

A rank-revealing method and its applications

T. Y. Li* and Zhonggang Zeng†

Abstract

A new rank revealing method is proposed. For a given matrix and a threshold for near-zero singular values, the method calculates approximate singular values below the threshold one by one along with approximate null vectors by employing a globally convergent iterative scheme as well as a deflation technique. As a result, the method returns the approximate rank of the matrix and an orthonormal basis for the approximate null space. When a row or column is added or deleted, strategies for updating/downdating the approximate rank and null space are quite straightforward and efficient. The new algorithm plays a key role for some important applications, such as numerical computation of the GCD of polynomials as well as identification of non-isolated zeros of polynomial systems as shown in the article.

1 Introduction

The numerical rank determination arises in many applications that involve matrix computations such as those discussed in a series of proceedings, *SVD and Signal Processing, I, II, III* [4, 9, 14]. While the singular value decomposition (SVD) is undoubtedly the most reliable method to determine the rank numerically, there are certain drawbacks. Among them, it is generally more expensive, it may not be able to take advantage of the matrix structure, and usually it is not easy to update or downdate when rows/columns are added or deleted. Alternative methods have been suggested, such as rank-revealing QR decomposition (RRQR) [1, 2, 3] and rank-revealing two-sided orthogonal decompositions (UTV, or URV/ULV) [5, 12, 13].

In this paper, a new rank-revealing algorithm is proposed. For a given $m \times n$ matrix A , instead of calculating its decomposition that may reveal the approximate rank, our method directly calculates the approximate rank and null space of A . We briefly outline the method as follows. Without loss of generality, we assume $m \geq n$, and let $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$ be the singular values of A . Since the smallest singular value $\sigma_{min} \equiv \sigma_n$ satisfies

$$\sigma_{min} = \min_{\|\mathbf{x}\|_2=1} \|A\mathbf{x}\|_2,$$

the problem of finding σ_{min} can be converted to solving the overdetermined system

$$\begin{pmatrix} \tau \mathbf{x}^\top \\ A \end{pmatrix} \mathbf{x} = \begin{pmatrix} \tau \\ 0 \end{pmatrix} \quad \text{where } \tau > \sigma_n \quad (1)$$

*Department of Mathematics, Michigan State University, East Lansing, MI 48824, email: li@math.msu.edu. Research supported in part by NSF under Grant DMS-0104009.

†Department of Mathematics, Northeastern Illinois University, Chicago, IL 60625, email: zzeng@neiu.edu.

for its least squares solution \mathbf{x} . For this purpose, one may use the Gauss-Newton iteration [2]

$$\begin{cases} \mathbf{x}_{j+1} = \mathbf{x}_j - \begin{pmatrix} 2\tau\mathbf{x}_j^\top \\ A \end{pmatrix}^+ \begin{pmatrix} \tau\mathbf{x}_j^\top\mathbf{x}_j - \tau \\ A\mathbf{x}_j \end{pmatrix} \\ \varsigma_{j+1} = \frac{\|A\mathbf{x}_{j+1}\|_2}{\|\mathbf{x}_{j+1}\|_2}, \quad j = 0, 1, \dots \end{cases} \quad (2)$$

Here, for a matrix B , B^+ stands for its pseudo-inverse. It can be shown that (Lemma 1 in §3) the Gauss-Newton iteration in (2) is essentially the inverse iteration on the matrix $A^\top A$ without undesirable matrix multiplication. The global convergence of the iteration is therefore warranted, and $(\varsigma_j, \mathbf{x}_j)$ will converge to the singular pair (σ_n, \mathbf{v}_n) . In this article, unless otherwise mentioned, we always use “singular vector” to represent “right” singular vector. After $\sigma_n = \sigma_{\min}$ is calculated along with its associated singular vector \mathbf{v}_n , the matrix

$$A_\varrho = \begin{pmatrix} \varrho\mathbf{v}_n^\top \\ A \end{pmatrix}, \quad \varrho \in \mathbf{R} \quad (3)$$

has the same set of singular value as well as their associated singular vectors as those of A except the smallest singular value σ_n of A is replaced by $\sqrt{\varrho^2 + \sigma_n^2}$ in A_ϱ with associated singular vector \mathbf{v}_n (Corollary 2 in §4). So, if we choose $\varrho = \|A\|_F$, then the replacement $\sqrt{\varrho^2 + \sigma_n^2}$ becomes the largest singular value of A_ϱ . In the meantime, the second smallest singular value σ_{n-1} of A becomes the smallest of A_ϱ , and the iteration in (2) for finding the smallest singular pair of A can be applied to A_ϱ to calculate the singular pair $(\sigma_{n-1}, \mathbf{v}_{n-1})$ of A . This process can be continued recursively to calculate as many singular values of A as desired in ascending order $\sigma_n \leq \sigma_{n-1} \leq \dots$, along with their associated singular vectors $\mathbf{v}_n, \mathbf{v}_{n-1}, \dots$. When σ_k is larger than the prescribed threshold $\delta\|A\|_\infty$, we will admit k as the approximate rank of A and the computed $\mathbf{v}_{k+1}, \dots, \mathbf{v}_n$ as an orthonormal basis for the approximate null space of A .

Our method has been successfully implemented and applied to many applications. The algorithm generally produces more reliable rank information than RRQR and it obviously costs much less than those methods using full SVD. On the other hand, updating and downdating in our method, elaborated in §6, are quite simple and straightforward, and consequently our method enjoys a considerable advantage over UTV decomposition in this respect.

While the rank revealing has a large variety of applications, the development of our algorithm follows the needs of two important applications emerged recently. They are: a stable numerical algorithm for the computation of GCD (greatest common divisor) of univariate polynomials and the identification of non-isolated numerical solutions of polynomial systems. The details of those applications will be illustrated in §7.

2 The basic algorithm

As before, let $A \in \mathbf{R}^{m \times n}$ ($m \geq n$) with singular values $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$. We first establish the equivalence between finding the smallest singular value $\sigma_{\min} \equiv \sigma_n$ of A and solving the least squares problem of the quadratic system

$$\begin{pmatrix} \tau\mathbf{x}^\top \\ A \end{pmatrix} \mathbf{x} = \begin{pmatrix} \tau \\ 0 \end{pmatrix} \quad \text{with } \tau > \sigma_n. \quad (4)$$

Proposition 1 Let $\mathbf{u} \in \mathbf{R}^n$ be a vector satisfying

$$\left\| \begin{pmatrix} \tau \mathbf{u}^\top \\ A \end{pmatrix} \mathbf{u} - \begin{pmatrix} \tau \\ 0 \end{pmatrix} \right\|_2^2 = \min_{\mathbf{x} \in \mathbf{R}^n} \left\| \begin{pmatrix} \tau \mathbf{x}^\top \\ A \end{pmatrix} \mathbf{x} - \begin{pmatrix} \tau \\ 0 \end{pmatrix} \right\|_2^2$$

with a scaling factor $\tau > \sigma_n$. Then \mathbf{u} is in the subspace \mathbf{W} spanned by the right singular vector(s) of A associated with the smallest singular value(s).

