The calculation of linear least squares problems

Åke Björck
Department of Mathematics,
Linköping University,
SE-581 83 Linköping, Sweden
E-mail: `akbjo@math.liu.se`

Dedicated to Michael A. Saunders on the occasion of his sixtieth birthday.

We first survey componentwise and normwise perturbation bounds for the standard least squares (LS) and minimum norm problems. Then some recent estimates of the optimal backward error for an alleged solution to an LS problem are presented. These results are particularly interesting when the algorithm used is not backward stable.

The QR factorization and the singular value decomposition (SVD), developed in the 1960s and early 1970s, remain the basic tools for solving both the LS and the total least squares (TLS) problems. Current algorithms based on Householder or Gram–Schmidt QR factorizations are reviewed. The use of the SVD to determine the numerical rank of a matrix, as well as for computing a sequence of regularized solutions, is then discussed. The solution of the TLS problem in terms of the SVD of the compound matrix $(b \;\; A)$ is described.

Some recent algorithmic developments are motivated by the need for the efficient implementation of the QR factorization on modern computer architectures. This includes blocked algorithms as well as newer recursive implementations. Other developments come from needs in different application areas. For example, in signal processing rank-revealing orthogonal decompositions need to be frequently updated. We review several classes of such decompositions, which can be more efficiently updated than the SVD.

Two algorithms for the orthogonal bidiagonalization of an arbitrary matrix were given by Golub and Kahan in 1965, one using Householder transformations and the other a Lanczos process. If used to transform the matrix $(b \;\; A)$ to upper bidiagonal form, this becomes a powerful tool for solving various LS and TLS problems. This bidiagonal decomposition gives a core regular subproblem for the TLS problem. When implemented by the Lanczos process it forms the kernel in the iterative method LSQR. It is also the basis of the partial least squares (PLS) method, which has become a standard tool in statistics.

We present some generalized QR factorizations which can be used to solve different generalized least squares problems. Many applications lead to LS problems where the solution is subject to constraints. This includes linear equality and inequality constraints. Quadratic constraints are used to regularize solutions to discrete ill-posed LS problems. We survey these classes of problems and discuss their solution.

As in all scientific computing, there is a trend that the size and complexity of the problems being solved is steadily growing. Large problems are often sparse or structured. Algorithms for the efficient solution of banded and block-angular LS problems are given, followed by a brief discussion of the general sparse case. Iterative methods are attractive, in particular when matrix-vector multiplication is cheap.

CONTENTS

1. Introduction

The method of least squares has been the standard procedure for the analysis of data from the beginning of 1800s. A famous example of its use is when Gauss successfully predicted the orbit of the asteroid Ceres in 1801. Two hundred years later, least squares remains a widely used computational principle in science and engineering.

In the simplest case the problem is, given $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, to find a vector $x \in \mathbb{R}^n$ such that

$$\min_x \|b - Ax\|_2, \tag{1.1}$$

where $\|\cdot\|_2$ denotes the Euclidean norm. A least squares solution x is characterized by $r \perp \mathcal{R}(A)$, where $r = b - Ax$ is the residual and $\mathcal{R}(A)$ the range space of A. The residual r is uniquely determined and the solution x is unique if and only if $\operatorname{rank}(A) = n$. If $\operatorname{rank}(A) < n$, we seek the unique least squares solution $x \perp \mathcal{N}(A)$, which is called the pseudo-inverse solution.

Under-determined systems arise from problems where there are more variables than needed to match the observed data. The model problem for this case is to find $y \in \mathbb{R}^m$ such that

$$\min \|y\|_2, \qquad A^T y = c, \tag{1.2}$$

where $c \in \mathbb{R}^n$. Here $y \in \mathbb{R}^m$, the minimum norm solution of the consistent under-determined system $A^T y = c$, is characterized by $y \perp \mathcal{N}(A^T)$. If the system $A^T y = c$ is not consistent we compute the pseudo-inverse solution.

When uncertainties are present also in the matrix A, the total least squares (TLS) model is more appropriate. The TLS problem is

$$\min \| \begin{pmatrix} E & r \end{pmatrix} \|_F, \qquad (A + E)x = b + r, \tag{1.3}$$

where $\| \cdot \|_F$ denotes the Frobenius matrix norm.

Models where the parameters x occur nonlinearly are common, but in this survey we will take the simplistic view that nonlinear problems can be solved by linearization.

From the time of Gauss until the computer age the basic computational tool for solving (1.1) was to form the normal equations $A^T Ax = A^T b$ and solve these by symmetric Gaussian elimination (which Gauss did), or later by the Cholesky factorization (Benoit 1924). This approach has the drawback that forming the matrix $A^T A$ will square the condition number of the original problem. This can lead to difficulties since least squares problems are frequently ill-conditioned.

In the 1950s algorithms based on Gram–Schmidt orthogonalization were widely used, although their numerical properties were not well understood at the time. Björck (1967*b*) analysed the modified Gram–Schmidt algorithm and showed its stability for solving linear least squares problems.

A breakthrough came with the seminal paper by Golub (1965), where it was shown how to compute a QR factorization of A using Householder transformations. A backward stable algorithm for the linear least squares problems was given. Another important development, which took place around the same time, was that of a stable algorithm for computing the singular value decomposition (SVD); see Golub and Kahan (1965) and Golub (1968), and the Algol program for computing the SVD in Golub and Reinsch (1970).

Modern numerical methods for solving least squares problems are surveyed in the two comprehensive monographs by Lawson and Hanson (1995) and Björck (1996). The latter contains a bibliography of 860 references, indicating the considerable research interest in these problems. Hansen (1998) gives an excellent survey of numerical methods for the treatment of numerically rank-deficient linear systems arising, for example, from discrete ill-posed problems. A comprehensive discussion of theory and methods for solving TLS problems is found in Van Huffel and Vandewalle (1991).

Although methods continue to evolve, variations of the QR factorization and SVD remain the basic tools for solving least squares problems. Much of the algorithmic development taking place has been motivated by needs in different application areas, *e.g.*, statistics, signal processing and control theory. For example, in signal processing data is often analysed in real time and estimates need to be updated at each time step. Other applications lead to generalized least squares problems, where the solution is subject to linear or quadratic constraints. A common trend, as in all scientific computing, is that the size and complexity of the problems being solved are steadily growing. There is also an increased need to take advantage of any structure that may exist in the model. Geodetic networks lead to huge sparse structured least squares problems, that have to be treated by sparse factorization methods. Other large-scale problems are better handled by a combination of direct and iterative methods.

The following survey of some areas of recent progress represents a highly subjective selection. Hopefully it will show that many interesting developments still take place in this field.

2. Perturbation analysis and stability

2.1. Perturbation analysis

Consider the least squares problem (1.1) with $\text{rank}(A) = n$ and solution x and residual vector $r = b - Ax$. Let the data A, b be perturbed to $A + \delta A$, $b + \delta b$ where $\text{rank}(A + \delta A) = n$. The perturbed solution by $x + \delta x$ and $r + \delta r$ satisfies the normal equations

$$(A + \delta A)^T (A + \delta A)(x + \delta x) = (A + \delta A)^T (b + \delta b).$$

Subtracting $A^T Ax = A^T b$ and solving for δx gives

$$\delta x \approx A^\dagger(\delta b - \delta A\, x) + (A^T A)^{-1} \delta A^T r, \quad A^\dagger = (A^T A)^{-1} A^T,$$

where $r = b - Ax$ is the residual and second-order terms have been neglected. For $r = 0$ this reduces to the well-known first-order perturbation bound for a square nonsingular linear system. For the residual we have $\delta r \approx (\delta b - \delta Ax) - A\delta x$ and hence

$$\delta r \approx P_{\mathcal{N}(A^T)}(\delta b - \delta A\, x) + (A^\dagger)^T \delta A^T\, r, \quad P_{\mathcal{N}(A^T)} = I - AA^\dagger.$$

Here $P_{\mathcal{N}(A^T)}$ is the orthogonal projection onto $\mathcal{N}(A^T)$. These equations yield the componentwise estimates (see Björck (1991))

$$|\delta x| \lessapprox |A^\dagger|\,(|\delta b| + |\delta A|\,|x|) + |(A^T A)^{-1}|\,|\delta A|^T\,|r|, \tag{2.1}$$

$$|\delta r| \lessapprox |P_{\mathcal{N}(A^T)}|\,(|\delta b| + |\delta A|\,|x|) + |(A^\dagger)^T|\,|\delta A|^T\,|r|, \tag{2.2}$$

where the inequalities are to be interpreted componentwise. Taking norms in (2.1) and using

$$\|A^\dagger\|_2 = 1/\sigma_n, \quad \|(A^TA)^{-1}\|_2 = 1/\sigma_n^2,$$

where σ_n is the smallest singular value of A, we obtain the approximate upper bound

$$\|\delta x\|_2 \lessapprox \frac{1}{\sigma_n}(\|\delta b\|_2 + \|\delta A\|_2\|x\|_2) + \frac{1}{\sigma_n^2}\|\delta A\|_2\|r\|_2. \tag{2.3}$$

It can be shown that for an arbitrary matrix A and vector b there are perturbations δA and δb such that this upper bound is almost attained. Note that when the residual $r \neq 0$ there is an additional term not present for consistent linear systems. The presence of this term, which will dominate if $\|r\|_2 > \sigma_n\|x\|_2$, was first pointed out by Golub and Wilkinson (1966).

Setting $\delta b = 0$ and assuming $x \neq 0$, we get for the normwise relative perturbation in x

$$\frac{\|\delta x\|_2}{\|x\|_2} \lessapprox \kappa(A)\frac{\|\delta A\|_2}{\|A\|_2}\left(1 + \frac{\|r\|_2}{\sigma_n\|x\|_2}\right), \tag{2.4}$$

where $\kappa(A) = \sigma_1/\sigma_n$ is the condition number of A.

For the minimum norm problem (1.2) with A^T of full row rank, the solution can be expressed in terms of the normal equation as $y = Az$, where $A^TAz = c$. Proceeding as before and neglecting second-order terms in the perturbation we obtain

$$\delta y \approx P_{\mathcal{N}(A^T)}\delta A\, A^\dagger y + (A^\dagger)^T(\delta c - \delta A^T y),$$

giving the componentwise approximate bound

$$|\delta y| \lessapprox |P_{\mathcal{N}(A^T)}|\,|\delta A|\,|A^\dagger|\,|y| + |(A^\dagger)^T|(|\delta c| + |\delta A|^T\,|y|). \tag{2.5}$$

Taking norms we get

$$\|\delta y\|_2 \lessapprox \frac{1}{\sigma_n}(\|\delta c\|_2 + 2\|\delta A\|_2\|y\|_2). \tag{2.6}$$

The statistical model leading to the least squares problem (1.1) is that the vector b of observations is related to the solution x by a linear relation $Ax = b + \epsilon$, where ϵ is a random error vector with zero mean and whose components are uncorrelated and have equal variance. More generally, if the covariance matrix of ϵ equals a symmetric positive definite matrix W, then the best linear unbiased estimate of x is the solution to the least squares problem $\min_x (Ax-b)^T W^{-1}(Ax-b)$, or equivalently

$$\min_x \|W^{-1/2}(b - Ax)\|_2. \tag{2.7}$$

If the errors are uncorrelated then W is a diagonal matrix and we set $D = \mathrm{diag}(d_1, \ldots, d_m) = W^{-1/2}$. Then (2.7) is a weighted least squares problem. If some components of the error vector have much smaller variance than the rest, $\kappa(DA) \gg \kappa(A) \geq 1$. The perturbation bound (2.4) then seems to indicate that the problem is ill-conditioned. This is not necessarily so and for such problems it is preferable to use the componentwise bounds (2.1)–(2.2). Special methods for weighted problems are discussed in Björck (1996, Section 4.4).

2.2. Backward error and stability

Consider an algorithm for solving the linear least squares problem (1.1). The algorithm is said to be numerically stable if, for any data A and b, there exist small perturbation matrices and vectors δA and δb, such that the computed solution $\bar{x}$ is the *exact* solution to

$$\min_x \|(A + \delta A)x - (b + \delta b)\|_2, \tag{2.8}$$

where $\|\delta A\| \leq \tau$, $\|\delta b\| \leq \tau$, with τ being a small multiple of the unit round-off u. Any computed solution $\bar{x}$ is called a stable solution if it satisfies (2.8). This does not mean that $\bar{x}$ is close to the exact solution x. If the least squares problem is ill-conditioned then a stable solution can be very different from x. For a stable solution the error $\|x - \bar{x}\|$ can be estimated using the perturbation results given in Section 2.1.

The method by Golub (1965) based on Householder QR factorization is known to be numerically stable with $\delta b = 0$ (Higham 2002, Theorem 20.3). Methods which explicitly form the normal equations are not backward stable. This is because *round-off errors that occur in forming A^TA and A^Tb are not in general equivalent to small perturbations in A and b.* Although the method of normal equations gives results of sufficient accuracy for many applications, its use can result in errors in the computed solution, which are of much larger size than for a stable method.

Many fast methods exist for solving structured least squares problems, *e.g.*, when A is a Toeplitz or Cauchy matrix. These are not in general backward stable (see Gu (1998*b*)), which is one reason why the following results are of interest.

Given an alleged solution $\tilde{x}$, a backward error is a perturbation δA, such that $\tilde{x}$ is the exact solution to the perturbed problem

$$\min_x \|(b + \delta b) - (A + \delta A)x\|_2. \tag{2.9}$$

If we could find the backward error of smallest norm, this could be used to verify numerically the stability properties of an algorithm. There is not much loss in assuming that $\delta b = 0$ in (2.10). Then the optimal backward error in the Frobenius norm is

$$\eta_F(\tilde{x}) = \min\{\|\delta A\|_F \mid \tilde{x} \text{ solves } \min_x \|b - (A + \delta A)x\|_2\}. \qquad (2.10)$$

How to find the optimal backward error for the linear least squares problem was an open problem for many years, until it was elegantly answered by Waldén, Karlsson and Sun (1995). They solved the problem by characterizing the set of all backward perturbations and by giving an optimal bound, which minimizes the Frobenius norm $\|\delta A\|_F$; see also Higham (2002, pp. 392–393). Their main result can be stated as follows.

Theorem 1. Let $\tilde{x}$ be an alleged solution and $\tilde{r} = b - A\tilde{x} \neq 0$. The optimal backward error in the Frobenius norm is

$$\eta_F(\tilde{x}) = \begin{cases} \|A^T\tilde{r}\|_2/\|\tilde{r}\|_2, & \text{if } \tilde{x} = 0, \\ \min\{\eta, \sigma_{\min}([A \;\; C])\}, & \text{otherwise}, \end{cases} \qquad (2.11)$$

where

$$\eta = \|\tilde{r}\|_2/\|\tilde{x}\|_2, \qquad C = I - (\tilde{r}\tilde{r}^T)/\|\tilde{r}\|_2^2,$$

and $\sigma_{\min}([A \;\; C])$ denotes the smallest (nonzero) singular value of the matrix $[A \;\; C] \in \mathbb{R}^{m\times(n+m)}$.

The task of computing $\eta_F(\tilde{x})$ is thus reduced to that of computing $\sigma_{\min}(\mathcal{A})$. Since this is expensive, approximations that are accurate and less costly have been derived. Karlsson and Waldén (1997) assume that a QR factorization of A is available and give lower and upper bounds for $\eta_F(\tilde{x})$ that only require $O(mn)$ operations. Gu (1998*a*) gives several approximations to $\eta_F(\tilde{x})$ that are optimal up to a factor less than 2. His bounds are formulated in terms of the singular value decomposition of A but his Corollary 2.2 can also be stated as follows.

Let $r_1 = P_{\mathcal{R}(A)}\tilde{r}$ be the orthogonal projection of $\tilde{r}$ onto the range of A. If $\|r_1\|_2 \leq \alpha\|r\|_2$ it holds that

$$\frac{\sqrt{5}-1}{2}\,\tilde{\sigma}_1 \leq \eta_F(\tilde{x}) \leq \sqrt{1+\alpha^2}\,\tilde{\sigma}_1, \qquad (2.12)$$

where

$$\tilde{\sigma}_1 = \left\|(A^TA + \eta I)^{-1/2}A^T\tilde{r}\right\|_2/\|\tilde{x}\|_2. \qquad (2.13)$$

Since $\alpha \to 0$ for small perturbations $\tilde{\sigma}_1$ is an asymptotic upper bound.

Optimal backward perturbation bounds for under-determined systems are derived in Sun and Sun (1997). The extension of backward error bounds to the case of constrained least squares problems is discussed by Cox and Higham (1999*b*).

