UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Low-Rank Revealing QR Factorization

Tony F. Chan
Per Christian Hansen

May 1991

CAM Report 91-08

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

LOW-RANK REVEALING QR FACTORIZATIONS

TONY F. CHAN
Department of Mathematics, UCLA
405 Hilgard Ave., Los Angeles
CA 90024, USA

PER CHRISTIAN HANSEN
UNI•C (Danish University Computing Center)
Building 305, Technical Univ. of Denmark
DK-2800 Lyngby, Denmark

Abstract

We present an algorithm for computing rank revealing QR factorizations of low-rank matrices. The algorithm produces tight upper and lower bounds for all the largest singular values, thus making it particularly useful for treating rank deficient problems by means of subset selection, truncated QR, etc. The algorithm is similar in spirit to an algorithm suggested earlier by Chan for matrices with a small nullity, and it can also be considered as an extension of ordinary column pivoting.

Keywords: Rank revealing QR factorization; Column pivoting; Numerical rank; Subset selection.

1 Introduction

Rank revealing factorizations, such as the rank revealing QR factorization introduced by Chan [6], provide the user with a convenient tool for treating rank deficient problems which is almost as reliable as the singular value decomposition, but much "cheaper" with respect to computational effort. There is a potential interest for such rank revealing factorizations in numerical linear algebra as well as in modern signal processing.

An $m \times n$ matrix A with $m \geq n$ and with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$ has numerical rank r if there is a well-defined gap between σ_r and σ_{r+1}. Given an $n \times n$ upper triangular matrix R, an $m \times n$ matrix Q with orthonormal columns, and an $n \times n$ permutation matrix Π, we say that the following QR factorization:

$$A\,\Pi = Q\,R = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \tag{1}$$

is *rank revealing* if R_{11} is $r \times r$, the smallest singular value of R_{11} satisfies $\sigma_r(R_{11}) = O(\sigma_r)$, and $\|R_{22}\|_2 = O(\sigma_{r+1})$. We note that RRQR factorizations are not unique; any pivoted QR factorization that satisfies the above requirements is an RRQR factorization. An important aspect is that RRQR algorithms also provide computable tight lower and upper bounds for the singular values of A. These bounds guarantee that one captures the correct rank of A (hence the name "rank revealing"), at a much lower cost than computing the SVD of A.

Particular interest for rank revealing QR factorizations has arisen in signal processing applications because of the need for treating large amounts of data efficiently and reliably. In these applications, the numerical rank of the involved matrix is often specifically related to the number of significant sources in a noisy signal. In some applications, the *numerical null-space* of the matrix plays a central role [4]; in other applications one is interested in *subset selection*, i.e. finding a set of linearly independent columns of the matrix, because each column represents a signal from a certain source and one wants to find the most important sources.

The original rank revealing QR algorithm RRQR was derived independently by Foster [11] and Chan [6], the main difference being that only the algorithm in [6] provides the abovementioned bounds for the singular values. A Fortran implementation is also available [17]. Improvements of this algorithm to make it more suitable for sparse matrices, and for high-speed computations via a block strategy, are suggested by Bischof & Hansen in [2, 3]. Some applications of rank revealing QR factorizations are described in [4, 7, 8, 14]. An important existence theorem for RRQR factorizations is given by Hong & Pan in [13]. Recently, a rank revealing complete orthogonal decomposition has also been proposed by Stewart [18]. Chan [5] introduced the notation of a rank revealing LU factorization for square matrices with nullity equal to one, and Lin et al. [15] extended this work to the general case.

There are many practical problems for which the numerical rank of the matrix is close to the number of columns of the matrix—we say that the numerical nullity is small. However, there are also problems for which the numerical rank is *small* compared to the number of columns. None of the currently available RRQR algorithms providing *tight bounds* for the singular values are particularly suited for low-rank problems because these algorithms (including Stewart's algorithm) assume that A has low nullity and therefore "peel off" all the small singular values starting with the smallest. Algorithms based on incremental condition estimation [1] are not suited either because they do not provide tight bounds for the singular values.

Ordinary column pivoting (OCP) in its original form [12, §5.4.1] does not provide tight bounds either, since we have $2^{1-r}|\rho_{rr}| \leq \sigma_r \leq \sqrt{n-r+1}\,|\rho_{rr}|$, where ρ_{rr} is the rth diagonal element of R [16, Thm. 6.31]. However, if OCP is supplemented with the computation—or estimation—of $\sigma_r(R_{11})$, the smallest singular value of R_{11}, then $\sigma_r(R_{11})$ is usually a much tighter lower bound for σ_r than $2^{1-r}|\rho_{rr}|$ in that we have $\sigma_r\, f_{\text{OCP}} \leq \sigma_r(R_{11}) \leq \sigma_r$, where f_{OCP} is a tightness-factor for ordinary column pivoting which depends on R, n, and r [9]. Hence, the ordinary column pivoting algorithm will in most situations indeed produce a rank revealing QR factorization of low-rank matrices with tight bounds for the singular values.

