Chapter 8

Linear Least Squares Problems

Of all the principles that can be proposed, I think there is none more general, more exact, and more easy of application than that which consists of rendering the sum of squares of the errors a minimum.
—Adrien Maria Legendre, Nouvelles méthodes pour la détermination des orbites des cométes. Paris 1805

8.1 Preliminaries

8.1.1 The Least Squares Principle

A fundamental task in scientific computing is to estimate parameters in a mathematical model from collected data which are subject to errors. The influence of the errors can be reduced by using a greater number of data than the number of unknowns. If the model is linear, the resulting problem is then to "solve" an in general inconsistent linear system $Ax = b$, where $A \in \mathbf{R}^{m\times n}$ and $m \geq n$. In other words, we want to find a vector $x \in \mathbf{R}^n$ such that Ax is in some sense the "best" approximation to the known vector $b \in \mathbf{R}^m$.

There are many possible ways of defining the "best" solution to an inconsistent linear system. A choice which can often be motivated for statistical reasons (see Theorem 8.1.6) and leads also to a simple computational problem is the following: Let x be a vector which minimizes the Euclidian length of the **residual vector** $r = b - Ax$; i.e., a solution to the minimization problem

$$\min_x \|Ax - b\|_2, \tag{8.1.1}$$

where $\|\cdot\|_2$ denotes the Euclidian vector norm. Note that this problem is equivalent to minimizing the sum of squares of the residuals $\sum_{i=1}^m r_i^2$. Hence, we call (8.1.1) a **linear least squares problem** and any minimizer x a **least squares solution** of the system $Ax = b$.

Example 8.1.1.

Consider a model described by a scalar function $y(t) = f(x,t)$, where $x \in \mathbf{R}^n$ is a parameter vector to be determined from measurements (y_i, t_i), $i = 1 : m$, $m > n$. In particular, let $f(x,t)$ be *linear* in x,

$$f(x,t) = \sum_{j=1}^{n} x_j \phi_j(t).$$

Then the equations $y_i = \sum_{j=1}^{n} x_j \phi_j(t_i)$, $i = 1 : m$ form an overdetermined system, which can be written in matrix form $Ax = b$, where $a_{ij} = \phi_j(t_i)$, and $b_i = y_i$.

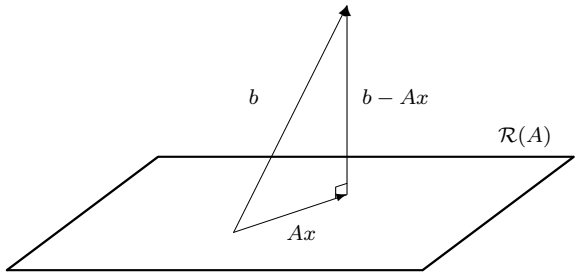

Figure 8.1.1. *Geometric characterization of the least squares solution.*

We shall see that a least squares solution x is characterized by $r \perp \mathcal{R}(A)$, where $\mathcal{R}(A)$ the range space of A. The residual vector r is always uniquely determined and the solution x is unique if and only if $\operatorname{rank}(A) = n$, i.e., when A has linearly independent columns. If $\operatorname{rank}(A) < n$, we seek the unique least squares solution of minimum Euclidean norm.

We now show a necessary condition for a vector x to minimize $\|b - Ax\|_2$.

Theorem 8.1.1.

Given the matrix $A \in \mathbf{R}^{m\times n}$ and a vector $b \in \mathbf{R}^m$. The vector x minimizes $\|b - Ax\|_2$ if and only if the residual vector $r = b - Ax$ is orthogonal to $\mathcal{R}(A)$, i.e. $A^T(b - Ax) = 0$, or equivalently x satisfies the **normal equations**.

$$A^T A x = A^T b \tag{8.1.2}$$

Proof. Let x be a vector for which $A^T(b - Ax) = 0$. Then for any $y \in \mathbf{R}^n$ $b - Ay = (b - Ax) + A(x - y)$. Squaring this and using (8.1.2) we obtain

$$\|b - Ay\|_2^2 = \|b - Ax\|_2^2 + \|A(x - y)\|_2^2 \geq \|b - Ax\|_2^2.$$

On the other hand assume that $A^T(b - Ax) = z \neq 0$. Then if $x - y = -\epsilon z$ we have for sufficiently small $\epsilon \neq 0$,

$$\|b - Ay\|_2^2 = \|b - Ax\|_2^2 - 2\epsilon\|z\|_2^2 + \epsilon^2\|Az\|_2^2 < \|b - Ax\|_2^2$$

so x does not minimize $\|b - Ax\|_2$. □

The matrix $A^TA \in \mathbf{R}^{n\times n}$ is symmetric and positive semidefinite since

$$x^TA^TAx = \|Ax\|_2^2 \geq 0.$$

The normal equations $A^TAx = A^Tb$ are always *consistent* since $A^Tb \in \mathcal{R}(A^T) = \mathcal{R}(A^TA)$ and therefore a least squares solution always exists. Any solution to the normal equations is a least squares solution.

By Theorem 8.1.1 any least squares solution x will decompose the right hand side b into two orthogonal components

$$b = Ax + r, \quad r \perp Ax. \tag{8.1.3}$$

Here $Ax = b - r = P_{\mathcal{R}(A)}b$ is the orthogonal projection of b (see Section 8.1.3) onto $\mathcal{R}(A)$ and $r \in \mathcal{N}(A^T)$ (cf. Figure 8.1.1). Note that although the least squares solution x may not be unique the decomposition in (8.1.3) always is unique.

We now introduce a related problem. Suppose that the vector $y \in \mathbf{R}^m$ is required to satisfy exactly $n < m$ linearly independent equations $A^Ty = c$. We want to find the **minimum norm solution**, i.e. to solve the problem

$$\min \|y\|_2 \quad \text{subject to} \quad A^Ty = c. \tag{8.1.4}$$

Let y be any solution of $A^Ty = c$, and write $y = y_1 + y_2$, where $y_1 \in \mathcal{R}(A)$. $y_2 \in \mathcal{N}(A^T)$. Then $A^Ty_2 = 0$ and hence y_1 is also a solution. Since $y_1 \perp y_2$ we have

$$\|y_1\|_2^2 = \|y\|_2^2 - \|y_2\|_2^2 \leq \|y\|_2^2,$$

with equality only if $y_2 = 0$. This shows that the minimum norm solution lies in $\mathcal{R}(A)$, i.e., $y = Az$ for some $z \in \mathbf{R}^n$, Substituting this in (8.1.4) gives the normal equations $A^TAz = c$, Since A has full column rank the matrix A^TA is nonsingular and the solution is given by

$$y = A(A^TA)^{-1}c \tag{8.1.5}$$

A slightly more general problem is the **conditional least squares** problem

$$\min_y \|y - b\|_2 \quad \text{subject to} \quad A^Ty = c. \tag{8.1.6}$$

By a similar argument as used above the solution satisfies $y - b \in \mathcal{R}(A)$. Setting $y = b - Az$, and substituting in $A^Ty = c$ we find that z satisfies the equation

$$A^TAz = A^Tb - c. \tag{8.1.7}$$

Hence, the unique solution to problem (8.1.6) is

$$y = (I - A(A^TA)^{-1}A^T)b + A(A^TA)^{-1}c. \tag{8.1.8}$$

where $P_{\mathcal{N}(A^T)} = I - A(A^TA)^{-1}A^T$ is the orthogonal projection onto $\mathcal{N}(A^T)$.

Example 8.1.2.

The height $h_k = h(t_k)$ of a falling body is measured at times $t_k = t_0 + k\Delta t$, $k = 1 : m$. The adjusted values $\hat{h}_k = h_k - y_k$ should lie on a parabola, that is, the third differences must vanish, This leads to the problem minimizing

$$\min_y \|y - h\|_2 \quad \text{subject to} \quad A^T y = 0$$

where $(m = 7)$

$$A^T = \begin{pmatrix} 1 & -3 & 3 & -1 & 0 & 0 & 0 \\ 0 & 1 & -3 & 3 & -1 & 0 & 0 \\ 0 & 0 & 1 & -3 & 3 & -1 & 0 \\ 0 & 0 & 0 & 1 & -3 & 3 & -1 \end{pmatrix}.$$

which is a conditional least squares problem.

The solution to the standard linear least squares problem $\min_x \|Ax - b\|_2$ is characterized by the two conditions $A^T r = 0$ and $r = b - Ax$. These are $n + m$ equations

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{8.1.9}$$

for the unknowns x and the residual r. This special case of is often called the **augmented system** for the least squares problem. The following theorem gives a unified formulation of the least squares and conditional least squares problems in terms of an augmented system.

Theorem 8.1.2.

Let the matrix if $A \in \mathbf{R}^{m\times n}$ have full column rank and consider the symmetric linear system

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}, \tag{8.1.10}$$

Then the system is nonsingular and gives the first order conditions for the following two optimization problems:

1. *Linear least squares problems*

$$\min_x \tfrac{1}{2}\|Ax - b\|_2^2 + c^T x. \tag{8.1.11}$$

2. *Conditional least squares problem*

$$\min_r \tfrac{1}{2}\|y - b\|_2, \quad \textit{subject to} \quad A^T y = c, \tag{8.1.12}$$

Proof. The system (8.1.10) can be obtained by differentiating (8.1.11) to give

$$A^T(Ax - b) + c = 0,$$

and setting $y = r = b - Ax$.

The system can also be obtained by differentiating the Lagrangian

$$L(x, y) = \frac{1}{2}y^T y - y^T b + x^T(A^T y - c)$$

of (8.1.12), and equating to zero. Here x is the vector of Lagrange multipliers . □

The augmented system plays a key role in the perturbation analysis of least squares problems (Section 8.2.3) as well as in the iterative refinement of least squares solutions (Section 8.3.7)

8.1.2 The Gauss–Markov Model

Gauss claims he discovered the method of least squares in 1795. He used it for analyzing surveying data and for astronomical calculation. A famous example is when Gauss successfully predicted the orbit of the asteroid Ceres in 1801.

Gauss [232] in 1821 put the method of least squares on a sound theoretical basis. To describe his results we first need to introduce some concepts from statistics. Let the probability that random variable $y \leq x$ be equal to $F(x)$, where $F(x)$ is nondecreasing, right continuous, and satisfies

$$0 \leq F(x) \leq 1, \quad F(-\infty) = 0, \quad F(\infty) = 1.$$

Then $F(x)$ is called the **distribution function** for y.

The **expected value** and the **variance** of y are defined as the Stieltjes integrals

$$\mathcal{E}(y) = \mu = \int_{-\infty}^{\infty} y dF(y), \qquad \mathcal{E}(y-\mu)^2 = \sigma^2 = \int_{-\infty}^{\infty} (y-\mu)^2 dF(y),$$

If $y = (y_1, \ldots, y_n)^T$ is a vector of random variables and $\mu = (\mu_1, \ldots, \mu_n)^T$, $\mu_i = \mathcal{E}(y_i)$, then we write $\mu = \mathcal{E}(y)$. If y_i and y_j have the joint distribution $F(y_i, y_j)$ the **covariance** between y_i and y_j is

$$\begin{aligned}\sigma_{ij} &= \mathcal{E}[(y_i - \mu_i)(y_j - \mu_j)] = \int_{-\infty}^{\infty} (y_i - \mu_i)(y_j - \mu_j) dF(y_i, y_j) \\ &= \mathcal{E}(y_i y_j) - \mu_i \mu_j.\end{aligned}$$

The covariance matrix $V \in \mathbf{R}^{n\times n}$ of y is defined by

$$V = \mathcal{V}(y) = \mathcal{E}[(y-\mu)(y-\mu)^T] = \mathcal{E}(yy^T) - \mu\mu^T.$$

where the diagonal element σii is the variance of y_i.

Definition 8.1.3.

Let $A \in \mathbf{R}^{m\times n}$ be a known matrix, $b \in \mathbf{R}^m$ a vector of observations, and $x \in \mathbf{R}^n$ an unknown parameter vector. The **Gauss–Markov model** *is a linear statistical model, where it is assumed that a linear relationship*

$$Ax = b + e, \quad \mathcal{E}(e) = 0, \quad \mathcal{V}(e) = \sigma^2 W, \tag{8.1.13}$$

holds.[28] *Here ϵ is a vector of random errors, $W \in \mathbf{R}^{m\times n}$ a symmetric nonnegative definite matrix and σ^2. an unknown constant. In the* **standard case** *the errors are assumed to be independently and identically distributed, i.e., $W = I_m$.*

We now prove some properties which will be useful in the following.

Lemma 8.1.4.

Let $B \in \mathbf{R}^{r\times n}$ be a matrix and y a random vector with $\mathcal{E}(y) = \mu$ and covariance matrix V. Then the expected value and covariance matrix of By is

$$\mathcal{E}(By) = B\mu, \qquad \mathcal{V}(By) = BVB^T. \tag{8.1.14}$$

In the special case that $B = b^T$ is a row vector $\mathcal{V}(b^T y) = \mu\|b\|_2^2$.

Proof. The first property follows directly from the definition of expected value. The second follows from the relation

$$\begin{aligned}\mathcal{V}(By) &= \mathcal{E}[(B(y-\mu)(y-\mu)^T B^T] \\ &= B\mathcal{E}[(y-\mu)(y-\mu)^T]B^T = BVB^T.\end{aligned}$$

□

We make the following definitions:

Definition 8.1.5.

Let $g = c^T y$, where c is a constant vector, be a linear function of the random vector y. Then $c^T y$ is an **unbiased estimate** *of the parameter θ if $\mathcal{E}(c^T y) = 0$. When such a function exists, θ is called an* **estimable parameter**. *Further, it is a minimum variance (best) linear unbiased estimate of θ if $\mathcal{V}(g)$ is minimized over all such linear estimators. .*

Theorem 8.1.6 (*The Gauss–Markov Theorem***).**

Consider the linear model (8.1.13) *with covariance matrix $\sigma^2 I$. Let $\hat{x}$ be the least square estimator, obtained by minimizing over x the sum of squares $\|Ax-b\|_2^2$. Then the best linear unbiased estimator of any linear functional $\theta = c^T x$ is $c^T\hat{x}$. The covariance matrix of the estimate $\hat{x}$ equals*

$$\mathcal{V}(\hat{x}) = V = \sigma^2(A^T A)^{-1} \tag{8.1.15}$$

[28] In statistical literature the Gauss–Markov model is written $X\beta = y + e$. We choose another notation in order to be consistent throughout the book.

Furthermore, the residual vector $r = b - A\hat{x}$ is uncorrelated with $\hat{x}$ and the quadratic form

$$s^2 = \frac{1}{m-n}\hat{r}^T\hat{r} \tag{8.1.16}$$

is an unbiased estimate of σ^2, that is, $\mathcal{E}(s^2) = \sigma^2$.

Proof. If we set $\hat{b} = b + e$, then $\mathcal{E}(\hat{b}) = b = Ax$ and $\mathcal{V}(\hat{b}) = \sigma^2 I$. Consider the estimate $\hat{\theta} = d^T\hat{b}$ of the linear functional $\theta = c^T x$. Since θ is unbiased, we have

$$\mathcal{E}(\hat{\theta}) = d^T\mathcal{E}(\hat{b}) = d^T Ax = c^T x,$$

which shows that $A^T d = c$. From Lemma 8.1.4 it follows that $\mathcal{V}(\hat{\theta}) = \sigma^2 d^T d$. Thus, we wish to minimize $d^T d$ subject to $A^T d = c$. Using the method of Lagrange multipliers, let

$$Q = d^T d - 2\lambda^T(A^T d - c),$$

where λ is a vector of Lagrange multipliers. A necessary condition for Q to be a minimum is that

$$\frac{\partial Q}{\partial d} = 2(d^T - \lambda^T A^T) = 0,$$

or $d = A\lambda$. Premultiplying this by A^T results in $A^T A\lambda = A^T d = c$ and since $A^T A$ is nonsingular this gives $d = A\lambda = A(A^T A)^{-1}c$. This gives

$$\hat{\theta} = d^T\hat{b} = c^T(A^T A)^{-1}A^T\hat{b} = c^T\hat{x}$$

where $\hat{x}$ is the solution to the normal equations. But the solution of the normal equations minimizes the sum of squares $(b - Ax)^T(b - Ax)$ with respect to x. ∎

Remark 8.1.1. In the literature Gauss–Markov theorem is sometimes stated in less general forms. It is important to note that in the theorem errors are *not* assumed to be normally distributed, nor are they assumed to be independent (only uncorrelated—a weaker condition). They are also *not* assumed to be identically distributed, but only having zero mean and the same variance.

Remark 8.1.2. It is fairly straight-forward to generalize the Gauss–Markov theorem to the complex case. The normal equations then become

$$A^H Ax = A^H b.$$

This has applications in complex stochastic processes; see Miller [438].

If $\operatorname{rank}(A) < n$, then the normal equations are singular but consistent. In this case a linear functional $c^T x$ is estimable if and only if

$$c \in \mathcal{R}(A^T).$$

The residual vector $\hat{r} = \hat{b} - Ax$ of the least squares solution satisfies $A^T\hat{r} = 0$, i.e. $\hat{r}$ is orthogonal to the column space of A. This condition gives n linear relations among the m components of $\hat{r}$. It can be shown that the residuals $\hat{r}$ and therefore, also s^2 are uncorrelated with $\hat{x}$, i.e.,

$$\mathcal{V}(\hat{r}, \hat{x}) = 0, \qquad \mathcal{V}(s^2, \hat{x}) = 0.$$

An estimate of the covariance of the linear functional $c^T x$ is given by $s^2(c^T(A^TA)^{-1}c)$. In particular, for the components $x_i = e_i^T x$,

$$s^2(e_i^T(A^TA)^{-1}e_i) = s^2(A^TA)^{-1}_{ii}.$$

the ith diagonal element of $(A^TA)^{-1}$.

It is often the case that the errors have a positive definite covariance matrix different from $\sigma^2 I$. The above results are easily modified to cover this case.

Theorem 8.1.7.

Consider a linear model with the error covariance matrix equal to the symmetric positive definite matrix $\mathcal{V}(e) = \sigma^2 V$. If A has full column rank, then the best unbiased linear estimate is given by the solution to

$$\min_x (Ax - b)^T V^{-1}(Ax - b). \tag{8.1.17}$$

The covariance matrix of the estimate $\hat{x}$ is

$$\mathcal{V}(\hat{x}) = \sigma^2(A^TV^{-1}A)^{-1} \tag{8.1.18}$$

and

$$s^2 = \frac{1}{m-n}(b - A\hat{x})^T V^{-1}(b - A\hat{x}), \tag{8.1.19}$$

is an unbiased estimate of σ

A situation that often occurs is that the error covariance is a diagonal matrix

$$V = \operatorname{diag}(v_{11}, v_{22}, \ldots, v_{mm})$$

Then (8.1.17) is called a **weighted least squares** problem. It can easily be transformed to the standard case by scaling the ith equation by $1/\sqrt{v_{ii}}$. Note that the smaller the variance the larger weight should be given to a particular equation. It is important to note that different scalings will give different solutions, unless the system is $Ax = b$ is consistent.

In the general Gauss-Markov model no assumption is made on the dimension or rank of A or the rank of the covariance matrix except that A and W have the same number of rows. Assume that $\operatorname{rank}(W) = k \le n$ and given in factored form

$$W = BB^T, \quad B \in \mathbf{R}^{m\times k} \tag{8.1.20}$$

If W is initially given, the B can be computed as the Cholesky factor of W. Then the Guass–Markov model can be replaced by the equivalent model

$$Ax = b + Bu, \quad \mathcal{V}(u) = \sigma^2 I. \tag{8.1.21}$$

The best linear estimate of x is a solution to the **constrained** linear least squares problem.

$$\min_{x,u} u^T u \quad \text{subject to} \quad b = Ax + Bu \tag{8.1.22}$$

Here we must require that the consistency condition

$$b \in \mathcal{R}(A,\ B)$$

is satisfied. If this does not hold, then b could not have come from the linear model (8.1.21). The solution $\hat{x}$ to (8.1.22) may not be unique. In this case we should take $\hat{x}$ to be the solution of minimum norm. For a full analysis of the general models we refer to Korouklis and Paige [385]. Solution methods for **constrained** linear least squares problem are treated in Section 8.6.3.

8.1.3 Orthogonal and Oblique Projections

We have seen that the least squares solution is characterized by the property that its residual is orthogonal to its projection onto $\mathcal{R}(A)$. In this section make a systematic study of both orthogonal and more general projection matrices.

Any matrix $P \in \mathbf{C}^{n\times n}$ such that $P^2 = P$ is called **idempotent** and a **projector**. An arbitrary vector $v \in \mathbf{C}^n$ is decomposed in a unique way as

$$v = Pv + (I - P)v = v_1 + v_2. \tag{8.1.23}$$

Here $v_1 = Pv$ is a projection of v onto $\mathcal{R}(P)$, the column space of P. Since $Pv_2 = (P - P^2)v = 0$ it follows that $(I - P)$ is a projection onto $\mathcal{N}(P)$, the null space of P.

If λ is an eigenvalue of a projector P, then from $P^2 = P$ it follows that $\lambda^2 = \lambda$. Hence, the eigenvalues of P are either 1 or 0 and $k = \text{trace}\,(P)$ is the rank of P.

If P is Hermitian, $P^H = P$, then

$$v_1^H v_2 = (Pv)^H(I - P)v = v^H P(I - P)v = v^H(P - P^2)v = 0.$$

In this case v_2 lies in the orthogonal complement of $\mathcal{R}(P)$; and P is an **orthogonal projector**.

It can be shown that the orthogonal projector P onto a given subspace $\mathcal{S}$ is unique, see Problem 8.1.1. The following property follows immediately from the Pythagorean theorem.

Lemma 8.1.8.

Let the orthogonal projection of vector $x \in \mathbf{C}^n$ onto a subspace $\mathcal{S} \subset \mathbf{C}^n$ be $z = Px \in \mathcal{S}$. Then z is the the point in $\mathcal{S}$ closest to x.

Let P be an orthogonal projector onto $\mathcal{S}$ and U_1 a unitary basis for $\mathcal{S}$. Then we can always find a unitary basis U_2 for the orthogonal complement of $\mathcal{S}$. Then $U = (U_1\ U_2)$ is unitary and $U^H U = UU^H = I$, and P can be expressed in the form

$$P = U_1 U_1^H, \qquad I - P = U_2 U_2^H; \tag{8.1.24}$$

For an orthogonal projector we have

$$\|Pv\|_2 = \|U_1^H v\|_2 \le \|v\|_2 \quad \forall \quad v \in \mathbf{C}^m, \tag{8.1.25}$$

where equality holds for all vectors in $\mathcal{R}(U_1)$ and thus $\|P\|_2 = 1$. The converse is also true; P is an orthogonal projection only if (8.1.25) holds.

A projector P such that $P \neq P^H$ is called an **oblique projector**. We can write the spectral decomposition

$$P = \begin{pmatrix} X_1 & X_2 \end{pmatrix} \begin{pmatrix} I_k & 0 \\ 0 & 0_{n-k} \end{pmatrix} \begin{pmatrix} \hat{Y}_1^H \\ \hat{Y}_2^H \end{pmatrix} = X_1 \hat{Y}_1^H. \tag{8.1.26}$$

where

$$\begin{pmatrix} \hat{Y}_1^H \\ \hat{Y}_2^H \end{pmatrix} (X_1 \ X_2) = \begin{pmatrix} \hat{Y}_1^H X_1 & \hat{Y}_1^H X_2 \\ \hat{Y}_2^H X_1 & \hat{Y}_2^H X_2 \end{pmatrix} = \begin{pmatrix} I_k & 0 \\ 0 & I_{n-k} \end{pmatrix}. \tag{8.1.27}$$

In particular, $\hat{Y}_1^H X_2 = 0$ and $\hat{Y}_2^H X_1 = 0$. Hence, the columns of X_2 form a basis for the orthogonal complement of $\mathcal{R}(\hat{Y}_1)$ and, similarly, the columns of $\hat{Y}_2$ form a basis for the orthogonal complement of $\mathcal{R}(X_1)$.

In terms of this spectral decomposition $I - P = X_2 \hat{Y}_2^H$ and the splitting (8.1.23) can be written

$$v = X_1(\hat{Y}_1^H v) + X_2(\hat{Y}_2^H v) = v_1 + v_2. \tag{8.1.28}$$

Here v_1 is the oblique projection of v onto $\mathcal{R}(P)$ along $\mathcal{N}(P)$.

Let Y_1 be an orthogonal matrix whose columns span $\mathcal{R}(\hat{Y}_1)$. Then there is a nonsingular matrix G_1 such that $\hat{Y}_1 = Y_1 G$. From (8.1.27) it follows that $G^H Y_1^H X_1 = I_k$, and hence $G^H = (Y_1^H X_1)^{-1}$. Similarly, $Y_2 = (Y_2^H X_2)^{-1} Y_2$ is an orthogonal matrix whose columns span $\mathcal{R}(\hat{Y}_2)$. Hence, we can write

$$P = X_1(Y_1^H X_1)^{-1} Y_1^H, \qquad I - P = X_2(Y_2^H X_2)^{-1} Y_2^H. \tag{8.1.29}$$

This shows that $\|P\|$ can be large when the matrix $Y_1^H X_1$ is ill-conditioned.

Example 8.1.3.

We illustrate the case when $n = 2$ and $n_1 = 1$. Let the vectors x_1 and y_1 be normalized so that $\|x_1\|_2 = \|y_1\|_2 = 1$ and let $y_1^H x_1 = \cos\theta$, where θ is the angle between x_1 and y_1, Since

$$P = x_1(y_1^H x_1)^{-1} y_1^H = \frac{1}{\cos\theta} x_1 y_1^H.$$

Hence, $\|P\|_2 = 1/\cos\theta \ge 1$, and $\|P\|_2$ becomes very large when y_1 is almost orthogonal to x_1. When $y_1 = x_1$ we have $\theta = 0$ and P is an orthogonal projection.

8.1.4 Generalized Inverses and the SVD

The SVD introduced in Section 7.1.5 is a powerful tool both for analyzing and solving linear least squares problems. The reason for this is that the orthogonal matrices that transform A to diagonal form do not change the l_2-norm. We have the following fundamental result.

Theorem 8.1.9.
Let $A \in \mathbf{R}^{m\times n}$ have the singular value decomposition

$$A = (\,U_1 \quad U_2\,)\begin{pmatrix}\Sigma_1 & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}V_1^T\\ V_2^T\end{pmatrix}, \tag{8.1.30}$$

where U_1 and V_1 have $r = \operatorname{rank}(A)$ columns. The least squares problem

$$\min_{x\in S}\|x\|_2, \qquad S = \{x \in \mathbf{R}^n |\ \|b - Ax\|_2 = \min\}. \tag{8.1.31}$$

always has a unique solution, which can be written as

$$x = V_1\Sigma_1^{-1}U_1^T b. \tag{8.1.32}$$

Proof. Using the orthogonal invariance of the l_2 norm we have

$$\begin{aligned}\|b - Ax\|_2 &= \|U^T(b - AVV^T x)\|_2\\ &= \left\|\begin{pmatrix}c_1\\ c_2\end{pmatrix} - \begin{pmatrix}\Sigma_1 & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}z_1\\ z_2\end{pmatrix}\right\|_2 = \left\|\begin{pmatrix}c_1 - \Sigma_1 z_1\\ c_2\end{pmatrix}\right\|_2.\end{aligned}$$

where $z_1, c_1 \in \mathbf{R}^r$ and

$$c = U^T b = \begin{pmatrix}c_1\\ c_2\end{pmatrix}.$$

The residual norm will attain its minimum value equal to $\|c_2\|_2$ for $z_1 = \Sigma_1^{-1}c_1$, z_2 arbitrary. Obviously the choice $z_2 = 0$ minimizes $\|x\|_2 = \|Vz\|_2 = \|z\|_2$. ∎

Note that problem (8.1.31) includes as special cases the solution of both overdetermined and underdetermined linear systems. We set

$$A^\dagger = (\,V_1 \quad V_2\,)\begin{pmatrix}\Sigma_1^{-1} & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}U_1^T\\ U_2^T\end{pmatrix} = V_1\Sigma_1^{-1}U_1^T \in \mathbf{R}^{n\times m} \tag{8.1.33}$$

We call $A^\dagger$ the **pseudo-inverse** of A and $x = A^\dagger b$ the pseudo-inverse solution of $Ax = b$. The pseudo-inverse solution (8.1.33) can also be written

$$x = \sum_{i=1}^{r}\frac{u_i^T b}{\sigma_i}\cdot v_i. \tag{8.1.34}$$

The following two cases are important special cases:

- In an overdetermined case where A has full column rank ($r = n$) the submatrix V_2 is empty and the pseudo-inverse becomes

$$A^\dagger = V\Sigma_1^{-1}U_1^T. \tag{8.1.35}$$

- In an underdetermined case where A has full row rank ($r = m$) the submatrix U_2 is empty and the pseudo-inverse becomes

$$A^\dagger = V_1 \Sigma_1^{-1} U^T. \tag{8.1.36}$$

Note that for computing the pseudo-inverse solution we only need to compute the "thin" SVD, i.e. the nonzero singular values, the matrix V_1 and the vector $U_1^T b$. Methods for computing the SVD are described in Section 10.5.3 and Section 10.6.4.

The matrix $A^\dagger$ is often called the **Moore–Penrose inverse**. Moore [443] developed the concept of the general reciprocal, which was rediscovered by Bjerhammar [55]. Penrose [1955], gave an elegant algebraic characterization and showed that $X = A^\dagger$ is uniquely determined by the four **Penrose conditions** :

$$(1) \quad AXA = A, \qquad (2) \quad XAX = X, \tag{8.1.37}$$

$$(3) \quad (AX)^T = AX, \qquad (4) \quad (XA)^T = XA. \tag{8.1.38}$$

It can be directly verified that $X = A^\dagger$ given by (8.1.33) satisfies these four conditions. In particular, this shows that $A^\dagger$ does not depend on the particular choices of U and V in the SVD. (See also Problem 8.1.2.) The following properties of the pseudoinverse easily follow from (8.1.36).

Theorem 8.1.10.

1. $(A^\dagger)^\dagger = A$; *2.* $(A^\dagger)^H = (A^H)^\dagger$;

3. $(\alpha A)^\dagger = \alpha^\dagger A^\dagger$; *4.* $(A^H A)^\dagger = A^\dagger (A^\dagger)^H$;

5. if U and V are unitary $(UAV^H)^\dagger = VA^\dagger U^H$;

6. if $A = \sum_i A_i$, where $A_i A_j^H = 0$, $A_i^H A_j = 0$, $i \neq j$, then $A^\dagger = \sum_i A_i^\dagger$;

7. if A is normal ($AA^H = A^H A$) then $A^\dagger A = AA^\dagger$ and $(A^n)^\dagger = (A^\dagger)^n$;

8. A, A^H, $A^\dagger$, and $A^\dagger A$ all have rank equal to trace $(A^\dagger A)$.

The orthogonal projections onto the four fundamental subspaces of A have the following simple expressions in terms of the pseudo-inverse :

$$P_{\mathcal{R}(A)} = AA^\dagger = U_1 U_1^T, \qquad P_{\mathcal{N}(A^T)} = I - AA^\dagger == U_2 U_2^T, \tag{8.1.39}$$

$$P_{\mathcal{R}(A^T)} = A^\dagger A = V_1 V_1^T, \qquad P_{\mathcal{N}(A)} = I - A^\dagger A = V_2 V_2^T.$$

These expressions are easily verified using the definition of an orthogonal projection and the Penrose conditions.

Another useful characterization of the pseudo-inverse solution is the following:

Theorem 8.1.11.

The pseudo-inverse solution $x = A^\dagger b$ is uniquely characterized by the two geometrical conditions

$$x \in \mathcal{R}(A^T), \qquad r = b - Ax \in \mathcal{N}(A^T). \tag{8.1.40}$$

Proof. These conditions are easily verified from (8.1.34). □

In the special case that $A \in \mathbf{R}^{m\times n}$ and $\operatorname{rank}(A) = n$ it holds that

$$A^\dagger = (A^TA)^{-1}A^T, \qquad (A^T)^\dagger = A(A^TA)^{-1} \tag{8.1.41}$$

These expressions follow from the normal equations (8.2.2) and (8.1.5). Some properties of the usual inverse can be extended to the pseudo-inverse, e.g., the relations

$$(A^\dagger)^\dagger = A, \qquad (A^T)^\dagger = (A^\dagger)^T,$$

easily follow form (8.1.33). In general $(AB)^\dagger \neq B^\dagger A^\dagger$. The following theorem gives a useful *sufficient* conditions for the relation $(AB)^\dagger = B^\dagger A^\dagger$ to hold.

Theorem 8.1.12.
If $A \in \mathbf{R}^{m\times r}$, $B \in \mathbf{R}^{r\times n}$, and $\operatorname{rank}(A) = \operatorname{rank}(B) = r$, then

$$(AB)^\dagger = B^\dagger A^\dagger = B^T(BB^T)^{-1}(A^TA)^{-1}A^T. \tag{8.1.42}$$

Proof. The last equality follows from (8.1.41). The first equality is verified by showing that the four Penrose conditions are satisfied. □

Any matrix A^- satisfying the first Penrose condition

$$AA^-A = A \tag{8.1.43}$$

is called a **generalized inverse** of A. It is also called an **inner inverse** or $\{1\}$-inverse. If it satisfies the second condition $AA^-A = A^-AA^- = A^-$ it is called an **outer inverse** or a $\{2\}$-inverse.

Let A^- be a $\{1\}$-inverse of A. Then for all b such that the system $Ax = b$ is consistent $x = A^-b$ is a solution. The general solution can be written

$$x = A^-b + (I - A^-A)y, \quad y \in \mathbf{C}^n.$$

We have also

$$(AA^-A^-)^2 = AA^-AA^- = AA^-, \qquad (A^-A)^2 = A^-AA^-A = A^-A.$$

This shows that AA^- and A^-A are idempotent and therefore (in general oblique) projectors

$$AX = P_{\mathcal{R}(A),S}, \qquad XA = P_{T,\mathcal{N}(A)},$$

where S and T are some subspaces complementary to $\mathcal{R}(A)$ and $\mathcal{N}(A)$, respectively.

Let $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$. Then $\|Ax-b\|_2$ is the minimized when x satisfies the normal equations $A^TAx = A^Tb$. Suppose now that a generalized inverse A^- satisfies

$$(AA^-)^T = AA^-. \tag{8.1.44}$$

Then AA^- is the orthogonal projector onto $\mathcal{R}(A)$ and A^- is called a **least squares inverse**. We have

$$A^T = (AA^-A)^T = A^TAA^-,$$

which shows that $x = A^-b$ satisfies the normal equations and therefore is a least squares solution. Conversely, if $A^- \in \mathbf{R}^{n\times m}$ has the property that for all b, $\|Ax-b\|_2$ is smallest when $x = A^-b$, then A^- is a a least squares inverse

The following dual result holds also: If A^- is a generalized inverse, and

$$(A^-A)^T = A^-A$$

then A^-A is the orthogonal projector orthogonal to $\mathcal{N}(A)$ and A^- is called a **minimum norm inverse**. If $Ax = b$ is consistent, then the unique solution for which $\|x\|_2$ is smallest satisfies the normal equations

$$x = A^Tz, \quad AA^Tz = b.$$

For a minimum norm inverse we have

$$A^T = (AA^-A)^T = A^-AA^T,$$

and hence $x = A^Tz = A^-(AA^T)z = A^-b$, which shows that $x = A^-b$ is the solution of smallest norm. Conversely, if $A^- \in \mathbf{R}^{n\times m}$ is such that, whenever $Ax = b$ has a solution, then $x = A^-b$ is a minimum norm solution, then A^- is a minimum norm inverse.

We now derive some perturbation bounds for the pseudo-inverse of a matrix $A \in \mathbf{R}^{m\times n}$. Let $B = A + E$ be the perturbed matrix. The theory is complicated by the fact that when the rank changes the perturbation in $A^\dagger$ may be unbounded when the perturbation $\|E\|_2 \to 0$. A trivial example of this is obtained by taking

$$A = \begin{pmatrix} \sigma & 0 \\ 0 & 0 \end{pmatrix}, \quad E = \begin{pmatrix} 0 & 0 \\ 0 & \epsilon \end{pmatrix},$$

where $\sigma > 0$, $\epsilon \neq 0$. Then $1 = \text{rank }(A) \neq \text{rank }(A + E) = 2$,

$$A^\dagger = \begin{pmatrix} \sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix}, \quad (A+E)^\dagger = \begin{pmatrix} \sigma^{-1} & 0 \\ 0 & \epsilon^{-1} \end{pmatrix},$$

and $\|(A+E)^\dagger - A^\dagger\|_2 = |\epsilon|^{-1} = 1/\|E\|_2$. This example shows that formulas derived by operating formally with pseudo-inverses may have no meaning numerically.

The perturbations for which the pseudo-inverse is well behaved can be characterized by the condition

$$\text{rank}\,(A) = \text{rank}\,(B) = \text{rank}\,(P_{\mathcal{R}(A)}BP_{\mathcal{R}(A^T)}); \tag{8.1.45}$$

The matrix B is said to be an **acute perturbation** of A if this condition holds; see Stewart [545, 1977]. In particular, we have the following result.

Theorem 8.1.13.
If rank $(A+E) = \text{rank } (A) = r$, *and* $\eta = \|A^\dagger\|_2\|E\|_2 < 1$, *then*

$$\|(A+E)^\dagger\|_2 \le \frac{1}{1-\eta}\|A^\dagger\|_2. \tag{8.1.46}$$

Proof. From the assumption and Theorem 1.2.7 it follows that

$$1/\|(A+E)^\dagger\|_2 = \sigma_r(A+E) \ge \sigma_r(A) - \|E\|_2 = 1/\|A^\dagger\|_2 - \|E\|_2 > 0,$$

which implies (8.1.46). □

Let $A, B \in \mathbf{R}^{m\times n}$, and $E = B - A$. If A and $B = A+E$ are square nonsingular matrices, then we have the well-known identity

$$B^{-1} - A^{-1} = -B^{-1}EA^{-1}.$$

In the general case **Wedin's pseudo-inverse identity** (see [605]) holds

$$B^\dagger - A^\dagger = -B^\dagger EA^\dagger + (B^TB)^\dagger E^T P_{\mathcal{N}(A^T)} + P_{\mathcal{N}(B)}E^T(AA^T)^\dagger, \tag{8.1.47}$$

This identity can be proved by expressing the projections in terms of pseudo-inverses using the relations in (8.1.39).

Let $A = A(\alpha)$ be a matrix, where α is a scalar parameter. Under the assumption that $A(\alpha)$ has local constant rank the following formula for the derivative of the pseudo-inverse $A^\dagger(\alpha)$ follows from (8.1.47):

$$\frac{dA^\dagger}{d\alpha} = -A^\dagger\frac{dA}{d\alpha}A^\dagger + (A^TA)^\dagger\frac{dA^T}{d\alpha}P_{N(A)} + P_{N(A^T)}\frac{dA^T}{d\alpha}(AA^T)^\dagger. \tag{8.1.48}$$

This formula is due to Wedin [605, p 21]. We observe that if A has full column rank, then the second term vanishes; if A has full row rank, then it is the third term that vanishes. The variable projection algorithm for separable nonlinear least squares is based on a related formula for the derivative of the orthogonal projection matrix $P_{\mathcal{R}(A)}$; see Section 11.2.5.

For the case when rank $(B) =$ rank (A) the following theorem applies.

Theorem 8.1.14. *If* $B = A + E$ *and* rank $(B) =$ rank (A), *then*

$$\|B^\dagger - A^\dagger\| \le \mu\|B^\dagger\|\ \|A^\dagger\|\ \|E\| \tag{8.1.49}$$

where $\mu = 1$ *for the Frobenius norm* $\|\cdot\|_F$, *and for the spectral norm* $\|\cdot\|_2$,

$$\mu = \begin{cases} \frac{1}{2}(1+\sqrt{5}) & \text{if rank } (A) < \min(m,n), \\ \sqrt{2} & \text{if rank } (A) = \min(m,n). \end{cases}$$

Proof. For the $\|\cdot\|_2$ norm, see Wedin [606]. The result that $\mu = 1$ for the Frobenius norm is due to van der Sluis and Veltkamp [583]. □

8.1.5 Matrix Approximation and the SVD

The singular values decomposition (SVD) plays a very important role in a number of least squares matrix approximation problems. In this section we have collected a number of results that will be used extensively in the following.

In the proof of Theorem 7.1.16 we showed that the largest singular value of A could be characterized by

$$\sigma_1 = \max_{\|x\|_2=1} \|Ax\|_2.$$

The other singular values can also be characterized by an extremal property, the **minimax characterization**.

Theorem 8.1.15.

Let $A \in \mathbf{R}^{m\times n}$ have singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$, $p = \min(m,n)$, and S be a linear subspace of $\mathbf{R}^n$ of dimension $\dim(S)$. Then

$$\sigma_i = \max_{dim(S)=i} \max_{\substack{x\in S \\ x^H=1}} \|Ax\|_2, \quad i = 1:p, \tag{8.1.50}$$

and

$$\sigma_i = \min_{dim(S)=p-i+1} \max_{\substack{x\in S \\ x^H=1}} \|Ax\|_2, \quad i = 1:p. \tag{8.1.51}$$

Proof. The result follows from the relationship shown in Theorem 10.5.2 and the corresponding result for the Hermitian eigenvalue problem in Theorem 10.2.8 (Fischer's theorem). □

The minimax characterization of the singular values may be used to establish the following relations between the singular values of two matrices A and B.

Theorem 8.1.16.

Let $A, B \in \mathbf{R}^{m\times n}$ have singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p$ and $\tau_1 \geq \tau_2 \geq \ldots \geq \tau_p$ respectively, where $p = \min(m,n)$. Then

$$\max_i |\sigma_i - \tau_i| \leq \|A - B\|_2, \tag{8.1.52}$$

$$\sum_{i=1}^{p} |\sigma_i - \tau_i|^2 \leq \|A - B\|_F^2. \tag{8.1.53}$$

Proof. See Stewart [543, pp. 321–322]. □

By the inequality (8.1.53) no singular value of a matrix can be perturbed more than the 2-norm of the perturbation matrix. In particular, perturbation of a single element of a matrix A result in perturbations of the same, or smaller, magnitude in the singular values. This result is important for the use of the SVD to determine the "numerical rank" of a matrix; see below.

If a matrix A is modified by appending a row or a column, the singular values of the modified matrix can be shown to interlace those of A.

Theorem 8.1.17.
Let

$$\hat{A} = (A,\ u) \in \mathbf{R}^{m\times n}, \quad m \geq n, \quad u \in \mathbf{R}^m.$$

Then the ordered singular values σ_i of A interlace the ordered singular values $\hat{\sigma}_i$ of $\hat{A}$ as follows

$$\hat{\sigma}_1 \geq \sigma_1 \geq \hat{\sigma}_2 \geq \sigma_2 \ldots \geq \hat{\sigma}_{n-1} \geq \sigma_{n-1} \geq \hat{\sigma}_n.$$

Similarly, if A is bordered by a row,

$$\hat{A} = \begin{pmatrix} A \\ v^T \end{pmatrix} \in \mathbf{R}^{m\times n}, \quad m > n, \quad v \in \mathbf{R}^n,$$

then

$$\hat{\sigma}_1 \geq \sigma_1 \geq \hat{\sigma}_2 \geq \sigma_2 \ldots \geq \hat{\sigma}_{n-1} \geq \sigma_{n-1} \geq \hat{\sigma}_n \geq \sigma_n.$$

Proof. The theorem is a consequence of the Cauchy interlacing theorem for Hermitian matrices to be proved in Chapter 9; see Theorem 10.2.11. This says that the eigenvalues of the leading principal minor of order $n-1$ of a Hermitian matrix B interlace those of B. Since

$$\begin{pmatrix} A^T \\ u^T \end{pmatrix} (A \quad u) = \begin{pmatrix} A^TA & A^Tu \\ u^TA & u^Tu \end{pmatrix},$$
$$\begin{pmatrix} A \\ v^T \end{pmatrix} (A^T \quad v) = \begin{pmatrix} AA^T & Av \\ v^TA^T & v^Tv \end{pmatrix}$$

The result now follows from the observation that the singular values of A.are the positive square roots of the eigenvalues of A^TA and AA^T. □

The best approximation of a matrix A by another matrix of lower rank can be expressed in terms of the SVD of A.

Theorem 8.1.18.
Let $\mathcal{M}_k^{m\times n}$ denote the set of matrices in $\mathbf{R}^{m\times n}$ of rank k. Assume that $A \in \mathcal{M}_r^{m\times n}$ and consider the problem

$$\min_{X\in\mathcal{M}_k^{m\times n}} \|A - X\|, \quad k < r.$$

Then the SVD expansion of A truncated to k terms $X = B = \sum_{i=1}^k \sigma_i u_i v_i^T$, solves this problem both for the l_2 norm and the Frobenius norm. Further, the minimum distance is given by

$$\|A - B\|_2 = \sigma_{k+1}, \qquad \|A - B\|_F = (\sigma_{k+1}^2 + \ldots + \sigma_r^2)^{1/2}.$$

The solution is unique for the Frobenius norm but not always for the l_2 norm.

Proof. Eckhard and Young [183] proved it for the Frobenius norm. Mirsky [439] generalized it to unitarily invariant norms, which includes the l_2-norm. □

Let $A \in \mathbf{C}^{m\times n}$, be a matrix of rank n with the "thin" SVD $A = U_1\Sigma V^H$. Since $A = U_1\Sigma V^H = U_1\Sigma U_1^H U_1 V^H$, we have

$$A = PH, \quad P = U_1 V^H, \quad H = V\Sigma V^H, \tag{8.1.54}$$

where P is unitary, $P^H P = I$, and $H \in \mathbf{C}^{n\times n}$ is Hermitian positive semidefinite. The decomposition (8.1.54) is called the **polar decomposition** of A. If $\operatorname{rank}(A) = n$, then H is positive definite and the polar decomposition is unique. If the polar decomposition $A = PH$ is given, then from a spectral decomposition $H = V\Sigma V^H$ one can construct the singular value decomposition $A = (PV)\Sigma V^H$. The polar decomposition is also related to the matrix square root and sign functions; see Section 10.8.4.

The significance of the factor P in the polar decomposition is that it is the unitary matrix closest to A.

Theorem 8.1.19.

Let $A \in \mathbf{C}^{m\times n}$ be a given matrix and $A = UH$ its polar decomposition. Then for any unitary matrix $U \in \mathcal{M}_{m\times n}$,

$$\|A - U\|_F \geq \|A - P\|_F.$$

Proof. This theorem was proved for $m = n$ and general unitarily invariant norms by Fan and Hoffman [197]. The generalization to $m > n$ follows from the additive property of the Frobenius norm. □

Less well known is that the optimal properties of the Hermitian polar factor H. Let $A \in \mathbf{C}^{n\times n}$ be a Hermitian matrix with at least one negative eigenvalue. Consider the problem of finding a perturbation E such that $A + E$ is positive semidefinite.

Theorem 8.1.20.

Let $A \in \mathbf{C}^{m\times n}$ be Hermitian and $A = UH$ its polar decomposition. Set

$$B = A + E = \frac{1}{2}(H + A), \qquad E = \frac{1}{2}(H - A).$$

Then for any positive semidefinite Hermitian matrix X it holds that

$$\|A - B\|_2 \leq \|A - X\|_2.$$

Proof. See Higham [323]. □

8.1.6 Principal Angles and Distance Between Subspaces

In many applications the relationship between two given subspaces needs to be investigated. For example, in statistical models canonical correlations measure how "close" two set of observations are.

This and similar questions can be answered by computing angles between subspaces. Let $\mathcal{F}$ and $\mathcal{G}$ be subspaces of $\mathbf{C}^n$ and assume that

$$p = \dim(\mathcal{F}) \geq \dim(\mathcal{G}) = q \geq 1.$$

The smallest angle $\theta_1 = \theta_1(\mathcal{F}, \mathcal{G}) \in [0, \pi/2]$ between $\mathcal{F}$ and $\mathcal{G}$ is defined by

$$\theta_1 = \max_{\substack{u \in \mathcal{F} \\ \|u\|_2 = 1}} \max_{\substack{v \in \mathcal{G} \\ \|v\|_2 = 1}} \theta(u, v).$$

where $\theta(u, v)$ is the acute angle between u and v. Assume that the maximum is attained for $u = u_1$ and $v = v_1$. Then θ_2 is defined as the smallest angle between the orthogonal complement of $\mathcal{F}$ with respect to u_1 and that of $\mathcal{G}$ with respect to v_1. Continuing in this way until one of the subspaces is empty, we are led to the following definition:

Definition 8.1.21.

The **principal angles** $\theta_k \in [0, \pi/2]$ *between two subspaces of* $\mathbf{C}^n$ *are recursively defined for* $k = 1 : q$*, by*

$$\theta_k = \max_{\substack{u \in \mathcal{F} \\ \|u\|_2 = 1}} \max_{\substack{v \in \mathcal{G} \\ \|v\|_2 = 1}} \theta(u, v). = \theta(u_k, v_k), \tag{8.1.55}$$

subject to the constraints

$$u^H u_j = 0, \qquad v^H v_j = 0, \quad j = 1 : k-1.$$

The vectors u_k *and* v_k*,* $k = 1 : q$*, are called* **principal vectors** *of the pair of spaces.*

The principal vectors are not always uniquely defined, but the principal angles are. The vectors $V = (v_1, \ldots, v_q)$ form a unitary basis for $\mathcal{G}$ and the vectors $U = (u_1, \ldots, u_q)$ can be complemented with $(p - q)$ unitary vectors so that $(u_1, \ldots, u_p)$ form a unitary basis for $\mathcal{F}$. It will be shown that it holds also that

$$u_j^H v_k = 0, \quad j \neq k, \quad j = 1 : p, \quad k = 1 : q.$$

Assume that the subspaces $\mathcal{F}$ and $\mathcal{G}$ are defined as the range of the unitary matrices $Q_F \in \mathbf{C}^{n \times p}$ and $Q_G \in \mathbf{C}^{n \times q}$. The following theorem shows the relationship between the principal angles and the SVD of the matrix $Q_F^H Q_G$.

Theorem 8.1.22.

Assume that the columns of $Q_F \in \mathbf{C}^{n \times p}$ *and* $Q_G \in \mathbf{C}^{n \times q}$*,* $p \geq q$*, form unitary bases for two subspaces of* $\mathbf{C}^n$*. Let the thin SVD of the matrix* $M = Q_F^H Q_G \in \mathbf{C}^{p \times q}$ *be*

$$M = YCZ^H, \quad C = \operatorname{diag}(\sigma_1, \ldots, \sigma_q), \tag{8.1.56}$$

where $Y^HY = Z^HZ = ZZ^H = I_q$ *and* $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_q$. *Then the principal angles are* $\theta_k = \arccos(\sigma_k)$ *and the associated principal vectors are given by*

$$U = Q_F Y, \qquad V = Q_G Z. \tag{8.1.57}$$

Proof. The singular values and vectors of M can be characterized by the property

$$\sigma_k = \max_{\|y\|_2=\|z\|_2=1} y^H M z = y_k^H M z_k, \tag{8.1.58}$$

subject to $y^H yj = z^H z_j = 0$, $j = 1 : k$. If we put $u = Q_F y \in \mathcal{F}$ and $v = Q_G z \in \mathcal{G}$, then it follows that $\|u\|_2 = \|y\|_2$, $\|v\|_2 = \|z\|_2$, and

$$u^H u_j = y^H y_j, \qquad v^H v_j = z^H z_j.$$

Since $y^H M z = y^H Q_F^H Q_G z = u^H v$, (8.1.58) is equivalent to

$$\sigma_k = \max_{\|u\|_2=\|v\|_2=1} u_k^H v_k,$$

subject to $u^H u_j = 0$, $v^H v_j = 0$, $j = 1 : k-1$. Now (8.1.57) follows directly from definition 8.1.21. □

The principal angles can be used to define the distance between two subspaces of the same dimension.

Definition 8.1.23.

The distance between two subspaces $\mathcal{F}$ *and* $\mathcal{G}$ *of* $\mathbf{C}^n$, *both of dimension* p, *equals*

$$\operatorname{dist}(\mathcal{F}, \mathcal{G}) = \sin\theta_{\max}(\mathcal{F}, \mathcal{G})$$

where $\theta_{\max}(\mathcal{F}, \mathcal{G})$ *is the largest principal angle between* $\mathcal{F}$ *and* $\mathcal{G}$. *Equivalently*

$$\theta_{\max}(\mathcal{F}, \mathcal{G}) = \max_{\substack{u\in\mathcal{F} \\ \|u\|_2=1}} \min_{\substack{v\in\mathcal{G} \\ \|v\|_2=1}} \theta(u, v). \tag{8.1.59}$$

where $\theta(u, v) = \arccos(u^H u)$ *is the acute angle between* u *and* v.

