U.C.L.A.
COMPUTATIONAL AND APPLIED MATHEMATICS

Computing Truncated SVD Least Squares Solutions by Rank Revealing QR-Factorizations

Tony F. Chan
Per Christian Hansen

November, 1986

CAM Report 86-01

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024

Computing Truncated SVD Least Squares Solutions by Rank Revealing QR-Factorizations

Tony F. Chan, Dept. of Mathematics, U.C.L.A.,
Los Angeles, California 90024, U.S.A.

Per Christian Hansen, Dept. of Computer Science, Stanford University,
Stanford, California 94305-2095, U.S.A.†

November, 1986

Abstract

Solutions to rank deficient least squares problems are conveniently expressed in terms of the singular value decomposition (SVD) of the coefficient matrix. When the matrix is *nearly* rank deficient, a common procedure is to neglect its smallest singular values, which leads to the truncated SVD (TSVD) solution. In this paper, we present an efficient method for computing the TSVD solution via a QR-factorization, without the need for computing a complete SVD. The numerical rank of the matrix is determined by means of a rank revealing QR-factorization, which provides upper and lower bounds on the small singular values and approximations to the corresponding singular vectors, which are then refined by inverse subspace iteration and used in conjunction with the QR factors to compute the TSVD solution.

Key words: least squares problems, truncated singular value decomposition, nearly rank deficiency, numerical rank, rank revealing QR-factorization, inverse subspace iteration, subset selection.

1. Introduction

In this paper, we consider an efficient and reliable numerical method for solving the linear least squares problem:

$$\min \|b - A\,x\|_2 \quad , \quad A \in R^{m\times n} \quad , \quad m \geq n \tag{1.1}$$

with the matrix A ill-conditioned and possibly rank deficient. This method is the *truncated SVD (TSVD) method*, which is based on the singular value decomposition (SVD) [10, §2.3] of A:

$$A = U\,\Sigma\,V^T = \sum_{i=1}^{n} u_i\,\sigma_i\,v_i^T \tag{1.2}$$

where the left and right singular vectors u_i and v_i are the columns of the matrices U and V, respectively. σ_i are the singular values of A. They are nonnegative and they appear in non-increasing order:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0 \quad . \tag{1.2a}$$

This work was supported by the Department of Energy under Contract DE AC02 81ER 10996. The second author was also sponsored by National Science Foundation Grant Number DCR 8412314 and by a Fulbright Supplementary Grant.

† On leave from Copenhagen University Observatory, Øster Voldgade 3, DK-1350 Copenhagen K, Denmark.

The number of strictly positive singular values is the rank of A.

It is well-known that the minimum norm least squares solution to the problem (1.1) can be written in terms of the SVD of A as:

$$x = \sum_{i=1}^{\text{rank}(A)} \frac{u_i^T b}{\sigma_i} v_i \quad . \tag{1.3}$$

The ill-posed nature of (1.1), associated with the ill-conditioned matrix A, is caused by the appearance of one or more small singular values in the denominators of the sum in Eq. (1.3). The basic idea of the TSVD method is therefore to truncate the sums in (1.2) and (1.3) at a value $k < \text{rank}(A)$ such that all the small singular values are discarded. This corresponds to defining a new matrix A_k by:

$$A_k \equiv \sum_{i=1}^{k} u_i \, \sigma_i \, v_i^T \quad , \quad k < \text{rank}(A) \tag{1.4}$$

and substituting this matrix for A in Eq. (1.1), leading to a new least squares problem:

$$\min \| b - A_k x \|_2 \quad . \tag{1.5}$$

The minimum norm solution [10, §6.1] to this new problem is termed the *TSVD solution* x_k to the problem (1.1), and it is obviously obtained from Eq. (1.3) by truncation of the sum at k:

$$x_k \equiv \sum_{i=1}^{k} \frac{u_i^T b}{\sigma_i} v_i = \left(\sum_{i=1}^{k} v_i \, \sigma_i^{-1} \, u_i^T \right) b = A_k^+ \, b \quad . \tag{1.6}$$

The such defined matrix A_k^+ is the pseudoinverse of the matrix A_k in (1.4) and (1.5) (while it is an outer inverse of A). The problem of computing x_k is therefore, at least in theory, related to the problem of computing the matrix A_k^+. Although we need not compute A_k^+ explicitly, (1.6) does emphasize the two quantities required to compute the TSVD solution, namely the number $n-k$ of small singular values $\sigma_{k+1}, \ldots, \sigma_n$ to be discarded and the subspace spanned by the corresponding right singular vectors $v_{k+1}, \ldots, v_n$, which the TSVD solution must be orthogonal to. These two quantities also distinguish the TSVD solution from the standard QR-factorization based procedure for rank deficient problems, such as HFTI by Lawson & Hanson [15], which relies on the heuristic procedure of column pivoting for identifying the numerical rank from the appearance of small elements in the bottom part of the computed R matrix. In Section 2, we return to the problem of choosing the truncation parameter k and describe a method, based on the singular value spectrum of A, for selecting a proper value of k.

While we can compute x_k from the SVD via (1.6), this is an expensive procedure. Computation of Σ and V in the SVD (1.2) of A involves about $(m + 17/3\, n)\, n^2$ flops [10, Table 6.5-1]. In this paper we demonstrate how to compute the TSVD solution from *any QR-factorization* of A with much less computational effort: only about $(m + p)\, n^2$ flops, where p is a small integer. An outline of our algorithm is as follows. From any QR-factorization of A, we compute a rank-revealing QR-factorization (RRQR) [2], from which upper and lower bounds for the small singular values are easily obtained. This is used to compute the numerical rank k of A. The RRQR also yields approximations to the last $n-k$ singular vectors, which are then improved by means of inverse subspace iteration. Finally, these vectors and the above RRQR are both used to compute the TSVD solution.