3. Orthogonal decompositions

3.1. Algorithms using Householder reflections

The QR factorization of a matrix $A \in \mathbb{R}^{m\times n}$ is

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{3.1}$$

where $R \in \mathbb{R}^{n\times n}$ is upper triangular and $Q \in \mathbb{R}^{m\times m}$ is orthogonal. If A has linearly independent columns, *i.e.*, $\operatorname{rank}(A) = n$, then R is nonsingular. If we partition

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}, \quad Q_1 \in \mathbb{R}^{m\times n}, \quad Q_2 \in \mathbb{R}^{m\times(m-n)},$$

we obtain the compact form $A = Q_1 R$ of the QR factorization. In the full rank case Q_1 and R are uniquely determined, provided R is normalized to have positive diagonal elements. Q_1 gives an orthogonal basis for $\mathcal{R}(A)$. Q_2, which is not uniquely determined, gives an orthogonal basis for $\mathcal{N}(A^T)$.

The standard method to compute the QR factorization (3.1) is to premultiply A with a product of Householder reflections $Q^T = P_n \cdots P_2 P_1$, where

$$P_j = I - 2v_j v_j^T / \|v_j\|_2^2, \quad j = 1 : n,$$

is constructed to zero out the elements below the main diagonal in the jth column of A. Since a Householder reflection is symmetric and orthogonal,

$$Q = P_1 P_2 \cdots P_n. \tag{3.2}$$

There is usually no need to form Q explicitly, since the matrix–vector products Qy and $Q^T z$ can be efficiently formed using only the Householder vectors $v_1, v_2, \ldots, v_n$. Since v_j only has nonzero elements in positions $j : m$, these can be stored in an $m \times n$ lower trapezoidal matrix. In the dense case this is the most compact representation possible of Q and Q^T.

Given the QR factorization (3.1), the solution x to the linear least squares problem (1.1) and the corresponding residual $r = b - Ax$ is computed:

$$\begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = Q^T b, \qquad x = R^{-1} d_1, \quad r = Q\begin{pmatrix} 0 \\ d_2 \end{pmatrix} = Q_2 d_2. \tag{3.3}$$

This algorithm is backward stable (with $\delta b = 0$) both for computing the solution x and the residual $r = b - Ax$; see Higham (2002, Theorem 20.3).

Note that the residual r solves the problem of computing the orthogonal projection of b onto $\mathcal{N}(A^T)$:

$$\min_r \|b - r\|_2 \quad \text{subject to} \quad A^T r = 0.$$

In some applications we are more interested in the residual r than in the solution x. From the stability (see also the error analysis in Björck (1967*a*)) it follows that the computed residual $\bar{r}$ using (3.3) satisfies a relation

$$(A+E)^T\bar{r} = 0, \quad \|E\|_2 \le cu\|A\|_2. \tag{3.4}$$

Here and in the following c is a generic constant that grows slowly with n. This implies

$$\|A^T\bar{r}\|_2 \le cu\|\bar{r}\|_2\|A\|_2, \tag{3.5}$$

that is, the computed residual is accurately orthogonal to $\mathcal{R}(A)$. On the other hand, if $\bar{r} = \mathrm{fl}(b - Ax)$, then the best bound we can guarantee is of the form $\|A^T\bar{r}\|_2 \le cu\|b\|_2\|A\|_2$, even if x is the *exact* least squares solution, When $\|\bar{r}\|_2 \ll \|b\|_2$ this is a much weaker bound than (3.5).

The solution to the minimum norm problem (1.2) can be computed from the QR factorization (3.1) using

$$z = R^{-T}c, \qquad y = Q\begin{pmatrix} z \\ 0 \end{pmatrix} = Q_1 z. \tag{3.6}$$

The fact that this algorithm is backward stable is a relatively new result and the first proof was published in Higham (1995, Theorem 20.3).

An implementation of Householder QR factorization is given in Businger and Golub (1965) (see Wilkinson and Reinsch (1971, Contribution I/8)). A more general implementation, that also solves least squares problems with linear constraints and performs a stable form of iterative refinement of the solution, is given in Björck and Golub (1967).

3.2. Algorithms using modified Gram–Schmidt

In Gram–Schmidt orthogonalization the kth column of Q in the QR factorization is computed as a linear combination of the first k columns in A. This is equivalent to computing the compact QR factorization[1]

$$A = (a_1, a_2, \dots, a_n) = (q_1, q_2, \dots, q_n)\begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \vdots \\ & & & r_{2n} \end{pmatrix}.$$

Gram–Schmidt QR factorization can also be described as employing a sequence of elementary orthogonal projections to orthogonalize a given sequence of vectors For any nonzero vector $a \in \mathbb{R}^m$ the orthogonal projector P onto the orthogonal complement of a is given by

$$P = I_m - qq^T, \quad q = a/\|a\|_2. \tag{3.7}$$

[1] Trefethen and Bau, III (1997) aptly calls Householder QR orthogonal triangularization and Gram–Schmidt QR triangular orthogonalization.

Two versions of the Gram–Schmidt algorithm exist, usually called the Classical Gram–Schmidt (CGS) and the Modified Gram–Schmidt (MGS) algorithms. Although these only differ in the order in which the operations are performed, MGS has much better numerical stability properties.

Setting $a_j = a_j^{(1)}$, $j = 1 : n$, in MGS at the beginning of step k, $k = 1 : n$, we have computed

$$(q_1, \ldots, q_{k-1}, a_k^{(k)}, \ldots, a_n^{(k)}), \tag{3.8}$$

where $a_k^{(k)}, \ldots, a_n^{(k)}$ are orthogonal to $q_1, \ldots, q_{k-1}$. First the vector q_k is obtained by normalizing $a_k^{(k)}$. The remaining columns are then made orthogonal[2] to q_k, using orthogonal projections

$$a_j^{(k+1)} = (I - q_k q_k^T) a_j^{(k)} = a_j^{(k)} - q_k (q_k^T a_j^{(k)}), \quad j = k+1 : n.$$

Owing to rounding errors the computed $Q_1 = (q_1, q_2, \ldots, q_n)$ will not be orthogonal to working accuracy. For MGS the loss of orthogonality can be bounded in terms of the condition number of A, namely,

$$\|I - Q_1^T Q_1\|_2 \le c_1 u \kappa(A),$$

where u is the unit round-off; see Björck (1967*b*), Björck and Paige (1992).

Because of the loss of orthogonality care is needed in using the MGS factorization. Using a remarkable connection between MGS and Householder QR factorization, Björck and Paige (1992) were able to analyse MGS and rigorously prove the stability of several algorithm based on the MGS factorization. If these algorithms are used with MGS there is *no need for reorthogonalization of the q vectors* for computing least squares solutions, orthogonal projections or solving minimum norm problems. Since few textbooks describe these stable algorithms we present them again here.

Linear least squares solution by MGS

Carry out MGS on $A \in R^{m \times n}$, rank$(A) = n$, to give $Q_1 = (q_1, \ldots, q_n)$ and R, and put $b^{(1)} = b$. Compute the vector $z = (z_1, \ldots, z_n)^T$ by

```
for $k = 1 : n$
    $z_k = q_k^T b^{(k)}$;    $b^{(k+1)} = b^{(k)} - z_k q_k$;
end
$r = b^{(n+1)}$;
solve $Rx = z$;
```

[2] MGS can also be organized so that all previous projections to a_k are applied in the kth step, but this version is not suitable for column pivoting.

This algorithm for solving linear least squares problems by MGS was quite widely used even in the 1960s. A common mistake, still to be found in some textbooks, is to compute x from $Rx = d$, where $d = Q_1^T b$, which may ruin the accuracy in the solution.

Orthogonal projection by MGS
To make MGS backward stable for r it suffices to add a loop where the vector $b^{(n+1)}$ is orthogonalized against $q_n, q_{n-1}, \ldots, q_1$:

```
for $k = n, n-1, \ldots, 1$
    $z_k = q_k^T b^{(k+1)}; \quad b^{(k)} = b^{(k+1)} - z_k q_k;$
end
$r = b^{(1)};$
```

Note that the reorthogonalization has to be done *in reverse order* to prove that $\bar{r}$ is stable; see Björck and Paige (1992).

Minimum norm solution by MGS
Carry out MGS on $A^T \in R^{m\times n}$, with $\operatorname{rank}(A) = n$ to give $Q_1 = (q_1, \ldots, q_n)$ and R. Then the minimum norm solution $y = y^{(0)}$ is obtained from

```
$R^T(\zeta_1, \ldots, \zeta_n)^T = c;$
$y^{(n)} = 0;$
for $k = n, \ldots, 2, 1$
    $\omega_k = q_k^T y^{(k)}; \quad y^{(k-1)} = y^{(k)} - (\omega_k - \zeta_k) q_k;$
end
```

If the columns of Q_1 were orthogonal to working accuracy, then $\omega_k = 0$, $k = m, \ldots, 1$. Here ω compensates for the lack of orthogonality to make this algorithm backward stable!

There are a few applications where it is advantageous to compute a matrix Q_1 that is orthogonal to working precision; see Giraud, Langou, and Rozložnık (2002). The classical schemes for reorthogonalization described in Hoffman (1989) will approximately double the cost of the factorization. A new, more efficient reorthogonalization scheme, based on a low-rank update of Q_1, is described in Giraud, Gratton and Langou (2003).

3.3. Algorithms using SVD

The singular value decomposition is in general the most versatile decomposition for treating rank-deficient and severely ill-conditioned least squares problem. We write the SVD in the partitioned form

$$A = (U_1 \ U_2) \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}, \tag{3.9}$$

$$\Sigma_1 = \operatorname{diag}(\sigma_1, \dots, \sigma_r), \quad \Sigma_2 = \operatorname{diag}(\sigma_{r+1}, \dots, \sigma_n), \tag{3.10}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$.

If $\sigma_{r+1} = 0$, then A has rank r and

$$x = A^\dagger b = V_1 \Sigma_1^{-1} U_1^T b = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} v_i \tag{3.11}$$

is the pseudo-inverse solution. This is also the unique solution to the least squares problem

$$\min_{x \in \mathcal{S}} \|x\|_2, \qquad \mathcal{S} = \{x \in \mathbb{R}^n \mid \|b - Ax\|_2 = \min\}. \tag{3.12}$$

The mathematical concept of rank is not, in general, computationally useful. The *numerical rank* of a matrix $A \in \mathbb{R}^{n \times m}$ is defined in terms of its singular values. A is said to have the numerical ϵ-rank equal to $r < n$ if its singular values satisfy

$$\sigma_r > \epsilon \geq \sigma_{r+1},$$

where ϵ is a problem-dependent parameter.

If A is ill-conditioned, but there is a gap between σ_r and σ_{r+1}, then the numerical rank r is well defined. The SVD can be used to extract the linearly independent information in A, to arrive at a more well-conditioned problem. We have

$$A = U_1 \Sigma_1 V_1^T + E, \quad E = U_2 \Sigma_2 V_2^T. \tag{3.13}$$

Among the perturbation that makes $A + E$ have exact rank r, (3.13) minimizes both $\|E\|_2 = \sigma_{r+1}$, and $\|E\|_F$. The approximate least squares solution (3.11) is called the truncated SVD (TSVD) solution. It is the least squares solution restricted to the subspace $\mathcal{R}(V_1)$. The matrices U_1 and V_2 give orthogonal bases for the numerical range space and null space, respectively, of A.

The total least squares problem is best analysed in terms of the SVD

$$\begin{pmatrix} b & A \end{pmatrix} = \hat{U} \hat{\Sigma} \hat{V}^T, \quad \hat{\Sigma} = \operatorname{diag}(\hat{\sigma}_1, \dots, \hat{\sigma}_{n+1}).$$

Assume for simplicity that $\operatorname{rank}(A) = n$ and $\hat{\sigma}_{n+1} < \hat{\sigma}_n$. Then the unique perturbation of minimum norm $\| \begin{pmatrix} r & E \end{pmatrix} \|_F$ that makes $(A + E)x = b + r$ consistent is the rank one perturbation

$$\begin{pmatrix} r & E \end{pmatrix} = -\hat{\sigma}_{n+1} \hat{u}_{n+1} \hat{v}_{n+1}^T = -\begin{pmatrix} b & A \end{pmatrix} \hat{v}_{n+1} \hat{v}_{n+1}^T. \tag{3.14}$$

Multiplying (3.14) from the right with $\hat{v}_{n+1}$ gives

$$\begin{pmatrix} b & A \end{pmatrix} \hat{v}_{n+1} = -\begin{pmatrix} r & E \end{pmatrix} \hat{v}_{n+1}. \tag{3.15}$$

Writing the relation $(A+E)x = b+r$ in the form

$$\begin{pmatrix} b & A \end{pmatrix} \begin{pmatrix} 1 \\ -x \end{pmatrix} = -\begin{pmatrix} r & E \end{pmatrix} \begin{pmatrix} 1 \\ -x \end{pmatrix}$$

and comparing with (3.15), it is easily seen that the TLS solution

$$x = \gamma(\hat{v}_{2,n+1}, \dots, \hat{v}_{n+1,n+1})^T, \quad \gamma = -1/\hat{v}_{1,n+1}$$

is obtained from the right singular vector $\hat{v}_{n+1}$.

In the classical algorithm (Golub and Reinsch 1970) the SVD is computed in three steps. In the first step A is transformed into upper bidiagonal form $U_1^T AV_1 = B$ using orthogonal transformations (see Section 7). In the second step the shifted QR algorithm is applied *implicitly* to the matrix $B^T B$ giving the SVD $B = U_2 \Sigma V_2^T$. Finally, with $U = U_1 U_2$ and $V = V_1 V_2$ we obtain the SVD $A = U\Sigma V^T$.

Subroutines for computing the SVD for dense rectangular matrices are available in most mathematical software libraries; see Table 2.1 of Hansen (1998) for a list. In MATLAB the command `[U,S,V] = svd(A)` computes the SVD of a matrix of dimension 500 in only about 12 seconds on a modest SUN-server (Eldén 2004).

A survey of direct methods for computing the SVD is given in Bai, Demmel, Dongarra, Ruhe and van der Vorst (2000, Section 6.2). A divide-and-conquer method for finding the the SVD of a bidiagonal matrix is implemented in the LAPACK subroutine xGESDD. This is faster than the QR algorithm for bidiagonal matrices larger than about 25×25. Bisection and inverse iteration can be used when we only want to compute the singular values in an interval and the corresponding left and right singular vectors. (This option is suitable for the TLS problem.) Bisection methods, analysed in Fernando (1998), rely on a very accurate algorithm for counting singular values of a bidiagonal matrix.

4. Generalized least squares problems

4.1. Generalized orthogonal decompositions

The motivation for introducing different generalizations of orthogonal decompositions is basically to avoid the explicit computation of matrix products and quotients of matrices. For example, let A and B be square and nonsingular matrices and assume we need the SVD of AB^{-1} (or AB). Then the explicit calculation of AB^{-1} (or AB) may result in a loss of precision and should be avoided.

An early application of generalized QR decomposition (GQR) is described in Hammarling (1976). The systematic use of GQR as a basic conceptual and computational tool are explored by Paige (1990), who shows that these decompositions allow the solution of very general formulations of several least squares problems. Further generalizations are discussed in De Moor and Van Dooren (1992), where the QR, URV and SVD decompositions are generalized to any number of matrices.

Routines for computing a GQR decomposition of a pair of matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times p}$ are included in LAPACK; see Anderson, Bai, Bischof, Demmel, Dongarra, Du Croz, Greenbaum, Hammarling, McKenney, Ostrouchov and Sorensen (1995, Section 2.3.3). The GQR decomposition is given by

$$A = QR, \qquad B = QTZ, \tag{4.1}$$

where $Q \in \mathbb{R}^{m\times m}$ and $Z \in \mathbb{R}^{p\times p}$ are orthogonal matrices and R and T have one of the forms

$$R = \begin{pmatrix} R_{11} \\ 0 \end{pmatrix} \quad (m \geq n), \qquad R = \begin{pmatrix} R_{11} & R_{12} \end{pmatrix} \quad (m < n), \tag{4.2}$$

and

$$T = \begin{pmatrix} 0 & T_{12} \end{pmatrix} \quad (m \leq p), \qquad T = \begin{pmatrix} T_{11} \\ T_{21} \end{pmatrix} \quad (m > p). \tag{4.3}$$

If B is square and nonsingular GQR implicitly gives the QR factorization of $B^{-1}A$. There is also a similar generalized RQ factorization related to the QR factorization of AB^{-1}. These generalized decompositions and their applications are discussed in Anderssen, Bai and Dongarra (1992).