Clearly $0 \leq \operatorname{dist}(\mathcal{F}, \mathcal{G}) \leq 1$, and $\operatorname{dist}(\mathcal{F}, \mathcal{F}) = 0$ if and only if $\mathcal{F} = \mathcal{G}$. The distance can also be expressed using the orthogonal projectors $P_{\mathcal{F}}$ and $P_{\mathcal{G}}$ onto $\mathcal{F}$ and $\mathcal{G}$

$$\operatorname{dist}(\mathcal{F}, \mathcal{G}) = \|P_{\mathcal{F}} - P_{\mathcal{G}}\|_2; \tag{8.1.60}$$

see Golub and Van Loan [277, Theorem 2.6.1].

In principle, a unitary basis for the *intersection of two subspaces* is obtained by taking the vectors u_k that corresponding to $\theta_k = 0$ or $\sigma_k = 1$. However, numerically small angles θ_k are well defined from $\sin\theta_k$ but not from $\cos\theta_k$. We now show how to compute $\sin\theta_k$.

We now change the notations slightly and write the SVD in (8.1.56) and the principal vectors as

$$M = Y_F C Y_G^H, \qquad U_F = Q_F Y_F, \quad U_G = Q_G Y_G.$$

Since Q_F is unitary it follows that $P_F = Q_F Q_F^H$ is the orthogonal projector onto $\mathcal{F}$. Then we have

$$P_F Q_G = Q_F Q_F^H Q_G = Q_F M = U_F C Y_G. \tag{8.1.61}$$

Squaring $Q_G = P_F Q_G + (I - P_F)Q_G$, using (8.1.61) and $P_F(I - P_F) = 0$ gives

$$Q_G^H (I - P_F)^2 Q_G = Y_G (I - C^2) Y_G^H,$$

which shows that the SVD of $(I - P_F)Q_G = Q_G - Q_F M$ can be written

$$(I - P_F)Q_G = W_F S Y_G^H, \quad S^2 = I - C^2,$$

and thus $S = \pm \mathrm{diag}\,(\sin\theta_k)$.

We assume for convenience in the following that $p + q \le n$. Then the matrix $W_F \in \mathbf{R}^{n\times q}$ can be chosen so that $W_F^H U_F = 0$.

$$(I - P_G)Q_F = Q_F - Q_G M = W_G S Y_F^H. \tag{8.1.62}$$

Combining this with $P_F Q_G = U_F C Y_G^H$ we can write

$$U_G = Q_G Y_F = (U_F C + W_F S) = (U_F \ W_F) \begin{pmatrix} C \\ S \end{pmatrix}.$$

If we put

$$P_{A,B} = U_G U_F^H = (U_F \ W_F) \begin{pmatrix} C \\ S \end{pmatrix} U_F^H$$

then the transformation $y = P_{A,B}x$, rotates a vector $x \in R(A)$ into a vector $y \in R(B)$, and $\|y\|_2 = \|x\|_2$. By analogy we have also the decomposition

$$(I - P_F)Q_G = Q_G - Q_F M = W_F S Y_G^H. \tag{8.1.63}$$

8.1.7 The CS Decomposition

More information about the relationship between two subspaces can be obtained from the **CS decomposition**. This is a special case a decomposition of a partitioned orthogonal matrix related to the SVD.

Theorem 8.1.24 (*Thin CS Decomposition*)**.**

Let $Q_1 \in \mathbf{R}^{(m\times n)}$ have orthonormal columns, that is $Q_1^T Q_1 = I$, and be partitioned as

$$Q_1 = \begin{pmatrix} Q_{11} \\ Q_{21} \end{pmatrix} \begin{matrix} \}m_1 \\ \}m_2 \end{matrix}, \tag{8.1.64}$$

where $m_1 \geq n$, and $m_2 \geq n$. Then there are orthogonal matrices $U_1 \in \mathbf{R}^{m_1\times m_1}$, $U_2 \in \mathbf{R}^{m_2\times m_2}$, and $V_1 \in \mathbf{R}^{n\times n}$ such that

$$\begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix}^T \begin{pmatrix} Q_{11} \\ Q_{21} \end{pmatrix} V_1 = \begin{pmatrix} C \\ 0 \\ S \\ 0 \end{pmatrix} \tag{8.1.65}$$

where

$$C = \operatorname{diag}(c_1, \ldots, c_n), \quad S = \operatorname{diag}(s_1, \ldots, s_n), \tag{8.1.66}$$

are square nonnegative diagonal matrices satisfying $C^2 + S^2 = I_n$. The diagonal elements in C and S are

$$c_i = \cos(\theta_i), \quad s_i = \sin(\theta_i), \quad i = 1 : n,$$

where without loss of generality, we may assume that

$$0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_n \leq \pi/2.$$

Proof. To construct U_1, V_1, and C, note that since U_1 and V_1 are orthogonal and C is a nonnegative diagonal matrix, $Q_{11} = U_1 C V_1^T$ is the SVD of Q_{11}. Hence, the elements c_i are the singular values of Q_{11}, and since $\|Q_{11}\|_2 \leq \|Q\|_2 = 1$, we have $c_i \in [0, 1]$.

If we put $\widetilde{Q}_{21} = Q_{21} V_1$, then the matrix

$$\begin{pmatrix} C \\ 0 \\ \widetilde{Q}_{21} \end{pmatrix} = \begin{pmatrix} U_1^T & 0 \\ 0 & I_{m_2} \end{pmatrix} \begin{pmatrix} Q_{11} \\ Q_{21} \end{pmatrix} V_1$$

has orthonormal columns. Thus, $C^2 + \widetilde{Q}_{21}^T \widetilde{Q}_{21} = I_n$, which implies that $\widetilde{Q}_{21}^T \widetilde{Q}_{21} = I_n - C^2$ is diagonal and hence the matrix $\widetilde{Q}_{21} = (\widetilde{q}_1^{(2)}, \ldots, \widetilde{q}_n^{(2)})$ has orthogonal columns.

We assume that the singular values $c_i = \cos(\theta_i)$ of Q_{11} have been ordered according to (8.1.24) and that $c_r < c_{r+1} = 1$. Then the matrix $U_2 = (u_1^{(2)}, \ldots, u_p^{(2)})$ is constructed as follows. Since $\|\widetilde{q}_j^{(2)}\|_2^2 = 1 - c_j^2 \neq 0, \quad j \leq r$ we take

$$u_j^{(2)} = \widetilde{q}_j^{(2)} / \|\widetilde{q}_j^{(2)}\|_2, \qquad j = 1, \ldots, r,$$

and fill the possibly remaining columns of U_2 with orthonormal vectors in the orthogonal complement of $\mathcal{R}(\widetilde{Q}_{21})$. From the construction it follows that $U_2 \in$

$\mathbf{R}^{m_2\times m_2}$ is orthogonal and that

$$U_2^T\widetilde{Q}_{21} = U_2Q_{21}V_1 = \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}, \qquad S = \operatorname{diag}(s_1,\ldots,s_q)$$

with $s_j = (1-c_j^2)^{1/2} > 0$, if $j = 1:r$, and $s_j = 0$, if $j = r+1:n$. □

In the theorem above we assumed that $n \le m/2$. The general case gives rise to four different forms corresponding to cases where Q_{11} and/or Q_{21} have too few rows to accommodate a full diagonal matrix of order n.

The proof of the CS decomposition is constructive. In particular, U_1, V_1, and C can be computed by a standard SVD algorithm. However, the above algorithm for computing U_2 is unstable when some singular values c_i are close to 1. and needs to be modified.

Using the same technique the following CS decomposition of a square partitioned orthogonal matrix can be shown.

Theorem 8.1.25 (*Full CS Decomposition*).

Let

$$Q = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \in \mathbf{R}^{m\times m}. \tag{8.1.67}$$

be an arbitrary partitioning of the orthogonal matrix Q. Then there are orthogonal matrices

$$\begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix}$$

such that

$$U^TQV = \left(\begin{array}{c|c} U_1^TQ_{11}V_1 & U_1^TQ_{12}V_2 \\ \hline U_2^TQ_{21}V_1 & U_2^TQ_{22}V_2 \end{array}\right) = \left(\begin{array}{ccc|ccc} I & 0 & 0 & 0 & 0 & 0 \\ 0 & C & 0 & 0 & S & 0 \\ 0 & 0 & 0 & 0 & 0 & I \\ \hline 0 & 0 & 0 & I & 0 & 0 \\ 0 & S & 0 & 0 & -C & 0 \\ 0 & 0 & I & 0 & 0 & 0 \end{array}\right) \tag{8.1.68}$$

where $C = \operatorname{diag}(c_,\ldots,c_n)$ and $S = \operatorname{diag}(s_,\ldots,s_n)$ are square diagonal matrices satisfying $C^2 + S^2 = I_n$, and $0 < c_i, s_i < 1$, $i = 1:n$.

Proof. For a proof, see Paige and Saunders [468]. □

The history of the CS decomposition and its many applications are surveyed in Paige and Wei [473].

Review Questions

1.1 State the Gauss–Markov theorem.

1.2 Assume that A has full column rank. Show that the matrix $P = A(A^TA)^{-1}A^T$ is symmetric and satisfies the condition $P^2 = P$.

1.3 (a) Give conditions for a matrix P to be the orthogonal projector onto a subspace $S \in \mathbf{R}^n$.

(b) Define the orthogonal complement of S in $\mathbf{R}^n$.

1.4 (a) Which are the four fundamental subspaces of a matrix? Which relations hold between them? Express the orthogonal projections onto the fundamental subspaces in terms of the SVD.

(b) Give two geometric conditions which are necessary and sufficient conditions for x to be the pseudo-inverse solution of $Ax = b$.

1.5 Which of the following relations are universally correct?

(a) $\mathcal{N}(B) \subseteq \mathcal{N}(AB)$. (b) $\mathcal{N}(A) \subseteq \mathcal{N}(AB)$. (c) $\mathcal{N}(AB) \subseteq \mathcal{N}(A)$.

(d) $\mathcal{R}(AB) \subseteq \mathcal{R}(B)$. (e) $\mathcal{R}(AB) \subseteq \mathcal{R}(A)$. (f) $\mathcal{R}(B) \subseteq \mathcal{R}(AB)$.

1.6 (a) What are the four Penrose conditions for X to be the pseudo-inverse of A?

(b)A matrix X is said to be a **left-inverse** if $XA = I$. Show that a left-inverse is an $\{1, 2, 3\}$-inverse, i.e. satisfies the Penrose conditions (1), (2), and (3). Similarly, show that a **right-inverse** is an $\{1, 2, 4\}$-inverse.

1.7 Let the singular values of $A \in \mathbf{R}^{m\times n}$ be $\sigma_1 \geq \cdots \geq \sigma_n$. What relations are satisfied between these and the singular values of

$$\widetilde{A} = (A, u), \qquad \hat{A} = \begin{pmatrix} A \\ v^T \end{pmatrix}?$$

1.8 (a) Show that $A^\dagger = A^{-1}$ when A is a nonsingular matrix.

(b) Construct an example where $G \neq A^\dagger$ despite the fact that $GA = I$.

Problems

1.1 (a) Compute the pseudo-inverse $x^\dagger$ of a column vector x.

(b) Take $A = (\,1 \quad 0\,)$, $B = (\,1 \quad 1\,)^T$, and show that $1 = (AB)^\dagger \neq B^\dagger A^\dagger = 1/2$.

1.2 (a) Verify that the Penrose conditions uniquely defines the matrix X. Do it first for $A = \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$, and then transform the result to a general matrix A.

1.3 (a) Show that if $w \in \mathbf{R}^n$ and $w^Tw = 1$, then the matrix $P(w) = I - 2ww^T$ is both symmetric and orthogonal.

(b) Given two vectors $x, y \in \mathbf{R}^n$, $x \neq y$, $\|x\|_2 = \|y\|_2$, then

$$(w)x = y, \qquad w = (y - x)/\|y - x\|_2.$$

1.4 Let $S \subseteq \mathbf{R}^n$ be a subspace, P_1 and P_2 be orthogonal projections onto $S = \mathcal{R}(P_1) = \mathcal{R}(P_2)$. Show that $P_1 = P_2$, i.e., the orthogonal projection onto S is

unique.

Hint: Show that for any $z \in \mathbf{R}^n$

$$\|(P_1 - P_2)z\|_2^2 = (P_1 z)^T (I - P_2)z + (P_2 z)^T (I - P_1)z = 0.$$

1.5 (R. E. Cline) Let A and B be any matrices for which the product AB is defined, and set

$$B_1 = A^\dagger AB, \qquad A_1 = AB_1 B_1^\dagger.$$

Show that $AB = AB_1 = A_1 B_1$ and that $(AB)^\dagger = B_1^\dagger A_1^\dagger$.

Hint: Use the Penrose conditions.

1.6 (a) Show that the matrix $A \in \mathbf{R}^{m\times n}$ has a **left inverse** $A^L \in \mathbf{R}^{n\times m}$, i.e., $A^L A = I$, if and only if $\operatorname{rank}(A) = n$. Although in this case $Ax = b \in \mathcal{R}(A)$ has a unique solution, the left inverse is not unique. Find the general form of Σ^L and generalize the result to A^L.

(b) Discuss the **right inverse** A^R in a similar way.

(c) show the relation $\operatorname{rank}(A) = \operatorname{trace}(A^\dagger A)$.

1.7 Show that $A^\dagger$ minimizes $\|AX - I\|_F$.

1.8 Prove *Bjerhammar's characterization* : Let A have full column rank and let B be any matrix such that $A^T B = 0$ and $(\,A \quad B\,)$ is nonsingular. Then $A^\dagger = X^T$ where

$$\begin{pmatrix} X^T \\ Y^T \end{pmatrix} = (\,A \quad B\,)^{-1}.$$

8.2 The Method of Normal Equations

8.2.1 Forming and Solving the Normal Equations

Consider the linear Gauss–Markov model

$$Ax = b + \epsilon, \qquad A \in \mathbf{R}^{m\times n}, \tag{8.2.1}$$

where ϵ has zero mean and variance-covariance matrix equal to $\sigma^2 I$. By the Gauss–Markov theorem x is a least squares estimate if and only if it satisfies the normal equations $A^T Ax = A^T b$.

Theorem 8.2.1.

The matrix $A^T A$ is positive definite if and only if the columns of A are linearly independent, i.e., when $\operatorname{rank}(A) = n$. *In this case the least squares solution is unique and given by*

$$x = (A^T A)^{-1} A^T b, \qquad r = (I - A(A^T A)^{-1} A^T)b. \tag{8.2.2}$$

Proof. If the columns of A are linearly independent, then $x \neq 0 \Rightarrow Ax \neq 0$. Therefore, $x \neq 0 \Rightarrow x^T A^T Ax = \|Ax\|_2^2 > 0$, and hence $A^T A$ is positive definite.On the other hand, if the columns are linearly dependent, then for some $x_0 \neq 0$ we have

$Ax_0 = 0$. Then $x_0^T A^T A x_0 = 0$, and therefore $A^T A$ is not positive definite. When $A^T A$ is positive definite it is also nonsingular and (8.2.2) follows. □

After forming $A^T A$ and $A^T b$ the normal equations can be solved by symmetric Gaussian elimination (which Gauss did), or by computing the Cholesky factorization (due to [45])

$$A^T A = R^T R,$$

where R is upper triangular.

We now discuss some details in the numerical implementation of the method of normal equations. We defer treatment of rank deficient problems to later and assume throughout this section that the numerical rank of A equals n. The first step is to compute the elements of the symmetric matrix $C = A^T A$ and the vector $d = A^T b$. If $A = (a_1, a_2, \ldots, a_n)$ has been partitioned by columns, we can use the inner product formulation

$$c_{jk} = (A^T A)_{jk} = a_j^T a_k, \qquad d_j = (A^T b)_j = a_j^T b, \quad 1 \le j \le k \le n. \tag{8.2.3}$$

Since C is symmetric it is only necessary to compute and store its lower (or upper) triangular which requires $\frac{1}{2}mn(n+1)$ multiplications. Note that if $m \gg n$, then the number of elements $\frac{1}{2}n(n+1)$ in the upper triangular part of $A^T A$ is much smaller than the number mn of elements in A. Hence, in this case the formation of $A^T A$ and $A^T b$ can be viewed as a *data compression*!

In the inner product formulation (8.2.3) the data A and b are accessed columnwise. This may not always be suitable since each column needs to be accessed many times. For example, if A is so large that it is held in secondary storage, this will be expensive. In an alternative row oriented algorithm can be used, where outer products of the rows are accumulated. Denoting the ith row of A by $\widetilde{a}_i^T$, $i = 1 : m$, we have

$$C = A^T A = \sum_{i=1}^{m} \widetilde{a}_i \widetilde{a}_i^T, \qquad d = A^T b = \sum_{i=1}^{m} b_i \widetilde{a}_i. \tag{8.2.4}$$

Here $A^T A$ is expressed as the sum of m matrices of rank one and $A^T b$ as a linear combination of the transposed rows of A. This approach has the advantage that just one pass *one pass* through the rows of A is required, each row being fetched (possibly from auxiliary storage) or formed by computation when needed. No more storage is needed than that for $A^T A$ and $A^T b$. This outer product form is also preferable if the matrix A is sparse; see the hint to Problem 7.7.1. Note that both formulas can be combined if we adjoin b as the $(n+1)$st column to A and form

$$(A,\ b)^T (A,\ b) = \begin{pmatrix} A^T A & A^T b \\ b^T A & b^T b \end{pmatrix}.$$

The matrix $C = A^T A$ is symmetric, and if $\operatorname{rank}(A) = n$ it is also positive definite. Gauss solved the normal equations by symmetric Gaussian elimination. Computing the Cholesky factorization

$$C = A^T A = R^T R, \quad R \in \mathbf{R}^{n \times n}, \tag{8.2.5}$$

is now the standard approach. The Cholesky factor R is upper triangular and nonsingular and can be computed by one of the algorithms given in Section 7.4.2. The least squares solution is then obtained by solving the two triangular systems

$$R^T z = d, \qquad Rx = z. \tag{8.2.6}$$

Forming the matrix $A^T A$ and computing its Cholesky factorization requires (neglecting lower order terms) $mn^2 + \frac{1}{3}n^3$ flops. If we have several right hand sides b_i, $i = 1 : p$, then the Cholesky factorization need only be computed once. Forming $A^T b_i$ and solving the two triangular systems requires $mn + n^2$ additional flops.for each right hand side.

Example 8.2.1.

In statistics, the **multiple linear regression** problem is the problem of fitting a given linear model to a set of data points (x_i, y_i), $i = 1 : m$. Assume that the model to be fitted is $y = \alpha + \beta x$. Substituting the data gives m linear equations for the unknown parameters α and βx

$$\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_m \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}.$$

The least squares solution is obtained by forming the normal equations

$$\begin{pmatrix} m & m\bar{x} \\ m\,\bar{x} & x^T x \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} m\,\bar{y} \\ x^T y \end{pmatrix}. \tag{8.2.7}$$

where

$$\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i, \qquad \bar{y} = \frac{1}{m}\sum_{i=1}^{m} y_i,$$

are the mean values. Eliminating α we obtain the "classical" formulas

$$\beta = \big(x^T y - m\,\bar{y}\bar{x}\big)\big/\big(x^T x - m\,\bar{x}^2\big), \tag{8.2.8}$$

The first equation in (8.2.7) gives $\bar{y} = \alpha + \beta\bar{x}$. This shows that $(\bar{y}, \bar{x})$ lies on the fitted line and we have

$$\alpha = \bar{y} - \beta\bar{x}. \tag{8.2.9}$$

A more accurate formula for β is obtained by first subtracting out the mean values from the data and writing the model as $y - \bar{y} = \beta(x - \bar{x})$. In the new variables the matrix of normal equation is *diagonal*, and we find

$$\beta = \sum_{i=1}^{m}(y_i - \bar{y})(x_i - \bar{x})_i \Big/ \sum_{i=1}^{m}(x_i - \bar{x})^2. \tag{8.2.10}$$

A drawback of this formula is that it requires two passes through the data.

This is easily generalized to fitting a general linear model with $n < m$ parameters which leads to a linear least squares problem $\min_x \|y - Ax\|_2$. In statistics the vector y is called the response variable and the parameters are the explanatory variables.

In many least squares problems the matrix A has the property that in each row all nonzero elements in A are contained in a narrow band. For such a matrix we define:

Definition 8.2.2.
Let f_i and l_i be the column subscripts of the first and last nonzero in the ith row of the matrix $A \in \mathbf{R}^{m\times n}$, i.e.,

$$f_i = \min\{j \mid a_{ij} \neq 0\}, \qquad l_i = \max\{j \mid a_{ij} \neq 0\}. \tag{8.2.11}$$

Then A is said to have row bandwidth w, where

$$w = \max_{1\leq i\leq m} w_i, \qquad w_i = (l_i - f_i + 1). \tag{8.2.12}$$

For this structure to have practical significance we need to have $w \ll n$. Matrices of small row bandwidth often occur naturally, since they correspond to a situation where only variables "close" to each other are coupled by observations. We now prove a relation between the row bandwidth of the matrix A and the bandwidth of the corresponding matrix of normal equations A^TA.

Theorem 8.2.3.
Assume that the matrix $A \in \mathbf{R}^{m\times n}$ has row bandwidth w. Then the symmetric matrix A^TA has bandwidth $r \leq w - 1$.

Proof. From the Definition 8.2.2 it follows that $a_{ij}a_{ik} \neq 0$ implies that $|j - k| < w$ and thus

$$|j - k| \geq w \quad \Rightarrow \quad a_{ij}a_{ik} = 0 \quad \forall \;\; i = 1 : m. \tag{8.2.13}$$

and hence

$$(A^TA)_{jk} = \sum_{i=1}^{m} a_{ij}a_{ik} = 0.$$

□

If the matrix A also has full column rank it follows that we can use the band Cholesky Algorithm 7.7 to solve the normal equations.

8.2.2 Computing the Covariance Matrix.

Consider a general univariate linear model with error covariance matrix equal to a symmetric positive definite matrix $\sigma^2\mathcal{V}$. By Theorem 8.1.7 the least squares

estimate is the solution to

$$\min_x (Ax-b)^T V^{-1}(Ax-b).$$

and the covariance matrix of the solution x is $V_x = \sigma^2 C_x$, where

$$C_x = (A^T V^{-1} A)^{-1} = (R^T R)^{-1} = R^{-1} R^{-T}. \tag{8.2.14}$$

Here R is the Cholesky factor of $A^T V^{-1} A$. In the special case of a weighted least squares problem $V^{-1} = D^2$ is a diagonal matrix with positive elements.

An unbiased estimate of σ^2 is given by

$$s^2 = \frac{1}{m-n}(b-Ax)^T V^{-1}(b-Ax),. \tag{8.2.15}$$

The least squares residual vector $r = b - A\hat{x}$ has variance-covariance matrix equal to $\sigma^2 V_r$, where

$$V_r = V^{-1/2} A (A^T V^{-1} A)^{-1} A^T V^{-1/2} = BB^T, \quad B = V^{-1/2} A R^{-1}. \tag{8.2.16}$$

The **normalized residuals**

$$\widetilde{r} = \frac{1}{s}(\operatorname{diag} V_r)^{-1/2}\hat{r}$$

are often used to detect and identify bad data, which is assumed to correspond to large components in $\widetilde{r}$. If the error covariance matrix is correct, then the components of $\widetilde{r}$ should be uniformly distributed random quantities. In particular, the histogram of the entries of the residual should look like a bell curve.

In order to assess the accuracy of the computed estimate of x it is often required to compute the matrix C_x or part of it. If, for simplicity, we assume that $V = I$, then C_x is obtained from

$$C_x = SS^T, \quad S = R^{-1}.$$

The upper triangular matrix S is computed by solving the matrix equation $RS = I$. Using the algorithm given in (7.2.41) this requires $n^3/3$ flops. The upper triangular part of the symmetric matrix SS^T can then be formed. The computation of C_x can be sequenced so that the elements of C_x overwrite those of R andrequires a total of $2n^3/3$ flops.

In many situations the variance of a linear functional $\varphi = f^T\hat{x}$ of the least squares solution is of interest. This equals

$$\mathcal{V}(\varphi) = f^T V_x f = \sigma^2 f^T R^{-1} R^{-T} f = \sigma^2 z^T z, \tag{8.2.17}$$

where $z = R^{-T} f$. Here the matrix C_x occurs only as an intermediate quantity and the variance can be computed by solving the lower triangular system $R^T z = f$ by forward substitution and then forming $\sigma^2 z^T z = \sigma^2 \|z\|_2^2$. This is a more stable and efficient approach than using the expression involving V_x. In particular, the variance of a single component $x_i = e_i^T x$ of the solution is obtained from

$$\mathcal{V}(x_i) = \sigma^2 z^T z, \quad R^T z = e_i, \quad i = 1:n. \tag{8.2.18}$$

Note that since R^T is lower triangular, z will have $i-1$ leading zeros. For $i = n$ only the last component is nonzero and equals r_{nn}^{-1}.

Example 8.2.2.

In an early application of least squares Laplace [394] estimated the mass of Jupiter and Uranus and assessed the accuracy of the results by computing the corresponding variances. He made use of 129 observations of the movements of Jupiter and Saturn collected by Alexis Bouvard (1767–1843), French astronomer and director of the Paris Observatory. The final normal equations are $A^TAx = A^Tb$, where

$$A^TA = \begin{pmatrix} 795938 & -12729398 & 6788.2 & -1959.0 & 696.13 & 2602 \\ -12729398 & 424865729 & -153106.5 & -39749.1 & -5459 & 5722 \\ 6788.2 & -153106.5 & 71.8720 & -3.2252 & 1.2484 & 1.3371 \\ -1959.0 & -153106.5 & -3.2252 & 57.1911 & 3.6213 & 1.1128 \\ 696.13 & -5459 & 1.2484 & 3.6213 & 21.543 & 46.310 \\ 2602 & 5722 & 1.3371 & 1.1128 & 46.310 & 129 \end{pmatrix},$$

$$A^Tb = \begin{pmatrix} 7212.600 \\ -738297.800 \\ 237.782 \\ -40.335 \\ -343.455 \\ -1002.900 \end{pmatrix}.$$

In these equations the mass of Uranus is $(1 + x_1)/19504$, the mass of Jupiter is $(1 + x_2)/1067.09$ if the mass of the sun is taken as unity. Bouvard also gave the square residual norm as

$$\|b - A\hat{x}\|_2^2 = 31096.$$

Working from these normal equations Laplace computed the least squares solution and obtained

$$x_1 = 0.0895435, \qquad x_2 = -0.0030431.$$

From this we can calculate the mass of Jupiter and Uranus as a fraction of the mass of the sun. We obtain 1070.3 for Saturn and 17918 for Uranus. The covariance matrix of the solution is $V_x = s^2(R^TR)^{-1}$, where $s^2 = 31096/(129 - 6)$ is an unbiased estimate of σ^2. We get

$$v_{11} = 0.5245452 \cdot 10^{-2}, \qquad v_{22} = 4.383233 \cdot 10^{-6}.$$

This shows that the computed mass of of Jupiter is very reliable, while the computed mass of Uranus is not. Laplace concludes that with a probability of $1 - 10^{-6}$ the error in the computed mass of Jupiter is less than one per cent.

There is an alternative way of computing C_x without inverting R. We have from (8.2.17), multiplying by R from the left,

$$RC_x = R^{-T}. \tag{8.2.19}$$

The diagonal elements of R^{-T} are simply r_{kk}^{-1}, $k = n, \ldots, 1$, and since R^{-T} is lower triangular it has $\frac{1}{2}n(n-1)$ zero elements. Hence, $\frac{1}{2}n(n+1)$ elements of R^{-T} are

known and the corresponding equations in (8.2.19) suffice to determine the elements in the upper triangular part of the symmetric matrix C_x.

To compute the elements in the last column c_n of C_x we solve the system

$$Rc_n = r_{nn}^{-1} e_n, \quad e_n = (0, \ldots, 0, 1)^T$$

by back-substitution. This gives

$$c_{nn} = r_{nn}^{-2}, \quad c_{in} = -r_{ii}^{-1} \sum_{j=i+1}^{n} r_{ij} c_{jn}, \quad i = n-1, \ldots, 1. \tag{8.2.20}$$

By symmetry $c_{ni} = c_{in}, \quad i = n-1, \ldots, 1$, so we know also the last row of C_x. Now assume that we have computed the elements $c_{ij} = c_{ji}$, $j = n, \ldots, k+1$, $i \le j$. We next determine the elements c_{ik}, $i \le k$. We have

$$c_{kk} r_{kk} + \sum_{j=k+1}^{n} r_{kj} c_{jk} = r_{kk}^{-1},$$

and since the elements $c_{kj} = c_{jk}$, $j = k+1 : n$, have already been computed,

$$c_{kk} = r_{kk}^{-1} \left(r_{kk}^{-1} - \sum_{j=k+1}^{n} r_{kj} c_{kj} \right). \tag{8.2.21}$$

Similarly, for $i = k-1 : (-1) : 1$,

$$c_{ik} = -r_{ii}^{-1} \left(\sum_{j=i+1}^{k} r_{ij} c_{jk} + \sum_{j=k+1}^{n} r_{ij} c_{kj} \right). \tag{8.2.22}$$

Using the formulas (8.2.20)–(8.2.22) all the elements of C_x can be computed in about $2n^3/3$ flops.

When the matrix R is sparse, Golub and Plemmons [260] have shown that the same algorithm can be used very efficiently to compute *all elements in C_x, associated with nonzero elements in R.* Since R has a nonzero diagonal this includes the diagonal elements of C_x giving the variances of the components x_i, $i = 1 : n$. If R has bandwidth w, then the corresponding elements in C_x can be computed in only $2nw^2$ flops; see Björck [61, Sec 6.7.4].

8.2.3 Perturbation Bounds for Least Squares Problems

We now consider the effect of perturbations of A and b on the least squares solution x. In this analysis the condition number of the matrix $A \in \mathbf{R}^{m \times n}$ will play a significant role. The following definition generalizes the condition number (6.6.3) of a square nonsingular matrix.

Definition 8.2.4.

Let $A \in \mathbf{R}^{m\times n}$ have rank $r > 0$ and singular values equal to $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. Then the condition number of A is

$$\kappa(A) = \|A\|_2 \|A^\dagger\|_2 = \sigma_1/\sigma_r,$$

where the last equality follows from the relations $\|A\|_2 = \sigma_1$, $\|A^\dagger\|_2 = \sigma_r^{-1}$.

Using the SVD $A = U\Sigma V^T$ we obtain

$$A^TA = V\Sigma^T(U^TU)\Sigma V^T = V\begin{pmatrix}\Sigma_r^2 & 0\\ 0 & 0\end{pmatrix}V^T. \tag{8.2.23}$$

Hence, $\sigma_i(A^TA) = \sigma_i^2(A)$, and it follows that $\kappa(A^TA) = \kappa^2(A)$. This shows that *the matrix of the normal equations has a condition number which is the square of the condition number of A.*

We now give a first order perturbation analysis for the least squares problem when $\operatorname{rank}(A) = n$. Denote the perturbed data $A + \delta A$ and $b + \delta b$ and assume that δA sufficiently small so that $\operatorname{rank}(A + \delta A) = n$. Let the perturbed solution be $x + \delta x$ and $r + \delta r$, where $r = b - Ax$ is the residual vector. The perturbed solution satisfies the augmented system

$$\begin{pmatrix} I & A+\delta A\\ (A+\delta A)^T & 0\end{pmatrix}\begin{pmatrix}\widetilde{s}\\ \widetilde{x}\end{pmatrix} = \begin{pmatrix} b+\delta b\\ 0\end{pmatrix}. \tag{8.2.24}$$

Subtracting the unperturbed equations and neglecting second order quantities the perturbations $\delta s = \delta r$ and δx satisfy the linear system

$$\begin{pmatrix} I & A\\ A^T & 0\end{pmatrix}\begin{pmatrix}\delta s\\ \delta x\end{pmatrix} = \begin{pmatrix} \delta b - \delta A x\\ -\delta A^T s\end{pmatrix}. \tag{8.2.25}$$

From the Schur–Banachiewicz formula (see Section 7.1.5) it follows that the inverse of the matrix in this system equals

$$\begin{aligned}\begin{pmatrix} I & A\\ A^T & 0\end{pmatrix}^{-1} &= \begin{pmatrix} (I - A(A^TA)^{-1}A^T) & A(A^TA)^{-1}\\ (A^TA)^{-1}A^T & -(A^TA)^{-1}\end{pmatrix}\\ &= \begin{pmatrix} P_{\mathcal{N}(A^T)} & (A^\dagger)^T\\ A^\dagger & -(A^TA)^{-1}\end{pmatrix}.\end{aligned} \tag{8.2.26}$$

We find that the perturbed solution satisfies

$$\delta x = A^\dagger(\delta b - \delta A\, x) + (A^TA)^{-1}\delta A^T r, \tag{8.2.27}$$

$$\delta r = P_{\mathcal{N}(A^T)}(\delta b - \delta A x) - (A^\dagger)^T\delta A^T\, r. \tag{8.2.28}$$

Assuming that the perturbations δA and δb satisfy

$$|\delta A| \leq \omega E, \qquad |\delta b| \leq \omega f, \tag{8.2.29}$$

and substituting in (8.2.27)–(8.2.28) yields the componentwise bounds

$$|\delta x| \lessapprox \omega \left(|A^\dagger|(f + E|x|) + |(A^TA)^{-1}|E^T|r| \right), \tag{8.2.30}$$

$$|\delta r| \lessapprox \omega \left(|I - AA^\dagger|(f + E|x|) + |(A^\dagger)^T|E^T|r| \right). \tag{8.2.31}$$

where terms of order $O(\omega^2)$ have been neglected. Note that if the system $Ax = b$ is consistent, then $r = 0$ and the bound for $|\delta x|$ is identical to that obtained for a square nonsingular linear system.

Taking norms in (8.2.27) and (8.2.28) and using

$$\|A^\dagger\|_2 = \|(A^\dagger)^T\|_2 = 1/\sigma_n, \qquad \|(A^TA)^{-1}\|_2 = 1/\sigma_n^2, \qquad \|P_{\mathcal{N}(A^T)}\|_2 = 1.$$

it follows that

$$\|\delta x\|_2 \lessapprox \frac{1}{\sigma_n}\|\delta b\|_2 + \frac{1}{\sigma_n}\|\delta A\|_2 \left(\|x\|_2 + \frac{1}{\sigma_n}\|r\|_2 \right), \tag{8.2.32}$$

$$\|\delta r\|_2 \lessapprox \|\delta b\|_2 + \|\delta A\|_2 \left(\|x\|_2 + \frac{1}{\sigma_n}\|r\|_2 \right), \tag{8.2.33}$$

If $r \neq 0$ there is a term proportional to σ_n^{-2} present in the bound for $\|\delta x\|_2$. A more refined perturbation analysis (see Wedin [606]) shows that if

$$\eta = \|A^\dagger\|_2 \|\delta A\|_2 \ll 1.$$

then $\operatorname{rank}(A + \delta A) = n$, and there are perturbations δA and δb such that these upper bounds are almost attained.

Assuming that $x \neq 0$ and setting $\delta b = 0$, we get an upper bound for the normwise relative perturbation

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \kappa_{LS} \frac{\|\delta A\|_2}{\|A\|_2}, \quad \kappa_{LS} = \kappa(A)\Big(1 + \frac{\|r\|_2}{\sigma_n \|x\|_2}\Big) \tag{8.2.34}$$

Hence, κ_{LS} is the condition number for the least squares problem. The following two important facts should be noted:

- κ_{LS} depends not only on A but also on $r = P_{\mathcal{N}(A^T)}b$.
- If $\|r\|_2 \ll \sigma_n \|x\|_2$, then $\kappa_{LS} \approx \kappa(A)$, but if $\|r\|_2 > \sigma_n \|x\|_2$ the second term in (8.2.34) will dominate,

Example 8.2.3.

The following simple example illustrates the perturbation analysis above. Consider a least squares problem with

$$A = \begin{pmatrix} 1 & 0 \\ 0 & \delta \\ 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 0 \\ \alpha \end{pmatrix}, \qquad \delta A = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \delta/2 \end{pmatrix}.$$

and $\kappa(A) = 1/\delta \gg 1$. If $\alpha = 1$, then

$$x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \delta x = \frac{2}{5\delta}\begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad r = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \delta r = -\frac{1}{5}\begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}.$$

For this right hand side $\|x\|_2 = \|r\|_2$ and $\kappa_{LS} = 1/\delta + 1/\delta^2 \approx \kappa^2(A)$. This is reflected in the size of δx.

If instead we take $\alpha = \delta$, then a short calculation shows that $\|r\|_2/\|x\|_2 = \delta$ and $\kappa_{LS} = 2/\delta$. The same perturbation δA now gives

$$\delta x = \frac{2}{5}\begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad \delta r = -\frac{\delta}{5}\begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}.$$

It should be stressed that in order for the perturbation analysis above to be useful, the matrix A and vector b should be scaled so that perturbations are "well defined" by bounds on $\|\delta A\|_2$ and $\|b\|_2$. It is not uncommon that the columns in $A = (a_1, a_2, \ldots, a_n)$ have widely differing norms. Then a much better estimate may often be obtained by applying (8.2.34) to the scaled problem $\min_{\widetilde{x}} \|\widetilde{A}\widetilde{x} - b\|_2$, chosen so that $\widetilde{A}$ has columns of unit length, i.e.,

$$\widetilde{A} = AD^{-1}, \qquad \widetilde{x} = Dx, \qquad D = \operatorname{diag}(\|a_1\|_2, \ldots, \|a_n\|_2).$$

By Theorem 8.2.5 this column scaling approximately minimizes $\kappa(AD^{-1})$ over $D > 0$. Note that scaling the columns changes also the norm in which the error in the original variables x is measured.

If the *rows* in A differ widely in norm, then (8.2.34) may also considerably overestimate the perturbation in x. As remarked above, we cannot scale the rows in A without changing the least squares solution.

Perturbation bounds with better scaling properties can be obtained by considering componentwise perturbations; see Section 8.2.4.

8.2.4 Stability and Accuracy with Normal Equations

We now turn to a discussion of the accuracy of the method of normal equations for least squares problems. First we consider rounding errors in the formation of the system of normal equations. Using the standard model for floating point computationthe computed matrix matrix $\bar{C} = fl(A^TA)$ satisfies

$$\bar{C} = fl(A^TA) = C + E, \quad |E| < \gamma_m |A|^T|A|. \tag{8.2.35}$$

where (see Lemma 7.1.21) $|\gamma_m| < mu/(1 - mu)u$ and u is the unit roundoff. A similar estimate holds for the rounding errors in the computed vector A^Tb.

It should be emphsized that it is *not* in general possible to show that $\bar{C} = (A + E)^T(A + E)$ for some small error matrix E. That is, the rounding errors performed in forming the matrix A^TA are not in general equivalent to small perturbations of

the initial data matrix A. This means that it is not in general true that the solution computed from the normal equations equals the exact solution to a problem where the data A and b have been perturbed by small amounts. In other words, *the method of normal equations is not backwards stable.* In spite of this it often gives satisfactory results. However, as the following example illustrates, when A^TA is ill-conditioned, *it might be necessary to use extra precision in forming and solving the normal equations in order to avoid loss of significant information in the data.*

Example 8.2.4.

Läuchli [398]: Consider the system $Ax = b$, where

$$A = \begin{pmatrix} 1 & 1 & 1 \\ \epsilon & & \\ & \epsilon & \\ & & \epsilon \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \qquad |\epsilon| \ll 1.$$

We have, exactly

$$A^TA = \begin{pmatrix} 1+\epsilon^2 & 1 & 1 \\ 1 & 1+\epsilon^2 & 1 \\ 1 & 1 & 1+\epsilon^2 \end{pmatrix}, \qquad A^Tb = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

$$x = \frac{1}{3+\epsilon^2}\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}^T, \qquad r = \frac{1}{3+\epsilon^2}\begin{pmatrix} \epsilon^2 & -1 & -1 & -1 \end{pmatrix}^T.$$

Now assume that $\epsilon = 10^{-4}$, and that we use eight-digit decimal floating point arithmetic. Then $1 + \epsilon^2 = 1.00000001$ rounds to 1, and the computed matrix A^TA will be singular. We have lost all information contained in the last three rows of A. Note that the residual in the first equation is $O(\epsilon^2)$ but $O(1)$ in the others.

Least squares problems of this form are called **stiff**. They occur when the error in some equations (here $x_1 + x_2 + x_3 = 1$) have a much smaller variance than in the others; see Section 8.6.1.

To assess the error in the least squares solution $\bar{x}$ computed by the method of normal equations, we must also account for rounding errors in the Cholesky factorization and in solving the triangular systems. Using the backward error bound given in Theorem 7.5.16 a perturbation analysis shows that provided $2n^{3/2}u\kappa(A^TA) < 0.1$, an upper bound for the error in the computed solution $\bar{x}$ is

$$\|\bar{x} - x\|_2 \leq 2.5n^{3/2}u\kappa(A^TA)\|x\|_2. \tag{8.2.36}$$

As seen in Section 8.2.4, for "small" residual least squares problem the true condition number is approximately $\kappa(A) = \kappa^{1/2}(A^TA)$. In this case, *the system of normal equations can be much worse conditioned than the least squares problem from which it originated.*

Sometimes ill-conditioning is caused by an unsuitable formulation of the problem. Then a different choice of parameterization can significantly reduce the condition number. In approximation problems one should try to use orthogonal, or

nearly orthogonal, base functions. In case the elements in A and b are the original data the ill-conditioning cannot be avoided in this way. In Secs. 8.3 and 8.4 we consider methods for solving least squares problems based on orthogonalization. These methods work directly with A and b and are backwards stable.

In Section 7.7.7 we discussed how the scaling of rows and columns of a linear system $Ax = b$ influenced the solution computed by Gaussian elimination. For a least squares problem $\min_x \|Ax - b\|_2$ a row scaling of $(A,\ b)$ is not allowed since such a scaling would change the exact solution. However, we can scale the columns of A. If we take $x = Dx'$, the normal equations will change into

$$(AD)^T(AD)x' = D(A^TA)Dx' = DA^Tb.$$

Hence, this corresponds to a *symmetric scaling* of rows and columns in A^TA. It is important to note that if the Cholesky algorithm is carried out without pivoting the computed solution is *not* affected by such a scaling, cf. Theorem 7.5.6. This means that even if no explicit scaling is carried out, the rounding error estimate (8.2.36) for the computed solution $\bar{x}$ holds for *all* D,

$$\|D(\bar{x} - x)\|_2 \leq 2.5n^{3/2}u\kappa(DA^TAD)\|Dx\|_2.$$

(Note, however, that scaling the columns changes the norm in which the error in x is measured.)

Denote the *minimum* condition number under a symmetric scaling with a positive diagonal matrix by

$$\kappa'(A^TA) = \min_{D>0} \kappa(DA^TAD). \tag{8.2.37}$$

The following result by van der Sluis [1969] shows the scaling where D is chosen so that in AD all column norms are equal, i.e. $D = \operatorname{diag}(\|a_1\|_2, \ldots, \|a_n\|_2)^{-1}$, comes within a factor of n of the minimum value.

Theorem 8.2.5. *Let $C \in \mathbf{R}^{n\times n}$ be a symmetric and positive definite matrix, and denote by $\mathcal{D}$ the set of $n\times n$ nonsingular diagonal matrices. Then if in C all diagonal elements are equal, and C has at most q nonzero elements in any row, it holds that*

$$\kappa(C) \leq q \min_{D\in\mathcal{D}} \kappa(DCD).$$

As the following example shows, this scaling can reduce the condition number considerably. In cases where the method of normal equations gives surprisingly accurate solution to a seemingly very ill-conditioned problem, the explanation often is that the condition number of the scaled problem is quite small!

Example 8.2.5. The matrix $A \in R^{21\times 6}$ with elements

$$a_{ij} = (i-1)^{j-1}, \qquad 1 \leq i \leq 21, \quad 1 \leq j \leq 6$$

arises when fitting a fifth degree polynomial $p(t) = x_0 + x_1 t + x_2 t^2 + \ldots + x_5 t^5$ to observations at points $x_i = 0, 1, \ldots, 20$. The condition numbers are

$$\kappa(A^T A) = 4.10 \cdot 10^{13}, \qquad \kappa(DA^T AD) = 4.93 \cdot 10^6.$$

where D is the column scaling in Theorem 8.2.5. Thus, the condition number of the matrix of normal equations is reduced by about seven orders of magnitude by this scaling!

8.2.5 Backward Error Analysis

An algorithm for solving the linear least squares problem is said to 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*

$$\min_x \|(A + \delta A)x - (b + \delta b)\|_2, \tag{8.2.38}$$

where $\|\delta A\| \leq \tau \|A\|$, $\|\delta b\| \leq \tau \|b\|$, with τ being a small multiple of the unit roundoff u. We shall see that methods in which the normal equations are explicitly formed cannot be backward stable. On the other hand, many methods based on orthogonal factorizations have been proved to be backward stable.

Any computed solution $\bar{x}$ is called a stable solution if it satisfies (8.2.38). 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 8.2.4.

Many special fast methods exist for solving structured least squares problems, e.g., where A is a Toeplitz matrix. These methods cannot be proved to be backward stable, which is one reason why a solution to the following problem is of interest:

For a consistent linear system we derived in Section 7.5.2 a simple posteriori bounds for the the smallest backward error of a computed solution $\bar{x}$. The situation is more difficult for the least squares problem.

Given an alleged solution $\widetilde{x}$, we want to find a perturbation δA of smallest norm such that $\widetilde{x}$ is the exact solution to the perturbed problem

$$\min_x \|(b + \delta b) - (A + \delta A)x\|_2. \tag{8.2.39}$$

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 (8.2.40). Then the optimal backward error in the Frobenius norm is

$$\eta_F(\widetilde{x}) = \min\{\|\delta A\|_F \mid \widetilde{x} \text{ solves } \min_x \|b - (A + \delta A)x\|_2\}. \tag{8.2.40}$$

Thus, the optimal backward error can be found by characterizing the set of all backward perturbations and then finding an optimal bound, which minimizes the Frobenius norm.

Theorem 8.2.6. *Let $\widetilde{x}$ be an alleged solution and $\widetilde{r} = b - A\widetilde{x} \neq 0$. The optimal backward error in the Frobenius norm is*

$$\eta_F(\widetilde{x}) = \begin{cases} \|A^T\widetilde{r}\|_2/\|\widetilde{r}\|_2, & \text{if } \widetilde{x} = 0, \\ \min\{\eta, \sigma_{min}([A \quad C])\} & \text{otherwise.} \end{cases} \tag{8.2.41}$$

where

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

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

The task of computing $\eta_F(\widetilde{x})$ is thus reduced to that of computing $\sigma_{min}([A \quad C])$. Since this is expensive, approximations that are accurate and less costly have been derived. If a QR factorization of A is available lower and upper bounds for $\eta_F(\widetilde{x})$ can be computed in only $\mathcal{O}(mn)$ operations. Let $r_1 = P_{\mathcal{R}(A)}\widetilde{r}$ be the orthogonal projection of $\widetilde{r}$ onto the range of A. If $\|r_1\|_2 \leq \alpha\|r\|_2$ it holds that

$$\frac{\sqrt{5}-1}{2}\,\widetilde{\sigma}_1 \leq \eta_F(\widetilde{x}) \leq \sqrt{1+\alpha^2}\,\widetilde{\sigma}_1, \tag{8.2.42}$$

where

$$\widetilde{\sigma}_1 = \left\|(A^TA + \eta I)^{-1/2}A^T\widetilde{r}\right\|_2/\|\widetilde{x}\|_2. \tag{8.2.43}$$

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

A simple way to improve the accuracy of a solution $\bar{x}$ computed by the method of normal equations is by fixed precision iterative refinement; see Section 7.7.8. This requires that the data matrix A is saved and used to compute the residual vector $b - A\bar{x}$. In this way information lost when A^TA was formed can be recovered. If also the corrections are computed from the normal equations, we obtain the following algorithm:

Iterative Refinement with Normal Equations:

Set $x_1 = \bar{x}$, and for $s = 1, 2, \ldots$ until convergence do

$$\begin{aligned} r_s &:= b - Ax_s, \qquad R^TR\delta x_s = A^Tr_s, \\ x_{s+1} &:= x_s + \delta x_s. \end{aligned}$$

Here R is computed by Cholesky factorization of the matrix of normal equation A^TA. This algorithm only requires one matrix-vector multiplication each with A and A^T and the solution of two triangular systems. Note that the first step, i.e., for $i = 0$, is identical to solving the normal equations. It can be shown that initially the errors will be reduced with rate of convergence equal to

$$\bar{\rho} = cu\kappa'(A^TA), \tag{8.2.44}$$

where c is a constant depending on the dimensions m, n. Several steps of the refinement may be needed to get good accuracy. (Note that $\bar{\rho}$ is proportional to $\kappa'(A^TA)$ even when no scaling of the normal equations has been performed!)

Example 8.2.6.

If $\kappa'(A^TA) = \kappa(A^TA)$ and $c \approx 1$ the error will be reduced to a backward stable level in p steps if $\kappa^{1/2}(A^TA) \leq u^{-p/(2p+1)}$. (As remarked before $\kappa^{1/2}(A^TA)$ is the condition number for a small residual problem.) For example, with $u = 10^{-16}$, the maximum value of $\kappa^{1/2}(A^TA)$ for different values of p are:

$$10^{5.3}, 10^{6.4}, 10^{8}, \qquad p = 1, 2, \infty.$$

For moderately ill-conditioned problems the normal equations combined with iterative refinement can give very good accuracy. For more ill-conditioned problems the methods based QR factorization described in Sections 8.3 and 8.4 are usually to be preferred.

8.2.6 The Peters–Wilkinson method

Standard algorithms for solving nonsymmetric linear systems $Ax = b$ are usually based on LU factorization with partial pivoting. Therefore it seems natural to consider such factorizations in particular, for least squares problems which are only mildly over- or under-determined, i.e. where $|m - n| \ll n$.

A rectangular matrix $A \in \mathbf{R}^{m\times n}$, $m \geq n$, can be reduced by Gaussian elimination to an upper trapezoidal form U. In general, column interchanges are needed to ensure numerical stability. Usually it will be sufficient to use partial pivoting with a linear independence check. Let $\widetilde{a}_{q,p+1}$ be the element of largest magnitude in column $p+1$. If $|\widetilde{a}_{q,p+1}| < tol$, column $p+1$ is considered to be linearly dependent and is placed last. We then look for a pivot element in column $p+2$, etc.

In the full column rank case, $\operatorname{rank}(A) = n$, the resulting LDU factorization becomes

$$\Pi_1 A \Pi_2 = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = LDU = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} DU, \tag{8.2.45}$$

where $L_1 \in \mathbf{R}^{n\times n}$ is unit lower triangular, D diagonal, and $U \in \mathbf{R}^{n\times n}$ is unit upper triangular and nonsingular. Thus, the matrix L has the same dimensions as A and a lower trapezoidal structure. Computing this factorization requires $\frac{1}{2}n^2(m - \frac{1}{3}n)$ flops.

Using the LU factorization (8.2.45) and setting $\widetilde{x} = \Pi_2^T x$, $\widetilde{b} = \Pi_1 b$, the least squares problem $\min_x \|Ax - b\|_2$ is reduced to

$$\min_y \|Ly - \widetilde{b}\|_2, \qquad DU\widetilde{x} = y. \tag{8.2.46}$$

If partial pivoting by rows is used in the factorization (8.2.45), then L is usually a well-conditioned matrix. In this case the solution to the least squares problem (8.2.46) can be computed from the normal equations

$$L^T L y = L^T \widetilde{b},$$

without substantial loss of accuracy. This is the approach taken by Peters and Wilkinson [484, 1970].

Forming the symmetric matrix L^TL requires $\frac{1}{2}n^2(m-\frac{2}{3}n)$ flops, and computing its Cholesky factorization takes $n^3/6$ flops. Hence, neglecting terms of order n^2, the total number of flops to compute the least squares solution by the Peters–Wilkinson method is $n^2(m-\frac{1}{3}n)$. Although this is always more expensive than the standard method of normal equations, it is a more stable method as the following example shows. The method is particularly suitable for weighted least squares problems; see Section 8.6.1.

