times 10^{-9} = O\left(\sqrt{\epsilon}\right), \quad \left|fl\left(\left\|A\hat{v}_{150}\right\|\right) - \sigma_{150}\right| = 2.7 \times 10^{-17} = O(\epsilon).$$

We see $\hat{\sigma}_{150}$ only had accuracy $O\left(\sqrt{\epsilon}\right)$, while our algorithm computed σ_{150} exactly in finite precision arithmetic.

For $n = 200$, the `svd.m` gave $\sigma_{200} = 1.5775535090217991e-18$, which is already at the level of ϵ. So A_{200} is singular in finite precision arithmetic. Note $\sigma_1 = \|A\| \approx 12.68$.

$$\hat{\sigma}_{200} = 1.058476490289012e-8, \quad fl(\|A\hat{v}_{200}\|) = 1.135738540436608e-14$$

and

$$|\ \hat{\sigma}_{200} - \sigma_{200}\ | = 1.1 \times 10^{-8} = O(\sqrt{\epsilon}), \quad |\ fl(\|A\hat{v}_{200}\|) - \sigma_{200}\ | = 1.1 \times 10^{-14} \approx 10\|A\|\epsilon.$$

we see our algorithm computed σ_{200} with accuracy $O(\epsilon)$.

Example 3. Consider the $(n+1) \times n$ matrix A whose first row and first subdiagonal are nonzero and all other entries are zero:

$$A = \begin{pmatrix} 1 & 1 & 1 \ldots & 1 \\ \frac{\sqrt{\epsilon}}{n} & & & \vdots \\ & \frac{2\sqrt{\epsilon}}{n} & \ldots & \vdots \\ & & \ddots & \vdots \\ & & & \sqrt{\epsilon} \end{pmatrix}.$$

A has one large singular value and $n-1$ clustered small singular values. Take $n = 100$. We computed σ_i and $\hat{\sigma}_i$, $i = 1, 2, \ldots, 100$ by the `svd.m` and the `eig.m`, respectively. We found that all the $\hat{\sigma}_i$, $i = 2, 3, \ldots, 100$ are inaccurate, 37 of which are imaginary. In fact, we had

$$\max_{i=2,3,\ldots,100} |\ \hat{\sigma}_i - \sigma_i\ | = 2.5 \times 10^{-7}, \quad \min_{i=2,3,\ldots,100} |\ \hat{\sigma}_i - \sigma_i\ | = 3.3 \times 10^{-10}.$$

However, our algorithm worked very reliably, and we found

$$\max_{i=2,3,\ldots,100} |\ \hat{\mu}_i - \sigma_i\ | = 5.1 \times 10^{-21} = O(\epsilon).$$

Example 4. We constructed a 100×50 matrix A whose singular values were known explicitly by taking $A = U\Sigma V^{\mathrm{T}}$ as follows: $\sigma_i = 1/i$ for $i = 1 : 47$, $\sigma_{48} = 10^{-3}$, $\sigma_{49} = 10^{-6}$, $\sigma_{50} = 10^{-12}$, U and V are generated randomly by the function `orth.m`. The matrix A has three small singular values, and the smallest singular value is very small. We tested our algorithm by taking $tol_1 = 10^{-2}$ and $tol_2 = 10^{-3}$. Our algorithm determined $k = 3$, and the three exact small singular values computed by the `svd.m` were

$$\sigma_{48} = 0.001000000000000,$$
$$\sigma_{49} = 9.999999999977340e-7,$$
$$\sigma_{50} = 9.999998989483858e-13.$$

The computed results were

$$\hat{\sigma}_{48} = 9.999999999938196e-4,$$
$$\hat{\sigma}_{49} = 9.999937496500932e-7,$$
$$\hat{\sigma}_{50} = 2.703165039356417e-9$$

and

$$\hat{\mu}_{48} = 0.001000000000000,$$
$$\hat{\mu}_{49} = 1.000000000000649e-6,$$
$$\hat{\mu}_{50} = 1.000013334730396e-12.$$

The errors were

$$|\hat{\sigma}_{48} - \sigma_{48}| = 6.2 \times 10^{-9} = O(\sqrt{\epsilon}),$$
$$|\hat{\sigma}_{49} - \sigma_{49}| = 6.3 \times 10^{-12},$$
$$|\hat{\sigma}_{50} - \sigma_{50}| = 2.3 \times 10^{-9} = O(\sqrt{\epsilon})$$

and

$$|\hat{\mu}_{48} - \sigma_{48}| = 1.3 \times 10^{-17} = O(\epsilon),$$
$$|\hat{\mu}_{49} - \sigma_{49}| = 2.9 \times 10^{-18} = O(\epsilon),$$
$$|\hat{\mu}_{50} - \sigma_{50}| = 1.8 \times 10^{-17} = O(\epsilon).$$

Clearly, our algorithm worked as reliably and accurately as the `svd.m`. It is seen from this example that bounds (46) and (47) were too conservative and they overestimated errors quite much. In fact, the bounds tell us that $|\hat{\mu}_{49} - \sigma_{49}| = O(10^{-13})$, $|\hat{\mu}_{50} - \sigma_{50}| = O(10^{-10})$.