In this paper, we propose a new algorithm, called L-RRQR, with a tightness-factor which is empirically even better than f_{OCP}. Our algorithm is conceptually similar to the algorithm RRQR, but much more suited for low-rank problems. Algorithm L-

RRQR is also similar in spirit to QCP in that both algorithms construct the triangular factor one column at a time starting from the left, thus "peeling off" one large singular value of A at a time, starting with the largest. However, through the use of singular vectors computed—or estimated—during L-RRQR, our algorithm ensures the computation of tight bounds for the singular values. The similarities as well as the differences between L-RRQR and QCP are demonstrated in this paper.

Whether the large singular values or the small ones are important depends on the application and perhaps in practice a combination of algorithms RRQR and L-RRQR can be useful.

We note that our low-rank RRQR algorithm is particularly well suited for performing subset selection whenever the size of the problem makes it prohibitive to try out all combinations of the columns of A (the naive approach), as well as to compute the SVD of A (the rigorous approach, cf. [12, §12.2]).

Section 2 gives a description of the algorithm for computing the low-rank RRQR factorization. In Section 3 we access the quality of the low-rank RRQR factorization, while Section 4 discusses some important implementation issues. Finally, in Section 5 we illustrate the performance of the low-rank RRQR algorithm in comparison with ordinary column pivoting by means of numerical tests. We assume that the reader is familiar with the details of ordinary column pivoting.

2 The L-RRQR Algorithm

Let us first motivate why a rank revealing QR factorization of the form (1) is so convenient for displaying the numerical rank of A. The following Lemma follows immediately from the interlacing inequalities for singular values [12, Cor. 8.3.3]:

Lemma 1 *If the matrix R is partitioned as in Eq. (1), and if $\sigma_r(R_{11})$ denotes the smallest singular value of R_{11}, then $\sigma_r \geq \sigma_r(R_{11})$ and $\sigma_{r+1} \leq \|R_{22}\|_2$.*

We see from this lemma that if R_{11} is well-conditioned and if $\|R_{22}\|_2 \ll \|R\|_2$, then we are certain that the dimension r of R_{11} is identical to the numerical rank of A. The original algorithm RRQR, which starts out with an arbitrary QR factorization of A, achieves these goals by constructing R_{22} recursively one row at a time from the bottom. Assume that algorithm RRQR has already generated a $k \times k$ trailing submatrix with small norm. In step $k+1$, algorithm RRQR then proceeds as follows:

1. compute the smallest singular value δ_k and the corresponding right singular vector x_k of the leading $(n-k) \times (n-k)$ triangular submatrix,
2. find the permutation which permutes the largest element in absolute value of x_k to the bottom,
3. apply the same permutation to the columns of the leading $(n-k) \times (n-k)$ submatrix,
4. compute a new QR factorization of the whole matrix.

After step $k+1$, the lower and upper bounds for σ_{n-k} are given by δ_k and $\|R_{22}^{(k)}\|_2$, respectively, where $R_{22}^{(k)}$ is the $(k+1) \times (k+1)$ triangular submatrix constructed so far. The process stops when a large δ_k is found, for then we are guaranteed that algorithm RRQR has captured all the small singular values of A.

We shall now derive an algorithm L-RRQR—similar to algorithm RRQR—that computes an RRQR factorization of A without having to produce all the small elements of the large submatrix R_{22} one row at a time. The fundamental difference between RRQR and L-RRQR is that we are now working with right singular vectors associated with the *largest* singular values. The following theorem presents the basic idea underlying L-RRQR and it also highlights the close similarity between L-RRQR and OCP.

Theorem 2 *Let $A\Pi = QR$ and let ρ_{11} denotes the (1,1)-element of R. If the permutation matrix Π is chosen such that*