Proof. Let $A = U\Sigma V^\top$ be the singular value decomposition of A with orthogonal U and V . Let $\mathbf{z} = V^\top \mathbf{x}$, or $\mathbf{x} = V\mathbf{z}$, where $\mathbf{x} = (x_1, \dots, x_n)^\top$ and $\mathbf{z} = (z_1, \dots, z_n)^\top$. Let

$$f(x_1, \dots, x_n) = \left\| \begin{pmatrix} \tau \mathbf{x}^\top \\ A \end{pmatrix} \mathbf{x} - \begin{pmatrix} \tau \\ 0 \end{pmatrix} \right\|_2^2 = \left\| \begin{pmatrix} \tau \mathbf{x}^\top \mathbf{x} - \tau \\ A\mathbf{x} \end{pmatrix} \right\|_2^2,$$

then

$$\begin{aligned} f(x_1, \dots, x_n) &= \tau^2 (\mathbf{x}^\top \mathbf{x} - 1)^2 + \|A\mathbf{x}\|_2^2 = \tau^2 (\mathbf{z}^\top \mathbf{z} - 1)^2 + \|\Sigma \mathbf{z}\|_2^2 \\ &= \tau^2 (z_1^2 + \dots + z_n^2 - 1)^2 + \sigma_1^2 z_1^2 + \dots + \sigma_n^2 z_n^2 \equiv g(z_1, \dots, z_n). \end{aligned}$$

Assume $g(\mathbf{z})$ reaches its minimum at $\mathbf{z} = \mathbf{y} \equiv (y_1, \dots, y_n)^\top$. Then

$$\left. \frac{\partial g}{\partial z_j} \right|_{\mathbf{z}=\mathbf{y}} = 0, \quad j = 1, \dots, n, \quad \text{i.e.,} \quad 4\tau^2 (y_1^2 + \dots + y_n^2 - 1) y_j + 2\sigma_j^2 y_j = 0.$$

If $\mathbf{y} \neq 0$, let $J = \{1 \leq j \leq n \mid y_j \neq 0\}$. Then for $j \in J$, $\sigma_j^2 = 2\tau^2 \left(1 - \sum_{l \in J} y_l^2\right)$. Hence, $\sigma_j^2 \leq 2\tau^2$, and $\sigma_j = \sigma$ for all $j \in J$ for certain $\sigma \in \{\sigma_1, \dots, \sigma_n\}$ with $\sigma < \sqrt{2}\tau$. It follows that

$$\begin{aligned} g(y_1, \dots, y_n) &= \tau^2 \left(\sum_{j \in J} y_j^2 - 1 \right)^2 + \sum_{j \in J} \sigma_j^2 y_j^2 = \tau^2 \left(\sum_{j \in J} y_j^2 - 1 \right)^2 + \sigma^2 \sum_{j \in J} y_j^2 \\ &= \tau^2 \left(\sum_{j \in J} y_j^2 - 1 \right)^2 + \sigma^2 \left(\sum_{j \in J} y_j^2 - 1 \right) + \sigma^2 \\ &= \tau^2 \left(-\frac{\sigma^2}{2\tau^2} \right)^2 + \sigma^2 \left(-\frac{\sigma^2}{2\tau^2} \right) + \sigma^2 = \sigma^2 - \frac{\sigma^4}{4\tau^2}. \end{aligned}$$

Therefore, the possible minimum values of $g(\mathbf{z})$ are $\sigma_j^2 - \frac{\sigma_j^4}{4\tau^2}$, $j = 1, \dots, n$, along with $g(0, \dots, 0) = \tau^2$. Those values are all attainable since, for every singular pair (σ_j, \mathbf{v}_j) , letting $\mathbf{z} = sV^\top \mathbf{v}_j$ with $s^2 = 1 - \frac{\sigma_j^2}{2\tau^2}$ yields

$$g(\mathbf{z}) = \tau^2 (s^2 - 1)^2 + \sigma_j^2 s^2 = \tau^2 \frac{\sigma_j^4}{4\tau^4} + \sigma_j^2 \left(1 - \frac{\sigma_j^2}{2\tau^2}\right) = \sigma_j^2 - \frac{\sigma_j^4}{4\tau^2}.$$

The function $h(\beta) = \beta^2 - \frac{\beta^4}{4\tau^2}$ is an increasing function for $\beta \in [0, \tau]$, so

$$\min_{j=1, \dots, n} \left\{ \sigma_j^2 - \frac{\sigma_j^4}{4\tau^2} \right\} = \sigma_n^2 - \frac{\sigma_n^4}{4\tau^2} \leq \sigma_n^2 < \tau^2$$

and $g(z_1, \dots, z_n)$ reaches the minimum if $\sigma = \sigma_n$. Consequently, $\sigma_j = \sigma_n$ for all $j \in J$, and $\mathbf{u} = \sum_{j \in J} y_j \mathbf{v}_j$ where \mathbf{v}_j is the singular vector associated with $\sigma_j, j = 1, \dots, n$. Q.E.D.

Based on Proposition 1, the smallest singular value of A can be calculated via solving system (4) by the Gauss-Newton iteration [2]:

$$\begin{cases} \mathbf{x}_{j+1} = \mathbf{x}_j - \begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ A \end{pmatrix}^+ \begin{pmatrix} \tau \mathbf{x}_j^\top \mathbf{x}_j - \tau \\ A \mathbf{x}_j \end{pmatrix} \\ \zeta_{j+1} = \frac{\|A \mathbf{x}_{j+1}\|_2}{\|\mathbf{x}_{j+1}\|_2}, \quad j = 0, 1, \dots \end{cases} \quad (5)$$

where for a matrix B with full column rank, $B^+ = (B^\top B)^{-1} B^\top$ is the pseudo-inverse of B . We shall prove in the next section (§3) that the scalar sequence $\zeta_j, j = 1, 2, \dots$ always converges to the smallest singular value σ_{min} . And if σ_{min} is a simple singular value, namely $\sigma_{n-1} \neq \sigma_n$, then the vector sequences $\frac{1}{\zeta_j} A \mathbf{x}_j$ and $\mathbf{x}_j, j = 1, 2, \dots$ converge to the corresponding left and right singular vectors respectively. When σ_{min} is not simple, ζ_j still converges to σ_{min} , while $\frac{1}{\zeta_j} A \mathbf{x}_j$ and \mathbf{x}_j converge into left and right singular subspaces associated with σ_{min} .

Notice that if A has more than one zero singular values, the matrix $\begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ A \end{pmatrix}$ will be rank deficient and its pseudo inverse becomes undefined. While exact rank deficiency rarely occurs in real computation, when it happens, replacing A by $A + E$ with tiny $\|E\|_2$ will ensure the existence of the pseudo-inverse. Such substitution has virtually no effect on the computing results. For details, see [6].

In the remainder of this paper, we shall frequently refer to the iteration (5) above as ‘‘applying the Gauss-Newton iteration on matrix A ’’ for solving the least squares quadratic system in (4).

3 The convergence theory

The standard theory of the Gauss-Newton iteration warrants its local convergence under some restrictions, and the convergence rate is at least linear. The following lemma shows that the Gauss-Newton iteration (5) on the overdetermined quadratic system (4) is essentially the inverse iteration on the matrix $A^\top A$, and the convergence is therefore global.