It may seem as a surprise, compared to the work involved in a complete SVD computation, that we can compute x_k much 'cheaper'. The reason why it is possible to avoid most of the computational effort in computing the complete SVD of A is that only a part of the information, provided by the complete SVD, is actually required to compute the TSVD solution: namely the $n-k$ smallest singular values and the corresponding singular vectors. And this information can, as we shall see, be extracted from a QR-factorization of A.

The TSVD solution is related to some other solutions to (1.1) obtained by other methods. For example, it is similar to the so-called 'deflated solutions' as defined by Chan [1], as well as the solution obtained by 'regularization in standard form' as shown by Hansen [11]. It is, however, in general

different from the solution obtained by 'subset selection' as described by Golub, Klema & Stewart [9], although the present algorithm may also be used as a first step in performing subset selection (see Section 6).

2. The Numerical Rank and the General Least Squares Solution

In this section we consider the selection of a proper value of the truncation parameter k in the truncated SVD expansion (1.3). The usual approach is to let k be equal to the *numerical rank* of A, defined as the number of singular values of A strictly larger than a certain threshold τ:

$$\sigma_1 \geq \cdots \geq \sigma_k > \tau \geq \sigma_{k+1} \quad . \tag{2.1}$$

This is a reasonable approach when there is a well-defined gap between the singular values σ_k and σ_{k+1} since in this case the numerical rank k is well-determined with respect to τ. If, however, there is no such gap present in the singular value spectrum of A, the problem of choosing k is much more complicated since then the numerical rank may not be well-determined with respect to τ. This problem is also reflected in the perturbation theory for the TSVD. Let E be a perturbation of A in (1.1) such that $\|E\|_2 < \sigma_k - \sigma_{k+1}$. Then it is shown in [11] that the perturbed pseudoinverse $(A+E)_k^+$ satisfies:

$$\frac{\|A_k^+ - (A+E)_k^+\|_2}{\|A_k^+\|_2} \leq 3 \frac{\eta_k}{(1-\eta_k)(1-\eta_k-\omega_k)} \tag{2.2}$$

where we have defined

$$\eta_k \equiv \frac{\|E\|_2}{\sigma_k} \quad , \quad \omega_k \equiv \frac{\sigma_{k+1}}{\sigma_k} \quad . \tag{2.2a}$$

From this result, it is seen that a small upper bound on the perturbation of the pseudoinverse A_k^+ (and thus on the TSVD solution x_k) depends not only on a small η_k, but also on a small ω_k, which means that there must be a distinct gap in the singular value spectrum between σ_k and σ_{k+1}. As we shall see in Section 4, this requirement on ω_k also enters in our algorithm. We will therefore make the assumption that such a gap is present in the singular value spectrum of A.

Since we have discarded $n-k$ singular values of A, such that the matrix A_k (1.4) has exact rank k, there are exactly k linearly independent solutions to the homogeneous problem associated with (1.5), and these are linear combinations of the last $n-k$ right singular vectors of A. The *general solution* to (1.5) can therefore formally be written as:

$$x_{GS} = x_k + \sum_{i=k+1}^{n} c_i \, v_i \quad , \quad c_i \text{ arbitrary} \tag{2.3}$$

in which $v_{k+1}, \ldots, v_n$ are the last $n-k$ right singular vectors of the matrix V in (1.2). The TSVD solution x_k (1.6) is the unique solution to (1.5), among all x_{GS}, which minimizes $\|x_{GS}\|_2$ (hence the name 'minimum norm solution'). Define the matrix $V_o = [\, v_{k+1} \cdots v_n \,] \in R^{n \times (n-k)}$. From (2.1) it immediately follows that:

$$\|A \, V_o\|_2 < \tau \quad . \tag{2.4}$$

Hence, the subspace spanned by these vectors will be termed the *numerical null-space* N_k *of* A:

$$N_k \equiv \text{span} \, \{v_{k+1}, \ldots, v_n\} \quad . \tag{2.5}$$

The general solution x_{GS} is completely specified once x_k and the matrix V_o has been computed. This general solution is required in a number of applications such as total linear least squares (TLLS) [10, §12.3], solution of nonlinear equations by Gauss-Newton with a rank deficient Jacobian [8, §4.7.5], and the study of Fredholm integral equations of the first kind [6,12]. These examples satisfy the above assumption that there is a well-defined gap in the singular value spectrum.

3. The Rank Revealing QR-Factorization

The first step of our algorithm is to determine the numerical rank k of the matrix A. Of course, we could compute the complete SVD and examine the singular values. However, this technique is quite expensive, so instead we want to extract the information from a QR-factorization of A. To be precise, we want a QR-factorization of A in the particular form:

$$A\,\Pi = Q\,R = Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{matrix} k = \text{numerical rank} \\ n-k \end{matrix} \tag{3.1}$$

with $\|R_{22}\|_2$ small such that the QR-factorization (3.1) exhibits any numerical rank deficiency in A. This particular form is termed the *rank revealing QR-factorization (RRQR)* of A [2]. The problem of computing the RRQR is equivalent to determining a column permutation Π such that $\|R_{22}\|_2$ is small. It is emphasized that the usual column pivoting strategy does not guarantee this [10, p. 167]. Recently, Chan [2] has described an algorithm for choosing a permutation Π that guarantees $\|R_{22}\|_2$ to be small. The permutation turns out to be identical to one proposed by Foster [7] for subset selection. In addition, Chan's algorithm also gives upper and lower bounds for the singular values of A plus an approximate null space.

To motivate the strategy used in [2], consider the following lemma.

