Chapter 2
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-Marie Legendre, Nouvelles Méthodes pour la Détermination des Orbites des Comètes. Paris, 1805

2.1 Introduction to Least Squares Methods

A fundamental task in scientific computing is to estimate parameters in a mathematical model from observations that are subject to errors. A common practice is to reduce the influence of the errors by using more observations than the number of parameters. Consider a model described by a scalar function $y(t) = f(c, t)$, where

$$f(c, t) = \sum_{j=1}^{n} c_j \phi_j(t) \tag{2.1.1}$$

is a linear combination of a set of basis functions $\phi_j(t)$, and $c = (c_1, \ldots, c_n)^T \in \mathbb{R}^n$ is a parameter vector to be determined from measurements (y_i, t_i), $i = 1{:}m$, $m > n$. The equations $y_i = f(c, t_i), \quad i = 1{:}m$, form a linear system, which can be written in matrix form as $Ac = y$, $a_{ij} = \phi_j(t_i)$. Due to of errors in the observations, the system is inconsistent, and we have to be content with finding a vector $c \in \mathbb{R}^n$ such that Ac in some sense is the "best" approximation to $y \in \mathbb{R}^m$.

There are many possible ways of defining the "best" solution to an inconsistent linear equation $Ax = b$. A natural objective is to make the **residual vector** $r = b - Ax$ small. A choice that can be motivated for statistical reasons (see Theorem 2.1.1, p. 241) and leads to a simple computational problem is to take c to be a vector that minimizes the sum of squares $\sum_{i=1}^{m} r_i^2$. This can be written as

$$\min_x \|Ax - b\|_2, \tag{2.1.2}$$

Å. Björck, *Numerical Methods in Matrix Computations*,
Texts in Applied Mathematics 59, DOI 10.1007/978-3-319-05089-8_2

which is the **linear least squares problem**. The minimizer x is called a **least squares solution** of the system $Ax = b$.

Many of the great mathematicians at the turn of the 19th century worked on methods for "solving" overdetermined linear systems. In 1799 Laplace used the principle of minimizing the sum of the absolute residuals with the added condition that they sum to zero. He showed that the solution must then satisfy exactly n out of the m equations. Gauss argued that since greater or smaller errors are equally possible in all equations, a solution that satisfies precisely n equations must be regarded as less consistent with the laws of probability. He was then led to the principle of least squares. Although the method of least squares was first published by Legendre in 1805, Gauss claimed he discovered the method in 1795 and used it for analyzing surveying data and for astronomical calculations. Its success in analyzing astronomical data ensured that the method of least squares rapidly became the method of choice for analyzing observations. Another early important area of application was Geodetic calculations.

Example 2.1.1 An example of large-scale least squares problems solved today, concerns the determination of the Earth's gravity field from highly accurate satellite measurements; see Duff and Gratton [77, 2006]). The model considered for the gravitational potential is

$$V(r,\theta,\lambda) = \frac{GM}{R}\sum_{l=0}^{L}\left(\frac{r}{R}\right)^{l+1}\sum_{m=0}^{l}P_{lm}(\cos\theta)\big[C_{lm}\cos m\lambda + S_{lm}\sin m\lambda\big],$$

where G is the gravitational constant, M is the Earth's mass, R is the Earth's reference radius, and P_{lm} are the normalized Legendre polynomials of order m. The normalized harmonic coefficients C_{lm}, and S_{lm} are to be determined. For $L = 300$, the resulting least squares problem, which involves 90,000 unknowns and millions of observations, needs to be solved on a daily basis. Better gravity-field models are important for a wide range of application areas. □

2.1.1 The Gauss–Markov Model

To describe Gauss's theoretical basis for the method of least squares we need to introduce some concepts from statistics. Let y be a random variable and $F(x)$ be the probability that $y \le x$. The function $F(x)$ is called the **distribution function** for y and is a nondecreasing and right-continuous function that satisfies

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

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

$$\mathcal{E}(y) = \mu = \int_{-\infty}^{\infty} y\,dF(y) \quad \text{and} \quad \mathcal{E}(y-\mu)^2 = \sigma^2 = \int_{-\infty}^{\infty}(y-\mu)^2\,dF(y).$$

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

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

and σ_{ii} is the variance of the component y_i. The covariance matrix of the vector y is

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

Definition 2.1.1 In the **Gauss–Markov model** it is assumed that a linear relationship $Ax = z$ holds, where $A \in \mathbb{R}^{m\times n}$ is a known matrix, $x \in \mathbb{R}^n$ is a vector of unknown parameters, and z is a constant but unknown vector. Let $b = z + e \in \mathbb{R}^m$ be a vector of observations, where e is a random error vector such that

$$\mathcal{E}(e) = 0, \qquad \mathcal{V}(e) = \sigma^2 V. \tag{2.1.3}$$

Here $V \in \mathbb{R}^{m\times m}$ is a symmetric nonnegative definite matrix and σ^2 an unknown constant. In the **standard case** the errors are assumed to be uncorrelated and with the same variance, i.e., $V = I_m$.

Remark 2.1.1 In statistical literature the Gauss–Markov model is traditionally written $X\beta = y + e$. For consistency, a different notation is used in this book.

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

Lemma 2.1.1 *Let $B \in \mathbb{R}^{r\times n}$ be a matrix and y a random vector with $\mathcal{E}(y) = \mu$ and covariance matrix $\sigma^2 V$. Then the expected value and covariance matrix of By is*

$$\mathcal{E}(By) = B\mu, \qquad \mathcal{V}(By) = \sigma^2 BVB^T. \tag{2.1.4}$$

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

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

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

□

The linear function $c^T y$ of the random vector y is an **unbiased estimate** of a parameter θ if $\mathcal{E}(c^T y) = \theta$. When such a function exists, θ is called an **estimable**

parameter. Furthermore, $c^T y$ is a minimum variance (best) linear unbiased estimate of θ if $\mathcal{V}(c^T y)$ is minimized over all such linear estimators.