Example 8.2.7. (Noble [451, 1976])
Consider the matrix A and its pseudo-inverse

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 1+\epsilon^{-1} \\ 1 & 1-\epsilon^{-1} \end{pmatrix}, \quad A^\dagger = \frac{1}{6}\begin{pmatrix} 2 & 2-3\epsilon^{-1} & 2+3\epsilon^{-1} \\ 0 & 3\epsilon^{-1} & -3\epsilon^{-1} \end{pmatrix}.$$

The (exact) matrix of normal equations is

$$A^TA = \begin{pmatrix} 3 & 3 \\ 3 & 3+2\epsilon^2 \end{pmatrix}.$$

If $\epsilon \le \sqrt{u}$, then in floating point computation $fl(3+2\epsilon^2) = 3$, and the computed matrix $fl(A^TA)$ has rank one. The LU factorization is

$$A = LDU = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},$$

where L and U are well-conditioned. The correct pseudo-inverse is now obtained from

$$A^\dagger = U^{-1}D^{-1}(L^TL)^{-1}L^T = \begin{pmatrix} 1 & -\epsilon \\ 0 & \epsilon \end{pmatrix}\begin{pmatrix} 1/3 & 0 \\ 0 & 1/2 \end{pmatrix}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & -1 \end{pmatrix}.$$

and there is no cancellation.

As seen in Example 8.2.4 weighted least squares problems of the form

$$\min \left\| \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} x - \begin{pmatrix} \gamma b_1 \\ b_2 \end{pmatrix} \right\|, \tag{8.2.47}$$

where $\gamma \gg 1$, are not suited to a direct application of the method of normal equations. If $p = \operatorname{rank}(A_1)$ steps of Gaussian elimination with pivoting are applied to the resulting factorization can be written

$$\Pi_1 \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} \Pi_2 = LDU, \tag{8.2.48}$$

where Π_1 and Π_2 are permutation matrices, and

$$L = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \in \mathbf{R}^{m\times n}, \quad U = \begin{pmatrix} U_{11} & U_{12} \\ & I \end{pmatrix} \in \mathbf{R}^{n\times n}.$$

Here $L_{11} \in \mathbf{R}^{p\times p}$ is unit lower triangular, and $U_{11} \in \mathbf{R}^{p\times p}$ is unit upper triangular. Assuming that A has full rank, D is nonsingular. Then (4.4.1) is equivalent to

$$\min_y \|Ly - \Pi_1 b\|_2, \qquad DU\Pi_2^T x = y.$$

The least squares problem in y is usually well-conditioned, since any ill-conditioning from the weights is usually reflected in D. Therefore, it can be solved by forming the normal equations.

Review Questions

2.1 Give a necessary and sufficient condition for x to be a solution to $\min_x \|Ax - b\|_2$, and interpret this geometrically. When is is the least squares solution x unique? When is $r = b - Ax$ unique?

2.2 What are the advantages and drawbacks with the method of normal equations for computing the least squares solution of $Ax = b$? Give a simple example, which shows that loss of information can occur in forming the normal equations.

2.3 Discuss how the accuracy of the method of normal equations can be improved by (a) scaling the columns of A, (b) iterative refinement.

2.4 Show that the more accurate formula in Example 8.2.1 can be interpreted as a special case of the method (8.5.7)–(8.5.8) for partitioned least squares problems.

2.5 (a) Let $A \in \mathbf{R}^{m\times n}$ with $m < n$. Show that A^TA is singular.
(b) Show, using the SVD, that $\operatorname{rank}(A^TA) = \operatorname{rank}(AA^T) = \operatorname{rank}(A)$.

2.6 Define the condition number $\kappa(A)$ of a rectangular matrix A. What terms in the perturbation of a least squares solution depend on κ and κ^2, respectively?

Problems

2.1 In order to estimate the height above sea level for three points, A,B, and C, the difference in altitude was measured between these points and points D,E, and F at sea level. The measurements obtained form a linear system in the heights x_A, x_B, and x_C of A,B, and C,

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_A \\ x_B \\ x_C \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 1 \end{pmatrix}.$$

Show that the least squares solution and residual vector are

$$x = \frac{1}{4}(5, 7, 12)^T, \qquad r = \frac{1}{4}(-1, 1, 0, 2, 3, -3)^T.$$

and verify that the residual vector is orthogonal to all columns in A.

2.2 (a) Consider the linear regression problem of fitting $y(t) = \alpha + \beta(t - c)$ by the method of least squares to the data

t	1	3	4	6	7
$f(t)$	-2.1	-0.9	-0.6	0.6	0.9

With the (unsuitable) choice $c = 1,000$ the normal equations

$$\begin{pmatrix} 5 & 4979 \\ 4979 & 4958111 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} -2.1 \\ -2097.3 \end{pmatrix}$$

become very ill-conditioned. Show that if the element 4958111 is rounded to $4958 \cdot 10^3$ then β is perturbed from its correct value 0.5053 to -0.1306!

(b) As shown in Example 8.2.1, a much better choice of base functions is shifting with the mean value of t, i.e., taking $c = 4.2$. It is not necessary to shift with the *exact* mean. Show that shifting with 4, the midpoint of the interval $(1, 7)$, leads to a very well-conditioned system of normal equations.

2.3 Denote by x_V the solution to the weighted least squares problem with covariance matrix V. Let x be the solution to the corresponding unweighted problem ($V = I$). Using the normal equations, show that

$$x_V - x = (A^T V^{-1} A)^{-1} A^T (V^{-1} - I)(b - Ax). \tag{8.2.49}$$

Conclude that weighting the rows affects the solution if $b \notin \mathcal{R}(A)$.

2.4 Assume that rank $(A) = n$, and put $\bar{A} = (A, b) \in \mathbf{R}^{m \times (n+1)}$. Let the corresponding cross product matrix, and its Cholesky factor be

$$\bar{C} = \bar{A}^T \bar{A} = \begin{pmatrix} C & d \\ d^T & b^T b \end{pmatrix}, \qquad \bar{R} = \begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix}.$$

Show that the solution x and the residual norm ρ to the linear least squares problem $\min_x \|b - Ax\|_2$ is given by

$$Rx = z, \qquad \|b - Ax\|_2 = \rho.$$

2.5 Let $A \in \mathbf{R}^{m \times n}$ and rank $(A) = n$. Show that the minimum norm solution of the underdetermined system $A^T y = c$ can be computed as follows:

(i) Form the matrix $A^T A$, and compute its Cholesky factorization $A^T A = R^T R$.

(ii) Solve the two triangular systems $R^T z = c$, $Rx = z$, and compute $y = Ax$.

2.6 (a) Let $A = (A_1\ A_2) \in \mathbf{R}^{m\times n}$ be partitioned so that the columns in A_1 are linearly independent. Show that for the matrix of normal equations

$$A^TA = \begin{pmatrix} A_1^TA_1 & A_1^TA_2 \\ A_2^TA_1 & A_2^TA_2 \end{pmatrix}$$

the Schur complement of $A_1^TA_1$ in A^TA can be written in factored form as

$$S = A_2^T(I - A_1(A_1^TA_1)^{-1}A_1^T)A_2,$$

where $P_1 = A_1(A_1^TA_1)^{-1}A_1^T$ is the orthogonal projection onto $\mathcal{R}(A_1)$.
(b) Consider the partitioned least squares problem

$$\min_{x_1,x_2} \left\| (A_1\ A_2) \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - b \right\|_2^2.$$

Show that the solution can be obtained by first solving

$$\min_{x_2} \|(I - P_1)(A_2x_2 - b)\|_2^2,$$

for x_2 and then computing x_1 as the solution to the problem

$$\min_{x_1} \|A_1x_1 - (b - A_2x_2)\|_2^2.$$

2.7 (S. M. Stiegler [555].) In 1793 the French decided to base the new metric system upon a unit, the meter, equal to one 10,000,000th part of the distance from the the north pole to the equator along a meridian arc through Paris. The following famous data obtained in a 1795 survey consist of four measured subsections of an arc from Dunkirk to Barcelona. For each subsection the length of the arc S (in modules), the degrees d of latitude and the latitude L of the midpoint (determined by the astronomical observations) are given.

Segment	Arc length S	latitude d	Midpoint L
Dunkirk to Pantheon	62472.59	2.18910°	49° 56′ 30″
Pantheon to Evaux	76145.74	2.66868°	47° 30′ 46″
Evaux to Carcassone	84424.55	2.96336°	44° 41′ 48″
Carcassone to Barcelona	52749.48	1.85266°	42° 17′ 20″

If the earth is ellipsoidal, then to a good approximation it holds

$$z + y\sin^2(L) = S/d,$$

where z and y are unknown parameters. The meridian quadrant then equals $M = 90(z + y/2)$ and the eccentricity is e is found from $1/e = 3(z/y + 1/2)$. Use least squares to determine z and y and then M and $1/e$.

2.8 Show that if $A, B \in \mathbf{R}^{m\times n}$ and rank $(B) \neq$ rank (A) then it is not possible to bound the difference between $A^\dagger$ and $B^\dagger$ in terms of the difference $B - A$. *Hint*: Use the following example. Let $\epsilon \neq 0$, $\sigma \neq 0$, take

$$A = \begin{pmatrix} \sigma & 0 \\ 0 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} \sigma & \epsilon \\ \epsilon & 0 \end{pmatrix},$$

and show that $\|B - A\|_2 = \epsilon$, $\|B^\dagger - A^\dagger\|_2 > 1/\epsilon$.

2.9 Show that for any matrix A it holds

$$A^\dagger = \lim_{\mu\to 0}(A^TA + \mu^2 I)^{-1}A^T = \lim_{\mu\to 0} A^T(AA^T + \mu^2 I)^{-1}. \tag{8.2.50}$$

2.10 (a) Let $A = (a_1, a_2)$, where $a_1^T a_2 = \cos\gamma$, $\|a_1\|_2 = \|a_2\|_2 = 1$. Hence, γ is the angle between the vectors a_1 and a_2. Determine the singular values and right singular vectors v_1, v_2 of A by solving the eigenvalue problem for

$$A^TA = \begin{pmatrix} 1 & \cos\gamma \\ \cos\gamma & 1 \end{pmatrix}.$$

Then determine the left singular vectors u_1, u_2 from (7.1.33).
(b) Show that if $\gamma \ll 1$, then $\sigma_1 \approx \sqrt{2}$ and $\sigma_2 \approx \gamma/\sqrt{2}$ and

$$u_1 \approx (a_1 + a_2)/2, \qquad u_2 \approx (a_1 - a_2)/\gamma.$$

2.11 The least squares problem $\min_x \|Ax - b\|_2$, where

$$A = \begin{pmatrix} 1 & 1 & 1 \\ \epsilon & & \\ & \epsilon & \\ & & \epsilon \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

is of the form (8.2.47). Compute the factorization $A = LDU$ that is obtained after one step of Gaussian elimination and show that L and U are well-conditioned. Compute the solution from

$$L^TLy = L^Tb, \qquad Ux = D^{-1}y.$$

by solving the system of normal equations for y and solving for x by back-substitution.

8.3 Orthogonal Factorizations

8.3.1 Elementary Orthogonal Matrices

When solving linear equations we made use of elementary elimination matrices of the form $L_j = I + l_j e_j^T$, see (7.2.17). These were used to describe the elementary steps in Gaussian elimination and LU factorization. We now introduce **elementary orthogonal matrices**, which are equal to the unit matrix modified by a matrix of

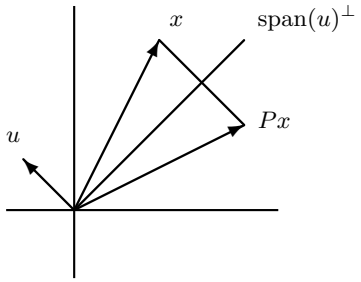

Figure 8.3.1. *A Householder reflection of the vector a.*

rank one. Such matrices are a flexible and very useful tool for constructing matrix algorithms for a variety of problems in linear algebra. This stems from the fact that because they preserve the Euclidean norm, their use leads to numerically stable algorithms.

Elementary orthogonal matrices of the form

$$P = I - 2\frac{uu^T}{u^T u}, \quad u \neq 0, \tag{8.3.1}$$

are called **Householder reflections**. It is easily seen that P is symmetric and setting $\beta = 2/(u^T u)$, we have

$$P^T P = P^2 = I - 2\beta uu^T + \beta^2 u(u^T u)u^T = I.$$

It follows that P is orthogonal and $P^{-1} = P$. For any vector x,

$$Px = (I - \beta uu^T)x = x - \beta(u^T x)u \in \text{span}[x, u].$$

In particular, $Pu = -u$, that is P reverses u and if $x \perp u$ then $Px = x$.

The effect of the transformation Px for a general vector x is to reflect x in the $(m-1)$ dimensional hyper plane with normal vector u; see Figure 8.1.3. This is equivalent to subtracting *twice* the orthogonal projection of x onto u. Note that the normal u is parallel to the vector $(x - Px)$.

The use of elementary reflectors in numerical linear algebra was made popular in matrix computation by Householder [341]. Matrices of the form (8.3.1) are therefore often called **Householder reflectors** and the vector u is called a **Householder vector**.

In applications of Householder reflectors the following construction is central. Given a nonzero vector $x \in \mathbf{R}^m$ we want to construct a plane reflection such that *multiplication by P zeros all components except the first in x*, i.e.,

$$Px = \pm\sigma e_1, \quad \sigma = \|x\|_2. \tag{8.3.2}$$

Here e_1 is the first unit vector and the second equation is a consequence of the fact that P is orthogonal. Multiplying (8.3.2) from the left by P and using $P^2 = I$, we find that

$$Pe_1 = \pm x/\|x\|_2,$$

i.e., this task is equivalent to finding a square orthogonal matrix P with its first column proportional to x. It is easily seen that (8.3.2) is satisfied if we take

$$u = \begin{pmatrix} u_1 \\ x_2 \end{pmatrix} .x \mp \sigma e_1 = \begin{pmatrix} \xi_1 \mp \sigma \\ x_2 \end{pmatrix}, \quad x = \begin{pmatrix} \xi_1 \\ x_2 \end{pmatrix}. \tag{8.3.3}$$

Note that the Householder vector u differs from x only in its first component. A short calculation shows that

$$\begin{aligned} \frac{1}{\beta} = \frac{1}{2}u^T u &= \frac{1}{2}(x \mp \sigma e_1)^T (x \mp \sigma e_1) \\ &= \frac{1}{2}(\sigma^2 \mp 2\sigma\xi_1 + \sigma^2) = \sigma(\sigma \mp \xi_1). \end{aligned}$$

If x is close to a multiple of e_1, then $\sigma \approx |\xi_1|$ and cancellation may lead to a large relative error in β.[29] To avoid this we take

$$u_1 = \operatorname{sign}(\xi_1)(|\xi_1| + \sigma), \qquad \beta = 1/(\sigma(\sigma + |\xi_1|)), \tag{8.3.4}$$

which gives

$$Px = -\operatorname{sign}(\xi_1)\sigma e_1 = \hat{\sigma} e_1.$$

Note that with this choice of sign the vector $x = e_1$ will be mapped onto $-e_1$.

Algorithm 8.1. *Construct Householder reflection.*

```
function [u,beta,sigma] = house(x)
  % HOUSE computes a Householder reflection
  % P = I - beta*u*u' where u = [1; u2]
  % such that P*x = sigma*e_1;
  u = x;
  sigma == norm(x);
  if sigma = 0
    x(1) = -1; beta = 2;
  else
    u1 = abs(x(1)) + sigma;
    beta = 1/(sigma*u1);
    s1 = sign(x(1));
    u(1) = s1*u1;
    sigma = -s1*sigma;
  end
```

The Householder reflection in (8.3.1) does not depend on the scaling of u. It is often more convenient to scale u so that its first component equals 1. Then $P = I - \beta u u^T$, where

$$u = \begin{pmatrix} 1 \\ u_2 \end{pmatrix}, \quad u_2 = \frac{\operatorname{sign}(\xi_1)}{\sigma + |\xi_1|}x_2, \qquad \beta = 1 + \frac{|\xi_1|}{\sigma}. \tag{8.3.5}$$

[29] It is possible to rewrite the formula in (8.3.4) for β so that the other choice of sign does not give rise to numerical cancellation; see Parlett [478, pp. 91].

(Check this!) This scaling has the advantage that we can stably reconstruct β from u_2 using

$$\beta = 2/(u^T u) = 2/(1 + u_2^T u_2).$$

The following algorithm constructs a Householder reflection $P = I - \beta u u^T$, where $u^T e_1 = 1$, such that

$$Px = -\text{sign}\,(\xi_1)\sigma e_1, \quad \sigma = \|x\|_2.$$

where $\|x\|_2 = \sigma \neq 0$ and $x^T e_1 = \xi_1$. Note that if $x = 0$, then we can just take $P = I$, i.e. skip the transformation.

If a matrix $A = (a_1, \ldots, a_n) \in \mathbf{R}^{m\times n}$ is *premultiplied* by P the product is $PA = (Pa_1, \ldots, Pa_n$, where

$$Pa_j = a_j - \beta(u^T a_j)u. \tag{8.3.6}$$

An analogous formula, exists for *postmultiplying* A with P, where P now acts on the *rows* of A. Hence, such products can be computed without explicitly forming P itself. The products

$$PA = A - \beta u(u^T A), \qquad AP = A - \beta(Au)u^T,$$

can be computed in in $4mn$ flops using one matrix–vector product followed by a rank one update.

Householder reflector can be generalized to the complex case A *unitary Householder matrix* has the form

$$P = I - \beta u u^H, \quad \beta = \frac{2}{u^H u}, \quad u \in \mathbf{C}^n. \tag{8.3.7}$$

It is easy to check that P is Hermitian and unitary

$$P = P^H = P^{-1}.$$

Let $z \in \mathbf{C}^n$ and u be such that

$$Pz = \sigma e_1.$$

Then $|\sigma| = \|z\|_2$, but it is in general not possible to have σ real. Since P is Hermitian $z^H P z = \sigma z^H e_1$ must be real. If we denote the first component of z by $z_1 = e^{i\theta_1}|z_1|$, we can take

$$\sigma = \|z\|_2 e^{i\theta_1}, \tag{8.3.8}$$

Then in (8.3.7) $u = z + \sigma e_1$ and

$$\frac{1}{\beta} = \frac{1}{2}(\|z\|_2^2 + 2|\sigma||z_1| + |\sigma|^2) = \|z\|_2(\|z\|_2 + |z_1|). \tag{8.3.9}$$

Note that with unitary matrices of the form $e^{i\theta_1}$ we can reduce a arbitrary vector to the form $ke_1 P$ with k real.

Another useful class of elementary orthogonal transformations are **plane rotations** also called **Givens rotations**. A rotation clockwise through an angle θ in $\mathbf{R}^2$ is represented by the matrix

$$G = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \tag{8.3.10}$$

Note that $G^{-1}(\theta) = G(-\theta)$, and $\det G(\theta) = +1$. Such transformations are also known as **Givens rotations**.[30]

In $\mathbf{R}^n$ the matrix representing a rotation in the plane spanned by the unit vectors e_i and e_j, $i < j$, is the following rank two modification of the unit matrix I_n

$$G_{ij}(\theta) = \begin{array}{c} \\ \\ i \\ \\ j \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccc} & & i & & j & & \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & & s & & \\ & & & \ddots & & & \\ & & -s & & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{pmatrix} \end{array}. \tag{8.3.11}$$

Premultiplying a vector $a = (\alpha_1, \ldots, \alpha_n)^T$ by $G_{ij}(\theta)$, $j > i$, will affect only the components $\alpha : i$ and α_j

$$G_{ij} \begin{pmatrix} \alpha_i \\ \alpha_j \end{pmatrix} = \begin{pmatrix} c\alpha_i + s\alpha_j \\ -s\alpha_i + c\alpha_j \end{pmatrix}. \tag{8.3.12}$$

A plane rotation may be multiplied into a vector at a cost of two additions and four multiplications. We can determine the rotation $G_{ij}(\theta)$ so that jth component becomes zero by taking

$$c = \alpha_i/\sigma, \qquad s = \alpha_j/\sigma. \tag{8.3.13}$$

Then

$$G_{ij} \begin{pmatrix} \alpha_i \\ \alpha_j \end{pmatrix} = \begin{pmatrix} \sigma \\ 0 \end{pmatrix}, \quad \sigma = (\alpha_i^2 + \alpha_j^2)^{1/2} \geq 0.$$

The following procedure computes the Givens rotation G in a way that guards against possible overflow. Note that c and s are only determined up to a common factor ± 1. If a nonnegative σ is required we can just use $-G$ if needed.

Algorithm 8.2.

```
function [c,s,r] = givens(a,b)
% GIVENS computes c and s in a Givens rotation
% Given scalars a and b computes c and s in
% a Givens rotation such that
```

[30] Named after Wallace Givens, who used them in [257] to reduce matrices to simpler.

```
% 0 = -s*a + c*b, and sigma = c*a + s*b
if b == 0
   c = 1.0; s = 0.0; sigma = a;
else if abs(b) > abs(a)
   t = a/b; tt = sqrt(1+t*t);
   s = 1/tt; c = t*s; sigma = tt*b;
else
   t = b/a; tt = sqrt(1+t*t);
   c = 1/tt; s = t*c; sigma = tt*a;
end
```

This algorith requires 5 flops and one square root. No trigonometric functions are involved.

Example 8.3.1.

The polar representation of a complex number $z = x + i\,y$ can be computed by a function call `[c,s,r] = givens(x,y)`. This gives $z = |r|e^{i\theta}$, where $e^{i\theta} = z/|r|$, and

$$z = \begin{cases} r(c + i\,s) & \text{if } \sigma \geq 0, \\ |r|(-c + i\,(-s)) & \text{if } \sigma < 0. \end{cases}$$

Premultiplication of a matrix $A \in R^{m\times n}$ with a Givens rotation G_{ij} will only affect the two *rows* i and j in A, which are transformed according to

$$a_{ik} := ca_{ik} + sa_{jk}, \tag{8.3.14}$$

$$a_{jk} := -sa_{ik} + ca_{jk}, \quad k = 1 : n. \tag{8.3.15}$$

The product requires $4n$ multiplications and $2n$ additions or $6n$ flops. An analogous algorithm, which only affects *columns* i and j, exists for postmultiplying A with G_{ij}.

An arbitrary vector $x \in \mathbf{R}^n$ can be transformed into σe_1 with $\sigma = \|x\|_2^2 \geq 0$, .using a sequence of Givens rotations Let G_{1k}, $k = 2 : m$ be a sequence of Givens rotations, where G_{1k} is determined to zero the kth component in the vector a. Then

$$G_{1n} \ldots G_{13}G_{12}a = \sigma e_1.$$

Note that G_{1k} will not destroy previously introduced zeros. Another possible sequence is $G_{k-1,k}$, $k = m : -1 : 2$, with $G_{k-1,k}$ chosen to zero the kth component.

The matrix G in (8.3.10) has determinant equal to $+1$ We could equally well work with Givens reflectors which have the form

$$G = \begin{pmatrix} -\cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{8.3.16}$$

Using trigonometric identities

$$G = I - (I - G) = I - 2uu^T, \quad u = \begin{pmatrix} \sin(\theta/2) \\ \cos(\theta/2) \end{pmatrix}.$$

which shows the relation to 2×2 Householder matrices.

For complex transformations we can use **unitary** Givens rotations of the form

$$G = \begin{pmatrix} \bar{c} & \bar{s} \\ -s & c \end{pmatrix}, \qquad c = e^{i\gamma}\cos\theta, \qquad s = e^{i\delta}\sin\theta. \tag{8.3.17}$$

From $\bar{c}c + \bar{s}s = \cos^2\theta + \sin^2\theta = 1$ it follows that $G^H G = I$, i.e., G is unitary. Given an arbitrary complex vector $(x_1\ x_2)^T \in \mathbf{C}^2$ we have

$$G\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} \bar{c}z_1 + \bar{s}z_2 \\ -sz_1 + cz_2 \end{pmatrix} = \begin{pmatrix} \sigma \\ 0 \end{pmatrix}, \qquad \sigma^2 = |z_1|^2 + |z_2|^2 > 0, \tag{8.3.18}$$

provided that

$$c = z_1/\sigma, \qquad s = z_2/\sigma.$$

A vector $x \in \mathbf{C}^n$ can be transformed into σe_1, by successive premultiplication with $(n-1)$ unitary plane rotations in planes $(1,2), (1,3), \ldots, (1,n)$. The rotations may be chosen so that

$$\sigma = x^H x = \|x\|_2^2$$

is real and nonnegative.

Example 8.3.2.

Often one needs to represent an orthogonal rotation $Q \in \mathbf{R}^{3\times 3}$, $\det(Q) = 1$ as three successive plane rotations or by the angles of these rotations. The classical choice corresponds to a product of three Givens rotations

$$G_{23}(\phi)G_{12}(\theta)G_{23}(\psi)Q = I.$$

The three angles ϕ, θ, and ψ are called the **Euler angles**.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & c_3 & s_3 \\ 0 & -s_3 & c_3 \end{pmatrix}\begin{pmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & c_1 & s_1 \\ 0 & -s_1 & c_1 \end{pmatrix}\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Here the first Givens rotation $G_{23}(\psi)$ is used to zero the element a_{31}. Next $G_{12}(\theta)$ is used to zero the element a_{21}. Finally, $G_{23}(\phi)$ is used to zero the element a_{32}. The final product is orthogonal and lower triangular and thus must be equal the unit matrix.

A problem with this representation is that the Euler angles may not depend continuously on the data. If Q equals the unit matrix plus small terms then a small perturbation may change an angle as much as 2π. A different set of angles based on zeroing the elements in the order a_{21}, a_{31}, a_{32} is to be preferred. This corresponds to a product of three Givens rotations

$$G_{23}(\phi)G_{13}(\theta)G_{12}(\psi)Q = I.$$

This yields a continuous representation of Q.

Given's rotation are ubiquitous in matrix algorithms and used to transform a matrix to a more compact form. To illustrate the rotation pattern it is convenient to use a Wilkinson diagram. The diagram below shows the zeroing of the (4,2) element in A by a rotation of rows 2 and 4.

$$\begin{matrix} \\ \rightarrow \\ \\ \rightarrow \\ \\ \\ \end{matrix} \begin{pmatrix} \times & \times & \times & \times \\ \otimes & \times & \times & \times \\ \otimes & \times & \times & \times \\ \otimes & \otimes & \times & \times \\ \otimes & \times & \times & \times \\ \otimes & \times & \times & \times \end{pmatrix}.$$

In a Wilkinson diagram $\times$ stands for a (potential) nonzero element and $\otimes$ for a nonzero element that has been zeroed out. The arrows points to the rows that took part in the last rotation.

It is essential to note that the matrix G_{ij} is never explicitly formed, but represented by (i, j) and the two numbers c and s. When a large number of rotations need to be stored it is more economical to store just a single number, from which c and s can be retrieved in a numerically stable way. Since the formula $\sqrt{1-x^2}$ is poor if $|x|$ is close to unity a slightly more complicated method than storing just c or s is needed. In a scheme devised by Stewart [544] one stores the number c or s of smallest magnitude. To distinguish between the two cases one stores the reciprocal of c. More precisely, if $c \neq 0$ we store

$$\rho = \begin{cases} s, & \text{if } |s| < |c|; \\ 1/c, & \text{if } |c| \leq |s| \end{cases}.$$

In case $c = 0$ we put $\rho = 1$, a value that cannot appear otherwise. To reconstruct the Givens rotation, if $\rho = 1$, we take $s = 1$, $c = 0$, and

$$\rho = \begin{cases} sF = \rho, & c = \sqrt{1-s^2}, & \text{if } |\rho| < 1; \\ c = 1/\rho, & s = \sqrt{1-c^2}, & \text{if } |\rho| > 1; \end{cases}$$

It is possible to rearrange the Givens rotations so that it uses only two instead of four multiplications per element and no square root. These modified transformations called "fast" Givens transformations, and are described in Golub and Van Loan [277, 1996, Sec. 5.1.13].

8.3.2 Householder QR Factorization

Orthogonality plays a key role in least squares problems; see Theorem 8.2.1. By using methods directly based on orthogonality the squaring of the condition number that results from forming the normal equations can be avoided.

We first show that any matrix $A \in \mathbf{R}^{m\times n}$ $(m \geq n)$ can be factored into the product of a *square* orthogonal matrix $Q \in \mathbf{R}^{m\times m}$ and an upper triangular matrix $R \in \mathbf{R}^{m\times n}$ with positive diagonal elements.

Theorem 8.3.1. *The Full QR Factorization*

Let $A \in \mathbf{R}^{m\times n}$ with $\operatorname{rank}(A) = n$. Then there is a square orthogonal matrix $Q \in \mathbf{R}^{m\times m}$ and an upper triangular matrix R with positive diagonal elements such that

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix} = (\, Q_1 \quad Q_2\,)\begin{pmatrix} R \\ 0 \end{pmatrix}. \tag{8.3.19}$$

Proof. The proof is by induction on n. Let A be partitioned in the form $A = (a_1, A_2)$, $a_1 \in \mathbf{R}^m$, where $\rho = \|a_1\|_2 > 0$. Put $y = a_1/\rho$, and let $U = (y, U_1)$ be an orthogonal matrix. Then since $U_1^T a_1 = 0$ the matrix $U^T A$ must have the form

$$U^T A = \begin{pmatrix} \rho & y^T A_2 \\ 0 & U_1^T A_2 \end{pmatrix} = \begin{pmatrix} \rho & r^T \\ 0 & B \end{pmatrix},$$

where $B \in \mathbf{R}^{(m-1)\times(n-1)}$. For $n = 1$, A_2 is empty and the theorem holds with $Q = U$ and $R = \rho$, a scalar. If $n > 1$ then $\operatorname{rank}(B) = n - 1 > 0$, and by the induction hypothesis there is an orthogonal matrix $\widetilde{Q}$ such that $\widetilde{Q}^T B = \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}$. (8.3.19) will hold if we define

$$Q = U\begin{pmatrix} 1 & 0 \\ 0 & \widetilde{Q} \end{pmatrix}, \qquad R = \begin{pmatrix} \rho & r^T \\ 0 & \widetilde{R} \end{pmatrix}.$$

□

The proof of this theorem gives a way to compute Q and R, provided we can construct an orthogonal matrix $U = (y,\ U_1)$ given its first column. Several ways to perform this construction using elementary orthogonal transformations were given in Section .

Note that from the form of the decomposition (8.3.19) it follows that R has the same singular values and right singular vectors as A. A relationship between the Cholesky factorization of A^TA and the QR factorization of A is given next.

Lemma 8.3.2.

Let $A \in \mathbf{R}^{m\times n}$ have rank n. Then if the R factor in the QR factorization of A has positive diagonal elements it equals the Cholesky factor of A^TA.

Proof. If $\operatorname{rank}(A) = n$ then the matrix A^TA is nonsingular. Then its Cholesky factor R_C is uniquely determined, provided that R_C is normalized to have a positive diagonal. From (8.3.54) we have $A^TA = R^TQ_1^TQ_1R = R^TR$, and hence $R = R_C$. Since $Q_1 = AR^{-1}$ the matrix Q_1 is also uniquely determined. □

The QR factorization can be written

$$A = (\, Q_1 \quad Q_2\,)\begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1R. \tag{8.3.20}$$

where the square orthogonal matrix $Qin\mathbf{R}^{m\times m}$ has been partitioned as

$$Q = (Q_1, Q_2), \quad Q_1 \in \mathbf{R}^{m\times n}, \quad Q_2 \in \mathbf{R}^{m\times(m-n)}.$$

Here $A = Q_1R$ is called the **thin QR factorization**. From (8.3.20) it follows that the columns of Q_1 and Q_2 form orthonormal bases for the range space of A and its orthogonal complement,

$$\mathcal{R}(A) = \mathcal{R}(Q_1), \qquad \mathcal{N}(A^T) = \mathcal{R}(Q_2), \tag{8.3.21}$$

and the corresponding orthogonal projections are

$$P_{\mathcal{R}(A)} = Q_1Q_1^T, \qquad P_{\mathcal{N}(A^T)} = Q_2Q_2^T. \tag{8.3.22}$$

Note that although the matrix Q_1 in (8.3.20) is uniquely determined, Q_2 can be any orthogonal matrix with range $\mathcal{N}(A^T)$. The matrix Q is *implicitly* defined as a product of Householder or Givens matrices.

The QR factorization of a matrix $A \in \mathbf{R}^{m\times n}$ of rank n can be computed using a sequence of n Householder reflectors. Let $A = (a_1, a_2, \ldots, a_n)$, $\sigma_1 = \|a_1\|_2$, and choose $P_1 = I - \beta_1 u_1 u_1^T$, so that

$$P_1a_1 = P_1\begin{pmatrix}\alpha_1\\ \hat{a}_1\end{pmatrix} = \begin{pmatrix}r_{11}\\ 0\end{pmatrix}, \qquad r_{11} = -\text{sign}\,(\alpha_1)\sigma_1.$$

By (8.3.4) we achieve this by choosing $\beta_1 = 1 + |\alpha_1|/\sigma_1$,

$$u_1 = \begin{pmatrix}1\\ \hat{u}_1\end{pmatrix}, \qquad \hat{u}_1 = \text{sign}\,(\alpha_1)\hat{a}_1/\rho_1, \qquad \rho_1 = \sigma_1\beta_1.$$

P_1 is then applied to the remaining columns $a_2, \ldots, a_n$, giving

$$A^{(2)} = P_1A = \begin{pmatrix} r_{11} & r_{12} & \ldots & r_{1n}\\ 0 & \tilde{a}_{22} & \ldots & \tilde{a}_{2n}\\ \vdots & \vdots & & \vdots\\ 0 & \tilde{a}_{n2} & \ldots & \tilde{a}_{nn}\end{pmatrix}.$$

Here the first column has the desired form and, as indicated by the notation, the first row is the final first row in R. In the next step the $(m-1)\times(n-1)$ block in the lower right corner is transformed. All remaining steps, $k = 2:n$ are similar to the first. Before the kth step we have computed a matrix of the form

$$A^{(k)} = \begin{matrix}\\ {\scriptstyle k-1}\end{matrix}\overset{\overset{k-1}{}}{\begin{pmatrix} R_{11}^{(k)} & R_{12}^{(k)}\\ 0 & \hat{A}^{(k)}\end{pmatrix}}, \tag{8.3.23}$$

where the first $k-1$ rows of $A^{(k)}$ are rows in the final matrix R, and $R_{11}^{(k)}$ is upper triangular. In step k the matrix $a^{(k)}$ is transformed,

$$A^{(k+1)} = P_kA^{(k)}, \qquad P_k = \begin{pmatrix} I_k & 0\\ 0 & \widetilde{P}_k\end{pmatrix}. \tag{8.3.24}$$

Here $\widetilde{P}_k = I - \beta_k u_k u_k^T$ is chosen to zero the elements below the main diagonal in the first column of the submatrix

$$\hat{A}^{(k)} = (a_k^{(k)}, \ldots, a_n^{(k)}) \in \mathbf{R}^{(m-k+1)\times(n-k+1)},$$

i.e. $\widetilde{P}_k a_k^{(k)} = r_{kk} e_1$. With $\sigma_k = \|a_k^{(k)}\|_2$, using (8.3.3), we get $r_{kk} = -\text{sign}\left(a_{kk}^{(k)}\right)\sigma_k$, and

$$\hat{u}_k = \text{sign}\,(\alpha_k^{(k)})\hat{a}_k^{(k)}/\rho_k, \qquad \beta_k = 1 + |a_{kk}|/\sigma_k. \tag{8.3.25}$$

where $\rho_k = \sigma_k \beta_k$. After n steps we have obtained the QR factorization of A, where

$$R = R_{11}^{(n+1)}, \qquad Q = P_1 P_2 \cdots P_n. \tag{8.3.26}$$

Note that the diagonal elements r_{kk} will be positive if $a_k^{(kk)}$ is negative and negative otherwise. Negative diagonal elements may be removed by multiplying the corresponding rows of R and columns of Q by -1.

Algorithm 8.3. *Householder QR Factorization.*

The algorithm computes the full QR factorization of a matrix $A \in \mathbf{R}^{m\times n}$ of rank n such that $Q = P_1 P_2 \cdots P_n$ where

$$P_k = \text{diag}\,(I_{k-1}, \widetilde{P}_k), \qquad \widetilde{P}_k = I - \beta_k u_k u_k^T, \quad k = 1 : n, \tag{8.3.27}$$

are Householder matrices.

```
for k = 1 : n
    [u_k, β_k, r_kk] = house(a_k^(k));
    for j = k + 1, ..., n
        γ_jk = β_k u_k^T a_j^(k);
        r_kj = a_kj^(k) − γ_jk;
        a_j^(k+1) = â_j^(k) − γ_jk û_k;
    end
end
```

If $m = n$ the last step can be skipped. The vectors $\hat{u}_k$ can overwrite the elements in the strictly lower trapezoidal part of A. Thus, all information associated with the factors Q and R can be overwritten A. The vector $(\beta_1, \ldots, \beta_n)$ of length n can be recomputed from

$$\beta_k = \tfrac{1}{2}(1 + \|\hat{u}_k\|_2^2)^{1/2},$$

and therefore need not be saved. In step k the application of the Householder transformation to the active part of the matrix requires $4(m-k+1)(n-k)$ flops. Hence, the total flop count is

$$4\sum_{k=1}^{n-1}(m-k+1)(n-k) = 4\sum_{p=1}^{n-1}((m-n)p + p(p+1)) = 2(mn^2 - n^3/3).$$

If $m = n$ this is $4n^3/3$ flops.

Theorem 8.3.3.

*Let $\bar{R}$ denote the upper triangular matrix R computed by the Householder QR algorithm. Then there exists an exactly orthogonal matrix $\hat{Q} \in \mathbf{R}^{m\times m}$ (*not *the matrix corresponding to exact computation throughout) such that*

$$A + E = \hat{Q}\begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}, \qquad \|e_j\|_2 \le \bar{\gamma}_n \|a_j\|_2, \quad j = 1:n. \tag{8.3.28}$$

where $\bar{\gamma}_n$ is defined in (7.1.79).

As have been stressed before it is usually not advisable to compute the matrix Q in the QR factorization explicitly, even when it is to be used in later computing matrix-vector products. In case that

$$Q = Q^{(0)} = P_1 P_2 \cdots P_n$$

from the Householder algorithm is explicitly required it can be accumulated using the backward recurrence

$$Q^{(n)} = I_m, \qquad Q^{(k-1)} = P_k Q^{(k)}, \quad k = n:-1:1. \tag{8.3.29}$$

which requires $4(mn(m-n) + n^3/3)$ flops. (Note that this is more efficient than the corresponding forward recurrence. By setting

$$Q^{(n)} = \begin{pmatrix} I_n \\ 0 \end{pmatrix}, \quad \text{or} \quad Q^{(n)} = \begin{pmatrix} 0 \\ I_{m-n} \end{pmatrix},$$

For $m = n$ this becomes $4n^3/3$ flops. The matrices Q_1 and Q_2, whose columns span the range space and nullspace of A, respectively, can be similarly computed in $2(mn^2 - n^3/3)$ and $2m^2n - 3mn^2 + n^3$ flops, respectively; see Problem 8.3.7 (b).

Example 8.3.3.

In structured problems the greater flexibility of Givens rotations is an advantage. An important example is the QR factorization of a Hessenberg matrix

$$H_n = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1,n-1} & h_{1,n} \\ h_{21} & h_{22} & \cdots & h_{2,n-1} & h_{2,n} \\ & h_{32} & \cdots & \vdots & \vdots \\ & & \ddots & h_{n-1,n-1} & h_{n-1,n} \\ & & & h_{n,n-1} & h_{n,n} \end{pmatrix} \in \mathbf{R}^{n\times n},$$

which can be computed efficiently using Givens rotations. We illustrate below the first two steps of the algorithm for $n = 5$ in a Wilkinson diagram. In the first step a rotation G_{12} in rows (1,2) is applied to zero out the element h_{21}; in the second step

a rotation G_{23} in rows (2,3) is applied to zero out the next subdiagonal element h_{32}, etc.

$$\begin{matrix}\rightarrow\\ \rightarrow\end{matrix}\begin{pmatrix}\times & \times & \times & \times & \times\\ \otimes & \times & \times & \times & \times\\ & \times & \times & \times & \times\\ & & \times & \times & \times\\ & & & \times & \times\\ & & & & \times\end{pmatrix}\qquad\begin{matrix}\rightarrow\\ \rightarrow\end{matrix}\begin{pmatrix}\times & \times & \times & \times & \times\\ \otimes & \times & \times & \times & \times\\ & \otimes & \times & \times & \times\\ & & \times & \times & \times\\ & & & \times & \times\\ & & & & \times\end{pmatrix}.$$

The arrows points to the rows that took part in the last rotation. After $n-1$ steps all subdiagonal elements have been zeroed out and we have obtained the QR factorization

$$Q^TH = Q\begin{pmatrix}R\\0\end{pmatrix},\quad Q^T = G_{n-1,n}\cdots G_{23}G_{12}. \tag{8.3.30}$$

The first step in the QR factorization takes $6n$ flops and the total work of this QR factorization is only about $3n$ flops. □

It is often advisable to use **column pivoting** in Householder QR factorization. The standard procedure is choose a pivot column that maximizes the diagonal element r_{kk} in the kth step. Assume that after $k-1$ steps we have computed the partial QR factorization

$$A^{(k)} = (P_{k-1}\cdots P_1)A(\Pi_1\cdots\Pi_{k-1}) = \begin{pmatrix}R_{11}^{(k)} & R_{12}^{(k)}\\ 0 & \widetilde{A}^{(k)}\end{pmatrix}, \tag{8.3.31}$$

Then the pivot column in the next step is chosen as a column of largest norm in the submatrix

$$\widetilde{A}^{(k)} = (\tilde{a}_k^{(k)},\ldots,\tilde{a}_n^{(k)}),$$

If $\widetilde{A}^{(k)} = 0$ the algorithm terminates. Otherwise, let

$$s_j^{(k)} = \|\tilde{a}_j^{(k)}\|_2^2,\quad j = k:n. \tag{8.3.32}$$

and interchange columns p and k, where p is the smallest index such that $s_p^{(k)} = \max_{j=k}^n s_j^{(k)}$, This pivoting strategy ensures that the computed triangular factor has the property stated in Theorem 8.3.4. Since the column lengths are invariant under orthogonal transformations the quantities $s_j^{(k)}$ can be updated

$$s_j^{(k+1)} = s_j^{(k)} - r_{jk}^2,\quad j = k+1:n. \tag{8.3.33}$$

We remark that the pivoting rule (8.3.56) is equivalent to maximizing the diagonal element r_{kk}, $k = 1:r$. Therefore, (in exact arithmetic it computes the Cholesky factor that corresponds to using the pivoting (7.3.11) in the Cholesky factorization.

If the column norms in $\tilde{a}^{(k)}$ were recomputed at each stage, then column pivoting would increase the operation count by 50%. Instead the norms of the columns of A can be computed initially, and recursively updated as the factorization proceeds. This reduces the overhead of column pivoting to $0(mn)$ operations. This pivoting strategy can also be implemented in the Cholesky and modified Gram–Schmidt algorithms.

Since column norms are preserved by orthogonal transformations the factor R has the following important property:

Theorem 8.3.4.

Suppose that R is computed by QR factorization with column pivoting. Then the elements in R satisfy the inequalities

$$r_{kk}^2 \geq \sum_{i=k}^{j} r_{ij}^2, \quad j = k+1, \ldots, n. \tag{8.3.34}$$

In particular, $|r_{kk}| \geq |r_{kj}|$, $j > k$, and the diagonal elements form a non-increasing sequence

$$|r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}|. \tag{8.3.35}$$

For any QR factorization it holds that

$$\sigma_1 = \max_{\|x\|=1} \|Rx\|_2 \geq \|Re_1\|_2 = |r_{11}|,$$

and thus $|r_{11}|$ is a lower bound for the largest singular value σ_1 of A. and singular values $1/\sigma_k(A)$. Since R and R^T have the same singular values, we also have

$$\sigma_n = \min_{\|x\|=1} \|R^T x\|_2 \leq \|R^T e_n\|_2 = |r_{nn}|,$$

which gives an upper bound for σ_n. For a triangular matrix satisfying (8.3.34) we also have the upper bound

$$\sigma_1(R) = \|R\|_2 \leq \|R\|_F = \Big(\sum_{i \leq j} r_{ij}^2\Big)^{1/2} \leq \sqrt{n}|r_{11}|.$$

$\sigma_1 \leq n^{1/2} r_{11}$. Using the interlacing property of singular values (Theorem 8.1.17), a similar argument gives the upper bounds

$$\sigma_k(R) \leq (n-k+1)^{1/2}|r_{k,k}|, \quad 1 \leq k \leq n. \tag{8.3.36}$$

If after k steps in the pivoted QR factorization it holds that

$$|r_{k,k}| \leq (n-k+1)^{-1/2}\delta,$$

then $\sigma_k(A) = \sigma_k(R) \leq \delta$, and A has numerical rank at most equal to $k-1$, and we should terminate the algorithm. Unfortunately, the converse is not true, i.e., the

rank is not always revealed by a small element $|r_{kk}|$, $k \le n$. Let R be an upper triangular matrix whose elements satisfy (8.3.34). The best known *lower* bound for the smallest singular value is

$$\sigma_n \ge 3|r_{nn}|/\sqrt{4^n + 6n - 1} \ge 2^{1-n}|r_{nn}|. \tag{8.3.37}$$

(For a proof of this see Lawson and Hanson [399, Ch. 6].)

The lower bound in (8.3.37) can almost be attained as shown in the example below due to Kahan. Then the pivoted QR factorization may not reveal the rank of A.

Example 8.3.4. Consider the upper triangular matrix

$$R_n = \operatorname{diag}(1, s, s^2, \ldots, s^{n-1}) \begin{pmatrix} 1 & -c & -c & \ldots & -c \\ & 1 & -c & \ldots & -c \\ & & 1 & & \vdots \\ & & & \ddots & -c \\ & & & & 1 \end{pmatrix}, \qquad s^2 + c^2 = 1.$$

It can be verified that the elements in R_n satisfies the inequalities in (8.3.37), and that R_n is invariant under QR factorization with column pivoting. For $n = 100$, $c = 0.2$ the last diagonal element of R is $r_{nn} = s^{n-1} = 0.820$. This can be compared with the smallest singular value which is $\sigma_n = 0.368 \cdot 10^{-8}$. If the columns are reordered as $(n, 1, 2, \ldots, n-1)$ and the rank is revealed from the pivoted QR factorization!

The above example has inspired research into alternative column permutation strategies. We return to this problem in Section 8.4.3, where algorithms for the subset selection problem are given.

8.3.3 Least Squares Problems by QR Factorization

We now show how to use the QR factorization to solve the linear least squares problem (8.1.1).

Theorem 8.3.5.
Let $A \in \mathbf{R}^{m\times n}$, $\operatorname{rank}(A) = n$, have the full QR factorization

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

Then the unique solution x to $\min_x ||Ax - b||_2$ and the corresponding residual vector $r = b - Ax$ are given by

$$d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = Q^T b, \quad x = R^{-1} d_1, \quad r = Q \begin{pmatrix} 0 \\ d_2 \end{pmatrix}, \tag{8.3.38}$$

and hence $\|r\|_2 = \|c_2\|_2$.

Proof. Since Q is orthogonal we have

$$\|Ax-b\|_2^2 = \left\|Q^T(Ax-b)\right\|_2^2 = \left\| \begin{pmatrix} Rx \\ 0 \end{pmatrix} - \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} \right\|_2^2 = \|Rx-c_1\|_2^2 + \|d_2\|_2^2.$$

Obviously the right hand side is minimized for $x = R^{-1}c_1$. With c defined by (8.3.38) we have using the orthogonality of Q that

$$b = QQ^Tb = Q_1d_1 + Q_2d_2 = Ax + r.$$

Since $Q_1c_1 = Q_1Rz = Ax$ it follows that $r = Q_2c_2$. ∎

By Theorem 8.3.5, when R and the Householder reflections $P_1, P_2, \ldots, P_n$ have been computed by Algorithm 8.3.2 the least squares solution x and residual r can be computed as follows:

$$d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = P_n \cdots P_2P_1b, \tag{8.3.39}$$

$$Rx = d_1, \qquad r = P_1 \cdots P_{n-1}P_n \begin{pmatrix} 0 \\ d_2 \end{pmatrix}, \tag{8.3.40}$$

and $\|r\|_2 = \|d_2\|_2$. Note that the matrix Q should not be explicitly formed.

The operation count for the Householder QR factorization is $2n^2(m-n/3)$ flops. For $m = n$ both methods require the same work but for $m \gg n$ the Householder QR method is twice as expensive. To compute Q^Tb and solve $Rx = d_1$ requires $4n(m-n/4)$ flops. In some cases only $\|r\|_2$ is required. If r is needed this requires another $4n(m-n/2)$ flops. This can be compared to the operation count for the method of normal equations, which requires $(mn^2 + n^3/3)$ flops for the factorization and $2n(m+n)$ for each right hand side.

The Householder algorithm is backward stable for computing both the solution x and the residual $r = b - Ax$. This means that the computed residual $\bar{r}$ satisfies

$$(A+E)^T\bar{r} = 0, \quad \|E\|_2 \le cu\|A\|_2, \tag{8.3.41}$$

for some constant $c = c(m,n)$. Hence, $A^T\bar{r} = -E^T\bar{r}$, and

$$\|A^T\bar{r}\|_2 \le cu\|\bar{r}\|_2\|A\|_2. \tag{8.3.42}$$

Note that this is much better result than if the residual is computed as

$$\widetilde{r} = fl(b - fl(Ax)) = fl\left(\begin{pmatrix} b & A \end{pmatrix} \begin{pmatrix} 1 \\ -x \end{pmatrix} \right),$$

even when x is the *exact* least squares solution. Using (2.3.13) and $A^Tr = 0$

$$|A^T\widetilde{r}| < \gamma_{n+1}|A^T|(|b| + |A||x|).$$

we obtain the normwise bound

$$\|A^T\bar{r}\|_2 \le n^{1/2}\gamma_{n+1}\|A\|_2(\|b\|_2 + n^{1/2}\|A\|_2\|x\|_2),$$

This is much weaker than (8.3.42), in particular when $\|\bar{r}\|_2 \ll \|b\|_2$.

The method in Theorem 8.3.5 generalizes easily to a method for solving the augmented system (8.1.10). Recall that the solution to the augmented system gives the solution to the two optimizations problems (8.1.11)–(8.1.12).

Theorem 8.3.6.

Let the full QR factorization of $A \in \mathbf{R}^{m\times n}$ with $\operatorname{rank}(A) = n$, be given by $A = (\,Q_1 \quad Q_2\,)\begin{pmatrix} R \\ 0 \end{pmatrix}$. Then the solution to the augmented system

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

can be computed as

$$z = R^{-T}c, \qquad \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = Q^T b, \tag{8.3.43}$$

$$x = R^{-1}(d_1 - z) \qquad y = Q\begin{pmatrix} z \\ d_2 \end{pmatrix}, \tag{8.3.44}$$

Further, we have

$$\min \|b - y\|_2 = \min \|Q^T b - Q^T y\|_2 = \|d_1 - z\|_2.$$

Proof. Using the QR factorization the subsystems $y + Ax = b$ and $A^T y = c$ of the augmented system can be written

$$y + Q\begin{pmatrix} R \\ 0 \end{pmatrix} x = b, \qquad (\,R^T \quad 0\,)\,Q^T y = c.$$

Multiplying the first system by Q^T and the second by R^{-T} gives

$$Q^T y + \begin{pmatrix} R \\ 0 \end{pmatrix} x = Q^T b, \qquad (\,I_n \quad 0\,)\,Q^T y = R^{-T}c, \qquad \min \|y\|_2 = \|z\|_2.$$

Using the second equation to eliminate the first n components of $Q^T y$ in the first euqtion, we can then solve for x. The last $m - n$ components of $Q^T y$ are obtained form the last $m - n$ equations in the first block. □

Note that either x or y can be computed without the other. Thus the algorithm (8.3.43)–(8.3.43) can be used to solve either the linear least squares problem (8.1.11) or the conditional least squares problem (8.1.12).

In the special case that $b = 0$ the algorithm simplifies to

$$z = R^{-T}c, \qquad y = Q\begin{pmatrix} z \\ 0 \end{pmatrix}, \tag{8.3.45}$$

and solves the minimum norm problem $\min \|y\|_2$ subject to $A^T y = c$.

The Householder QR algorithm, and the resulting method for solving the least squares problem are backwards stable, both for x and r, and the following result holds.

Theorem 8.3.7.