Similar generalizations for the SVD were first discussed in Van Loan (1976) and Paige and Saunders (1981). Paige (1986) gave an algorithm for computing the the quotient SVD (QSVD) of two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$; when B is square and nonsingular this is equivalent to the SVD of AB^{-1}. The computation of the SVD of AB, the product SVD (PSVD), is discussed in Heath, Laub, Paige and Ward (1986). These generalized SVDs are special cases of a more general theory developed by De Moor and Zha (1991). The GSVD algorithms in LAPACK are based on Bai and Demmel (1993), where several improvements of Paige's algorithm are given. An important role in these algorithms is played by a new accurate algorithm for the 2×2 triangular GSVD.

In Golub, Solna and Van Dooren (1995), an algorithm is developed for computing the SVD of an expression of the form

$$A = A_p^{s_p} \cdots A_2^{s_2} A_1^{s_1}, \quad s_i = \pm 1,$$

that is, a sequence of products or quotients of matrices A_i of compatible dimensions. To illustrate the idea, consider for simplicity the case when $s_i = 1$. Then it is possible to construct orthogonal matrices Q_i, $i = 0 : p$, such that the product

$$B = Q_p^T A_p Q_{p-1} \cdots Q_2^T A_2 Q_1 Q_1^T A_1 Q_0$$

is a bidiagonal matrix. The SVD of B can then be found by standard methods.

4.2. Generalized least squares problems

An important class of generalized least squares problems is related to symmetric linear systems of the form

$$\begin{pmatrix} V & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}, \tag{4.4}$$

where $A \in \mathbb{R}^{m \times n}$ $(m \geq n)$, and $V \in \mathbb{R}^{m \times m}$, is symmetric and positive semi-definite. The system matrix in (4.4) is symmetric but indefinite; it is nonsingular if and only if A has full column rank and $(V \; A)$ full row rank. Linear systems of this form (4.4) represent the condition for equilibrium of a physical system and therefore occur in numerous applications. It is also called a saddle point system and in optimization it is known as a KKT (Karush–Kuhn–Tucker) system. In applications V and A are often large and sparse matrices.

If V is positive definite and A has full column rank, then the system (4.4) is nonsingular and gives the first-order conditions for the solution of the following two optimization problems.

1. *Generalized linear least squares problem* (GLLS)

$$\min_x (Ax - b)^T V^{-1} (Ax - b) + 2c^T x. \tag{4.5}$$

If $c = 0$, the solution x gives the best linear unbiased estimate for the linear model $Ax + \epsilon = b$, where $V = \sigma^2 V$ is the covariance matrix of the error vector ϵ.

2. *Equality-constrained quadratic optimization* (ECQO)

$$\min_y \frac{1}{2} y^T V y - b^T y, \qquad A^T y = c. \tag{4.6}$$

This problem occurs as a subproblem in linearly constrained optimization. Another application, for which $c = 0$, is structural optimization (Heath, Plemmons and Ward 1984). Here A^T is called the equilibrium matrix, V the element flexibility matrix, y is the force, and x a Lagrange multiplier vector.

There are two different approaches to the solution of systems of the form (4.4). In the *range space method* the y variables are eliminated to obtain

$$A^TV^{-1}Ax = A^TV^{-1}b - c, \qquad y = V^{-1}(b - Ax). \tag{4.7}$$

For $V = I$ the first equation in (4.7) is the normal equations for the least squares problem. If V is positive definite then one way to solve these equations is to compute the Cholesky factorization $V = BB^T$ and then solve

$$\min_x \|B^{-1}(Ax - b)\|_2 \tag{4.8}$$

using the QR factorization of $B^{-1}A$. However, a more stable approach is to use a GQR factorization of the matrix pair A, B.

In the *null space method* the solution y to (4.7) is split as

$$y = y_1 + y_2, \qquad y_1 \in \mathcal{R}(A), \qquad y_2 \in \mathcal{N}(A^T). \tag{4.9}$$

Let y_1 be the minimum norm solution of $A^Ty = c$. This can be computed using the QR factorization of A. If we set $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$ then

$$y_1 = Q_1z_1, \qquad z_1 = R^{-T}c.$$

Next y_2 is obtained by solving the reduced system

$$Q_2^TVQ_2z_2 = Q_2^T(b - Vy_1), \qquad y_2 = Q_2z_2. \tag{4.10}$$

Finally, form

$$y = Q_1z_1 + Q_2z_2 \quad \text{and} \quad x = R^{-1}Q_1^T(b - Vy).$$

In the special case that $V = I$, the numerical stability of methods which use Q and R, or only R, in the QR factorization of A are studied in Björck and Paige (1994). Backward stability was proved for several methods.

Recently perturbation analyses and condition numbers for problem (4.4) have been given in Arioli (2000) and Gulliksson and Wedin (2000). Arioli (2000) also gives a round-off error analysis of a null space method for solving (4.4) and applies this to developing methods for solving a problem arising from a mixed finite element discretization of a magnetostatic problem.

5. Blocked algorithms

5.1. Partitioned algorithms

The impact of the architecture of modern computers on algorithms of numerical linear algebra is surveyed in depth in Dongarra, Duff, Sorensen and van der Vorst (1998). One conclusion is that to obtain near-peak performance for large dense matrix computations on current computing architectures requires code that is dominated by level 3 Basic Linear Algebra Subroutines (BLAS 3). These kernels perform various types of matrix–matrix multiplication and involve less data movement per floating point computation; see Dongarra, Du Croz, Duff and Hammarling (1990). The subroutines in LAPACK, including those for QR factorization, are therefore organized in partitioned or blocked form, in which the operations have been reordered and grouped into matrix operations. These partitioned algorithms are as stable as their point counterparts. This is not the case for all block algorithms; see Higham (1997) for the distinction between block and partitioned algorithms.

For the QR factorization $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$ is partitioned as

$$A = (A_1,\ A_2), \qquad A_1 \in \mathbb{R}^{m\times nb}, \tag{5.1}$$

where nb is a suitable block size and the QR factorization

$$Q_1^T A_1 = \begin{pmatrix} R_1 \\ 0 \end{pmatrix}, \qquad Q_1 = P_1 P_2 \cdots P_{nb}, \tag{5.2}$$

is computed, where $P_i = I - u_i u_i^T$ are Householder reflections. Then the remaining columns A_2 are are updated

$$Q_1^T A_2 = Q_1^T \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} = \begin{pmatrix} R_{12} \\ \tilde{A}_{22} \end{pmatrix}. \tag{5.3}$$

In the next step we partition $\tilde{A}_{22} = (B_1, B_2)$, and compute the QR factorization of $B_1 \in \mathbb{R}^{(m-r)\times r}$. Then B_2 is updated as above, and we continue in this way until the columns in A are exhausted.

A major part of the computation is spent in the updating step (5.3). As written this step cannot use BLAS-3, which slows down the execution. To achieve better performance it is essential to speed this part up. The solution is to aggregate the Householder transformations so that their application can be expressed as matrix operations; see Schreiber and Van Loan (1989). For use in the next subsection, we show a slightly more general result due to Elmroth and Gustavson (2000).

Assume that $r = r_1 + r_2$, and

$$Q_1 = P_1 \cdots P_{r_1} = I - Y_1 T_1 Y_1^T, \qquad Q_2 = P_{r_1+1} \cdots P_r = I - Y_2 T_2 Y_2^T,$$

where $T_1, T_2 \in \mathbb{R}^{r\times r}$ are upper triangular. Then

$$Q = Q_1 Q_2 = (I - Y_1 T_1 Y_1^T)(I - Y_2 T_2 Y_2^T) = (I - YTY^T), \tag{5.4}$$

where

$$\hat{Y} = (Y_1,\ Y_2), \qquad \hat{T} = \begin{pmatrix} T_1 & -(T_1 Y_1^T)(Y_2 T_2) \\ 0 & T_2 \end{pmatrix}. \tag{5.5}$$

Note that Y is formed by concatenation, but computing the off-diagonal block in T requires extra operations.

For the partitioned algorithm we use the special case when $r_2 = 1$ to aggregate the Householder transformations for each processed block. Starting with $Q_1 = I - \tau_1 u_1 u_1^T$, we set $Y = u_1$, $T = \tau_1$ and update

$$Y := (Y,\ u_{k+1}), \qquad T := \begin{pmatrix} T & -\tau_k T Y^T u_k \\ 0 & \tau_k \end{pmatrix}, \quad k = 2 : nb. \tag{5.6}$$

Note that Y will have a trapezoidal form and thus the matrices Y and R can overwrite the matrix A. With the representation $Q = (I - YTY^T)$ the updating of A_2 becomes

$$B = Q_1^T A = (I - YT^TY^T)A_2 = A_2 - YT^TY^TA_2,$$

which now involves only matrix operations. An analogous partitioned version of MGS is discussed in Björck (1994).

This partitioned algorithm requires more storage and operations than the point algorithm, namely those needed to produce and store the T matrices. However, for large matrices this is more than offset by the increased rate of execution.

5.2. Recursive algorithms

As shown by Elmroth, Gustavson, Jonsson and Kågström (2004) recursive algorithms can be developed into highly efficient algorithms for high performance computers and are an alternative to the partitioned algorithms currently used by LAPACK. The reason for this is that recursion leads to automatic variable blocking that dynamically adjusts to an arbitrary number of levels of memory hierarchy.

The recursive QR factorization

$$A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

starts with a QR factorization of the first $\lfloor n/2 \rfloor$ columns of A and updating of the remaining part of the matrix

$$Q_1^T A_1 = \begin{pmatrix} R_{11} \\ 0 \end{pmatrix}, \qquad Q_1^T A_2 = Q_1^T \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} = \begin{pmatrix} R_{12} \\ \tilde{A}_{22} \end{pmatrix}.$$

Next $\tilde{A}_{22}$ is recursively QR-decomposed, giving Q_2, R_{22}, and $Q = Q_1 Q_2$.

As an illustration we give a simple implementation in Matlab, which is convenient to use since it allows for the definition of recursive functions.

```
function [Y,T,R] = recqr(A);
%
% RECQR computes the QR factorization of the m by n matrix A,
% (m >= n). Output is the n by n triangular factor R, and
% Q = (I - YTY')  represented in aggregated form, where Y is
% m by n and unit lower trapezoidal, and T is n by n upper
% triangular It uses [u,tau,sigma] = house(a) to compute
% a Householder transformation P = I - tau uu', such that
% Pa = sigma e1, sigma = -sign(a_1)norm(a).
    [m,n] = size(A);
    if n == 1
    [Y,T,R] = house(A);
    else
      n1 = floor(n/2);
      n2 = n - n1; j = n1+1;
      [Y1,T1,R1]= recqr(A(1:m,1:n1));
      B = A(1:m,j:n) - (Y1*T1')*(Y1'*A(1:m,j:n));
      [Y2,T2,R2] = recqr(B(j:m,1:n2));
      R = [R1, B(1:n1,1:n2); zeros(n-n1,n1), R2];
      Y2 = [zeros(n1,n2); Y2];
      Y = [Y1, Y2];
      T = [T1, -T1*(Y1'*Y2)*T2; zeros(n2,n1), T2];
    end
%
```

The above algorithm is just a prototype and needs to be improved and tuned in several ways. A serious defect is the overhead in storage and operations caused by the T matrices. In the partitioned algorithm n/nb T-matrices of size $nb \times nb$ are formed and stored, giving a storage overhead of $\frac{1}{2}n \cdot nb$. In the recursive QR algorithm in the end a T-matrix of size $n \times n$ is formed and stored, leading to a much too large storage and operation overhead.

Elmroth and Gustavson (2000) develop and analyse recursive algorithms for the QR factorization. They find that the best option is a hybrid between the partitioned and the recursive algorithm, where the recursive QR algorithm is used to factorize the blocks in the partitioned algorithm. In Elmroth and Gustavson (2001) these hybrid QR algorithms are used to implement recursive algorithms for computing least squares solutions to overdetermined linear systems and minimum norm solutions to under-determined linear systems. These implementations are shown to be significantly faster – usually 50–100% and sometimes much more – than the corresponding current LAPACK algorithms based on the partitioned approach.

6. Rank-revealing decompositions

6.1. Column pivoting

Golub (1965) remarks that the accuracy in the QR factorization is slightly improved if the following column pivoting strategy is used. Assume that after k steps we have computed the partial QR factorization

$$A^{(k)} = (P_k \cdots P_1)A(\Pi_1 \cdots \Pi_k) = \begin{pmatrix} R_{11}^{(k)} & R_{12}^{(k)} \\ 0 & \tilde{A}^{(k)} \end{pmatrix}, \tag{6.1}$$

where $\Pi_1, \dots, \Pi_k$ are permutation matrices performing the column interchanges. The remaining steps will only affect the submatrix $\tilde{A}^{(k)}$. The pivot column in the step $k+1$ is chosen as a column of largest norm in the submatrix

$$\tilde{A}^{(k)} = (\tilde{a}_{k+1}^{(k)}, \dots, \tilde{a}_n^{(k)}) \in \mathbb{R}^{(m-k)\times(n-k)},$$

i.e., Π_{k+1} interchanges columns p and $k+1$, where p is the smallest index such that

$$s_p^{(k)} \geq s_j^{(k)}, \qquad s_j^{(k)} = \|\tilde{a}_j^{(k)}\|_2, \quad j = k+1:n. \tag{6.2}$$

If $s_p^{(k)} = 0$ then the algorithm terminates with $\tilde{A}^{(k)} = 0$ in (6.1), which implies that $\operatorname{rank}(A) = k$. This pivoting strategy can be viewed as choosing a remaining column of largest distance to the subspace spanned by the previously chosen columns and is equivalent to maximizing the diagonal element $r_{k+1,k+1}$.

Golub (1965) also notes that 'the strategy above is most appropriate when one has a sequence of vectors $b_1, b_2, \dots, b_p$ for which one desires a least squares estimate. In many problems there is one vector b and one wishes to express it in as few columns of A as possible.' For this case one should at each stage choose the column of $A^{(k)}$ that will maximally reduce the sum of squares of the residuals after the kth stage. If

$$(P_k \cdots P_1)b = \begin{pmatrix} c_1^{(k)} \\ c_2^{(k)} \end{pmatrix},$$

this is equivalent to choosing a pivot which maximizes

$$t_j^{(k)} = |(c_2^{(k)})^T \tilde{a}_j^{(k)}| / \|\tilde{a}_j^{(k)}\|_2, \quad k < j \leq n.$$

The partitioned algorithm as reviewed in Section 5 cannot easily be implemented for the pivoted QR factorization. This is because in order to choose a pivot column all remaining columns need first to be updated. Therefore it is not possible to accumulate several Householder transformations and perform the update simultaneously. Quintana-Ortí, Sun and Bischof (1998) show how the pivoted QR algorithm can be implemented so that half of the work is performed in BLAS-3 kernels.

6.2. Rank-revealing QR decompositions

Rank-deficient problems are common, *e.g.*, in statistics, where the term collinearity is used. Although the SVD is generally the most reliable method for computing the numerical rank of a matrix it has the disadvantage of a high computational cost. Alternative decompositions based on QR factorization with column pivoting were first proposed in Faddeev, Kublanovskaya and Faddeeva (1968) and Hanson and Lawson (1969).

Let $A \in \mathbb{R}^{m\times n}$ be a matrix with singular values $\sigma_1 \geq \sigma_1 \geq \cdots \geq \sigma_n \geq 0$. By a rank-revealing QR (RRQR) decomposition of A we mean a decomposition of the form

$$A\Pi = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \tag{6.3}$$

where Π is a permutation matrix, $R_{11} \in \mathbb{R}^{k\times k}$ upper triangular, and

$$\sigma_k(R_{11}) \geq \frac{1}{c_1}\sigma_k, \qquad \|R_{22}\|_2 \leq c_2\sigma_{k+1}. \tag{6.4}$$

for some not too large constants c_1 and c_2. If σ_{k+1} is small and $\sigma_k \gg \sigma_{k+1}$, then the factorization (6.4) will reveal that the numerical rank of A is k.

In (6.3) the matrix Q_1 gives an orthogonal basis for the numerical range of A and

$$W = \begin{pmatrix} R_{11}^{-1}R_{12} \\ -I \end{pmatrix} \in \mathbb{R}^{n\times(n-k)} \tag{6.5}$$

gives a basis for the numerical null space of AP. If a more accurate basis than (6.5) is desired this can be computed using a few inverse iterations, as suggested in Chan and Hansen (1990).

Hong and Pan (1992) prove the existence of a decomposition (6.3)–(6.4) with

$$c = \sqrt{k(n-k) + \min(k, n-k)}, \quad \forall k, \quad 0 < k < n.$$

Thus, whenever there is a well-determined gap in the singular value spectrum, $\sigma_r \gg \sigma_{r+1}$, there exists for $k = r$ an RRQR decomposition that reveals the numerical rank of A.