1. *either $|(\Pi^T x)_1| = \|\Pi^T x\|_\infty$, where x is the right singular vector associated with $\|A\|_2$,*
2. *or such that the first column $(A\Pi)_1$ of $A\Pi$ satisfies $\|(A\Pi)_1\|_2 = \max_j \|a_j\|_2$,*

then we have

$$|\rho_{11}| \geq \|A\|_2/\sqrt{n}. \tag{2}$$

Proof. Consider first case 1 and let $y = \Pi^T x$. Since x has unit norm, it follows that $|y_1| \geq 1/\sqrt{n}$. Moreover, using that $A^+ = \Pi R^+ Q^T$, we readily obtain $\mu^{-1} = \|x^T A^+\|_2 = \|x^T \Pi R^+ Q^T\|_2 = \|y^T R^+\|_2$. Since x has no component in the null space of A, the vector y has no component in the null space of R. Hence, we obtain from Theorem 8 in the Appendix that

$$\mu^{-1} = \|y^T R^+\|_2 \geq \left\| y^T \begin{pmatrix} \rho_{11}^{-1} \\ 0 \end{pmatrix} \right\|_2 = |y_1 \, \rho_{11}^{-1}| \geq |\rho_{11}^{-1}|/\sqrt{n}.$$

This is Eq. (2). For case 2, we have

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{n} \max_j \|a_j\|_2 = \sqrt{n}\,|\rho_{11}|$$

which is also (2). □

Theorem 2 guarantees that we can find a permutation matrix Π such that the (1,1)-element of R is large, i.e. of the order $\|A\|_2$. The computation of Π can be based on finding the column with maximum norm (OCP strategy) or on finding the largest element in absolute value in the singular vector x (L-RRQR strategy). Both ordinary column pivoting and our low-rank RRQR algorithm are based on this fundamental result: they recursively generate large diagonal elements of R starting from the (1,1)-element, until a submatrix R_{22} with small norm is eventually achieved (by small we mean $\|R_{22}\|_2 = O(\sigma_{r+1})$). As we shall see in the next section, the main difference between the two algorithms is that our use of the additional information contained in the singular vector x allows L-RRQR to yield empirically better tightness-factors for the singular values than OCP does.

Let us now consider algorithm L-RRQR in more detail. By means of the right singular vector x associated with the largest singular value of A we first create a large (1,1)-element in R by choosing the permutation matrix Π according to strategy 1 in Theorem 2. Once the large (1,1)-element has been created, we apply the process recursively to the $(n-1)\times(n-1)$ bottom right submatrix of the updated triangular factor. In this way, we identify the large singular values of A one at a time. The details of algorithm L-RRQR are given below for step $k+1$:

1. compute the largest singular value and the corresponding right singular vector x_k of the trailing $(n-k)\times(n-k)$ triangular submatrix,
2. find the permutation Π_k which permutes the largest element in absolute value of x_k to the top,
3. apply the same permutation Π_k to the columns of the trailing $(n-k)\times(n-k)$ submatrix,
4. compute a new QR factorization of the whole matrix.

During the algorithm, the vectors $\Pi_k^T x_k$ of decreasing length, properly padded "on top" with zeros, are used to construct a lower trapezoidal matrix $W^{(r)} \in \Re^{n\times r}$

$$W^{(r)} = \begin{pmatrix} & & 0 \\ \Pi_1^T x_1 & \cdots & \\ & & \Pi_r^T x_r \end{pmatrix}, \tag{3}$$

which provides an approximation to the row space of R. As we shall see in the next section this matrix allows us to guarantees that we compute an RRQR factorization of A. If p denotes the index of the pivot element in x_k, then we construct Π such that it produces a right cyclic shift of columns k through p of R. The resulting matrix $R\Pi$ has a downward spike in its first column; for example, if $n=6$, $k=1$, and $p=4$ then the structure of $R\Pi$ is:

$$R\Pi = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & & \times & \times & \times & \times \\ \times & & & \times & \times & \times \\ \times & & & & \times & \times \\ & & & & \times & \times \\ & & & & & \times \end{pmatrix}.$$

The spike can easily be annihilated by a series of $p-k$ left Givens rotations involving row pairs $(p-1,p),(p-2,p-1),\ldots,(k,k-1)$. The only fill-in during this process is on the diagonal, and the complexity is $O(pn)$ flops.

The "price" that one has to pay for the better tightness-factors is an increase in computational overhead, namely the computation of a singular vector of length $n-k$ for $k=0,1,\ldots,r$. If the QCP strategy is simply replaced by the L-RRQR strategy then the overhead becomes $O(mn)$ flops. However, we can reduce this overhead to $O(n^2)$ flops by computing an initial QR factorization of A and then apply algorithm L-RRQR to the such computed triangular matrix.

If A has full rank, then we could in principle work explicitly with R^{-1} using the original algorithm RRQR—or any other rank revealing algorithm—for finding all the

small singular values σ_i^{-1} of R^{-1}. Chandrasekaran and Ipsen [9] recently proposed such an algorithm with guaranteed tight bounds. We want to avoid computation of R^{-1} for several reasons: R^{-1} is not required, it is very sensitive to rounding errors for the ill-conditioned problems that we have in mind, and it does not exist if A has zero singular values which is likely for low-rank problems.