*Let $\bar{R}$ denote the computed R. Then there exists an exactly orthogonal matrix $\widetilde{Q} \in \mathbf{R}^{m\times m}$ (*not *the matrix corresponding to exact computation throughout) such that*

$$A + \Delta A = \widetilde{Q}\begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}, \quad \|\Delta A\|_F \le c\gamma_{mn}\|A\|_F, \tag{8.3.46}$$

where c is a small constant. Further, the computed solution $\bar{x}$ is the exact solution of a slightly perturbed least squares problem

$$\min_x \|(A + \delta A)x - (b + \delta b)\|_2,$$

where the perturbation can be bounded in norm by

$$\|\delta A\|_F \le c\gamma_{mn}\|A\|_F, \qquad \|\delta b\|_2 \le c\gamma_{mn}\|b\|_2, \tag{8.3.47}$$

Proof. See Higham [328, Theorem 19.5]. □

The column pivoting strategy suggested earlier may not be appropriate, when we are given one vector b, which is to be approximated by a linear combination of as few columns of the matrix A as possible. This occurs, e.g., in regression analysis, where each column in A corresponds to one factor. Even when the given b is parallel to one column in A it could happen that we had to complete the full QR factorization of A before this fact was discovered. Instead we would like at each stage to select the column that maximally reduces the sum of squares of the residuals.

8.3.4 Gram–Schmidt QR Factorization

We start by considering the case of orthogonalizing *two* given linearly independent vectors a_1 and a_2 in $\mathbf{R}^n$. We set

$$q_1 = a_1/r_{11}, \quad r_{11} = \|a_1\|_2.$$

and seek a unit vector $q_2 \in \operatorname{span}[a_1, a_2]$ such that $q_1^T q_2 = 0$. By subtracting from a_2 its orthogonal projection onto q_1, we get

$$\hat{q}_2 = (I - q_1 q_1^T)a_2 = a_2 - r_{12}q_1, \quad r_{12} = q_1^T a_2, \tag{8.3.48}$$

We have $\hat{q}_2 \neq 0$, since otherwise a_2 would not be linearly independent of a_1.It is easily verified that $q_1^T\hat{q}_2 = q_1^T a_2 - r_{12}q_1^T q_1 = 0$. Thus, we can set

$$q_2 = \hat{q}_2/r_{22}, \quad r_{22} = \|\hat{q}_2\|_2.$$

Since q_1 and q_2 both are linear combinations of a_1 and a_2, they span the same subspace of $\mathbf{R}^n$. Further, we have the relation

$$(\, a_1 \quad a_2 \,) = (\, q_1 \quad q_2 \,)\begin{pmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{pmatrix}.$$

Clearly, the process works also for complex vectors $a_k \in \mathbf{C}^m$,

The above algorithm can be extended to orthogonalizing any sequence of linearly independent vectors $a_1, \ldots, a_n$ in $\mathbf{R}^m$ $(m \geq n)$. This process, known as **Gram–Schmidt orthogonalization**,[31] [32] uses elementary orthogonal projections. After step k, $k = 1 : n$, orthonormal vectors $q_1, \ldots, q_k$ have been computed such that

$$\operatorname{span}[q_1, \ldots, q_k]. = \operatorname{span}[a_1, \ldots, a_k] \tag{8.3.49}$$

We give two variants of the algorithm. Although these only differ in which order the operations are carried out, they have greatly different numerical stability.

Classical Gram–Schmidt (CGS):

For $k = 1 : n$ orthogonalize a_k against $q_1, \ldots, q_{k-1}$:

$$\hat{q}_k = a_k - \sum_{i=1}^{k-1} r_{ik} q_i, \quad r_{ik} = q_i^T a_k, \quad i = 1 : k-1, \tag{8.3.50}$$

and normalize

$$r_{kk} = \|\hat{q}_k\|_2, \qquad q_k = \hat{q}_k / r_{kk}. \tag{8.3.51}$$

The unnormalized vector $\widetilde{q}_k$ is just the orthogonal projection a_k onto the complement of $\operatorname{span}[a_1, \ldots, a_{k-1}]$.

Modified Gram–Schmidt (MGS)

Set $a_j^{(1)} = a_j$, $j = 1 : n$. For $k = 1 : n$, normalize

$$r_{kk} = \|a_k^{(k)}\|_2, \qquad q_k = a_k^{(k)} / r_{kk}.$$

Orthogonalize $a_j^{(k)}$, $j = k+1 : n$, against q_k:

$$a_j^{(k+1)} = a_j^{(k)} - r_{kj} q_k, \quad r_{kj} = q_k^T a_j^{(k)}, \quad j = k+1 : n. \tag{8.3.52}$$

At the beginning of step k in the MGS algorithm we have computed

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

where $a_k^{(k)}, \ldots, a_n^{(k)}$ are orthogonal to $q_1, \ldots, q_{k-1}$.

In CGS the orthogonalization of a_k can be written

$$\hat{q}_k = (I - Q_{k-1} Q_{k-1}^T) a_k, \quad Q_{k-1} = (q_1, \ldots, q_{k-1}).$$

What makes the difference in numerical stability is that in MGS *the projections $r_{ik} q_i$ are subtracted from a_k as soon as they are computed*, which corresponds to computing

$$\hat{q}_k = (I - q_{k-1} q_{k-1}^T) \cdots (I - q_1 q_1^T) a_k. \tag{8.3.53}$$

[31] Jørgen Pedersen Gram (1850–1916), Danish mathematician, Gram worked for Hafnia Insurance Company and made contributions to probability and numerical analysis.

[32] Erhard Schmidt (1876–1959), German mathematician obtained his Ph.D. from the University of Göttingen in 1905 under Hilbert's supervision. In 1920 Schmidt was appointed to the post of Director of the Institute of Applied Mathematics at the University of Berlin in 1917.

Clearly, for $n = 2$ MGS and CGS are identical. For $k > 2$ these two expressions are identical only if the $q_1, \ldots, q_{k-1}$ are orthogonal. As describe in CGS the matrix R is determined row by row, in MGS column by column. The rowwise order used in MGS is convenient when column pivoting is to be performed. However, a columnwise MGS version is also possible for applications where the columns are a_k are generated one at a time.

In matrix terms the Gram–Schmidt process uses elementary column operations to transform the matrix A into an orthogonal matrix Q to give R and Q (explicitly) in the thin QR factorization. This can be contrasted with the Householder QR factorization, where the matrix A is premultiplied by a sequence of elementary orthogonal transformations to produce R and Q (in product form) in the full QR factorization.

Theorem 8.3.8.

Let the matrix $A = (a_1, a_2, \ldots, a_n) \in \mathbf{C}^{m\times n}$ have linearly independent columns. Then the Gram–Schmidt algorithm computes a matrix $Q_1 \in \mathbf{R}^{m\times n}$ with orthonormal columns $Q_1^H Q_1 = I_n$ and an upper triangular matrix $R \in \mathbf{C}^{n\times n}$ with real positive diagonal elements such that

$$A = (q_1, q_2, \ldots, q_n) \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \vdots \\ & & & r_{nn} \end{pmatrix} \equiv Q_1 R. \tag{8.3.54}$$

Proof. Note that $\hat{q}_k \neq 0$, since otherwise a_k is a linear combination of the vectors $a_1, \ldots, a_{k-1}$, which contradicts the assumption. Combining (8.3.50) and (8.3.51) we obtain

$$a_k = r_{kk} q_k + \sum_{i=1}^{k-1} r_{ik} q_i = \sum_{i=1}^{k} r_{ik} q_i, \quad k = 1 : n,$$

which is equivalent to (8.3.54). Since the vectors q_k are mutually orthogonal by construction, the theorem follows. □

Although mathematically equivalent, MGS has greatly superior numerical properties compared to CGS, and is therefore usually to be preferred. (We say that two formulas or algorithms are mathematically equivalent if they produce the same result in exact arithmetic.) In fact, because it is more convenient, MGS had almost always been used in practice long before its superior numerical stability was appreciated.

Algorithm 8.4. *Modified Gram–Schmidt.*

Given $A \in R^{m\times n}$ with $\operatorname{rank}(A) = n$ the following algorithm computes the factorization $A = Q_1 R$:

```
function [Q,R] = mgs(A);
[m,n] = size(A);
Q = A;
for k = 1:n
   R(k,k) = norm(Q(:,k));
   Q(:,k) = Q(:,k)/R(k,k);
   for j = k+1:n
      R(k,j) = Q(:,k)'*Q(:,j);
      Q(:,j) = Q(:,j) - Q(:,k)*R(k,j);
   end
end
```

The operations in Algorithm 8.3.4 can be sequenced so that the elements in R are computed in a columnwise fashion. However, the rowwise version given above is more suitable if column pivoting is to be performed. Gram–Schmidt orthogonalization requires approximately $2mn^2$ flops, which is slightly more than the $2(mn^2 - n^3/3)$ needed for Householder QR factorization.

It can be shown by an elementary error analysis that if $\bar{Q}_1$ and $\bar{R}$ are computed by MGS, then it holds that

$$A + E = \bar{Q}_1\bar{R}, \quad \|E\|_2 \le c_0 u\|A\|_2. \tag{8.3.55}$$

Thus, the product $\bar{Q}_1\bar{R}$ accurately represents A.

Assume that the columns $a_1, \ldots, a_{k-1}$ are linearly independent but a_k is a linear combination of $a_1, \ldots, a_{k-1}$. Then $a_k^{(k-1)} = 0$ and the Gram–Schmidt process breaks down. But if $\text{rank}\,(A) > k$ there must be a vector a_j, $j > k$, which is linearly independent of $a_1, \ldots, a_{k-1}$.We can then interchange columns k and j and proceed until all remaining columns are linearly dependent on the computed q vectors.

This suggest that we augment the MGS method with **column pivoting** as follows. Compute

$$\|a_j^{(k)}\|_2, \quad j = k : n, \tag{8.3.56}$$

and let the maximum be $|a_j^{(p)}\|_2$.(If there are more than one maximum, choose the first.) If this maximum is zero, then all columns $a_k, \ldots, a_n$ are linearly dependent and the reduction is complete. Otherwise interchange columns p and k and proceed. After the kth step we can cheaply update the column norms using

$$\|a_j^{(k+1)}\|_2 = \|a_j^{(k)}\|_2 - r_{kj}^2, \quad j = k+1 : n.$$

With column pivoting MGS can be used also when the matrix A has linearly dependent columns. If $\text{rank}\,(A) = r$ it will compute a factorization of the form

$$A\Pi = QR, \quad Q \in \mathbf{R}^{m\times r}, \quad R = (\, R_{11} \quad R_{12}\,) \in \mathbf{R}^{r\times n}, \tag{8.3.57}$$

where Π is a permutation matrix, $Q^TQ = I$, and R_{11} is upper triangular and nonsingular. Indeed, MGS with column pivoting is a good algorithm for determining the rank of a given matrix A.

8.3.5 Loss of Orthogonality in GS and MGS

In some applications it may be essential that the computed columns of Q are orthogonal to working accuracy . For example, this is the case when MGS is used in algorithms for computing eigenvectors to symmetric matrices.

We will now use the standard model for floating point computation, and the basic results in Section 2.3.2 to analyze the rounding errors when the Gram–Schmidt process is applied to orthogonalize two linearly independent vectors a_1 and a_2 in $\mathbf{R}^n$. For the *computed* scalar product $\bar{r}_{12} = fl(q_1^T a_2)$ we get

$$|\bar{r}_{12} - r_{12}| < \gamma_m \|a_2\|_2, \quad \gamma_m = \frac{mu}{1 - mu/2},$$

where u is the unit roundoff. Using $|r_{12}| \leq \|a_2\|_2$ we obtain for $\bar{q}_2 = fl(a_2 - fl(\bar{r}_{12} q_1))$

$$\|\bar{q}_2 - \hat{q}_2\|_2 < \gamma_{m+2} \|a_2\|_2.$$

Since $q_1^T \hat{q}_2 = 0$, it follows that $|q_1^T \bar{q}_2| < \gamma_{m+2} \|a_2\|_2$ and the loss of orthogonality

$$\frac{|q_1^T \bar{q}_2|}{\|\bar{q}_2\|_2} \approx \frac{|q_1^T \bar{q}_2|}{\|\hat{q}_2\|_2} < \gamma_{m+2} \frac{\|a_2\|_2}{\|\widetilde{q}_2\|_2} = \frac{\gamma_{m+2}}{\sin \phi(q_1, a_2)}, \tag{8.3.58}$$

is proportional to $\phi(q_1, a_2)$, the angle between q_1 and a_2.

We conclude that in the Gram–Schmidt process a severe loss of orthogonality may occur whenever cancellation takes place in subtracting the orthogonal projection on q_i from $a_k^{(i)}$, that is when

$$a_j^{(k+1)} = (I - q_k q_k^T) a_j^{(k)}, \qquad \|a_k^{(i+1)}\|_2 \ll \alpha \|a_k^{(i)}\|_2. \tag{8.3.59}$$

Example 8.3.5.

For the extremely ill-conditioned matrix

$$A = (a_1, a_2) = \begin{pmatrix} 1.2969 & 0.8648 \\ 0.2161 & 0.1441 \end{pmatrix},$$

in Example 7.5.4 the Gram–Schmidt algorithm in IEEE double precision gives

$$q_1 = \begin{pmatrix} 0.98640009002732 \\ 0.16436198585466 \end{pmatrix},$$

$r_{12} = q_1^T a_2 = 0.87672336001729$, Subtracting the orthogonal projection onto q_1 we get

$$\hat{q}_2 = a_2 - r_{12} q_1 = \begin{pmatrix} -0.12501091273265 \\ 0.75023914025696 \end{pmatrix} 10^{-8}.$$

Normalizing this vector gives

$$q_2 = \begin{pmatrix} -0.16436196071471 \\ 0.98640009421635 \end{pmatrix},$$

Table 8.3.1. *Loss of orthogonality and CGS and MGS.*

k	$\kappa(A_k)$	$\|I_k - Q_C^T Q_C\|_2$	$\|I_k - Q_M^T Q_M\|_2$
1	1.000e+00	1.110e-16	1.110e-16
2	1.335e+01	2.880e-16	2.880e-16
3	1.676e+02	7.295e-15	8.108e-15
4	1.126e+03	2.835e-13	4.411e-14
5	4.853e+05	1.973e-09	2.911e-11
6	5.070e+05	5.951e-08	3.087e-11
7	1.713e+06	2.002e-07	1.084e-10
8	1.158e+07	1.682e-04	6.367e-10
9	1.013e+08	3.330e-02	8.779e-09
10	1.000e+09	5.446e-01	4.563e-08

and

$$R = \begin{pmatrix} 1.31478090189963 & 0.87672336001729 \\ 0 & 0.00000000760583 \end{pmatrix}.$$

Severe cancellation has taken place when computing $\hat{q}_2$. This leads to a serious loss of orthogonality between q_1 and q_2:

$$q_1^T q_2 = 2.5486557 \cdot 10^{-8},$$

The loss of orthogonality is roughly equal to a factor $\kappa(A) \approx 10^{-8}$.

Due to round-off there will be a gradual (sometimes catastrophic) loss of orthogonality in Gram–Schmidt orthogonalization. In this respect CGS and MGS behave very differently for $n > 2$. (Recall that for $n = 2$ MGS and CGS are the same.) For MGS the loss of orthogonality occurs in a predictable manner and is proportional to the condition number $\kappa(A)$.

Theorem 8.3.9.

Let $\bar{Q}$ and $\bar{R}$ denote the factors computed by the MGS algorithm. Then there are constants $c_i = c_i(m, n)$, $i = 1, 2$, such that if $c_1 \kappa(A) u < 1$,

$$\|I - \bar{Q}_1^T \bar{Q}_1\|_2 \leq \frac{c_1}{1 - c_1 u \kappa_2(A)} u \kappa_2(A). \tag{8.3.60}$$

Proof. See Björck [57]. □

The computed vectors q_k from CGS depart from orthogonality much more rapidly. After a small modification of the algorithm an upper bound for the loss of orthogonality proportional to κ^2 has been proved. Even computing $Q_1 = AR^{-1}$, where R is determined by Cholesky factorization often gives better orthogonality than CGS; see the example below.

Example 8.3.6.

A matrix $A \in \mathbf{R}^{50\times 10}$ was generated by computing

$$A = U\,DV^T, \quad D = \operatorname{diag}(1, 10^{-1}, \ldots, 10^{-9})$$

with U and V orthonormal matrices. Hence, A has singular values $\sigma_i = 10^{-i+1}$, $i = 1:10$, and $\kappa(A) = 10^9$. Table 8.3.1 shows the condition number of $A_k = (a_1, \ldots, a_k)$ and the loss of orthogonality in CGS and MGS after k steps as measured by $\|I_k - Q_k^T Q_k\|_2$. Note that for MGS the loss of orthogonality is more gradual than for CGS and proportional to $\kappa(A_k)$,

In several applications it is important that the computed $\bar{Q}_1$ is orthogonal to working accuracy. We shall call this the **orthogonal basis problem**. To achieve this we can **reorthogonalize** the computed vectors in the Gram–Schmidt algorithm, This must be carried out whenever substantial cancellation occurs when subtracting a projection.

Assume that we are given a matrix

$$Q_1 = (q_1, \ldots, q_{k-1}), \quad \|q_j\|_2 = 1, \quad j = 1:k-1.$$

Adding the new vector a_k, we want to compute a vector $\hat{q}_k$ such that

$$\hat{q}_k \perp \operatorname{span}(Q_1), \qquad \hat{q}_k \in \operatorname{span}(Q_1, a_k).$$

If $(Q_1 a_k)$ has full numerical rank and Q_1 is accurately orthogonal, then it has been proved that *one reorthogonalization will always suffice*. For $k = 2$ this result is due to Kahan and Parlett; see Parlett [478, Sec. 6.9]. As an example, reorthogonalizing the computed vector $a_2^{(2)}$ in Example 8.3.5 against q_1 gives

$$q_1^T q_2 = 2.5486557 \cdot 10^{-8}, \qquad \widetilde{q}_2 = \begin{pmatrix} -0.16436198585466 \\ 0.98640009002732 \end{pmatrix}.$$

The vector $\widetilde{q}_2$ is exactly orthogonal to q_1.

If A has full numerical rank and you reorthogonalize as you go along, then for both CGS and MGS one reorthogonalization suffices to produce vectors that are orthonormal to machine precision. Hence, reorthogonalization will double the cost. An MGS algorithm with reorthogonalization is given below. Here a columnwise version is used and the reorthogonalization against previously computed vectors is performed in backward order.

Algorithm 8.5. *MGS with reorthogonalization.*

```
function [Q,R] = mgs2(A);
[m,n] = size(A);
Q = A; R = zeros(n);
for k = 1:n
   for i = [1:k-1,k-1:-1:1]
```

```
        V = Q(:,i)'*Q(:,k);
        Q(:,k) = Q(:,k) - V*Q(:,i);
        R(i,k) = R(i,k) + V;
    end
    R(k,k) = norm(Q(:,k));
    Q(:,k) = Q(:,k)/R(k,k);
end
```

In **selective reorthogonalization** a test is performed at each step to see whether or not it is necessary to reorthogonalize. Usually reorthogonalization is carried out whenever

$$\|\bar{q}_k\|_2 < \alpha \|a_k\|_2. \tag{8.3.61}$$

for some chosen tolerance α. When α is large reorthogonalization will occur more frequently and the orthogonality will be good. If α is small, reorthogonalization will be rarer, but the orthogonality less good. Typically α is chosen in the range $0.1 \leq \alpha \leq 1/\sqrt{2}$. In practice, the value $\alpha = 1/\sqrt{2}$ is often used. In selective reorthogonalization the columns of Q_1 may not be orthogonal to working accuracy. Then it might be necessary to reorthogonalize more than once.

8.3.6 MGS as a Householder Method

A key observation for understanding the numerical stability of MGS algorithms is the surprising result that it can be interpreted as a Householder QR factorization of the matrix A augmented with a square matrix of zero elements on top.[33] This is not true only in theory, but in the presence of rounding errors as well. We first look at the theoretical result.

Let $A \in \mathbf{R}^{m\times n}$ have rank n and consider the two QR factorizations

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad \widetilde{A} = \begin{pmatrix} O \\ A \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}, \tag{8.3.62}$$

$Q \in \mathbf{R}^{m\times m}$ and $P \in \mathbf{R}^{(n+m)\times(n+m)}$ are orthogonal matrices. If the upper triangular matrices R and $\widetilde{R}$ are both chosen to have positive diagonal elements then by uniqueness $R = \widetilde{R}$. In exact computation $P_{21} = Q_1$. The last m columns of P are arbitrary up to an $m \times m$ multiplier.

The important result is that the MGS QR factorization is also *numerically* equivalent to Householder applied to the augmented matrix, To see this, recall that the Householder transformation $Px = e_1\rho$ uses

$$P = I - 2vv^T/\|v\|_2^2, \quad v = x - \rho e_1, \quad \rho = \pm\|x\|_2.$$

If the second factorization in (8.3.62) is obtained using Householder transformations, then

$$P^T = P_n \cdots P_2 P_1, \quad P_k = I - 2\hat{v}_k\hat{v}_k^T/\|\hat{v}_k\|_2^2, \quad k = 1:n, \tag{8.3.63}$$

[33] This observation was made by Charles Sheffield, apparently when comparing Fortran code for Householder and MGS QR factorization.

where the vectors $\hat{v}_k$ are described below. Now, from MGS applied to $A^{(1)} = A$, $r_{11} = \|a_1^{(1)}\|_2$, and $a_1^{(1)} = q_1' = q_1 r_{11}$. The first Householder transformation applied to the augmented matrix

$$\widetilde{A}^{(1)} \equiv \begin{pmatrix} O_n \\ A^{(1)} \end{pmatrix}, \qquad \widetilde{a}_1^{(1)} = \begin{pmatrix} 0 \\ a_1^{(1)} \end{pmatrix},$$
$$\hat{v}_1^{(1)} \equiv \begin{pmatrix} -e_1 r_{11} \\ q_1' \end{pmatrix} = r_{11} v_1, \qquad v_1 = \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix}.$$

But $\|v_1\|_2^2 = 2$, giving

$$P_1 = I - 2\hat{v}_1 \hat{v}_1^T / \|\hat{v}_1\|_2^2 = I - 2 v_1 v_1^T / \|v_1\|_2^2 = I - v_1 v_1^T,$$

and

$$P_1 \widetilde{a}_j^{(1)} = \widetilde{a}_j^{(1)} - v_1 v_1^T \widetilde{a}_j^{(1)} = \begin{pmatrix} 0 \\ a_j^{(1)} \end{pmatrix} - \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix} q_1^T a_j^{(1)} = \begin{pmatrix} e_1 r_{1j} \\ a_j^{(2)} \end{pmatrix},$$

so

$$P_1 \widetilde{A}^{(1)} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \\ 0 & a_2^{(2)} & \cdots & a_n^{(2)} \end{pmatrix}.$$

Clearly the first row is *numerically* the same as the first row in R produced in the first step of MGS on A. Also the vectors $a_j^{(2)}$ are the same. We see that the next Householder transformation produces the second row of R and $a_j^{(3)}$, $j = 3 : n$, just as in MGS. Carrying on this way we can conclude that this Householder QR is numerically equivalent to MGS applied to A, and that every P_k is effectively defined by Q_1, since

$$P_k = I - v_k v_k^T, \qquad v_k = \begin{pmatrix} -e_k \\ q_k \end{pmatrix}, \quad k = 1 : n. \tag{8.3.64}$$

From the numerical equivalence it follows that the backward error analysis for the Householder QR factorization of the augmented matrix can also be applied to the modified Gram–Schmidt algorithm on A. Let $\bar{Q}_1 = (\bar{q}_1, \ldots, \bar{q}_n)$ be the matrix of vectors computed by MGS, and for $k = 1, \ldots, n$ define

$$\bar{v}_k = \begin{pmatrix} -e_k \\ \bar{q}_k \end{pmatrix}, \qquad \bar{P}_k = I - \bar{v}_k \bar{v}_k^T, \qquad \bar{P} = \bar{P}_1 \bar{P}_2 \ldots \bar{P}_n, \tag{8.3.65}$$
$$\widetilde{q}_k = \bar{q}_k / \|\bar{q}_k\|_2, \quad \widetilde{v}_k = \begin{pmatrix} -e_k \\ \widetilde{q}_k \end{pmatrix}, \quad \widetilde{P}_k = I - \widetilde{v}_k \widetilde{v}_k^T, \quad \widetilde{P} = \widetilde{P}_1 \widetilde{P}_2 \ldots \widetilde{P}_n.$$

Then $\bar{P}_k$ is the computed version of the Householder matrix applied in the kth step of the Householder QR factorization of the augmented matrix and $\widetilde{P}_k$ is its orthonormal equivalent, so that $\widetilde{P}_k^T \widetilde{P}_k = I$. From the error analysis for Householder QR (see Theorem 8.3.7) it follows that for $\bar{R}$ computed by MGS,

$$\begin{pmatrix} E_1 \\ A + E_2 \end{pmatrix} = \widetilde{P} \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}, \quad \bar{P} = \widetilde{P} + E',$$

where

$$\|E_i\|_2 \le c_i u \|A\|_2, \quad i = 1, 2; \quad \|E'\|_2 \le c_3 u, \tag{8.3.66}$$

and c_i are constants depending on m, n and the details of the arithmetic. Using this result it can be shown (see Björck and Paige [65]) that there exist an *exactly orthonormal matrix* $\hat{Q}_1$ and E such that

$$A + E = \hat{Q}_1 \bar{R}, \qquad \hat{Q}_1^T \hat{Q}_1 = I, \quad \|E\|_2 \le cu\|A\|_2. \tag{8.3.67}$$

If $\bar{\sigma}_1 \ge \cdots \ge \bar{\sigma}_n$ are the singular values of $\bar{R}$ and $\sigma_1 \ge \cdots \ge \sigma_n$ the singular values of A, then it follows that

$$|\bar{\sigma}_i - \sigma_i| \le c_2 u \sigma_1.$$

The result (8.3.67) shows that $\bar{R}$ computed by MGS is comparable in accuracy to the upper triangular matrix from the Householder QR factorization applied to A.

Using the relationship algorithms for least squares problems using the MGS factorization can be developed that give results comparable in accuracy with those obtained by the corresponding Householder QR algorithm. We first derive the MGS algorithm for solving the least squares problems. Clearly, the two problems

$$\min_x \|Ax - b\|_2 \quad \text{and} \quad \min_x \left\| \begin{pmatrix} 0 \\ A \end{pmatrix} x - \begin{pmatrix} 0 \\ b \end{pmatrix} \right\|_2.$$

have the same solution. As above the q_k from MGS corresponds to the Householder transformation P_k, $1 : n$, in (8.3.64). To compute d and h in

$$\begin{pmatrix} d \\ h \end{pmatrix} = P_n \dots P_1 \begin{pmatrix} 0 \\ b \end{pmatrix},$$

define

$$\begin{pmatrix} d_1 \\ h_1 \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}, \qquad \begin{pmatrix} d_{k+1} \\ h_{k+1} \end{pmatrix} = P_k \dots P_1 \begin{pmatrix} 0 \\ b \end{pmatrix} = P_k \begin{pmatrix} d_k \\ h_k \end{pmatrix}.$$

Now using induction we see d_k has all but its first $k-1$ elements zero, and

$$\begin{pmatrix} d_{k+1} \\ h_{k+1} \end{pmatrix} = \begin{pmatrix} d_k \\ h_k \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} (-e_k^T \quad q_k^T) \begin{pmatrix} d_k \\ h_k \end{pmatrix} = \begin{pmatrix} d_k + e_k(q_k^T h_k) \\ h_k - q_k(q_k^T h_k) \end{pmatrix}.$$

Starting with $h_1 := b$, this givs the computation

$$\delta_k := q_k^T h_k; \quad h_{k+1} := h_k - q_k \delta_k, \quad k = 1 : n.$$

so $h = h^{(n+1)}, d = d^{(n+1)} = (\delta_1, \dots, \delta_n)^T$. The computation for d and h is exactly the same as the one that would arise if MGS had been applied to (A, b) instead of just A. This leads to a backward stable MGS algorithm for computing x.

Using the analogy with Householder QR (cf. (8.3.40)) a backward stable approximation of r is obtained setting

$$\begin{pmatrix} 0 \\ r \end{pmatrix} = P \begin{pmatrix} 0 \\ h_{n+1} \end{pmatrix}, \quad P = P_1 \cdots P_{n-1} P_n.$$

where P_k is given by (8.3.64). More generally, to compute an expression $P\begin{pmatrix} z \\ h \end{pmatrix}$, define

$$\begin{pmatrix} w_n \\ y_n \end{pmatrix} = \begin{pmatrix} z \\ h \end{pmatrix}, \quad \begin{pmatrix} w_{k-1} \\ y_{k-1} \end{pmatrix} = P_k \dots P_n \begin{pmatrix} z \\ h \end{pmatrix} = P_k \begin{pmatrix} w_k \\ y_k \end{pmatrix}.$$

This gives the recursion

$$\begin{pmatrix} w_{k-1} \\ y_{k-1} \end{pmatrix} = \begin{pmatrix} w_k \\ y_k \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} (-e_k^T w^{(k)} + q_k^T y_k),$$

which shows in this step only the kth element of w_k is changed from $\zeta_k = e_k^T z$ to $\omega_k = q_k^T y_k$. This gives

$$y_{k-1} := y_k - q_k(\omega_k - \zeta_k), \quad \omega_k := q_k^T y_k \quad k = n : -1 : 1. \tag{8.3.68}$$

so $y = y_0$, $w = (\omega_1, \dots, \omega_n)^T$. In particular, setting $z = 0$ and $h = h_{n+1}$

$$y_{k-1} = y_k - q_k \omega_k, \quad \omega_k = q_k^T y_k \quad k = n : -1 : 1.$$

Note that $w = (\omega_1, \dots, \omega_n)^T$ is ideally zero, but can be significant when $\kappa(A)$ is large. The computation of y can be seen as a reorthogonalize of h_{n+1} against the vectors q_k in backward order. The derivation is summarized in the following algorithm:

Algorithm 8.6. *Linear Least Squares Solution by MGS.*

Assume that MGS has been applied to $A \in R^{m\times n}$, $\operatorname{rank}(A) = n$, to give Q and R. The following function computes the solution x to the linear least squares $\min_x \|Ax - b\|_2$, the residual r, and its Euclidean norm.

```
function [x,r,rho] = mgss(Q,R,b);
[m,n] = size(Q);
d = zeros(n,1);
for k = 1:n            %MGS on b
   d(k) = Q(:,k)'*b;
   b = b - d(k)*Q(:,k);
end
x = R\d; r = b;
for k = n:-1:1         %Reorthogonalize r
   w = Q(:,k)'*r;
   r = r - w*Q(:,k);
end
rho = norm(r);
```

The MGS algorithm is backward stable also for computing the residual $r = b - Ax$. This means that the computed residual $\bar{r}$ satisfies

$$\|A^T \bar{r}\|_2 \le cu\|\bar{r}\|_2 \|A\|_2. \tag{8.3.69}$$

for some constant $c = c(m, n)$. As remarked for the Householder algorithm this is much better than if the residual is computed from its definition $r = b - Ax$. If only x and the residual norm $\|r\|$ is needed, then the reorthogonalization of r can be skipped. In that case only $2n(m + n)$ flops are needed for each right hand side.

Remark 8.3.1.

The algorithm above is the same that is obtained if the MGS algorithm is applied to the right-hand side b, i.e., when the right hand side b is treated as an extra $(n + 1)$st column appended to A, except that the normalization is skipped. This gives the factorization

$$(\, A \quad b\,) = (\, Q_1 \quad r\,) \begin{pmatrix} R & z \\ 0 & 1 \end{pmatrix}. \tag{8.3.70}$$

and hence $r = b - Q_1 z$. By (8.3.55) the product of the computed factors accurately reproduce the matrix $(A,\, b)$, It follows that

$$\|Ax - b\|_2 = \left\| (\, A \quad b\,) \begin{pmatrix} x \\ -1 \end{pmatrix} \right\|_2 = \|Q_1(Rx - z) - r\|_2.$$

If $Q_1^T r = 0$ the minimum of the last expression occurs when $Rx - z = 0$ and it is not necessary to require that Q_1 is accurately orthogonal for this conclusion to hold.

An MGS algorithm for solving the minimum norm problem

$$\min \|y\|_2 \quad \text{subject to} \quad A^T y = c,$$

cen be developed using the technique above. Using the Householder QR factorization the solution is obtained from

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

Suppose that MGS has been applied to A giving R and $Q_1 = (q_1, \ldots, q_n)$. Then $R^T z = c$ is solved for $z = (\zeta_1, \ldots, \zeta_n)^T$. Next, set $y_n = 0$, and use the recursion (8.3.68)

$$y_{k-1} = y_k - q_k(\omega_k - \zeta_k), \quad \omega_k = q_k^T b, \quad k = n : -1 : 1.$$

to compute $y = y_0$.

Algorithm 8.7. *Minimum Norm Solution by MGS.*

Assume that MGS has been applied to $A \in R^{m\times n}$, rank $(A) = n$, to give Q and R. The following function computes the solution y to the linear system $A^T y = c$ which minimizes $\|y\|_2$

```
function [y,rho] = mgsmn(Q,R,c)
```

```
[m,n] = size(Q);
z = R'\c;
y = zeros(m,1);
for k = n:-1:1
   w = Q(:,k)'*y;
   y = y - (w - z(k))*Q(:,k);
end
rho = norm(y);
```

The two MGS algorithm can be generalized to give Based on the Householder QR algorithm given in Theorem 8.3.6 a backward stable MGS algorithm can also be developedfor solving the augmented system.

8.3.7 Error Estimation and Iterative Refinement

Using QR factorization with column pivoting a lower bound for $\kappa(A) = \kappa(R)$ can be obtained from the diagonal elements of R. We have $|r_{11}| \leq \sigma_1 = \|R\|_2$, and since the diagonal elements of R^{-1} equal r_{ii}^{-1}, $i = 1 : n$, it follows that $|r_{nn}^{-1}| \leq \sigma_n^{-1} = \|R^{-1}\|_2$, provided $r_{nn} \neq 0$. Combining these estimates we obtain the *lower bound*

$$\kappa(A) = \sigma_1/\sigma_n \geq |r_{11}/r_{nn}| \tag{8.3.71}$$

Although this may considerably underestimate $\kappa(A)$, it has proved to give a fairly reliable estimate in practice. Extensive numerical testing has shown that (8.3.71) usually underestimates $\kappa(A)$ only by a factor of 2–3, and seldom by more than 10.

When column pivoting has not been performed, the above estimate of $\kappa(A)$ is not reliable. Then a condition estimator similar to that described in Section 7.5.3 can be used. Let u be a given vector, and define v and w from

$$R^T v = u, \qquad Rw = v.$$

We have $w = R^{-1}(R^{-T}u) = (A^TA)^{-1}u$ so this is equivalent to one step of inverse iteration with A^TA, and requires about $0(n^2)$ multiplications. Provided that u is suitably chosen (cf. Section 7.5.3).

$$\sigma_n^{-1} \approx \|w\|_2/\|v\|_2$$

will usually be a good estimate of σ_n^{-1}. We can also take u as a random vector and perform and 2–3 steps of inverse iteration. This condition estimator will usually detect near rank deficiency even in the case when this is not revealed by a small diagonal element in R.

More reliable estimates can be based on the componentwise error bounds (8.2.30)–(8.2.31). In particular, if $E = |A|$, $f = |b|$, we obtain taking norms the estimates

$$\|\delta x\| \lessapprox \omega \left(\| \, |A^\dagger|(|b| + |A||x|) \| + \| \, |(A^TA)^{-1}||A|^T|r| \, \| \right), \tag{8.3.72}$$

$$\|\delta r\| \lessapprox \omega \left(\| \, |I - AA^\dagger|(|A||x| + |b|) \| + \| \, |(A^\dagger)^T||A|^T|r| \, \| \right). \tag{8.3.73}$$

For maximum norm the estimate for $\|\delta x\|$ can be written

$$\|\delta x\|_\infty \le \omega(\| \, |B_1|g_1\|_\infty + \| \, |B_2|g_2\|_\infty), \tag{8.3.74}$$

where

$$B_1 = A^\dagger, \quad g_1 = |b| + |A||x|, \quad B_2 = (A^TA)^{-1}, \quad g_2 = |A^T||r|. \tag{8.3.75}$$

The estimate for $\|\delta r\|_\infty$ has a similar form.

Consider now a general expression of the form $\| \, |B^{-1}|d\|_\infty$, where $d > 0$ is a known nonnegative vector. Writing $D = \operatorname{diag}(d)$ and $e = (1, 1, \ldots, 1)$, we have[34]

$$\| \, |B^{-1}|d\|_\infty = \| \, |B^{-1}|De\|_\infty = \| \, |B^{-1}D|e\|_\infty = \| \, |B^{-1}D|\|_\infty = \|B^{-1}D\|_\infty.$$

There are algorithms that produce reliable order-of-magnitude estimates of $\|C^T\|_1 = \|C\|_\infty$, where $C = B^{-1}D$, using only a few matrix-vector products of the form Cx and C^Ty for some carefully selected vectors x and y. If A has full rank and a QR factorization of A is known, then we have

$$A^\dagger = R^{-1}Q^T, \qquad (A^\dagger)^T = QR^{-T}.$$

Hence, the required products can be computed inexpensively. For details we refer to Higham [328, Chapter 15].

Let $\bar{x}$ be a computed least squares solution and $\bar{r} = b - A\bar{x}$ the computed residual vector. Denote by $x = \bar{x} + e$ the exact solution. Then, since

$$\|b - A\bar{x}\|_2 = \|\bar{r} - Ae\|_2$$

the correction e is itself the solution to a least squares problem. If a QR factorization of A has been computed then it is cheap to solve for the correction vector e. This observation can be used to devise an algorithm for the iterative refinement of a least squares solution similar to that outlined for linear systems in Section . However, it turns out that this is only satisfactory provided the residual vector is r is sufficiently small. Otherwise the solution and residual vector both have to refined simultaneously as we now describe.

Let x be the solution and r be the residual vector of the least squares problem $\min_x \|Ax - b\|_2$, $A \in \mathbf{R}^{m\times n}$. Then x and the scaled residuals $y = r/\alpha$, $\alpha > 0$, satisfy the symmetric indefinite system of equations

$$\begin{pmatrix} \alpha I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix} \tag{8.3.76}$$

with $c = 0$. Let λ be an eigenvalues of the system matrix M_α and $(x, y)^T$ the corresponding eigenvector. Then

$$\alpha x + Ay = \lambda x, \qquad A^Tx = \lambda y,$$

[34] This clever observation is due to Arioli, Demmel, and Duff [12].

and it follows that

$$\alpha\lambda y + A^T A y = \lambda^2 y.$$

If $y \neq 0$ then y is an eigenvector and $(\lambda^2 - \lambda\alpha) = \sigma_i^2$, where σ_i, $i = 1 : n$ are the singular values of $A^T A$. On the other hand $y = 0$, implies that $A^T x = 0$, and $\lambda = \alpha$. Thus, the eigenvalues equal

$$\lambda_i = \frac{\alpha}{2} \pm \left(\frac{\alpha^2}{4} + \sigma_i^2\right)^{1/2}, \quad i = 1 : n,$$

Further, there are $(m - n)$ eigenvalues equal to α.

If we use a solution algorithm for (8.3.76) which is numerically invariant under a scaling α of the $(1, 1)$-block then the relevant condition number is the smallest condition number of $M(\alpha)$. It can be shown that the minimum occurs for $\alpha^2 = \frac{1}{2}\sigma_n$ and

$$\min_\alpha \kappa(M_\alpha) = \frac{1}{2} + \left(\frac{1}{4} + 2\kappa(A)^2\right)^{1/2} \leq 2\kappa(A), \tag{8.3.77}$$

where $\kappa(A) = \sigma_1/\sigma_n$.

We now show how to use the QR factorization to solve the augmented system (8.3.76). Let

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad Q \in \mathbf{R}^{m\times m}, \quad R \in \mathbf{R}^{n\times n}.$$

Using this we can transform the system (8.3.76) into

$$\begin{pmatrix} I & \begin{pmatrix} R \\ 0 \end{pmatrix} \\ \begin{pmatrix} R^T & 0 \end{pmatrix} & 0 \end{pmatrix} \begin{pmatrix} Q^T y \\ x \end{pmatrix} = \begin{pmatrix} Q^T b \\ c \end{pmatrix}.$$

It is easily verified that this gives the solution method

$$z = R^{-T} c, \qquad \begin{pmatrix} d \\ e \end{pmatrix} = Q^T b, \qquad y = Q\begin{pmatrix} z \\ e \end{pmatrix}, \qquad x = R^{-1}(d - z). \tag{8.3.78}$$

Here, if $c = 0$ then $z = 0$ and we retrieve the Householder QR algorithm for linear least squares problems. If $b = 0$, then $d = f = 0$, and (8.3.78) gives the QR algorithm for the minimum norm solution of $A^T y = c$. Clearly the algorithm in (8.3.78) is numerically invariant under a scaling of the

In Section 7.5.6 we considered mixed precision iterative refinement to compute an accurate solution $\bar{x}$ to a linear system $Ax = b$. In this scheme the residual vector $\bar{r} = b - A\bar{x}$ is computed in high precision. Then the system $A\delta = \bar{r}$ for a correction δ to $\bar{x}$ using a lower precision LU factorization of A. If this refinement process is iterated we obtain a solution with an accuracy comparable to that obtained by doing all computations in high precision. Moreover the overhead cost of the refinement is small.

We would like to have a similar process to compute highly accurate solutions to the linear least squares problems $\min_x \|Ax - b\|_2$. In the naive approach, we

would then compute the residual $\bar{r} = b - A\bar{x}$ in high precision and then solve $\min_x \|A\delta x - \bar{r}\|_2$ for a correction δ. If a QR factorization in low precision of A is known we compute

$$\delta x = R^{-1}Q^T\bar{r},$$

and then iterate the process. The accuracy of this refinement scheme is not satisfactory unless the true residual $r = b - Ax$ of the least squares problem equals zero. The solution to an efficient algorithm for iterative refinement is to apply the refinement to the augmented system and refine both the solution x and the residual r in each step. In floating-point arithmetic with base β this process of iterative refinement can be described as follows:

$s := 0; \quad x^{(0)} := 0; r^{(0)} := b;$
repeat
  $f^{(s)} := b - r^{(s)} - Ax^{(s)};$
  $g^{(s)} := c - A^T r^{(s)};$ (in precision $u_2 = \beta^{-t_2}$)
    (solve augmented system in precision $u_1 = \beta^{-t_1}$)
  $x^{(s+1)} := x^{(s)} + \delta x^{(s)};$
  $r^{(s+1)} := r^{(s)} + \delta r^{(s)};$
  $s := s + 1;$
end

Using the Householder QR factorization with the computed factors $\bar{Q}$ and $\bar{R}$ the method in (8.3.78) can be used to solve for the corrections giving

$$z^{(s)} = \bar{R}^{-T}g^{(s)}, \qquad \begin{pmatrix} d^{(s)} \\ e^{(s)} \end{pmatrix} = \bar{Q}^T f^{(s)}, \tag{8.3.79}$$

$$\delta r^{(s)} = \bar{Q}\begin{pmatrix} z^{(s)} \\ e^{(s)} \end{pmatrix}, \qquad \delta x^{(s)} = \bar{R}^{-1}(d^{(s)} - z^{(s)}). \tag{8.3.80}$$

Recall that i $\bar{Q} = P_n \cdots P_2 P_1$, where P_i are Householder reflections, then $\bar{Q}^T = P_1 P_2 \cdots P_n$. The computation of the residuals and corrections, takes $4mn$ flops in high precision. Computing the solution from (8.3.79)–(8.3.80) takes $2n^2$ for operations with $\bar{R}$ and takes $8mn - 4n^2$ for operations with $\bar{Q}$. The total work for a refinement step is an order of magnitude less than the $4n^3/3$ flops required for the QR factorization.

A portable and parallelizable implementation of this algorithm using the extended precision BLAS is available and described in [148].

Review Questions

3.1 Let $w \in \mathbf{R}^n$, $\|w\|_2 = 1$. Show that $I - 2ww^T$ is orthogonal. What is the geometrical significance of the matrix $I - 2ww^T$? Give the eigenvalues and eigenvectors of these matrices.

3.2 Define a Givens transformations $G_{ij}(\phi) \in \mathbf{R}^{n\times n}$. Give a geometrical interpretations for the case $n = 2$.

3.3 Describe the difference between the classical and modified Gram–Schmidt methods for computing the factorization $A = Q_1 R$. What can be said about the orthogonality of the computed matrix Q_1 for these two algorithms?

3.4 Define the QR factorization of a matrix $A \in \mathbf{R}^{m\times n}$, in the case that rank $(A) = n \le m$. What is its relation to the Cholesky factorization of $A^T A$?

Problems

3.1 Show that if H is a reflector, that is H is Hermitian and $H^2 = I$, then $P = (I - H)/2$ is an orthogonal projector. Conversely, if P is an orthogonal projector show that $H = I - 2P$ is a projector.

3.2 Compute using Householder reflectors P_1, P_2, the factorization

$$Q^T A = P_2 P_1 A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad A = (a_1, a_2) = \begin{pmatrix} 1 & 5 \\ 2 & 6 \\ 3 & 7 \\ 4 & 8 \end{pmatrix},$$

to four decimal places

3.3 (a) Specialize the formulas in (8.3.5) for a Householder reflection P to the case $n = 2$. What is the relation between this and the corresponding Givens rotations?

(b) How many flops are needed to apply the reflector P to a matrix of dimension $2 \times n$?

3.4 Solve the least squares problem $\min_x \|Ax - b\|_2$, where

$$A = \begin{pmatrix} \sqrt{2} & 0 \\ 1 & -1 \\ 1 & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.$$

using a QR factorization computed with Givens transformation;

3.5 (a) Derive a *square root free* version of the modified Gram–Schmidt orthogonalization method, by omitting the normalization of the vectors $\widetilde{q}_k$. Show that this version computes a factorization

$$A = \widetilde{Q}_1 \widetilde{R},$$

where $\widetilde{R}$ is **unit** upper triangular.

(b) Suppose the square root free version of modified Gram–Schmidt is used. Modify Algorithm 8.3.6 for computing the least squares solution and residual from this factorization.

Comment: There is no square root free version of Householder QR factorization!

3.6 For any c and s such that $c^2 + s^2 = 1$ we have

$$A = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} 0 & s \\ 0 & c \end{pmatrix} = QR.$$

Here rank $(A) = 1 < 2 = n$. Show that the columns of Q do not provide any information about an orthogonal basis for $\mathcal{R}(A)$.

3.7 (a) If the matrix Q in the QR factorization is explicitly required in the Householder algorithm it can be computed by setting $Q^{(n)} = I_m$, and computing $Q = Q^{(0)}$ by the backward recursion

$$Q^{(k-1)} = P_k Q^{(k)}, \quad k = n : -1 : 1.$$

Show that if advantage is taken of the property that $P_k = \text{diag}\,(I_{k-1}, \widetilde{P}_k)$ this accumulation requires $4(mn(m-n) + n^3/3)$ flops. What is the corresponding operation count if forward recursion is used?

(b) Show how we can compute

$$Q_1 = Q \begin{pmatrix} I_n \\ 0 \end{pmatrix}, \qquad Q_2 = Q \begin{pmatrix} 0 \\ I_{m-n} \end{pmatrix}$$

separately in $2(mn^2 - n^3/3)$ and $2(2m^2n - 3mn^2 + n^3)$ multiplications, respectively.

3.8 Let $Q = Q_1 = (q_1, q_2, \ldots, q_n) \in \mathbf{R}^{n\times n}$ be a real orthogonal matrix.

(a) Determine a reflector $P_1 = I - 2v_1 v_1^T$ such that $P_1 q_1 = e_1 = (1, 0, \ldots, 0)^T$, and show that $P_1 Q_1 = Q_2$ has the form

$$Q_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & \widetilde{Q}_2 & \\ 0 & & & \end{pmatrix},$$

where $\widetilde{Q}_2 = (\widetilde{q}_1, \widetilde{q}_2, \ldots, \widetilde{q}_n) \in \mathbf{R}^{(n-1)\times(n-1)}$ is a real orthogonal matrix.

(b) Show, using the result in (a), that Q can be transformed to diagonal form with a sequence of orthogonal transformations

$$P_{n-1} \cdots P_2 P_1 Q = \text{diag}\,(1, \ldots, 1, \pm 1).$$

3.9 In several applications one needs to compute the QR factorization of a matrix

$$A = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$$

where R_1 and R_2 are square upper triangular matrices. Show how to compute the QR factorization of A in $2n^3/3$ flops using suitably chosen Householder transformations which don not introduce any nonzero elements outside the triangular structures.

Hint: In step k a full submatrix of size $(k+1) \times (n-k+1)$ consisting of

selected rows is operated on. Below is a picture of the reduction when $n = 4$ and the two first columns have been processed

$$\begin{pmatrix} * & * & * & * \\ & * & * & * \\ & & \times & \times \\ & & & \times \\ \otimes & \otimes & * & * \\ & \otimes & * & * \\ & & \times & \times \\ & & & \times \end{pmatrix}$$

Here $*$ denotes a modified element and $\otimes$ an element that has been zeroed out. In the next step the submatrix consisting of rows 3,5,6, and 7 will be operated on.

3.10 Show how to compute the QR factorization of a lower triangular matrix $L \in Rnn$ in n^3 flops using Givens rotations.

Hint: In the first step elements in the first column are zeroed out by rotating rows $(1,2),(1,3),\ldots,(1,n)$ in this order. Then the first row and column are finished. The reduction continues in a similar way on the remaining lower triangular matrix of order $(n-1)$.

3.11 Show how to compute the QR factorization of the product $A = A_p \cdots A_2 A_1$ without explicitly forming the product matrix A.

Hint: For $p = 2$ first determine Q_1 such that $Q_1^T A_1 = R_1$, and form $A_2 Q_1$. Then, if Q_2 is such that $Q_2^T A_2 Q_1 = R_2$ it follows that $Q_2^T A_2 A_1 = R_2 R_1$.

3.12 Show that if the column operations in MGS are carried out also on a second block row in the augmented matrix, the result can be written

$$\begin{pmatrix} A & b \\ I & 0 \end{pmatrix} \quad \longrightarrow \quad \begin{pmatrix} Q_1 & r \\ R^{-1} & -x \end{pmatrix}.$$

Hence, the MGS algorithm can be made to provide in a single sweep operation the solution x, the residual r, as well as the matrix R^{-1} which is required for computing the covariance matrix $\sigma R^{-1} R^{-T}$.

8.4 Rank Deficient Problems

8.4.1 Rank Revealing QR Factorizations

Let $A \in \mathbf{R}^{m\times n}$ be a matrix with $\operatorname{rank}(A) = r \le \min(m,n)$, and Π a permutation matrix such that the first r columns in $A\Pi$ are linearly independent. Then the QR factorization of $A\Pi$ will have the form

$$A\Pi = (\, Q_1 \quad Q_2 \,) \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \tag{8.4.1}$$

where $R_{11} \in \mathbf{R}^{r\times r}$ is upper triangular with positive diagonal elements and $R_{11} \in \mathbf{R}^{r\times(n-r)}$. Note that it is not required that $m \ge n$. The matrices Q_1 and Q_2 form

orthogonal bases for the two fundamental subspaces$\mathcal{R}(A)$ and $\mathcal{N}(A^T)$, respectively. The factorization (8.4.1) is not unique since there are many ways to select r linearly independent columns of A.

To simplify notations we assume in the following that $\Pi = I$. (This is no restriction since the column permutation of A can always be assumed to have been carried out in advance.) Using (8.4.1) the least squares problem $\min_x \|Ax - b\|_2$ is reduced to

$$\min_x \left\| \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \right\|_2, \tag{8.4.2}$$

where $c = Q^T b$. Since R_{11} is nonsingular the first r equations can be satisfied exactly for any x_2. Hence, the general least squares solutions becomes

$$x_1 = R_{11}^{-1}(c_1 - R_{12}x_2) \tag{8.4.3}$$

where x_2 can be chosen arbitrarily. By setting $x_2 = 0$ a particular solution $x_1 = R_{11}^{-1}c_1$ is obtained for which at most $r = \text{rank}(A)$ components are nonzero. This solution is appropriate in applications where it is required to fit the vector b using *as few columns of* A as possible. It is not unique and depends on the initial column permutation. Any such solution x such that Ax only involves at most r columns of A, is called a **basic solution**.