To find a permutation Π such that the rank of A is revealed is not always simple. Note that an exhaustive search has combinatorial complexity! It is still an open question if an algorithm of polynomial complexity exists for finding an optimal permutation. Fortunately there are algorithms that in practice work almost always.

From the interlacing properties of singular values (Golub and Van Loan 1996, Corollary 8.6.3) it follows by induction that, for any decomposition of the form (6.3), we have the inequalities

$$\sigma_{\min}(R_{11}) \le \sigma_k(A), \quad \sigma_{\max}(R_{22}) \ge \sigma_{k+1}(A).$$

Hence, to achieve a rank-revealing QR decomposition we want to find a permutation P that aims to solve the two problems

$$\text{(i)} \quad \max_{\Pi} \sigma_{\min}(R_{11}); \qquad \text{(ii)} \quad \min_{\Pi} \sigma_{\max}(R_{22}).$$

These two problems are dual in a certain sense.

Problem (i) is equivalent to the subset selection problem of determining the $k < n$ most linearly independent columns of A. An SVD-based algorithm for solving this problem was given in Golub, Klema and Stewart (1976) and an RRQR algorithm in Chan and Hansen (1992). Although the methods will not in general compute equivalent solutions, the subspaces spanned by the two sets of selected columns can be shown to be almost identical whenever the ratio σ_{k+1}/σ_k is small.

The column pivoting strategy used in Golub (1965) chooses as the next pivot column the one having maximum distance from the subspace spanned by the already chosen columns. Hence this is a greedy algorithm that addresses problem (i). While this strategy can fail on certain matrices (see Golub and Van Loan (1996, Section 5.5.7)), it is widely used due to its simplicity and practical reliability. In several other RRQR algorithms this pivoting strategy is used in a preprocessing stage. In the second stage a new permutation matrix is determined so that the rank-revealing property is improved.

Several algorithms have been suggested, which address problem (ii). These are based on the following property.

Lemma 1. (Chan and Hansen (1990)) Given any column permutation P and $V \in \mathbb{R}^{n\times p}$, the QR factorization of AP yields an R_{22} such that

$$\|R_{22}\|_2 \le \|AV\|_2 \|W_2^{-1}\|_2, \qquad W = P^T V = \begin{pmatrix} W_1 \\ W_2 \end{pmatrix}. \tag{6.6}$$

This means that if the columns of V lie approximately in the numerical null space of A and we can find a permutation such that $\|W_2^{-1}\|_2$ is not large, then the bottom $p\times p$ block of R in the QR factorization will be small. Ideally, for $p = 1$, we take $v \approx v_n$, the right singular vector corresponding to σ_n. The permutation P is chosen so that the component of largest absolute value is moved to the end.

Early algorithms based on the above lemma include Golub *et al.* (1976) and Chan (1987). Algorithms that satisfy (6.3)–(6.4) with c_1, c_2 equal to low-order polynomials in n and m are developed in Pan and Tang (1999) and Chandrasekaran and Ipsen (1994). These consist of two main stages: an initial pivoted QR factorization of A followed by a rank-revealing stage in which the triangular factor is modified. Chandrasekaran and Ipsen (1994) provide a common framework for these algorithms.

In Bischof and Quintana-Ortí (1998*b*) more efficiently implementable variants of RRQR algorithm for triangular matrices are developed. In the first stage a pivoted QR factorization is computed where the pivoting is restricted to a pivot window. In the post-processing stage either Algorithm 3 in Pan and Tang (1999) or Hybrid III in Chandrasekaran and Ipsen (1994) is used. These hybrid algorithms are nearly as fast as current partitioned QR algorithms without pivoting; see Bischof and Quintana-Ortí (1998*a*)

6.3. The URV and ULV decompositions

In signal processing problems the data analysed often arrives in real time and it is necessary to update matrix decompositions at each time step. For such applications the SVD has the disadvantage that it cannot in general be updated in less than $O(n^3)$ operations, when rows and columns are added or deleted to A. In special cases simplified updating schemes may be viable, such as the fast SVD updating algorithm in Moonen, Van Dooren and Vandewalle (1992), which is a combination of a QR updating followed by a Jacobi-type SVD method.

Although the RRQR decomposition can be updated, it is less suitable in applications where a basis for the approximate null space of A is needed, since the matrix W in (6.5) cannot easily be updated. For this reason Stewart (1991) introduced the URV rank-revealing decomposition

$$A = URV^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}, \tag{6.7}$$

where U and V are orthogonal matrices, $R_{11} \in \mathbb{R}^{k\times k}$, and

$$\sigma_k(R_{11}) \geq \frac{1}{c}\sigma_k, \qquad \left(\|R_{12}\|_F^2 + \|R_{22}\|_F^2\right)^{1/2} \leq c\sigma_{k+1}. \tag{6.8}$$

Note that here both submatrices R_{12} and R_{22} have small elements.

From (6.7) we have

$$\|AV_2\|_2 = \left\| \begin{pmatrix} R_{12} \\ R_{22} \end{pmatrix} \right\|_F \leq c\sigma_{k+1},$$

and hence the orthogonal matrix V_2 can be taken as an approximation to the numerical null space $\mathcal{N}_k$.

Algorithms for computing a URV decomposition start with an initial QR decomposition, followed by a rank-revealing stage in which singular vectors corresponding to the smallest singular values of R are estimated. Assume that w is a unit vector such that $\|Rw\| = \sigma_n$. Let P and Q be orthogonal matrices such that $Q^T w = e_n$ and $P^T RQ = \hat{R}$, where $\hat{R}$ is upper triangular. Then

$$\|\hat{R}e_n\| = \|P^T RQQ^T w\| = \|P^T Rw\| = \sigma_n,$$

which shows that the entire last column in $\hat{R}$ is small. Given w the matrices P and Q can be constructed as a sequence of Givens rotations (see Stewart (1992), where algorithms are also given for updating a URV decomposition when a new row is appended).

As for the RRQR decompositions, the URV decomposition yields approximations to the singular values. Mathias and Stewart (1993) derive the following bounds:

$$f\sigma_i \le \sigma_i(R_{11}) \le \sigma_i, \quad i = 1 : r,$$

and

$$\sigma_i \le \sigma_{i-k}(R_{22}) \le \sigma_i/f, \quad i = r+1 : n,$$

where

$$f = \left(1 - \frac{\|R_{12}\|_2^2}{\sigma_{\min}(R_{11}^2 - \|R_{22}\|_2^2}\right)^{1/2}.$$

Hence the smaller the norm of the off-diagonal block R_{12}, the better the bounds will be. Similar bounds can be given for the angle between the range of V_2 and the right singular subspace corresponding to the smallest $n - r$ singular values of A.

Stewart (1993) gives an alternative decomposition that is more satisfactory for applications where an accurate approximate null space is needed, as in subspace tracking. This is the rank-revealing ULV decomposition

$$A = U \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} V^T, \tag{6.9}$$

where the middle matrix has lower triangular form. For this decomposition

$$\|AV_2\|_2 = \|L_{22}\|_F, \qquad V = (V_1,\ V_2),$$

and hence the size of $\|L_{21}\|$ does not adversely affect the null space approximation. On the other hand the URV decomposition usually gives a superior approximation for the numerical range space and the updating algorithm for URV is much simpler.

We finally mention that rank-revealing QR decompositions can be effectively computed only if the numerical rank r is either high, $r \approx n$ or low, $r \ll n$. The low rank case is discussed in Chan and Hansen (1994). MATLAB templates for rank-revealing UTV decompositions are described in Fierro, Hansen and Hansen (1999).

The ULV and URV decompositions are rank-revealing, but do not attempt to give good approximations to the singular values. The pivoted QLP decomposition, also introduced by Stewart (1999), yields for the extra cost of one more QR decomposition quite accurate approximations to the singular values of A. The QLP decomposition can be considered as the first step in a rapidly converging iterative algorithm for computing the SVD of A, which is analysed in Huckaby and Chan (2003).

The QLP algorithm starts by computing the pivoted QR factorization

$$Q^T A\Pi = \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R \in \mathbb{R}^{n\times n}. \tag{6.10}$$

In the second step the *upper* triangular matrix R is transformed into a *lower* triangular matrix L, using postmultiplication by a product P of Householder transformations,

$$RP = L, \quad L \in \mathbb{R}^{n\times n}. \tag{6.11}$$

No pivoting is used in this step. (Transposing (6.11) shows that this LQ factorization of R is equivalent to a QR factorization of the lower triangular matrix R^T.) Combining these two factorizations (6.10) and (6.11) we obtain

$$A\Pi = Q \begin{pmatrix} L \\ 0 \end{pmatrix} P^T. \tag{6.12}$$

To compute the QLP decomposition requires roughly $mn^2 - n^3/3$ flops for the decomposition (6.10) and $2n^3/3$ flops for the decomposition (6.11).

Suppose that after k steps of the pivoted Householder QR algorithm (6.10) we have computed the partial QR factorization

$$Q_k A\Pi_k = A^{(k+1)} = \begin{pmatrix} R_{11} & R_{12} \\ 0 & \tilde{A}_{22} \end{pmatrix},$$

where $\begin{pmatrix} R_{11} & R_{12} \end{pmatrix}$ are the first k rows of R in the QR factorization of A. By postmultiplying with k Householder transformations we obtain

$$\begin{pmatrix} R_{11} & R_{12} \end{pmatrix} P_k = \begin{pmatrix} L_{11} & 0 \end{pmatrix},$$

where L_{11} is the first k rows of L in the QLP decomposition. This observation shows that the two factorization can be interleaved, *i.e.*, in the kth step we first compute the kth row of R and then the kth row of L. To determine the first k diagonal elements of L, which give the QLP approximations to the first k singular values of A, it is only necessary to perform k steps in each of the two factorization. This is advantageous when the numerical rank is much less than n.

Despite the simplicity of the QLP decomposition the diagonal elements of L usually give remarkably good approximations to all the singular values

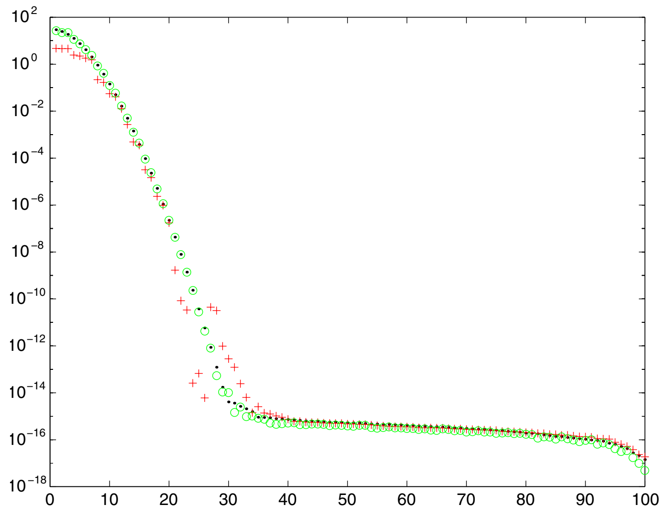

Figure 6.1. Diagonal elements of L (circles) in the QLP and in R (plus) in pivoted QR compared with singular values (points) of the matrix K.

of A. In particular a good estimate of $\sigma_1 = \|A\|_2$ can be obtained in $O(n^2)$ operations from the first row $\begin{pmatrix} r_{11} & r_{12} \end{pmatrix}$ of R setting

$$\sigma_1 \approx l_{11} = (r_{11}^2 + \|r_{12}\|_2^2)^{1/2}.$$

As an illustration we consider the integral equation of the first kind

$$\int_{-1}^{1} k(s,t) f(s)\,\mathrm{d}s = g(t), \quad k(s,t) = \mathrm{e}^{-(s-t)^2},$$

on $-1 \le t \le 1$. If this equation is discretized using a uniform mesh on $[-1,1]$ and the trapezoidal rule, the resulting linear system $Kf = g$, $K \in \mathbb{R}^{n\times n}$, is very ill-conditioned. In Figure 6.1 the singular values σ_k of the matrix K_n, $n = 100$, are displayed together with (absolute values of) the diagonal elements of R and L in the QLP decomposition. The diagonal elements of L are seen to track both large and small singular values much more accurately than those of R.

7. Bidiagonal reduction

Any matrix $A \in \mathbb{R}^{m\times n}$ can be decomposed as

$$A = UBV^T, \tag{7.1}$$

where B is an upper (or lower) bidiagonal matrix and U and V are orthogonal matrices. This important decomposition first appeared in Golub and Kahan (1965). It is usually the first step in computing the SVD of A, but is also a powerful tool in itself for solving various least squares problems. It is the core decomposition used in the iterative method LSQR (Paige and Saunders 1982*b*) for solving least squares problems. Recently Paige and Strakoš (2002) (see also Paige (2002)) have shown that the reduction to upper bidiagonal form of $\begin{pmatrix} b & A \end{pmatrix}$ provides an elegant way to extract a core problem, both for the linear least squares problem (1.1) and the total least squares problem (1.3). Because of its importance we review this decomposition in some detail below.

Golub and Kahan (1965) gave two quite different algorithms for computing the decomposition (7.1). In the first algorithm U and V are formed as products of two sequences of Householder transformations. In the second algorithm the successive columns in U and V are generated by a Lanczos process. Once the first column $u_1 = Ue_1$ (or $v_1 = Ve_1$) is fixed, the decomposition (7.1) is uniquely determined in the nondegenerate case. Therefore the two algorithms will, using exact arithmetic, produce the same bidiagonal decomposition. However, the Lanczos method is less stable numerically and is mainly of interest when A is a large and sparse matrix.

7.1. Bidiagonalization using Householder transformations

With no loss of generality we assume that $m \geq n$. Following the first algorithm in Golub and Kahan (1965), for $k = 1, 2, \ldots,$ we alternately multiply $(b,\ A)$ from the left and right with Householder transformations Q_k and P_k, respectively. Here Q_k is chosen to zero the last $m-k$ elements in the kth column of $(b,\ A)$ and P_k is chosen to zero the last $n-k$ elements in the kth row of A. The final result is the decomposition

$$U^T \begin{pmatrix} b & AV \end{pmatrix} = \begin{pmatrix} \beta_1 e_1 & B \\ 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_n & \alpha_n \\ & & & \beta_{n+1} \end{pmatrix}, \tag{7.2}$$

where e_1 is the first unit vector, and

$$U = Q_1 Q_2 \cdots Q_{n+1} \in \mathbb{R}^{m\times(n+1)}, \qquad V = P_1 P_2 \cdots P_{n-1} \in \mathbb{R}^{n\times n}. \tag{7.3}$$

Note that the first column in U equals $u_1 = Ue_1 = b/\beta_1$ and that $U^T AV = B \in \mathbb{R}^{(n+1)\times n}$ is a *lower* bidiagonal matrix.

Setting $y = V^T x$ and using the invariance of the Euclidean norm it follows that

$$\begin{aligned} \|b - Ax\|_2 &= \left\| \begin{pmatrix} b & A \end{pmatrix} \begin{pmatrix} -1 \\ x \end{pmatrix} \right\|_2 = \left\| U^T \begin{pmatrix} b & AV \end{pmatrix} \begin{pmatrix} -1 \\ V^T x \end{pmatrix} \right\|_2 \\ &= \|\beta_1 e_1 - By\|_2. \end{aligned} \tag{7.4}$$

Hence, if y solves the bidiagonal least squares problem

$$\min_y \|By - \beta_1 e_1\|_2, \tag{7.5}$$

then $x = Vy$ solves $\min_x \|Ax - b\|_2$.

After $k < n$ steps of the bidiagonal reduction we have computed an upper bidiagonal matrix B_k and orthogonal matrices $U_k = Q_1 \cdots Q_k$, and $V_k = P_1 \cdots P_k$, such that $U_k^T A V_k = B_k$. From the construction of the Householder matrices P_j it follows that

$$U \begin{pmatrix} I_k \\ 0 \end{pmatrix} = Q_1 \cdots Q_k Q_{k+1} \cdots Q_{n+1} \begin{pmatrix} I_k \\ 0 \end{pmatrix} = Q_1 \cdots Q_k \begin{pmatrix} I_k \\ 0 \end{pmatrix} = U_k. \tag{7.6}$$

This shows that the first k columns in U_k and the final matrix U are equal. Similarly the first k columns in V_k equals those in V.