Lemma 3.1. *Given any column permutation* Π *and an* $X \in R^{n\times(n-k)}$ *such that*

$$\|A\,X\|_2 = \varepsilon \quad , \tag{3.2}$$

the QR-factorization (3.1) of $A\,\Pi$ *yields an* R_{22} *that satisfies*

$$\|R_{22}\|_2 \le \varepsilon\,\|Y_2^{-1}\|_2 \quad , \tag{3.3}$$

where the matrix Y_2 *is defined by:*

$$\Pi^T X = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix} \quad . \tag{3.4}$$

Proof. Using (3.1), (3.2) and (3.4), we immediately obtain:

$$\varepsilon = \|A\,X\|_2 = \|R\,\Pi^T X\|_2 \ge \|R_{22}\,Y_2\|_2 \ge \|R_{22}\|_2 \|Y_2^{-1}\|_2^{-1} \;\Rightarrow\; \|R_{22}\|_2 \le \varepsilon \|Y_2^{-1}\|_2 \quad . \quad \square$$

The above lemma shows that, if we can identify an approximate null space of A of dimension $n-k$, represented by a matrix X that gives a small ε in (3.2), then the near rank-deficiency of A is revealed by a small $R_{22} \in R^{(n-k)\times(n-k)}$ in the QR-factorization (3.1) provided that we use a permutation Π such that $\|Y_2^{-1}\|_2$ is not large. The strategy in computing the RRQR is therefore to find a matrix X such that ε is small and to find a permutation Π of the rows of X such that $\|Y_2^{-1}\|_2$ is as small as possible. For the special case $k = n-1$, one can take $X = v_n$ and Π the permutation that brings the largest element in absolute value of v_n to the last position [9]. The vector v_n can be estimated as in the LINPACK condition estimator [4, 5] or computed by inverse iteration [16]. In the general case $k \le n-1$, one approach is to take $X = V_o$ giving $\varepsilon = \sigma_{k+1}$ and Π determined in such a way that the bottom $(n-k)\times(n-k)$ submatrix of $\Pi^T V_o$ is as well-conditioned as possible, as proposed in [9]. The idea of the RRQR algorithm in [2] is to devise a more efficient algorithm based on the QR-factorization by repeated application of the $k = n-1$ procedure. Essentially, the matrices X and Π are constructed from the right singular vectors corresponding to the smallest singular value of appropriately chosen, increasingly smaller subsets of columns of A.

Algorithm RRQR:

1. Compute any QR-factorization of A: $A\,\Pi = Q\,R$ and set $k \leftarrow n$.
2. Loop to identify the small singular values of A.

2a. Partition $R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix}$.

2b. Get an initial guess δ_k of the smallest singular value and the corresponding right singular vector $\overline{w}_k$ of R_{11}, e.g. by using the LINPACK condition estimator [4, 5].

2c. If the initial guess δ_k is numerically nonzero, then improve δ_k and $\overline{w}_k$ by means of inverse iteration applied implicitly to $R_{11}^T R_{11}$.

2d. If $\delta_k > \tau$ then k is the numerical rank of A (see (3.6) below): goto 3.

2e. Determine a permutation $\overline{P}$ such that $|(\overline{P}^T \overline{w}_k)_k| = \|\overline{P}^T \overline{w}_k\|_\infty$.

2f. Compute the QR-factorization: $R_{11}\overline{P} = \overline{Q}\,\overline{R}_{11}$.

2g. Update Π, Q, and R: $\Pi \leftarrow \Pi \begin{bmatrix} \overline{P} & 0 \\ 0 & I_{n-k} \end{bmatrix}$, $Q \leftarrow Q \begin{bmatrix} \overline{Q} & 0 \\ 0 & I_{n-k} \end{bmatrix}$, $R \leftarrow \begin{bmatrix} \overline{R}_{11} & \overline{Q}^T R_{12} \\ 0 & R_{22} \end{bmatrix}$.

2h. Assign $\begin{bmatrix} \overline{w}_k \\ 0 \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix}$ to the k'th column of W, and compute: $W \leftarrow \begin{bmatrix} \overline{P}^T & 0 \\ 0 & I_{n-k} \end{bmatrix} W$.

2i. Let $k \leftarrow k-1$.

3. End.

In step 2c we accept a singular value as numerically nonzero if it exceeds $n\,\varepsilon_M\,\sigma_1$, where ε_M is the machine epsilon. Steps 2e and 2f should be implemented such as described by [7] and [2] to reduce the computational effort to $O(\frac{1}{2}(n-k)n^2)$ flops. The dominating effort is therefore the backsubstitutions in step 2c which means that the numerical rank k of A can be determined in $O(p_1(n-k)n^2)$ flops, where p_1 is the average number of iterations used.

From the definition of $\overline{w}_i$ in steps 2b-c of RRQR, it follows that Π and W satisfy:

$$\|A\,\Pi W\|_F^2 = \sum_{i=k+1}^{n} \delta_i^2 \quad . \tag{3.5}$$

Since the δ_i's are the smallest singular values of increasingly smaller *subsets* of columns of A, it immediately follows from the interlacing inequalities for singular values [10, Corollary 8.3-3] that the δ_i are nonincreasing for increasing values of i and that:

$$\delta_i \le \sigma_i \quad , \quad k \le i \le n \quad . \tag{3.6}$$

Therefore, the RRQR algorithm can be viewed as a method for identifying an approximate null space of A represented by the matrix $X = \Pi W$.

The following theorem gives upper and lower bounds on the smallest singular values of A, which are by-products of algorithm RRQR.

Theorem 3.2. *Let the quantities δ_i be computed by means of algorithm RRQR above. Also, let the matrix $W \in R^{n\times(n-k)}$ be partitioned as:*

$$W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix} \tag{3.7}$$

and let W_2^i and R_{22}^i denote the bottom $(n-i)\times(n-i)$-blocks of W_2 and R_{22} for $k \le i \le n$, respectively. Then:

$$\frac{\sigma_i}{\sqrt{n-i}\;\|(W_2^i)^{-1}\|_2} \le \delta_i \le \sigma_i \le \|R_{22}^i\|_2 \le \sigma_i \sqrt{n-i}\;\|(W_2^i)^{-1}\|_2 \quad . \tag{3.8}$$

Also, we have the a priori bound:

$$\|(W_2^i)^{-1}\|_2 < \sqrt{n}\;2^{n-i} \quad . \tag{3.9}$$

Proof. We shall prove each of the four inequalities in (3.8) starting from the right. The first follows from Lemma 3.1, (3.5) and (3.6). The second follows from the variational characterization of singular values, and the next one is simply (3.6). The last one follows from (3.3) and the second inequality in (3.8).