Letting x_2 be an arbitrary vector we have

$$x_1 = d - Cx_2, \quad d = R_{11}^{-1}c_1, \quad C = R_{11}^{-1}R_{12}, \tag{8.4.4}$$

where the $C \in \mathbf{R}^{r\times(n-r)}$ can be computed by solving the triangular systems $R_{11}C = R_{12}$.In particular, the solution of minimum norm, i.e., the pseudo-inverse solution is obtained by solving

$$\min_x \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_2 = \min_{x_2} \left\| \begin{pmatrix} d \\ 0 \end{pmatrix} - \begin{pmatrix} C \\ -I_{n-r} \end{pmatrix} x_2 \right\|_2. \tag{8.4.5}$$

which is a the linear least squares problem for x_2. Note that this problem always has full rank and hence has a unique solution. To compute x_2 we could form and solve the normal equations

$$(I + CC^T)x_2 = C^T d.$$

It is usually preferable to compute the Householder QR factorization

$$Q_C^T \begin{pmatrix} C \\ I_{n-r} \end{pmatrix} = \begin{pmatrix} R_C \\ 0 \end{pmatrix}, \qquad Q_C^T \begin{pmatrix} d \\ 0 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix},$$

taking into account the special structure of the matrix. Noticing that the pseudo-inverse solution $x = A^\dagger b$ equals *the residual* of the problem (8.4.5), we get

$$x = A^\dagger b = Q_C \begin{pmatrix} 0 \\ d_2 \end{pmatrix}.$$

We further note that for any $z \in \mathbf{R}^{n-r}$ it holds that

$$A \begin{pmatrix} C \\ -I_{n-r} \end{pmatrix} z = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} R_{11}^{-1} R_{12} \\ -I_{n-r} \end{pmatrix} z = 0,$$

It follows that a basis for the null space of A is given by the columns of the matrix W, where

$$\mathcal{N}(A) = \mathcal{R}(W), \quad W = \begin{pmatrix} C \\ -I_{n-r} \end{pmatrix}. \tag{8.4.6}$$

8.4.2 Complete QR Factorizations

In signal processing problems it is often the case that one wants to determine the rank of A as well as the range (signal subspace) and null space of A. Since the data analyzed arrives in real time it is necessary to update an appropriate matrix decompositions at each time step. For such applications the SVD has the disadvantage that it cannot be updated in less than $\mathcal{O}(n^3)$ operations, when rows and/or columns are added or deleted to A. Although the RRQR factorization can be updated, it is less suitable in applications where a basis for the approximate null space of A is needed. The matrix W in (8.4.6) may be ill-conditioned and cannot easily be updated.

This motivates the introduction of the **complete QR factorization** of a matrix $A \in \mathbf{R}^{m\times n}$. This is a factorization of the form

$$A = U \begin{pmatrix} T & 0 \\ 0 & 0 \end{pmatrix} V^T = U_1 T V_1^T, \tag{8.4.7}$$

where $T \in \mathbf{R}^{r\times r}$ is an upper or lower triangular matrix with positive diagonal elements and

$$U = (\, U_1 \quad U_2 \,) \in \mathbf{R}^{m\times m}, \qquad V = (\, V_1 \quad V_2 \,) \in \mathbf{R}^{n\times n},$$

are orthogonal matrices partitioned so that $U_1 \in \mathbf{R}^{m\times r}$ and $V_1 =\in \mathbf{R}^{n\times r}$. An advantage of the complete QR factorization of A is that, like the SVD, it gives orthogonal bases for all four fundamental subspaces of A. In particular, V_2 gives an orthogonal basis for the null space $\mathcal{N}(A)$. This is often useful in signal processing applications, where one wants to determine the part of the signal that corresponds to noise.

From the orthogonal invariance of the 2-norm it follows that the minimum norm solution of the least squares problem $\min_x \|Ax - b\|_2$ is $x = V_1 T^{-1} U_1^T b$ and the pseudo-inverse of A equals

$$A^\dagger = V \begin{pmatrix} T^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^T = V_1 T^{-1} U_1^T. \tag{8.4.8}$$

compute the factorization (8.4.7) we can start from a rank revealing QR factorization

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix}.$$

and then construct a sequence of Householder reflections P_k, $k = r : (-1) : 1$, such that

$$(\, R_{11} \quad R_{12}\,)\, P_r \cdots P_1 = (\, T \quad 0\,)$$

Here P_k zeros elements in row k and only affects columns $k, r+1 : n$. Then (8.4.7) holds with T upper triangular and

$$V = \Pi P_1 \cdots P_r.$$

The diagram below shows the reduction when $r = 4$ and $n = 6$ and the two last rows of R_{12} have been annihilated.

$$\left(\begin{array}{cccc|cc} \times & \times & * & * & * & * \\ & \times & * & * & * & * \\ & & * & * & \otimes & \otimes \\ & & & * & \otimes & \otimes \end{array}\right)$$

Here $*$ denotes a modified element and $\otimes$ an element that has been zeroed out. In the next step the submatrix consisting of columns 2,5, and 6 will be transformed by a Householder reflection P_2 to zero the elements in position (2,5) and (2,6). In general, in step k a full matrix of size $k \times (n-r+1)$ is transformed by a Householder reflection. The transformations require a total of $2r^2(n-r+1)$ flops.

Even for a rank deficient matrix A of rank $r < n$, a rank revealing QR factorization will yield a factorization

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

where R_{22} is nonzero but $\|R_{22}\| \le \epsilon$, where ϵ is small. In a similar way the complete QR factorization can be generalized to the case when A is only numerically rank deficient. We will consider URV factorizations, which have of the form

$$A = URV^T = (\, U_1 \quad U_2\,) \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}, \tag{8.4.9}$$

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

$$\sigma_r(R_{11}) \ge \frac{1}{c}\sigma_r, \qquad \left(\|R_{12}\|_F^2 + \|R_{22}\|_F^2\right)^{1/2} \le c\sigma_{r+1}. \tag{8.4.10}$$

Note that here both submatrices R_{12} and R_{22} are required to have small elements. From (8.4.9) we have

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

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

Algorithms for computing an URV decomposition start with an initial QR factorization with column pivoting. Then a rank revealing stage follows, 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 a 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, Efficient algorithms can be given for updating an URV decomposition when a new row is appended to A.

Like the RRQR factorizations the URV decomposition yield approximations to the singular values. In [428] the following bounds are derived

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

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.

An alternative decomposition that is more satisfactory for applications where an accurate approximate null space is needed, is the rank-revealing **ULV decomposition**

$$A = U \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} V^T. \tag{8.4.11}$$

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. For recent work see also citebaes:05,baer:08

We finally mention that rank-revealing QR factorizations 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 [103]. Matlab templates for rank-revealing UTV decompositions are described in [203].

8.4.3 Column Subset Selection Problem

In the column **subset selection** problem we are given a matrix $A \in \mathbf{R}^{m\times n}$ and want to determine a subset A_1 of $k < n$ columns such that

$$\|A - (A_1 A_1^{\dagger})A\|$$

is minimized over all $\binom{n}{k}$ possible choices. In other words, we want to find a permutation P such that the smallest singular value of the k first columns of AP are maximized.

The column subset selection problem is closely related to rank-revealing QR factorization. The following theorem, which we state without proof, shows that a column permutation Π always exists such that the numerical rank of A is revealed by the QR factorization of $A\Pi$.

Theorem 8.4.1 (*H. P. Hong and C. T. Pan* [337]).

Let $A \in \mathbf{R}^{m\times n}$, $(m \geq n)$, and r be a given integer $0 < r < n$. Then there exists a permutation matrix Π_r, such that the QR factorization has the form

$$Q^T A\Pi_r = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \tag{8.4.12}$$

with $R_{11} \in \mathbf{R}^{r\times r}$ upper triangular, and

$$\sigma_{\min}(R_{11}) \geq \frac{1}{c}\sigma_r(A), \qquad \sigma_{\max}(R_{22}) \leq c\sigma_{r+1}(A), \tag{8.4.13}$$

where $c = \sqrt{r(n-r) + \min(r, n-r)})$.

If A has a well defined numerical rank $r < n$, i.e.,

$$\sigma_1 \geq \ldots \geq \sigma_r \gg \delta \geq \sigma_{r+1} \geq \ldots \geq \sigma_n,$$

then the above theorem says that if the ratio σ_k/σ_{k+1} is sufficiently large then there is a permutation of the columns of A such that the rank of A is revealed by the QR factorization. Unfortunately, to find such a permutation is an NP hard problem.

There are heuristic algorithms which almost always succeeds in practice. The following SVD-based algorithm for finding a good approximation to the solution of the subset selection problem is due to Golub, Klema, and Stewart [266]; see also Golub and Van Loan [277, Sec. 12.2].

Let k be the numerical rank of A determined from the SVD $A = U\Sigma V^T$. If P is any permutation matrix, then $U^T(AP)(P^TV) = \Sigma$, Thus, permutation of the columns of A correspond to a permutation of the rows of the orthogonal matrix V of right singular vectors. The theoretical basis for this column selection strategy is the following result.

Theorem 8.4.2 (Theorem 6.1, [266]).

Let $A = U\Sigma V^T \in \mathbf{R}^{m\times n}$ be the SVD of A. Partition A and V conformally as $A = (\,A_1 \quad A_2\,)$ and $V = (\,V_1 \quad V_2\,)$, where

$$(\,V_1 \quad V_2\,) = \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} \end{array}, \tag{8.4.14}$$

and $\hat{V}_{11}$ is nonsingular. Then

$$\frac{\sigma_k(A)}{\|V_{11}^{-1}\|_2} \le \sigma_k(A_1) \le \sigma_k(A). \tag{8.4.15}$$

Proof. The upper bound in (8.4.15) follows directly from the interlacing property of the singular values of A_1 and A; see Theorem 8.1.17. If we set $A\,(\,V_1 \quad V_2\,) = (\,S_1 \quad S_2\,)$, then $S_1^T S_2 = 0$. Now since $A = SV^T$, we have

$$A_1 = S_1 V_{11}^T + S_2 V_{21}^T.$$

If we define $\inf(A) = \inf_{\|x\|_2=1} \|Ax\|_2$, then

$$\inf(A_1) \ge \inf(S_1 V_{11}) \ge \sigma_k(A)\inf(V_{11}).$$

Since $\inf(V_{11}) = \|V_{11}^{-1}\|_2$ the lower bound follows. □

The above result suggests that we choose a permutation P such that the submatrix $\hat{V}_{11}$ consisting of the first k columns os $\hat{V} = VP$ is as well-conditioned as possible. This can be achieved by using QR factorization with column pivoting to compute

$$Q^T\,(\,V_{11}^T \quad V_{21}^T\,)\,P = (\,R_{11} \quad R_{12}\,).$$

The least squares problem $\min_z \|A_1 z - b\|_2$ can then be solved by QR factorization.

From the interlacing properties of singular values (Theorem 8.1.17) it follows by induction that for any factorization of the form (8.4.12), we have the inequalities

$$\sigma_{\min}(R_{11}) \le \sigma_r(A), \qquad \sigma_{\max}(R_{22}) \ge \sigma_{r+1}(A). \tag{8.4.16}$$

Hence, to achieve (8.4.13) we want to choose the permutation Π to maximize $\sigma_{\min}(R_{11})$ and simultaneously minimize $\sigma_{\max}(R_{22})$. These two problems are in a certain sense dual; cf. Problem 8.4.3.

From Theorem 8.1.25 it follows that in (8.4.14) $\|V_{11}\|_2 = \|V_{22}\|_2$. When $k > n/2$ it is more economical to use the QR factorization with column pivoting applied to the smaller submatrix $V_2^T = (V_{12}^T,\ V_{22}^T)$. This will then indicate *what columns in A to drop.*

In case $k = n-1$, the column permutation P corresponds to the largest component in the right singular vector $V_2 = v_n$ belonging to the smallest singular value σ_n. The index indicates what column to drop. In a related strategy due to T. F. Chan [99] repeated use is made of this, by dropping one column at a time. The right singular vectors needed are determined by inverse iteration (see Section 9.4.3).

A comparison between the RRQR and the above SVD-based algorithms is given by Chan and Hansen [102, 1992]. Although in general the methods will not necessarily compute equivalent solutions, the subspaces spanned by the two sets of selected columns are still almost identical whenever the ratio σ_{k+1}/σ_k is small. The best bounds are those given by Gu and Eisenstat [292]. Chandrasekaran and Ipsen [105].

8.4.4 Modified Least Squares Problems

Suppose that we have solved the least squares problem $\min_x \|Ax - b\|_2$, where $A \in \mathbf{R}^{m\times n}$, $m > n$. It may often be required to later solve a related least squares problem where a simple modification of the data $(A,\ b)$ has been performed. For example, one may want to add new observations or discard observations with unacceptably large residuals. In various time-series problems data are arriving sequentially and a least squares solution has to be updated at each time step. Such modifications are usually referred to as **updating** when (new) data is added and **downdating** when (old) data is removed. In time-series problems a **sliding window method** is often used. At each time step a new row of data is added and the oldest row of data deleted. Applications in signal processing often require real-time solutions so efficiency is critical.

Other applications arise in optimization and statistics. Indeed, the first systematic use of such algorithms seems to have been in optimization. In linear regression efficient and stable procedure for adding and/or deleting observations is needed. In stepwise regression one wants to examine different models by adding and/or deleting variables in each step. Another important application occurs in active set methods for solving least squares problems with inequality constraints.

We will consider the following type of modifications:

1. A general rank-one change to to $(\,A \quad b\,)$.
2. Adding or deleting a row of $(\,A \quad b\,)$.
3. Adding of deleting a column of A.

If a row is added to A, then by the interlacing property (Theorem 8.1.17) the smallest singular value of the modified matrix will not decrease. On the other hand, when a row is deleted the rank can decrease. Similarly, when a column is deleted the smallest singular value will not decrease. When a column is added the modified matrix may become singular. This indicates that deleting a column and adding a row are "easy" operations, whereas adding a column and deleting a row can be more delicate operations. In general, we can not expect modification methods to be stable when the unmodified problem is much worse conditioned than the modified problem,

Another aspect that has to be considered is what factorization is available of the original matrix A. The simplest case is when the full QR factorization with $Q \in \mathbf{R}^{m\times m}$ explicitly known. In cases when $n \ll m$ it may be more desirable to update the thin QR factorization. Finally, sometimes only the factor R may be known. This is the case, e.g., when the initial problem has been solved by the normal equations.

Recursive Least Squares.

We first describe a simple algorithm for adding and deleting rows based on the Cholesky factorization and normal equations. The solution to the least squares

problem satisfies the normal equations $A^TAx = A^Tb$. If the equation $w^Tx = \beta$ is added, then the updated solution $\widetilde{x}$ satisfies the modified normal equations

$$(A^TA + ww^T)\widetilde{x} = A^Tb + \beta w, \tag{8.4.17}$$

where we assume that rank $(A) = n$. Let R is the Cholesky factor of A^TA. We would like to avoid computing the Cholesky factorization of the modified problem from scratch. In the **covariance matrix method**. the covariance matrix

$$C = (A^TA)^{-1} = R^{-1}R^{-T},$$

is updated. Since $\widetilde{C}^{-1} = C^{-1} + ww^T$, we have by the Sherman–Morrison formula (7.1.26)

$$\widetilde{C} = C - \frac{1}{1 + w^Tu}uu^T, \quad u = Cw. \tag{8.4.18}$$

Adding the term $ww^Tx = (w^Tx)w$ to both sides of the unmodified normal equations and subtracting from (8.4.17) gives

$$(A^TA + ww^T)(\widetilde{x} - x) = (\beta - w^Tx)w.$$

Solving for the updated solution gives the following basic formula:

$$\widetilde{x} = x + (\beta - w^Tx)\widetilde{u}, \quad \widetilde{u} = \widetilde{C}w. \tag{8.4.19}$$

Equations (8.4.18)–(8.4.19) define a **recursive least squares** algorithm associated with the Kalman filter. The vector $\widetilde{u} = \widetilde{C}w$ which weights the predicted residual $\beta - w^Tx$ of the new observation is called the **Kalman gain vector** .

The equations (8.4.18)–(8.4.19) can, with slight modifications, be used also for *deleting* an observation $w^Tx = \beta$. We have

$$\widetilde{C} = C + \frac{1}{1 - w^Tu}uu^T, \qquad \widetilde{x} = x - (\beta - w^Tx)\widetilde{u}, \tag{8.4.20}$$

provided that $1 - w^Tu \neq 0$.

The simplicity and recursive nature of this updating algorithm has made it popular for many applications. The main disadvantage of the algorithm is its serious sensitivity to roundoff errors. The updating algorithms based on orthogonal transformations developed in Section 8.4.4 are therefore generally to be preferred.

The method can also be used together with updating schemes for the Cholesky factor factor R or its inverse R^{-1}, where $A^TA = R^TR$. The Kalman gain vector can then be computed from

$$z = \widetilde{R}^{-T}w, \qquad p = \widetilde{R}^{-1}z.$$

Such methods are often referred to as "square root methods" in the signal processing literature. Schemes for updating R^{-1} are described in [61, Sec. 3.3]. Since no back-substitutions is needed in these schemes, they are easier to parallelize.

Updating the QR Factorization

Algorithms for updating the thin QR factorization when $Q \in \mathbf{R}^{m\times n}$ and $R \in Rnn$ are explicitly known were first described by Daniel, Gragg, Kaufman, and Stewart [134]. These algorithm use Gram–Schmidt with reorthogonalization combined with Givens rotations. Fortran subroutines that were modifications of the Algol procedures given there appeared in Reichel and Gragg [497].

We first consider updating the QR factorization of a matrix $A \in \mathbf{R}^{m\times n}$ of full column rank. A first observation is not feasible to update a QR factorization where Q is stored implicitly as a product of Householder transformations. Therefore, we assume that

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{8.4.21}$$

where the orthogonal factor $Q \in \mathbf{R}^{m\times m}$ is known explicitly. These updating algorithms require $O(m^2)$ multiplications, and are (almost) normwise backward stable. We want to compute the factorization when A is subject to a rank one change

$$\widetilde{A} = A + uv^T = \widetilde{Q}\begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}, \tag{8.4.22}$$

$u \in \mathbf{R}^m$ and $v \in \mathbf{R}^n$. We assume that $\operatorname{rank}(A) = \operatorname{rank}(\widetilde{A}) = n$, so that R and $\widetilde{R}$ are uniquely determined. The algorithm proceeds as follows:

1. Compute the vector $w = Q^T u \in \mathbf{R}^m$, so that

$$A + uv^T = Q\left[\begin{pmatrix} R \\ 0 \end{pmatrix} + wv^T\right]. \tag{8.4.23}$$

2. Determine a sequence of Givens rotations $J_k = G_{k,k+1}(\theta_k)$, $k = m-1 : (-1) : 1$ such that

$$J_1^T \cdots J_{m-1}^T w = \alpha e_1, \qquad \alpha = \pm\|w\|_2.$$

These transformations zero the last $m-1$ components of w from bottom up. (For details on how to compute J_k see Section 8.3.1.) Apply these transformations to R to obtain

$$\widetilde{H} = J_1^T \cdots J_{m-1}^T\left[\begin{pmatrix} R \\ 0 \end{pmatrix} + wv^T\right] = H + \alpha e_1 v^T. \tag{8.4.24}$$

(Note that $J_{n+1}, \ldots, J_{m-1}$ have no effect on R.) Because of the structure of the Givens rotations the matrix H will be an upper Hessenberg matrix, i.e., H is triangular except for extra nonzero elements $h_{k+1,k}, k = 1, 2, \ldots, n$ (e.g., $m = 6$, $n = 4$),

$$H = \begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Since only the first row of H is modified by the term $\alpha e_1 v^T$, $\widetilde{H}$ is also upper Hessenberg.

3. Determine Givens rotations $\widetilde{J}_k = G_{k,k+1}(\phi_k)$, $k = 1 : n$, to zero the element in position $(k+1, k)$, so that

$$\widetilde{J}_n^T \cdots \widetilde{J}_1^T \widetilde{H} = \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix} \tag{8.4.25}$$

is upper triangular.

4. Accumulate the transformations into Q to get

$$\widetilde{Q} = QU, \qquad U = J_{m-1} \cdots J_1 \widetilde{J}_1 \cdots \widetilde{J}_n.$$

This gives the desired factorization (8.4.22).

The work needed for this update is as follows: computing $w = Q^T u$ takes m^2 flops. Computing H and $\widetilde{R}$ takes $4n^2$ flops and accumulating the transformations J_k and $\widetilde{J}_k$ into Q takes $4(m^2 + mn)$ flops for a total of $5m^2 + 4mn + 4n^2$ flops. Hence, the work has been decreased from $O(mn^2)$ to $O(m^2)$. However, if n is small updating may still be more expensive than computing the factorization from scratch.

There is a simple relationship between the problem of updating matrix factorizations and that of updating the least squares solutions. Recall that if A has full column rank and the R-factor of the matrix (A, b) is

$$\begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix}, \tag{8.4.26}$$

then the solution to the least squares problem $\min_x \|Ax - b\|_2$ is given by

$$Rx = z, \qquad \|Ax - b\|_2 = \rho. \tag{8.4.27}$$

The upper triangular matrix (8.4.26) can be computed either from the QR factorization of (A, b) or as the Cholesky factor of $(A, b)^T(A, b)$. Hence, to get an updating algorithm for least squares solutions, we only have to apply an updating algorithm for matrix factorization to the extended matrix (A, b).

To update a least squares solution under a rank one change one applies the updating procedure above to compute the QR factorization of

$$(\, A + uv^T \quad b \,) = (\, A \quad b \,) + u\,(\, v^T \quad 0 \,).$$

where the right hand side has been appended. [164, 1979, Chap. 10].

Algorithms for modifying the full QR factorization of (A, b) when a column is deleted or added are special cases of the rank one change; see Björck [61, Section . 3.2]. Adding a row is simple, since the QR factorization is easily organized to treat a row at a time. A more subtle problem is to modify the QR factorization when a row is *deleted*, which is called the **downdating** problem. This corresponds to the problem of deleting an observation in a least squares problem. In the sliding window method one row is added and simultaneously one row is deleted. Another

instance when a row has to be removed is when it has somehow been identified as faulty.

There is no loss of generality in assuming that it is the *first row* of A that is to be deleted. We wish to obtain the QR factorization of the matrix $\widetilde{A} \in \mathbf{R}^{(m-1)\times n}$ when

$$A = \begin{pmatrix} a_1^T \\ \widetilde{A} \end{pmatrix} = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{8.4.28}$$

is known. We now show that this is equivalent to finding the QR factorization of (e_1, A), where a dummy column $e_1 = (1, 0, \ldots, 0)^T$ has been added. We have

$$Q^T(e_1, A) = \begin{pmatrix} q_1 & R \\ q_2 & 0 \end{pmatrix},$$

where $q^T = (q_1^T, q_2^T) \in \mathbf{R}^m$ is the *first row* of Q. We now determine Givens rotations $J_k = G_{k,k+1}$, $k = m-1:-1:1$, so that

$$J_1^T \cdots J_{m-1}^T q = \alpha e_1, \quad \alpha = \pm 1. \tag{8.4.29}$$

Then we have

$$J_1^T \cdots J_{m-1}^T \begin{pmatrix} q_1 & R \\ q_2 & 0 \end{pmatrix} = \begin{pmatrix} \alpha & v^T \\ 0 & \widetilde{R} \\ 0 & 0 \end{pmatrix}, \tag{8.4.30}$$

where the matrix $\widetilde{R}$ is upper triangular. Note that the transformations $J_{n+1}, \ldots, J_{m-1}$ will not affect R. Further, if we compute

$$\bar{Q} = Q J_{m-1} \cdots J_1,$$

it follows from (8.4.29) that the first row of $\bar{Q}$ equals αe_1^T. Since $\bar{Q}$ is orthogonal it must have the form

$$\bar{Q} = \begin{pmatrix} \alpha & 0 \\ 0 & \widetilde{Q} \end{pmatrix},$$

with $|\alpha| = 1$ and $\widetilde{Q} \in \mathbf{R}^{(m-1)\times(m-1)}$ orthogonal. From (8.4.29),

$$\begin{pmatrix} a_1^T \\ \widetilde{A} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \widetilde{Q} \end{pmatrix} \begin{pmatrix} v^T \\ \widetilde{R} \\ 0 \end{pmatrix},$$

and hence the desired factorization is

$$\widetilde{A} = \widetilde{Q} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}.$$

This algorithm for downdating is a special case of the rank one change algorithm, and is obtained by taking $u = -e_1$, $v^T = a_1^T$ in (8.4.22).

8.4.5 Golub–Kahan Bidiagonalization

So far we have considered methods for solving least squares problems based on reducing the matrix $A \in \mathbf{R}^{m\times n}$ to upper triangular (or trapezoidal) form using unitary operations. It is possible to carry this reduction further and obtain a lower bidiagonal matrix ($m \geq n$)

$$U^TAV = \begin{pmatrix} B \\ 0 \end{pmatrix}, \quad B = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_n \\ & & & \beta_{n+1} \end{pmatrix} \in \mathbf{R}^{(n+1)\times n}, \tag{8.4.31}$$

where U and V are square orthogonal matrices. The bidiagonal form is the closest that can be achieved by a finite process to the diagonal form in SVD. The bidiagonal decomposition (8.4.31) therefore is usually the first step in computing the SVD of A; see Section 9.7. It is also powerful tool for solving various least squares problems.

Note that from (8.4.31) it follows that $A^T = VB^TU^T$, where B^T is upper bidiagonal. Thus, if we apply the same process to A^T we obtain an *upper* bidiagonal form. A complex matrix $A \in \mathbf{C}^{m\times n}$ can be reduced by a similar process to *real* bidiagonal form using unitary transformations U and V. We consider here only the real case, since the generalization to the complex case is straightforward.

The simplest method for constructing the bidiagonal decomposition is the **Golub–Kahan algorithm** in which the reduction is achieved by applying a sequence of Householder reflections alternately from left and right. We set $A = A^{(1)}$ and in the first double step compute

$$A^{(2)} = Q_1(AP_1) = \begin{pmatrix} \alpha_1 & 0 & \cdots & 0 \\ \beta_2 & \widetilde{a}_{22} & \cdots & \widetilde{a}_{2n} \\ 0 & \widetilde{a}_{32} & \cdots & \widetilde{a}_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \widetilde{a}_{m2} & \cdots & \widetilde{a}_{mn} \end{pmatrix}.$$

Here the Householder reflection P_1 is chosen to zero the last $n-1$ elements in the first row of A. Next Q_1 is chosen to zero the last $m-2$ elements in the the first column of AP_1. This key thing to observe is that Q_1 will leave the first row in AP_1 unchanged and thus not affect the zeros introduced by P_1. All later steps are similar. In the kth step, $k = 1 : \min(m, n)$, we compute

$$A^{(k+1)} = Q_k(A^{(k)}P_k),$$

where Q_k and P_k are Householder reflections. Here P_k is chosen to zero the last $n-k$ elements in the kth row of $A^{(k)}$. Then Q_k is chosen to zero the last $m-(k+1)$ elements in the kth column of $A^{(k)}P_k$.

The process is continued until either the rows or columns are exhausted- When $m > n$ the process ends with the factorization

$$U = Q_1Q_2\cdots Q_n, \qquad V = P_1P_2\cdots P_{n-1}. \tag{8.4.32}$$

Note that since Q_k, only works on rows $k+1:m$, and P_k, only works on columns $k:m$. It follows that

$$u_1 = e_1. \quad u_k = Ue_k = Q_1 \cdots Q_k e_k, \quad k = 2:n, \tag{8.4.33}$$

$$v_k = Ve_k = P_1 \cdots P_k e_k, \quad k = 1:n-1, \quad v_n = e_n. \tag{8.4.34}$$

If $m \le n$ then we obtain

$$U^T A V = (\, B \quad 0\,), \quad B = \begin{pmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \ddots & & \\ & & \ddots & \alpha_{m-1} & \\ & & & \beta_m & \alpha_m \end{pmatrix} \in \mathbf{R}^{m\times m}.$$

$$U = Q_1 Q_2 \cdots Q_{m-2}, \qquad V = P_1 P_2 \cdots P_{m-1}.$$

The above process can always be carried through although some elements in B may vanish. Note that the singular values of B equal those of A; in particular, $\operatorname{rank}(A) = \operatorname{rank}(B)$. Using complex Householder transformations (see Section sec8.3.2) a complex matrix A can be reduced to *real* bidiagonal form by the algorithm above. In the nondegenerate case when all elements in B are nonzero, the bidiagonal decomposition is uniquely determined first column $u_1 := Ue_1$, which can be chosen arbitrarily.

The Householder reduction to bidiagonal form is backward stable in the following sense. The computed $\bar{B}$ can be shown to be the exact result of an orthogonal transformation from left and right of a matrix $A + E$, where

$$\|E\|_F \le cn^2 u \|A\|_F, \tag{8.4.35}$$

and c is a constant of order unity. Moreover, if we use the information stored to generate the products $U = Q_1 \cdots Q_n$ and $V = P_1 \cdots P_{n-2}$, then the computed matrices are close to the exact matrices U and V which reduce $A + E$. This will guarantee that the singular values and transformed singular vectors of $\bar{B}$ are accurate approximations to those of a matrix close to A.

The bidiagonal reduction algorithm as described above requires approximately

$$4(mn^2 - n^3/3) \text{ flops}$$

when $m \ge n$, which is twice the work for a Householder QR factorization. The Householder vectors associated with U can be stored in the lower triangular part of A and those associated with V in the upper triangular part of A. Normally U and V are not explicitly required. They can be accumulated at a cost of $4(m^2 n - mn^2 + \frac{1}{3}n^3)$ and $\frac{4}{3}n^3$ flops respectively.

When $m \gg n$ it is more efficient to use a two-step procedure as originally suggested by Lawson and Hanson [399] and later analyzed by T. Chan. In the first step the QR factorization of A is computed (possibly using column pivoting)

$$AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R \in \mathbf{R}^{n\times n},$$

which requires $4mn^2 - \frac{2}{3}n^3$ flops. In the second step the upper triangular matrix R is transformed to bidiagonal form using the algorithm described above. Note that no advantage can be taken of the triangular structure of R in the Householder algorithm. Already the first postmultiplication of R with P_1 will cause the lower triangular part of R to fill in. Hence, the Householder reduction of R to bidiagonal form will require $\frac{4}{3}n^3$ flops. The complete reduction to bidiagonal form then takes a total of

$$2(mn^2 + n^3) \text{ flops.}$$

This is less than the original Golub–Kahan algorithm when $m/n > 5/3$.

8.4.6 Regularization by TSVD and Partial Least Squares

In Example 7.1.5 we considered an integral equation of the first kind.

$$\int K(s,t)f(t)dt = g(s), \tag{8.4.36}$$

where the operator K is compact. This is an ill-posed problem in the sense that the solution f does not depend continuously on the data g. This is because there are rapidly oscillating functions $f(t)$, which come arbitrarily close to being annihilated by the integral operator. A shown in In Example 7.1.5, when the the integral equation is discretized, this gives rise to a linear system $Ax = b$, or, more generally, a linear least squares problem

$$\min_x \|Ax - b\|_2, \quad A \in \mathbf{R}^{m\times n}, \tag{8.4.37}$$

The singular values σ_i of A will cluster at zero leading to a huge condition number of A. However, the *exact* right hand side b will be such that in the expansion of the solution in terms of the SVD

$$x_i = \sum_{i=1}^{n} \frac{c_i}{\sigma_i} v_i, \quad c_i = u_i^T b, \tag{8.4.38}$$

the coefficients c_i decrease faster than σ_i. This is a consequence of the **Picard condition** for the continuous problem. In spite the large condition number of A, the problem is in some sense quite well-conditioned. However, in practice, the exact right hand side is cantaminated by noise in a way that affects *all coefficients* c_i in (8.4.38) more or less equally. Therefore, unless some sort of regularization is introduced, the computed solution may be useless.

The solution to the linear least squares problem (8.4.37) can be expressed in terms of the SVD expansion

$$x = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} \cdot v_i.$$

If the matrix A is ill-conditioned and possibly rank deficient. with numerical rank $k < n$, then a more stable approximate solution is obtained by discarding terms

corresponding to small singular values in this expansion. This gives the **truncated SVD** (TSVD) solution

$$x = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} \cdot v_i. \tag{8.4.39}$$

The TSVD solution is a reliable way to determine an approximate pseudo-inverse solution of a numerically rank deficient least squares problems. However, it is also an expensive method since it requires the computation of the SVD. Often a truncated QR factorization works well, provided that some form of column pivoting is carried out. An alternative to truncated SVD is to constrain the solution by adding a quadratic constraint; see Section 8.6.4.

Example 8.4.1.

The linear system $Ax = b$, $A \in \mathbf{R}^{n\times n}$ in Example 7.1.5 was derived from a discretization of an integral equation of the first kind. For $n = 100$ the singular values σ_k of A are close to the roundoff level for $k > 30$. Hence, the linear system is highly ill-conditioned. Let $b = Ax + \eta$ be a perturbed right-hand side, where x is a smooth solution and η a random noise vector with normally distributed components of mean zero and variance 10^{-3}. Figure 8.4.1 shows that the smallest error occurs

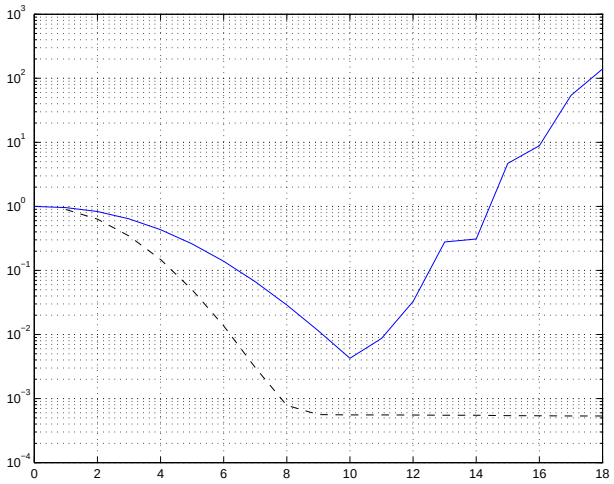

Figure 8.4.1. *Relative error* $\|x_k - x\|_2/\|x\|_2$ *and residual* $\|Ax_k - b\|_2/\|b\|_2$ *for TSVD solutions truncated after* k *steps.*

for $k = 10$. For larger values of k the error increases very rapidly. In practice, the error is unlnown and we can only observe the residual. If the 'noise level in the right hand side is known, the expansion can be truncated as soon as the residual norm has dropped below this level. This is known as the **discrepacy principle**.

In scientific computing one often has one set of variables called the factors that is used to control, explain, or predict another variable. In multiple linear regression

the set of factors may be large and possible highly collinear. Then one wants to express the solution by restricting it to lie in a lower dimensional subspace. One way to achieve this is to use a TSVD solution, in which a subspace spanned by the right singular vectors corresponding to the large singular values is used.

An alternative is to use a solution computed from a partial bidiagonalization of $(\,b \quad A\,)$. This is equivalent to the **partial least squares** (PLS) method. It has been observed that compared to TSVD, PLS gives the same accuracy often with a lower dimensional subspace

We first derive an algorithm for solving the linear least squares problem $\min \|Ax - b\|_2$, where $A \in \mathbf{R}^{m\times n}$, $m \geq n$, by bidiagonal decomposition. Let Q_1 be a Householder reflection such that

$$Q_1 b = \beta_1 e_1. \tag{8.4.40}$$

Using the Golub–Kahan algorithm to transform $P_1 A$ to lower triangular form, we obtain

$$U_n^T\,(\,b \quad A\,)\begin{pmatrix} 1 & 0 \\ 0 & V_n \end{pmatrix} = U_n^T\,(\,b \quad AV_n\,) = \begin{pmatrix} \beta_1 e_1 & B_n \\ 0 & 0 \end{pmatrix}, \tag{8.4.41}$$

where e_1 is the first unit vector, $B_n \in\in \mathbf{R}^{(n+1)\times n}$ is lower bidiagonal, and

$$U_n = (u_1, u_2, \ldots, u_n) = Q_1 Q_2 \cdots Q_{n+1}, \tag{8.4.42}$$

$$V_n = (v_1, v_2, \ldots, v_n) = P_1 P_2 \cdots P_{n-1}. \tag{8.4.43}$$

(Note the minor difference in notation in that Q_{k+1} now zeros elements in the kth column of A.)

Setting $x = V_n y$ and using the invariance of the l_2-norm it follows that

$$\begin{aligned} \|b - Ax\|_2 &= \left\|(\,b \quad A\,)\begin{pmatrix} -1 \\ x \end{pmatrix}\right\|_2 = \left\|U_n^T\,(\,b \quad AV_n\,)\begin{pmatrix} -1 \\ y \end{pmatrix}\right\|_2 \\ &= \|\beta_1 e_1 - B_n y\|_2. \end{aligned}$$

Hence, if y is the solution to the bidiagonal least squares problem

$$\min_y \|B_n y - \beta_1 e_1\|_2, \tag{8.4.44}$$

then $x = V_n y$ solves the least squares problem $\|Ax - b\|_2$.

To solve (8.4.44) stably we need the QR factorization of $(B_k \mid \beta_1 e_1)$. This can be constructed by premultiplying with a sequence of Givens rotations, where $G_{k,k+1}$, $k = 1, 2, \ldots$ is used to zero the element β_{k+1}

$$Q^T(B_n \mid \beta_1 e_1) = \left(\begin{array}{c|c} R_n & \begin{matrix} f_k \\ \bar{\phi}_{n+1} \end{matrix} \end{array}\right) = \left(\begin{array}{cccc|c} \rho_1 & \theta_2 & & & \phi_1 \\ & \rho_2 & \ddots & & \phi_2 \\ & & \ddots & \theta_n & \vdots \\ & & & \rho_n & \phi_n \\ \hline & & & & \phi_{n+1} \end{array}\right) \tag{8.4.45}$$

where Q is a product of n Givens rotations. The solution y is then obtained by back-substitution from $R_n y = d_n$. The norm of the corresponding residual vector equals $|\bar{\phi}_{n+1}|$.

To zero out the element β_2 we premultiply rows (1,2) with a rotation G_{12}, giving

$$\left(\begin{array}{cc} c_1 & s_1 \\ -s_1 & c_1 \end{array}\right)\left(\begin{array}{cc|c} \alpha_1 & 0 & \beta_1 \\ \beta_2 & \alpha_2 & 0 \end{array}\right) = \left(\begin{array}{cc|c} \rho_1 & \theta_2 & \phi_1 \\ 0 & \bar{\rho}_2 & \bar{\phi}_2 \end{array}\right). \tag{8.4.46}$$

(Here and in the following only elements affected by the rotation are shown.) Here the elements ρ_1, θ_2 and ϕ_1 in the first row are final, but $\bar{\rho}_2$ and $\bar{\phi}_2$ will be transformed into ρ_2 and ϕ_2 in the next step.

Continuing in this way in step j the rotation $G_{j,j+1}$ is used to zero the element β_{j+1}. In steps, $j = 2 : n-1$, the rows $(j, j+1)$ are transformed

$$\left(\begin{array}{cc} c_j & s_j \\ -s_j & c_j \end{array}\right)\left(\begin{array}{cc|c} \bar{\rho}_j & 0 & \bar{\phi}_j \\ \beta_{j+1} & \alpha_{j+1} & 0 \end{array}\right) = \left(\begin{array}{cc|c} \rho_j & \theta_{j+1} & \phi_j \\ 0 & \bar{\rho}_{j+1} & \bar{\phi}_{j+1} \end{array}\right). \tag{8.4.47}$$

where

$$\begin{aligned} &\phi_j = c_j\bar{\phi}_j, \qquad \bar{\phi}_{j+1} = -s_j\bar{\phi}_j, \qquad \rho_j = \sqrt{\bar{\rho}_j^2 + \beta_{j+1}^2}, \\ &\theta_{j+1} = s_j\alpha_{j+1}, \qquad \bar{\rho}_{n+1} = c_j\alpha_{j+1}. \end{aligned}$$

Note that by construction $|\bar{\phi}_{j+1}| \le \bar{\phi}_j$. Finally, in step n we obtain

$$\left(\begin{array}{cc} c_n & s_n \\ -s_n & c_n \end{array}\right)\left(\begin{array}{cc|c} \bar{\rho}_n & \bar{\phi}_n \\ \beta_{n+1} & 0 \end{array}\right) = \left(\begin{array}{cc|c} \rho_n & \phi_n \\ 0 & \bar{\phi}_{n+1} \end{array}\right). \tag{8.4.48}$$

After n steps, we have obtained the factorization (8.4.45) with

$$G_n = G_{n,n+1}\cdots G_{23}G_{12}.$$

Now consider the result after $k < n$ steps of the above bidiagonalization process have been carried out. At this point we have computed $Q_1, Q_2, \ldots, Q_{k+1}$, $P_1, P_2, \ldots, P_k$ such that the first k columns of A are in lower bidiagonal form, i.e.

$$Q_{k+1}\cdots Q_2Q_1\, A\, P_1P_2\cdots P_k\begin{pmatrix} I_k \\ 0 \end{pmatrix} = \begin{pmatrix} B_k \\ 0 \end{pmatrix} = \begin{pmatrix} I_{k+1} \\ 0 \end{pmatrix} B_k,$$

where $B_k \in \mathbf{R}^{(k+1)\times k}$ is a leading submatrix of B_n. Multiplying both sides with $Q_1Q_2\cdots Q_{k+1}$ and using orthogonality we obtain the relation

$$AV_k = U_{k+1}B_k = \hat{B}_k + \beta_{k+1}v_{k+1}e_k^T, \quad k = 1 : n, \tag{8.4.49}$$

where

$$\begin{aligned} &P_1P_2\cdots P_k\begin{pmatrix} I_k \\ 0 \end{pmatrix} = V_k = (v_1, \ldots, v_k), \\ &Q_1Q_2\cdots Q_{k+1}\begin{pmatrix} I_{k+1} \\ 0 \end{pmatrix} = U_{k+1} = (u_1, \ldots, u_{k+1}). \end{aligned}$$

If we consider the intermediate result after applying also P_{k+1} the first $k+1$ rows have been transformed into bidiagonal form, i.e.

$$(\, I_{k+1} \quad 0\,)\, Q_{k+1} \cdots Q_2 Q_1\, A\, P_1 P_2 \cdots P_{k+1} = (\, B_k \quad \alpha_{k+1} e_{k+1}\,)\,(\, I_{k+1} \quad 0\,).$$

Transposing this gives a second relation

$$U_{k+1}^T A = B_k V_k^T + \alpha_{k+1} e_{k+1} v_{k+1}^T, \tag{8.4.50}$$

An nice feature of PLS is that if a zero element occurs in B, then the algorithm stops with an exact least squares solution. As an example, consider the case below, where the first two columns of B have been computed ($m = 7$, $n = 5$).

$$Q_3 Q_2 Q_1\,(\, b \quad A\,)\, P_1 P_2 = \left(\begin{array}{ccc|ccc} \beta_1 & \alpha_1 & & & & \\ & \beta_2 & \alpha_2 & & & \\ & & \beta_3 & \times & \otimes & \otimes \\ \hline & & & \times & \times & \times \\ & & & \otimes & \times & \times \\ & & & \otimes & \times & \times \\ & & & \otimes & \times & \times \end{array}\right),$$

In the next step elements in the third row are zeroed out and α_3 determined. If $\alpha_3 = 0$, then the problem decomposes and an overdetermined core system of dimension 3×2 can be extracted. If $\alpha_3 \neq 0$, then elements in the third column of the transformed matrix A are zeroed and β_3 determined. If $\beta_4 = 0$, then the problem decomposes and a square core system of dimension 3×3 can be extracted.

In general, assume that the first zero element to occur is $\alpha_{k+1} = 0$. Then we have obtained the decomposition

$$\widetilde{U}_{k+1}^T A \widetilde{V}_k = \begin{pmatrix} B_k & 0 \\ 0 & A_k \end{pmatrix},$$

where $A_k \in \mathbf{R}^{(m-k-1)\times(n-k)}$, and

$$\widetilde{U}_{k+1} = Q_{k+1} \cdots Q_2 Q_1, \qquad \widetilde{V}_k = P_1 P_2 \cdots P_k,$$

are square orthogonal matrices. Then, setting $x = \widetilde{V}_k y$, the transformed least squares problem takes the form

$$\min_y \left\| \begin{pmatrix} B_k & 0 \\ 0 & A_k \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} \right\|_2, \quad y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \tag{8.4.51}$$

$y_1 \in \mathbf{R}^k$, $y_2 \in \mathbf{R}^{n-k}$. This problem is separable and decomposes into two independent subproblems

$$\min_{y_1} \|B_k y_1 - \beta_1 e_1\|_2, \qquad \min_{y_2} \|A_k y_2\|_2. \tag{8.4.52}$$

By construction B_k has nonzero elements in its two diagonals. Thus, it has full column rank and the solution y_1 to the first subproblem is unique. Further, the

minimum norm solution of the initial problem is obtained simply by taking $y_2 = 0$. The first subproblem (8.4.52) is called a **core subproblem**. It can be solved by QR factorization exactly as outlined for the full system when $k = n$.

When $\beta_{k+1} = 0$ is the first zero element to occur, then the reduced problem has a separable form similar to (8.4.52). The core subproblem is

$$\hat{B}_k y_1 = \beta_1 e_1, \quad \hat{B}_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \end{pmatrix} \in \mathbf{R}^{k\times k}. \tag{8.4.53}$$

In the kth step the kth column in L is computed and the kth columns in U and V determined

$$u_k = Q_1 \cdots Q_k e_k, \quad v_k = P_1 \cdots P_k e_k,$$

If $\alpha_k \beta_{k+1} = 0$, the problem decomposes and the reduction can be terminated

Paige and Strakoš [471] have shown the following important properties of the core subproblem obtained by the bidiagonalization algorithm:

Theorem 8.4.3.

Assume that the bidiagonalization of $(b \quad A)$ *terminates prematurely with* $\alpha_k = 0$ *or* $\beta_{k+1} = 0$. *Then the core corresponding subproblem* (8.4.52) *or* (8.4.53) *is minimally dimensioned. Further, the singular values of the core matrix* B_k *or* $\hat{B}_k$, *are simple and the right hand side* βe_1 *has nonzero components in along each left singular vector.*

Proof. Sketch: The minimal dimension is a consequence of the uniqueness of the decomposition (8.4.41), as long as no zero element in B appears. That the matrix $\hat{B}_k$ has simple singular values follows from the fact that all subdiagonal elements are nonzero. The same is true for the square bidiagonal matrix $(B_k \; 0)$ and therefore also for B_k. Finally, if βe_1 did not have nonzero components along a left singular vector, then the reduction must have terminated earlier. For a complete proof we refer to [471].) □

We remark that the solution steps can be interleaved with the reduction to bidiagonal form. This makes it possible to compute a sequence of *approximate solutions* $x_k = P_1 P_2 \cdots P_k y_k$, where $y_k \in \mathbf{R}^k$ solves

$$\min_y \|\beta_1 e_1 - B_k y\|_2, \quad k = 1, 2, 3, \ldots. \tag{8.4.54}$$

After each (double) step in the bidiagonalization we advance the QR decomposition of B_k. The norm of the least squares residual corresponding to x_k is then given by

$$\|b - Ax_k\|_2 = |\bar{\phi}_{k+1}|.$$

The sequence of residual norms is nonincreasing. We stop and accept $x = V_k y_k$ as an approximate solution of the original least squares problem if this residual is sufficiently small.

The PLS algorithm often requires a lower dimensional subspace than TSVD to achieve similar accuracy. A heuristic explanation for this observation is that the TSVD subspaces do not depend on b and are equally suitable for any right-hand side. The Krylov subspaces, on the other hand, are tailored to the particular right-hand side b of the problem; see Hanke [306].

The PLS method can also be implemented using a Lanczos-type algorithm for bidiagonalization. This algorithm, called LSQR, Paige and Saunders [470, 469] is suitable for solving *sparse* linear least squares. A number of important properties of the successive approximations x_k in PLS are best discussed in connection with LSQR¡, see Algorithm11.3.4.

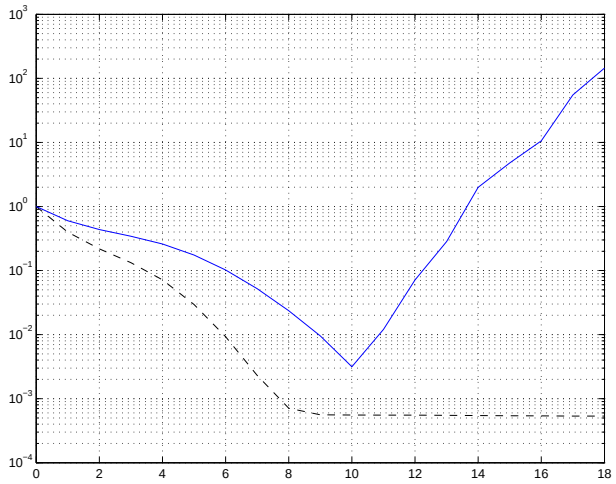

Figure 8.4.2. *Relative error* $\|x_k - x\|_2/\|x\|_2$ *and residual* $\|Ax_k - b\|_2/\|b\|_2$ *for PLS solutions after* k *steps.*

Example 8.4.2.

Consider the ill-posed linear system $Ax = b$, $A \in \mathbf{R}^{n\times n}$ in Example 8.4.1. Figure 8.4.2 shows the relative error $\|x_k - x\|_2/\|x\|_2$ and residual $\|Ax_k - b\|_2/\|b\|_2$ after k steps of PLS. For this problem the results are almost identical for PLS and TSVD. Note that in PLS only a partial bidiagonalization of A and generation of the matrix V_k is needed.

The PLS method can also be implemented using a Lanczos-type algorithm for bidiagonalization. This algorithm, called LSQR, is suitable for solving *sparse* linear least squares. A number of important properties of the successive approximations x_k in PLS are best discussed in connection with LSQR; see Section 10.7.4. The Householder algorithm described above is to be preferred for dense problems, because of its superior numerical stability.

Review Questions

4.1 When and why should column pivoting be used in computing the QR factorization of a matrix? What inequalities will be satisfied by the elements of R if the standard column pivoting strategy is used?

4.2 Show that the singular values and condition number of R equal those of A. Give a simple lower bound for the condition number of A in terms of its diagonal elements. Is it advisable to use this bound when no column pivoting has been performed?

4.3 Give a simple lower bound for the condition number of A in terms of the diagonal elements of R. Is it advisable to use this bound when no column pivoting has been performed?

4.4 What is meant by a Rank-revealing QR factorization? Does such a factorization always exist?

4.5 How is the *numerical* rank of a matrix A defined? Give an example where the numerical rank is not well determined.

Problems

4.1 (a) Describe how the QR factorizations of a matrix of the form

$$\begin{pmatrix} A \\ \mu D \end{pmatrix}, \quad A \in \mathbf{R}^{m\times n},$$

where $D \in \mathbf{R}^{n\times n}$ is diagonal, can be computed using Householder transformations in mn^2 flops.

(b) Estimate the number of flops that are needed for the reduction using Householder transformations in the special case that $A = R$ is upper triangular? Devise a method using Givens rotations for this special case!

Hint: In the Givens method zero one diagonal at a time in R working from the main diagonal inwards.

4.2 Let the vector v, $\|v\|_2 = 1$, satisfy $\|Av\|_2 = \epsilon$, and let Π be a permutation such that

$$|w_n| = \|w\|_\infty, \qquad \Pi^T v = w.$$

(a) Show that if R is the R factor of $A\Pi$, then $|r_{nn}| \le n^{1/2}\epsilon$.

Hint: Show that $\epsilon = \|Rw\|_2 \ge |r_{nn}w_n|$ and then use the inequality $|w_n| = \|w\|_\infty \ge n^{-1/2}\|w\|_2$.

(b) Show using (a) that if $v = v_n$, the right singular vector corresponding to the smallest singular value $\sigma_n(A)$, then

$$\sigma_n(A) \ge n^{-1/2}|r_{nn}|.$$

4.3 Consider a nonsingular 2×2 upper triangular matrix and its inverse

$$R = \begin{pmatrix} a & b \\ 0 & c \end{pmatrix}, \qquad R^{-1} = \begin{pmatrix} a^{-1} & a^{-1}bc^{-1} \\ 0 & c^{-1} \end{pmatrix}.$$

(a) Suppose we want to choose Π to *maximize* the $(1,1)$ element in the QR factorization of $R\Pi$. Show that this is achieved by taking $\Pi = I$ if $|a| \geq \sqrt{b^2 + c^2}$, else $\Pi = \Pi_{12}$, where Π_{12} interchanges columns 1 and 2.