7.2. The core subproblem

Assume first that α_j, $\beta_{j+1} \neq 0$, $j = 1 : k-1$, for some $1 \leq k \leq n$, but $\alpha_k = 0$. Then after k steps we have obtained a decomposition of the form

$$U_k^T A V_k = \begin{pmatrix} B_k & 0 \\ 0 & A_k \end{pmatrix},$$

where $B_k \in \mathbb{R}^{k \times (k-1)}$ is a leading submatrix of B and $A_k \in \mathbb{R}^{(m-k)\times(n-k+1)}$. The resulting transformed least squares problem

$$\min_y \left\| \begin{pmatrix} B_k & 0 \\ 0 & A_k \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} \right\|_2, \quad y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \tag{7.7}$$

is separable and decomposes into the two independent subproblems

$$\min_{y_1} \|B_k y_1 - \beta_1 e_1\|_2 \quad \text{and} \quad \min_{y_2} \|A_k y_2\|_2. \tag{7.8}$$

The first subproblem is similar to (7.5) and since B_k has full column rank the solution y_1 is unique. We call this the *core subproblem*. Clearly the minimum norm solution $x = Vy$ is obtained by taking $y_2 = 0$.

Assume next that also $\alpha_k \neq 0$, but $\beta_{k+1} = 0$. Then the reduced matrix again has the separable form (7.7), where now $B_k \in \mathbb{R}^{k\times k}$ and $A_k \in \mathbb{R}^{(m-k)\times(n\times k)}$. The core subproblem then simplifies to $B_k y_1 = \beta_1 e_1$, with B_k square, nonsingular and lower triangular. The unique solution y_1 is obtained simply by forward substitution. Taking $y_2 = 0$, the corresponding residual $b - AVk_y$ is zero and hence in this case the original system $Ax = b$ is consistent.

We give two simple examples of termination. Assume first that $b \perp \mathcal{R}(A)$. Then the reduction will terminate with $\alpha_1 = 0$, and $x = 0$ is the minimal norm least squares solution. As a second example, assume that the bidiagonalization terminates with $\beta_2 = 0$. Then the system $Ax = b$ is consistent and the minimum norm solution equals

$$x = (\beta_1/\alpha_1)v_1, \quad v_1 = V_1 e_1 = P_1 e_1.$$

The minimally dimensioned core subproblem, obtained by terminating the bidiagonalization of $(b\,A)$ when the first zero element is encountered, has several important properties. The matrix B_k has full column rank and its singular values are simple. Further, the right-hand side βe_1 has non-zero components along each left singular vector of B_k. These properties considerably simplify the solution of the LS or TLS subproblem; see Paige and Strakoš (2002).

7.3. Bidiagonalization using a Lanczos process

In the second approach to bidiagonalization in Golub and Kahan (1965), the columns of U and V are generated sequentially by a Lanczos process. The following algorithm is identical to the procedure Bidiag 1 in Paige and Saunders (1982*b*). From (7.2) we get the equations

$$AV = UB \quad \text{and} \quad A^T U = V B^T. \tag{7.9}$$

Setting

$$U = (u_1\ u_2 \cdots\ u_{n+1}), \qquad V = (v_1\ v_2 \cdots\ v_n),$$

and equating columns in the two equations (7.9), we obtain the relations

$$\begin{aligned} A^T u_1 &= \alpha_1 v_1, \qquad A^T u_j = \beta_j v_{j-1} + \alpha_j v_j, \quad j = 2, \ldots, n, \\ A v_j &= \alpha_j u_j + \beta_{j+1} u_{j+1}, \quad j = 1, \ldots, n. \end{aligned}$$

Given the unit starting vector $u_1 = b/\beta_1$, $\beta_1 = \|b\|_2$, these relations can be used to recursively compute the column vectors $v_1, u_2, v_2, \ldots, u_{n+1}$. We get

$$v_j = r_j/\alpha_j, \qquad u_{j+1} = s_j/\beta_{j+1}, \quad j = 1, \ldots, n, \tag{7.10}$$

where

$$r_j = A^T u_j - \beta_j v_{j-1}, \qquad \alpha_j = \|r_j\|_2, \tag{7.11}$$

$$s_j = A v_j - \alpha_j u_j, \qquad \beta_{j+1} = \|s_j\|_2. \tag{7.12}$$

The advantage of using this process to generate B_k is that we only need to be able to compute matrix–vector products with A and A^T.

The recurrence relations can also be written in matrix form as

$$U_{k+1}(\beta_1 e_1) = b, \tag{7.13}$$

$$A V_k = U_{k+1} B_k, \tag{7.14}$$

$$A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T. \tag{7.15}$$

In exact arithmetic it holds that $V_k^T V_k = U_k^T U_k = I_k$. From the uniqueness of the bidiagonal decomposition it follows that B_k, U_k and V_k, are the same as generated by the Householder algorithm.

If $\|r_j\|_2$ and $\|s_j\|_2 = 0$ for $j \leq n$, this process will terminate prematurely with $\alpha_j = 0$ and $\beta_{j+1} = 0$, respectively. However, as shown previously, in these cases the least squares problem is separable and the minimum norm solution can be obtained from the partial decomposition computed so far.

7.4. A Krylov subspace method for least squares

For a square matrix $C \in \mathbb{R}^{n\times n}$ and a vector $z \in \mathbb{R}^n$ we define the Krylov subspace

$$\mathcal{K}_k(C, z) = \operatorname{span}\Big\{z, Cz, \dots, C^{k-1}z\Big\}. \tag{7.16}$$

From the Lanczos recurrence relations (7.10)–(7.12) it follows by induction that $u_{j+1} \in \mathcal{K}_j(AA^T, u_1)$ and $v_j \in \mathcal{K}_j(A^TA, A^Tu_1)$, $j = 1 : n$. Hence the columns of U_{k+1} and V_k form orthonormal bases for the Krylov subspaces

$$\mathcal{R}(U_k) = \mathcal{K}_k(AA^T, u_1), \qquad \mathcal{R}(V_k) = \mathcal{K}_k(A^TA, A^Tu_1). \tag{7.17}$$

Suppose that after performing the first k steps of bidiagonalization we seek an approximate least squares solution of the form

$$x_k = V_k y_k. \tag{7.18}$$

By (7.17) this is equivalent to restricting x_k to lie in the Krylov subspace $\mathcal{K}_k(A^TA, A^Tb)$. Using (7.14) we obtain

$$b - Ax_k = b - AV_k y_k = U_{k+1}(\beta_1 e_1 - B_k y_k)$$

and from the orthogonality of the columns of U_{k+1} it follows that

$$\|b - Ax_k\|_2 = \|\beta_1 e_1 - B_k y_k\|_2.$$

Hence $\|b - Ax_k\|_2$ is minimized over all $x_k \in \mathcal{R}(V_k)$ by taking y_k as a solution to the least squares problem

$$\min_{y_k} \|\beta_1 e_1 - B_k y_k\|_2. \tag{7.19}$$

This problem is of exactly the same form as that obtained when the reduction is completed. Since we are minimizing $\|b - Ax\|_2$ over an increasing nested set of subspaces it follows that the sequence $\|b - Ax_k\|_2$, $k = 1 : n$, will be non-increasing. Hence we can solve (7.19) for $k = 1, 2, \dots$ and stop when the residual norm of the solution is small enough. We then accept $x = V_k y_k$ as an approximate solution of the original least squares problem. Except for some details this is essentially the LSQR algorithm by Paige and Saunders (1982*b*). The convergence properties of this algorithm will be discussed in Section 10.2.

The bidiagonal least squares problem can easily be solved by reducing B_k to *upper* bidiagonal form. The QR decomposition of B_k is computed by premultiplication with a sequence of Givens rotations. If these are also applied to the right-hand side $\beta_1 e_1$ we obtain

$$O_k^T \begin{pmatrix} B_k & \beta_1 e_1 \end{pmatrix} = \begin{pmatrix} R_k & d_k \\ 0 & \bar{\phi}_{k+1} \end{pmatrix}, \tag{7.20}$$

where

$$Q_k^T = G_{k,k+1} \cdots G_{23} G_{12}, \quad G_{j,j+1} = \begin{pmatrix} c_j & s_j \\ -s_j & c_j \end{pmatrix}, \quad j = 1 : k, \tag{7.21}$$

is a product of Givens rotations and

$$R_k = \begin{pmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \rho_3 & \ddots & \\ & & & \ddots & \theta_k \\ & & & & \rho_k \end{pmatrix}, \qquad f_k = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_k \end{pmatrix}. \tag{7.22}$$

The solution and residual to (7.19) can then be computed from upper bidiagonal linear system

$$R_k y_k = d_k, \qquad s_k = \bar{\phi}_{k+1} Q_k e_{k+1}. \tag{7.23}$$

The residual $A^T r_k$ to the normal equations will be zero for the exact solution x and this quantity can therefore be used as a stopping criterion. Using (7.15) we obtain

$$A^T r_k = \bar{\phi}_{k+1} A^T U_{k+1} Q_k e_{k+1} = \bar{\phi}_{k+1} \alpha_{k+1} c_k v_{k+1},$$

where $c_k = e_{k+1}^T Q_k e_{k+1}$, the $(k+1)$st diagonal element of Q_k, equals the element c_k in $G_{k,k+1}$. Hence the norm

$$\|A^T r_k\|_2 = \bar{\phi}_{k+1} \alpha_{k+1} |c_k| \tag{7.24}$$

is cheaply computable.

Paige and Saunders (1982*b*) showed an ingenious way to interleave the solution of (7.19) with the reduction to bidiagonal form. The QR decomposition (7.20) can be efficiently updated. Assume that we have computed R_{k-1}, f_{k-1} and $\bar{\phi}_k$ in (7.21)–(7.22). To update these quantities when a column is added to B_k we first apply the Givens rotation to rows $k-1, k$ in the last column in B_k:

$$G_{k-1,k} \begin{pmatrix} 0 \\ \alpha_k \end{pmatrix} = \begin{pmatrix} \theta_k \\ \bar{\rho}_k \end{pmatrix}.$$

Next we construct and apply a Givens rotation $G_{k,k+1}$ to zero out the element β_{k+1}

$$G_{k,k+1}\begin{pmatrix}\bar{\rho}_k & \bar{\phi}_k \\ \beta_{k+1} & 0\end{pmatrix} = \begin{pmatrix}\rho_k & \phi_k \\ 0 & \bar{\phi}_{k+1}\end{pmatrix}.$$

(Here only elements affected by the rotation are shown.)

From (7.23) we note that y_k will normally differ from y_{k-1} in all its elements. However, since R_k and f_k differ from R_{k-1} and f_{k-1} only in the last row and column, we can write

$$x_k = V_k R_k^{-1} f_k = D_k f_k = x_{k-1} + \phi_k d_k,$$

where $D_k = \begin{pmatrix} d_1 & d_2 & \cdots & d_k\end{pmatrix}$ is obtained from $R_k^T D_k = V_k^T$ by forward substitution, giving

$$d_k = \rho_k^{-1}(v_k - \theta_k d_{k-1}). \tag{7.25}$$

Only the last iterates d_k and x_k have to be saved. Although the residual $r_k = U_{k+1}s_k$ is not cheaply computable, by (7.23) its norm equals

$$\|r_k\|_2 = |\bar{\phi}_{k+1}|.$$

The Householder algorithm gives a backward stable algorithm for computing the sequence of Krylov subspace approximations x_k, $k = 1, 2, \ldots$. For problems where A is dense the cost is comparable to that for the Lanczos approach.

In many least squares problems the 'effective rank' of the problem is much smaller than n, *i.e.*, a good approximate solution can be found in a subspace of much smaller dimension than n. For example, this is the case in multiple linear regression problems, where many columns of A are nearly linearly dependent. The Krylov subspace method described above is a standard tool for regression in chemometrics. In this context it is known as the partial least squares (PLS) method; see Wold, Ruhe, Wold and Dunn (1984). It is known that PLS often gives a faster reduction of the residual than TSVD; see Eldén (2004). PLS is often implemented by a deflation method called NIPALS (Nonlinear Iterative Partial Least Squares), or using the LSQR. Since many of these problems are neither sparse nor particularly large an implementation based on Householder bidiagonalization should be preferred.

8. Constrained and regularized problems

8.1. Constrained least squares problems

In various applications the solution to a least squares problem is required to satisfy a subsystem of linear equations exactly. This is the least squares problem with equality constraints (LSE)

$$\min \|b - Ax\|_2 \quad \text{subject to} \quad Bx = d, \tag{8.1}$$

where $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$, with $m+p \geq n \geq p$. If we assume that

$$\operatorname{rank}(B) = p, \qquad \mathcal{N}(A) \cap \mathcal{N}(B) = 0,$$

then this problem has a unique solution. A survey of solution methods is given in Björck (1996, Section 5.1). Perturbation bounds for problem LSE are derived in Eldén (1980). An analysis of the accuracy and stability of three different implementations of the null space method is given in Cox and Higham (1999*a*), where a forward error bound suitable for practical use is also derived.

Problem LSE arises, *e.g.*, in the solution of inequality-constrained least squares problem (LSI):

$$\min \|b - Ax\|_2 \quad \text{subject to} \quad l \leq Bx \leq u, \tag{8.2}$$

where the inequalities are to be interpreted componentwise, $l_i \leq (Bx)_i \leq u_i$, $i = 1:p$. We assume that linear equality constraints, if present, have been eliminated and that the set $\mathcal{M} = \{x \mid l \leq Bx \leq u\}$ is not empty.

A special case is the least distance problem (LDP)

$$\min \|x_1\|_2 \quad \text{subject to} \quad l \leq Bx \leq u, \tag{8.3}$$

where $x^T = \begin{pmatrix} x_1^T & x_2^T \end{pmatrix}$.

Questions of existence, uniqueness and boundedness of solutions to problem LSI are given by Lötstedt (1983). It is convenient to split the solution into two mutually orthogonal components

$$x = x_R + x_N, \qquad x_R \in \mathcal{R}(A^T), \qquad x_N \in \mathcal{N}(A). \tag{8.4}$$

The existence of a bounded solution x to (8.2) follows from the facts that the objective function $\|Ax - b\|_2$ is bounded below by 0 and that the constraint set $l \leq Cx \leq u$ is convex and polyhedral. It can further be shown that x_R and Ax are uniquely determined; see Lötstedt (1983, Theorem 1). In particular, if $\operatorname{rank}(A) = n$ then $\mathcal{N}(A)$ is empty and the solution is unique. The sensitivity of the solution of problem LSI to perturbations in the data A, B, b is also studied in Lötstedt (1983).

An important special case of problem LSI is when the inequalities are simple bounds, problem BLS:

$$\min_{l \leq x \leq u} \|Ax - b\|_2. \tag{8.5}$$

(Some lower and upper bounds may not be present.) For reasons of computational efficiency it is essential that such constraints be considered separately from more general constraints in (8.2). If $\operatorname{rank}(A) = n$ the BLS problem is a strictly convex optimization problem and there exists a unique solution for any vector b.

Sometimes only one-sided bounds apply. After a shift these can then be transformed into $x \geq 0$ and we have a least squares problems with nonnegativity constraints (NNLS):

$$\min_{x \geq 0} \|Ax - b\|_2. \tag{8.6}$$

Problems BLS and NNLS arise naturally in many applications, *e.g.*, reconstruction problems in geodesy and tomography, contact problems for mechanical systems, control problems, *etc.* It can often be argued that a linear model is only realistic when the variables are constrained within meaningful intervals.

To determine a unique solution for the BLS problem when $\operatorname{rank}(A) < n$, we may look for a solution to the problem

$$\min_{x \in \mathcal{M}} \|x\|_2, \quad \mathcal{M} = \Bigl\{x \mid \min_{l \leq x \leq u} \|Ax - b\|_2\Bigr\}. \tag{8.7}$$

Lötstedt (1984) developed a two-stage algorithm to solve problem (8.7). In the first stage a particular solution x to (8.5) is determined. If x is decomposed according to (8.4) then x_R is uniquely determined, but any x_N such that x remains feasible is admissible. Since $\|x\|_2^2 = \|x_R\|_2^2 + \|x_N\|_2^2$ in the second stage we need to solve

$$\min_{l \leq x \leq u} \|x_N\|_2, \quad x_N \in \mathcal{N}(A). \tag{8.8}$$

Let

$$A^T = (U_1 \; U_2) \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \tag{8.9}$$

be a full orthogonal decomposition of A^T with S nonsingular (this could be the SVD of A^T). Then $\mathcal{R}(A^T)$ is spanned by U_1 and $\mathcal{N}(A)$ by U_2 and we have

$$x_R = U_1(U_1^T x), \qquad x_N = U_2(U_2^T x) = U_2 z.$$

Since U_2 has orthonormal columns, problem (8.8) is equivalent to

$$\min \|z\|_2, \quad l - x_R \leq U_2 z \leq u - x_R, \tag{8.10}$$

which is a least distance problem for z.

In general, methods for problems with inequality constraints are iterative in nature. At the solution only a certain subset of the inequalities will be active, *i.e.*, satisfied with equality. If this set was known the solution to the LSI problem could be found from a problem with equality constraints, for which efficient solution techniques exist. In active set methods a sequence of equality-constrained problems are solved corresponding to predictions of the correct active set.