Example 5. We used the function `rand` to generate a 100×30 matrix A as follows:

$$A = \left(\texttt{rand(100, 27)}\ \ \texttt{rand(100, 1)}\epsilon^{1/4}\ \ \texttt{rand(100, 1)}\sqrt{\epsilon}\ \ \texttt{rand(100, 1)} \times 100\epsilon\right)$$

with $\|A\| \approx 26.165$. Such a matrix has three small singular values σ_{28}, σ_{29} and σ_{30}. In test, we took $tol_1 = 10^{-2}$ and $tol_2 = 10^{-3}$, which determined $k = 3$. The exact small singular values were

$$\sigma_{28} = 2.973849952448024e-4,$$
$$\sigma_{29} = 3.638262812082207e-8,$$
$$\sigma_{30} = 5.822652131502450e-14.$$

The computed results were

$$\hat{\sigma}_{28} = 2.973849955245767e-4,$$
$$\hat{\sigma}_{29} = 3.703680977022078e-7,$$
$$\hat{\sigma}_{30} = 1.724557290795108e-8$$

and

$$\hat{\mu}_{28} = 2.973849952447045e - 4,$$
$$\hat{\mu}_{29} = 3.638262812084353e - 8,$$
$$\hat{\mu}_{30} = 0.$$

The errors were

$$| \hat{\sigma}_{28} - \sigma_{28} | = 2.8 \times 10^{-13},$$
$$| \hat{\sigma}_{29} - \sigma_{29} | = 3.3 \times 10^{-7} = O(\sqrt{\epsilon}),$$
$$| \hat{\sigma}_{30} - \sigma_{30} | = 1.7 \times 10^{-8} = O(\sqrt{\epsilon})$$

and

$$| \hat{\mu}_{28} - \sigma_{28} | = 9.8 \times 10^{-17} = O(\epsilon),$$
$$| \hat{\mu}_{29} - \sigma_{29} | = 2.2 \times 10^{-20} = O(\epsilon),$$
$$| \hat{\mu}_{30} - \sigma_{30} | = 5.8 \times 10^{-14} \approx n\|A\|\epsilon = O(\epsilon),$$

respectively.

For this matrix the smallest singular value σ_{30} can be considered to be at the level of ϵ. Although Theorem 5.2 predicts $| \hat{\mu}_{29} - \sigma_{29} | = O(10^{-12})$ and $| \hat{\mu}_{30} - \sigma_{30} | = O(10^{-10})$, our algorithm computed the three small singular values with accuracy $O(\epsilon)$.

We have also tested a lot of examples generated randomly. The matrices tested were constructed to have one or several small singular values ranging from 10^{-4} to the order of ϵ. For all the tests, we found that our algorithm worked as reliably and accurately as the `svd.m`. These examples showed that the bounds in Theorem 5.2 were too conservative and can be possibly refined.

Finally, for the examples tested, we mention that the variants proposed in Section 8 computed small singular values with similar accuracy.

10. Concluding remarks

Using the cross-product matrix $A^{\mathrm{T}}A$, we have developed a new algorithm to accurately compute small singular values of A. In order to compute some or all of Σ, U, V, we have proposed several variants of the algorithm. A detailed theoretical analysis and typical numerical examples have shown that the algorithm works very reliably and accurately whenever small singular values are well separated from those large ones. Our algorithm is not only cheaper than and as reliable as the Golub–Reinsch and Chan SVD algorithms, but also can be easily used to update or downdate a new Σ and V efficiently and accurately when a row is appended to or deleted from A. In the meanwhile, it has an important application in refined projection methods for large scale matrix eigenproblems. So it should be very appealing in many applications. A minor disadvantage of our algorithm is that it must store both A and $A^{\mathrm{T}}A$. This needs to save extra $n(n+1)/2$ elements if one only stores the upper triangular part of $A^{\mathrm{T}}A$.

Acknowledgements I am indebted to Professor G. W. Stewart for stimulating and fruitful discussions on the computation of refined Ritz vectors [9]. This made me think over accurate computation of small singular values using the cross-product matrix $A^{\mathrm{T}}A$ and finally produce a preliminary version of this paper. My further probe was attributed to Professors Haesun Park and Tony Chan whose many constructive suggestions led me to make a quantitative analysis on most of the bounds.

References

[1] Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S., Sorensen, D.: Lapack Users' Guide, 2nd edn. SIAM, Philadelphia (1995)
[2] Björck, Å.: Numerical Methods for Least Squares Problems. SIAM, Philadelphia, Pennsylvania (1996)
[3] Demmel, J.: Applied Numerical Linear Algebra. SIAM, Philadelphia, Pennsylvania (1997)
[4] Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. John Hopkins University Press, Baltimore (1996)
[5] Higham, N.J.: Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia, Pennsylvania (1996)
[6] Jia, Z.: Composite orthogonal projection methods for large matrix eigenproblems. Sci. China, Ser. A **42**, 577–585 (1999)
[7] Jia, Z.: A refined subspace iteration algorithm for large sparse eigenproblems. Appl. Numer. Math. **32**, 35–52 (2000)
[8] Jia, Z.: On residuals of refined projection methods for large matrix eigenproblems. Comput. Math. Appl. **41**(7/8), 813–820 (2001)
[9] Jia, Z., Stewart, G.W.: An analysis of the Rayleigh–Ritz method for approximating eigenspaces. Math. Comput. **70**, 637–647 (2001)
[10] Stewart, G.W., Sun, J.-G.: Matrix Perturbation Theory. Academic, Boston (1990)
[11] Stewart, G.W.: On orthogonalization in the inverse power method, TR-4071, Department of Computer Science, University of Maryland, College Park (1999)
[12] Stewart, G.W.: Matrix Algorithms, vol. 2: Eigensystems. SIAM, Philadelphia, Pennsylvania (2001)
[13] Trefethen, L.N., Bau III, D.: Numerical Linear Algebra. SIAM, Philadelphia, Pennsylvania (1997)
[14] van der Vorst, H.A.: Computational methods for large Eigenvalue problems. In: Ciarlet, P.G., Lions, J.L. (eds.) Handbook of Numerical Analysis, vol. VIII, pp. 3–179. North-Holland (Elsevier), Amsterdam (2002)