(b) Unless $b = 0$ the permutation chosen in (a) may not *minimize* the (2,2) element in the QR factorization of $R\Pi$. Show that this is achieved by taking $\Pi = I$ if $|c^{-1}| \geq \sqrt{a^{-2} + b^2(ac)^{-2}}$ else $\Pi = \Pi_{12}$. Hence, the test compares *row* norms in R^{-1} instead of *column* norms in R.

4.6 To minimize $\|x\|_2$ is not always a good way to resolve rank deficiency, and therefore the following generalization of problem (8.4.5) is often useful: For a given matrix $B \in \mathbf{R}^{p\times n}$ consider the problem

$$\min_{x\in S} \|Bx\|_2, \qquad S = \{x \in \mathbf{R}^n \mid \|Ax - b\|_2 = \min\}.$$

(a) Show that this problem is equivalent to

$$\min_{x_2} \|(BC)x_2 - (Bd)\|_2,$$

where C and d are defined by (8.4.4).

(b) Often one wants to choose B so that $\|Bx\|_2$ is a measure of the smoothness of the solution x. For example one can take B to be a discrete approximation to the second derivative operator,

$$B = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -2 & 1 \end{pmatrix} \in \mathbf{R}^{(n-2)\times n}.$$

Show that provided that $\mathcal{N}(A) \cap \mathcal{N}(B) = \emptyset$ this problem has a unique solution, and give a basis for $\mathcal{N}(B)$.

4.5 Let $A \in \mathbf{R}^{m\times n}$ with $\operatorname{rank}(A) = r$. A rank revealing LU factorizations of the form

$$\Pi_1 A \Pi_2 = \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \end{pmatrix},$$

where Π_1 and Π_2 are permutation matrices and $L_{11}, U_{11} \in \mathbf{R}^{r\times r}$ are triangular and nonsingular can also be used to compute pseudo-inverse solutions $x = A^{\dagger}b$. Show, using Theorem 8.1.12 that

$$A^{\dagger} = \Pi_2 \begin{pmatrix} I_r & S \end{pmatrix}^{\dagger} U_{11}^{-1} L_{11}^{-1} \begin{pmatrix} I_r \\ T \end{pmatrix}^{\dagger} \Pi_1,$$

where $T = L_{21}L_{11}^{-1}$, $S = U_{11}^{-1}U_{12}$. (Note that S is empty if $r = n$, and T empty if $r = m$.)

4.6 Consider the block upper-bidiagonal matrix

$$A = \begin{pmatrix} B_1 & C_1 & \\ & B_2 & C_2 \\ & & B_3 \end{pmatrix}$$

Outline an algorithm for computing the QR factorization of A, which treats one block row at a time. (It can be assumed that A has full column rank.) Generalize the algorithm to an arbitrary number of block rows!

4.7 (a) Suppose that we have computed the pivoted QR factorization of A,

$$Q^T A\Pi = \begin{pmatrix} R \\ 0 \end{pmatrix} \in \mathbf{R}^{m\times n},$$

of a matrix $A \in \mathbf{R}^{m\times n}$. Show that by postmultiplying the *upper* triangular matrix R by a sequence of Householder transformations we can transform R into a *lower* triangular matrix $L = RP \in \mathbf{R}^{n\times n}$ and that by combining these two factorizations we obtain

$$Q^T A\Pi P = \begin{pmatrix} L \\ 0 \end{pmatrix}. \tag{8.4.55}$$

Comment: This factorization, introduced by G. W. Stewart, is equivalent to one step of the basic unshifted QR–SVD algorithm; see Section 10.4.1.

(b) Show that the total cost for computing the QLP decomposition is roughly $2mn^2 + 2n^3/3$ flops. How does that compare with the cost for computing the bidiagonal decomposition of A?

(c) Show that the two factorizations can be interleaved. What is the cost for performing the first k steps?

4.8 Work out the details of an algorithm for transforming a matrix $A \in \mathbf{R}^{m\times n}$ to *lower* bidiagonal form. Consider both cases when $m > n$ and $m \le n$.

Hint: It can be derived by applying the algorithm for transformation to upper bidiagonal form to A^T.

4.9 Trefethen and Bau [575, pp. 237–238] have suggested a blend of the Golub–Kahan and Chan methods for bidiagonal reduction when $n > m > 2n$. They note that after k steps of the Golub–Kahan reduction the aspect ratio of the reduced matrix is $(m-k)/(n-k)$. and thus increases with k.
Show that to minimize the total operation count one should switch to the Chan algorithm when $(m-k)/(n-k) = 2$. What is the resulting operation count ?

8.5 Structured and Sparse Least Squares Problems

8.5.1 Banded Least Squares Problems

We now consider orthogonalization methods for the special case when A is a banded matrix of row bandwidth w, see Definition 8.2.2. From Theorem 8.2.3 we know that

A^TA will be a symmetric band matrix with only the first $r = w - 1$ super-diagonals nonzero. Since the factor R in the QR factorization equals the unique Cholesky factor of A^TA it will have only w nonzeros in each row.

Even though the *final* factor R is independent of the row ordering in A, the intermediate fill will vary. For banded rectangular matrices the QR factorization can be obtained efficiently by sorting the rows of A and suitably subdividing the Householder transformations. The rows of A should be sorted by leading entry order (i.e., increasing minimum column subscript order) That is, if $f_i, i = 1 : m$ denotes the column indices of the first nonzero element in row i we should have,

$$i \le k \Rightarrow f_i \le f_k.$$

Such a band matrix can then be written as

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

is said to be in **standard form**. 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 factorization 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 has 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 [399, Ch. 27], requires $(m + 3n/2)w(w + 1)$ multiplications or about twice the work of the less stable Cholesky approach. For a detailed description of this algorithm, see Lawson and Hanson [399, Ch. 11].

In Volume I, Section 4.4.4 we considered the interpolation of a function f with a linear combination of $m + k$ cubic B-splines k on the grid $\Delta = \{x_0 < x_1 < \cdots < x_m\}$. If function values $f_j = f(\tau_j)$, are given at distinct points $\tau_1 < \tau_2 < \ldots < \tau_n$, where $n \ge m + k$. This leads to a least squares approximation problem

$$\min \sum_{j=1}^{n} e_j^2, \qquad e_j = w_i \Big(f_j - \sum_{i=-k}^{m-1} c_i B_{i,k+1}(\tau_j) \Big). \tag{8.5.1}$$

where w_j are positive weights. This is an overdetermined linear system for c_i, $i = -k, \ldots, m-1$. The elements of its coefficient matrix $B_{i,k+1}(\tau_j)$ can be evaluated by the recurrence (4.6.19). The coefficient matrix has a band structure since in the jth row the ith element will be zero if $\tau_j \notin [x_i, x_{i+k+1}]$. It can be shown, see de Boor [1978, p. 200], that the coefficient matrix will have full rank equal to $m + k$ if and only if there is a subset of points τ_j satisfying

$$x_{j-k-1} < \tau_j < x_j, \qquad \forall j = 1 : m + k. \tag{8.5.2}$$

Example 8.5.1.

The least squares approximation of a discrete set of data by a linear combination of cubic B-splines gives rise to a banded linear least squares problem. Let

$$s(t) = \sum_{j=1}^{n} x_j B_j(t),$$

where $B_j(t)$, $j = 1 : n$ are the normalized cubic B-splines, and let (y_i, t_i), $i = 1 : m$ be given data points. If we determine x to minimize

$$\sum_{i=1}^{m} (s(t_i) - y_i)^2 = \|Ax - y\|_2^2,$$

then A will be a banded matrix with $w = 4$. In particular, if $m = 13$, $n = 8$ the matrix may have the form shown in Figure 8.5.1. Here A consists of blocks A_k^T, $k = 1 : 7$. In the Figure 8.5.1 we also show the matrix after the first three blocks have been reduced by Householder transformations $P_1, \ldots, P_9$. Elements which have been zeroed by P_j are denoted by j and fill elements by $+$. In step $k = 4$ only the indicated part of the matrix is involved.

$$\begin{pmatrix}
\times & \times & \times & \times & & & & \\
1 & \times & \times & \times & + & & & \\
1 & 2 & \times & \times & + & + & & \\
 & 3 & 4 & \times & \times & + & & \\
 & 3 & 4 & 5 & \times & + & & \\
 & & 6 & 7 & 8 & \times & & \\
 & & 6 & 7 & 8 & 9 & & \\
 & & 6 & 7 & 8 & 9 & & \\
 & & & \times & \times & \times & \times & \\
 & & & \times & \times & \times & \times & \\
 & & & & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times
\end{pmatrix}$$

Figure 8.5.1. *The QR factorization of a banded rectangular matrix A.*

In the algorithm the Householder transformations can also be applied to one or several right hand sides b to produce

$$c = Q^T b = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}, \quad c_1 \in \mathbf{R}^n.$$

The least squares solution is then obtained from $Rx = c_1$ by back-substitution. The vector c_2 is not stored but used to accumulate the residual sum of squares $\|r\|_2^2 = \|c_2\|_2^2$.

It is also possible to perform the QR factorization by treating one row at a time using Givens' rotations. Each step then is equivalent to updating a full

triangular matrix formed by columns $f_i(A)$ to $l_i(A)$. Further, if the matrix A is in standard form the first $f_i(A)$ rows of R are already finished at this stage. The reader is encouraged to work through Example 8.5.1 below in order to understand how the algorithm proceeds!

A special form of banded system that occurs in many applications is the **block-bidiagonal form**,

$$\begin{pmatrix} A_1 & B_1 & & & \\ & A_2 & B_2 & & \\ & & A_3 & \ddots & \\ & & & \ddots & B_{n-1} \\ & & & & A_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix}, \tag{8.5.3}$$

Assume that by orthogonal transformations the k first block rows have been reduced to the upper triangular form

$$\begin{pmatrix} R_1 & S_1 & & & \\ & R_2 & S_2 & & \\ & & \ddots & \ddots & \\ & & & R_{k-1} & S_{k-1} \\ & & & & \bar{R}_k \end{pmatrix}. \tag{8.5.4}$$

The last diagonal block here is the only one to be altered in the next step. In step $k+1$ the blocks A_{k+1} and B_{k+1} are added to the matrix together with observations b_{k+1}. We can now choose an orthogonal transformations that performs the reduction to upper triangular form

$$Q_k^T \begin{pmatrix} \bar{R}_k & 0 \\ A_{k+1} & B_{k+1} \end{pmatrix} = \begin{pmatrix} R_k & S_k \\ & \bar{R}_{k+1} \\ 0 & 0 \end{pmatrix}.$$

The above structure arises in a least squares formulation of the Kalman filter, which is a popular method used to estimate the behavior of certain linear dynamic systems; see Paige and Saunders [466] for details.

8.5.2 Blocked Form of QR Factorization

In many least squares problems the unknowns x can be naturally partitioned into two groups with n_1 and n_2 components, respectively. Then the problem has the form

$$\min_{x_1,x_2} \left\| \begin{pmatrix} A_1 & A_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - b \right\|_2, \tag{8.5.5}$$

where $A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \in \mathbf{R}^{m\times n}$ and $n = n_1+n_2$. For example, in separable nonlinear least squares problems subproblems of the form (8.5.5) arise, see Section 9.2.4.

Assume that the matrix A has full column rank and let $P_{\mathcal{R}(A_1)}$ be the orthogonal projection onto $\mathcal{R}(A_1)$. For any x_2 we can split the vector $b - A_2x_2 = r_1 + r_2$

into two orthogonal components

$$r_1 = P_{\mathcal{R}(A_1)}(b - A_2x_2), \quad r_2 = (I - P_{\mathcal{R}(A_1)})(b - A_2x_2).$$

Then the problem (8.5.5) takes the form

$$\min_{x_1,x_2} \left\| (A_1x_1 - r_1) - P_{\mathcal{N}(A_1^T)}(b - A_2x_2) \right\|_2. \tag{8.5.6}$$

Here, since $r_1 \in \mathcal{R}(A_1)$ the variables x_1 can always be chosen so that $A_1x_1 - r_1 = 0$. It follows that in the least squares solution to (8.5.5) x_2 is the solution to the reduced least squares problem

$$\min_{x_2} \|P_{\mathcal{N}(A_1^T)}(A_2x_2 - b)\|_2. \tag{8.5.7}$$

where the variables x_1 have been eliminated. When this reduced problem has been solved for x_2 then x_1 can be computed from the least squares problem

$$\min_{x_1} \|A_1x_1 - (b - A_2x_2)\|_2. \tag{8.5.8}$$

Sometimes it may be advantageous to carry out a *partial* QR factorization, where only the $k = n_1 < n$ columns of A are reduced. Using Householder QR

$$Q_1^T(A\ b) = \begin{pmatrix} R_{11} & \widetilde{A}_{12} & \widetilde{b}_1 \\ 0 & \widetilde{A}_{22} & \widetilde{b}_2 \end{pmatrix},$$

with R_{11} nonsingular. Then x_2 is the solution to the reduced least squares problem

$$\min_{x_2} \|\widetilde{b}_2 - \widetilde{A}_{22)}x_2\|_2,$$

If this is again solved by Householder QR nothing new comes out—we have only emphasized a new aspect of this algorithm. However, it may be advantageous to use another method for the reduced problem.

In some applications, e.g., when A has block angular structure (see Section 8.5.3), it may be preferable instead not to save R_{11} and R_{12} and instead to refactorize A_1 and solve (8.5.8) for x_1.

If we use perform n_1 steps of MGS on the full problem, this yields the partial factorization

$$(A\ b) = \begin{pmatrix} Q_1 & \widetilde{A}_2 & \widetilde{b} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} & z_1 \\ 0 & I & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

where R_{11} is nonsingular. The residual is decomposed as as $r = r_1 + r_2$, $r_1 \perp r_2$, where

$$r_1 = Q_1(z_1 - R_{12}x_2 - R_{11}x_1), \qquad r_2 = \widetilde{b}_1 - \widetilde{A}_2x_2.$$

Then x_2 is the solution to the reduced least squares problem $\min_{x_2} \|\widetilde{b} - \widetilde{A}x_2\|_2$. With x_2 known x_1 can been computed by back-substitution from $R_{11}x_1 = z_k - R_{12}x_2$.

To obtain near-peak performance for large dense matrix computations on current computing architectures requires code that is dominated by matrix-matrix

operations since these involve less data movement per floating point computation. The QR factorization should therefore be organized in partitioned or blocked form, where the operations have been reordered and grouped into matrix operations.

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

$$A = (A_1,\ A_2), \qquad A_1 \in \mathbf{R}^{m\times nb}, \tag{8.5.9}$$

where nb is a suitable block size and the QR factorization

$$Q_1^T A_1 = \begin{pmatrix} R_1 \\ 0 \end{pmatrix}, \qquad Q_1 = P_1 P_2 \cdots P_{nb}, \tag{8.5.10}$$

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

$$Q_1^T A_2 = Q_1^T \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} = \begin{pmatrix} R_{12} \\ \widetilde{A}_{22} \end{pmatrix}. \tag{8.5.11}$$

In the next step we partition $\widetilde{A}_{22} = (B_1, B_2)$, and compute the QR factorization of $B_1 \in \mathbf{R}^{(m-r)\times r}$. 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 in spent in the updating step (8.5.11). As written this step cannot use BLAS-3, which slows down the execution. To achieve better performance it is essential that this part is sped up. The solution is to aggregate the Householder transformations so that their application can be expressed as matrix operations. For use in the next subsection, we give a slightly more general result.

Lemma 8.5.1.

Let $P_1, P_2, \ldots, P_r$ be a sequence of Householder transformations. Set $r = r_1 + r_2$, and assume that

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

where $T_1, T_2 \in \mathbf{R}^{r\times r}$ are upper triangular matrices. Then for the product $Q_1 Q_2$ we have

$$Q = Q_1 Q_2 = (I - Y_1 T_1 Y_1^T)(I - Y_2 T_2 Y_2^T) = (I - Y T Y^T) \tag{8.5.12}$$

where

$$\hat{Y} = (Y_1,\ Y_2), \qquad \hat{T} = \begin{pmatrix} T_1 & -(T_1 Y_1^T)(Y_2 T_2) \\ 0 & T_2 \end{pmatrix}. \tag{8.5.13}$$

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

$$Y := (Y,\ u_{k+1}), \qquad T := \begin{pmatrix} T & -\tau_k T Y^T u_k \\ 0 & \tau_k \end{pmatrix}, \quad k = 2 : nb. \tag{8.5.14}$$

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

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

which now involves only matrix operations. 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.

As mentioned in Chapter 7 recursive algorithms can be developed into highly efficient algorithms for high performance computers and are an alternative to the currently more used partitioned algorithms 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.

Consider the partitioned QR factorization

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

where Let A_1 consist of the first $\lfloor n/2 \rfloor$ columns of A. To develop a recursive algorithm we start with a QR factorization of A_1 and update the remaining part A_2 of the matrix,

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

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

As an illustration we give below 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
    [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 algorithm uses the function house(a) to compute a Householder transformation $P = I - +\tau\, uu^T$ such that $Pa = \sigma\, e1$, $\sigma = -\text{sign}\,(a_1)\|a\|_2$. A serious defect of this algorithm is the overhead in storage and operations caused by the T matrices. In the partitioned algorithm n/nb T-matrices of size we 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. Therefore, a better solution is to use a hybrid between the partitioned and the recursive algorithm, where the recursive QR algorithm is used to factorize the blocks in the partitioned algorithm.

8.5.3 Block Angular Least Squares Problems

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**:

$$A = \left(\begin{array}{cccc|c} A_1 & & & & B_1 \\ & A_2 & & & B_2 \\ & & \ddots & & \vdots \\ & & & A_M & B_M \end{array}\right), \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \\ x_{M+1} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{pmatrix}, \tag{8.5.15}$$

where

$$A_i \in \mathbf{R}^{m_i \times n_i}, \quad B_i \in \mathbf{R}^{m_i \times n_{M+1}}, \quad i = 1 : M,$$

and

$$m = m_1 + m_2 + \cdots + m_M, \qquad n = n_1 + n_2 + \cdots + n_{M+1}.$$

Note that the variables $x_1, \ldots, x_M$ are coupled only to the variables x_{M+1}, which reflects a "local connection" structure in the underlying physical problem. Applications where the form (8.5.15) arises naturally include photogrammetry, Doppler radar and GPS positioning, and geodetic survey problems. The block matrices A_i, B_i $i = 1 : M$, may also have some structure that can be taken advantage of, but in the following we ignore this; see Cox [122].

The normal matrix of (8.5.15) has a doubly bordered block diagonal, or ar-

rowhead, form,

$$A^TA = \left(\begin{array}{cccc|c} A_1^TA_1 & & & & A_1^TB_1 \\ & A_2^TA_2 & & & A_2^TB_2 \\ & & \ddots & & \vdots \\ & & & A_M^TA_M & A_M^TB_M \\ \hline B_1^TA_1 & B_2^TA_2 & \cdots & B_M^TA_M & C \end{array}\right),$$

where

$$C = \sum_{k=1}^{M} B_k^TB_k.$$

We assume in the following that $\operatorname{rank}(A) = n$, which implies that the Cholesky factor R of A^TA is nonsingular. It will have a block structure similar to that of A,

$$R = \left(\begin{array}{cccc|c} R_1 & & & & S_1 \\ & R_2 & & & S_2 \\ & & \ddots & & \vdots \\ & & & R_M & S_M \\ \hline & & & & R_{M+1}. \end{array}\right), \tag{8.5.16}$$

Identifying the blocks in the relation $R^TR = A^TA$ we get

$$R_i^TR_i = A_i^TA_i, \qquad R_i^TS_i = A_i^TB_i, \quad i = 1:M,$$
$$R_{M+1}^TR_{M+1} + \sum_{i=1}^{M} S_i^TS_i. = C.$$

Hence, $R_i \in \mathbf{R}^{n_i \times n_i}$, $i = 1:M$, are the Cholesky factors of $A_i^TA_i$ and

$$S_i = R_i^{-T}(A_i^TB_i), \quad i = 1:M.$$

Finally, R_{M+1} is obtained as the Cholesky factor of

$$\widetilde{C} = C - \sum_{i=1}^{M} S_i^TS_i.$$

The matrix R in (8.5.16) can also be computed by a sequence of QR factorizations. The following algorithm solves least squares problems of block angular form based using QR factorization. It proceeds in three steps:

1. Initialize an upper triangular R_{M+1} of dimension n_{M+1}, a vector c_{M+1} and a scalar ρ to zero.

2. For $i = 1:M$

(a) Reduce the blocks (A_i, B_i) and the right-hand side b_i by orthogonal transformations, yielding

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

where R_i is upper triangular.

(b) Apply orthogonal transformations to update

$$\begin{pmatrix} R_{M+1} & c_{M+1} \\ T_i & d_i \end{pmatrix} \quad \text{to yield} \quad \begin{pmatrix} R_{M+1} & c_{M+1} \\ 0 & e_i \end{pmatrix}$$

(c) Update the residual norm $\rho = (\rho^2 + \|e_i\|_2^2)^{1/2}$.

3. Compute x_{M+1} from the triangular system

$$R_{M+1}x_{M+1} = c_{M+1},$$

4. For $i = 1 : M$ compute x_i by back-substitution in the triangular systems

$$R_i x_i = c_i - S_i x_{M+1}.$$

In steps 2 (a) and 4 the computations can be performed in parallel on the M subsystems. There are alternative ways to organize this algorithm. When x_{M+1} has been computed, then x_i, solves the least squares problem

$$\min_{x_i} \|A_i x_i - g_i\|_2, \quad g_i = b_i - B_i x_{M+1}, \quad i = 1 : M.$$

Hence, it is possible to discard the R_i, S_i and c_i in step 1 if the QR factorizations of A_i are recomputed in step 3. In some practical problems this modification can reduce the storage requirement by an order of magnitude, while the recomputation of R_i may only increase the operation count by a few percent.

Using the structure of the R-factor in (8.5.15), the diagonal blocks of the variance-covariance matrix $C = (R^T R)^{-1} = R^{-1}R^{-T}$ can be written

$$\begin{aligned} &C_{M+1,M+1} = R_{M+1}^{-1} R_{M+1}^{-T}, \\ &C_{i,i} = R_i^{-1}(I + W_i^T W_i) R_i^{-T}, \quad W_i^T = S_i R_{M+1}^{-1}, \quad i = 1, \ldots, M. \end{aligned} \tag{8.5.17}$$

If we compute the QR factorizations

$$Q_i \begin{pmatrix} W_i \\ I \end{pmatrix} = \begin{pmatrix} U_i \\ 0 \end{pmatrix}, \quad i = 1, \ldots, M,$$

we have $I + W_i^T W_i = U_i^T U_i$ and then

$$C_{i,i} = (U_i R_i^{-T})^T (U_i R_i^{-T}), \quad i = 1, \ldots, M.$$

This assumes that all the matrices R_i and S_i have been retained.

A procedure, called substructuring or dissection can often be used for obtaining a block angular form of a problem. The idea is similar but preceeded the

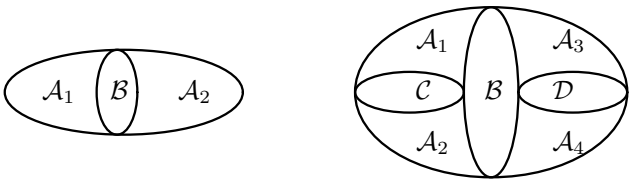

Figure 8.5.2. *One and two levels of dissection of a region.*

nested dissection method presented in Section 7.7.5. Such a technique dates back to Helmert [316, 1880], who used it for breaking down geodetic problems into geographically defined subproblems. Consider a geodetic position network consisting of geodetic stations connected through observations. To each station corresponds a set of unknown coordinates to be determined. The idea is to choose a set of stations $\mathcal{B}$, which separates the other stations into two regional blocks $\mathcal{A}_1$ and $\mathcal{A}_2$ so that station variables in $\mathcal{A}_1$ are not connected by observations to station variables in $\mathcal{A}_2$. The variables are then ordered so that those in $\mathcal{A}_1$ appear first, those in $\mathcal{A}_2$ second, and those in $\mathcal{B}$ last. The equations are ordered so that those including $\mathcal{A}_1$ come first, those including $\mathcal{A}_2$ next, and those only involving variables in $\mathcal{B}$ come last. The dissection can be continued by dissecting the regions $\mathcal{A}_1$ and $\mathcal{A}_2$ each into two subregions, and so on in a recursive fashion. The blocking of the region for one and two levels of dissection is pictured in Figure 8.5.1. The corresponding block structure induced induced in the matrix are

$$A = \left(\begin{array}{cc|c} A_1 & & B_1 \\ & A_2 & B_2 \end{array}\right), \qquad A = \left(\begin{array}{cccc|ccc} A_1 & & & & C_1 & & B_1 \\ & A_2 & & & C_2 & & B_2 \\ & & A_3 & & & D_3 & B_3 \\ & & & A_4 & & D_4 & B_4 \end{array}\right),$$

The block of rows corresponding to $\mathcal{A}_i$, $i = 1, 2, \ldots$, can be processed independently. The remaining variables are then eliminated, etc.; compare the block angular structure in (8.5.15). There is a finer structure in A not shown. For example, in one level of dissection most of the equations involve variables in $\mathcal{A}_1$ or $\mathcal{A}_2$ only, but not in $\mathcal{B}$.

It is advantageous to perform the dissection in such a way that in each stage of the dissection the number of variables in the two partitions are roughly the same. Also, the number of variables in the separator nodes should be as small as possible. If each dissection is done so that the variables contained in the two partitions are at least halved, then after at most $\log^2 n$ levels each partition contains only one variable. Of course, it is usually preferable to stop before this point.

8.5.4 Block Triangular Form

In Section 7.7.7 we showed that before solving a sparse linear system $Ax = b$ it was advantageous to permute the matrix into block triangular form. A similar preprocessing can be done for an arbitrary rectangular matrix $A \in \mathbf{R}^{m\times n}$, $m \geq n$. By row and column permutations the matrix can be brought into the block

triangular form

$$PAQ = \begin{pmatrix} A_h & U_{hs} & U_{hv} \\ & A_s & U_{sv} \\ & & A_v \end{pmatrix}. \tag{8.5.18}$$

Here the diagonal block A_h is underdetermined (i.e., has more columns than rows), A_s is square and A_v is overdetermined (has more rows than columns), and all three blocks have a nonzero diagonal; see the example in Figure 8.5.2. The submatrices A_v and A_h^T both have the strong Hall property. The off-diagonal blocks are possibly nonzero matrices of appropriate dimensions. The block decomposition (8.5.18) of A is called the **coarse decomposition** of A. One or two of the diagonal blocks may be absent in the coarse decomposition. It may be possible to further decompose the

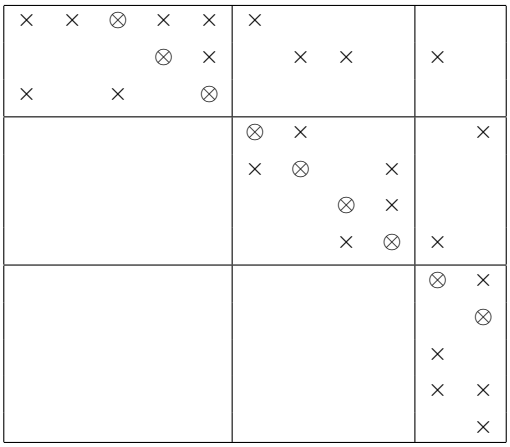

Figure 8.5.3. *The coarse block triangular decomposition of A.*

diagonal blocks in (8.5.18) to obtain the **fine decompositions** of these submatrices. Each of the blocks A_h and A_v may be further decomposable into block diagonal form,

$$A_h = \begin{pmatrix} A_{h1} & & \\ & \ddots & \\ & & A_{hp} \end{pmatrix}, \quad A_v = \begin{pmatrix} A_{v1} & & \\ & \ddots & \\ & & A_{vq} \end{pmatrix},$$

where each $A_{h1}, \ldots, A_{hp}$ is underdetermined and each $A_{v1}, \ldots, A_{vq}$ is overdetermined. The square submatrix A_s may be decomposable in block upper triangular form

$$A_s = \begin{pmatrix} A_{s1} & U_{12} & \ldots & U_{1,t} \\ & A_{s2} & \ldots & U_{2,t} \\ & & \ddots & \vdots \\ & & & A_{st} \end{pmatrix} \tag{8.5.19}$$

where the square diagonal blocks $A_{s1}, \ldots, A_{st}$ have nonzero diagonal elements. The resulting Dulmage–Mendelsohn decomposition can be shown to be essentially

unique. Any one block triangular form can be obtained from any other by applying row permutations that involve the rows of a single block row, column permutations that involve the columns of a single block column, and symmetric permutations that reorder the blocks.

An algorithm for the more general block triangular form described above due to Pothen and Fan [490], uses the concept of m atchings in bipartite graphs. The algorithm consists of the following steps:

1. Find a maximum matching in the bipartite graph $G(A)$ with row set R and column set C.

2. According to the matching, partition R into the sets VR, SR, HR and C into the sets VC, SC, HC corresponding to the horizontal, square, and vertical blocks.

3. Find the diagonal blocks of the submatrix A_v and A_h from the connected components in the subgraphs $G(A_v)$ and $G(A_h)$. Find the block upper triangular form of the submatrix A_s from the strongly connected components in the associated directed subgraph $G(A_s)$, with edges directed from columns to rows.

The reordering to block triangular form in a preprocessing phase can save work and intermediate storage in solving least squares problems. If A has structural rank equal to n, then the first block row in (8.5.18) must be empty, and the original least squares problem can after reordering be solved by a form of block back-substitution. First compute the solution of

$$\min_{\widetilde{x}_v} \|A_v \widetilde{x}_v - \widetilde{b}_v\|_2, \tag{8.5.20}$$

where $\widetilde{x} = Q^T x$ and $\widetilde{b} = Pb$ have been partitioned conformally with PAQ in (8.5.18). The remaining part of the solution $\widetilde{x}_k, \ldots, \widetilde{x}_1$ is then determined by

$$A_{si}\widetilde{x}_i = \widetilde{b}_i - \sum_{j=i+1}^{k} U_{ij}\widetilde{x}_j, \quad i = k : -1 : 1. \tag{8.5.21}$$

Finally, we have $x = Q\widetilde{x}$. We can solve the subproblems in (8.5.20) and (8.5.21) by computing the QR factorizations of A_v and $A_{s,i}$, $i = 1 : k$. Since $A_{s1}, \ldots, A_{sk}$ and A_v have the strong Hall property, the structures of the matrices R_i are correctly predicted by the structures of the corresponding normal matrices.

If the matrix A has structural rank less than n, then we have an underdetermined block A_h. In this case we can still obtain the form (8.5.19) with a square block A_{11} by permuting the extra columns in the first block to the end. The least squares solution is then not unique, but a unique solution of minimum length can be found as outlined in Section 2.7.

8.5.5 Sparse Least Squares Problems

For well-conditioned least squares problems solving the normal equations

$$A^TAx = A^Tb$$

(see Section 8.2) can give quite sufficiently accurate results. In this approach the matrix $C = A^TA$ is formed and its Cholesky factorization $C = LL^T$ computed. The least squares solution is then obtained by solving two triangular systems $L(L^Tx) = A^Tb$. Much of the well developed techniques for solving sparse symmetric positive definite systems can be used; see Section 7.7.5. Let the rows and columns of A be reordered as $B = QAP$, where P and Q are permutation matrices. Since $Q^TQ = I$, it follows that

$$B^TB = P^TA^TQ^TQAP = (AP)^TAP.$$

This shows that a reordering of the rows will not affect the Cholesky factor L.

The first step in computing the Cholesky factorization of a sparse matrix C the is a symbolic analyze phase. In this, using the nonzero structure of C, a fill reducing column permutation P of of C is determined. Simultaneously a storage scheme for the Cholesky factor of PCP^T is set up. To compute the structure of the symmetric matrix A^TA, we partition A by rows. If we let $a_i^T = e_i^TA$ be the ith row of A, then

$$A^TA = \sum_{i=1}^{m} a_i a_i^T, \tag{8.5.22}$$

This expresses A^TA as the sum of m rank one matrices. Invoking the no-cancellation assumption this shows that the nonzero structure of A^TA is the direct sum of the structures of $a_ia_i^T$, $i = 1 : m$. Note that the nonzeros in any row a_i^T will generate a a full submatrix in A^TA. In the graph $G(A^TA)$ this corresponds to asubgraph where all pairs of nodes are connected. Such a subgraph is called a **clique**. Also note that the structure of A^TA is not changed by dropping any row of A whose nonzero structure is a subset of another row. This observation can often speed up the algorithm considerably.

There is an efficient algorithm due to George and Ng [244] to perform the symbolic factorization of A^TA directly using the structure of A. This removes the step of determining the structure of A^TA.

For ill-conditioned or stiff problems methods based on the QR factorization should be preferred. As is well known, the factor R in the QR factorization of A is mathematically equivalent to the upper triangular Cholesky factor L^T of A^TA. In particular, the nonzero structure must be the same.

However, this approach may be too generous in allocating space for nonzeros in R. To see this, consider a matrix with the structure

$$\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \times & \\ & & & & & \times \end{pmatrix} \tag{8.5.23}$$

For this matrix $R = A$ since A is already upper triangular. But, since A^TA is full the algorithm above will predict R to be full. This overestimate occurs because we start with the structure of A^TA, and not with the structure of A. The elements in A^TA are not independent, and cancellation *can* occur in the Cholesky factorization irrespective of the numerical values of the nonzero elements in A. We call this **structural cancellation**,in contrast to numerical cancellation, which occurs only for certain values of the nonzero elements in A.

Another approach to predicting the structure of R is to perform the Givens or Householder algorithm symbolically working on the structure of A. It can be shown that the structure of R as predicted by symbolic factorization of A^TA includes the structure of R as predicted by the symbolic Givens method, which includes the structure of R.

Definition 8.5.2.

A matrix $A \in \mathbf{R}^{m\times n}$, $m \geq n$, is said to have the strong **Hall property** *if for every subset of k columns, $0 < k < m$, the corresponding submatrix has nonzeros in at least $k+1$ rows. (Thus, when $m > n$, every subset of $k \leq n$ has the required property, and when $m = n$, every subset of $k < n$ columns has the property.)*

For matrices with strong Hall propery it can be proved that structural cancellation will not occur. Then the structure of A^TA will correctly predict that of R, excluding numerical cancellations. (If A is orthogonal then $A^TA = I$ is sparse, but this is caused by numerical cancellation.) Obviously the matrix in (8.5.23) does not have the strong Hall property since the first column has only one nonzero element. However, the matrix with the nonzero structure

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & & & & \\ & \times & & & \\ & & \times & & \\ & & & \times & \\ & & & & \times \end{pmatrix} \tag{8.5.24}$$

obtained by deleting the first column has this property.

The next step before performing the numerical phase of the sparse QR factorization is to find a suitable row permutation P_r. Since

$$(P_rA)^T(P_rA) = A^T(P_r^TP_r)A = A^TA,$$

it follows that the resulting factor R is independent of the row ordering. However, the number of intermediate fill and the operation count will depend on the row ordering. This fact was stressed already in the discussion of QR factorization of banded matrices; see Section 8.5.1. Provided the rows of A do not have widely differing norms, a reordering of the rows will not affect the numerical stability. Hence, the ordering can be chosen based on sparsity consideration only. The following heuristic row ordering algorithm is an extension of that recommended for banded sparse matrices.

Algorithm 8.8. *Row Ordering Algorithm.*
Denote the column index for the first and last nonzero elements in the ith row of A by $f_i(A)$ and $l_i(A)$, respectively. First sort the rows after increasing $f_i(A)$, so that $f_i(A) \le f_k(A)$ if $i < k$. Then for each group of rows with $f_i(A) = k$, $k = 1, \ldots, \max_i f_i(A)$, sort all the rows after increasing $l_i(A)$.

An alternative row ordering has been found to work well is obtained by ordering the rows after increasing values of $l_i(A)$. With this ordering only the columns $f_i(A)$ to $l_i(A)$ of R_{i-1} when row a_i^T is being processed will be involved, since all the previous rows only have nonzeros in columns up to at most $l_i(A)$. Hence, R_{i-1} will have zeros in column $l_{i+1}(A), \ldots, n$, and no fill will be generated in row a_i^T in these columns.

We now discuss the numerical phase of sparse QR factorization. For dense problems the most effective serial method for computing is to use a sequence of Householder reflections. In this we put $A^{(1)} = A$, and compute $A^{(k+1)} = P_k A^{(k)}$, $k = 1 : n$, where P_k is chosen to annihilate the subdiagonal elements in the kth column of $A^{(k)}$. In the sparse case this method will cause each column in the remaining unreduced part of the matrix, which has a nonzero inner product with the column being reduced, to take on the sparsity pattern of their union. Hence, even though the final R may be sparse, a lot of intermediate fill may take place with consequent cost in operations and storage. However, as shown in Section 8.5.1, the Householder method can be modified to work efficiently for sparse banded problems, by applying the Householder reductions to a sequence of small dense subproblems.

The problem of intermediate fill in the factorization can be avoided by using instead a **row sequential QR algorithm** employing Givens rotations. Initialize R_0 to bhave the structure of the final factor R with all elements equal to zero. The rows a_k^T of A are processed sequentially, $k = 1 : m$, and we denote by R_{k-1} the upper triangular matrix obtained after processing the first $(k-1)$ rows. Let the kth row be

$$a_k^T = (a_{k1}, a_{k2}, \ldots, a_{kn}).$$

This is processed as follows: we uncompress this row into a full vector and scan the nonzeros from left to right. For each $a_{kj} \neq 0$ a Givens rotation involving row j in R_{k-1} is used to annihilate a_{kj}. This may create new nonzeros both in R_{k-1} and in the row a_k^T. We continue until the whole row a_k^T has been annihilated. Note that if $r_{jj} = 0$, this means that this row in R_{k-1} has not yet been touched by any rotation and hence the entire jth row must be zero. When this occurs the remaining part of row k is inserted as the jth row in R.

To illustrate this algorithm we use an example taken from George and Ng [242]. Assume that the first k rows of A have been processed to generate $R^{(k)}$. In Figure 8.5.3 nonzero elements of $R^{(k-1)}$ are denoted by $\times$, nonzero elements introduced into $R^{(k)}$ and a_k^T during the elimination of a_k^T are denoted by $+$; all the elements involved in the elimination of a_k^T are circled. Nonzero elements created in a_k^T during the elimination are of course ultimately annihilated. The sequence of row indices involved in the elimination are $\{2, 4, 5, 7, 8\}$, where 2 is the column index of the first nonzero in a_k^T.

$$\begin{bmatrix} R_{k-1} \\ a_k^T \end{bmatrix} = \begin{bmatrix} \times & 0 & \times & 0 & 0 & \times & 0 & 0 & 0 & 0 \\ & \otimes & 0 & \oplus & \otimes & 0 & 0 & 0 & 0 & 0 \\ & & \times & 0 & \times & 0 & 0 & 0 & \times & 0 \\ & & & \otimes & \oplus & 0 & \otimes & 0 & 0 & 0 \\ & & & & \otimes & \oplus & 0 & 0 & 0 & 0 \\ & & & & & \times & 0 & 0 & \times & 0 \\ & & & & & & \otimes & \otimes & 0 & 0 \\ & & & & & & & \otimes & 0 & 0 \\ & & & & & & & & \times & \times \\ & & & & & & & & 0 & \times \\ 0 & \otimes & 0 & \otimes & \oplus & 0 & \oplus & \oplus & 0 & 0 \end{bmatrix}$$

Figure 8.5.4. *Circled elements $\otimes$ in R_{k-1} are involved in the elimination of a_k^T; fill elements are denoted by $\oplus$.*

Note that unlike the Householder method intermediate fill now only takes place in the row that is being processed. It can be shown that if the structure of R has been predicted from that of A^TA, then any intermediate matrix R_{i-1} will fit into the predicted structure.

For simplicity we have not included the right-hand side in Figure 8.5.3, but the Givens rotations should be applied simultaneously to b to form Q^Tb. In the implementation by George and Heath [236] the Givens rotations are not stored but discarded after use. Hence, only enough storage to hold the final R and a few extra vectors for the current row and right-hand side(s) is needed in main memory.

The orthogonal factor Q is often much less sparse than R. For example, in a grid problem with n unknowns it is known that the number of nonzero elements in R is of the order $O(n \log n)$, whereas the number of nonzeros in Q is $O(n^{3/2}$; see Gilbert, Ng and Peyton [248].

In general, one should if possible avoid computing Q explicitly. If Q is required, then the Givens rotations could be saved separately. This in general requires far less storage and fewer operations than computing and storing Q itself. However, discarding Q creates a problem if we later wish to solve additional problems having the same matrix A but a different right-hand side b, since we cannot form Q^Tb. One could recompute the factorization, but in most cases a satisfactory method to deal with this problem is to use the **corrected seminormal equations**; see [61, Sec. 6.6.5].

Example 8.5.2 (*T. Elfving and I. Skoglund* **[191]).**

To illustrate the effect of different column orderings we consider a model used for substance transport in rivers. In this time series data L_{ij}, $i = 1 : n$, $j = 1 : m$, are observed. Here n is the length of the study period expressed in years and m is the number of samples from each year. The unknown parameters are split in two sets $x^T = (x_1,\ x_2)$ and a regularization matrix is added. The following figures

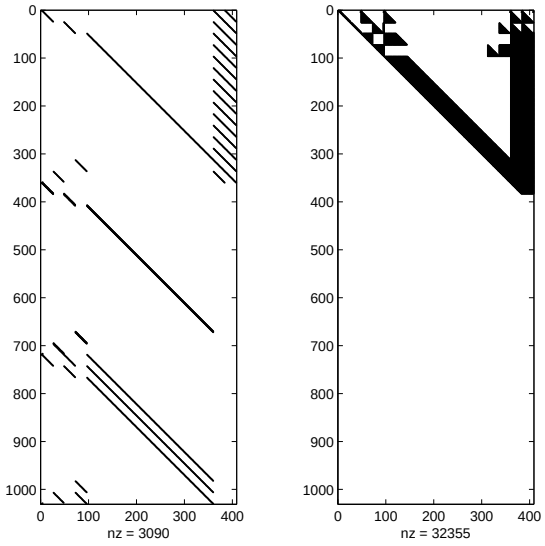

Figure 8.5.5. *Nonzero pattern of A and its R factor after reordering using* MATLAB*'s colperm.*

shows the location of nonzero elements in the matrix AP and its R-factor after using two different column orderings available in MATLAB. The first (`colperm`) is a reordering according to increasing count of nonzero entries in columns. The second (`colamd` in MATLAB) is an approximate minimum degree ordering on A^TA; see Section 7.7.6.

8.5.6 Multifrontal QR decomposition.

A significant advance in direct methods for sparse matrix factorization is the **multifrontal method** by Duff and Reid [179, 1983]. This method reorganizes the factorization of a sparse matrix into a sequence of partial factorizations of small dense matrices, and is well suited for parallelism. A multifrontal algorithm for the QR factorization was first developed by Liu [412, 1986]. Liu generalized the row-oriented scheme of George and Heath by using submatrix rotations and remarked that this scheme is essentially equivalent to a multifrontal method. He showed that his algorithm can give a significant reduction in QR decomposition time at a modest increase in working storage. George and Liu [240, 1987] presented a modified version of Liu's algorithm which uses Householder transformations instead of Givens rotations.

There are several advantages with the multifrontal approach. The solution of the dense subproblems can more efficiently be handled by vector machines. Also, it leads to independent subproblems which can be solved in parallel. The good data locality of the multifrontal method gives fewer page faults on paging systems,

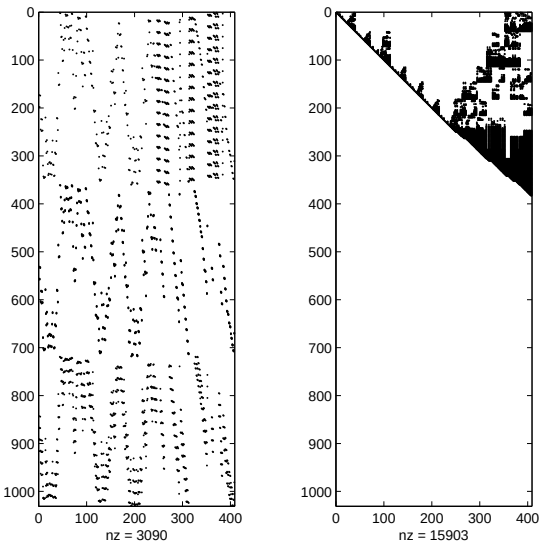

Figure 8.5.6. *Nonzero pattern of A and its R factor after reordering using* MATLAB*'s colamd.*

and out-of-core versions can be developed. Multifrontal methods for sparse QR decompositions have been extensively studied

$$A = \begin{pmatrix} \times & & & & \times & & \times & \times & \\ \times & & & & \times & & \times & \times & \\ \times & & & & \times & & \times & \times & \\ & \times & & & \times & & & \times & \times \\ & \times & & & \times & & & \times & \times \\ & \times & & & \times & & & \times & \times \\ & & \times & & & \times & \times & \times & \\ & & \times & & & \times & \times & \times & \\ & & \times & & & \times & \times & \times & \\ & & & \times & & \times & & \times & \times \\ & & & \times & & \times & & \times & \times \\ & & & \times & & \times & & \times & \times \end{pmatrix}.$$

Figure 8.5.7. *A matrix A corresponding to a 3×3 mesh.*

We first describe the multiple front idea on the small 12×9 example in Figure 6.6.3, adopted from Liu [412, 1986]. This matrix arises from a 3×3 mesh problem using a nested dissection ordering, and its graph $G(A^TA)$ is given in Figure 6.6.4.

We first perform a QR decomposition of rows 1–3. Since these rows have nonzeros only in columns $\{1, 5, 7, 8\}$ this operation can be carried out as a QR decomposition of a small dense matrix of size 3×4 by leaving out the zero columns. The first row equals the first of the final R of the complete matrix and can be stored

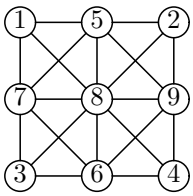

Figure 8.5.8. *The graph $G(A^TA)$ and a nested dissection ordering.*

away. The remaining two rows form an **update matrix** F_1 and will be processed later. The other three block rows 4–6, 7–9, and 10–12 can be reduced in a similar way. Moreover, these tasks are independent and can be done in parallel. After this first stage the matrix has the form shown in Figure 6.6.5. The first row in each of the four blocks are final rows in R and can be removed, which leaves four upper trapezoidal update matrices, F_1–F_4.

$$\begin{pmatrix}
\times & & & & \times & & \times & \times & \\
 & & & & \times & & \times & \times & \\
 & & & & & & \times & \times & \\
 & \times & & & \times & & & \times & \times \\
 & & & & \times & & & \times & \times \\
 & & & & & & & \times & \times \\
 & & \times & & & \times & \times & \times & \\
 & & & & & \times & \times & \times & \\
 & & & & & & \times & \times & \\
 & & & \times & & \times & & \times & \times \\
 & & & & & \times & & \times & \times \\
 & & & & & & & \times & \times
\end{pmatrix}.$$

Figure 8.5.9. *The reduced matrix after the first elimination stage.*

In the second stage we can simultaneously merge F_1, F_2 and F_3, F_4 into two upper trapezoidal matrices by eliminating columns 5 and 6. In merging F_1 and F_2 we need to consider only the set of columns $\{5, 7, 8, 9\}$. We first reorder the rows after the index of the first nonzero element, and then perform a QR decomposition:

$$Q^T \begin{pmatrix} \times & \times & \times & \\ \times & & \times & \times \\ & \times & \times & \\ & & \times & \times \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \end{pmatrix}.$$

The merging of F_3 and F_4 is performed similarly. Again, the first row in each reduced matrix is a final row in R, and is removed. In the final stage we merge the remaining two upper trapezoidal (in this example triangular) matrices and produce the final factor R. This corresponds to eliminating columns $7, 8$,and 9.

The scheme described here can also be viewed as a special type of variable row pivoting method. This pivoting scheme has never become very popular because

of the difficulty of generating good orderings for the rotations and because these schemes are complicated to implement. Also, the dynamic storage structure needed tends to reduce the efficiency.

The organization of the multifrontal method is based on the elimination tree, and nodes in the tree are visited in turn given by the ordering. Each node x_j in the tree is associated with a frontal matrix F_j, which consists of the set of rows A_j in A with the first nonzero in location j, together with one update matrix contributed by each child node of x_j. After eliminating the variable j in the frontal matrix, the first row in the reduced matrix is the jth row of the upper triangular factor R. The remaining rows form a new update matrix U_j, and is stored in a stack until needed. Hence, a formal description of the method is as follows.

Algorithm 8.9.
Multifrontal Sparse QR Algorithm.

For $j := 1$ to n do

1. Form the frontal matrix F_j by combining the set of rows A_j and the update matrix U_s for each child x_s of the node x_j in the elimination tree $T(A^TA)$;

2. By an orthogonal transformation, eliminate variable x_j in F_j to get $\bar{U}_j$. Remove the first row in $\bar{U}_j$, which is the jth row in the final matrix R. The rest of the matrix is the update matrix U_j;

end.

The node ordering of an elimination tree is such that children nodes are numbered before their parent node. Such orderings are called **topological orderings**. All topological orderings of the elimination tree are equivalent in the sense that they give the same triangular factor R. A **postordering** is a topological ordering in which a parent node j always has node $j-1$ as one of its children. Postorderings are particularly suitable for the multifrontal method, and can be determined by a depth-first search; see Liu [413, 1990]. For example, the ordering of the nodes in the tree in Figure 6.6.6 can be made into a postordering by exchanging labels 3 and 5. The important advantage of using a postordering in the multifrontal method is that data management is simplified since the update matrices can be managed in a stack on a last-in–first-out basis. This also reduces the storage requirement.

The frontal matrices in the multifrontal method are often too small to make it possible to efficiently utilize vector processors and matrix-vector operations in the solution of the subproblems. A useful modification of the multifrontal method, therefore, is to amalgamate several nodes into one supernode. Instead of eliminating one column in each node, the decomposition of the frontal matrices now involves the elimination of several columns, and it may be possible to use Level 2 or even Level 3 BLAS.

In general, nodes can be grouped together to form a supernode if they correspond to a block of contiguous columns in the Cholesky factor, where the diagonal block is full triangular and these rows all have identical off-block diagonal column

structures. Because of the computational advantages of having large supernodes, it is advantageous to relax this condition and also amalgamate nodes which satisfy this condition if some local zeros are treated as nonzeros. A practical restriction is that if too many nodes are amalgamated then the frontal matrices become sparse. (In the extreme case when all nodes are amalgamated into one supernode, the frontal matrix becomes equal to the original sparse matrix!) Note also that non-numerical operations often make up a large part of the total decomposition time, which limits the possible gain.

In many implementations of the multifrontal algorithms the orthogonal transformations are not stored, and the seminormal equations are used for treating additional right-hand sides. If Q is needed then it should not be stored explicitly, but instead be represented by the Householder vectors of the frontal orthogonal transformations. For a K by K grid problem with $n = K^2$, $m = s(K-1)^2$ it is known that $\operatorname{nnz}(R) = O(n \log n)$ but Q has $O(n\sqrt{n})$ nonzeros. Lu and Barlow [418] show that Storing the frontal Householder matrices only requires $O(n \log n)$ storage.

8.5.7 Kronecker Product Problems

Sometimes least squares problems occur which have a highly regular block structure. Here we consider least squares problems of the form

$$\min_x \|(A \otimes B)x - c\|_2, \quad c = \operatorname{vec} C, \tag{8.5.25}$$

where the $A \otimes B$ is the **Kronecker product** of $A \in \mathbf{R}^{m\times n}$ and $B \in \mathbf{R}^{p\times q}$. This product is the $mp \times nq$ block matrix,

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}.$$

Problems of Kronecker structure arise in several application areas including signal and image processing, photogrammetry, and multidimensional approximation. It applies to least squares fitting of multivariate data on a rectangular grid. Such problems can be solved with great savings in storage and operations. Since often the size of the matrices A and B is large, resulting in models involving several hundred thousand equations and unknowns, such savings may be essential.

We recall from Section 7.7.3 the important rule (7.7.14) for the inverse of a Kronecker product

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}.$$

In particular, if P and Q are orthogonal $n \times n$ matrices then

$$(P \otimes Q)^{-1} = P^{-1} \otimes Q^{-1} = P^T \otimes Q^T = (P \otimes Q)^T,$$

where the last equality follows from the definition. Hence, $P \otimes Q$ is an orthogonal $n^2 \times n^2$ matrix. The rule for the inverse holds also for pseudo-inverses.