To prove (3.9) we use the same technique as in [7], but here applying the norm $\|\cdot\|_2$. Define $Z = W_2^i$. Due to the construction of W, we have:

$$n^{-1/2} \le |z_{jj}| \le 1 \quad , \quad 1 \le j \le i$$

$$0 \le |z_{j\zeta}| \le n^{-1/2} \quad , \quad 1 \le j, \zeta \le i \quad , \quad j \ne \zeta \quad .$$

Let $\eta = n - i$ and $y = Z x$ with $\|y\|_2 = 1 \Rightarrow |y_j| \le 1$. Then, from the equation:

$$x_j = z_{jj}^{-1} \left[y_j - \sum_{\zeta=j+1}^{\eta} z_{j\zeta} x_\zeta \right] \quad , \quad 1 \le j \le i \quad ,$$

we get

$$|x_j| \le \sqrt{n} + \sum_{\zeta=j+1}^{\eta} |x_\zeta| \quad , \quad 1 \le j \le i \quad .$$

It is easy to show by induction that $|x_j| \le \sqrt{n}\ 2^{j-1}$, so that:

$$\|x\|_2^2 = n\,[4^0 + 4^1 + \cdots + 4^{\eta-1}] = n\,\frac{4^\eta - 1}{\eta - 1} < n\,4^\eta$$

and therefore $\|x\|_2 < \sqrt{n}\ 2^\eta$. From the definition of the matrix 2-norm it then follows that:

$$\|(W_2^i)^{-1}\|_2 = \|Z^{-1}\|_2 = \min_x \frac{\|x\|_2}{\|Z x\|_2} = \min_x \frac{\|x\|_2}{\|y\|_2} = \min_x \|x\|_2 < \sqrt{n}\ 2^\eta \quad . \qquad \square$$

Theorem 3.1 implies that, as long as the numerical rank $k \approx n$ such that the term 2^{n-i} in (3.9) is small, the quantities δ_i and $\|R_{22}^i\|_2$ are guaranteed to be tight upper and lower bounds of the singular values σ_i. This means that the algorithm RRQR will determine the correct numerical rank k as long as that there is a well-defined gap between singular values σ_k and σ_{k+1}.

4. Determination of the Numerical Null-Space

Algorithm RRQR gives the numerical rank k and an approximate null space, represented by the matrix W. However, the TSVD solution requires the true numerical null-space N_k of A. In this section, we show how to refine matrix W to obtain an accurate basis for N_k.

Before we give the refinement procedure, we first study how good an approximation W is. From (3.5) and (3.6) it immediately follows that:

$$\|A\,\Pi W\|_2^2 \le \|A\,\Pi W\|_F^2 = \sum_{i=k+1}^{n} \delta_i^2 \le \sum_{i=k+1}^{n} \sigma_i^2 \le (n-k)\,\sigma_{k+1}^2 \quad ,$$

and therefore:

$$\frac{\|A\,\Pi W\|_2}{\|A\|_2} \le \sqrt{n-k}\ \omega_k \quad , \tag{4.1}$$

where ω_k is defined in (2.2a). This result suggests that the range of ΠW is a good approximation to the numerical null-space N_k provided that ω_k is small. That this is true follows from the following theorem:

Theorem 4.1. *Let the matrix W be partitioned as in Eq. (3.7). Then the subspace angle θ between the range of ΠW and the numerical null-space N_k of A is bounded as:*

$$\sin\theta \le \left(1+\sqrt{n-k}\,\|W_2^{-1}\|_2\right)\omega_k \tag{4.2}$$

where $\omega_k = \sigma_{k+1}/\sigma_k$, and where $\|W_2^{-1}\|_2 \le \sqrt{n}\,2^{n-k}$ due to Eq. (3.9).

Proof. First, we define a QR-factorization of $\Pi W = \tilde{V}_o R_W$ and obtain the bound:

$$\|R_W^{-1}\|_2 = \|W^+\Pi^T\tilde{V}_o\|_2 \le \|W^+\|_2 = \sigma_{\min}(W)^{-1} \le \sigma_{\min}(W_2)^{-1} = \|W_2^{-1}\|_2 \quad . \tag{4.3}$$

In terms of the SVD (1.2) of A, we then define the matrices

$$\begin{aligned} \Sigma_k &= \mathrm{diag}\,(\sigma_1, \ldots, \sigma_k) & \Sigma_o &= \mathrm{diag}\,(\sigma_{k+1}, \ldots, \sigma_n) \\ U_k &= [u_1 \cdots u_k] & U_o &= [u_{k+1} \cdots u_n] \\ V_k &= [v_1 \cdots v_k] & V_o &= [v_{k+1} \cdots v_n] \end{aligned} \tag{4.4}$$

and write: $A\tilde{V}_o = U_k\Sigma_k V_k^T\tilde{V}_o + U_o\Sigma_o V_o^T\tilde{V}_o$. Finally, let $\phi = \frac{1}{2}\pi - \theta$ be the angle between the subspaces range$(\tilde{V}_o)$ and range(V_o). Then, from [10, pp. 428-429]:

$$\begin{aligned} \sin\theta = \cos\phi &= \sigma_{\min}(V_k^T\tilde{V}_o) = \sigma_{\min}(\Sigma_k^{-1}U_k^TU_k\Sigma_kV_k^T\tilde{V}_o) \\ &\le \|\Sigma_k^{-1}\|_2\,\|U_k^T\|_2\,\sigma_{\min}(U_k\Sigma_kV_k^T\tilde{V}_o) = \sigma_k^{-1}\sigma_{\min}(U_k\Sigma_kV_k^T\tilde{V}_o) \\ &= \sigma_k^{-1}\sigma_{\min}(A\tilde{V}_o - U_o\Sigma_oV_o^T\tilde{V}_o) \le \sigma_k^{-1}\left(\sigma_{\min}(U_o\Sigma_oV_o^T\tilde{V}_o) + \|A\tilde{V}_o\|_2\right) \\ &\le \sigma_k^{-1}\left(\|U_o\|_2\,\|\Sigma_o\|_2\,\sigma_{\min}(V_o^T\tilde{V}_o) + \|A\tilde{V}_oR_WR_W^{-1}\|_2\right) \\ &\le \sigma_k^{-1}\left(\sigma_{k+1}\cos\phi + \|A\Pi W\|_2\,\|R_W^{-1}\|_2\right) \le \sigma_k^{-1}\left(\sigma_{k+1} + \sqrt{n-k}\,\sigma_{k+1}\|W_2^{-1}\|_2\right) \\ &\le \left(1+\sqrt{n-k}\,\|W_2^{-1}\|_2\right)\omega_k \quad . \end{aligned}$$