Lemma 1 *Let $A \in \mathbf{R}^{m \times n}$ be of full column rank, and let $\{\mathbf{x}_j\}$ be a vector sequence generated by the iteration (5). Then there are constants $c_j, j = 0, 1, \dots$ such that*

$$\mathbf{x}_{j+1} = c_j (A^\top A)^{-1} \mathbf{x}_j. \quad (6)$$

Proof. For simplicity, let \mathbf{x} and \mathbf{y} denote \mathbf{x}_j and \mathbf{x}_{j+1} respectively. Now,

$$\begin{aligned} \mathbf{y} &= \mathbf{x} - \begin{pmatrix} 2\tau \mathbf{x}^\top \\ A \end{pmatrix}^+ \begin{pmatrix} \tau \mathbf{x}^\top \mathbf{x} - \tau \\ A \mathbf{x} \end{pmatrix} \\ &= \mathbf{x} - \left[(2\tau \mathbf{x}, A^\top) \begin{pmatrix} 2\tau \mathbf{x}^\top \\ A \end{pmatrix} \right]^{-1} (2\tau \mathbf{x}, A^\top) \begin{pmatrix} \tau \mathbf{x}^\top \mathbf{x} - \tau \\ A \mathbf{x} \end{pmatrix} \end{aligned}$$

$$\begin{aligned}
&= \mathbf{x} - \left(4\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)^{-1} \left[\left(2\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)\mathbf{x} - 2\tau^2\mathbf{x}\right] \\
&= \mathbf{x} - \left(4\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)^{-1} \left[\left(4\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)\mathbf{x} - 2\tau^2\mathbf{x}(\mathbf{x}^\top\mathbf{x}) - 2\tau^2\mathbf{x}\right] \\
&= \left(4\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)^{-1} 2\tau^2 \left(1 + \mathbf{x}^\top\mathbf{x}\right)\mathbf{x}.
\end{aligned}$$

This yields

$$\begin{aligned}
\left(4\tau^2\mathbf{x}\mathbf{x}^\top + A^\top A\right)\mathbf{y} &= 2\tau^2 \left(1 + \mathbf{x}^\top\mathbf{x}\right)\mathbf{x} \\
\left(A^\top A\right)\mathbf{y} &= \tau^2 \left(2 + 2\mathbf{x}^\top\mathbf{x} - 4\mathbf{x}^\top\mathbf{y}\right)\mathbf{x} \\
\mathbf{y} &= 2\tau^2 \left(1 + \mathbf{x}^\top\mathbf{x} - 2\mathbf{x}^\top\mathbf{y}\right) \left(A^\top A\right)^{-1}\mathbf{x}. \tag{7}
\end{aligned}$$

So, $\mathbf{y} = c \left(A^\top A\right)^{-1}\mathbf{x}$ with $c = \frac{2\tau^2(1 + \mathbf{x}^\top\mathbf{x})}{1 + 4\tau^2\mathbf{x}^\top(A^\top A)^{-1}\mathbf{x}}$. Q.E.D.

We now consider the determination of the *approximate rank* of A . A commonly used definition is given below.

Definition 1 For a given tolerance $\theta > 0$, the *approximate rank* of matrix $A \in \mathbf{R}^{m \times n}$ is k if its singular values satisfy

$$\sigma_1 \geq \dots \geq \sigma_k = \widehat{\sigma} > \theta > \widetilde{\sigma} = \sigma_{k+1} \geq \dots \geq \sigma_n.$$

If the approximate rank of A is k , then $\mathbf{W} = \text{span}\{\mathbf{v}_{k+1}, \dots, \mathbf{v}_n\}$ is called the *approximate null space* of A , where \mathbf{v}_j is the associated singular vector of σ_j for $j = k + 1, \dots, n$.

Apparently, this approximate rank depends critically on the tolerance $\theta > 0$ one chooses. The orthogonal complement of \mathbf{W} , denoted by \mathbf{W}^\perp , is $\text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$, and every vector $\mathbf{z} \in \mathbf{R}^n$ can be written as $\mathbf{z} = \widehat{\mathbf{z}} + \widetilde{\mathbf{z}}$ with $\widehat{\mathbf{z}} \in \mathbf{W}^\perp$ and $\widetilde{\mathbf{z}} \in \mathbf{W}$. We say a sequence of non-zero vectors $\{\mathbf{z}_j\}$ converges into \mathbf{W} if

$$\lim_{j \rightarrow \infty} \frac{\|\widehat{\mathbf{z}}_j\|_2}{\|\widetilde{\mathbf{z}}_j\|_2} = 0, \quad \|\widehat{\mathbf{z}}_j\|_2 \neq 0.$$

The following proposition ensures that the vector sequence $\{\mathbf{x}_j\}$ generated by iteration (5) converges into the approximate null space of A .

Proposition 2 Suppose A has an approximate rank k with an approximate null space \mathbf{W} and \mathbf{x}_0 is not orthogonal to \mathbf{W} . Then the iteration (5) generates a vector sequence $\{\mathbf{x}_j\}$ and a scalar sequence $\{\varsigma_j\}$ where \mathbf{x}_j converges into \mathbf{W} linearly in the following sense

$$\frac{\|\widehat{\mathbf{x}}_j\|_2}{\|\widetilde{\mathbf{x}}_j\|_2} \leq \left(\frac{\widetilde{\sigma}}{\widehat{\sigma}}\right)^{2j} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\widetilde{\mathbf{x}}_0\|_2}, \quad j = 0, 1, \dots, \tag{8}$$

and ς_j satisfies

$$\sigma_n \leq \varsigma_j \leq \widetilde{\sigma} + \left(\frac{\widetilde{\sigma}}{\widehat{\sigma}}\right)^{2j} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\widetilde{\mathbf{x}}_0\|_2} \sigma_1. \tag{9}$$

Proof. Let $\mathbf{x}_0 = u_1 \mathbf{v}_1 + \cdots + u_n \mathbf{v}_n$. From Lemma 1,

$$\mathbf{x}_1 = \theta \left(\frac{u_1}{\sigma_1^2} \mathbf{v}_1 + \cdots + \frac{u_n}{\sigma_n^2} \mathbf{v}_n \right)$$

for certain $\theta \in \mathbf{R}$ and with $\alpha = \frac{\theta}{\sigma}$,

$$\mathbf{x}_1 = \alpha \left(\frac{\overset{\sim}{\sigma}^2}{\sigma_1^2} u_1 \mathbf{v}_1 + \cdots + \frac{\overset{\sim}{\sigma}^2}{\sigma_n^2} u_n \mathbf{v}_n \right) = \widehat{\mathbf{x}}_1 + \check{\mathbf{x}}_1$$

where

$$\begin{aligned} \|\widehat{\mathbf{x}}_1\|_2 &= \left\| \alpha \left(\frac{\overset{\sim}{\sigma}^2}{\sigma_1^2} u_1 \mathbf{v}_1 + \cdots + \frac{\overset{\sim}{\sigma}^2}{\sigma_k^2} u_k \mathbf{v}_k \right) \right\|_2 \leq |\alpha| \left(\frac{\overset{\sim}{\sigma}}{\sigma} \right)^2 \|\widehat{\mathbf{x}}_0\|_2 \\ \|\check{\mathbf{x}}_1\|_2 &= \left\| \alpha \left(\frac{\overset{\sim}{\sigma}^2}{\sigma_{k+1}^2} u_{k+1} \mathbf{v}_{k+1} + \cdots + \frac{\overset{\sim}{\sigma}^2}{\sigma_n^2} u_n \mathbf{v}_n \right) \right\|_2 \geq |\alpha| \|\check{\mathbf{x}}_0\|_2. \end{aligned}$$

Since $\|\check{\mathbf{x}}_0\|_2 \neq 0$, we have $\frac{\|\widehat{\mathbf{x}}_1\|_2}{\|\check{\mathbf{x}}_1\|_2} \leq \left(\frac{\overset{\sim}{\sigma}}{\sigma} \right)^2 \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\check{\mathbf{x}}_0\|_2}$. By a simple induction, the inequality (8) follows. For inequality (9),

$$\begin{aligned} \sigma_n &\leq \frac{\|A \mathbf{x}_j\|_2}{\|\mathbf{x}_j\|_2} \leq \frac{\|A \widehat{\mathbf{x}}_j\|_2}{\|\mathbf{x}_j\|_2} + \frac{\|A \check{\mathbf{x}}_j\|_2}{\|\mathbf{x}_j\|_2} \\ &\leq \left\| A \left(\frac{\widehat{\mathbf{x}}_j}{\|\check{\mathbf{x}}_j\|_2} \right) \right\|_2 + \left\| A \left(\frac{\check{\mathbf{x}}_j}{\|\check{\mathbf{x}}_j\|_2} \right) \right\|_2 \leq \sigma_1 \left[\left(\frac{\overset{\sim}{\sigma}}{\sigma} \right)^{2j} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\check{\mathbf{x}}_0\|_2} \right] + \overset{\sim}{\sigma}. \end{aligned}$$

Q.E.D.