An implementation of an active set method for NNLS is given in Lawson and Hanson (1995). This can also be used to solve problem LDP using a dual approach; see Cline (1975). Consider the least distance problem with lower bounds

$$\min_x \|x\|_2, \quad \text{subject to} \quad c \le Bx. \tag{8.11}$$

Let $u \in \mathbb{R}^{m+1}$ be the solution to the NNLS problem

$$\min_u \|Au - b\|_2, \quad \text{subject to} \quad u \ge 0, \tag{8.12}$$

where

$$A = \begin{pmatrix} B^T \\ c^T \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{matrix} \}n \\ \}1 \end{matrix}. \tag{8.13}$$

Let the residual corresponding to the solution be

$$r \equiv b - Au = \begin{pmatrix} r_1 \\ \gamma \end{pmatrix} \begin{matrix} \}n \\ \}1 \end{matrix}, \qquad \sigma = \|r\|_2.$$

If $\sigma \neq 0$, then the vector $x = -r_1/\gamma$, is the unique solution to (8.11). If $\sigma = 0$, then the constraints $g \le Bx$ are inconsistent and (8.11) has no solution. Hence this relation also gives a method to determine if a set of linear inequalities has a feasible solution by solving an NNLS problem.

Problem LSI can be transformed into an LDP problem by using the orthogonal decomposition (8.9). An algorithm based on this transformation and the dual approach for solving the LDP problem is given by Lawson and Hanson (1995, Chapter 23) (see also Haskell and Hanson (1981)). The method proposed by Schittkowski (1983) for solving the LDP problem is a primal method.

8.2. Regularization of discrete ill-posed problems

Inverse problems are problems where we want to determine the structure of a physical system from its measured behaviour. Such problems are often ill-posed in the sense that their solution does not depend continuously on the data. Inverse problems arise in many application such as astronomy, computerized tomography, geophysics, signal processing, *etc.*

The discretization of ill-posed problems gives rise to a class of least squares problems

$$\min_x \|Ax - b\|_2, \quad A \in \mathbb{R}^{m\times n}, \tag{8.14}$$

that share a number of properties. The singular values of A decay gradually and cluster at zero, resulting in a huge condition number; *cf.* Figure 6.1. However, the components of the right-hand side b along singular vectors corresponding to small singular values decay rapidly, so that the systems are effectively well-conditioned in the sense of Chan and Foulser (1988).

Owing to the huge condition number and the presence of noise in the right-hand side b, additional information, *e.g.*, in the form of constraints, must be imposed on the solution in order to get a regularized problem with a well-determined solution. Neglecting this can be catastrophic, since it may lead to a meaningless solution of huge norm, or even to failure of the algorithm.

One common regularization method is to project the linear system onto a smaller dimensional problem by solving

$$\min_{x \in \mathcal{V}_k} \|b - Ax\|_2,$$

where $\mathcal{V}_k$ is a suitably chosen subspace of dimension $k < n$. One possible choice is the subspace spanned by the first k right-singular vectors of A, which leads to a TSVD solution (3.11). Another choice is to use the Krylov subspaces $\mathcal{V}_k = \mathcal{K}_k(A^TA, A^Tu_1)$ (7.17), which corresponds to using LSQR or the PLS method. There is some evidence that for ill-posed problems the Krylov subspaces often have better approximation properties than the singular subspaces used in TSVD; see Hanke (2001).

Another widely used regularization method is Tikhonov regularization (Tikhonov 1963). In this method an approximate solution is obtained by solving a least squares problem with a quadratic constraint

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad \|Lx\|_2 \le \gamma, \quad L \in \mathbb{R}^{p\times n}. \tag{8.15}$$

Here $\gamma > 0$ is the regularization parameter, which is used to find a balance between the size of the residual $\|Ax - b\|_2$ and size of the solution as measured by the norm (or seminorm) $\|Lx\|_2$. This makes it possible to include *a priori* information about the size or smoothness of the solution. In statistics Tikhonov's method is known as ridge regression. A survey of the properties of least squares problems with a quadratic constraint is given by Gander (1981). Problem (8.15) is related to, but less general than, the trust-region subproblem in optimization; see Rojas and Sorensen (2002).

In the following we discuss the simple but important case when $L = I$. Methods to transform (8.15) into this standard form are described in Hansen (1998, Section 2.3). In (8.15) the constraint is binding if $\|A^\dagger b\|_2 > \gamma$, which is invariably the case in regularization of ill-posed problems. Then $x = x(\lambda)$ solves the least squares problem

$$\min_x \left\| \begin{pmatrix} A \\ \sqrt{\lambda}L \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2, \tag{8.16}$$

where the Lagrange multiplier λ is determined by the secular equation $g(\lambda) = \|x(\lambda)\|_2 - \gamma = 0$.

Using the singular value decomposition $A = U\Sigma V^T$ and setting $c = U^T b$, the secular equation becomes

$$g(\lambda) = \left(\sum_{i=1}^{n} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2 + \lambda)^2} \right)^{1/2} - \gamma = 0. \qquad (8.17)$$

Here each term is a convex and strictly decreasing function of λ. Since the Euclidean norm is monotonic, the same then also holds for $g(\lambda)$. This shows that if $\gamma < g(0)$ then the secular equation has a unique root $\lambda > 0$.

To determine λ by an iterative method, faster convergence is obtained by writing the secular equation in the form

$$f(\lambda) = \frac{1}{\|x(\lambda)\|_2} - \frac{1}{\gamma} = 0. \qquad (8.18)$$

Reinsch (1971) showed that $f(\lambda)$ is a concave and strictly increasing function for $\lambda > 0$. It follows that Newton's method is monotonically convergent to the solution λ^* from any starting value $\lambda_0 \in [0, \lambda^*]$. Usually about 4–6 iterations suffice, even when $\lambda_0 \ll \lambda^*$.

Writing $x(\lambda) = (A^T A + \lambda I)^{-1} A^T b$ and taking the derivative with respect to λ, we find that

$$f'(\lambda) = -\frac{x^T(\lambda) x'(\lambda)}{\|x(\lambda)\|_2^3}, \qquad x'(\lambda) = -(A^T A + \lambda I)^{-1} x(\lambda). \qquad (8.19)$$

Here

$$x(\lambda)^T x(\lambda)' = -x(\lambda)^T (A^T A + \lambda I)^{-1} x(\lambda) = -\|z(\lambda)\|_2^2, \qquad (8.20)$$

and Newton's method for equation (8.18) becomes

$$\lambda_{k+1} = \lambda_k + \left(\frac{\|x(\lambda_k)\|_2}{\gamma} - 1 \right) \frac{\|x(\lambda_k)\|_2^2}{\|z(\lambda_k)\|_2^2}. \qquad (8.21)$$

Given the QR decomposition

$$Q(\lambda_k)^T \begin{pmatrix} A \\ \sqrt{\lambda_k} I \end{pmatrix} = \begin{pmatrix} R(\lambda_k) \\ 0 \end{pmatrix}, \qquad Q(\lambda_k)^T \begin{pmatrix} b \\ 0 \end{pmatrix} = \begin{pmatrix} c_1(\lambda_k) \\ c_2(\lambda_k) \end{pmatrix}, \qquad (8.22)$$

and using (8.20) we obtain

$$x(\lambda_k) = R(\lambda_k)^{-1} c_1(\lambda_k), \qquad z(\lambda_k) = R(\lambda_k)^{-T} x(\lambda_k).$$

The main cost per iteration step in Newton's method is the QR decomposition (8.22). Computing the derivative costs only one triangular solve. Hence Newton's method is to be preferred to the secant method and other methods based on interpolation.

When $A \in \mathbb{R}^{m \times n}$ is a full matrix computing the Householder QR decomposition in each iteration requires about mn^2 multiplications. As pointed out by Moré (1978), if $m > n$ it is more efficient to initially compute the

QR of A at a cost of $mn^2 - n^3/3$ multiplications. For each value λ_k we can then compute $R(\lambda_k)$ from the QR decomposition of

$$\begin{pmatrix} R \\ \sqrt{\lambda_k} I \end{pmatrix}$$

in $n^3/3$ multiplications. However, if A is sparse R may have many more nonzero elements than A and then this modification will instead *increase* the amount of work.

The repeated QR decomposition can be avoided by computing the SVD or a bidiagonal decomposition of A. The bidiagonal form can be updated with $O(n)$ multiplications when λ changes, as shown by Eldén (1977). Since the initial reduction requires $4n^3/3$ flops ($2n^3/3$ if $m = n$) the reduction to bidiagonal form pays off in the dense case if more than four Newton iterations are needed.

9. Direct methods for sparse problems

A matrix A is called sparse if many of its entries are zero. Clearly a square and banded matrix is sparse, but sparse matrices that have much more irregular sparsity pattern occur in many applications in science and engineering. Often these problems are huge and it is essential that advantage is taken of sparsity for storage and operations. Matrix operations on general sparse matrices are supported in MATLAB: see Gilbert, Moler and Schreiber (1992) for a discussion of design and implementation of these algorithms.

9.1. QR factorization of banded matrices

A banded matrix $A \in \mathbb{R}^{m\times n}$, $m \geq n$, is a matrix for which in each row the nonzero elements lie in a narrow band. By the bandwidth of A we mean smallest number w such that

$$|j-k| \geq w \quad \Rightarrow \quad a_{ij}a_{ik} = 0 \quad i = 1:n. \tag{9.1}$$

It is easy to deduce that if A has bandwidth w then the upper triangular part of A^TA and its Cholesky factor R also have bandwidth w. Forming A^TA and computing its Cholesky factorization therefore requires only about $\frac{1}{2}(m+n)w^2$ multiplications. The QR decomposition of a banded matrix can also be computed efficiently, but the implementation is not quite trivial! The standard Householder QR factorization algorithm can be very inefficient and cause unnecessary intermediate fill-in. Similarly, if a row-wise reduction with Givens rotations is used the operation count and intermediate storage requirement can differ strongly for different row orderings of A.

Reid (1967) showed that for banded rectangular matrices the QR factorization can be obtained very efficiently by sorting the rows of A and suitably subdividing the Householder transformations. The rows of A are first sorted by leading entry order (*i.e.*, increasing minimum column subscript order) so that the matrix is represented as q blocks

$$A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_q \end{pmatrix}, \quad q \le n,$$

where in block A_i the first nonzero element of each row is in column i. The Householder QR process is then applied to the matrix in q major steps. In the first step a QR decomposition of the first block A_1 is computed, yielding R_1. Next, at step k, $k = 2 : q$, R_{k-1} will be merged with A_k, yielding

$$Q_k^T \begin{pmatrix} R_{k-1} \\ A_k \end{pmatrix} = R_k.$$

Since the rows of block A_k have their first nonzero elements in column k, the first $k-1$ rows of R_{k-1} will not be affected. The matrix Q can be implicitly represented in terms of the Householder vectors of the factorization of the subblocks. This sequential Householder algorithm, which is also described in Lawson and Hanson (1995, Chapter 27), requires $(m + 3n/2)w(w + 1)$ multiplications or about twice the work of the less stable Cholesky approach.

A banded upper triangular matrix can be reduced to bidiagonal form using an algorithm similar to the one used by Schwarz (1968) for reducing a symmetric banded matrix to tridiagonal form. However, because each zero element introduced generates a new nonzero element that has to be 'chased' across the border of the matrix, the reduction is much more expensive than the QR decomposition The reduction of a banded upper triangular matrix to bidiagonal form requires $\approx 4n^2(w - 2)$ multiplications. A computational routine for this called xGBBRD in LAPACK uses a vectorized version due to Kaufman (1984), in which several elements are chased in parallel.

A special case occurs in regularization, where we need the repeated QR decomposition of

$$\begin{pmatrix} R_1 \\ \sqrt{\lambda} R_2 \end{pmatrix},$$

for banded matrices R_1 and R_2. Note that if the above row ordering algorithm is applied this will interleave the rows of R_1 and R_2. Eldén (1984) gives a row-wise Givens algorithm which requires approximately $2n(w_1^2 + w_2^2)$ multiplications and is optimal in that no unnecessary fill-in is created.

There is often a substantial similarity in the structure of many large-scale sparse least squares problems. In particular, the problem can often be put in the following bordered block diagonal or block angular form:

$$A = \left(\begin{array}{cccc|c} A_1 & & & & B_1 \\ & A_2 & & & B_2 \\ & & \ddots & & \vdots \\ & & & A_M & B_M \end{array}\right), \quad x = \left(\begin{array}{c} x_1 \\ x_2 \\ \vdots \\ x_M \\ \hline z \end{array}\right), \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{pmatrix}, \tag{9.2}$$

where $A \in \mathbb{R}^{m \times n}$,

$$A_i \in \mathbb{R}^{m_i \times n_i}, \quad B_i \in \mathbb{R}^{m_i \times p}, \quad i = 1, 2, \ldots, M.$$

We assume in the following, for simplicity, that $\operatorname{rank}(A) = n$. Note that the variables $x_1, \ldots, x_M$ are coupled only to the variables z, which reflects a 'local connection' structure in the underlying physical problem. There is usually further structure in the individual blocks A_i and B_i that should be taken advantage of.

Applications where the form (9.2) arises naturally in many applications including photogrammetry (Golub, Luk and Pagano 1979), Doppler radar positioning (Manneback, Murigande and Toint 1985), geodetic survey problems (Golub and Plemmons 1980) and GPS positioning (Chang and Paige 2003),

It is easily seen that then the factor R in the QR decomposition of A will have the block structure

$$R = \left(\begin{array}{cccc|c} R_1 & & & & S_1 \\ & R_2 & & & S_2 \\ & & \ddots & & \vdots \\ & & & R_M & S_M \\ \hline & & & & R_{M+1} \end{array}\right), \tag{9.3}$$

where by assumption $R_i \in \mathbb{R}^{n_i \widetilde{n}_i}$, $i = 1, \ldots, M+1$ is nonsingular.

The following algorithm for solving least squares problems of block angular form by QR decomposition is given in Golub *et al.* (1979).

(1) For $i = 1, 2, \ldots, M$ reduce the diagonal block A_i to upper triangular form by a sequence of orthogonal transformations applied to (A_i, B_i) and the right-hand side b_i, yielding

$$Q_i^T (A_i, B_i) = \begin{pmatrix} R_i & S_i \\ 0 & T_i \end{pmatrix}, \qquad Q_i^T b_i = \begin{pmatrix} c_i \\ d_i \end{pmatrix}.$$

It is usually advantageous to continue the reduction in step (1) so that the matrices T_i, $i = 1, \ldots, M$, are brought into upper trapezoidal form.

(2) Compute the QR decomposition

$$\tilde{Q}_{M+1}^T \begin{pmatrix} T_1 \\ \vdots \\ T_M \end{pmatrix} = \begin{pmatrix} R_{M+1} \\ 0 \end{pmatrix}, \quad \tilde{Q}_{M+1}^T \begin{pmatrix} d_1 \\ \vdots \\ d_M \end{pmatrix} = \begin{pmatrix} c_{M+1} \\ d_{M+1} \end{pmatrix}.$$

Then the linking variables z are obtained from the triangular system

$$R_{M+1} z = c_{M+1}$$

and the residual norm equals $\rho = \|d_{M+1}\|_2$.

(3) For $i = M, \ldots, 1$ compute $x_M, \ldots, x_1$ by back-substitution in the triangular systems

$$R_i x_i = c_i - S_i z.$$

Note that in steps (1) and (3) the computations can be performed in parallel on the M independent subsystems.

There are many alternative ways to organize this algorithm. Cox (1990) considers the following modifications to reduce the storage requirement. By merging steps (1) and (2) it is not necessary to hold all blocks T_i simultaneously in memory. Even more storage can be saved by discarding R_i and S_i after T_i has been computed in step (1). These matrices are then recomputed when needed for step (3). Indeed, only R_i needs to be recomputed, since when z has been computed in step (2), we can determine x_i by solving the least squares problems

$$\min_{x_i} \|A_i x_i - g_i\|_2, \qquad g_i = b_i - B_i z, \quad i = 1, \ldots, M.$$

Hence, to determine x_i we only need to (re-)compute the QR factorization of (A_i, g_i). In some practical problems this modification can reduce the storage requirement by an order of magnitude, while the recomputation of R_i only increases the operation count by a few per cent.

9.3. QR decomposition for general sparse matrices

The factor R in the QR decomposition of A is mathematically equivalent to the Cholesky factor of the cross product matrix A^TA. Although this is true in exact arithmetic the difficulty in recognizing numerical cancellation means that the computed structure of the Cholesky factor can overestimate the structure of R. However, in many cases it accurately predicts the structure of R.