Lemma 8.5.3.
Let $A^\dagger$ and $B^\dagger$ be the pseudo-inverses of A and B. Then

$$(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger.$$

Proof. The theorem follows by verifying that $X = A^\dagger \otimes B^\dagger$ satisfies the four Penrose conditions in (8.1.37)–(8.1.38). □

Using Lemmas 7.7.6 and 8.5.3 the solution to the Kronecker least squares problem (8.5.25) can be written

$$x = (A \otimes B)^\dagger c = (A^\dagger \otimes B^\dagger)\mathrm{vec}\, C = \mathrm{vec}\,(B^\dagger C\,(A^\dagger)^T). \tag{8.5.26}$$

This formula leads to a great reduction in the cost of solving Kronecker least squares problems. For example, if A and B are both $m \times n$ matrices, the cost of computing is reduced from $O(m^2n^4)$ to $O(mn^2)$.

In some areas the most common approach to computing the least squares solution to (8.5.25) is to use the normal equations. If we assume that both A and B have full column rank, then we can use the expressions

$$A^\dagger = (A^TA)^{-1}A^T, \qquad B^\dagger = (B^TB)^{-1}B^T.$$

However, because of the instability associated with the explicit formation of A^TA and B^TB, an approach based on orthogonal factorizations should generally be preferred. If we have computed the complete QR factorizations of A and B,

$$A\Pi_1 = Q_1 \begin{pmatrix} R_1 & 0 \\ 0 & 0 \end{pmatrix} V_1^T, \qquad B\Pi_2 = Q_2 \begin{pmatrix} R_2 & 0 \\ 0 & 0 \end{pmatrix} V_2^T,$$

with R_1, R_2 upper triangular and nonsingular, then from Section 2.7.3 we have

$$A^\dagger = \Pi_1 V_1 \begin{pmatrix} R_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q_1^T, \qquad B^\dagger = \Pi_2 V_2 \begin{pmatrix} R_2^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q_2^T.$$

These expressions can be used in (8.5.26) to compute the pseudo-inverse solution of problem (8.5.25) even in the rank deficient case.

The SVD of a Kronecker product $A \otimes B$ can be simply expressed in terms of the SVD of A and B. If A and B have the singular value decompositions

$$A = U_1\Sigma_1V_1^T, \qquad B = U_2\Sigma_2V_2^T.$$

then using Lemma 8.5.3 it follows that

$$A \otimes B = (U_1 \otimes U_2)(\Sigma_1 \otimes \Sigma_2)(V_1 \otimes V_2)^T. \tag{8.5.27}$$

When using this to solve the least squares problem (8.5.25) we can work directly with the two small SVDs of A and B. The solution $x = (A \otimes B)^\dagger c$ can be written as $x = \mathrm{vec}\,(X)$ where

$$X = B^\dagger C\,(A^\dagger)^T = V_2\Sigma_2^\dagger(U_2^T\,C\,U_1)(\Sigma_1^T)^\dagger V_1^T. \tag{8.5.28}$$

8.5.8 Tensor Computations

A **tensor** is a multidimensional array. For example, a third-order tensor A has elements with three indices a_{ijk}. When one or more indices are held constant in a tensor, subarrays are formed. In a third-order tensor fixing one index gives a matrix and fixing two indices gives a vector. Thus, a third-order tensor can be thought of built up of slices of matrices in three different ways, depending on what index is fixed.

Tensor decompoasitions has been used for some time in psychometrics and chemometrics. Lately it has been taken up in other fields such as signal processing and linear algebra. Both with respect to notations and theory this field is still in its infancy.

In matrix computations an important role is played by different matrix decompositions. An important example is the SVD

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

of as matrix $A \in \mathbf{R}^{m\times n}$. This expresses a matrix as the weighted sum of rank one matrices $u_i v_i^T$, with $\|u_i|_2 = \|v_i|_2 = 1$. Truncating the SVD expansion gives the best approximation to A by a matrix of rank $r < n$; see Eckart–Young Theorem 8.1.18.

A natural question is if the SVD can somehow be generalized to tensors. One approach is to seek an expansion of the form

$$A = \sum_{i=1}^{n} \sigma_i T_i,$$

where T_i are tensors of rank one. A rank one matrix $X = (x_{ij})$ can always be written as an outer product of two vectors ab^T, and $x_{ij} = \alpha_i\beta_j$. Similarly, a three-dimensional tensor of rank one has the form

$$T = (t_{ijk}), \quad t_{ijk} = \alpha_i\beta_j\gamma_k.$$

The Frobenius norm of a three-dimensional tensor is defined as

$$\|A\|_F^2 = \sum_{i,j,k} a_{ijk}^2.$$

The tensor A is said to have rank r if there is an expansion $A = \sum_{i=1}^{r} \sigma_i T_i$, where T_i are rank one tensors. In many applications we would like to approximate a tensor A with a tensor of lower rank, i.e., to find rank one tensors T_i so that

$$\|A - \sum_{i=1}^{r} \sigma_i T_i\|_F$$

is minimized. This is the so called **parafac** (parallel factorization) decomposition of A.

Tensor decompositions: see [381, 380, 396, 397, 189].

Review Questions

5.1 What is meant by the standard form of a banded rectangular matrix A? Why is it important that a banded matrix is permuted into standard form before its orthogonal factorization is computed?

5.2 In least squares linear regression the first column of A often equals the vector $a_1 = e = (1, 1, \ldots, 1)^T$ (cf. Example 8.2.1). Setting $A = (\, e \quad A_2 \,)$, show that performing one step in MGS is equivalent to "subtracting out the means".

Problems

5.1 Consider the two-block least squares problem (8.5.5). Work out an algorithm to solve the reduced least squares problem $\min_{x_2} \|P_{\mathcal{N}(A_1^T)}(A_2 x_2 - b)\|_2$ using the method of normal equations.

Hint: First show that $P_{\mathcal{N}(A_1^T)}(A) = I - A_1(R_1^T R_1)^{-1} A_1^T$, where R_1 is the Cholesky factor of $A_1^T A_1$.

5.2 (a) Write a MATLAB program for fitting a straight line $c_1 x + c_2 y = d$ to given points $(x_i, y_i) \in \mathbf{R}^2$, $i = 1 : m$. The program should handle all exceptional cases, e.g., $c_1 = 0$ and/or $c_2 = 0$.

(b) Suppose we want to fit two set of points $(x_i, y_i) \in \mathbf{R}^2$, $i = 1, \ldots, p$, and $i = p+1, \ldots, m$, to two *parallel* lines

$$cx + sy = h_1, \qquad cx + sy = h_2, \quad c^2 + s^2 = 1,$$

so that the sum of orthogonal distances are minimized. Generalize the approach in (a) to write an algorithm for solving this problem.

(c) Modify the algorithm in (a) to fit two *orthogonal* lines.

5.3 Use the Penrose conditions to prove the formula

$$(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger,$$

where $\otimes$ denotes the Kronecker product

5.4 Test the recursive QR algorithm `recqr(A)` given in Section sec8.2.rec on some matrices. Check that you obtain the same result as from the built-in function `qr(A)`.

8.6 Some Generalized Least Squares Problems

8.6.1 Least Squares for General Linear Models

We consider a general univariate linear model where the error covariance matrix equals $\sigma^2 V$, where V is a positive definite symmetric matrix. Then the best unbiased linear estimate equals the minimizer of

$$(Ax - b)^T V^{-1} (Ax - b) \tag{8.6.1}$$

For the case of a general positive definite covariance matrix we assume that the Cholesky factorization $V = R^T R$ of the covariance matrix can be computed. Then the least squares problem (8.6.1) becomes

$$(Ax - b)^T V^{-1}(Ax - b) = \|R^{-T}(Ax - b)\|_2^2. \tag{8.6.2}$$

This could be solved in a straightforward manner by forming $\widetilde{A} = R^{-T}A$, $\widetilde{b} = R^{-T}b$ and solving the standard least squares problem

$$\min_x \|\widetilde{A}x - \widetilde{b}\|_2.$$

A simple special case of (8.6.1) is when the covariance matrix V is a positive diagonal matrix,

$$V = \sigma^2 \mathrm{diag}\,(v_1, v_2, \ldots, v_m) > 0.$$

This leads to the **weighted** linear least squares problem (8.1.17)

$$\min_x \|D(Ax - b)\|_2 = \min_x \|(DA)x - Db|_2 \tag{8.6.3}$$

where $D = \mathrm{diag}\,(v_{ii}^{-1/2})$. Note that the weights will be large when the corresponding error component in the linear model has a small variance. We assume in the following that the matrix A is row equilibrated, that is,

$$\max_{1 \le j \le n} |a_{ij}| = 1, \quad i = 1 : m.$$

and that the weights $D = \mathrm{diag}\,(d_1, d_2, \ldots, d_n)$ have been ordered so that

$$\infty > d_1 \ge d_2 \ge \cdots \ge d_m > 0. \tag{8.6.4}$$

Note that only the *relative size* of the weights influences the solution.

If the ratio $\gamma = d_1/d \gg 1$ we call the weighted problems **stiff**. Stiff problems arise, e.g., in electrical networks, certain classes of finite element problems, and in interior point methods for constrained optimization. Special care is needed in solving stiff weighted linear least squares problems. The method of normal equations is not well suited for solving such problems. To illustrate this, we consider the special case where only the first $p < n$ equations are weighted,

$$\min_x \left\| \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} x - \begin{pmatrix} \gamma b_1 \\ b_2 \end{pmatrix} \right\|_2^2, \tag{8.6.5}$$

$A_1 \in \mathbf{R}^{p \times n}$ and $A_2 \in \mathbf{R}^{(m-p) \times n}$. Such problems occur, for example, when the method of weighting is used to solve a least squares problem with the linear equality constraints $A_1 x = b_1$; see Section 5.1.4. For this problem the matrix of normal equations becomes

$$B = (\, \gamma A_1^T \quad A_2^T \,) \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} = \gamma^2 A_1^T A_1 + A_2^T A_2.$$

If $\gamma > u^{-1/2}$ (u is the unit roundoff) and $A_1^T A_1$ is dense, then $B = A^T A$ will be completely dominated by the first term and the data contained in A_2 may be lost. If the number p of very accurate observations is less than n, then this is a disaster, since the solution depends critically on the less precise data in A_2. (The matrix in Example 2.2.1 is of this type.)

Clearly, if the problem is stiff the condition number $\kappa(DA)$ will be large. An upper bound of the condition number is given by

$$\kappa(DA) \le \kappa(D)\kappa(A) = \gamma\kappa(A).$$

It is important to note that this does not *mean that the problem of computing x from given data $\{D, A, b\}$ is ill-conditioned when $\gamma \gg 1$.*

We now examine the use of methods based on the QR factorization of A for solving weighted problems. We first note that it is essential that *column pivoting* is performed when QR factorization is used for weighted problems. To illustrate the need for column pivoting, consider an example of the form (8.6.5), where

$$A_1 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \end{pmatrix},$$

Then stability is lost without column pivoting because the first two columns of the matrix A_1 are linearly dependent. When column pivoting is introduced this difficulty disappears.

The following example shows that the Householder QR method can give poor accuracy for weighted problems even if column pivoting is used.

Example 8.6.1 (*Powell and Reid [492]***).**

Consider the least squares problem with

$$DA = \begin{pmatrix} 0 & 2 & 1 \\ \gamma & \gamma & 0 \\ \gamma & 0 & \gamma \\ 0 & 1 & 1 \end{pmatrix}, \qquad Db = \begin{pmatrix} 3 \\ 2\gamma \\ 2\gamma \\ 2 \end{pmatrix}.$$

The exact solution is equal to $x = (1, 1, 1)$. Using exact arithmetic we obtain after the first step of QR factorization of A by Householder transformations the reduced matrix

$$\widetilde{A}^{(2)} = \begin{pmatrix} \frac{1}{2}\gamma - 2^{1/2} & -\frac{1}{2}\gamma - 2^{-1/2} \\ -\frac{1}{2}\gamma - 2^{1/2} & \frac{1}{2}\gamma - 2^{-1/2} \\ 1 & 1 \end{pmatrix}.$$

If $\gamma > u^{-1}$ the terms $-2^{1/2}$ and $-2^{-1/2}$ in the first and second rows are lost. However, this is equivalent to the loss of all information present in the first row of A. This loss is disastrous because the number of rows containing large elements is less than the number of components in x, so there is a substantial dependence of the solution x on the first row of A. (However, compared to the method of normal equations, which fails already when $\gamma > u^{-1/2}$, this is an improvement!)

As shown by Cox and Higham [120], provided that an initial **row sorting** is performed the Householder QR method with column pivoting has very good stability properties for weighted problems. The rows of $\widetilde{A} = DA$ and $\widetilde{b} = Db$ should be sorted after decreasing ∞-norm,

$$\max_j |\widetilde{a}_{1j}| \geq \max_j |\widetilde{a}_{2j}| \geq \cdots \geq \max_j |\widetilde{a}_{mj}|. \tag{8.6.6}$$

(In Example 8.6.1 this will permute the two large rows to the top.) Row pivoting could also be used, but row sorting has the advantage that after sorting the rows, any library routine for QR with column pivoting can be used.

If the Cholesky factor R is ill-conditioned there could be a loss of accuracy in forming $\widetilde{A}$ and $\widetilde{b}$. But assume that column pivoting has been used in the Cholesky factorization and set $R = D\hat{R}$, where $\hat{R}$ is *unit upper triangular* and D diagonal. Then any ill-conditioning is usually reflected in D. If necessary a presorting of the rows of $\widetilde{A} = D^{-1}\hat{R}^{-T}A$ can be performed before applying the Householder QR method to the transformed problem.

In **Paige's method** [463] the general linear model with covariance matrix $V = BB^T$ is reformulated as

$$\min_{v,x} \|v\|_2 \quad \text{subject to} \quad Ax + Bv = b. \tag{8.6.7}$$

This formulation has the advantage that it is less sensitive to an ill-conditioned V and allows for rank deficiency in both A and V. For simplicity we consider here only the case when V is positive definite and $A \in \mathbf{R}^{m\times n}$ has full column rank n.

The first step is to compute the QR factorization

$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} n \\ m-n \end{matrix}, \quad Q^T b = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}, \tag{8.6.8}$$

where R is nonsingular. The same orthogonal transformation is applied also to B, giving

$$Q^T B = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} n \\ m-n \end{matrix},$$

where by the nonsingularity of B it follows that rank $(C_2) = m-n$. The constraints in (8.6.7) can now be written in the partitioned form

$$\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} v + \begin{pmatrix} R \\ 0 \end{pmatrix} x = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix},$$

or

$$Rx = c_1 - C_1 v, \qquad C_2 v = c_2. \tag{8.6.9}$$

where $C_2 = \mathbf{R}^{(m-n)\times m}$. For any vector v we can determine x so that the first block of these equations is satisfied.

An orthogonal matrix $P \in R^{m\times m}$ can then be determined such that

$$C_2 P = (\,0 \quad S\,), \tag{8.6.10}$$

where S is upper triangular and nonsingular. Setting $u = P^T v$ the second block of the constraints in (8.6.9) becomes

$$C_2 P(P^T v) = (\,0 \quad S\,)\, u = c_2.$$

Since P is orthogonal $\|v\|_2 = \|u\|_2$ and so the minimum in (8.6.7) is found by taking

$$v = P \begin{pmatrix} 0 \\ u_2 \end{pmatrix}, \quad u_2 = S^{-1} c_2. \tag{8.6.11}$$

Then x is obtained from the triangular system $Rx = c_1 - C_1 v$. It can be shown that the computed solution is an unbiased estimate of x for the model (8.6.7) with covariance matrix $\sigma^2 C$, where

$$C = R^{-1} L^T L R^{-T}, \quad L^T = C_1^T P_1. \tag{8.6.12}$$

Paige's algorithm (8.6.8)–(8.6.11) as described above does not take advantage of any special structure the matrix B may have. It requires a total of about $4m^3/3 + 2m^2 n$ flops. If $m \gg n$ the work in the QR factorization of C_2 dominates. It is not suitable for problems where B is sparse, e.g., diagonal as in the case of a weighted least squares problem.

Several generalized least squares problems can be solved by using a generalized SVD (GSVD) or QR (GQR) factorization involving a pair of matrices A, B. One motivation for using this approach is to avoid the explicit computation of 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.

The factorization used in Paige's method is a special case of the generalized QR (GQR) factorization. Let $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times p}$ be a pair of matrices with the same number of rows. The GQR factorization of A and B is a factorization of the form

$$A = QR, \qquad B = QTZ, \tag{8.6.13}$$

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

$$R = \begin{pmatrix} R_{11} \\ 0 \end{pmatrix} \quad (m \geq n), \qquad R = (\,R_{11} \quad R_{12}\,) \quad (m < n), \tag{8.6.14}$$

and

$$T = (\,0 \quad T_{12}\,) \quad (m \leq p), \qquad T = \begin{pmatrix} T_{11} \\ T_{21} \end{pmatrix} \quad (m > p). \tag{8.6.15}$$

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}. Routines for computing a GQR factorization of are included in LAPACK. These factorizations allow the solution of very general formulations of several least squares problems.

8.6.2 Indefinite Least Squares

A matrix $Q \in \mathbf{R}^{n\times n}$ is said to be J-orthogonal if

$$Q^T J Q = J, \tag{8.6.16}$$

where J is a **signature matrix**, i.e. a diagonal matrix with elements equal to ± 1. This implies that Q is nonsingular and $QJQ^T = J$. Such matrices are useful in the treatment of problems where there is an underlying indefinite inner product.

In order to construct J-orthogonal matrices we introduce the **exchange operator**. Consider the block 2×2 system

$$Qx = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}. \tag{8.6.17}$$

Solving the first equation for x_1 and substituting in the second equation gives

$$\operatorname{exc}(Q) \begin{pmatrix} y_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ y_2 \end{pmatrix},$$

where where x_1 and y_1 have been exchanged and

$$\operatorname{exc}(Q) = \begin{pmatrix} Q_{11}^{-1} & -Q_{11}^{-1}Q_{12} \\ Q_{21}Q_{11}^{-1} & Q_{22} - Q_{21}Q_{11}^{-1}Q_{12} \end{pmatrix} \tag{8.6.18}$$

Here the $(2,2)$ block is the Schur complement of Q_{11} in Q. From the definition it follows that the exchange operator satisfies

$$\operatorname{exc}(\operatorname{exc}(Q)) = Q,$$

that is it is involutary.

Theorem 8.6.1.

Let $Q \in \mathbf{R}^{n\times n}$ be partitioned as in (8.6.17). If Q is orthogonal and Q_{11} nonsingular then $\operatorname{exc}(Q)$ is J-orthogonal. If Q is J-orthogonal then $\operatorname{exc}(Q)$ is orthogonal.

Proof. A proof due to Chris Paige is given in Higham [330]. □

The indefinite least squares problem (ILS) has the form

$$\min_x (b - Ax)^T J (b - Ax), \tag{8.6.19}$$

where $A \in \mathbf{R}^{m\times n}$, $m \geq n$, and $b \in \mathbf{R}^m$ are given. In the following we assume for simplicity that

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad J = \begin{pmatrix} I_{m_1} & 0 \\ 0 & -I_{m_2} \end{pmatrix}, \tag{8.6.20}$$

where $m_1+m_2 = m$, $m_1 m_2 \neq 0$. If the Hessian matrix $A^T JA$ is positive definite this problem has a unique solution which satisfies the normal equations$A^T JAx = A^T Jb$, or

$$(A_1^T A_1 - A_2^T A_2)x = A_1^T b_1 - A_2^T b_2. \tag{8.6.21}$$

Hence, if we form the normal equations and compute the Cholesky factorization $A^T JA = R^T R$ the solution is given by $x = R^{-1}R^{-T}c$, where $c = (A^T Jb)$. However, by numerical stability considerations we should avoid explicit formation of $A_1^T A_1$ and $A_2^T A_2$. We now show how J-orthogonal transformations can be used to solve this problem more directly.

Consider the orthogonal plane rotation

$$G = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}, \quad c^2 + s^2 = 1.$$

where $c \neq 0$. As a special case of Theorem 8.6.1 it follows that

$$\breve{G} = \operatorname{exc}(G) = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix}, \tag{8.6.22}$$

is J-orthogonal

$$\breve{G}^T J \breve{G} = I, \qquad J = \operatorname{diag}(1, -1).$$

The matrix $\breve{G}$ is called a hyperbolic plane rotation since it can be written as

$$\breve{G} = \begin{pmatrix} \cosh\theta & -\sinh\theta \\ -\sinh\theta & \cosh\theta \end{pmatrix}, \quad \breve{c}^2 - \breve{s}^2 = 1.$$

A hyperbolic rotation $\breve{G}$ can be used to zero a selected component in a vector. Provided that $|\alpha| > |\beta|$ we have

$$\breve{G}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \sigma \\ 0 \end{pmatrix},$$

provided that

$$\sigma = \sqrt{\alpha^2 - \beta^2} = \sqrt{(\alpha+\beta)(\alpha-\beta)}, \qquad c = \sigma/\alpha, \quad s = \beta/\alpha. \tag{8.6.23}$$

Note that the elements of a hyperbolic rotation $\breve{G}$ are unbounded. Therefore, such transformations must be used with care. The direct application of $\breve{G}$ to a vector does not lead to a stable algorithm. Instead, as first suggested by Chambers [95], we note the equivalence of

$$\breve{G}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \quad \Leftrightarrow \quad G\begin{pmatrix} y_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ y_2 \end{pmatrix},$$

where $\breve{G} = \operatorname{exc}(G)$. We first determine y_1 from the hyperbolic rotation and then y_2 from the Given's rotation giving

$$y_1 = (x_1 - sx_2)/c, \qquad y_2 = cx_2 - sy_1. \tag{8.6.24}$$

We now describe an alternative algorithm for solving the indefinite least squares problem (8.6.19), which combines Householder transformations and hyperbolic rotations. In the first step we use two Householder transformations. The first transforms rows $1:m_1$ in A_1 and zeros the elements $2:m_1$. The second transforms rows $1:m_2$ in A_2 and zeros elements $2:m_2$. we now zero out the element left in the first column of A_2 using a hyperbolic rotation in the plane $(1, m_1+1)$. In the next step we proceed similarly. We first zero elements $3:m_1$ in the second column of A_1 and elements $1:m_2$ in the second column of A_2. A hyperbolic rotation in the plane $(2:m_1+1)$ is then used to zero the remaining element in the second column of A_2. After the first two steps we have reduced the matrix A to the form

This method uses n hyperbolic rotations and n Householder transformations for the reduction. Since the cost for the hyperbolic rotations is $= (n^2)$ flops the total cost is about the same as for the usual Householder QR factorization.

Note that this can be combined with column pivoting so that at each step we maximize the diagonal element in R. It suffices to consider the first step; all remaining steps are similar. Changing notations to $A = (a_1, \ldots, a_n) \equiv A_1$, $B = (b_1, \ldots, b_n) \equiv A_2$, we do a modified Golub pivoting. Let p be the smallest index for which

$$s_p \geq s_j, \qquad s_j = \|a_j\|_2^2 - \|b_j\|_2^2, \quad \forall j = 1:n,$$

and interchange columns 1 and p in A and B.

A special case of particular interest is when A_2 consists of a single row. In this case only hyperbolic transformations on A_2 occur.

8.6.3 Linear Equality Constraints

In some least squares problems in which the unknowns are required to satisfy a system of linear equations *exactly*. One source of such problems is in curve and surface fitting, where the curve is required to interpolate certain data points. Such problems can be considered as the limit of a sequence of weighted problems when some weights tend to infinity.

Given matrices $A \in \mathbf{R}^{m\times n}$ and $B \in \mathbf{R}^{p\times n}$ we consider the problem **LSE** to find a vector $x \in \mathbf{R}^n$ which solves

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad Bx = d. \tag{8.6.25}$$

A solution to problem (8.6.25) exists if and only if the linear system $Bx = d$ is consistent. If $\operatorname{rank}(B) = p$ then B has linearly independent rows, and $Bx = d$ is consistent for any right hand side d. A solution to problem (8.6.25) is unique if and only if the null spaces of A and B intersect only trivially, i.e., if $\mathcal{N}(A)\cap\mathcal{N}(B) = \{0\}$, or equivalently

$$\operatorname{rank}\begin{pmatrix} A \\ B \end{pmatrix} = n. \tag{8.6.26}$$

If (8.6.26) is not satisfied then there is a vector $z \neq 0$ such that $Az = Bz = 0$. Hence, if x solves (8.6.25) then $x + z$ is a different solution. In the following we therefore assume that $\operatorname{rank}(B) = p$ and that (8.6.26) is satisfied.

A robust algorithm for problem LSE should check for possible inconsistency of the constraints $Bx = d$. If it is not known a priori that the constraints are consistent, then problem LSE may be reformulated as a **sequential least squares problem**

$$\min_{x \in S} \|Ax - b\|_2, \qquad S = \{x \mid \|Bx - d\|_2 = \min\ \}. \tag{8.6.27}$$

The most efficient way to solve problem LSE is to derive an equivalent unconstrained least squares problem of lower dimension. There are basically two different ways to perform this reduction: direct elimination and the null space method. We describe both these methods below.

In the method of **direct elimination** we start by reducing the matrix B to upper trapezoidal form. It is essential that column pivoting is used in this step. In order to be able to solve also the more general problem (8.6.27) we compute a QR factorization of B such that

$$Q_B^T B \Pi_B = \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix}, \quad R_{11} \in \mathbf{R}^{r \times r}, \tag{8.6.28}$$

where $r = \operatorname{rank}(B) \le p$ and R_{11} is upper triangular and nonsingular. Using this factorization, and setting $\bar{x} = \Pi_B^T x$, the constraints become

$$(R_{11}, R_{12})\bar{x} = R_{11}\bar{x}_1 + R_{12}\bar{x}_2 = \bar{d}_1, \qquad \bar{d} = Q_B^T d = \begin{pmatrix} \bar{d}_1 \\ \bar{d}_2 \end{pmatrix}, \tag{8.6.29}$$

where $\bar{d}_2 = 0$ if and only if the constraints are consistent. If we apply the permutation Π_B also to the columns of A and partition the resulting matrix conformally with (8.6.28), $\bar{A}\Pi_B = (A_1, A_2)$, then $Ax - b = A_1\bar{x}_1 + A_2\bar{x}_2 - b$. Solving (8.6.29) for $\bar{x}_1 = R_{11}^{-1}(\bar{d}_1 - R_{12}\bar{x}_2)$, and substituting, we find that the unconstrained least squares problem

$$\begin{aligned} &\min_{\bar{x}_2} \|\hat{A}_2\bar{x}_2 - \hat{b}\|_2, \qquad \hat{A}_2 \in \mathbf{R}^{m \times (n-r)} \\ &\hat{A}_2 = \bar{A}_2 - \bar{A}_1 R_{11}^{-1} R_{12}, \qquad \hat{b} = b - \bar{A}_1 R_{11}^{-1}\bar{d}_1. \end{aligned} \tag{8.6.30}$$

is equivalent to the original problem LSE. Here $\hat{A}_2$ is the Schur complement of R_{11} in

$$\begin{pmatrix} R_{11} & R_{12} \\ \bar{A}_1 & \bar{A}_2 \end{pmatrix}.$$

It can be shown that if the condition in (8.6.26) is satisfied, then $\operatorname{rank}(A_2) = r$. Hence, the unconstrained problem has a unique solution, which can be computed from the QR factorization of $\hat{A}_2$. The coding of this algorithm can be kept remarkably compact as exemplified by the Algol program of Björck and Golub [63, 1967].

In the null space method we postmultiply B with an orthogonal matrix Q to transform B to lower triangular form. We also apply Q to the matrix A, which gives

$$\begin{pmatrix} B \\ A \end{pmatrix} Q = \begin{pmatrix} B \\ A \end{pmatrix} (\, Q_1 \quad Q_2\,) = \begin{pmatrix} L & 0 \\ AQ_1 & AQ_2 \end{pmatrix}, \quad L \in \mathbf{R}^{p \times p}. \tag{8.6.31}$$

where Q_2 is an orthogonal basis for the null space of B, Note that this is equivalent to computing the QR factorization of B^T. The matrix Q can be constructed as a product of Householder transformations. The solution is now split into the sum of two orthogonal components by setting

$$x = Qy = x_1 + x_2 = Q_1 y_1 + Q_2 y_2, \quad y_1 \in \mathbf{R}^p, \quad y_2 \in \mathbf{R}^{(n-p)}, \tag{8.6.32}$$

where $Bx_2 = BQ_2 y_2 = 0$. From the assumption that rank $(B) = p$ it follows that L is nonsingular and the constraints equivalent to $y_1 = L^{-1}d$ and

$$b - Ax = b - AQy = c - AQ_2 y_2, \quad c = b - (AQ_1)y_1.$$

Hence, y_2 is the solution to the unconstrained least squares problem

$$\min_{y_2} \|(AQ_2)y_2 - c\|_2. \tag{8.6.33}$$

This can be solved, for example, by computing the QR factorization of AQ_2. If (8.6.26) is satisfied then rank $(AQ_2) = n - p$, then the solution to (8.6.33) is unique. If $y_2 = (AQ_2)^{\dagger}(b - AQ_1 y_1)$ is the minimum length solution to (8.6.33), then since

$$\|x\|_2^2 = \|x_1\|_2^2 + \|Q_2 y_2\|_2^2 = \|x_1\|_2^2 + \|y_2\|_2^2$$

$x = Qy$ is the minimum norm solution to problem LSE.

The representation in (8.6.32) of the solution x can be used as a basis for a perturbation theory for problem LSE. A strict analysis is given by Eldén [187, 1982], but the result is too complicated to be given here. If the matrix B is well conditioned, then the sensitivity is governed by $\kappa(AQ_2)$, for which $\kappa(A)$ is an upper bound.

The method of direct elimination and the null space method both have good numerical stability. If Gaussian elimination is used to derive the reduced unconstrained problem the operation count for the method of direct elimination is slightly lower.

8.6.4 Quadratic Constraints and Tikhonov Regularization

Least squares problems with quadratic constraints arise, e.g., when one wants to balance a good fit to the data points and a smooth solution. Such problems arise naturally from inverse problems where one tries to determine the structure of a physical system from its behavior. An example in the form of an integral equation of the first kind was given in Example 7.1.5. Typically, the singular values of the discretized problem will decay exponentially to zero. As shown in Section 8.4.6, any attempt to solve such a problem without restricting x will lead to a meaningless solution of very large norm, or even to failure of the algorithm.

One of the most successful methods for solving ill-conditioned problems is **Tikhonov regularization**[35] (see [564, 1963]). In this method the solution space is

[35] Andrei Nicholaevich Tikhonov (1906–1993), Russian mathematician. He made deep contributions in topology and function analysis, but was also interested in applications to mathematical physics. In the 1960's he introduced the concept of "regularizing operator" for ill-posed problems, for which he was awarded the Lenin medal.

restricted by imposing an a priori bound on $\|Lx\|_2$ for a suitably chosen matrix $L \in \mathbf{R}^{p\times n}$. Typically L is taken to be the identity matrix I or a discrete approximation to some derivative operator, e.g.,

$$L = \begin{pmatrix} 1 & -1 & & \\ & 1 & -1 & \\ & & \ddots & \ddots \\ & & & 1 & -1 \end{pmatrix} \in \mathbf{R}^{(n-1)\times n}, \tag{8.6.34}$$

which, except for a scaling factor, approximates the first derivative operator.

The above approach leads us to take x as the solution to the problem

$$\min_f \|Ax - b\|_2 \quad \text{subject to} \quad \|Lx\|_2 \le \gamma. \tag{8.6.35}$$

Here the parameter γ governs the balance between a small residual and a smooth solution. The determination of a suitable γ is often a major difficulty in the solution process. Alternatively, we could consider the related problem

$$\min_f \|Lx\|_2 \quad \text{subject to} \quad \|Ax - b\|_2 \le \rho. \tag{8.6.36}$$

In the statistical literature the solution of problem (8.6.35) is called a **ridge estimate**. Problems (8.6.35) and (8.6.36) are special cases of the general problem LSQI.

Problem LSQI:

Least Squares with Quadratic Inequality Constraint.

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad \|Lx - d\|_2 \le \gamma, \tag{8.6.37}$$

where $A \in \mathbf{R}^{m\times n}$, $L \in \mathbf{R}^{p\times n}$, $\gamma > 0$.

Conditions for existence and uniqueness and properties of solutions to problem LSQI have been given by Gander [222, 1981]. Clearly, problem LSQI has a solution if and only if

$$\min_x \|Lx - d\|_2 \le \gamma, \tag{8.6.38}$$

and in the following we assume that this condition is satisfied. We define a L-generalized solution $x_{A,L}$ to the problem $\min_x \|Ax - b\|_2$ to be a solution to the problem (cf. Section 2.7.4)

$$\min_{x\in S} \|Lx - d\|_2, \qquad S = \{x \in \mathbf{R}^n \| \ \|Ax - b\|_2 = \min\}. \tag{8.6.39}$$

These observation gives rise to the following theorem.

Theorem 8.6.2.

Assume that problem LSQI has a solution. Then either $x_{A,L}$ is a solution or (8.6.46) *holds and the solution occurs on the boundary of the constraint region. In the latter case the solution $x = x(\lambda)$ satisfies the generalized normal equations*

$$(A^TA + \lambda L^TL)x(\lambda) = A^Tb + \lambda L^Td, \tag{8.6.40}$$

where $\lambda \geq 0$ *satisfies the* **secular equation**

$$\|Lx(\lambda) - d\|_2 = \gamma. \tag{8.6.41}$$

Proof. Since the solution occurs on the boundary we can use the method of Lagrange multipliers and minimize $\psi(x, \lambda)$, where

$$\psi(x, \lambda) = \tfrac{1}{2}\|Ax - b\|_2^2 + \tfrac{1}{2}\lambda(\|Lx - d\|_2^2 - \gamma^2). \tag{8.6.42}$$

A necessary condition for a minimum is that the gradient of $\psi(x, \lambda)$ with respect to x equals zero, which gives (8.6.40). □

As we shall see, only positive values of λ are of interest. Note that (8.6.40) are the normal equations for the least squares problem

$$\min_x \left\| \begin{pmatrix} A \\ \mu L \end{pmatrix} x - \begin{pmatrix} b \\ \mu d \end{pmatrix} \right\|_2, \quad \mu = \sqrt{\lambda}. \tag{8.6.43}$$

Hence, to solve (8.6.40) for a given value of λ, it is not necessary to form the cross-product matrices A^TA and L^TL. Instead we can solve (8.6.43) by QR factorization.

In the following we assume that the constraint $\|Lx(\lambda) - d\|_2 \leq \gamma$ is binding, which is case if (8.6.46) holds. Then there is a unique solution to problem LSQI if and only if the null spaces of A and L intersect only trivially, i.e., $\mathcal{N}(A) \cap \mathcal{N}(L) = \{0\}$, or equivalently

$$\operatorname{rank} \begin{pmatrix} A \\ L \end{pmatrix} = n. \tag{8.6.44}$$

A particularly simple but important case is when $L = I_n$ and $d = 0$, i.e.,

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad \|x\|_2 \leq \gamma, \tag{8.6.45}$$

We call this the **standard form** of LSQI. Notice that for this case we have $x_{A,I} = A^I b$ and the constraint is binding only if

$$\|Lx_{A,L} - d\|_2 > \gamma. \tag{8.6.46}$$

The special structure can be taken advantage of when computing the Householder QR factorization of the matrix in (8.6.43). Below the shape of the transformed matrix after $k = 2$ steps is shown ($m = n = 5$):

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & + & + & + \\ & 0 & + & + & + \\ & & \times & & \\ & & & \times & \\ & & & & \times \end{pmatrix}$$

Notice that the first two rows of D have filled in, but the remaining rows of D are still not touched. For each step $k = 1 : n$ there are m elements in the current column to be annihilated. Therefore, the operation count for the Householder QR factorization will increase with $2n^3/3$ to $2mn^2$ flops. If $A = R$ already is in upper triangular form then the flop count for the reduction is reduced to approximately $2n^3/3$ (cf. Problem 8.1b).

If $L = I$ the singular values of the modified matrix in (8.6.43) are equal to

$$\widetilde{\sigma}_i = (\sigma_i^2 + \lambda)^{1/2}, \quad i = 1 : n.$$

In this case the solution can be expressed in terms of the SVD as

$$x(\lambda) = \sum_{i=1}^{n} f_i \frac{c_i}{\sigma_i} v_i, \quad f_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda}. \tag{8.6.47}$$

The quantities f_i are often called **filter factors**. Notice that as long as $\sqrt{\lambda} \ll \sigma_i$ we have $f_i \approx 1$, and if $\sqrt{\lambda} \gg \sigma_i$ then $f_i \ll 1$. This establishes a relation to the truncated SVD solution (8.4.39) which corresponds to a filter factor which is a step function $f_i = 1$ if $\sigma_i > \delta$ and $f_i = 0$ otherwise. Rules based on the **discrepancy principle** are perhaps the most reliable for chosing the regularization parameter. However, these requires knowledge of the noise level in the data.

Transformation to Standard Form

A Problem LSQI with $L \neq I$ can be transformed to standard form as we now describe. If L is nonsingular we can achieve this by the change of variables $x = L^{-1}y$. This gives the problem

$$\min_y \left\| \begin{pmatrix} AL^{-1} \\ \mu I \end{pmatrix} y - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2. \tag{8.6.48}$$

The matrix $\widetilde{A} = AL^{-1}$ can be formed by solving the matrix equation $\widetilde{A}L = A$.

The general case with no restrictions on L has been treated by Eldén [187]. For simplicity, we assume here that $L \in \mathbf{R}^{(n-t)\times n}$ has full row rank, which is often the case in practice. The transformation to standard form can then be achieved using the pseudo-inverse of L. For example, with L as in (8.6.34) the rank deficiency is $t = 1$. Let the QR factorization of L^T be

$$L^T = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} R_L \\ 0 \end{pmatrix} \in \mathbf{R}^{n\times(n-t)},$$

where R_2 nonsingular and the orthogonal columns of V_2 span the null space of L. We split x into two orthogonal components

$$x = L^{\dagger}y + V_2 w = V_1 R_L^{-T} y + V_2 w, \tag{8.6.49}$$

where $L^{\dagger} = V_1 R_L^{-T}$ is the pseudo-inverse of L. Form $AV = \begin{pmatrix} AV_1 & AV_2 \end{pmatrix}$ and compute

$$AL^{\dagger} = AV_1 R_L^{-T}$$

by solving the upper triangular system $R_L\widetilde{A}^T = (AV_1)^T$. We further need the Householder QR factorization

$$AV_2 = Q\begin{pmatrix} U \\ 0 \end{pmatrix} \in \mathbf{R}^{m\times t}, \quad Q = (\,Q_1 \quad Q_2\,).$$

If A and L have no null space in common, then AV_2 has rank t and U is nonsingular. Combining these results gives

$$\begin{aligned} Q^T(Ax-b) &= Q^T(AV_1R_L^{-T}y + AV_2w - b) \\ &= \begin{pmatrix} Q_1^T(AL^\dagger y - b) + Uw \\ Q_2^T(AL^\dagger y - b) \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}. \end{aligned}$$

Thus, we can always determine w so that $r_1 = 0$. Hence, the problem decouples and y is the solution to

$$\min_y \left\| \begin{pmatrix} \widetilde{A} \\ \mu I \end{pmatrix} y - \begin{pmatrix} \widetilde{b} \\ 0 \end{pmatrix} \right\|_2, \quad \widetilde{A} = Q_2^T A V_1 R_L^{-T}, \quad \widetilde{b} = Q_2^T b, \tag{8.6.50}$$

which is of standard form. We then solve $Uw = Q_1^T(AL^\dagger y - b)$ and retrieve x from (8.6.49).

An important special case is when in LSQI we have $A = K$, $L = L$, and both K and L are upper triangular Toeplitz matrices, i.e.,

$$K = \begin{pmatrix} k_1 & k_2 & \dots & k_{n-1} & k_n \\ & k_1 & k_2 & & k_{n-1} \\ & & \ddots & \ddots & \vdots \\ & & & k_1 & k_2 \\ & & & & k_1 \end{pmatrix}$$

and L is as in (8.6.34). Such systems arise when convolution-type Volterra integral equations of the first kind,

$$\int_0^t K(t-s)f(t)dt = g(s), \qquad 0 \le t \le T,$$

are discretized. Eldén [188] has developed a method for solving problems of this kind which only uses $\frac{9}{2}n^2$ flops for each value of μ. It can be modified to handle the case when K and L also have a few nonzero diagonals below the main diagonal. Although K can be represented by n numbers this method uses $n^2/2$ storage locations. A modification of this algorithm which uses only $O(n)$ storage locations is given in Bojanczyk and Brent [69, 1986].

Solving the Secular Equation.

Methods for solving problem LSQI are usually based on solving the secular equation (8.6.41) using Newton's method. The secular equation can be written in the form

$$f_p(\lambda) = \|x(\lambda)\|^p - \gamma^p = 0. \tag{8.6.51}$$

where $p = \pm 1$ and $x(\lambda)$ be the solution to the least squares problem (8.6.43). From $\|x(\lambda)\|_2^p = (x(\lambda)^T x(\lambda))^{p/2}$, taking derivatives with respect to λ, we find

$$f_p'(\lambda) = p\frac{x^T(\lambda)x'(\lambda)}{\|x(\lambda)\|_2^{2-p}}, \qquad x'(\lambda) = -(A^TA + \lambda I)^{-1}x(\lambda). \tag{8.6.52}$$

Since $x(\lambda) = (A^TA + \lambda I)^{-1}A^Tb$, we obtain

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

The choice $p = +1$ gives rise to the iteration

$$\lambda_{k+1} = \lambda_k + \left(1 - \frac{\gamma}{\|x(\lambda_k)\|_2}\right)\frac{\|x(\lambda_k)\|_2^2}{\|z(\lambda_k)\|_2^2}, \tag{8.6.53}$$

whereas $p = -1$ gives

$$\lambda_{k+1} = \lambda_k - \left(1 - \frac{\|x(\lambda_k)\|_2}{\gamma}\right)\frac{\|x(\lambda_k)\|_2^2}{\|z(\lambda_k)\|_2^2}. \tag{8.6.54}$$

due to to Hebden [314] and Reinsch [501]). The asymptotic rate of convergence for Newton's method is quadratic. For $p = \pm 1$ converge is monotonic, provided that the initial approximation satisfies $0 \le \lambda^{(0)} < \lambda$. Therefore, $\lambda^{(0)} = 0$ is often used as a starting approximation. Close to the solution $\|x(\lambda_k)\| \approx \gamma$, and then the Newton correction is almost the same independent of p. However, for small λ, we can have $\|x(\lambda_k)\| \gg \gamma$ and then $p = -1$ gives much more rapid convergence than $p = 1$.

It has been shown (Reinsch [501]) that $h(\lambda)$ is convex, and hence that the iteration (8.6.54) is monotonically convergent to the solution λ^* if started within $[0, \lambda^*]$. Note that the correction to λ_k in (8.6.54) equals the Newton correction in (8.6.53) multiplied by the factor $\|x(\lambda)\|/\gamma$. Using the QR factorization

$$Q(\lambda)^T\begin{pmatrix} A \\ \sqrt{\lambda}I \end{pmatrix} = \begin{pmatrix} R(\lambda) \\ 0 \end{pmatrix}, \qquad c_1(\lambda) = (\, I_n \quad 0\,)\, Q(\lambda)^T\begin{pmatrix} b \\ 0 \end{pmatrix}, \tag{8.6.55}$$

we have

$$x(\lambda) = R(\lambda)^{-1}c_1(\lambda), \qquad z(\lambda) = R(\lambda)^{-T}x(\lambda). \tag{8.6.56}$$

The main cost in this method is for computing the QR factorization (8.6.55) in each iteration step. On the other hand computing the derivative costs only one triangular solve. Assume that the function qr computes the "thin" QR factorization, with $Q \in \mathbf{R}^{m\times n}$. Then Hebden's algorithm is:

Algorithm 8.10. *Hebden's method.*

The algorithm performs p steps of Hebden's iteration to the constrained least squares problem $\min \|Ax - b\|_2$ subject to $\|x\|_2 = gamma > 0$, using QR factorization starting from a user supplied value λ_0.

$\lambda = \lambda_0$;
for $k = 1 : p \ldots$
  $[Q, R] = \mathrm{qr}\big([A; \sqrt{\lambda}I]\big)$;
  $c = Q^T * b$;
  $x = R^{-1} * c$;
  if $k \le p$ $\%update\lambda$
    $z = R^{-T} * x$;
    $n_x = \|x\|_2$; $n_z = \|z\|_2$;
    $\lambda = \lambda + (n_x/\gamma - 1) * (n_x/n_z)^2$;
  end
end

It is slightly more efficient to initially compute the QR factorizations of A in $mn^2 - n^3/3$ multiplications. Then for each new value of λ the QR factorization of

$$Q^T(\lambda)\begin{pmatrix} R(0) \\ \sqrt{\lambda}I \end{pmatrix} = \begin{pmatrix} R(\lambda) \\ 0 \end{pmatrix}$$

can be computed in just $n^3/3$ multiplications. Then p Newton iterations will require a total of $mn^2 + (p-1)n^3/3$ multiplications. In practice $p \approx 6$ iterations usually suffice to achieve full accuracy. Further, savings are possible by initially transforming A to bidiagonal form; see Eldén [188].

8.6.5 Linear Orthogonal Regression

Let P_i, $i = 1 : m$, be a set of given points in $\mathbf{R}^n$. In the linear **orthogonal regression** problem we want to fit a hyper plane M to the points in such a way that the sum of squares of the orthogonal distances from the given points to M is minimized.

We first consider the special case of fitting a straight line to points in the plane. Let the coordinates of the points be (x_i, y_i) and let the line have the equation

$$c_1 x + c_2 y + d = 0, \tag{8.6.57}$$

where $c_1^2 + c_2^2 = 1$. Then the orthogonal distance from the point $P_i = (x_i, y_i)$ to the line equals $r_i = c_1 x_i + c_2 y_i + d$. Thus, we want to minimize

$$\sum_{i=1}^{m} (c_1 x_i + c_2 y_i + d)^2, \tag{8.6.58}$$

subject to the constraint $c_1^2 + c_2^2 = 1$. This problem can be written in matrix form

$$\min_{c,d} \left\| \begin{pmatrix} e & Y \end{pmatrix} \begin{pmatrix} d \\ c \end{pmatrix} \right\|_2, \quad \text{subject to} \quad c_1 + c_2 = 1,$$

where $c = (c_1 \ c_2)^T$ and

$$\begin{pmatrix} e & Y \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \vdots & \vdots & \\ 1 & x_m & y_m \end{pmatrix}.$$

By computing the QR factorization of the matrix $\begin{pmatrix} e & Y \end{pmatrix}$ and using the invariance of the Euclidean norm this problem is reduced to

$$\min_{d,c} \left\| R \begin{pmatrix} d \\ c \end{pmatrix} \right\|_2, \quad R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{pmatrix}.$$

For any values of c_1 and c_2 we can always determine d so that $r_{11}d + r_{12}c_1 + r_{13}c_2 = 0$. Thus, it remains to determine c so that $\|Bc\|_2$ is minimized, subject to $\|c\|_2 = 1$, where

$$Bc = \begin{pmatrix} r_{22} & r_{23} \\ 0 & r_{33} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}.$$

By the min-max characterization of the singular values (Theorem 8.1.15) the solution equals the right singular vector corresponding to the smallest singular value of the matrix B. Let the SVD be

$$\begin{pmatrix} r_{21} & r_{22} \\ 0 & r_{33} \end{pmatrix} = (u_1 \ u_2) \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \end{pmatrix},$$

where $\sigma_1 \geq \sigma_2 \geq 0$. (A stable algorithm for computing the SVD of an upper triangular matrix is given in Algorithm 9.4.2; see also Problem 9.4.5.) Then the coefficients in the equation of the straight line are given by

$$\begin{pmatrix} c_1 & c_2 \end{pmatrix} = v_2^T.$$

If $\sigma_2 = 0$ but $\sigma_1 > 0$ the matrix B has rank one. In this case the given points lie on a straight line. If $\sigma_1 = \sigma_2 = 0$, then $B = 0$, and all points coincide, i.e. $x_i = \bar{x}$, $y_i = \bar{y}$ for all $i = 1 : m$. Note that v_2 is uniquely determined if and only if $\sigma_1 \neq \sigma_2$. It is left to the reader to discuss the case $\sigma_1 = \sigma_2 \neq 0$!

In [226, Chapter 6] a similar approach is used to solve various other problems, such as fitting two parallel or orthogonal lines or fitting a rectangle or square.

We now consider the general problem of fitting $m > n$ points $P_i \in \mathbf{R}^n$ to a hyper plane M so that the sum of squares of the orthogonal distances is minimized. The equation for the hyper plane can be written

$$c^T z = d, \quad z, c \in \mathbf{R}^n, \quad \|c\|_2 = 1,$$

where $c \in \mathbf{R}^n$ is the normal vector of M, and $|d|$ is the orthogonal distance form the origin to the plane. Then the orthogonal projections of the points y_i onto M are given by

$$z_i = y_i - (c^T y_i - d)c. \tag{8.6.59}$$

It is readily verified that the point z_i lies on M and the residual $(z_i - y_i)$ is parallel to c and hence orthogonal to M. It follows that the problem is equivalent to minimizing

$$\sum_{i=1}^{m} (c^T y_i - d)^2, \quad \text{subject to} \quad \|c\|_2 = 1.$$

If we put $Y^T = (y_1, \ldots, y_m) \in \mathbf{R}^{n\times m}$ and $e = (1, \ldots, 1)^T \in \mathbf{R}^m$, this problem can be written in matrix form

$$\min_{c,d} \left\| (-e \quad Y) \begin{pmatrix} d \\ c \end{pmatrix} \right\|_2, \quad \text{subject to} \quad \|c\|_2 = 1. \tag{8.6.60}$$

For a fixed c, this expression is minimized when the residual vector $(Yc - de)$ is orthogonal to e, that is $e^T(Yc - de) = e^T Yc - de^T e = 0$. Since $e^T e = m$ it follows that

$$d = \frac{1}{m} c^T Y^T e = c^T \bar{y}, \qquad \bar{y} = \frac{1}{m} Y^T e, \tag{8.6.61}$$

where $\bar{y}$ is the mean value of the given points y_i. Hence, d is determined by the condition that the mean value $\bar{y}$ lies on the optimal plane M. Note that this property is shared by the usual linear regression.

We now subtract the mean value $\bar{y}$ from each given point, and form the matrix

$$\overline{Y}^T = (\bar{y}_1, \ldots, \bar{y}_m), \qquad \bar{y}_i = y_i - \bar{y}, \quad i = 1 : m.$$

Since by (8.6.61)

$$(-e \quad Y) \begin{pmatrix} d \\ c \end{pmatrix} = Yc - e\bar{y}^T c = (Y - e\bar{y}^T)c = \bar{Y}c,$$

problem (8.6.60) is equivalent to

$$\min_n \|\bar{Y}c\|_2, \qquad \|c\|_2 = 1 \tag{8.6.62}$$

By the min-max characterization of the singular values a solution to (8.6.62) is given by $c = v_n$, where v_n is a right singular vector of $\bar{Y}$ corresponding to the smallest singular value σ_n. We further have

$$c = v_n, \qquad d = v_n^T \bar{y}, \qquad \sum_{i=1}^{m} (v_n^T y_i - d)^2 = \sigma_n^2,$$

The fitted points $z_i \in M$ are obtained from

$$z_i = \bar{y}_i - (v_n^T \bar{y}_i) v_n + \bar{y},$$

i.e., by first orthogonalizing the shifted points $\bar{y}_i$ against v_n, and then adding the mean value back.

Note that the orthogonal regression problem always has a solution. The solution is unique when $\sigma_{n-1} > \sigma_n$, and the minimum sum of squares equals σ_n^2. Further, $\sigma_n = 0$, if and only if the given points y_i, $i = 1 : m$ all lie on the hyper plane M. In the extreme case, all points coincide and then $\bar{Y} = 0$, and any plane going through $\bar{y}$ is a solution.

The above method solves the problem of fitting a $(n-1)$ dimensional linear manifold to a given set of points in $\mathbf{R}^n$. It is readily generalized to the fitting of an $(n-p)$ dimensional manifold by orthogonalizing the shifted points y against the p right singular vectors of Y corresponding to p smallest singular values. A least squares problem that often arises is to fit to given data points a geometrical element, which may be defined in implicit form. For example, the problem of fitting circles, ellipses, spheres, and cylinders arises in applications such as computer graphics, coordinate meteorology, and statistics. Such problems are nonlinear and will be discussed in Section 9.2.6.