As an important special case, if there is a significant gap in magnitude between σ_{n-1} and σ_n , then the iteration (5) converges to σ_n and its associated singular vector \mathbf{v}_n .

Corollary 1 *Let $\sigma_{n-1} > \sigma_n$ and \mathbf{x}_0 be an initial vector satisfying $\mathbf{x}_0^\top \mathbf{v}_n \neq 0$. Then in the Gauss-Newton iteration in (5) the matrix $B_j = \begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ A \end{pmatrix}$ is of full rank with well defined pseudo-inverse and the sequences $\{\varsigma_j\}$ and $\left\{ \frac{\mathbf{x}_j}{\|\mathbf{x}_j\|_2} \right\}$ converge to σ_n and \mathbf{v}_n respectively with*

$$\begin{aligned} \left\| \frac{\mathbf{x}_j}{\|\mathbf{x}_j\|_2} - \mathbf{v}_n \right\|_2 &\leq \left(\frac{\sigma_n}{\sigma_{n-1}} \right)^{2j} \left[1 + \left(\frac{\sigma_n}{\sigma_{n-1}} \right)^{2j} \right] \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\check{\mathbf{x}}_0\|_2} \\ |\varsigma_j - \sigma_n| &\leq \left(\frac{\sigma_n}{\sigma_{n-1}} \right)^{2j} \sigma_1 \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\check{\mathbf{x}}_0\|_2}, \quad j = 1, 2, \dots \end{aligned}$$

Proof. Since $\sigma_{n-1} > \sigma_n \geq 0$, $A \mathbf{v}_j \neq 0$ for $j = 1, \dots, n-1$. So, B_0 is of full rank because of the assumption $\mathbf{x}_0^\top \mathbf{v}_n \neq 0$. Similarly B_j is of full rank for all $j > 0$ since $\mathbf{x}_j^\top \mathbf{v}_n \neq 0$ from (8). The proof of the remaining assertions is a straightforward verification. Q.E.D.

Proof. For simplicity, let $\sigma = \sigma_n$ and SVD of A be $A = U\Sigma V^\top$. We have

$$\begin{pmatrix} 1 & & & \\ & U^\top & & \end{pmatrix} \begin{pmatrix} \rho \mathbf{v}^\top \\ A \end{pmatrix} V = \begin{pmatrix} \rho \mathbf{v}^\top V \\ U^\top AV \end{pmatrix} = \begin{pmatrix} 0 & \cdots & 0 & \rho \\ \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_{n-1} & \\ & & & \sigma \end{pmatrix}.$$

By applying Givens transformation from the left on ρ and σ , it is clear that the singular value σ of A is replaced by the singular value $\sqrt{\rho^2 + \sigma^2}$ of A_ρ while the associated singular vectors remain the same. Q.E.D.

5 The overall algorithm

As given in Definition 1, the approximate rank k of matrix A depends critically on the chosen tolerance $\theta > 0$ for which singular values of A satisfy

$$\sigma_1 \geq \cdots \geq \sigma_k > \theta > \sigma_{k+1} \geq \cdots \geq \sigma_n. \quad (13)$$

Unfortunately, no uniform threshold for such tolerances exists. Since the sizes of singular values of the target matrix can be manipulated by scaling the entries of the matrix in many occasions, one usually chooses the tolerance relative to the size of A , and the infinite norm $\|A\|_\infty$ is often used for this measure for its easy evaluation. That is, the user must decide a small number $\delta > 0$ based on the nature of the applications and uses $\theta = \delta \|A\|_\infty$ as the tolerance.

The reciprocal of the gap $\gamma = \frac{\sigma_k}{\sigma_{k+1}}$ can be considered a condition number for this rank revealing problem, meaning: the larger the gap is the more accurate the approximate rank will be. If γ is large, say 10^3 , then every iterative step in (5) will improve the convergence by 6 digits because in Proposition 2 the sequences $\{\mathbf{x}_j\}$ and $\{\varsigma_j\}$ satisfy

$$\frac{\|\widehat{\mathbf{x}}_j\|_2}{\|\underline{\mathbf{x}}_j\|_2} \leq (10^{-3})^{2j} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\underline{\mathbf{x}}_0\|_2} \quad \text{and} \quad \sigma_n \leq \varsigma_j \leq \sigma_{k+1} + (10^{-3})^{2j} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\underline{\mathbf{x}}_0\|_2} \sigma_1.$$

After three iteration steps they become

$$\frac{\|\widehat{\mathbf{x}}_3\|_2}{\|\underline{\mathbf{x}}_3\|_2} \leq 10^{-18} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\underline{\mathbf{x}}_0\|_2} \quad \text{and} \quad \sigma_n \leq \varsigma_3 \leq \sigma_{k+1} + 10^{-18} \frac{\|\widehat{\mathbf{x}}_0\|_2}{\|\underline{\mathbf{x}}_0\|_2} \sigma_1.$$

Since the machine epsilon of IEEE standard double precision is about 2.2×10^{-16} , therefore \mathbf{x}_3 in this case is sufficiently accurate to be an approximate null vector unless the randomly chosen initial vector \mathbf{x}_0 is almost orthogonal to the approximate null space.

Let $(\sigma_1, \mathbf{v}_1), \dots, (\sigma_n, \mathbf{v}_n)$ be the singular pairs of A with σ_j 's satisfying (13). For an input tolerance $\delta > 0$, our algorithm begins with calculating the approximate smallest singular pair $(\hat{\sigma}_n, \hat{\mathbf{v}}_n)$. If $\hat{\sigma}_n > \delta \|A\|_\infty$, then A will be classified as being full rank, and the process stops.

```

Pseudo-code RANKREV:
input: Matrix  $A \in \mathbf{R}^{m \times n}$ , threshold  $\delta > 0$ 
output: rank  $k$ , orthonormal basis  $\{\mathbf{w}_{k+1}, \dots, \mathbf{w}_n\}$  of the null space.
QR decomposition  $A = QR$ 
Initialize  $B = R$ ,  $\tau = \|A\|_\infty$ 
For  $k = n, n-1, \dots, 1$  do
    generate a random unit vector  $\mathbf{x}_0$ 
    for  $j = 0, 1, 2$  do
         $D = \begin{bmatrix} 2\tau \mathbf{x}_j^\top \\ B \end{bmatrix}$ ,  $\mathbf{b} = \begin{bmatrix} \tau \mathbf{x}_j^\top \mathbf{x}_j - \tau \\ B \mathbf{x}_j \end{bmatrix}$ 
        Hessenberg QR decomposition  $D = QR$ 
        backward substitution to solve  $R\mathbf{z} = Q^\top \mathbf{b}$  for  $\mathbf{z}$ 
         $\mathbf{x}_{j+1} = \mathbf{x}_j - \mathbf{z}$ ,  $\varsigma = \left\| R \mathbf{x}_{j+1} \right\|_2 \left\| \mathbf{x}_{j+1} \right\|_2^{-1}$ 
        if  $\varsigma < \delta \tau$  then
             $\mathbf{w}_k = \mathbf{x}_{j+1} / \left\| \mathbf{x}_{j+1} \right\|_2$ 
            break  $j$ -loop
        end if
    end do
    if  $\varsigma > \delta \tau$  then
        break  $k$ -loop
    else
        Hessenberg QR decomposition  $[\tau \mathbf{w}_k^\top; B] = QR$ 
        update  $B = R$ 
    end if
end do