A reordering of the rows and columns of A can be written $\hat{A} = P_r A P_c$, where P_r and P_c are permutation matrices. Since

$$\hat{A}^T \hat{A} = P_c^T A^T P_r^T P_r A P_c = P_c^T A^T A P_c,$$

this corresponds to a symmetric reordering of $A^T A$. Hence the column ordering of A will affect the structure and number of nonzeros $\text{nnz}(R)$ in R. A reordering of the rows in A has no effect on the final R, but influences the sparsity of Q and the number of operations needed to perform the decomposition.

Before computing R numerically, it is important to find a to find a column ordering that approximately minimizes the number of nonzero elements in R. (Finding the optimal ordering is known to be an NP-complete problem.) The simplest ordering methods use *a priori* information, such as ordering the columns in order of increasing column count. Such orderings are usually inferior to ordering methods obtained from a symmetric ordering on the structure of the normal matrix, using minimum degree or nested dissection. The graph $G(A^T A)$ representing the structure of $A^T A$ can be constructed directly from the structure of the matrix A as being the direct sum of all the subgraphs $G(a_i a_i^T)$, $i = 1, \ldots, m$. Note that the nonzeros in any row a_i^T will generate a subgraph where all pairs of nodes are connected. Good surveys of the state of the art in symmetric ordering algorithms and direct methods for sparse linear systems are given in Duff (1997) and Dongarra *et al.* (1998, Chapter 6).

After a column ordering P_c has been determined, the rows in the permuted matrix AP_c should be reordered to minimize intermediate fill-in. The following heuristic row ordering method usually works well. Denote the column index for the first and last nonzero elements in the ith row by f_i and l_i, respectively. The rows are ordered by increasing values of f_i, and in each group of rows with equal f_i after increasing l_i. (This is the row ordering used for rectangular banded matrices.)

For the numerical QR decomposition several implementations of multifrontal methods for QR have been developed; see Matstoms (1995), Amestoy, Duff and Puglisi (1996). Multifrontal methods have the advantage of allowing dense matrix kernels to be used in the sparse matrix code and can be considered as a generalization of methods for banded matrices.

A problem with the QR decomposition is that for a large class of sparse matrices the matrix Q will be much less sparse than R. If $A \in \mathbb{R}^{m \times n}$ is a matrix of full column rank, such that its column intersection graph is a member of a $\sqrt{n}$-separable class of graphs, then it is shown in Gilbert, Ng and Peyton (1997) that there exists a column permutation P such that

$$\text{nnz}(R) = O(n \log n), \qquad \text{nnz}(Q_1) = O(n\sqrt{n}),$$

where $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$. These bounds are best possible within a constant factor for a large class of matrices. It is therefore not advisable to store Q_1 (or Q) explicitly. In MATLAB sparse QR the matrix Q *is* provided if wanted, but this option should be used with care!

Because of the lack of sparsity in Q a common practice when solving sparse least squares problems is to compute $Q^T b$ 'on the fly' for any right-hand side available at the time of decomposition and discard the orthogonal transformations after they have been used. This approach, advocated in George and Heath (1980), is also taken in several later multifrontal QR codes; see Matstoms (1994), Sun (1996) and Pierce and Lewis (1997).

To discard Q creates a problem if additional right-hand sides b are to be treated later. George and Heath (1980) suggested that if the original matrix A is saved one can use R from the QR decomposition, rewriting the normal equations as

$$R^T R x = A^T b, \tag{9.4}$$

known as the seminormal equations (SNE). Even with R from the QR decomposition in (9.4) this is not an acceptable-error stable algorithm for the least squares problem. However, if improved by one step of iterative refinement in fixed precision it is, in most cases, numerically acceptable. A detailed error analysis of this algorithm, called CSNE, is given in Björck (1987).

Lu and Barlow (1996) use the multifrontal QR factorization method and represent Q implicitly by storing the frontal Householder vectors. Let A_i be the matrix consisting of those rows of A that have their leading nonzero element in column i. Provided that the number of rows of each A_i is bounded by a constant, Lu and Barlow (1996) show that their method requires only $O(n \log n)$ storage if $A \in \mathbb{R}^{m \times n}$ is a member of a $\sqrt{n}$-separable class of graphs. Adlers (2000) has developed a similar implementation for MATLAB, which provides Q by storing the frontal Householder vectors. Operator overloading is used to implement the matrix–vector products Qy and $Q^T y$.

10. Iterative methods

10.1. Introduction

For several classes of large sparse least squares problems iterative methods are useful alternatives to direct methods. Iterative methods for the linear least squares problem (1.1) can be derived by applying an iterative method for symmetric positive definite linear systems to the normal equations $A^T A x = A^T b$. However, it is important to avoid the explicit formation of $A^T A$ since this leads to a loss of stability. Also, $A^T A$ can be much less sparse than A, leading to higher cost of storage and operations. Instead the factored form of the normal equations $A^T(Ax-b)=0$ should be employed. This is well illustrated by the non-stationary Richardson's method for least squares problems, which should be written

$$x_{k+1} = x_k + \omega_k A^T r_k, \quad r_k = b - Ax_k. \tag{10.1}$$

This method, also known as Landweber's method, can be shown to converge, provided that, for some $\epsilon > 0$,

$$0 < \epsilon < \omega_k < (2-\epsilon)/\sigma^2_{\max}(A), \quad \forall k.$$

As is typical for iterative methods for least squares problems the matrix A need only accessed through its action in the matrix–vector operations Ax_k and $A^T r_k$. This can also be an advantage for problems where A is dense but structured. Consider, *e.g.*, a rectangular Toeplitz matrix

$$T = \begin{pmatrix} t_0 & t_1 & \dots & t_n \\ t_{-1} & t_0 & \dots & t_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ t_{-m} & t_{-m+1} & \dots & t_0 \end{pmatrix} \in \mathbb{R}^{(m+1)\times(n+1)}, \quad \text{for } m \geq n,$$

defined by the $(n+m+1)$ values of $t_{-m}, \dots, t_0, \dots, t_n$. Toeplitz linear least squares problems of large dimension arise in many applications, *e.g.*, in signal restoration, acoustics, and seismic exploration. Matrix–vector products Tx_k and $T^T r_k$ can be computed at a cost of $O(m \log m)$ by embedding T in a circulant matrix and using FFT; see Chan, Nagy and Plemmons (1994). Fast multiplication algorithms also exist for several other classes of structured matrices, *e.g.*, Vandermonde and Cauchy matrices; see Gohberg and Olshevsky (1994).

10.2. Implementation of Krylov subspace methods

In Sections 7.4–7.5 we outlined the LSQR Krylov subspace algorithm for the least squares problem. A different way to compute the same sequence of approximations x_k is to use the conjugate gradient method (CG), developed in the early 1950s. This has become a basic tool for solving large sparse symmetric positive definite linear systems. If applied to the normal equations it can also be used to solve linear least squares problems. The conjugate gradient algorithm for the normal equations generates approximations x_k in the Krylov subspace

$$x_k \in \mathcal{K}_k(A^T A, s_0), \quad s_0 = A^T b. \tag{10.2}$$

The iterates x_k generated are optimal in the sense that, for each k, $y = x_k$ minimizes the error functional

$$E(y) = (x-y)^T (A^T A)(x-y), \qquad y \in \mathcal{K}_k(A^T A, s_0). \tag{10.3}$$

Using $A(x - x_k) = b - r - Ax_k = r_k - r$, we obtain

$$E(x_k) = \|r - r_k\|^2 = \|r_k\|^2 - \|r\|^2, \tag{10.4}$$

where the second expression follows from the fact that $r \perp r - r_k$.

The following version of CGLS was originally given by Hestenes and Stiefel (1952, p. 424) and Stiefel (1952/53).

Initialize

$$r_0 = b, \quad s_0 = p_1 = A^T r_0, \quad \gamma_0 = \|s_0\|_2^2, \tag{10.5}$$

and for $k = 1, 2, \dots$ compute

$$\begin{aligned} q_k &= Ap_k, \\ \alpha_k &= \gamma_{k-1}/\|q_k\|_2^2, \\ x_k &= x_{k-1} + \alpha_k p_k, \\ r_k &= r_{k-1} - \alpha_k q_k, \quad s_k = A^T r_k, \\ \gamma_k &= \|s_k\|_2^2, \\ \beta_k &= \gamma_k/\gamma_{k-1}, \\ p_{k+1} &= s_k + \beta_k p_k. \end{aligned}$$

Each iteration requires two matrix vector products and $2m + 3n$ multiplications. Storage is required for the n-vectors x, p and m-vectors r, q.

The variational property of the conjugate gradient method implies that, in exact arithmetic, the error functional $\|r - r_k\|_2$ (as well as $\|r_k\|_2$) decreases monotonically as a function of k. In Hestenes and Stiefel (1952, p. 416) it is proved that the error functional $\|x - x_k\|_2$ also decreases monotonically. However, when $\kappa(A)$ is large, $\|A^T(r - r_k)\|_2$ will often exhibit large oscillations. We stress that this behaviour is *not* a result of rounding errors.

If A has $t \leq n$ distinct singular values, then (in exact arithmetic) the solution is obtained in at most t steps. It is well known that an upper bound on the rate of convergence is given by

$$\|r - r_k\|_2^2 \leq 2\left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|r_0\|_2^2, \tag{10.6}$$

where $\kappa = \kappa(A)$; see Björck (1996, Section 7.4). The convergence also depends on the distribution of the singular values of A and often superlinear convergence is observed.

The LSQR algorithm (Paige and Saunders 1982*a*) generates, in exact arithmetic, the same sequence of approximations x_k as CGLS. In LSQR the Lanczos process described in Section 7.3 is used to compute u_k, v_k and B_k, for $k = 1, 2, \dots$. This is interleaved with the solution of the bidiagonal systems as described in Section 7.5.

In addition to two matrix vector products LSQR requires $3m + 5n$ multiplications and storage of two m-vectors u, Av, and three n-vectors x, v, w. Although this is slightly more storage and operations than CGLS, this is partly offset by the fact that viable rules for stopping the iterations are more costly for CGLS than for LSQR. For LSQR Paige and Saunders (1982*b*) consider several stopping rules which use (estimates of) $\|r_k\|$, $\|x_k\|$, $\|A^T r_k\|$, $\|A\|$, and $\|A^\dagger\|$. All these quantities can be obtained at minimal cost in LSQR. For CGLS $\|s_k\|$ is available but $\|r_k\|$ has to be separately computed, if needed, at an extra cost of m multiplications.

In finite precision orthogonality will be lost for U_k and V_k. This causes a slowdown of convergence, but does not affect the final accuracy. A comparison of the stability of LSQR and CGLS is given in Björck, Elfving and Strakoš (1998). It may be believed that LSQR, since its derivation avoids any references to the normal equations, should have better stability properties than CGLS. To some extent that is true, but the difference is rather small. Björck *et al.* (1998) compares the achievable accuracy in finite precision of different implementations of Krylov subspace methods for solving (1.1). The conclusion from both theoretical analysis and experimental evidence is that LSQR and CGLS are both well behaved and achieve a final accuracy consistent with a backward stable method.

A slightly different version of CGLS is obtained if, instead of r_k, the residual of the normal equations $s_k = A^T r_k$ is recurred. For this, line 4 in algorithm CGLS becomes

$$s_k = s_{k-1} - \alpha_k (A^T q_k).$$

As shown in Björck *et al.* (1998), this small change can substantially lower the achievable final accuracy in CGLS. This can be explained by noting that in this version the right-hand side b is used only in the initialization $p_0 = s_0 = A^T b$ and *no reference to b is made in the iterative phase.* A componentwise round-off analysis shows that round-off occurring in computing $A^T b$ perturbs the solution by δx, where

$$|\delta x| \le \gamma_m |(A^T A)^{-1}|\,|A^T|\,|b|. \tag{10.7}$$

Using norms we obtain the bound

$$\|\delta x\|_2 \le \gamma_m \|(A^T A)^{-1}\|_2\,\|A^T\|_2\,\|b\|_2 = \gamma_m \kappa^2(A)\|b\|_2/\|A\|_2, \tag{10.8}$$

where $\gamma_m = mu/(1 - mu)$. Comparing with the perturbation bounds in Section 2 we see that this error term is a factor $\|b\|_2/\|r\|_2$ larger than is allowed for a backward stable algorithm. Note that for nearly consistent systems we obtain $\|b\|_2 \gg \|r\|_2$.

REFERENCES

M. Adlers (2000), Topics in sparse least squares problems, PhD thesis, Linköping Studies in Science and Technology, Linköping University, Sweden.

P. R. Amestoy, I. S. Duff and C. Puglisi (1996), 'Multifrontal QR factorization in a multiprocessor environment', *Numer. Linear Algebra Appl.* **3**, 275–300.

E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov and D. Sorensen, eds (1995), *LAPACK Users' Guide*, 2nd edn, SIAM, Philadelphia.

E. Anderssen, Z. Bai and J. Dongarra (1992), 'Generalized QR factorization and its applications', *Linear Algebra Appl.* **162–164**, 243–271.

M. Arioli (2000), 'The use of QR factorization in sparse quadratic programming and backward error issues', *SIAM J. Matrix Anal. Appl.* **21**, 825–839.

Z. Bai and J. W. Demmel (1993), 'Computing the generalized singular value decomposition', *SIAM J. Sci. Comput.* **14**, 1464–1486.

Z. Bai, J. Demmel, J. Dongarra, A. Ruhe and H. van der Vorst, eds (2000), *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, PA.

C. Benoit (1924), 'Sur la méthode de résolution des équations normales, etc. (procédés du commandant Cholesky)', *Bull. Géodésique* **2**, 67–77.

C. H. Bischof and G. Quintana-Ortí (1998*a*), 'Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices', *ACM Trans. Math. Software* **24**, 254–257.

C. H. Bischof and G. Quintana-Ortí (1998*b*), 'Computing rank-revealing QR factorizations of dense matrices', *ACM Trans. Math. Software* **24**, 226–253.

Å. Björck (1967*a*), 'Iterative refinement of linear least squares solutions, I', *BIT* **7**, 257–278.

Å. Björck (1967*b*), 'Solving linear least squares problems by Gram–Schmidt orthogonalization', *BIT* **7**, 1–21.

Å. Björck (1987), 'Stability analysis of the method of semi-normal equations for least squares problems', *Linear Algebra Appl.* **88/89**, 31–48.

Å. Björck (1991), Error analysis of least squares algorithms, in *Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms* (G. H. Golub and P. Van Dooren, eds), Vol. 70 of *NATO ASI Series*, Springer, Berlin, pp. 41–73.

Å. Björck (1994), 'Numerics of Gram–Schmidt orthogonalization', *Linear Algebra Appl.* **197–198**, 297–316.

Å. Björck (1996), *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia.

Å. Björck and G. H. Golub (1967), 'Iterative refinement of linear least squares solution by Householder transformation', *BIT* **7**, 322–337.

Å. Björck and C. C. Paige (1992), 'Loss and recapture of orthogonality in the modified Gram–Schmidt algorithm', *SIAM J. Matrix Anal. Appl.* **13**, 176–190.

Å. Björck and C. C. Paige (1994), 'Solution of augmented linear systems using orthogonal factorizations', *BIT* **34**, 1–26.

Å. Björck, T. Elfving and Z. Strakoš (1998), 'Stability of conjugate gradient and Lanczos methods for linear least squares problems', *SIAM J. Matrix Anal. Appl.* **19**, 720–736.

P. Businger and G. H. Golub (1965), 'Linear least squares solutions by Householder transformations', *Numer. Math.* **7**, 269–276.

R. H. Chan, J. G. Nagy and R. J. Plemmons (1994), 'Circulant preconditioned Toeplitz least squares iterations', *SIAM J. Matrix Anal. Appl.* **15**, 80–97.

T. F. Chan (1987), 'Rank revealing QR-factorizations', *Linear Algebra Appl.* **88/89**, 67–82.

T. F. Chan and D. E. Foulser (1988), 'Effectively well-conditioned linear systems', *SIAM J. Sci. Statist. Comput.* **9**, 963–969.

T. F. Chan and P. C. Hansen (1990), 'Computing truncated SVD least squares solutions by rank revealing QR factorizations', *SIAM J. Sci. Statist. Comput.* **11**, 519–530.

T. F. Chan and P. C. Hansen (1992), 'Some applications of the rank revealing QR factorization', *SIAM J. Sci. Statist. Comput.* **13**, 727–741.

T. F. Chan and P. C. Hansen (1994), 'Low-rank revealing QR factorizations', *Numer. Linear Algebra Appl.* **1**, 33–44.

S. Chandrasekaran and I. C. F. Ipsen (1994), 'On rank-revealing factorizations', *SIAM J. Matrix Anal. Appl.* **15**, 592–622.