8.6.6 The Orthogonal Procrustes Problem

Let A and B be given matrices in $\mathbf{R}^{m\times n}$. A problem that arises, e.g., in factor analysis in statistics is the **orthogonal Procrustes**[36]**problem**

$$\min_{Q} \|A - BQ\|_F \quad \text{subject to} \quad Q^TQ = I. \tag{8.6.63}$$

Another example is in determining rigid body movements. Suppose that $a_1, a_2, \ldots, a_m$ are measured positions of $m \geq n$ landmarks in a rigid body in $\mathbf{R}^n$. Let $b_1, b_2, \ldots, b_m$ be the measured positions after the body has been rotated. We seek an orthogonal matrix $Q \in \mathbf{R}^{n\times n}$ representing the rotation of the body.

The solution can be computed from the polar decomposition of B^TA as shown by the following generalization of Theorem 8.1.19 by P. Schönemann [522]:

Theorem 8.6.3.

Let $\mathcal{M}_{m\times n}$ denote the set of all matrices in $\mathbf{R}^{m\times n}$ with orthogonal columns. Let A and B be given matrices in $\mathbf{R}^{m\times n}$. If $B^TA = PH$ is the polar decomposition then

$$\|A - BQ\|_F \geq \|A - BP\|_F$$

for any matrix $Q \in \mathcal{M}_{m\times n}$.

Proof. Recall from (7.1.61) that $\|A\|_F^2 = \operatorname{trace}(A^TA)$ and that $\operatorname{trace}(X^TY) = \operatorname{trace}(YX^T)$. Using this and the orthogonality of Q, we find that

$$\|A - BQ\|_F^2 = \operatorname{trace}(A^TA) + \operatorname{trace}(B^TB) - 2\operatorname{trace}(Q^TB^TA).$$

It follows that the problem (8.6.63) is equivalent to maximizing $\operatorname{trace}(Q^TB^TA)$. Let the SVD of B^TA be $B^TA = U\Sigma V^T$, where $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$. Set $Q = UZV^T$,

[36] Procrustes was a giant of Attica in Greece who seized travelers, tied them to a bedstead, and either stretched them or chopped off their legs to make them fit it.

where Z is orthogonal. Then we have $\|Z\|_2 = 1$ and hence the diagonal elements of Z must satisfy $|z_{ii}| \leq 1$, $i = 1 : n$. Hence,

$$\begin{aligned}\operatorname{trace}(Q^T B^T A) &= \operatorname{trace}(V Z^T U^T B^T A) = \operatorname{trace}(Z^T U^T B^T A V) \\ &= \operatorname{trace}(Z^T \Sigma) = \sum_{i=1}^{n} z_{ii}\sigma_i \leq \sum_{i=1}^{n} \sigma_i.\end{aligned}$$

The upper bound is obtained for $Q = UV^T$. This solution is not unique unless $\operatorname{rank}(A) = n$. □

In many applications it is important that Q corresponds to a pure rotation, that is $\det(Q) = 1$. If $\det(UV^T) = 1$, the optimal is $Q = UV^T$ as before. If $\det(UV^T) = -1$, the optimal solution can be shown to be (see [311])

$$Q = UZV^T, \quad Z = \operatorname{diag}(1, \ldots, 1, -1)$$

which has determinant equal to 1. For this choice

$$\sum_{i=1}^{n} z_{ii}\sigma_i = \operatorname{trace}(\Sigma) - 2\sigma_n.$$

Thus, in both cases the optimal solution can be written

$$Q = UZV^T, \quad Z = \operatorname{diag}(1, \ldots, 1, \det(UV^T)).$$

Perturbation bounds for the polar factorization are derived in Barrlund [33]. A perturbation analysis of the orthogonal Procrustes problem is given by Söderlind [537].

In analysis of rigid body movements there is also a translation $c \in Rn$ involved. Then we have the model

$$A = BQ + ec^T, \quad e = (1, 1, \ldots, 1)^T \in \mathbf{R}^m,$$

where we now want to estimate also the translation vector $c \in \mathbf{R}^n$. The problem now is

$$\min_{Q,c} \|A - BQ - ec^T\|_F \quad \text{subject to} Q^T Q = I \tag{8.6.64}$$

and $\det(Q) = 1$. For any Q including the optimal Q we do not yet know, the best least squares estimate of c is characterized by the condition that the residual is orthogonal to e. Multiplying by e^T we obtain

$$0 = e^T(A - BQ - ec^T) = e^T A - (e^T B)Q - mc^T = 0,$$

where $e^T A/m$ and $e^T B/m$ are the mean values of the rows in A and B, respectively. Hence, the optimal translation satisfies

$$c = \frac{1}{m}((B^T e)Q - A^T e). \tag{8.6.65}$$

Substituting this expression into (8.6.64) we can eliminate c and the problem becomes

$$\min_Q \|\widetilde{A} - \widetilde{B}Q\|_F,$$

where

$$\widetilde{A} = A - \frac{1}{m}ee^TA, \qquad \widetilde{B} = B - \frac{1}{m}ee^TB.$$

This is now a standard orthogonal Procrustes problem and the solution is obtained from the SVD of $\widetilde{A}^T\widetilde{B}$.

If the matrix A is close to an orthogonal matrix, then an iterative method for computing the polar decomposition can be used. Such methods are developed in Section 10.8.4.

Review Questions

7.1 What is meant by a saddle-point system? Which two optimization problems give rise to saddle-point systems?

Problems

7.1 Consider the overdetermined linear system $Ax = b$ in Example 8.2.4. Assume that $\epsilon^2 \le u$, where u is the unit roundoff, so that $fl(1+\epsilon^2) = 1$.

(a) Show that the condition number of A is $\kappa = \epsilon^{-1}\sqrt{3+\epsilon^2} \approx \epsilon^{-1}\sqrt{3}$.

(b) Show that if no other rounding errors are made then the maximum deviation from orthogonality of the columns computed by CGS and MGS, respectively, are

$$\text{CGS}: \quad |q_3^Tq_2| = 1/2, \qquad \text{MGS}: \quad |q_3^Tq_1| = \frac{\epsilon}{\sqrt{6}} \le \frac{\kappa}{3\sqrt{3}}u.$$

Note that for CGS orthogonality has been completely lost!

7.2 (Stewart and Stewart [553]).
If properly implemented, hyperbolic Householder transformations have the same good stability as the mixed scheme of hyperbolic rotations.

(a) Show that the hyperbolic rotations $\breve{G}$ can be rewritten as

$$\breve{G} = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + \frac{1}{c}\begin{pmatrix} t \\ -s/t \end{pmatrix}\begin{pmatrix} t & -s/t \end{pmatrix}, \quad t = \sqrt{1-c},$$

which now has the form of a hyperbolic Householder transformation. If $\breve{G}$ is J-orthogonal so is

$$J\breve{G} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \frac{1}{c}\begin{pmatrix} t \\ s/t \end{pmatrix}\begin{pmatrix} t & -s/t \end{pmatrix}, \quad t = \sqrt{1-c},$$

(b) Show that the transformation can be computed form

$$S\breve{G}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \gamma \begin{pmatrix} 1 \\ s/(1-c) \end{pmatrix}, \qquad \gamma = \frac{1}{c}((1-c)x - sy).$$

7.3 Assume that $A \in \mathbf{R}^{m\times m}$ is symmetric and positive definite and $B \in \mathbf{R}^{m\times n}$ a matrix with full column rank. Show that

$$M = \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & -S \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix},$$

where $S = B^T A^{-1} B$ is the Schur complement (cf. (7.1.19)). Conclude that M is indefinite! (M is called a saddle point matrix.)

8.7 The Total Least Squares Problem

8.7.1 Total Least Squares and the SVD

In the standard linear model (8.1.13) it is assumed that the vector $b \in \mathcal{R}^m$ is related to the unknown parameter vector $x \in \mathcal{R}^n$ by a linear relation $Ax = b + e$, where $A \in \mathcal{R}^{m\times n}$ is an exactly known matrix and e a vector of random errors. If the components of e are uncorrelated, have zero means and the same variance, then by the Gauss–Markov theorem (Theorem 8.1.6) the best unbiased estimate of x is obtained by solving the least squares problem

$$\min_x \|r\|_2, \qquad Ax = b + r. \tag{8.7.1}$$

The assumption in the least squares problem that all errors are confined to the right hand side b is frequently unrealistic, and sampling or modeling errors often will affect also the matrix A. In the **errors-in-variables model** it is assumed that a linear relation

$$(A + E)x = b + r,$$

where the rows of the errors $(E,\ r)$ are *independently and identically distributed with zero mean and the same variance*. If this assumption is not satisfied it might be possible to find scaling matrices $D = \operatorname{diag}(d_1, \ldots, d_m)$ and $T = \operatorname{diag}(d_1, \ldots, d_{n+1})$ such that $D(A,\ b)T$ satisfies this assumptions.

Estimates of the unknown parameters x in this model can be obtained from the solution of the **total least squares** (TLS) problem The term "total least squares problem" was coined by Golub and Van Loan in [276]. The concept has been independently developed in statistics where it is known as "latent root regression".

$$\min_{E,\,r} \|(r,\ E)\|_F, \qquad (A + E)x = b + r, \tag{8.7.2}$$

where $\|\cdot\|_F$ denotes the Frobenius matrix norm defined by

$$\|A\|_F^2 = \sum_{i,j} a_{ij}^2 = \operatorname{trace}(A^T A).$$

The constraint in (8.7.2) implies that $b + r \in \mathcal{R}(A + E)$. Thus, the total least squares is equivalent to the problem of finding the "nearest" compatible linear system, where the distance is measured by the Frobenius norm. If a minimizing perturbation $(E,\ r)$ has been found for the problem (8.7.2) then any x satisfying $(A + E)x = b + r$ is said to solve the TLS problem.

The TLS solution will depend on the scaling of the data $(A,\ b)$. In the following we assume that this scaling has been carried out in advance, so that any statistical knowledge of the perturbations has been taken into account. In particular, the TLS solution depends on the relative scaling of A and b. If we scale x and b by a factor γ we obtain the **scaled TLS problem**

$$\min_{E,\,r} \|(E,\,\gamma r)\|_F \qquad (A + E)x = b + r.$$

Clearly, when γ is small perturbations in b will be favored. In the limit when $\gamma \to 0$ we get the ordinary least squares problem. Similarly, when γ is large perturbations in A will be favored. In the limit when $1/\gamma \to 0$, this leads to the **data least squares** (DLS) problem

$$\min_{E} \|E\|_F, \qquad (A + E)x = b, \tag{8.7.3}$$

where it is assumed that the errors in the data is confined to the matrix A.

In the following we assume that $b \notin \mathcal{R}(A)$, for otherwise the system is consistent. The constraint in (8.7.2) can be written

$$(\, b + r \quad A + E \,) \begin{pmatrix} -1 \\ x \end{pmatrix} = 0.$$

This constraint is satisfied if the matrix $(b+r\ A+E)$ is rank deficient and $(\,-1 \quad x\,)^T$ lies in its null space. Hence, the TLS problem involves finding a perturbation matrix having minimal Frobenius norm, which lowers the rank of the matrix $(\, b \quad A\,)$.

The total least squares problem can be analyzed in terms of the SVD

$$(\, b \quad A\,) = U\Sigma V^T = \sum_{i=1}^{k+1} \sigma_i u_i v_i^T, \tag{8.7.4}$$

where $\sigma_1 \geq \ldots \geq \sigma_n \geq \sigma_{n+1} \geq 0$ are the singular values of $(\, b \quad A\,)$. By the minimax characterization of singular values (Theorem 8.1.17) the singular values of $\hat{\sigma}_i$ of A interlace those of $(\, b \quad A\,)$, that is

$$\sigma_1 \geq \hat{\sigma}_1 \geq \sigma_2 > \cdots \geq \sigma_n \geq \hat{\sigma}_n \geq \sigma_{n+1}. \tag{8.7.5}$$

Assume first that $\hat{\sigma}_n > \sigma_{n+1}$. Then it follows that $\operatorname{rank}(A) = n$ and by (8.7.5) $\sigma_n > \sigma_{n+1}$. If $\sigma_{n+1} = 0$, then $Ax = b$ is consistent; otherwise by Theorem 8.1.18 the unique perturbation of minimum norm $\|\,(\, r \quad E\,)\,\|_F$ that makes $(A+E)x = b+r$ consistent is the rank one perturbation

$$(\, r \quad E\,) = -\sigma_{n+1} u_{n+1} v_{n+1}^T \tag{8.7.6}$$

for which $\min_{E,\,r} \| \begin{pmatrix} r & E \end{pmatrix} \|_F = \sigma_{n+1}$. Multiplying (8.7.6) from the right with v_{n+1} gives

$$\begin{pmatrix} b & A \end{pmatrix} v_{n+1} = -\begin{pmatrix} r & E \end{pmatrix} v_{n+1}. \tag{8.7.7}$$

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 (8.7.7) it is easily seen that the TLS solution can be written in terms of the right singular vector v_{n+1} as

$$x = -\frac{1}{\omega} y, \quad v_{n+1} = \begin{pmatrix} \omega \\ y \end{pmatrix}, \tag{8.7.8}$$

If $\omega = 0$ then the TLS problem has no solution. From (8.7.4) it follows that $\begin{pmatrix} b & A \end{pmatrix}^T U = V \Sigma^T$ and taking the $(n+1)$st column of both sides

$$\begin{pmatrix} b^T \\ A^T \end{pmatrix} u_{n+1} = \sigma_{n+1} v_{n+1}. \tag{8.7.9}$$

Hence, if $\sigma_{n+1} > 0$, then $\omega = 0$ if and only if $b \perp u_{n+1}$. (This case can only occur when $\hat{\sigma}_n = \sigma_{n+1}$, since otherwise the TLS problem has a unique solution.) The case when $b \perp u_{n+1}$is called **nongeneric**. It can be treated by adding constraints on the solution; see the discussion [587].

Example 8.7.1.

For

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \qquad E = \begin{pmatrix} 0 & 0 \\ 0 & \epsilon \\ 0 & 0 \end{pmatrix}, \tag{8.7.10}$$

the system $(A+E)x = b$ is consistent for any $\epsilon > 0$. There is no smallest value of ϵ and $\|x\|_2 \to \infty$ when $\epsilon \to 0$ and the TLS problem fails to have a finite solution. Here A is singular, $\hat{\sigma}_2 = \sigma_3 = 0$ and $b \perp u_3 = e_3$.

Suppose now that σ_{n+1} is a repeated singular value,

$$\sigma_p \ > \sigma_{p+1} = \cdots = \sigma_{n+1}, \quad p < n.$$

Set $V_2 = (v_{p+1}, \ldots, v_{n+1})$, and let $V_2 z$ be any unit vector in the subspace spanned by the right singular vectors corresponding to the multiple minimal singular value. Then any vector such that

$$x = -\frac{1}{\omega} y, \quad V_2 z = \begin{pmatrix} \omega \\ y \end{pmatrix},$$

is a TLS solution. A unique TLS solution of minimum norm can be obtained as follows. Since $V_2 z$ has unit length, minimizing $\|x\|_2$ is equivalent to choosing the

unit vector z to maximize $\omega = e_1^T V_2 z$. Let take $z = Qe_1$, where Q is a Householder transformation such that

$$V_2 Q = \begin{pmatrix} \omega & 0 \\ y & V_2' \end{pmatrix}$$

Then a TLS solution of minimum norm is given by (8.7.8). If $\omega \neq 0$ there is no solution and the problem is nongeneric. By an argument similar to the case when $p = n$ this can only happen if $b \perp u_j$, $j = p : n$.

8.7.2 Conditioning of the TLS Problem

We now consider the conditioning of the total least squares problem and its relation to the least squares problem. We denote those solutions by x_{TLS} and x_{LS}, respectively.

In Section 7.1.6 we showed that the SVD of a matrix A is related to the eigenvalue problem for the symmetric matrix $A^T A$. From this it follows that in the generic case the TLS solution can also be characterized by

$$\begin{pmatrix} b^T \\ A^T \end{pmatrix} (b \quad A) \begin{pmatrix} -1 \\ x \end{pmatrix} = \sigma_{n+1}^2 \begin{pmatrix} -1 \\ x \end{pmatrix}, \tag{8.7.11}$$

i.e. $\begin{pmatrix} -1 \\ x \end{pmatrix}$ is an eigenvector corresponding to the smallest eigenvalue $\lambda_{n+1} = \sigma_{n+1}^2$ of the matrix obtained by "squaring"$(b \quad A)$. From the properties of the Rayleigh quotient of symmetric matrices (see Section 9.3.4) it follows that x_{TLS} is characterized by minimizing

$$\rho(x) = \frac{(b - Ax)^T (b - Ax)}{x^T x + 1} = \frac{\|b - Ax\|_2^2}{\|x\|_2^2 + 1}, \tag{8.7.12}$$

Thus, whereas the LS solution minimizes $\|b - Ax\|_2^2$ the TLS solution minimizes the "orthogonal distance" function $\rho(x)$ in (8.7.11).

From the last block row of (8.7.11) it follows that

$$(A^T A - \sigma_{n+1}^2 I) x_{TLS} = A^T b. \tag{8.7.13}$$

Note that if $\hat{\sigma}_n > \sigma_{n+1}$ then the matrix $(A^T A - \sigma_{n+1}^2 I)$ is symmetric positive definite, which ensures that the TLS problem has a unique solution. This can be compared with the corresponding normal equations for the least squares solution

$$A^T A x_{LS} = A^T b. \tag{8.7.14}$$

In (8.7.13) a positive multiple of the unit matrix is *subtracted* from the matrix $A^T A$ of normal equations. Thus, TLS can be considered as a *deregularizing* procedure. (Compare Section 8.1.5, where a multiple of the unit matrix was added to improve the conditioning.) Hence, the TLS solution is always *worse conditioned* than the corresponding LS problem, From a statistical point of view this can be

interpreted as removing the bias by subtracting the error covariance matrix (estimated by $\sigma_{n+1}^2 I$ from the data covariance matrix $A^T A$. Subtracting (8.7.14) from (8.7.14) we get

$$x_{TLS} - x_{LS} = \sigma_{n+1}^2 (A^T A - \sigma_{n+1}^2 I)^{-1} x_{LS}.$$

Taking norms we obtain

$$\frac{\|x_{TLS} - x_{LS}\|_2}{\|x_{LS}\|_2} \leq \frac{\sigma_{n+1}^2}{\hat{\sigma}_n^2 - \sigma_{n+1}^2}.$$

It can be shown that an approximate condition number for the TLS solution is

$$\kappa_{TLS} \approx \frac{\hat{\sigma}_1}{\hat{\sigma}_n - \sigma_{n+1}} = \kappa(A) \frac{\hat{\sigma}_n}{\hat{\sigma}_n - \sigma_{n+1}}. \tag{8.7.15}$$

When $\hat{\sigma}_n - \sigma_{n+1} \ll \hat{\sigma}_n$ the TLS condition number can be much worse than for the LS problem.

Example 8.7.2.

Consider the overdetermined system

$$\begin{pmatrix} \hat{\sigma}_1 & 0 \\ 0 & \hat{\sigma}_2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \beta \end{pmatrix}. \tag{8.7.16}$$

Trivially, the LS solution is $x_{\rm LS} = (c_1/\hat{\sigma}_1, c_2/\hat{\sigma}_2)^T$, $\|r_{\rm LS}\|_2 = |\beta|$. If we take $\hat{\sigma}_1 = c_1 = 1$, $\hat{\sigma}_2 = c_2 = 10^{-6}$, then $x_{\rm LS} = (1,\ 1)^T$ is independent of β, and hence does not reflect the ill-conditioning of A. However,

$$\kappa_{LS}(A, b) = \kappa(A) \left(1 + \frac{\|r_{\rm LS}\|_2}{\|\hat{\sigma}_1 x_{\rm LS}\|_2} \right)$$

will increase proportionally to β. The TLS solution is of similar size as the LS solution as long as $|\beta| \leq \hat{\sigma}_2$. However, when $|\beta| \gg \hat{\sigma}_2$ then $\|x_{\rm TLS}\|_2$ becomes large.

In Figure 8.7.1 the two condition numbers are plotted as a function of $\beta \in [10^{-8}, 10^{-4}]$. For $\beta > \hat{\sigma}_2$ the condition number κ_{TLS} grows proportionally to β^2. It can be verified that $\|x_{\rm TLS}\|_2$ also grows proportionally to β^2.

Setting $c_1 = c_2 = 0$ gives $x_{LS} = 0$. If $|\beta| \geq \sigma_2(A)$, then $\sigma_2(A) = \sigma_3(A, b)$ and the TLS problem is nongeneric.

8.7.3 Some Generalized TLS Problems

We now consider the more general TLS problem with $d > 1$ right-hand sides

$$\min_{E,\ F} \| \begin{pmatrix} E & F \end{pmatrix} \|_F, \qquad (A + E)X = B + F, \tag{8.7.17}$$

where $B \in \mathbf{R}^{m \times d}$. The consistency relations can be written

$$\begin{pmatrix} B + F & A + E \end{pmatrix} \begin{pmatrix} -I_d \\ X \end{pmatrix} = 0,$$

Thus, we now seek perturbations $(E,\ F)$ that reduces the rank of the matrix $(\,B \quad A\,)$ by d. We call this a **multidimensional** TLS problem. As remarked before, for this problem to be meaningful the rows of the error matrix $(\,B+F \quad A+E\,)$ should be independently and identically distributed with zero mean and the same variance.

In contrast to the usual least squares problem, the multidimensional TLS problem is different from separately solving d one-dimensional TLS problems with right-hand sides $b_1, \ldots, b_d$. This is because in the multidimensional problem we require that *the matrix A be similarly perturbed for all right-hand sides.* This should give an improved predicted power of the TLS solution.

The solution to the TLS problem with multiple right-hand sides can be expressed in terms of the SVD

$$(\,B \quad A\,) = U\Sigma V^T = U_1\Sigma_1 V_1^T + U_2\Sigma_2 V_2^T, \tag{8.7.18}$$

where

$$\Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_n), \quad \Sigma_2 = \operatorname{diag}(\sigma_{n+1}, \ldots, \sigma_{n+d}),$$

and U and V partitioned conformally with $(\,B \quad A\,)$. Assuming that $\sigma_n > \sigma_{n+1}$, the minimizing perturbation is unique and given by the rank d matrix

$$(\,F \quad E\,) = -U_2\Sigma_2 V_2^T = -(\,B \quad A\,)\,V_2 V_2^T,$$

for which $\|\,(\,F \quad E\,)\,\|_F = \sum_{j=1}^{d} \sigma_{n+j}^2$ and $(\,B+F \quad A+E\,)\,V_2 = 0$. Assume that

$$V_2 = \begin{pmatrix} V_{12} \\ V_{22} \end{pmatrix}.$$

with $V_{12} \in \mathbf{R}^{d\times d}$ nonsingular. Then the solution to the TLS problem is unique and given by

$$X = -V_{22}V_{12}^{-1} \in \mathbf{R}^{n\times d}.$$

We show that if $\sigma_n(A) > \sigma_{n+1}\,(\,B \quad A\,)$, then V_{12} is nonsingular. From (8.7.18) it follows that $BV_{12} + AV_{22} = U_2\Sigma_2$. Now, suppose that V_{12} is singular. Then $V_{12}x = 0$ for some unit vector x. It follows that $U_2\Sigma_2 x = AV_{12}x$. From $V_2^T V_2 = V_{12}^T V_{12} + V_{22}^T V_{22} = I$ it follows that $V_{22}^T V_{22} x = x$ and hence $\|V_{22}x\|_2 = 1$. But then

$$\sigma_{n+1}\,(\,B \quad A\,) \geq \|U_2\Sigma_2 x\|_2 = \|AV_{12}x\|_2 \geq \sigma_n(A),$$

a contradiction. Hence, V_{12} is nonsingular.

From the above characterization it follows that the TLS solution satisfies

$$\begin{pmatrix} B^T \\ A^T \end{pmatrix} (\,B \quad A\,) \begin{pmatrix} -I_d \\ -X \end{pmatrix} = \begin{pmatrix} -I_d \\ X \end{pmatrix} C, \tag{8.7.19}$$

where

$$C = V_{22}\Sigma_2^2 V_{22}^{-1} \in \mathcal{R}^{d\times d}. \tag{8.7.20}$$

Note that C is symmetrizable but not symmetric! Multiplying (8.7.19) from the left with $(I_d \ X^T)$ gives

$$(B - AX)^T(B - AX) = (X^T X + I_d)C,$$

and if $(X^TX + I_d)$ is nonsingular,

$$C = (X^TX + I_d)^{-1}(B - AX)^T(B - AX), \tag{8.7.21}$$

The multidimensional TLS solution X_{TLS} minimizes $\|C\|_F$, which generalizes the result for $d = 1$.

The last block component of (8.7.19) reads

$$A^TAX - XC = A^TB,$$

which is a Sylvester equation for X. This has a unique solution if and only if A^TA and C have no common eigenvalues. which is the case if $\hat{\sigma}_n > \sigma_{n+1}$.

Now assume that $\sigma_k > \sigma_{k+1} = \cdots = \sigma_{n+1}$, $k < n$, and set $V_2 = (v_{k+1}, \ldots, v_{n+d})$. Let Q be a product of Householder transformations such that

$$V_2Q = \begin{pmatrix} \Gamma & 0 \\ Z & Y \end{pmatrix},$$

where $\Gamma \in \mathbf{R}^{d\times d}$ is lower triangular. If Γ is nonsingular, then the TLS solution of minimum norm is given by

$$X = -Z\Gamma^{-1}.$$

In many parameter estimation problems, some of the columns are known exactly. It is no restriction to assume that the error-free columns are in leading positions in A. In the multivariate version of this **mixed LS-TLS problem** one has a linear relation

$$(A_1, A_2 + E_2)X = B + F, \qquad A_1 \in \mathbf{R}^{m\times n_1},$$

where $A = (A_1,\ A_2) \in \mathbf{R}^{m\times n}$, $n = n_1 + n_2$. It is assumed that the rows of the errors $(E_2,\ F)$ are independently and identically distributed with zero mean and the same variance. The mixed LS-TLS problem can then be expressed

$$\min_{E_2,F} \|(E_2,\ F)\|_F, \qquad (A_1,\ A_2 + E_2)X = B + F. \tag{8.7.22}$$

When A_2 is empty, this reduces to solving an ordinary least squares problem. When A_1 is empty this is the standard TLS problem. Hence, this mixed problem includes both extreme cases.

The solution of the mixed LS-TLS problem can be obtained by first computing a QR factorization of A and then solving a TLS problem of reduced dimension.

Algorithm 8.11. Mixed LS-TLS problem.

Let $A = (A_1,\ A_2) \in \mathbf{R}^{m\times n}$, $n = n_1 + n_2$, $m \geq n$, and $B \in \mathbf{R}^{m\times d}$. Assume that the columns of A_1 are linearly independent. Then the following algorithm solves the mixed LS-TLS problem (8.7.22).

Step 1. Compute the QR factorization

$$(A_1,\ A_2,\ B) = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix},$$

where Q is orthogonal, and $R_{11} \in \mathbf{R}^{n_1 \times n_1}$, $R_{22} \in \mathbf{R}^{(n_2+d)\times(n_2+d)}$ are upper triangular. If $n_1 = n$, then the solution X is obtained by solving $R_{11}X = R_{12}$ (usual least squares); otherwise continue (solve a reduced TLS problem).

Step 2. Compute the SVD of R_{22}

$$R_{22} = U\Sigma V^T, \qquad \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_{n_2+d}),$$

where the singular values are ordered in decreasing order of magnitude.

Step 3a. Determine $k \le n_2$ such that

$$\sigma_k > \sigma_{k+1} = \cdots = \sigma_{n_2+d} = 0,$$

and set $V_{22} = (v_{k+1}, \ldots, v_{n_2+d})$. If $n_1 > 0$ then compute V_2 by back-substitution from

$$R_{11}V_{12} = -R_{12}V_{22}, \qquad V_2 = \begin{pmatrix} V_{12} \\ V_{22} \end{pmatrix},$$

else set $V_2 = V_{22}$.

Step 3b. Perform Householder transformations such that

$$V_2 Q = \begin{pmatrix} \Gamma & 0 \\ Z & Y \end{pmatrix},$$

where $\Gamma \in \mathbf{R}^{d\times d}$ is upper triangular. If Γ is nonsingular then the solution is

$$X = -Z\Gamma^{-1}.$$

Otherwise the TLS problem is nongeneric and has no solution.

Note that the QR factorization in the first step would be the first step in computing the SVD of A.

8.7.4 Bidiagonalization and TLS Problems.

One way to avoid the complications of nongeneric problems is to compute a regular core TLS problem by bidiagonalizing of the matrix $(\,b \quad A\,)$. Consider the TLS problem

$$\min_{E,r} \|(E,\ r)\|_F, \qquad (A+E)x = b + r.$$

It was shown in Section 8.4.5 that we can always find square orthogonal matrices $\widetilde{U}_{k+1}$ and $\widetilde{V}_k = P_1 P_2 \cdots P_k$ such that

$$\widetilde{U}_{k+1}^T \begin{pmatrix} b & A\widetilde{V}_k \end{pmatrix} = \begin{pmatrix} \beta_1 e_1 & B_k & 0 \\ 0 & 0 & A_k \end{pmatrix}, \tag{8.7.23}$$

where

$$B_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{pmatrix} \in \mathbf{R}^{(k+1)\times k},$$

and

$$\beta_j \alpha_j \neq 0, \quad j = 1:k. \tag{8.7.24}$$

Setting $x = \widetilde{V}_k \begin{pmatrix} y \\ z \end{pmatrix}$, the approximation problem $Ax \approx b$ then decomposes into the two subproblems

$$B_k y \approx \beta_1 e_1, \qquad A_k z \approx 0.$$

It seems reasonable to simply take $z = 0$, and separately solve the first subproblem, which is the minimally dimensioned **core subproblem**. Setting

$$V_k = \widetilde{V}_k \begin{pmatrix} I_k \\ 0 \end{pmatrix}, \qquad U_{k+1} = \widetilde{U}_{k+1} \begin{pmatrix} I_{k+1} \\ 0 \end{pmatrix},$$

it follows that

$$(\, b \quad AV_k \,) = U_{k+1} \, (\, \beta_1 e_1 \quad B_k \,).$$

If $x = V_k y \in \mathcal{R}(V_k)$ then

$$(A + E)x = (A + E)V_k y = (U_{k+1} B_k + E V_k) y = \beta_1 U_{k+1} e_1 + r,$$

Hence, the consistency relation $(A + E_k)x = b + r$ becomes

$$(B_k + F)y = \beta_1 e_1 + s, \quad F = U_{k+1}^T E V_k, \quad s = U_{k+1}^T r. \tag{8.7.25}$$

Using the orthogonality of U_{k+1} and V_k it follows that

$$\|(E, r)\|_F = \|(F, s)\|_F. \tag{8.7.26}$$

Hence, to minimize $\|(E, r)\|_F$ we should take y_k to be the solution to the TLS core subproblem

$$\min_{F,s} \|(F,\ s)\|_F, \qquad (B_k + F)y = \beta_1 e_1 + s. \tag{8.7.27}$$

From (8.7.24) and Theorem 8.4.3 it follows that the singular values of the matrix B_k are simple and that the right hand side βe_1 has nonzero components along each left singular vector. This TLS problem therefore must have a unique solution. Note that we can assume that $\beta_{k+1} \neq 0$, since otherwise the system is compatible.

To solve this subproblem we need to compute the SVD of the bidiagonal matrix

$$(\beta_1 e_1,\ B_k) = \begin{pmatrix} \beta_1 & \alpha_1 & & & \\ & \beta_2 & \alpha_2 & & \\ & & \beta_3 & \ddots & \\ & & & \ddots & \alpha_k \\ & & & & \beta_{k+1} \end{pmatrix} \in \mathbf{R}^{(k+1)\times(k+1)}. \tag{8.7.28}$$

The SVD of this matrix

$$(\beta_1 e_1,\ B_k) = P\text{diag}\,(\sigma_1,\ldots,\sigma_{k+1})Q^T, \quad P,Q \in \mathbf{R}^{(k+1)\times(k+1)}$$

can be computed, e.g., by the implicit QR-SVD algorithm; see Section 9.7.6. (Note that the first stage in this is a transformation to bidiagonal form, so the work in performing the reduction (8.7.23) has not been wasted!) Then with

$$q_{k+1} = Qe_{k+1} = \begin{pmatrix} \omega \\ z \end{pmatrix}.$$

Here it is always the case that $\omega \neq 0$ and the solution to the original TLS problem (8.7.27) equals

$$x_{TLS} = V_k y = -\omega^{-1} V_k z.$$

Further, the norm of the perturbation equals

$$\min_{E,r} \|(E,r)\|_F = \sigma_{k+1}.$$

8.7.5 Solving Overdetermined Systems in the l_p Sense.

In some applications it might be more adequate to minimize some l_p-norm of the residual vector other than the l_2 norm. We consider the problem of minimizing

$$\|r\|_p = \left(\sum_{i=1}^{m} |r_i|^p\right)^{1/p}, \quad 1 \le p \le \infty. \tag{8.7.29}$$

For $1 < p < \infty$ this problem is strictly convex and hence has exactly one solution. For $p = 1$ the solution may not be unique. Minimization in the l_1-norm or l_∞-norm is complicated by the fact that the function $f(x) = \|Ax - b\|_p$ is only piecewise differentiable when $p = 1$ or $p = \infty$.

Example 8.7.3.

To illustrate the effect of using a different norm we consider the problem of estimating the scalar x from m observations $b \in \mathbf{R}^m$. This is equivalent to minimizing $\|Ax - b\|_p$, with $A = e = (1,1,\ldots,1)^T$. It is easily verified that if $b_1 \ge b_2 \ge \ldots \ge b_m$, then the solution x_p for some different values p are

$$\begin{aligned} x_1 &= b_{\frac{m+1}{2}}, \quad (m \text{ odd}) \\ x_2 &= \frac{1}{m}(b_1 + b_2 + \ldots + b_m), \\ x_\infty &= \frac{1}{2}(b_1 + b_m). \end{aligned}$$

These estimates correspond to the median, mean, and midrange respectively. Note that the estimate x_1 is insensitive to the extreme values of b_i, while x_∞ depends

only on the extreme values. The l_∞ solution has the property that the absolute error in at least n equations equals the maximum error.

A similar effect is also achieved with p greater than but close to 1. Approximation in the maximum norm is often called Chebyshev approximation. We avoid this terminology, since Chebyshev's name is also naturally associated with two entirely different approximation methods: interpolation in the Chebyshev abscissae, and expansion in a series of Chebyshev polynomials.

To determine the l-1 solution of an overdetermined linear system $Ax = b$, $A \in \mathbf{R}^{m\times n}$, we want to find a vector x that minimizes

$$\phi(x) = \|Ax - b\|_1 = \sum_{i=1}^{m} |a_i^T x - b_i|. \tag{8.7.30}$$

Here a_i^T denotes the ith row of A. This problem can be cast in the form

$$\begin{aligned} \text{minimize} \quad & e^T w = \sum_{i=1}^{m} w_i, \\ \text{subject to} \quad & w_i \geq (a_i^T x - b_i) \\ \text{and} \quad & w_i \geq (b_i - a_i^T x), \quad i = 1 : m, \end{aligned}$$

where $e = (1, \ldots, 1)^T$. This is a linear programming problem with $2m$ linear inequalities. In principle this problem can be solved by the simplex method; see Section 9.3.3.

The simple example above illustrates that the l_1 norm of the residual vector has the advantage of giving a **robust** solution, i.e., a small number of isolated large errors will usually not have a big effect on the solution. More generally, in robust linear regression possible outsiders among the data points are identified and given less weight. The most popular among the robust estimators is Huber's M-estimator. This uses the least squares estimator for "normal" data but the l_1 norm estimator for data points that disagree with the normal picture. More precisely, the Huber M-estimate minimizes the objective function

$$\psi(x) = \sum_{i=1}^{m} \rho(r_j(x)/\sigma), \tag{8.7.31}$$

where

$$\rho(t) = \begin{cases} \frac{1}{2}t^2, & \text{if } |t| \leq \gamma; \\ \gamma|t| - \frac{1}{2}\gamma^2, & \text{if } |t| > \gamma, \end{cases} \tag{8.7.32}$$

γ is a tuning parameter, and σ a scaling factor which depends on the data to be estimated. In the following we assume that σ is a constant, and then it is no restriction to take $\sigma = 1$.

Like the l_p estimator for $1 < p < 2$, the Huber estimator can be viewed as a compromise between l_2 and l_1 approximation. For large values of γ it will be close to the least squares estimator; for small values of γ it is close to the l_1 estimator.

An efficient continuation algorithm for solving the l_1 problem, which uses a finite sequence of Huber estimates with decreasing parameter γ has been developed by Madsen and Nielsen [422].

O'Leary [454, 1990] has studied different implementations of Newton-like methods. The Newton step s for minimizing $\psi(x)$ in (8.7.31) ($\sigma = 1$) is given by the solution of

$$A^T DAs = A^T y,$$

where

$$y_i = \rho'(r_i), \qquad D = \operatorname{diag}(\rho''(r_i)), \quad i = 1:m.$$

O'Leary recommends that at the initial iterations the cutoff value γ in the Huber function (8.7.32) is decreased from a very large number to the desired value. This has the effect of starting the iteration from the least squares solution and helps prevent the occurrence of rank deficient Hessian matrices H.

Another approach to solve the l_p norm problem when $1 < p < 3$, is the method of **iteratively reweighted least squares** (IRLS) see Osborne [458]. This amounts to solving a certain sequence of weighted least squares problems. We start by noting that, provided that $|r_i(x)| = |b - Ax|_i > 0$, $i = 1:m$, the problem (8.7.29) can be restated in the form $\min_x \psi(x)$, where

$$\psi(x) = \sum_{i=1}^{m} |r_i(x)|^p = \sum_{i=1}^{m} |r_i(x)|^{p-2} r_i(x)^2. \tag{8.7.33}$$

This can be interpreted as a weighted least squares problem

$$\min_x \|D(r)^{(p-2)/2}(b - Ax)\|_2, \qquad D(r) = \operatorname{diag}(|r|), \tag{8.7.34}$$

where $\operatorname{diag}(|r|)$ denotes the diagonal matrix with ith component $|r_i|$. The diagonal weight matrix $D(r)^{(p-2)/2}$ in (8.7.34) depends on the unknown solution x, but we can attempt to use the following iterative method.

Algorithm 8.12.
IRLS for l_p Approximation $1 < p < 2$
Let $x^{(0)}$ be an initial approximation such that $r_i^{(0)} = (b - Ax^{(0)})_i \neq 0$, $i = 1, \ldots, n$.

for $k = 0, 1, 2, \ldots$
$r_i^{(k)} = (b - Ax^{(k)})_i$;
$D_k = \operatorname{diag}\left(\left(|r_i^{(k)}|\right)^{(p-2)/2}\right)$;
solve $\delta x^{(k)}$ from
$\min_{\delta x} \left\|D_k(r^{(k)} - A\delta x)\right\|_2$;
$x^{(k+1)} = x^{(k)} + \delta x^{(k)}$;
end

Since $D_k b = D_k(r^{(k)} - Ax^{(k)})$, it follows that $x^{(k+1)}$ in IRLS solves $\min_x \|D_k(b - Ax)\|_2$, but the implementation above is to be preferred. It has been assumed that

in the IRLS algorithm, at each iteration $r_i^{(k)} \neq 0$, $i = 1, \ldots, n$. In practice this cannot be guaranteed, and it is customary to modify the algorithm so that

$$D_k = \operatorname{diag}\left(\left(100ue + |r^{(k)}|\right)^{(p-2)/2}\right),$$

where u is the machine precision and $e^T = (1, \ldots, 1)$ is the vector of all ones. Because the weight matrix D_k is not constant, the simplest implementations of IRLS recompute, e.g., the QR factorization of $D_k A$ in each step. It should be pointed out that the iterations can be carried out entirely in the r space without the x variables. Upon convergence to a residual vector r_{opt} the corresponding solution can be found by solving the consistent linear system $Ax = b - r_{\text{opt}}$.

It can be shown that in the l_p case any fixed point of the IRLS iteration satisfies the necessary conditions for a minimum of $\psi(x)$. The IRLS method is convergent for $1 < p < 3$, and also for $p = 1$ provided that the l_1 approximation problem has a unique nondegenerate solution. However, the IRLS method can be extremely slow when p is close to unity.

Review Questions

8.1 Formulate the total least squares (TLS) problem. The solution of the TLS problem is related to a theorem on matrix approximation. Which?

Problems and Computer Exercises

8.1 Consider a TLS problem where $n = 1$ and

$$C = (A, b) = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}.$$

Show that the unique minimizing ΔC gives

$$C + \Delta C = (A + E, b + r) = \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix}$$

so the perturbed system is not compatible, but that an arbitrary small perturbation ϵ in the (2,1) element will give a compatible system with solution $x = 2/\epsilon$.

8.2 (a) Let $A \in \mathbf{R}^{m\times n}$, $m \geq n$, $b \in \mathbf{R}^m$, and consider the total least squares (TLS) problem. $\min_{E,r} \|(E, r)\|_F$, where $(A+E)x = b+r$. If we have the QR factorization

$$Q^T(A,\ b) = \begin{pmatrix} S \\ 0 \end{pmatrix}, \qquad S = \begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix}.$$

then the ordinary least squares solution is $x_{LS} = R^{-1}z$, $\|r\|_2 = \rho$.

Show that if a TLS solution x_{TLS} exists, then it holds

$$\begin{pmatrix} R^T & 0 \\ z^T & \rho \end{pmatrix} \begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix} \begin{pmatrix} x_{TLS} \\ -1 \end{pmatrix} = \sigma_{n+1}^2 \begin{pmatrix} x_{TLS} \\ -1 \end{pmatrix},$$

where σ_{n+1} is the smallest singular value of $(A,\ b)$.

(b) Write a program using inverse iteration to compute x_{TLS}, i.e., for $k = 0, 1, 2, \ldots$, compute a sequence of vectors $x^{(k+1)}$ by

$$\begin{pmatrix} R^T & 0 \\ z^T & \rho \end{pmatrix} \begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix} \begin{pmatrix} y^{(k+1)} \\ -\alpha \end{pmatrix} = \begin{pmatrix} x^{(k)} \\ -1 \end{pmatrix}, \qquad x^{(k+1)} = y^{(k+1)}/\alpha.$$

As starting vector use $x^{(0)} = x_{LS}$ on the assumption that x_{TLS} is a good approximation to x_{LS}. Will the above iteration always converge? Try to make it fail!

(c) Study the effect of scaling the right hand side in the TLS problem by making the substitution $z := \theta z$, $\rho := \theta\rho$. Plot $\|x_{TLS}(\theta) - x_{LS}\|_2$ as a function of θ and verify that when $\theta \to 0$, then $x_{TLS} \to x_{LS}$.

Hint For generating test problems it is suggested that you use the function qmult(A) from the MATLAB collection of test matrices by N. Higham to generate a matrix $C = (A,\ b) = Q_1 * D * Q_2^T$, where Q_1 and Q_2 are random real orthogonal matrices and D a given diagonal matrix. This allows you to generate problems where C has known singular values and vectors.

Notes and Further References

Modern numerical methods for solving least squares problems are surveyed in the two comprehensive monographs [399] and [61]. The latter contains a bibliography of 860 references, indicating the considerable research interest in these problems. Hansen [310] gives an excellent survey of numerical methods for the treatment of numerically rank deficient linear systems arising, for example, from discrete ill-posed problems.

Several of the great mathematicians at the turn of the 19th century worked on methods for solving overdetermined linear systems. Laplace in 1799 used the principle of minimizing the sum of absolute errors $|r_i|$, with the added conditions that the errors sum to zero. This leads to a solution x that satisfies at least n equations exactly. The method of least squares was first published as an algebraic procedure by Legendre 1805 in [401]. Gauss justified the least squares principle as a statistical procedure in [233], where he claimed to have used the method since 1795. This led to one of the most famous priority dispute in the history of mathematics. Gauss further developed the statistical aspects in 1821–1823. For an interesting accounts of the history of the invention of least squares, see Stiegler [555, 1981].

Because of its success in analyzing astronomical data the method of least squares rapidly became the method of choice when analyzing observation. Geodetic calculations was another early area of application of the least squares principle. In the last decade applications in control and signal processing has been a source of inspiration for developments in least squares calculations.

Section 8.1

Gauss [232] in 1821 put the method of least squares on a sound theoretical basis. In his textbook from 1912 Markov [424] refers to Gauss' work and may have clarified some implicit assumptions but proves nothing new; see Placket [487, 488]

A detailed account of the interesting early history of the SVD is given by Stewart [547, 1993]. Beltrami [42][37]in 1873 derived the SVD for a real, square, nonsingular matrix having distinct singular values. A year later Jordan [356] independently published a derivation of the SVD which handled multiple singular values. Jordan also stated the variational characterization of the largest singular value as the maximum of a function. Auton [16] extended the SVD to complex matrices and Eckart and Young [183] to rectangular matrices. The relation to the polar decomposition is due to Autonne [16]. Picard [486] seems to have been the first to call the numbers σ_k singular values.

Both Beltrami and Jordan were concerned with diagonalizing a finite bilinear form $P = x^T Ay$. Schmidt [521] developed in 1907 the infinte-dimensional analogue of the SVD as a tool for solving integral equations. He used it to obtain a low-rank approximation to the integral operator. Weyl [607] developed a general perturbation theory and gave a more elegant proof of Schmidt's approximation theorem.

The first stable algorithm for computing the SVD the singular value was developed by Golub, Reinsch, and Wilkinson in the late 1960's. Several applications of the SVD to matrix approximation can be found in Golub and Van Loan [277, Sec. 12.4].

Theorem 8.1.2 can be generalized to to the semidefinite case, see Gulliksson and Wedin [294, Theorem 3.2]. A case when B is indefinite and nonsingular is considered in Section 8.6.2

A good introduction to generalized inverses is given by Ben-Israel and Greville [43, 1976]. Generalized inverses should be used with caution since the notation tends to hide intrinsic computational difficulties associated with rank deficient matrices. A more complete and thorough treatment is given in the monograph by the same authors [44, 2003]. The use of generalized inverses in geodetic calculations is treated in Bjerhammar [56, 1973].

Section 8.2

Peters and Wilkinson [484, 1970] developed methods based on Gaussian elimination from a uniform standpoint and the excellent survey by Noble [451, 1976]. Sautter [520, 1978] gives a detailed analysis of stability and rounding errors of the LU algorithm for computing pseudo-inverse solutions. For a fuller treatment of weighted least squares and the general linear model, see Björck [61, Chap.3].

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 Karlsson et al. [598]; see also [368]. Gu [287] gives several approximations to that are optimal up

[37]Eugenio Beltrami (1835–1900), was born in Cremona, Italy, which then belonged to Austria. He studied applied mathematics in Pavia and Milan. In 1864 was appointed to the chair of geodesy at the University of Pisa and from 1966 he was professor of rational mechanics in Bologna. In 1973 he moved to Rome, which in 1870 had became the new capital of Italy. After three years there he moved back to Pavia. Beltrami contributed to work in differential geometry on curves and surfaces and gave a concrete realization of the non-euclidian geomtery.

to a factor less than 2. Optimal backward perturbation bounds for underdetermined systems are derived in [560]. The extension of backward error bounds to the case of constrained least squares problems is discussed by Cox and Higham [121].

Section 8.3

Jacobi [350] used Givens rotations already in 1845 to achieve diagonal dominance in systems of normal equations and then applying a simple iterative scheme that became known as Jacobi's method. See Section 10.1.

Complex Givens rotations and complex Householder transformations are treated in detail by Wilkinson [611, pp. 47–50]. Lehoucq [402, 1996] gives a comparison of different implementations of complex Householder transformations. The reliable construction of real and complex Givens rotations are considered in great detail in Bindel, Demmel and Kahan [53]. Wilkinson [611] showed the backward stability of algorithm based on a sequence of Householder transformations.

The different computational variants of Gram–Schmidt have an interesting history. The "modified" Gram–Schmidt (MGS) algorithm was in fact already derived by Laplace in 1816 as an elimination method using weighted row sums. Laplace did not interpret his algorithm in terms of orthogonalization, nor did he use it for computing least squares solutions! Bienaymé in 1853 gave a similar derivation of a slightly more general algorithm; see Björck [60, 1994]. What is now called the "classical" Gram–Schmidt (CGS) algorithm first appeared explicitly in papers by Gram 1883 and Schmidt 1908. Schmidt treats the solution of linear systems with infinitely many unknowns and uses the orthogonalization as a theoretical tool rather than a computational procedure.

In the 1950's algorithms based on Gram–Schmidt orthogonalization were frequently used, although their numerical properties were not well understood at the time. Björck [57] analyzed the modified Gram–Schmidt algorithm and showed its stability for solving linear least squares problems.

The systematic use of orthogonal transformations to reduce matrices to simpler form was initiated by Givens [257] and Householder [341, 1958]. The application of these transformations to linear least squares is due to Golub [261, 1965], where it was shown how to compute a QR factorization of A using Householder transformations. An Algol implementation of this method was given in [87].

Section 8.4

An efficient algorithm for solving rank-deficient least squares problem is given by Foster and Kommu [216]. This uses a truncated pivoted QR factorization where the rank of the trailing diagonal block is estimated by a condition estimator. A different approach to the subset selection problem has been given by de Hoog and Mattheij [141]. They consider choosing the square subset A_1 of rows (or columns), which maximizes $det(A_1)$.

Matlab templates for computing UTV decompositions has been given by Fierro, Hansen, and Hansen [202].

Partial least squares originated in statistical applications, specifically economy Herman Wold [615]. It became very popular in chemometrics, where it was introduced by Svante Wold; see [616]. Now it is becoming a method of choice also in a large number of applications in the social sciences. Unfortunately the statisticians

use a different implementation, the numerical stability of which has not be proved.

Section 8.5

The QR algorithm for banded rectangular matrices was first given by Reid [498]. Rank-revealing QR (RRQR) factorizations have been studied by a number of authors. A good survey can be found in Hansen [310]. The URV and ULV decompositions were introduced by Stewart [546, 548].

For a detailed discussion of dissection and orthogonal factorizations in geodetic survey problems, see Golub and Plemmons [260].

This block triangular form (8.5.18) of a sparse matrix is based on a canonical decomposition of bipartite graphs discovered by Dulmage, Mendelsohn, and Johnson in a series of papers; see [180].

The row sequential sparse QR algorithm employing Givens rotations is due to George and Heath [236, 1980]. Liu [412] introduces the notion of *row merge tree* for sparse QR factorization by Givens rotations. This row merging scheme can be viewed as the implicitly using the multifrontal method on A^TA. This algorithm can give a significant reduction in QR factorization time at a modest increase in working storage. George and Liu [240] give a modified version of Liu's algorithm, which uses Householder transformations instead of Givens rotations. For other multifrontal codes developed for sparse QR factorizations, see Matstoms [430]. ,Supernodes and other modifications of the multifrontal method are discussed by Liu [413] and Matstoms [429].

Section 8.6

An early reference to the exchange operator is in network analysis; see the survey of Tsatsomeros [579]. J-orthogonal matrices also play a role in the solution of the generalized eigenvalue problem $Ax = \lambda Bx$; see Section 10.7. For a systematic study of J-orthogonal matrices and their many applications we refer to Higham [330]. An error analysis of Chamber's algorithm is given by Bojanczyk et al. [70].

The systematic use of GQR as a basic conceptual and computational tool are explored by [464]. These generalized decompositions and their applications are discussed in [11]. Algorithms for computing the bidiagonal decomposition are due to Golub and Kahan [265, 1965]. The partial least squares (PLS) method, which has become a standard tool in chemometrics, goes back to Wold et al. [616].

The term "total least squares problem", which was coined by Golub and Van Loan [276], renewed the interest in the "errors in variable model". A thorough and rigorous treatment of the TLS problem is found in Van Huffel and Vandewalle [587]. The important role of the core problem for weighted TLS problems was discovered by Paige and Strakoš [472].

Section 8.7

For the the l_1 problem algorithms which use linear programming techniques have been developed by Barrodale and Roberts [34]. The Harwell Subroutine Library uses a version of their algorithm. Other algorithms for this problem based on projected gradient techniques are given by Bartels, Conn, and Sinclair [35] A general treatment of robust statistical procedures is given by Huber [344]. A great deal of work has been devoted to developing methods for computing the Huber M-

estimator; see, e.g., Clark and Osborne [112, 1986], Ekblom [186, 1988], and Madsen and Nielsen [421, 1990]