```

Figure 1: Pseudo-code of RankRev

Otherwise the algorithm continues by calculating approximate singular pairs $(\hat{\sigma}_{n-1}, \hat{\mathbf{v}}_{n-1})$, $(\hat{\sigma}_{n-2}, \hat{\mathbf{v}}_{n-2})$, \dots . Once $\hat{\sigma}_k > \delta \|A\|_\infty$ is encountered, the process will be terminated with k being taken as the approximate rank of A and $\text{span}\{\hat{\mathbf{v}}_{k+1}, \dots, \hat{\mathbf{v}}_n\}$ becomes the approximate null space. If the gap $\gamma = \frac{\hat{\sigma}_k}{\hat{\sigma}_{k+1}}$ is not as large, usually it needs more than three iteration steps in (5) for each singular value. The users need to make decision on the number of iteration steps based on the nature of the application. The overall algorithm RANKREV is shown in a pseudo-code in Fig. 1.

Example: We first generate a 20×12 random matrix B and compute its singular value decomposition $B = U\Sigma V^\top$. Then by resetting the diagonal matrix Σ with the singular values replaced by

1, 1e-1, 1e-2, 1e-3, 1e-7, 1e-8, 1e-11, 5e-12, 1e-13, 1e-14, 0.99e-14, 0.97e-14,

the test matrix A is set to be $B = U\Sigma V^\top$ with this new Σ . For threshold $\delta = 1.0e - 10$, Algorithm RANKREV provides 6 approximate singular values. The comparison of the exact singular values and their approximation is shown in Table 1.

	exact singular value	approximate singular value
σ_7	1.00e-11	0.85234e-11
σ_8	5.00e-12	7.23564e-12
σ_9	1.00e-13	0.99995e-13
σ_{10}	1.00e-14	1.0007e-14
σ_{11}	0.99e-14	0.98321e-14
σ_{12}	0.97e-14	0.98630e-14

Table 1: Exact and approximate singular values

In rank revealing, the main interest lies in the magnitude rather than the correct digits of the approximate singular value, and our algorithm achieves this goal in this example. On the other hand, the residual $\|A(\hat{\mathbf{v}}_7, \dots, \hat{\mathbf{v}}_{12})\|_2 \approx 10^{-11}$ and the distance between the exact null space \mathbf{W} and approximate null space $\widehat{\mathbf{W}}$ is 3×10^{-7} . Apparently, the approximate null space $\widehat{\mathbf{W}}$ is quite accurate while the gap σ_6/σ_7 is just 10^3 . This distance can be even smaller if the gap is larger.

Practically, the iteration (5) is carried out by finding a least squares solution $\Delta \mathbf{x}$ ($= \mathbf{x}_{j+1} - \mathbf{x}_j$) to the linear system

$$\begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ A \end{pmatrix} \Delta \mathbf{x} = - \begin{pmatrix} \tau(\mathbf{x}_j^\top \mathbf{x}_j - 1) \\ A \mathbf{x}_j \end{pmatrix} \quad (14)$$

at the j -th stage. To avoid unnecessary QR decomposition of the full matrix at each step, we may calculate the QR decomposition of A *before* the iteration and update the QR decomposition at each step.

With QR factorization $A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$, finding the least squares solution to (14) is equivalent to solving the least squares problem of

$$\begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ R \end{pmatrix} \Delta \mathbf{x} = - \begin{pmatrix} \tau \mathbf{x}_j^\top \mathbf{x}_j - \tau \\ R \mathbf{x}_j \end{pmatrix} \quad (15)$$

for $\Delta \mathbf{x} = \mathbf{x}_{j+1} - \mathbf{x}_j$, in which one uses the QR decomposition of the upper Hessenberg matrix $\begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ R \end{pmatrix}$. Updating the QR factorization of an n -column upper-Hessenberg matrix requires n Givens transformations which cost $O(n^2)$ flops in total. After QR update, solving (15) for its least squares solution requires a total of $O(n^2)$ flops in backward substitutions.

The final QR factorization of $\begin{pmatrix} 2\tau \mathbf{x}_j^\top \\ R \end{pmatrix}$ can also be recycled as the QR decomposition of B in (10) with $\rho = 2\tau$ and $\mathbf{w} = \mathbf{x}_j$. The computations are all in the order of $O(n^2)$ except the first QR factorization of A which costs $O(mn^2)$. Frequently, the QR decomposition of A had already been calculated for other purposes.

6 Updating and downdating

After the approximate rank k and the orthonormal basis $\{\mathbf{v}_{k+1}, \dots, \mathbf{v}_n\}$ of the approximate null space \mathbf{W} of A are computed, if a column or a row is augmented to A , re-calculating the approximate rank and null space is called updating. If a column or a row is deleted from A , the additional rank determination is called downdating. Updating and downdating subject to adding/deleting a row or a column can be efficiently carried out in our rank revealing method. We shall address those four cases separately below.

It will become apparent in our discussion that efficient updating/downdating in our method is actually the problem of efficient updating/downdating the QR decomposition of certain matrices. When A is of full rank, in this case $k = n$, the problem becomes updating/downdating the QR decomposition of A itself which were discussed in detail in [6]. We therefore assume $k < n$ in all the cases we shall discuss.

6.1 Column updating

Without loss of generality, we assume a vector \mathbf{a}_{n+1} is attached to $A = (\mathbf{a}_1, \dots, \mathbf{a}_n)$ in the right and let $A' = (\mathbf{a}_1, \dots, \mathbf{a}_n, \mathbf{a}_{n+1})$. Clearly the null space \mathbf{W}' of A' contains $\{\mathbf{v}'_{k+1}, \dots, \mathbf{v}'_n\}$ where

$$\mathbf{v}'_j = \begin{pmatrix} \mathbf{v}_j \\ 0 \end{pmatrix}, \quad j = k+1, \dots, n.$$