Theorem 2.1.1 (The Gauss–Markov Theorem) *Consider a linear Gauss–Markov model $Ax = z$, where the matrix $A \in \mathbb{R}^{m\times n}$ has rank n. Let $b = z + e$, where e is a random vector with zero mean and covariance matrix $\sigma^2 I$. Then the best linear unbiased estimator of x is the vector $\widehat{x}$ that minimizes the sum of squares $\|Ax - b\|_2^2$. This vector is unique and equal to the solution to the* **normal equations**

$$A^T Ax = A^T b. \tag{2.1.5}$$

More generally, $c^T\widehat{x}$ is the best linear unbiased estimator of any linear functional $\theta = c^T x$. The covariance matrix of $\widehat{x}$ is

$$\mathcal{V}(\widehat{x}) = \sigma^2 (A^T A)^{-1}. \tag{2.1.6}$$

An unbiased estimate of σ^2 is given by

$$s^2 = \frac{\widehat{r}^T\widehat{r}}{m-n},$$

where $\widehat{r} = b - A\widehat{x}$ is the estimated residual vector.

Proof Let $\widehat{\theta} = d^T b$ be an unbiased estimate of $\theta = c^T x$. Then, since

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

$A^T d = c$ and from Lemma 2.1.1 it follows that $\mathcal{V}(g) = \sigma^2\|d\|_2^2$. Thus, we wish to minimize $d^T d$ subject to $A^T d = c$. Set

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

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

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

or $d = Az$. Premultiplying this by A^T gives $A^T Az = A^T d = c$. Since the columns of A are linearly independent, $x \neq 0$ implies that $Ax \neq 0$ and therefore $x^T A^T Ax = \|Ax\|_2^2 > 0$. Hence, $A^T A$ is positive definite and nonsingular. We obtain $z = (A^T A)^{-1}c$ and the best unbiased linear estimate is

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

where $\widehat{x}$ is the solution to the normal equations.

It remains to show that the same result is obtained if the sum of squares $Q(x) = (b - Ax)^T(b - Ax)$ is minimized. Taking derivatives with respect to x gives

$$\frac{\partial Q}{\partial x} = -2A^T(b - Ax) = 0,$$

which gives the normal equations. One can readily show that this is a minimum by virtue of the inequality $\|b - Ay\|_2^2 = \|b - Ax\|_2^2 + \|A(x - y)\|_2^2 \geq \|b - Ax\|_2^2$, which holds if x satisfies the normal equations. □

Remark 2.1.2 In the literature, the Gauss–Markov theorem is sometimes stated in less general forms. In the theorem, errors are *not* assumed to be normally distributed, nor are they assumed to be independent and identically distributed (only uncorrelated and to have zero mean and equal variance—a weaker condition).

Remark 2.1.3 It is straightforward to generalize the Gauss–Markov theorem to the complex case. The normal equations then become $A^HAx = A^Hb$. This has applications, e.g., in complex stochastic processes; see Miller [208, 1973].

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

$$s^2 = \|\widehat{r}\|_2^2/(m - n) \tag{2.1.7}$$

are uncorrelated with $\widehat{x}$, i.e., $\mathcal{V}(\widehat{r}, \widehat{x}) = 0$ and $\mathcal{V}(s^2, \widehat{x}) = 0$. An estimate of the variance of the linear functional c^Tx is given by $s^2(c^T(A^TA)^{-1}c)$. In particular, for the components $x_i = e_i^Tx$,

$$s^2(e_i^T(A^TA)^{-1}e_i) = s^2(A^TA)_{ii}^{-1}, \tag{2.1.8}$$

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

Gauss gave the first justification of the least squares principle as a statistical procedure in [111, 1809]. He assumed that the errors were uncorrelated and normally distributed with zero mean and equal variance. Later, Gauss gave the principle of least squares a sound theoretical foundation in two memoirs *Theoria Combinationis* [113, 1821] and [114, 1823]. Here the optimality of the least squares estimate is shown without assuming a particular distribution of the random errors.

The recently reprinted text by Lawson and Hanson [190, 1974] contains much interesting original material and examples, including Fortran programs. Numerical methods for solving least squares problems are treated in more detail in Björck [27, 1996] and Björck [28, 2004]. For results on accuracy and stability of the algorithm used the masterly presentation by Higham [162, 2002] is indispensable. Modern computational methods with examples from practical applications are featured in Hansen et al. [154, 2012].

For more detailed accounts of the invention of the principle of least squares the reader is referred to the excellent reviews by Placket [236, 1972], Stigler [275, 1981], [276, 1986], and Goldstine [125, 1977]. Markov may have clarified some implicit assumptions of Gauss in his textbook [204, 1912], but proved nothing new; see Placket [235, 1949] and [236, 1972]. An English translation of the memoirs of Gauss has been given by Stewart [112, 1995].

2.1.2 Projections and Geometric Characterization

The solution to the least squares problem $\min_x \|Ax - b\|_2$ has a geometric interpretation that involves an orthogonal projection, which we now introduce.

A matrix $P \in \mathbb{C}^{n\times n}$ such that $P^2 = P$ is called a **projector**. If P is a projector and $v \in \mathbb{C}^n$ an arbitrary vector, then the decomposition

$$v = Pv + (I - P)v \equiv v_1 + v_2 \tag{2.1.9}$$

is unique and $v_1 = Pv$ is the projection of v onto $\mathcal{R}(P)$. Furthermore, $Pv_2 = (P - P^2)v = 0$ and

$$(I - P)^2 = I + P^2 - 2P = I - P,$$

which shows that $I - P$ is a projection onto $\mathcal{N}(P)$. If λ is an eigenvalue of a projector P, then there is a nonzero vector x such that $Px = \lambda x$. But then $P^2x = \lambda Px = \lambda^2 x$ and it follows that $\lambda^2 = \lambda$. Hence, the eigenvalues of P are either 1 or 0 and the rank r of P equals the sum of its eigenvalues, i.e., $r = \text{trace}(P)$.

If P is Hermitian, $P^H = P$, then P is a **unitary projector** and

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

i.e., $v_2 \perp v_1$. Then we write $P^\perp = I - P$. It can be shown that the unitary projector P onto a given subspace $\mathcal{S}$ is unique, see Problem 2.1.2. If $\mathcal{S}$ is real, then P is real and called an orthogonal projector. If P is a unitary projector, then $\|v\|_2^2 = (v_1 + v_2)^H (v_1 + v_2) = \|v_1\|_2^2 + \|v_2\|_2^2$, which is the Pythagorean theorem. It follows that

$$\|Pv\|_2 \le \|v\|_2 \quad \forall\, v \in \mathbb{C}^m. \tag{2.1.10}$$

Equality holds for all vectors in $\mathcal{R}(P)$ and thus $\|P\|_2 = 1$. The converse is also true (but not trivial to prove); P is a unitary projection only if (2.1.10) holds. The following property follows immediately from the Pythagorean theorem.