Here, we have used (4.3) and the formulas $\sigma_i(A+B) \le \sigma_i(A) + \|B\|_2$ and $\sigma_i(AB) \le \sigma_i(A)\,\|B\|_2$ [14, p. 89]. □

Theorem 4.1 means that if ω_k is small (corresponding to a large gap in the singular value spectrum of A) and $k \approx n$, then the range of ΠW is a good approximation to the numerical null-space N_k since $\|W_2^{-1}\|_2$ can not be large, cf. (3.9). Hence, W is a good starting matrix for simultaneous inverse iteration with R^TR in order to determine an accurate basis for N_k, guaranteeing fast convergence fast within a few steps. This leads to the following algorithm SPIT (subspace iteration) to determine a matrix $\bar{V}_o$ whose columns are orthonormal basis vectors for N_k [3].

Algorithm SPIT:

1. Let $i \leftarrow 0$ and $\eta_o \leftarrow$ number of numerically nonzero singular values.
2. Set $\bar{V}^{(0)} \leftarrow [w_{k+1} \cdots w_{\eta_o}]$, orthonormalize its columns, and set $\bar{R} \leftarrow$ upper $\eta_o \times \eta_o$-block of R.
3. Subspace iteration. Repeat until $\sin\psi < \varepsilon$:
   3a. $i \leftarrow i+1$, $\bar{U}^{(i)} \leftarrow \bar{R}^{-T}\bar{V}^{(i-1)}$, $\bar{V}^{(i)} \leftarrow \bar{R}^{-1}\bar{U}^{(i-1)}$. Scaling is necessary to avoid overflow!
   3b. Orthonormalize the columns of $\bar{V}^{(i)}$.
   3c. Estimate $\sin\psi$, where ψ is the subspace angle between range$(\bar{V}^{(i-1)})$ and range$(\bar{V}^{(i)})$.
4. Orthonormalize the columns of $\bar{U}^{(i)}$ to get $\bar{U}_o$.
5. If $\eta_o < n$ then append $\bar{V}^{(i)}$ with the last $n-\eta_o$ columns of W.
6. Let $\bar{V}_o \leftarrow \Pi\bar{V}^{(i)}$.
7. End.

We have avoided the numerical problems in step 3a associated with the numerically zero

singular values by not iterating on the corresponding subspace, since this subspace is already computed with a sufficient precision $n\,\varepsilon_M\,\sigma_1$, cf. step 2c of RRQR. Further, this subspace will remain orthogonal to the span of the columns of $\bar{V}^{(i)}$ such that the final $\bar{V}^{(i)}$ produced in step 5 will have orthogonal columns. The quantity ε in step 3 determines the accuracy of the computed $\bar{V}_o$, and should be set according to the required accuracy of the computed TSVD solution x_k. In step 3c, notice that $\sin\psi$ is given by [10, pp. 428-429]:

$$\sin\psi \equiv \|(I-\bar{V}^{(i)}\bar{V}^{(i)T})\,\bar{V}^{(i-1)}\|_2 = \|\bar{P}^{(i)}\bar{V}^{(i-1)}\|_2 \tag{4.5}$$

in which $\bar{P}^{(i)}$ is the projection matrix for orthogonal projection onto the orthogonal complement of range$(\bar{V}^{(i)})$. Hence, the matrix $\bar{P}^{(i)}\bar{V}^{(i-1)}$ can be computed simply by orthonormalizing the columns of $\bar{V}^{(i-1)}$ with respect to the columns of $\bar{V}^{(i)}$ (e.g. by MGS). The quantity $\sin\psi$ can then easily be estimated by a simple estimate of $\|\bar{P}^{(i)}\bar{V}^{(i-1)}\|_2$. The computational effort is therefore dominated by the subspace iterations in step 3a. Hence, $\bar{V}_o$ is determined in $O(p_2(n-k)n^2)$ flops, where p_2 is the number of iterations.

5. Determination of the TSVD Solution

The remaining problem now is to compute the TSVD solution x_k (2.3) from the quantities already obtained from RRQR and SPIT. Suppose that the matrix R in (3.1) has no exact zero singular values. In theory, one could then compute the vector $R^{-1}Q^Tb$, which is a member of the general solution x_{GS} in (2.3), and then deflate this solution; i.e., orthogonalize with respect to the numerical nullspace N_k. In practice, however, the matrix R is usually highly ill-conditioned which will completely destroy the accuracy of the such computed TSVD solution. This is obvious because the computed vector

$$R^{-1}Q^Tb = \sum_{i=1}^{n} \sigma_i^{-1}\,u_i^TQ^Tb \tag{5.1}$$

will be dominated by those contributions, corresponding to the small singular values $\sigma_{k+1}, \ldots, \sigma_n$, that are to be removed again in the deflation process. The rounding errors associated with this component will therefore contaminate the computed solution.