3 Properties of the Low-Rank RRQR Factorization

The purpose of this section is to analyze the quality of the low-rank RRQR factorization as obtained via the algorithm L-RRQR and to compare the quality with that of the RRQR factorization obtained via ordinary column pivoting.

The two obvious questions are: how well-conditioned is R_{11} and how small is $\|R_{22}\|_2$ after exiting algorithm L-RRQR? Both questions turn out to be related to the quality of the singular value estimates computed during the algorithm, that is, to the smallest singular value $\sigma_k(R_{11})$ of the $k \times k$ submatrix R_{11} and to the largest singular value of the $(n-k+1) \times (n-k+1)$ submatrices R_{22} for $k = 1, \ldots, r+1$. For convenience, define

$$\mu_k \equiv \|R_{22}\|_2, \qquad k = 1, \ldots, r+1. \tag{4}$$

In order to access the quality of $\sigma_k(R_{11})$ and μ_k regarded as estimates of σ_k, the kth singular value of A, we must incorporate information about the singular vectors constructed during the algorithm. We collect these vectors in a lower trapezoidal $n \times (r+1)$ matrix $W^{(r+1)}$ as in Eq. (3). We now provide two lemmas that lead us the desired accuracy estimates.

Lemma 3 *Let w_k denote the k-th column of the matrix $W^{(r+1)}$. Then the following four relations hold:*

$$\left.\begin{array}{ll} 1. & \|w_k\|_2 = 1 \\ 2. & (w_k)_j = 0 \quad \text{for} \quad j < k \\ 3. & |(w_k)_k| = \|w_k\|_\infty \geq 1/\sqrt{n-k+1} \\ 4. & \|w_k^T R^+\|_2 = \mu_k^{-1} \leq \sigma_k^{-1} \end{array}\right\} \qquad k = 1, \ldots, r+1. \tag{5}$$

Proof. The first three relations are obvious. The fourth equation is proved in Theorem 7 in the Appendix. The inequality then follows from Lemma 1. □

Lemma 4 *Partition the matrix $W^{(k)} = \begin{pmatrix} W_1^{(k)} \\ W_2^{(k)} \end{pmatrix}$ with $W_1^{(k)}$ being $k \times k$ and lower triangular. If μ_k is given by Eq. (4), then*

$$\sigma_k(R_{11}) \geq \frac{\mu_k}{\sqrt{k}\,\|(W_1^{(k)})^{-1}\|_2}, \qquad k = 1, \ldots, r+1. \tag{6}$$

Proof. It is convenient to introduce the matrix $Y_k \equiv (W^{(k)})^T R^+$. Using the fourth relation in Lemma 3 we then obtain for $k = 1, \ldots, r+1$:

$$\|Y_k\|_2 \leq \|Y_k\|_F = \|(W^{(k)})^T R^+\|_F = (\mu_1^{-2} + \cdots + \mu_k^{-2})^{\frac{1}{2}} \leq \sqrt{k}\,\mu_k^{-1}.$$

Since each w_k is a linear combination of the rows of R, it follows from Theorem 8 in the Appendix that

$$Y_k = (W^{(k)})^T R^+ = (W^{(k)})^T \begin{pmatrix} R_{11}^{-1} & -R_{11}^{-1} R_{12} R_{22}^+ \\ 0 & R_{22}^+ \end{pmatrix}$$

and therefore

$$\|Y_k\|_2 \geq \|(W_1^{(k)})^T R_{11}^{-1}\|_2 \geq \frac{\|R_{11}^{-1}\|_2}{\|(W_1^{(k)})^{-1}\|_2} = \frac{1}{\sigma_k(R_{11})\,\|(W_1^{(k)})^{-1}\|_2}.$$

These two relations yield (6). □

Theorem 5 *Let $W_1^{(k)}$ denote the first k rows of $W^{(k)}$, and let $\sigma_k(R_{11})$ denote the smallest singular value of the $k \times k$ submatrix R_{11}. With μ_k defined as in (4), we have for $k = 1, \ldots, r+1$:*

$$\frac{\sigma_k}{\sqrt{k}\,\|(W_1^{(k)})^{-1}\|_2} \leq \sigma_k(R_{11}) \leq \sigma_k \leq \mu_k \leq \sigma_k \sqrt{k}\,\|(W_1^{(k)})^{-1}\|_2. \tag{7}$$

Proof. The two innermost inequalities follow from Lemma 1 while the two outermost inequalities follow from Lemma 4 combined with Lemma 1. □