Lemma 2.1.2 *Let $z = Px \in \mathcal{S}$ be the unitary projection of $x \in \mathbb{C}^n$ onto the subspace $\mathcal{S} \subset \mathbb{C}^n$. Then z is the point in $\mathcal{S}$ closest to x.*

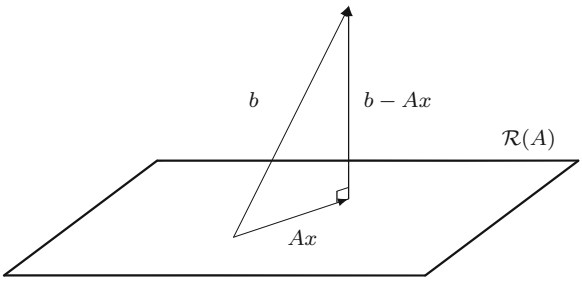


Fig. 2.1 Geometric characterization of least squares solutions

Let $U = (U_1\ U_2)$ be a unitary matrix and U_1 a basis for the subspace $\mathcal{S}$. Then the unitary projectors onto $\mathcal{S}$ and $\mathcal{S}^{\perp}$ are

$$P = U_1 U_1^H, \qquad P^{\perp} = U_2 U_2^H = I - U_1 U_1^H. \tag{2.1.11}$$

In particular, if q_1 is a unit vector, then $P^{\perp} = I - q_1 q_1^T$ is called an **elementary unitary projector**.

A least squares solution x decomposes the right-hand side b into two orthogonal components,

$$b = Ax + r, \qquad Ax \in \mathcal{R}(A), \quad r \in \mathcal{N}(A^H), \tag{2.1.12}$$

where $\mathcal{R}(A)$ is the column space of A. This simple geometrical characterization is illustrated Fig. 2.1.

If rank $(A) < n$, then the solution x to the normal equations is not unique. But the residual vector $r = b - Ax$ is always uniquely determined by the condition (2.1.12). Let $P_A b$ denote the unique orthogonal projection of b onto $\mathcal{R}(A)$. Then any solution to the consistent linear system

$$Ax = P_A b, \tag{2.1.13}$$

is a least squares solution. The unique solution of minimum norm $\|x\|_2$ is called the **pseudoinverse solution** and denoted by $x^{\dagger}$. It is a linear mapping of b, $x^{\dagger} = A^{\dagger} b$, where $A^{\dagger}$ is the **pseudoinverse** of A. A convenient characterization is:

Theorem 2.1.2 *The pseudoinverse solution of the least squares problem* $\min_x \|Ax - b\|_2$ *is uniquely characterized by the two conditions*

$$r = b - Ax \perp \mathcal{R}(A), \qquad x \in \mathcal{R}(A^H). \tag{2.1.14}$$

Proof Let x be any least squares solution and set $x = x_1 + x_2$, where $x_1 \in \mathcal{R}(A^H)$ and $x_2 \in \mathcal{N}(A)$. Then $Ax_2 = 0$, so $x = x_1$ is also a least squares solution. By the Pythagorean theorem, $\|x\|_2^2 = \|x_1\|_2^2 + \|x_2\|_2^2$, which is minimized when $x_2 = 0$. □

The solution to the linear least squares problem $\min_x \|Ax-b\|_2$, where $A \in \mathbb{R}^{m\times n}$, is fully characterized by the two equations $A^T y = 0$ and $y = b - Ax$. Together, these form a linear system of $n + m$ equations for x and the residual vector y:

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}, \tag{2.1.15}$$

with $c = 0$. System (2.1.15) is often called the **augmented system**. It is a special case of a saddle-point system and will be used in the perturbation analysis of least squares problems (Sect. 2.2.2) as well as in the iterative refinement of least squares solutions (Sect. 2.3.8).

The following theorem shows that the augmented system gives a unified formulation of two dual least squares problems.

Theorem 2.1.3 *If the matrix $A \in \mathbb{R}^{m\times n}$ has full column rank, then the augmented system* (2.1.15) *is nonsingular and gives the first-order conditions for the following two optimization problems:*

1. *The linear least squares problem:*

$$\min_x \tfrac{1}{2}\|Ax - b\|_2^2 + c^T x. \tag{2.1.16}$$

2. *The conditional least squares problem:*

$$\min_y \tfrac{1}{2}\|y - b\|_2 \quad \textit{subject to} \quad A^T y = c. \tag{2.1.17}$$

Proof The system (2.1.15) can be obtained by differentiating (2.1.16) to obtain $A^T(Ax - b) + c = 0$, and setting $y = r = b - Ax$. It can also be obtained by differentiating the Lagrangian function

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

of (2.1.17) and equating to zero. Here x is the vector of Lagrange multipliers. □

The standard least squares problem is obtained by taking $c = 0$ in (2.1.15). If we instead take $b = 0$, then by (2.1.17) the solution y is the **minimum-norm solution** of the system $A^T y = c$. We assume that A^T has full row rank so this system is consistent. The solution, which satisfies the normal equations

$$A^T A x = A^T b - c, \tag{2.1.18}$$

can be written

$$y = b - A(A^T A)^{-1}(A^T b - c) = P_A^{\perp} b + A(A^T A)^{-1} c, \tag{2.1.19}$$

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

Example 2.1.2 The heights $h(t_k)$ of a falling body at times $t_k = t_0 + k\Delta t$ lie on a parabola, i.e., the third differences of $h(t_k)$ will vanish. Let $\bar{h}_k$ be measured values of $h(t_k), k = 1{:}m$. Least squares approximations $h_k = \bar{h}_k - y_k$, where y_k are corrections, are found by solving the conditional least squares problem

$$\min \|y\|_2 \quad \text{subject to} \quad A^T y = c,$$

where $c = A^T \bar{h}$ and $(m = 7)$

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

Note that the corrected values satisfy $\bar{h}_k - y_k \perp \mathcal{R}(A)$. □

Lanczos [184, 1958] used the augmented system for computing pseudoinverse solutions of rectangular systems. At that time Lanczos was not aware of the connection with earlier work on the SVD and he developed the theory independently.