Those \mathbf{v}'_j 's remain orthonormal. The approximate rank of A' is either k or $k+1$. Only when it stays at k , there is an additional vector in the orthonormal basis of \mathbf{W}' . To find this possible additional vector, the Gauss-Newton iteration (5) may be applied on the matrix

$$\hat{A} = \begin{pmatrix} \tau \mathbf{v}_{k+1}^\top \\ \vdots \\ \tau \mathbf{v}_n^\top \\ A' \end{pmatrix} = \begin{pmatrix} \tau \mathbf{v}_{k+1}^\top & 0 \\ \vdots & \vdots \\ \tau \mathbf{v}_n^\top & 0 \\ A & \mathbf{a}_{n+1} \end{pmatrix}$$

via its QR decomposition. In the end of algorithm RANKREV for the rank decision of A , the QR decomposition

$$\begin{pmatrix} \tau \mathbf{v}_{k+1}^\top \\ \vdots \\ \tau \mathbf{v}_n^\top \\ A \end{pmatrix} = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

is available. Let

$$Q^\top \hat{A} = Q^\top \begin{pmatrix} \tau \mathbf{v}_{k+1}^\top & 0 \\ \vdots & \\ \tau \mathbf{v}_n^\top & 0 \\ A & \mathbf{a}_{n+1} \end{pmatrix} = \begin{pmatrix} R & \mathbf{d}_1 \\ 0 & \mathbf{d}_2 \end{pmatrix},$$

and H be the Householder transformation matrix satisfying

$$H \mathbf{d}_2 = \begin{pmatrix} \gamma \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Then

$$\hat{A} = \begin{pmatrix} \tau \mathbf{v}'_{k+1} \\ \vdots \\ \tau \mathbf{v}'_n \\ A' \end{pmatrix} = \begin{pmatrix} \tau \mathbf{v}'_{k+1} & 0 \\ \vdots & \\ \tau \mathbf{v}'_n & 0 \\ A & \mathbf{a}_{n+1} \end{pmatrix} = Q \begin{pmatrix} I_{n \times n} & 0 \\ 0 & H^\top \end{pmatrix} \begin{pmatrix} R & \mathbf{d}_1 \\ \gamma & 0 \\ 0 & \vdots \\ 0 & 0 \end{pmatrix} = Q' \begin{pmatrix} R' \\ 0 \end{pmatrix}.$$

We may now apply the Gauss-Newton iteration (5) on \hat{A} via R' to compute the additional basis vector in the null space of A' , or declare its absence. The additional flops are in the order of $O(n^2)$.

6.2 Column downdating

Let $\tilde{A} = (\mathbf{a}_1, \dots, \mathbf{a}_{l-1}, \mathbf{a}_{l+1}, \dots, \mathbf{a}_n)$ be downdated from A , and $\tilde{\mathbf{W}}$ denote the null space of \tilde{A} . If the dimension $n - k$ of the null space $\mathbf{W} = \text{span}\{\mathbf{v}_{k+1}, \dots, \mathbf{v}_n\}$ is one, then the dimension of $\tilde{\mathbf{W}}$ is either 0 or 1. One can achieve the downdating of $\tilde{\mathbf{W}}$ by simply applying the Gauss-Newton iteration (5) on \tilde{A} via its QR decomposition — a downdating of the QR decomposition of A (see [6]).

If $n - k > 1$, write

$$V = [\mathbf{v}_{k+1}, \dots, \mathbf{v}_n] = \begin{pmatrix} v_{1,k+1} & \cdots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n,k+1} & \cdots & v_{nn} \end{pmatrix} \in \mathbf{R}^{n \times (n-k-1)}.$$

Let H be the Householder transformation matrix satisfying

$$H \begin{pmatrix} v_{l,k+1} \\ v_{l,k+2} \\ \vdots \\ v_{l,n} \end{pmatrix} = \begin{pmatrix} * \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

This makes

$$VH^\top = \begin{pmatrix} * & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & * \\ * & 0 & \cdots & 0 \\ * & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & * \end{pmatrix} \longleftarrow l\text{-th row.}$$

The columns of VH^\top are orthonormal because

$$(VH^\top)^\top (VH^\top) = H(V^\top V)H^\top = H^\top I_{n-k} H = I_{n-k},$$

and since $\|A(VH^\top)\|_2 \leq \|AV\|_2 \|H^\top\|_2 = \|AV\|_2$, those columns also form a basis of W . Delete the first column of VH^\top and let $[\hat{v}_{k+2}, \dots, \hat{v}_n]$ be the remaining matrix. For $j = k + 2, \dots, n$, removing the zero component of \hat{v}_j at the l -th entry, the resulting vector \tilde{v}_j yields

$$A\tilde{v}_j = \tilde{A}\tilde{v}_j, \quad j = k + 2, \dots, n,$$

and $\{\tilde{v}_{k+2}, \dots, \tilde{v}_n\}$ forms an orthonormal basis for a subspace of $\tilde{\mathbf{W}}$. An additional null vector in the orthonormal basis of $\tilde{\mathbf{W}}$ exists when the dimension of $\tilde{\mathbf{W}}$ is $n - k$. To verify this possibility by applying the Gauss-Newton iteration (5) on

$$\hat{A} = \begin{pmatrix} \tau \tilde{\mathbf{v}}_{k+2}^\top \\ \vdots \\ \tau \tilde{\mathbf{v}}_n^\top \\ \tilde{A} \end{pmatrix}$$

as before requires the QR decomposition of \hat{A} . In the first place, the QR decomposition of \tilde{A} , $\tilde{A} = \tilde{Q}\tilde{R}$, is a downdating of the QR decomposition of A which requires no more than $O(n^2)$ flops [6].

Since

$$\hat{A} = \begin{pmatrix} \tau \tilde{\mathbf{v}}_{k+2}^\top \\ \vdots \\ \tau \tilde{\mathbf{v}}_n^\top \\ \tilde{A} \end{pmatrix} = \begin{pmatrix} \tau \tilde{\mathbf{v}}_{k+2}^\top \\ \vdots \\ \tau \tilde{\mathbf{v}}_n^\top \\ \tilde{Q}\tilde{R} \end{pmatrix} = \begin{pmatrix} I_{n-k-1} & 0 \\ 0 & \tilde{Q} \end{pmatrix} \begin{pmatrix} \tau \tilde{\mathbf{v}}_{k+2}^\top \\ \vdots \\ \tau \tilde{\mathbf{v}}_n^\top \\ \tilde{R} \end{pmatrix},$$

the QR decomposition of \hat{A} may be obtained by taking the QR decomposition of

$$\begin{pmatrix} \tau \tilde{\mathbf{v}}_{k+2}^\top \\ \vdots \\ \tau \tilde{\mathbf{v}}_n^\top \\ \tilde{R} \end{pmatrix},$$

which requires $O((n - k)n^2)$ flops. On the other hand, iteration (5) requires $O(n^2)$ flops.

6.3 Row updating

Let $A' = \begin{pmatrix} \mathbf{x}^\top \\ A \end{pmatrix}$ and \mathbf{W}' be the approximate null space of A' . Clearly $\mathbf{W}' \subset \mathbf{W}$, and if \mathbf{x} is approximately orthogonal to \mathbf{W} , then $\mathbf{W}' = \mathbf{W}$. No additional calculation is required in this situation.

If $n - k = 1$, then, as in column downdating, the rank updating becomes an updating on the QR decomposition of A (see [6]), and we may employ the Gauss-Newton iteration (5) to calculate the approximate null space of A' and clarify its dimension being 0 or 1.

When $n - k > 1$, let $W = (\mathbf{v}_{k+1}, \dots, \mathbf{v}_n) \in \mathbf{R}^{n \times (n-k)}$. When $\|W^\top \mathbf{x}\|_2$ is not small, then the approximate null space \mathbf{W}' of A' is of dimension $n - k - 1$. The following process calculates an orthonormal basis of \mathbf{W}' .

1. Calculate $\mathbf{y} = W^\top \mathbf{x} \in \mathbf{R}^{n-k}$.
2. Expand the vector \mathbf{y} to an $(n - k) \times (n - k)$ orthogonal matrix $(\mathbf{y}, \mathbf{y}_{k+2}, \dots, \mathbf{y}_n)$ by first constructing a Householder transformation matrix $H = (\mathbf{h}_{k+1}, \dots, \mathbf{h}_n)$ which makes $H\mathbf{y} = (1, 0, \dots, 0)^\top$, then letting $\mathbf{y}_j = \mathbf{h}_j$, $j = k + 2, \dots, n$.