Theorem 5 is important because it shows that if the submatrix $W_1^{(k)}$ is well-conditioned then the lower and upper bounds for σ_k, namely $\sigma_k(R_{11})$ and μ_k, are guaranteed to be tight bounds. Fortunately, as long as k is not large, it follows from [7, Thm 3.2] that $W_1^{(k)}$ is indeed well-conditioned, and the following conservative upper bound holds:

$$\|(W_1^{(k)})^{-1}\|_2 < \sqrt{n}\,2^k. \tag{8}$$

In practice, this bound is very pessimistic. We stress that it is the quantity $\|(W_1^{(k)})^{-1}\|_2$, and not the bound in (8), that enters the a posteori tightness bounds (7) for the singular values. The tightness-factor for L-RRQR, mentioned in the Introduction, is obviously

$$f_{\text{LRRQR}} = (\sqrt{k}\,\|(W_1^{(k)})^{-1}\|_2)^{-1}. \tag{9}$$

The bounds $\sigma_k(R_{11}) \leq \sigma_k \leq \mu_k$ obviously also hold for QR factorizations produced by ordinary column pivoting, for which Chandrasekaran and Ipsen [9] have recently derived the tightness-factor

$$f_{\text{OCP}} = (\sqrt{n-k+1}\,\|\bar{R}_{11}^{-1}\|_2)^{-1}, \tag{10}$$

where the matrix $\bar{R}_{11}$ satisfies $R_{11} = D_{11}\bar{R}_{11}$ and D_{11} is a diagonal matrix consisting of the diagonal elements of R_{11}. As we illustrate in Section 5, the tightness-bounds obtained via OCP and Eq. (10) are usually more pessimistic than those obtained via L-RRQR and Eq. (9).

Let us now turn to the quality of the subset-selection column space provided by algorithm L-RRQR, i.e. the space spanned by the first r columns of $A\,\Pi$. It is

natural to compare this subspace with the subspace $\mathcal{R}_r(A)$ spanned by the first r left singular vectors from the SVD of A. The columns of $W^{(r)}$, on the other hand, are approximate right singular vectors of R, so the columns of $\Pi W^{(r)}$ are approximate right singular vectors of A. It is therefore also natural to compare the column space of $\Pi W^{(r)}$ with the row space $\mathcal{R}_r(A^T)$ spanned by the first r right singular vectors of A (we note in passing that $\mathcal{R}_r(A^T)$ is the "signal subspace" in signal processing). We compare subspaces by means of the subspace angle which defines a distance function between subspaces; see e.g. [12, §12.4.3] for details about subspace angles. We show in the following theorem that both approximations are good as long as the gap between σ_r and σ_{r+1} is well-defined.

Theorem 6 *Let θ denote the subspace angle between $\mathcal{R}_r(A)$ and the span of the first r columns of $A\Pi$. Also, let ϕ denote the subspace angle between $\mathcal{R}_r(A^T)$ and the column space of $\Pi W^{(r)}$. If $W_1^{(r)}$ consists of the uppermost r rows of $W^{(r)}$, then*

$$\sin\theta \leq \frac{\sigma_{r+1}}{\sigma_r(R_{11})} \leq \frac{\sigma_{r+1}}{\sigma_r}\sqrt{r}\,\|(W_1^{(r)})^{-1}\|_2 \tag{11}$$

$$\sin\phi \leq \frac{\sigma_{r+1}}{\sigma_r}\left(1+\sqrt{r}\,\|(W_1^{(r)})^{-1}\|_2\right). \tag{12}$$

Proof. The first inequality in Eq. (11) is proved in [8, Thm. 3], and the second follows from Theorem 5. The proof for Eq. (12) follows exactly the same line as the proof given in [7, Thm. 4.1] for the low-nullity RRQR factorization, and we shall not give the details here. □

Theorem 6 ensures that if there is a well-defined gap in the singular value spectrum such that $\sigma_{r+1} \ll \sigma_r$, then we are guaranteed that the subspace angles θ and ϕ are small. Therefore, the first r columns of $A\Pi$ are guaranteed to capture the column space of A, and the columns of $\Pi W^{(r)}$ are guaranteed to capture the row space of A. We are not aware of similar results for $\sin\theta$ for OCP.

As we have already mentioned in the introduction, the RRQR factorization can be used to compute a variety of quantities related to rank deficient problems, such as matrix approximations, truncated QR solutions, and total least squares solutions. These aspects are, however, beyond the scope of this paper and instead we refer to the survey in [8].

4 Implementation Issues

There are some important issues concerning the practical implementation of the algorithm L-RRQR in order to make it more efficient. We will survey these issues here and refer to the original papers for more details.