In many least squares problems the unknowns x can be naturally partitioned into two groups, i.e.,

$$\min_{x_1,x_2} \left\| b - \begin{pmatrix} A_1 & A_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_2, \quad x_1 \in \mathbb{R}^{n_1}, \quad x_2 \in \mathbb{R}^{n_2}, \tag{2.1.20}$$

where $A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \in \mathbb{R}^{m\times n}$. Assume that A has full column rank and let P_{A_1} be the orthogonal projection onto $\mathcal{R}(A_1)$. For any x_2, the residual vector $b - A_2x_2$ can be split into two orthogonal components,

$$r_1 = P_{A_1}(b - A_2x_2), \qquad r_2 = P_{A_1}^{\perp}(b - A_2x_2),$$

where $P_{A_1}^{\perp} = I - P_{A_1}$. Problem (2.1.20) then becomes

$$\min_{x_1,x_2} \left\| (r_1 - A_1x_1) + P_{A_1}^{\perp}(b - A_2x_2) \right\|_2. \tag{2.1.21}$$

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 (2.1.20), x_2 is the solution to the reduced least squares problem

$$\min_{x_2} \|P_{A_1}^{\perp}(b - A_2x_2)\|_2, \tag{2.1.22}$$

where the variables x_1 have been eliminated. Let x_2 be the solution to this reduced problem. Then x_1 can be found by solving the least squares problem

$$\min_{x_1} \|(b - A_2x_2) - A_1x_1\|_2. \tag{2.1.23}$$

One application of this is in **two-level** least squares methods. Here $n_1 \ll n_2$ and the projection matrix $P_{A_1}^{\perp}$ is computed by a direct method. The reduced problem (2.1.22) is then solved by an iterative least squares method.

2.1.3 The Method of Normal Equations

From the time of Gauss until the beginning of the computer age, least squares problems were solved by forming the normal equations and solving them by some variant of symmetric Gaussian elimination. We now discuss the details in the numerical implementation of this method. Throughout this section we assume that the matrix $A \in \mathbb{R}^{m\times n}$ has full column rank.

The first step is to compute the symmetric positive definite matrix $C = A^TA$ and the vector $d = A^Tb$. This requires $mn(n+1)$ flops and can be done in two different ways. In the inner product formulation $A = (a_1, a_2, \ldots, a_n)$ and b are accessed columnwise. We have

$$c_{jk} = (A^TA)_{jk} = a_j^Ta_k, \qquad d_j = (A^Tb)_j = a_j^Tb, \quad 1 \le j \le k \le n. \tag{2.1.24}$$

Since C is symmetric, it is only necessary to compute and store the $\frac{1}{2}n(n+1)$ elements in its lower (or upper) triangular part. Note that if $m \gg n$, then the number of required elements is much smaller than the number mn of elements in A. In this case forming the normal equations can be viewed as a data compression.

In (2.1.24) each column of A needs to be accessed many times. If A is held in secondary storage, a row oriented outer product algorithm is more suitable. Denoting the ith row of A by $\widetilde{a}_i^T$, $i = 1{:}m$, we have

$$C = \sum_{i=1}^{m} \widetilde{a}_i\widetilde{a}_i^T, \qquad d = \sum_{i=1}^{m} b_i\widetilde{a}_i. \tag{2.1.25}$$

Here A^TA is expressed as the sum of m matrices of rank one and A^Tb as a linear combination of the transposed rows of A. This approach has the advantage that just *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 the upper (or lower) triangular part of A^TA and A^Tb. This outer product form may also be preferable if A is sparse; see Problem 1.7.3.

To solve the normal equations, the Cholesky factorization

$$C = A^TA = R^TR, \quad R \in \mathbb{R}^{n\times n} \tag{2.1.26}$$

is computed by one of the algorithms given in Sect. 1.3.1. The least squares solution is then obtained by solving $R^TRx = d$, or equivalently the two triangular systems

$$R^Tz = d, \qquad Rx = z \tag{2.1.27}$$

Forming C and computing its Cholesky factorization requires (neglecting lower-order terms) $mn^2 + \frac{1}{3}n^3$ flops. Forming d and solving the two triangular systems requires $mn + n^2$ flops for each right-hand side.

If b is adjoined as the $(n+1)$st column to A, the corresponding cross product matrix is

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

The corresponding Cholesky factor has the form

$$\tilde{R} = \begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix},$$

where $R^T R = A^T A$, $R^T z = A^T b$, and $z^T z + \rho^2 = b^T b$. It follows that the least squares solution satisfies $Rx = z$. The residual vector satisfies $r + Ax = b$, where $Ax \perp r$. By the Pythagorean theorem it follows that $\rho = \|r\|_2$.

Example 2.1.3 In linear regression a model $y = \alpha + \beta x$ is fitted to a set of data (x_i, y_i), $i = 1{:}m$. The parameters α and βx are determined as the least squares solution to the system of equations

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

Using the normal equations and eliminating α gives the "textbook" formulas

$$\beta = \left(x^T y - m\,\bar{y}\bar{x}\right) \Big/ \left(x^T x - m\,\bar{x}^2\right), \qquad \alpha = \bar{y} - \beta\bar{x}. \tag{2.1.28}$$

where $\bar{x} = \frac{1}{m} e^T x$ and $\bar{y} = \frac{1}{m} e^T y$ are the mean values. The normal equations will be ill-conditioned if $x^T x \approx m\bar{x}^2$.

This is an example where ill-conditioning can be caused by an unsuitable formulation of the problem. A more accurate expression for β can be obtained by first subtracting mean values from the data. The model then becomes $y - \bar{y} = \beta(x - \bar{x})$ for which the normal equation matrix is diagonal (show this!). We obtain

$$\beta = \sum_{i=1}^{m} (y_i - \bar{y})(x_i - \bar{x}_i) \Big/ \sum_{i=1}^{m} (x_i - \bar{x})^2. \tag{2.1.29}$$

This more accurate formula requires two passes through the data. Accurate algorithms for computing sample means and variances are given by Chan et al. [47, 1983]. □

Preprocessing the data by subtracting the means is common practice in data analysis and called **centering** the data. This is equivalent to using the revised model

$$\begin{pmatrix} e & A \end{pmatrix} \begin{pmatrix} \xi \\ x \end{pmatrix} = b, \tag{2.1.30}$$

where as before $e = (1, \ldots, 1)^T$. Subtracting the means is equivalent to multiplying $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ with the projection matrix $(I - ee^T/m)$, giving

$$\overline{A} = A - \frac{1}{m} e(e^T A), \quad \overline{b} = b - \frac{e^T b}{m} u. \tag{2.1.31}$$

The solution is obtained by first solving the reduced least squares problem $\min_x \|\overline{A}x - \overline{b}\|_2$ and then setting

$$\xi = e^T(b - Ax)/m.$$

Note that this is just a special case of the two-block solution derived in Sect. 2.1.2.