The problem is overcome by also deflating the right-hand side b with respect to the subspace span $\{u_{k+1}, \ldots, u_n\}$, where u_i are the columns of U in the SVD (1.2) of A. This idea is similar to those used in [1, 16]. Notice first that if A has $n-\eta_o$ numerically zero singular values then the corresponding left singular vectors of A are given with sufficient accuracy by:

$$[\,u_{\eta_o+1}\cdots u_n\,] = Q\begin{bmatrix}0\\ I_{n-\eta_o}\end{bmatrix}\begin{matrix}\eta_o\\ n-\eta_o\end{matrix}, \tag{5.2}$$

which means that deflation with respect to the associated subspace simply corresponds to neglection of the last $n-\eta_o$ elements of Q^Tb. To deflate with respect to the remaining left singular vectors $\{u_{k+1}, \ldots, u_p\}$, corresponding to the small but numerically nonzero singular values, one can deflate Q^Tb with respect to the columns of $\bar{U}_o$ provided by step 4 of SPIT. The algorithm TSOL for computing the TSVD solution x_k then becomes.

Algorithm TSOL:

1. Let $\beta \in R^{\eta_o}$ consist of the first η_o components of Q^Tb (η_o defined in step 1 of SPIT).
2. Compute $\xi \leftarrow \bar{R}^{-1}(I_{\eta_o} - \bar{U}_o\,\bar{U}_o^T)\,\beta$, where $\bar{R}$ and $\bar{U}_o$ are defined in steps 2 and 4 of SPIT.
3. If $\eta_o < n$, append ξ with zeros: $\xi \leftarrow \begin{bmatrix}\xi\\ 0\end{bmatrix}\begin{matrix}\eta_o\\ n-\eta_o\end{matrix} \in R^n$.
4. Compute the TSVD solution $x_k = (I_n - \bar{V}_o\,\bar{V}_o^T)\,\Pi\,\xi$.

5. End.

This algorithm involves one backsubstitution such that x_k is determined in $\frac{1}{2}n^2$ flops.

Summarizing, the operations count for computing the TSVD solution x_k via RRQR, SPIT, and TSOL can be shown to be bounded:

$$\left[m + (n-k)(p_1+p_2+\tfrac{1}{2}) + p_1 + \tfrac{1}{2} \right] n^2 \quad \text{flops} \tag{5.3}$$

where p_1 is the average number of inverse iterations per singular value in RRQR, and p_2 is the number of inverse subspace iteratons used in SPIT. Thus, if the numerical rank $k \approx n$, then the RRQR-SPIT-TSOL procedure does not cost much more computational effort than the cost of computing one QR-factorization.

6. Extensions

Although the above algorithm is designed for problems with $m \geq n$, it can easily be extended to the underdetermined case $B \in R^{m \times n}$ with $m < n$. In this case, first compute a QR-factorization of B^T:

$$B^T = [Q_1, Q_2] \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \begin{matrix} m \\ n-m \end{matrix} \tag{6.1}$$

and then apply our algorithm to $A = R_1^T$. Obviously, the TSVD solution $x_k^{(B)}$ lies in the range of Q_1, and the columns of Q_2 span an exact null space of B. Hence, the general solution, represented by $x_k^{(B)}$ and $\overline{V}_o^{(B)}$, is obtained from the results $x_k^{(A)}$ and $\overline{V}_o^{(A)}$ of the RRQR-SPIT-TSOL algorithm by setting:

$$x_k^{(B)} \leftarrow Q_1 x_k^{(A)} \quad , \quad \overline{V}_o^{(B)} \leftarrow [\overline{V}_o^{(A)}, Q_2] \quad . \tag{6.2}$$

We notice that the our algorithm may also be used for subset selection as described by Golub, Klema, & Stewart [9]. In their method, the column pivoting for subset selection is determined from examining the matrix V_o in (4.4) obtained from a complete SVD of A. Since this matrix V_o and the matrix $\overline{V}_o$ computed by SPIT span the same subspace N_k, we can apply the same procedure to $\overline{V}_o$ and obtain a similar subset selection procedure. This permutation is, in general, different from from the ones obtained from the procedure in [9] as well as the permutations provided by RRQR and the method of Foster [7]. However, they should all produce residuals of the same size.

Finally, we stress that we have made no a priori assumption about the pivoting in the initial QR-factorization of A. Hence, if A is sparse, this QR-factorization can be computed with pivoting for sparsity. Also, the QR-factorization in step 2f of RRQR can be performed by taking advantage of the sparsity of R as described by Heath [13]. This means that our algorithm is also suited for sparse matrices as long as the numerical rank $k \approx n$ (since the matrices $\overline{U}^{(i)}$ and $\overline{V}^{(i)}$ are *full* matrices).

7. Numerical Examples

In this section we give a few illustrative examples of the application of our algorithm. The first four examples are constructed to illustrate the features and properties of the algorithm; the fifth example is a practical example from the application of integral equations in two-dimensional potential theory.

For examples 1-4, we generated four small matrices of dimensions $m = 25$ and $n = 10$ with different values of ω_k. All matrices were generated by replacing the singular values in the SVD of randomly generated matrices. The first seven singular values are fixed at:

$$\sigma_1 = 1\ ,\ \sigma_2 = 0.5\ ,\ \sigma_3 = 0.2\ ,\ \sigma_4 = 0.1\ ,\ \sigma_5 = 0.05\ ,\ \sigma_6 = 0.02\ ,\ \sigma_7 = 0.01 \tag{7.1}$$

while the last singular values σ_8, σ_9, and σ_{10} are varied. In Table 1 we tabulate, for each of the small singular values σ_i, the upper and lower bounds δ_i and $\|(R_{22}^i)^{-1}\|_2$ from algorithm RRQR and the number of inverse iterations used in RRQR to obtain it.

Consider first the quality of the bounds on the singular values. In all four examples, these bounds are very tight, and the numerical rank k is easily identified as $k = 7$. Moreover, the number of inverse iterations needed is very small (although the number of iterations required to estimate $\sigma_k = \sigma_7$ increases with increasing ω_k).

In Table 2 we tabulate results associated with the null-space computation and the TSVD solutions. The first part of the table compares the angle θ between the approximate null-space, as computed by RRQR, and the exact N_k, with the bound given in (4.2). Notice that the columns of the matrix ΠW indeed give an approximate basis for N_k, and that the quality (as expected) depends on the quantity ω_k. The upper bound for $\sin\theta$ is overly pessimistic; but it indicates the general behavior of $\sin\theta$.