The heart of the low-rank RRQR algorithm is the computation of a good approximation to right singular vector corresponding to the largest singular value μ_k of the submatrix R_{22}. Fortunately enough, this is fairly easy to compute in practice. For example, one can apply a few steps of the power method to $R_{22}^T R_{22}$, or one can apply a few Lanczos bidiagonalization steps to R_{22}. Both methods require

$O(n^2)$ operations, and we have particularly good experience with the latter. A good starting vector is essential, and it is our experience that a good choice is the vector

$$x_0 = (\mathtt{sign}(\mathtt{diag}(R_{22})))^T$$

because it guarantees a large ratio $\|R_{22}x_0\|_2/\|x_0\|_2$.

One way of minimizing the computational overhead in step 4 of algorithm L-RRQR, involved with restoration of upper triangular form after the permutation of a column to the front of R_{22}, is to allow for a certain "sloppiness" when choosing the pivot element in the vector x in order to favor columns already residing in the front part of the matrix. This idea was introduced in [2, §5], where it was shown that the condition of $W_1^{(r)}$ does not deteriorate much if one chooses a pivot element x_p within a factor ρ of the largest element in x. The corresponding conservative upper bound for $\|(W_1^{(r)})^{-1}\|_2$, similar to (8), then becomes

$$\|(W_1^{(k)})^{-1}\|_2 < \sqrt{n}\,(2\rho)^k. \tag{13}$$

The advantage of this strategy is that it can significantly reduce the amount of work, and often pivoting is not necessary at all. This works particularly well if one starts with an initial QR factorization based on incremental condition estimation which in itself is a good approximation to an RRQR factorization—see [2] for more details. The disadvantage with introducing the factor ρ is that it deteriorates the singular value estimates accordingly.

Another means for making the algorithm more efficient is to employ some block strategy, i.e. to try to capture more than one large singular value of R_{22} at a time [3]. This is basically related to replacing the abovementioned power method or Lanczos process for computing x with a method that computes subspaces instead. It is our feeling that the Lanczos process with full reorthogonalization is a good candidate for such a method because it is reliable and yet its computational overhead is small compared to the computation of the QR factorization of A itself. The number of Lanczos steps must be set to some predefined number that balances the cost of reorthogonalization with the work of annihilating the spikes in $R_{22}\Pi$ from pivoted columns (this annihilation process is only slightly more complicated than that in the single-step procedure). Notice that such a block version of the L-RRQR algorithm must still be iterative, because there is no guarantee that we capture all the large singular values of R in one block step.

5 Numerical Results

In order to get a feeling of the reliability of algorithm L-RRQR, and to compare it with ordinary column pivoting with the bound (10), we tested both algorithms on large sets of randomly generated matrices with low rank. In this section we report the results from one of these experiments (the other experiments produced very similar results).

We generated 100 test matrices of dimensions $m = 200$ and $n = 100$ and with numerical rank $r = 15$. The singular values are distributed logarithmically in two sets between $\sigma_1 = 1$ and $\sigma_r = 10^{-5}$, and between $\sigma_{r+1} = 10^{-6}$ and $\sigma_n = 10^{-12}$.