A different choice of parametrization can often significantly reduce the condition number of the normal equation. In approximation problems one should try to use orthogonal, or nearly orthogonal, basis functions. For example, in fitting a polynomial $P(x)$ of degree k, an approach (often found in elementary textbooks) is to set

$$P(x) = a_0 + a_1 x + \cdots + a_k x^k$$

and solve the normal equations for the coefficients a_i, $i = 0{:}k$. Much better accuracy is obtained if $P(x)$ is expressed as a linear combination of a suitable set of orthogonal polynomials; see Forsythe [102, 1957] and Dahlquist and Björck [63, 2008], Sect. 4.5.6. If the elements in A and b are the original data, ill-conditioning cannot be avoided in this way. In Sects. 2.3 and 2.4 we consider methods for solving least squares problems using orthogonal factorizations.

By Theorem 2.7.2, in a Gauss–Markov linear model with error covariance matrix $\sigma^2 V$, the unbiased estimates of the covariance matrix of $\widehat{x}$ and σ^2 are given by

$$V_x = s^2(A^T V^{-1} A)^{-1}, \quad s^2 = \frac{1}{m-n} \widehat{r}^T V^{-1} \widehat{r}. \tag{2.1.32}$$

The estimated covariance matrix of the residual vector $\widehat{r} = b - A\widehat{x}$ is

$$\sigma^2 V_r = \sigma^2 A(A^T V^{-1} A)^{-1} A^T. \tag{2.1.33}$$

In order to assess the accuracy of the computed estimate of x it is often required to compute the matrix V_x or part of it. Let R be the upper triangular Cholesky factor of $A^T A$. For the standard linear model ($V = I, \sigma^2 = 1$)

$$V_x = (R^T R)^{-1} = R^{-1} R^{-T}. \tag{2.1.34}$$

The inverse matrix $S = R^{-T}$ is again lower triangular and can be computed in $n^3/3$ flops, as outlined in Sect. 1.2.6. The computation of the lower triangular part of the symmetric matrix $S^T S$ requires another $n^3/3$ flops.

Usually, only a selection of elements in V_x are wanted. The covariance of two linear functionals $f^T x$ and $g^T x$ is

$$\operatorname{cov}(f^T x, g^T x) = \sigma^2 f^T V_x g = \sigma^2 (f^T R^{-1})(R^{-T} g) = \sigma^2 u^T v. \tag{2.1.35}$$

Here u and v can be calculated by solving the two lower triangular systems $R^T u = f$ and $R^T v = g$ by forward substitution. The covariance of the components $x_i = e_i^T x$ and $x_j = e_j^T x$ of the solution is obtained by taking $f = e_i$ and $g = e_j$. In particular, the variances of x_i, $i = 1{:}n$, are

$$\operatorname{var}(x_i) = \sigma^2 \|u_i\|_2^2, \quad R^T u_i = e_i, \quad i = 1{:}n. \tag{2.1.36}$$

The vector u_i is the ith column of the lower triangular matrix $S = R^{-T}$. Thus, it has $i - 1$ leading zeros. For $i = n$ all components in u_i are zero except the last, which is $1/r_{nn}$.

If the error covariance matrix is correct, then the components of the **normalized residuals**

$$\widetilde{r} = \frac{1}{s} (\operatorname{diag} V_r)^{-1/2} \widehat{r} \tag{2.1.37}$$

should be uniformly distributed random quantities. In particular, the histogram of the entries of the residual should look like a bell curve.[1] The normalized residuals are often used to detect and identify bad data, which correspond to large components in $\widetilde{r}$.

Example 2.1.4 Least squares methods were first applied in astronomic calculations. In an early application Laplace [186, 1820] 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 Bouvard.[2] In the normal equations $A^T A x = A^T b$, the mass of Uranus is $(1 + x_1)/19504$ and the mass of Jupiter $(1 + x_2)/1067.09$, where the mass of the sun is taken as unity.

Working from these normal equations Laplace obtained the least squares solution $x_1 = 0.0895435$, $x_2 = -0.0030431$. This gave the mass of Jupiter and Uranus as a fraction of the mass of the sun as 1070.3 for Jupiter and 17,918 for Uranus. Bouvard also gave the square residual norm as $\|b - A\widehat{x}\|_2^2 = 31{,}096$. The covariance matrix

[1] The graph of the probability density function of a normally distributed random variable with mean value μ and variance σ^2 is given by $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}$. It is "bell" shaped and known as a bell curve.

[2] Alexis Bouvard (1767–1843), French astronomer and director of the Paris Observatory.

of the solution is $V_x = \sigma^2(R^TR)^{-1}$, and $s^2 = 31096/(129-6)$ is an unbiased estimate of σ^2. Laplace computed the first two diagonal elements

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

Taking square roots he obtained the standard deviations 0.072426 of x_1 and 0.002094 of x_2. From this Laplace concluded that the computed mass of of Jupiter is very reliable, while that of Uranus is not. He further could state that with a probability of $1 - 10^{-6}$ the error in the computed mass of Jupiter is less than one per cent. A more detailed discussion of Laplace's paper is given by Langou [185, 2009]. □

2.1.4 Stability of the Method of Normal Equations

We first determine the condition number of the matrix $C = A^HA$. Using the SVD of $A \in \mathbb{C}^{m\times n}$, we obtain

$$A^HA = V\begin{pmatrix}\Sigma & 0\end{pmatrix}(U^HU)\begin{pmatrix}\Sigma\\ 0\end{pmatrix}V^H = V\begin{pmatrix}\Sigma^2 & 0\\ 0 & 0\end{pmatrix}V^H. \tag{2.1.38}$$

This shows that $\sigma_i(C) = \sigma_i(A)^2$, $i = 1{:}n$, and

$$\kappa(C) = \frac{\sigma_{\max}(C)}{\sigma_{\min}(C)} = \frac{\sigma_{\max}(A)^2}{\sigma_{\min}(A)^2} = \kappa(A)^2.$$

Hence, by explicitly forming the normal equations, the condition number is squared.

By Theorem 1.2.3 the relevant condition number for a consistent linear system is $\kappa(A)$. Thus, at least for small residual problems, *the system of normal equations can be much worse conditioned than the least squares problem from which it originated.* This may seem surprising, since this method has been used since the time of Gauss. The explanation is that in hand calculations extra precision was used when forming and solving normal equations. As a rule of thumb, it suffices to carry twice the number of significant digits as in the entries in the data A and b.