3. Set $E = (\mathbf{y}_{k+2}, \dots, \mathbf{y}_n) \in \mathbf{R}^{(n-k) \times (n-k-1)}$, and let $WE = (\mathbf{v}'_{k+2}, \dots, \mathbf{v}'_n) \in \mathbf{R}^{n \times (n-k-1)}$.

The vectors $\mathbf{v}'_{k+2}, \dots, \mathbf{v}'_n$ are orthonormal, because

$$(WE)^\top (WE) = E^\top (W^\top W) E = I_{(n-k-1) \times (n-k-1)}.$$

Moreover

$$\begin{aligned} \mathbf{x}^\top WE &= \mathbf{y}^\top E = 0 \\ \text{and, } \|A'WE\|_2 &= \|AWE\|_2 \leq \|AW\|_2 \|E\|_2 = \|AW\|_2 \leq \tilde{\sigma}. \end{aligned}$$

Therefore, those orthonormal vectors $\mathbf{v}'_{k+2}, \dots, \mathbf{v}'_n$ form an orthogonal basis of \mathbf{W}' .

6.4 Row downdating

Here, we consider deleting the top row only. The procedure is similar when an arbitrary row is deleted.

If downdating of the top row is needed, we will adjust our rank revealing method. After finding a null vector \mathbf{w} , instead of stacking $\varrho \mathbf{w}^\top$ on top of A , we attach $\varrho \mathbf{w}^\top$ at the bottom so that the top row of A to be deleted still stays on top. All the theoretical results of our method remain valid. From the QR decomposition of A , the QR decomposition of $\begin{pmatrix} A \\ \varrho \mathbf{w}^\top \end{pmatrix}$ becomes $\begin{pmatrix} Q & \\ & 1 \end{pmatrix} \begin{pmatrix} R \\ \varrho \mathbf{w}^\top \end{pmatrix}$. Now $\begin{pmatrix} R \\ \varrho \mathbf{w}^\top \end{pmatrix}$ is no longer an upper Hessenberg matrix. However, its QR decomposition only requires n Givens rotations which cost essentially the same as converting a Hessenberg matrix.

With this adjustment, the procedure RANKREV produces an approximate rank k of A , an orthonormal basis $\{\mathbf{w}_{k+1}, \dots, \mathbf{w}_n\}$ for the approximate null space \mathbf{W} of A , and a QR decomposition

$$\begin{pmatrix} A \\ \varrho \mathbf{w}_n^\top \\ \vdots \\ \varrho \mathbf{w}_{k+1}^\top \end{pmatrix} = Q_0 \begin{pmatrix} R_0 \\ 0 \end{pmatrix}. \quad (16)$$

When the top row of A is deleted, all those vectors $\mathbf{w}_{k+1}, \dots, \mathbf{w}_n$ remain as null vectors of the resulting matrix A_1 with the possibility of an additional null vector in the orthogonal basis of its null space W_1 .

It takes $n - 1$ Givens transformations to convert (16) to the following form [6, §12.5.3]

$$\begin{pmatrix} A \\ \varrho \mathbf{w}_n^\top \\ \vdots \\ \varrho \mathbf{w}_{k+1}^\top \end{pmatrix} = \begin{pmatrix} 1 & \\ & Q_1 \end{pmatrix} \begin{bmatrix} \mathbf{r}^\top \\ \left(\begin{matrix} R_1 \\ 0 \end{matrix} \right) \end{bmatrix} = \begin{bmatrix} \mathbf{r}^\top \\ Q_1 \left(\begin{matrix} R_1 \\ 0 \end{matrix} \right) \end{bmatrix} \quad (17)$$

where R_1 is upper triangular, Q_1 is orthogonal and \mathbf{r}^\top is the top row of A . This transformation requires $O(n^2)$ flops.

- $S_m(p, q)$ is of rank $n + m - 2l + 1$ if and only if $\deg(\text{GCD}(p, q)) = l \geq 1$.
- $S_r(p, q)$ has full rank if and only if $\deg(\text{GCD}(p, q)) \leq m - r$.
- If $S_{r-1}(p, q)$ is of full rank and $S_r(p, q)$ is rank deficient, then the null space of $S_r(p, q)$ is spanned by the vector $\begin{pmatrix} \mathbf{w} \\ -\mathbf{v} \end{pmatrix}$ whose components $\mathbf{v} \in \mathbf{R}^{n-m+r}$ and $\mathbf{w} \in \mathbf{R}^r$ are coefficient vectors of $v(x)$ and $w(x)$ in (18) respectively. Also, $\deg(\text{GCD}(p, q)) = m - r + 1$.

From those results, one may calculate $\text{GCD}(p, q)$ by finding the first rank deficient Sylvester matrix $S_r(p, q)$ in the sequence

$$S_1(p, q), S_2(p, q), \dots, S_m(p, q)$$

and a (single vector) basis of the null space. A GCD-finder constructed in this way can be illustrated in the following process.

First, we form $S_1(p, q)$, set the first permutation $P_1 = I_{(n-m+2) \times (n-m+2)}$, and calculate its QR decomposition

$$T_1 = S_1(p, q)P_1 = \left[\begin{array}{c|c} \text{---} & \text{---} \\ \hline \end{array} \right] = \left[\begin{array}{c} \text{---} \\ \hline \end{array} \right] \left[\begin{array}{c} \text{---} \\ \hline \end{array} \right]$$

Q_1 R_1

If $S_1(p, q)$ rank deficient, then $\text{GCD}(p, q) = q$. The process needs to continue only if $S_1(p, q)$ is of full rank. In general, if $S_j(p, q)$ is of full rank with its pivoted QR decomposition $T_j = S_j(p, q)P_j = Q_j R_j$ being available, we attach one zero row to the bottom of T_j and add two columns

$$\left[\begin{array}{c|c} Q_j^\top & 1 \\ \hline \end{array} \right] \begin{bmatrix} 0 \\ \vdots \\ 0 \\ q_0 \\ \vdots \\ q_m \end{bmatrix} \quad \text{and} \quad \left[\begin{array}{c|c} Q_j^\top & 1 \\ \hline \end{array} \right] \begin{bmatrix} 0 \\ \vdots \\ 0 \\ p_0 \\ \vdots \\ p_n \end{bmatrix}$$

to the right of the resulting matrix to form T_{j+1} . With a proper permutation matrix P_{j+1} , we have $T_{j+1}P_{j+1}^\top = S_{j+1}(p, q)$. Therefore

$$T_j = S_j(p, q)P_j = Q_j \left[\begin{array}{c} \text{---} \\ \hline \end{array} \right] \longrightarrow \left[\begin{array}{c|c} Q_j & \mathbf{1} \\ \hline \end{array} \right] \left[\begin{array}{c} \text{---} \\ \hline \end{array} \right]$$

R_j R_j

Pseudo-code GCD:

```

input: coefficient vectors for  $p(x)$ ,  $q(x)$ 
output:  $d = \deg(\text{GCD}(p, q))$ , coefficients of  $v(x)$  and  $w(x)$  in (18)
  QR decomposition  $QR = S_1(p, q)$ 
  For  $j = 1, 2, \dots, m$  do
    Gauss-Newton iteration (5) on  $R$ , get  $\varrho$  and  $\mathbf{x}$ 
    if  $\varrho$  is small enough, then
      extract coefficients of  $v(x)$  and  $w(x)$  from  $\mathbf{x}$ 
      solve (19) for the coefficients of  $u(x)$ 
      exit
    else
      if  $j \leq m$  then
        update  $Q_{j+1}R_{j+1} = S_{j+1}(p, q)P_{j+1}$ 
      else
         $\deg(\text{GCD}(p, q)) = 0$ ,  $v(x) = p(x)$ ,  $w(x) = q(x)$ 
      end if
    end if
  end do

```

Figure 2: Pseudo-code of GCD

and the result shows it admits very small singular values, \mathbf{x}_0 may lie on a solution component of $P(\mathbf{x}) = 0$ with positive dimension or it may still be an isolated zero with multiplicity ≥ 2 . Our strategy to distinguish those cases is given below.