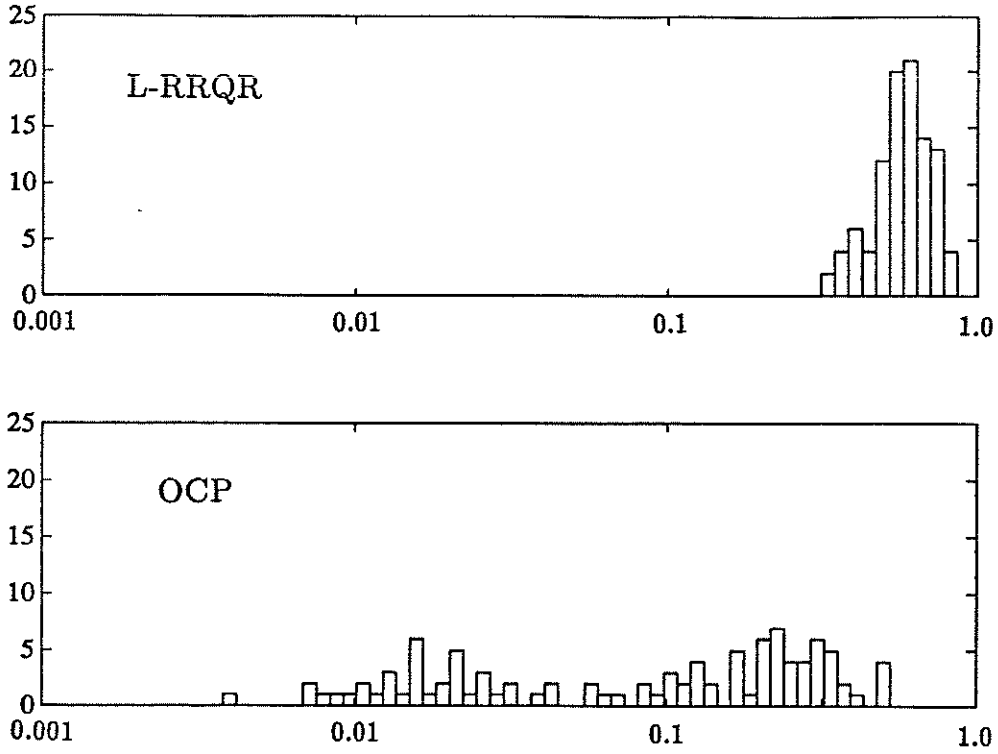

Figure 1: Histograms of the tightness factors for algorithm L-RRQR (top part) and for ordinary column pivoting (bottom part).

Each matrix was factorized by means of both algorithm L-RRQR and the ordinary column pivoting algorithm. Although the permutations from the two algorithms were rarely identical, both algorithms produced almost similar estimates $\sigma_r(R_{11})$ which were never off σ_r by more than a factor 0.7. Hence, we conclude that *both algorithms are very likely to produce reliable RRQR factorizations for low-rank matrices.*

The major difference between the two algorithms lies in the a posteori tightness factors (9) and (10). Histograms of the tightness factors for the two algorithms are shown in Fig. 1, where the top part shows the factor $f_{\text{LRRQR}} = (\sqrt{r}\,\|(W_1^{(r)})^{-1}\|_2)^{-1}$ for algorithm L-RRQR while the bottom part shows the factor $f_{\text{OCP}} = (\sqrt{n-r+1}\,\|\tilde{R}_{11}^{-1}\|_2)^{-1}$ for ordinary column pivoting. Clearly, *the tightness factor for algorithm* L-RRQR *is much better on the average.*

6 Conclusion

We have developed an algorithm L-RRQR which, just like ordinary column pivoting (OCP), seeks to generate large diagonal elements of the triangular factor. L-RRQR can be considered an extension of OCP in the sense that a more sophisticated strategy is used for finding the pivot column: L-RRQR uses information from a right singular vector while OCP simply scans the column norms of the matrix.

From our experiments we conclude that both algorithms are very likely to produce reliable RRQR factorizations for low-rank matrices. The advantage of the L-RRQR algorithm over ordinary column pivoting is that it yields better a posteori tightness bounds—at the expense of more work involved in estimating the largest singular value and corresponding singular vector in each step of the algorithm. Whether this overhead is necessary depends on how good tightness bounds are required.

Acknowledgements

Both authors are supported by a NATO Collaborative Research Grant 5-2-05/ RG900098 and by National Science Foundation under contract NSF-ASC-9003002. The first author is also supported by the Army Research Office under contract DAAL03-88-K-0085 and by the Dept. of Energy under contract DE-FG-03-87-ER-25037.

References

[1] C. H. Bischof, *Incremental Condition Estimation*, SIAM J. Matrix Anal. Appl. **11** (1990), 312–322.

[2] C. H. Bischof & P. C. Hansen, *Structure preserving and rank revealing QR factorizations*, SIAM J. Sci. Stat. Comput. **12** (1991), 1332–1350.

[3] C. H. Bischof & P. C. Hansen, *A block algorithm for computing rank-revealing QR factorizations*, Report MCS-P251-0791, Mathematics and Computer Science Division, Argonne National Laboratory, August 1991; to appear in Numerical Algorithms.

[4] C. H. Bischof & G. M. Shroff, *On updating signal subspaces*, IEEE Trans. Signal Proc. **40** (1992), 96–105.

[5] T. F. Chan, *On the existence and computation of LU-factorizations with small pivots*, Math. Comp. **42** (1984), 535–547.

[6] T. F. Chan, *Rank revealing QR factorizations*, Lin. Alg. Appl. **88/89** (1987), 67–82.

[7] T. F. Chan & P. C. Hansen, *Computing truncated SVD least squares solutions by rank revealing QR-factorizations*, SIAM J. Sci. Stat. Comput. **11** (1990), 519–530.

[8] T. F. Chan & P. C. Hansen, *Some applications of the rank revealing QR factorization*, CAM Report 90-09, Dept. of Mathematics, UCLA; SIAM J. Sci. Stat. Comput., to appear.

[9] S. Chandrasekaran & I. Ipsen, *On rank-revealing QR factorizations*, Report YALEU/DCS/RR-880, Dept. of Computer Science, Yale University, Dec. 1991.