The rounding errors performed in forming the matrix of the normal equations A^TA are not in general equivalent to small perturbations of the initial data matrix A. From the standard model for floating point computation the computed matrix $\bar{C} = fl(A^TA)$ satisfies

$$\bar{C} = fl(A^TA) = C + E, \quad |E| < \gamma_m|A|^T|A|. \tag{2.1.39}$$

where (see Lemma 1.4.2) $|\gamma_m| < m\mathbf{u}/(1-m\mathbf{u})$ and $\mathbf{u}$ is the unit roundoff. A similar estimate holds for the rounding errors in the computed vector A^Tb. Hence, even the exact solution of the computed normal equations is not equal to the exact solution to a problem where the data A and b have been perturbed by small amounts. In other words, although the method of normal equations often gives a satisfactory result

it *cannot be backward stable*. The following example illustrates how *significant information in the data may be lost*.

Example 2.1.5 Läuchli [188, 1961]: Consider the system $Ax = b$, where

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

We have, exactly

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

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

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 $A^T A$ 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. □

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. For $A \in \mathbb{R}^{n\times n}$, using the error bound given in Theorem 1.4.4 a perturbation analysis shows that provided $2n^{3/2}\mathbf{u}\kappa(A)^2 < 0.1$, an upper bound for the error in the computed solution $\bar{x}$ is

$$\|\bar{x} - x\|_2 \le 2.5n^{3/2}\mathbf{u}\kappa(A)^2\|x\|_2. \tag{2.1.40}$$

In Sect. 1.2.8 we studied 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 $\min_x \|Ax - b\|_2$ scaling the rows is not allowed. The row scaling is determined by the error variances, and changing this will change the problem. However, we are free to scale the columns of A. Setting $x = Dx'$ gives the normal equations

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

Hence, this corresponds to a *symmetric scaling* of rows and columns in $A^T A$. The Cholesky algorithm is numerically invariant under such a scaling; cf. Theorem 1.2.8. *This means that even if this scaling is not carried out explicitly, the rounding error estimate* (2.1.40) *for the computed solution* $\bar{x}$ *holds for all* $D > 0$, *and we have*

$$\|D(\bar{x} - x)\|_2 \le 2.5n^{3/2}\mathbf{u}\kappa^2(AD)\|Dx\|_2.$$

(Note 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

$$\kappa'(A^TA) = \min_{D>0} \kappa(DA^TAD). \tag{2.1.41}$$

By (2.2.33), choosing D so that all columns in AD have equal 2-norm will overestimate the minimum condition number by at most a factor n.

Example 2.1.6 Column 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. The matrix $A \in \mathbb{R}^{21\times 6}$ with elements

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

arises when fitting a fifth degree polynomial $p(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_5 t^5$ to observations at points $t_i = 0, 1, \ldots, 20$. The condition numbers are

$$\kappa(A^TA) = 4.10 \cdot 10^{13}, \qquad \kappa(DA^TAD) = 4.93 \cdot 10^{6},$$

where D is the column scaling in (2.2.33). Here scaling reveals that the condition number is seven orders of magnitude smaller than the first estimate. □

Iterative refinement is a simple way to improve the accuracy of a solution $\bar{x}$ computed by the method of normal equations; see Sect. 1.4.6.

Algorithm 2.1.1 (*Iterative Refinement*) *Set* $x_1 = \bar{x}$, *and for* $s = 1, 2, \ldots$ *until convergence do*

1. *Compute the residual* $r_s = b - Ax_s$.
2. *Solve for the correction* $R^T R\delta x_s = A^T r_s$.
3. *Add correction* $x_{s+1} = x_s + \delta x_s$.

Information lost by forming A^TA and A^Tb is recovered in computing the residual. Each refinement step requires only one matrix-vector multiplication each with A and A^T and the solution of two triangular systems. Note that the first step ($i = 0$) is identical to solving the normal equations. The errors will initially be reduced by a factor

$$\bar{\rho} = c\mathbf{u}\kappa'(A^TA), \tag{2.1.42}$$

even if no extra precision is used in Step 1. (Note that this is true even when no scaling of the normal equations has been performed!) Here c depends on the dimensions m, n, but is of moderate size. Several steps of refinement may be needed to get good accuracy.

Example 2.1.7 If $c \approx 1$, the error will be reduced to a backward stable level in p steps if $\kappa(A) \leq \mathbf{u}^{-p/(2p+1)}$. (As remarked before, $\kappa(A)$ is the condition number for a small residual problem.) For example, with $\mathbf{u} = 10^{-16}$, the maximum value of $\kappa(A)$ 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 on QR factorization described in Sect. 2.3 should be preferred. □

Exercises

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

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

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

2.1.2 Let $S \subseteq \mathbb{R}^n$ be a subspace, and let P_1 and P_2 be orthogonal projections onto $S = \mathcal{R}(P_1) = \mathcal{R}(P_2)$. Show that for any $z \in \mathbb{R}^n$,

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

and hence $P_1 = P_2$, i.e., the orthogonal projection onto S is unique.

2.1.3 Let $A \in \mathbb{R}^{m\times n}$ 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 $y = Ax$.

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

$$A^T A = \begin{pmatrix} A_1^T A_1 & A_1^T A_2 \\ A_2^T A_1 & A_2^T A_2 \end{pmatrix}$$

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

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

where $P_1 = A_1 (A_1^T A_1)^{-1} A_1^T$ 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 for x_2 and then for x_1:

$$\min_{x_2} \|(I - P_1)(A_2 x_2 - b)\|_2^2, \qquad \min_{x_1} \|A_1 x_1 - (b - A_2 x_2)\|_2^2.$$

2.1.5 (Stigler [275, 1981]) 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 north pole to the equator along a meridian arc through Paris. The following famous data obtained in a 1795 survey

consist of four measured sections of an arc from Dunkirk to Barcelona. For each section the length S of the arc (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 that