X.-W. Chang and C. C. Paige (2003), 'An orthogonal transformation algorithm for GPS positioning', *SIAM J. Sci. Comput.* **24**, 1710–1732.

A. K. Cline (1975), The transformation of a quadratic programming problem into solvable form, Technical Report ICASE 75-14, NASA, Langley Research Center, Hampton, VA.

A. J. Cox and N. J. Higham (1999*a*), 'Accuracy and stability of the null space method for solving equality constrained least squares problems', *BIT* **39**, 34–50.

A. J. Cox and N. J. Higham (1999*b*), 'Backward error bounds for constrained least squares problems', *BIT* **39**, 210–227.

M. G. Cox (1990), The least-squares solution of linear equations with block-angular observation matrix, in *Reliable Numerical Computation* (M. G. Cox and S. J. Hammarling, eds), Oxford University Press, UK, pp. 227–240.

B. De Moor and P. Van Dooren (1992), 'Generalizations of the singular value and QR decompositions', *SIAM J. Matrix. Anal. Appl.* **13**, 993–1014.

B. De Moor and H. Zha (1991), 'A tree of generalizations of the ordinary singular value decomposition', *Linear Algebra Appl.* **147**, 469–500.

J. Dongarra, I. S. Duff, D. Sorensen and H. van der Vorst (1998), *Numerical Linear Algebra for High-Performance Computers*, SIAM, Philadelphia, PA.

J. J. Dongarra, J. Du Croz, I. S. Duff and S. Hammarling (1990), 'A set of level 3 basic linear algebra subprograms', *ACM Trans. Math. Software* **16**, 1–17.

I. S. Duff (1997), Sparse numerical linear algebra: Direct methods and preconditioning, in *The State of the Art in Numerical Analysis* (I. S. Duff and G. A. Watson, eds), Oxford University Press, London, UK, pp. 27–62.

L. Eldén (1977), 'Algorithms for the regularization of ill-conditioned least squares problems', *BIT* **17**, 134–145.

L. Eldén (1980), 'Perturbation theory for the least squares problem with linear equality constraints', *SIAM J. Numer. Anal.* **17**, 338–350.

L. Eldén (1984), 'An efficient algorithm for the regularization of ill-conditioned least squares problems with a triangular Toeplitz matrix', *SIAM J. Sci. Statist. Comput.* **5**, 229–236.

L. Eldén (2004), 'Partial least squares vs Lanczos bidiagonalization, I: Analysis of a projection method for multiple projection', *Comp. Stat. Data Anal.* **46**, 11–31.

E. Elmroth and F. G. Gustavson (2000), 'Applying recursion to serial and parallel QR factorization leads to better performance', *IBM J. Res. Develop.* **44**, 605–624.

E. Elmroth and F. G. Gustavson (2001), 'A faster and simpler recursive algorithm for the LAPACK routine DGELS', *BIT* **41**, 936–949.

E. Elmroth, F. G. Gustavson, I. Jonsson and B. Kågström (2004), 'Recursive blocked algorithms and hybrid data structures for dense matrix library software', *SIAM Review* **46**, 3–45.

D. K. Faddeev, V. N. Kublanovskaya and V. N. Faddeeva (1968), 'Solution of linear algebraic systems with rectangular matrices', *Proc. Steklov Inst. Math.* **96**, 93–111.

K. V. Fernando (1998), 'Accurately counting singular values of bidiagonal matrices and eigenvalues of skew-symmetric tridiagonal matrices', *SIAM J. Matrix Anal. Appl.* **20**, 373–399.

R. D. Fierro, P. Hansen, and P. S. K. Hansen (1999), 'UTV tools: Matlab templates for rank revealing UTV decompositions', *Numer. Algorithms* **20**, 165–194.

W. Gander (1981), 'Least squares with a quadratic constraint', *Numer. Math.* **36**, 291–307.

J. A. George and M. T. Heath (1980), 'Solution of sparse linear least squares problems using Givens rotations', *Linear Algebra Appl.* **34**, 69–83.

J. R. Gilbert, C. Moler and R. Schreiber (1992), 'Sparse matrices in MATLAB: Design and implementation', *SIAM J. Matrix. Anal. Appl.* **13**, 333–356.

J. R. Gilbert, E. G. Ng and B. W. Peyton (1997), 'Separators and structure prediction in sparse orthogonal factorization', *Linear Algebra Appl.* **262**, 83–97.

L. Giraud, J. Langou, and M. Rozložnık (2002), On the loss of orthogonality in the Gram–Schmidt orthogonalization process, Technical report TR/PA/02/33, CERFACS, Toulouse, France.

L. Giraud, S. Gratton and J. Langou (2003), A reorthogonalization procedure for modified Gram–Schmidt algorithm based on a rank-k update, Technical Report TR/PA/03/11, CERFACS, Toulouse, France.

I. Gohberg and V. Olshevsky (1994), 'Complexity of multiplication with vectors for structured matrices', *Linear Algebra Appl.* **202**, 163–192.

G. H. Golub (1965), 'Numerical methods for solving least squares problems', *Numer. Math.* **7**, 206–216.

G. H. Golub (1968), 'Least squares, singular values and matrix approximations', *Aplikace Matematiky* **13**, 44–51.

G. H. Golub and W. Kahan (1965), 'Calculating the singular values and pseudo-inverse of a matrix', *SIAM J. Numer. Anal. Ser. B* **2**, 205–224.

G. H. Golub and R. J. Plemmons (1980), 'Large-scale geodetic least-squares adjustment by dissection and orthogonal decomposition', *Linear Algebra Appl.* **34**, 3–28.

G. H. Golub and C. Reinsch (1970), 'Singular value decomposition and least squares solution', *Numer. Math.* **14**, 403–420.

G. H. Golub and C. F. Van Loan (1996), *Matrix Computations*, 3rd edn, Johns Hopkins University Press, Baltimore.

G. H. Golub and J. H. Wilkinson (1966), 'Note on the iterative refinement of least squares solution', *Numer. Math.* **9**, 139–148.

G. H. Golub, V. Klema and G. W. Stewart (1976), Rank degeneracy and least squares problems, Technical Report STAN-CS-76-559, August 1976, Computer Science Department, Stanford University, CA.

G. H. Golub, F. T. Luk and M. Pagano (1979), A sparse least squares problem in photogrammetry, in *Proceedings of the Computer Science and Statistics 12th Annual Symposium on the Interface* (J. F. Gentleman, ed.), University of Waterloo, Canada, pp. 26–30.

G. H. Golub, K. Solna and P. Van Dooren (1995), A QR-like SVD algorithm for a product/quotient of several matrices, in *SVD and Signal Processing, III: Algorithms, Architectures and Applications* (M. Moonen and B. De Moor, eds), Elsevier Science BV, Amsterdam, pp. 139–147.

M. Gu (1998*a*), 'Backward perturbation bounds for linear least squares problems', *SIAM J. Matrix. Anal. Appl.* **20**, 363–372.

M. Gu (1998*b*), 'New fast algorithms for structured linear least squares problems', *SIAM J. Matrix. Anal. Appl.* **20**, 244–269.

M. Gulliksson and P.-Å. Wedin (2000), 'Perturbation theory for generalized and constrained linear least squares', *Numer. Linear Algebra Appl.* **7**, 181–196.

S. Hammarling (1976), The numerical solution of the general Gauss–Markoff linear model, in *Mathematics in Signal Processing* (T. S. Durrani *et al.*, eds), Clarendon Press, Oxford.

M. Hanke (2001), 'On Lanczos methods for the regularization of discrete ill-posed problems', *BIT* **41**, 1008–1018.

P. C. Hansen (1998), *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*, SIAM, Philadelphia.

R. J. Hanson and C. L. Lawson (1969), 'Extensions and applications of the Householder algorithm for solving linear least squares problems', *Math. Comput.* **23**, 787–812.

K. H. Haskell and R. J. Hanson (1981), 'An algorithm for linear least squares problems with equality and nonnegativity constraints', *Math. Program.* **21**, 98–118.

M. T. Heath, A. J. Laub, C. C. Paige and R. C. Ward (1986), 'Computing the SVD of a product of two matrices', *SIAM J. Sci. Statist. Comput.* **7**, 1147–1149.

M. T. Heath, R. J. Plemmons and R. C. Ward (1984), 'Sparse orthogonal schemes for structural optimization using the force method', *SIAM J. Sci. Statist. Comput.* **5**, 514–532.

M. R. Hestenes and E. Stiefel (1952), 'Methods of conjugate gradients for solving linear systems', *J. Res. Nat. Bur. Standards* **B49**, 409–436.

N. J. Higham (1995), *Accuracy and Stability of Numerical Algorithms*, 1st edn, SIAM, Philadelphia, PA.

N. J. Higham (1997), Recent developments in dense numerical linear algebra, in *The State of the Art in Numerical Analysis* (I. S. Duff and G. A. Watson, eds), Oxford University Press, pp. 1–26.

N. J. Higham (2002), *Accuracy and Stability of Numerical Algorithms*, 2nd edn, SIAM, Philadelphia, PA.

W. Hoffman (1989), 'Iterative algorithms for Gram–Schmidt orthogonalization', *Computing* **41**, 335–348.

H. P. Hong and C. T. Pan (1992), 'Rank-revealing QR factorization and the singular value decomposition', *Math. Comput.* **58**, 213–232.

D. A. Huckaby and T. F. Chan (2003), 'On the convergence of Stewart's QLP algorithm for approximating the SVD', *Numer. Algorithms* **32**, 287–316.

R. Karlsson and B. Waldén (1997), 'Estimation of optimal backward perturbation bounds for the linear least squares problem', *BIT* **37**, 862–869.

L. Kaufman (1984), 'Banded eigenvalue solvers on vector machines', *ACM Trans. Math. Software* **10**, 73–86.

C. L. Lawson and R. J. Hanson (1995), *Solving Least Squares Problems*, Classics in Applied Mathematics, SIAM, Philadelphia. Revised republication of work first published by Prentice-Hall, Englewood Cliffs, NJ (1974).

P. Lötstedt (1983), 'Perturbation bounds for the linear least squares problem subject to linear inequality constraints', *BIT* **23**, 500–519.

P. Lötstedt (1984), 'Solving the minimal least squares problem subject to bounds on the variables', *BIT* **24**, 206–224.

S.-M. Lu and J. L. Barlow (1996), 'Multifrontal computation with the orthogonal factors of sparse matrices', *SIAM J. Matrix Anal. Appl.* **17**, 658–679.

P. Manneback, C. Murigande and P. L. Toint (1985), 'A modification of an algorithm by Golub and Plemmons for large linear least squares in the context of Doppler positioning', *IMA J. Numer. Anal.* **5**, 221–234.

R. Mathias and G. W. Stewart (1993), 'A block QR algorithm and the singular value decompositions', *Linear Algebra Appl.* **182**, 91–100.

P. Matstoms (1994), 'Sparse QR factorization in MATLAB', *ACM Trans. Math. Software* **20**, 136–159.

P. Matstoms (1995), 'Parallel sparse QR factorization on shared memory architectures', *Parallel Comput.* **21**, 473–486.

M. Moonen, P. Van Dooren and J. Vandewalle (1992), 'An SVD updating algorithm for subspace tracking', *SIAM J. Matrix Anal. Appl.* **13**, 1015–1038.

J. J. Moré (1978), The Levenberg–Marquardt algorithm: Implementation and theory, in *Numerical Analysis: Proceedings Biennial Conference Dundee 1977* (G. A. Watson, ed.), Vol. 630 of *Lecture Notes in Mathematics*, Springer, Berlin, pp. 105–116.

C. C. Paige (1986), 'Computing the generalized singular value decomposition', *SIAM J. Sci. Statist. Comput.* **7**, 1126–1146.

C. C. Paige (1990), Some aspects of generalized QR factorizations, in *Reliable Numerical Computation* (M. G. Cox and S. J. Hammarling, eds), Clarendon Press, Oxford, pp. 71–91.

C. C. Paige (2002), Unifying least squares, total least squares and data least squares, in *Total Least Squares and Errors-in-Variables Modeling* (S. V. Huffel and P. Lemmerling, eds), Kluwer Academic Publishers, Dordrecht, pp. 25–34.

C. C. Paige and M. A. Saunders (1981), 'Toward a generalized singular value decomposition', *SIAM J. Numer. Anal.* **18**, 398–405.

C. C. Paige and M. A. Saunders (1982*a*), 'Algorithm 583 LSQR: Sparse linear equations and sparse least squares', *ACM Trans. Math. Software* **8**, 195–209.

C. C. Paige and M. A. Saunders (1982*b*), 'LSQR: An algorithm for sparse linear equations and sparse least squares', *ACM Trans. Math. Software* **8**, 43–71.

C. C. Paige and Z. Strakoš (2002), 'Scaled total least squares fundamentals', *Numer. Math.* **91**, 117–146.

C. T. Pan and P. T. P. Tang (1999), 'Bounds on singular values revealed by QR factorizations', *BIT* **39**, 740–756.

D. J. Pierce and J. G. Lewis (1997), 'Sparse multifrontal rank revealing QR factorization', *SIAM J. Matrix Anal. Appl.* **18**, 159–181.

G. Quintana-Ortí, X. Sun and C. H. Bischof (1998), 'A BLAS-3 version of the QR factorizations with column pivoting', *SIAM J. Sci. Comput.* **19**, 1486–1494.

J. K. Reid (1967), 'A note on the least squares solution of a band system of linear equations by Householder reductions', *Comput. J.* **10**, 188–189.

C. H. Reinsch (1971), 'Smoothing by spline functions', *Numer. Math.* **16**, 451–454.

M. Rojas and D. C. Sorensen (2002), 'A trust region approach to the regularization of large-scale discrete forms of ill-posed problems', *SIAM J. Sci. Comput.* **23**, 1843–1861.

K. Schittkowski (1983), 'The numerical solution of constrained linear least-squares problems', *IMA J. Numer. Anal.* **3**, 11–36.

R. Schreiber and C. F. Van Loan (1989), 'A storage efficient WY representation for products of Householder transformations', *SIAM J. Sci. Statist. Comput.* **10**, 53–57.

H. R. Schwarz (1968), 'Tridiagonalization of a symmetric bandmatrix', *Numer. Math.* **12**, 231–241.

G. W. Stewart (1991), On an algorithm for refining a rank-revealing URV factorization and a perturbation theorem for singular values, Technical Report UMIACS-TR-91-38, CS-TR-2626, Department of Computer Science, University of Maryland, College Park, MD.

G. W. Stewart (1992), 'An updating algorithm for subspace tracking', *IEEE Trans. Signal Process.* **40**, 1535–1541.

G. W. Stewart (1993), 'Updating a rank-revealing ULV decomposition', *SIAM J. Matrix Anal. Appl.* **14**, 494–499.

G. W. Stewart (1999), 'The QLP approximation to the singular value decomposition', *SIAM J. Sci. Comput.* **20**, 1336–1348.

E. Stiefel (1952/53), 'Ausgleichung ohne Aufstellung der Gausschen Normalgleichungen', *Wiss. Z. Tech. Hochsch. Dresden* **2**, 441–442.

C. Sun (1996), 'Parallel sparse orthogonal factorization on distributed-memory multiprocessors', *SIAM J. Sci. Comput.* **17**, 666–685.

J.-G. Sun and Z. Sun (1997), 'Optimal backward perturbation bounds for underdetermined systems', *SIAM J. Matrix Anal. Appl.* **18**, 393–402.

A. N. Tikhonov (1963), 'Regularization of incorrectly posed problems', *Soviet Math.* **4**, 1624–1627.

L. N. Trefethen and D. Bau, III (1997), *Numerical Linear Algebra*, SIAM, Philadelphia.

S. Van Huffel and J. Vandewalle (1991), *The Total Least Squares Problem: Computational Aspects and Analysis*, Vol. 9 of *Frontiers in Applied Mathematics*, SIAM, Philadelphia.

C. F. Van Loan (1976), 'Generalizing the singular value decomposition', *SIAM J. Numer. Anal.* **13**, 76–83.

B. Waldén, R. Karlsson and J.-G. Sun (1995), 'Optimal backward perturbation bounds for the linear least squares problem', *Numer. Linear Algebra Appl.* **2**, 271–286.

J. H. Wilkinson and C. Reinsch (1971), *Handbook for Automatic Computation, Vol. II: Linear Algebra*, Springer, New York.

S. Wold, A. Ruhe, H. Wold and W. J. Dunn (1984), 'The collinearity problem in linear regression: The partial least squares (PLS) approach to generalized inverses', *SIAM J. Sci. Statist. Comput.* **5**, 735–743.