If $P_x(\mathbf{x}_0)$ permits only one singular value that appears tiny and if \mathbf{x}_0 is not an isolated solution, then \mathbf{x}_0 must lie on a one (complex) dimensional solution component M of $P(\mathbf{x}) = 0$. We will begin to identify this path to a substantial length by a path following scheme developed in [7]. If this attempt fails, no such solution component M may exist and \mathbf{x}_0 will be classified as an isolated solution of $P(\mathbf{x}) = 0$.

When $P_x(\mathbf{x}_0)$ has $k > 1$ very small singular values as a result of our rank revealing algorithm, we augment $P(\mathbf{x}) = 0$ with $k - 1$ generic hyperplanes

$$\mathbf{a}_j^H(\mathbf{x} - \mathbf{x}_0) = 0, \quad j = 1, \dots, k - 1$$

at \mathbf{x}_0 . The enlarged system

$$\widehat{P}(\mathbf{x}) = \begin{cases} P(\mathbf{x}) = 0 \\ \mathbf{a}_1^H(\mathbf{x} - \mathbf{x}_0) = 0 \\ \vdots \\ \mathbf{a}_{k-1}^H(\mathbf{x} - \mathbf{x}_0) = 0 \end{cases} \quad (20)$$

will produce a one dimensional component \widehat{M} of $\widehat{P}(\mathbf{x}) = 0$ if the solution component M of $P(\mathbf{x}) = 0$ to which \mathbf{x}_0 belongs is of dimension k . Thus, the assertion that $\dim(M) = k$ is valid only if we can identify \widehat{M} by following this path to a satisfactory length. If the path following can not be carried out successfully, such component \widehat{M} may not exist. We will then remove

hyperplane $\mathbf{a}_{k-1}^H(\mathbf{x} - \mathbf{x}_0) = 0$ in (20) and restart our effort to identify the one dimensional component $\widehat{\widehat{M}}$ produced by

$$\widehat{\widehat{P}}(\mathbf{x}) = \begin{cases} P(\mathbf{x}) = 0 \\ \mathbf{a}_1^H(\mathbf{x} - \mathbf{x}_0) = 0 \\ \vdots \\ \mathbf{a}_{k-2}^H(\mathbf{x} - \mathbf{x}_0) = 0. \end{cases} \quad (21)$$

The existence of such component $\widehat{\widehat{M}}$ of $\widehat{\widehat{P}} = 0$ implies the solution component M of $P(\mathbf{x}) = 0$ containing \mathbf{x}_0 is of dimension $k - 1$. If it fails, the process may be continued in the same manner and the dimension of M will ultimately (very soon in practice) be determined. Of course, when $\dim(M) = 0$, \mathbf{x}_0 is an isolated zero even though $P_x(\mathbf{x}_0)$ may have very small singular values from our rank revealing algorithm.

Example [11]: Consider the polynomial system $P(\mathbf{x}) = (p_1(\mathbf{x}), p_2(\mathbf{x}), p_3(\mathbf{x}))$, $\mathbf{x} = (u, v, w) \in \mathbf{C}^3$ where

$$\begin{aligned} p_1(\mathbf{x}) &= (v - u^2) \cdot (u^2 + v^2 + w^2 - 1)(u - 0.5) \\ p_2(\mathbf{x}) &= (w - u^3)(u^2 + v^2 + w^2 - 1)(v - 0.5) \\ p_3(\mathbf{x}) &= (v - u^2)(w - u^3)(u^2 + v^2 + w^2 - 1)(w - 0.5). \end{aligned}$$

Obviously, the solution set of $P(\mathbf{x}) = 0$ consists of

1. A two-dimensional component $u^2 + v^2 + w^2 = 1$;
2. Four one-dimensional components
 - (a) line $u = 0.5, v = (0.5)^3$;
 - (b) line $u = \sqrt{0.5}, v = 0.5$;
 - (c) line $u = -\sqrt{0.5}, v = 0.5$;
 - (d) twisted cubic $v = u^2, w = u^3$;
3. One isolated solution $(u, v, w) = (0.5, 0.5, 0.5)$.

When the polyhedral homotopy continuation method [8] was used to solve $P(\mathbf{x}) = 0$, 129 numerical solutions were obtained. We applied our method to all those solutions, and the result shows

- 112 of them lie on the 2-dimensional component,
- 16 of them lie on 1-dimensional components (four on line 2a, four on line 2b, four on line 2c, four on line 2d),
- one isolated solution.

When we classified a solution \mathbf{x}_0 that is lying on a 2-dimensional component of $P(\mathbf{x}) = 0$ for instance, we substituted \mathbf{x}_0 into $u^2 + v^2 + w^2 = 1$ to verify the accuracy of our identification. And the results were all accurate.

References

- [1] C. H. Bischof and G. Quintana-Orti, *Codes for rank-revealing QR factorizations of dense matrices*, Argonne Preprint ANL-MCS-P560-0196, 1996.
- [2] A. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
- [3] T.R. Chan, *Rank revealing QR factorizations*, Lin. Alg. Appl., 88/89, pp67-82, 1987.
- [4] F. Deprettere, *SVD and Signal Processing, Algorithms, Applications, and Architectures*, North-Holland, Amsterdam, 1988.
- [5] R. D. Fierro, P. C. Hansen and P. S. K. Hansen, *UTV tools: Matlab templates for rank-revealing UTV decompositions*, Numer. Algorithms, 20, pp 165-194, 1999.
- [6] G. H. Golub and C.F. Van Loan, *Matrix Computations*, 3rd Ed., Johns Hopkins University Press, 1996.
- [7] Y. C. Kuo and T. Y. Li, *Determining whether a zero of a polynomial system is isolated*, A preprint, 2003.
- [8] T. Y. Li, *Numerical solution of multivariate polynomial systems by homotopy continuation methods*, ACTA Numerica (1997), pp 399-436.
- [9] M. Moonen and B. De Moor, *SVD and Signal Processing, III, Algorithms, Applications, and Architectures*, Elsevier, Amsterdam, 1995.
- [10] D. Rupprecht, *An algorithm for computing certified approximate GCD of n univariate polynomials*, J. Pure and Appl. Alg., 139, pp 255–284, 1999.
- [11] A. J. Sommese, J. Verschelde and C. W. Wampler, *Numerical decomposition of the solution sets of polynomial systems into irreducible components*, SIAM J. Numer. Anal., 38(6), pp 2022–2046, 2001
- [12] G.W. Stewart, *UTV decompositions*, in D.F. Griffith and G.A. Watson (Eds), *Numerical Analysis, 1993*, Pitman Research Notes in Mathematical Sciences, New York, 1994.
- [13] G.W. Stewart, *Matrix Algorithms, Volume I: Basic Decompositions*, SIAM, Philadelphia, 1998.
- [14] R. Vaccaro, *SVD and Signal Processing, II, Algorithms, Applications, and Architectures*, Elsevier, Amsterdam, 1991.
- [15] Z. Zeng, *Computing multiple roots of inexact polynomials*, to appear (2003).
- [16] Z. Zeng, *MultRoot - A Matlab package computing polynomial roots and multiplicities*, to appear (2003).