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

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

2.1.6 The Hald cement data (see [145, 1952]), p. 647, is used in several books and papers as an example of regression analysis. The right-hand side is the heat evolved in cement during hardening, and the explanatory variables are four different ingredients of the mix and a constant term. There are $m = 13$ observations:

$$A = (e,\ A_2) = \begin{pmatrix} 1 & 7 & 26 & 6 & 60 \\ 1 & 1 & 29 & 15 & 52 \\ 1 & 11 & 56 & 8 & 20 \\ 1 & 11 & 31 & 8 & 47 \\ 1 & 7 & 52 & 6 & 33 \\ 1 & 11 & 55 & 9 & 22 \\ 1 & 3 & 71 & 17 & 6 \\ 1 & 1 & 31 & 22 & 44 \\ 1 & 2 & 54 & 18 & 22 \\ 1 & 21 & 47 & 4 & 26 \\ 1 & 1 & 40 & 23 & 34 \\ 1 & 11 & 66 & 9 & 12 \\ 1 & 1 & 68 & 8 & 12 \end{pmatrix}, \quad b = \begin{pmatrix} 78.5 \\ 74.3 \\ 104.3 \\ 87.6 \\ 95.9 \\ 109.2 \\ 102.7 \\ 72.5 \\ 93.1 \\ 115.9 \\ 83.8 \\ 113.3 \\ 109.4 \end{pmatrix}. \tag{2.1.43}$$

(a) Solve the least squares problem $\|Ax - b\|_2$ by the method of normal equations. Show that $\kappa(A) \approx 3.245 \cdot 10^3$.

(b) The first column of ones has been added to extract the mean values. Set $x = (\xi,\ y)$ and compute $B = A_2 - ep^T$ and $c = b - \beta e$, where $p = (e^T A_2)/m$, $\beta = e^T b/m$, to obtain the reduced problem $\min_x \|By = c\|_2$. Show that this problem is well conditioned: $\kappa(B) = 23.0$. The intercept variable ξ can then be obtained from $\xi + p^T y = \beta$.

(c) Show that for this problem, normalizing the column of B to have unit length only decreases the condition number a negligible amount, $\kappa(BD) = 19.6$.

2.2 Least Squares Problems and the SVD

The SVD, introduced in Sect. 1.1.9, is a powerful tool for both analyzing and solving linear least squares problems. The reason is that the unitary matrices that transform $A \in \mathbb{C}^{m\times n}$ to diagonal form do not change the ℓ_2-norm. The SVD also has a key

role in many algorithms for approximating a given matrix with a matrix of lower rank. This has many important applications, e.g., in data compression and model reduction. In this section we collect a number of results that will be used extensively in the following.

One of the most important properties of the singular values is that they are characterized by an extremal property. Related to this is that the best approximation of a matrix $A \in \mathbb{C}^{m\times n}$ by a matrix of lower rank is obtained by truncating the SVD expansion of A. This is the basis for numerous applications.

2.2.1 SVD and the Pseudoinverse

We have the following fundamental result.

Theorem 2.2.1 *Consider the least squares problem* $\min_x \|Ax - b\|_2$, *where* $A \in \mathbb{C}^{m\times n}$ *and* $r = \text{rank}\,(A) \le \min(m, n)$. *Let* A *have the* SVD

$$A = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^H \\ V_2^H \end{pmatrix}, \tag{2.2.1}$$

where U_1 *and* V_1 *have* r *columns and the diagonal matrix* $\Sigma_1 > 0$. *Then the unique pseudoinverse solution is*

$$x = V_1 \Sigma_1^{-1} U_1^H b. \tag{2.2.2}$$

Proof Setting $x = Vz$, and using the unitary invariance of the spectral norm, we have

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

where $c_i = U_i^H b$, $i = 1, 2$. The residual norm will attain its minimum value equal to $\|c_2\|_2$ for $z_1 = \Sigma_1^{-1} c_1$ and z_2 arbitrary. Obviously the choice $z_2 = 0$ minimizes $\|x\|_2 = \|Vz\|_2 = \|z\|_2$. □

Note that Theorem (2.2.1) applies to the solution of both overdetermined and underdetermined linear systems. The pseudoinverse of A is

$$A^\dagger = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} U_1^H \\ U_2^H \end{pmatrix} = V_1 \Sigma_1^{-1} U_1^H \in \mathbb{C}^{n\times m}, \tag{2.2.3}$$

which maps b to x and represents the SVD of $A^\dagger$. The pseudoinverse solution (2.2.3) can also be written

$$x = \sum_{i=1}^{r} \frac{c_i}{\sigma_i} v_i, \quad c_i = u_i^H b. \tag{2.2.4}$$

Hence, x lies in a subspace of dimension less than or equal to r. Note that for computing the pseudoinverse solution, we only need to compute the "thin" SVD, i.e., the nonzero singular values, the matrix V_1 and the vector $U_1^H b$. Methods for computing the SVD are described in Sects. 3.5.3 and 3.6.3.

The pseudoinverse is relevant when it is reasonable to use the 2-norm. The same is true for orthogonal transformations, the SVD, and even the notion of symmetric and Hermitian matrices. If another inner product is more relevant, then the definition of pseudoinverse should be modified accordingly.

The following definition generalizes the condition number (1.2.45) of a square nonsingular matrix.

Definition 2.2.1 Let $A \in \mathbb{R}^{m\times n}$ have rank $r > 0$ and singular values equal to $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Then the 2-norm condition number of A is

$$\kappa_2(A) = \|A\|_2 \|A^\dagger\|_2 = \sigma_1/\sigma_r. \tag{2.2.5}$$

From (2.2.4) it follows that for a particular right-hand side b the singular values corresponding to $u_i^T b = 0$ do not enter in the solution. Therefore, the effective condition number of A with respect to b is obtained by taking the maximum and minimum in (2.2.5) only over singular values for which $c_i \neq 0$. This concept has been made more precise by Chan and Foulser [45, 1988].

The orthogonal projections onto the four fundamental subspaces (1.1.93)–(1.1.94) of A have the following simple expressions in terms of the pseudoinverse:

$$P_A = AA^\dagger = U_1 U_1^H, \qquad P_A^\perp = I - AA^\dagger = U_2 U_2^H, \tag{2.2.6}$$

$$P_{A^H} = A^\dagger A = V_1 V_1^H, \qquad P_{A^H}^\perp = I - A^\dagger A = V_2 V_2^H. \tag{2.2.7}$$

The matrix $A^\dagger$ is often called the **Moore–Penrose inverse**. Moore [210, 1920] developed the concept of the general reciprocal, later rediscovered by Bjerhammar [21, 1951]. Penrose [231, 1955] gave an elegant algebraic characterization and showed that $X = A^\dagger$ is uniquely determined by the four **Penrose conditions**:

$$(1)\quad AXA = A, \qquad (2)\quad XAX = X, \tag{2.2.8}$$

$$(3)\quad (AX)^H = AX, \qquad (4)\quad (XA)^H = XA. \tag{2.2.9}$$

It can be directly verified that $X = A^\dagger$ given by (2.2.3) 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. The following properties of the pseudoinverse easily follow from (2.2.3).