The second part of Table 2 shows the performance of algorithm SPIT. We see that as long as there is a well-determined gap between σ_k and σ_{k+1} then the simultaneous inverse iterations converge fast. In example 4, there is hardly any gap at all, and the iterations converge very slowly. The accuracy of the computed $\overline{V}_o$ reflects the stopping criteria $\varepsilon = 10^{-10}$ used in step 3 of SPIT.

In the last part of Table 2 we give the results from solving the least squares problem (1.1) by means of algorithm TSOL. Right-hand sides $\boldsymbol{b}$ were generated by the formula:

$$\boldsymbol{b} = \sum_{i=1}^{m} \boldsymbol{u}_i \tag{7.2}$$

such that $\boldsymbol{b}$ has large components in both the range of A_k and its orthogonal complement. We computed two solutions:

$\tilde{\boldsymbol{x}}_k$: computed by TSOL,

$\tilde{\tilde{\boldsymbol{x}}}_k$: computed by TSOL *without* deflation of the right-hand side $\boldsymbol{b}$.

These two solutions are compared with the true solution $\boldsymbol{x}_k$. We see that the relative accuracy of $\tilde{\boldsymbol{x}}_k$ is the same as the accuracy of $\overline{V}_o$. We also see that deflation of $\boldsymbol{b}$ is in fact necessary when very small singular values are present in the matrix R.

The conclusion to be drawn from these numerical examples is that our algorithm performs well as long as the assumption $\omega_k \ll 1$ is satisfied. And even when ω_k is not small, our algorithm is in fact able to give good results at the cost of the larger computational effort involved in the large number of simultaneous inverse iterations.

The practical example comes from the numerical analysis of linear algebraic equations derived from first kind integral equations in two-dimensional potential theory [12]. In particular, we consider the homogeneous Fredholm integral equations of the first kind:

$$\int_0^2 \ln|s-x|\, f(s)\, ds = 0 \quad , \quad x \in [0,2] \quad . \tag{7.3}$$

The solution to this equation has a physical interpretation, for example, as the steady-state charge distribution on a line segment form $s = -2$ to $s = 2$. Due to symmetry it is only necessary to consider the interval $x \in [0,2]$. Choosing $m = n = 10$ and discretizing (7.3) as described in [12] we are led to a matrix A which is then processed by RRQR and SPIT, and the results are shown in Table 3. The results *guarantee* that the numerical rank of A is $k = 9$ (i.e., *one* small singular value). The solution $f(s)$ to (7.3) can then be computed from the numerical null-space of A; i.e., from $\overline{V}_o \in R^{n\times 1}$ produced by SPIT, and the such computed $f(s)$ agrees with the true TSVD solution as computed in [12]. Hence, we have shown that the study of the integral equation (7.3) can be performed perfectly well by means

of our algorithm, thus avoiding the large computational effort of computing the complete SVD of A.

8. Conclusion

The TSVD solution to a rank deficient least squares problem usually requires computation of the singular value decomposition (SVD) of the matrix, which involves a large amount of computational effort compared to a QR-factorization. In this paper, we have shown that the TSVD solution can also be computed much more efficiently from a rank revealing QR-factorization (RRQR) of the matrix, followed by inverse subspace iterations to improve the estimated null space of the matrix. Unlike methods based on heuristic procedures for rank determination, our algorithm is guaranteed to perform reliably when the matrix has a well-determined numerical rank, as is the case in a number of practical applications. This suggests the TSVD method as a favorable alternative to regularization for such problems.

Acknowledgements

The authors would like to thank Prof. Gene Golub for his hospitality during our visits at the Dept. of Computer Science at Stanford University. The second author thanks Prof. Hans Bruun Nielsen and Prof. Ake Björck for pointing out the usefulness of inverse subspace iterations in this context.

References

[1] T.F. Chan, *Deflated Decomposition of Solutions of Nearly Singular Systems*, SIAM J. Numer. Anal., 21 (1984), pp. 738-754.

[2] T.F. Chan, *Rank Revealing QR-Factorizations*, Research Report YALEU/DCS/RR-398, Dept. Computer Science, Yale University, June, 1985. To appear in a special issue of Lin. Alg. & Its Appl. in honor of J.H. Wilkinson. A shorter version is: Alternative to the SVD: Rank Revealing QR-Factorizations, Proceedings 696, Advanced Algorithms and Architectures for Signal Processing, SPIE's 30th Annual Technical Symposium, San Diego, August, 1986.

[3] T.F. Chan & D.C. Resasco, *Generalized Deflated Block-Elimination*, SIAM J. Numer. Anal., 23 (1986), pp. 913-924.

[4] A.K. Cline, C.B. Moler, G.W. Stewart, & J.H. Wilkinson, *An Estimate for the Condition Number of a Matrix*, SIAM J. Numer. Anal., 16 (1979), pp. 368-375.

[5] J.J. Dongarra, J.R. Bunch, C.B. Moler, & G.W. Stewart, *LINPACK Users Guide*, SIAM Publications, Philadelphia, 1979.

[6] L. Eldén, *The Numerical Solution of a Non-Characteristic Cauchy Problem for a Parabolic Equation*, in P. Deuflhard & E. Hairer (Eds.), *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, Birkhäuser, Basel, 1983.

[7] L.V. Foster, *Rank and Null Space Calculations Using Matrix Decomposition without Column Interchanges*, Lin. Alg. & Its Appl., 74 (1986), pp. 47-71.

[8] P.E. Gill, W. Murray, & M.H. Wright, *Practical Optimization*, Academic Press, London, 1981.

[9] G.H. Golub, V. Klema, & G.W. Stewart, *Rank Degeneracy and Least Squares Problems*, Technical Report TR-456, Dept. Computer Science, University of Maryland, College Park, 1976.

[10] G.H. Golub & C.F. Van Loan, *Matrix Computations*, North Oxford Academic, New York, 1983.