[10] R. E. Cline, *Representations for the generalized inverse of a partitioned matrix*, J. SIAM 12 (1964), 588–600.

[11] L. V. Foster, *Rank and null space calculations using matrix decomposition without column interchanges*, Lin. Alg. Appl. 74 (1986), 47–71.

[12] G. H. Golub & C. F. Van Loan, *Matrix Computations*, 2. Ed., Johns Hopkins, 1989.

[13] Y. P. Hong & C.-T. Pan, *Rank-revealing QR factorizations and SVD*, Report, Dept. of Mathematical Sciences, Northern Illinois University. To appear in Math. Comp.

[14] S. F. Hsieh, K. J. R. Liu & K. Yao, *Applications of truncated QR methods to sinusoidal frequency estimation*, Proc. ICASSP 90 — 1990 International Conference on Acoustics, Speech, and Signal Processing.

[15] T.-M. Hwang, W.-W. Lin & E. K. Yang, *Rank revealing LU-factorizations*, Lin. Alg. Appl., to appear.

[16] C. L. Lawson & R. J. Hanson, *Solving Least Squares Problems*, Prentice Hall, 1974.

[17] L. Reichel & W. B. Gragg, *Algorithm 686: Fortran subroutines for updating the QR factorization*, ACM Trans. Math. Software 16 (1990), 369–377.

[18] G. W. Stewart, *An updating algorithm for subspace tracking*, IEEE Trans. Signal Proc., to appear.

Appendix: Pseudoinverse of a Triangular Matrix

In this appendix we give a formula for the pseudoinverse of a triangular matrix and prove two useful results. Let the triangular matrix R be partitioned such that

$$R = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \begin{matrix} k \\ n-k \end{matrix} \qquad \text{where } R_{11} \text{ has full rank.}$$

The submatrix R_{22} may be rank deficient. This covers all interesting situations. Then the pseudoinverse of R is given by

$$R^+ = \begin{pmatrix} R_{11}^{-1} & -R_{11}^{-1}R_{12}R_{22}^+ \\ 0 & R_{22}^+ \end{pmatrix} + \begin{pmatrix} -R_{11}^{-1}R_{12} \\ I \end{pmatrix} P_{22}\, K \Big(I\,,\; R_{12}R_{22}^+ \Big), \qquad (14)$$

where $P_{22} = I - R_{22}^+ R_{22}$ is the projection matrix onto the orthogonal complement of the null space of R_{22}, and the nonsingular matrix K is given by

$$K = \Big(I + (R_{11}^{-1}R_{12}P_{22})^T (R_{11}^{-1}R_{12}P_{22}) \Big)^{-1}.$$

Eq. (14) follows from a more general result for the pseudoinverse of a partitioned matrix, cf. e.g. [10, Theorem 2].

Theorem 7 *If w is a right singular vector of R_{22} associated with a nonzero singular value $\sigma = \|R_{22}\, w\|_2 > 0$, then*

$$\left\| \left(0^T\,,\, w^T\right) R^+ \right\|_2 = \sigma^{-1}. \tag{15}$$

Proof. From (14) we obtain

$$\left(0^T\,,\, w^T\right) R^+ = \left(0^T\,,\, w^T R_{22}^+\right) + w^T P_{22}\, K\left(I\,,\, R_{12} R_{22}^+\right) = \left(0^T\,,\, w^T R_{22}^+\right)$$

since $P_{22} w = 0$. This yields (15). □

Theorem 8 *For any vector $w \in \Re^n$ with no component in the null space of R, we have*

$$w^T R^+ = w^T \begin{pmatrix} R_{11}^{-1} & -R_{11}^{-1} R_{12} R_{22}^+ \\ 0 & R_{22}^+ \end{pmatrix}. \tag{16}$$

Proof. Since w has no component in the null space of R, there exists a $\xi \in \Re^n$ such that $w = R^T \xi$, and therefore we have $w^T R^+ = \xi^T R\, R^+$. Consider the product

$$R\, R^+ = R \begin{pmatrix} R_{11}^{-1} & -R_{11}^{-1} R_{12} R_{22}^+ \\ 0 & R_{22}^+ \end{pmatrix} + R \begin{pmatrix} -R_{11}^{-1} R_{12} \\ I \end{pmatrix} P_{22}\, K\left(I\,,\, R_{12} R_{22}^+\right).$$

Since

$$R \begin{pmatrix} -R_{11}^{-1} R_{12} \\ I \end{pmatrix} P_{22} = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \begin{pmatrix} -R_{11}^{-1} R_{12} \\ I \end{pmatrix} P_{22} = \begin{pmatrix} 0 \\ R_{22} P_{22} \end{pmatrix} = 0,$$

we obtain (16). □