[11] P.C. Hansen, *The Truncated SVD as a Method for Regularization*, Report NA-86-36, Dept. Computer Science, Stanford University, October, 1986. Submitted to BIT for publication.

[12] P.C. Hansen & S. Christiansen, *An SVD Analysis of Linear Algebraic Equations Derived from First Kind Integral Equations*, J. Comp. Appl. Math., 12&13 (1985), pp. 3341-357.

[13] M.T. Heath, *Some Extensions of an Algorithm for Sparse Linear Least Squares Problems*, SIAM J. Sci. Stat. Comput., 3 (1982), pp. 223-237.

[14] A.S. Householder, *The Theory of Matrices in Numerical Analysis*, Dover Publications, New York, 1974.

[15] C.L. Lawson & R.J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, N.J., 1974.

[16] G.W. Stewart, *On the Implicit Deflation of Nearly Singular Systems of Linear Equations*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 136-140.

Example	i	δ_i	σ_i	$\|(R_{22}^i)^{-1}\|_2$	RRQR iterations
1	7	$8.28 \cdot 10^{-3}$	$1.00 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	3
	8	$5.27 \cdot 10^{-18}$	$9.71 \cdot 10^{-18}$	$1.89 \cdot 10^{-17}$	2
	9	$5.85 \cdot 10^{-18}$	$7.38 \cdot 10^{-18}$	$1.43 \cdot 10^{-17}$	0
	10	$8.95 \cdot 10^{-18}$	$4.48 \cdot 10^{-18}$	$9.87 \cdot 10^{-18}$	0
2	7	$7.79 \cdot 10^{-3}$	$1.00 \cdot 10^{-2}$	$2.14 \cdot 10^{-2}$	5
	8	$9.86 \cdot 10^{-6}$	$1.00 \cdot 10^{-5}$	$1.92 \cdot 10^{-5}$	2
	9	$9.15 \cdot 10^{-7}$	$1.00 \cdot 10^{-6}$	$1.81 \cdot 10^{-6}$	2
	10	$1.00 \cdot 10^{-7}$	$1.00 \cdot 10^{-7}$	$1.75 \cdot 10^{-7}$	2
3	7	$7.81 \cdot 10^{-3}$	$1.00 \cdot 10^{-2}$	$2.14 \cdot 10^{-2}$	5
	8	$9.86 \cdot 10^{-4}$	$1.00 \cdot 10^{-3}$	$1.91 \cdot 10^{-3}$	2
	9	$9.15 \cdot 10^{-5}$	$1.00 \cdot 10^{-4}$	$1.81 \cdot 10^{-4}$	2
	10	$1.00 \cdot 10^{-5}$	$1.00 \cdot 10^{-5}$	$1.75 \cdot 10^{-5}$	2
4	7	$8.30 \cdot 10^{-3}$	$1.00 \cdot 10^{-2}$	$2.15 \cdot 10^{-2}$	5
	8	$4.94 \cdot 10^{-3}$	$5.00 \cdot 10^{-3}$	$9.00 \cdot 10^{-3}$	3
	9	$1.87 \cdot 10^{-3}$	$2.00 \cdot 10^{-3}$	$3.60 \cdot 10^{-3}$	2
	10	$1.00 \cdot 10^{-3}$	$1.00 \cdot 10^{-3}$	$1.71 \cdot 10^{-3}$	2

Table 1.

Example	1	2	3	4
$\omega_k = \sigma_{k+1}/\sigma_k$	$9.71 \cdot 10^{-16}$	$1.00 \cdot 10^{-3}$	$1.00 \cdot 10^{-1}$	$5.00 \cdot 10^{-1}$
$\left(1+\sqrt{n}\,\|\|W_2^{-1}\|\|_2\right)\omega_k$	$9.63 \cdot 10^{-15}$	$5.05 \cdot 10^{-3}$	$5.00 \cdot 10^{-1}$	$2.52 \cdot 10^{0}$
$\sin\theta\{\text{range}(\Pi W), N_k\}$	$7.43 \cdot 10^{-16}$	$4.27 \cdot 10^{-8}$	$4.28 \cdot 10^{-4}$	$1.15 \cdot 10^{-2}$
iterations in SPIT	1	2	5	14
$\sin\theta\{\text{range}(\overline{V}_o), N_k\}$	$6.05 \cdot 10^{-16}$	$1.20 \cdot 10^{-15}$	$1.91 \cdot 10^{-14}$	$2.44 \cdot 10^{-11}$
$\frac{\|x_k - \bar{x}_k\|_2}{\|x_k\|_2}$	$5.45 \cdot 10^{-15}$	$1.22 \cdot 10^{-15}$	$4.78 \cdot 10^{-11}$	$4.78 \cdot 10^{-11}$
$\frac{\|x_k - \bar{\bar{x}}_k\|_2}{\|x_k\|_2}$	$8.70 \cdot 10^{-2}$	$3.01 \cdot 10^{-12}$	$4.78 \cdot 10^{-11}$	$4.78 \cdot 10^{-11}$

Table 2.

Example	i	δ_i	σ_i	$\|(R_{22}^i)^{-1}\|_2$	RRQR iterations
5	1	-	$2.09 \cdot 10^{0}$	-	-
	9	$4.60 \cdot 10^{-2}$	$6.01 \cdot 10^{-2}$	$6.33 \cdot 10^{-2}$	2
	10	$2.61 \cdot 10^{-6}$	$2.61 \cdot 10^{-6}$	$3.72 \cdot 10^{-6}$	2

Table 3.

Table captions

Table 1
Comparison of the smallest singular values of A with the upper and lower bounds from RRQR. The number of iterations used in RRQR is also given.

Table 2
Results associated with the null-space computation: ω_k, the bound in (4.2), the angle θ between range(ΠW) and N_k, the number of iterations in SPIT, and the angle θ between range($\overline{V}_o$) and N_k. Also shown are the relative errors in $\bar{x}_k$ and $\bar{\bar{x}}_k$.

Table 3
Results associated with the discretization of the integral equation (7.3).