Theorem 2.2.2

1. $(A^\dagger)^\dagger = A$;
2. $(A^\dagger)^H = (A^H)^\dagger$;
3. $(\alpha A)^\dagger = \alpha^{-1} A^\dagger$, $\alpha \neq 0$;
4. $(A^H A)^\dagger = A^\dagger (A^\dagger)^H$;
5. *if U and V are unitary, then* $(UAV^H)^\dagger = VA^\dagger U^H$;
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* ($AA^H = A^HA$), *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* $\operatorname{trace}(A^\dagger A)$.

Although some properties of the usual inverse can be extended to the pseudoinverse, it does not share all properties of the ordinary inverse. For example, in general

$$(AB)^\dagger \neq B^\dagger A^\dagger, \qquad AA^\dagger \neq A^\dagger A. \tag{2.2.10}$$

The following theorem gives a useful *sufficient* conditions for the relation $(AB)^\dagger = B^\dagger A^\dagger$ to hold.

Theorem 2.2.3 *If* $A \in \mathbb{C}^{m\times r}$, $B \in \mathbb{C}^{r\times n}$, *and* $\operatorname{rank}(A) = \operatorname{rank}(B) = r$, *then*

$$(AB)^\dagger = B^H(BB^H)^{-1}(A^HA)^{-1}A^H = B^\dagger A^\dagger. \tag{2.2.11}$$

Proof The square matrices A^HA and BB^H have rank r and hence are nonsingular. The result is verified by showing the Penrose conditions are satisfied. □

In some contexts it is sufficient to use a weaker form of generalized inverse than the pseudoinverse. Any matrix A^- satisfying the first Penrose condition $AA^-A = A$ is called an **inner inverse** or {1}-inverse. If it satisfies the second condition $A^-AA^- = A^-$ 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

$$x = A^-b + (I - A^-A)y, \quad y \in \mathbb{C}^n. \tag{2.2.12}$$

For any {1}-inverse of A

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

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

Let $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$. Then $\|Ax - b\|_2$ is minimized when x satisfies the normal equations $A^HAx = A^Hb$. Suppose that a generalized inverse A^- satisfies

$$(AA^-)^H = AA^-. \tag{2.2.13}$$

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

$$A^H = (AA^-A)^H = A^HAA^-,$$

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

The following dual result holds also: If A^- is a generalized inverse, and $(A^-A)^H = A^-A$, then A^-A is the orthogonal projector onto $\mathcal{N}(A)$ 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

$$x = A^H z, \quad AA^H z = b.$$

For a minimum-norm inverse A^- we have $A^H = (AA^-A)^H = A^-AA^H$. Hence, $x = A^H z = A^-(AA^H)z = A^-b$, which shows that $x = A^-b$ is the solution of smallest norm. Conversely, let $A^- \in \mathbb{C}^{n\times m}$ be such that, whenever $Ax = b$ has a solution, then $x = A^-b$ is a minimum-norm solution. Then A^- is a minimum-norm inverse.

A good introduction to generalized inverses is given by Ben-Israel and Greville [16, 1976]. A more complete and thorough treatment is given in the monograph [17, 2003] by the same authors. A collection of papers on this subject appeared in Nashed [214, 1976]. Generalized inverses should be used with caution, because the notation tends to hide intrinsic computational difficulties associated with nearly rank-deficient matrices.

2.2.2 Perturbation Analysis

We first derive some perturbation bounds for the pseudoinverse of a matrix $A \in \mathbb{C}^{m\times n}$. 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 $A(\epsilon) = \begin{pmatrix} \sigma & 0 \\ 0 & \epsilon \end{pmatrix}$, for which rank $(A) = 2$, if $\epsilon \neq 0$, but rank $(A(0)) = 1$ and

$$\|(A + E)^\dagger - A^\dagger\|_2 = |\epsilon|^{-1} = 1/\|E\|_2.$$

This example shows that formulas derived by operating formally with pseudoinverses may have no meaning numerically.

The perturbations for which the pseudoinverse is well behaved can be characterized by the condition

$$\text{rank}\,(A) = \text{rank}\,(B) = \text{rank}\,(P_{\mathcal{R}(A)} B P_{\mathcal{R}(A^H)}). \tag{2.2.14}$$

The matrix B is said to be an **acute perturbation** of A if (2.2.14) holds; see Stewart [265, 1977].

Theorem 2.2.4 *If* rank $(A + E) =$ rank $(A) = r$ *and* $\eta = \|A^\dagger\|_2\|E\|_2 < 1$, *then*

$$\|(A + E)^\dagger\|_2 \le \frac{1}{1 - \eta}\|A^\dagger\|_2. \tag{2.2.15}$$

Proof From the assumption and Theorem 2.2.8 it follows that

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

which implies (2.2.15). □

Let $A, B \in \mathbb{C}^{m\times n}$, 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 identity** holds:

$$B^{\dagger} - A^{\dagger} = -B^{\dagger}EA^{\dagger} + (B^H B)^{\dagger}E^H P_{\mathcal{N}(A^H)} + P_{\mathcal{N}(B)}E^H(AA^H)^{\dagger} \tag{2.2.16}$$

(see [291, 1969] and [292, 1973]). This identity can be proved by expressing the projections in terms of pseudoinverses using the relations in (2.2.6). Observe that if A has full column rank, then the second term vanishes; if B has full row rank, then the third term vanishes. If A and B have full column rank, we obtain from Wedin's identity

$$B^{\dagger}A - I = (B^{\dagger} - A^{\dagger})A = -B^{\dagger}E + (B^H B)^{\dagger}E^H P_{\mathcal{N}(A^H)}. \tag{2.2.17}$$

Setting $B = A(\alpha) = A + \alpha E$, where α is a scalar parameter, we have $E = dA/d\alpha$. Letting $\alpha \to 0$ and assuming that $A(\alpha)$ has constant local rank, the following formula for the derivative of the pseudoinverse $A^{\dagger}(\alpha)$ is obtained from Wedin's identity:

$$\frac{dA^{\dagger}}{d\alpha} = -A^{\dagger}\frac{dA}{d\alpha}A^{\dagger} + (A^H A)^{\dagger}\frac{dA^H}{d\alpha}P_{\mathcal{N}(A^H)} + P_{\mathcal{N}(A)}\frac{dA^H}{d\alpha}(AA^H)^{\dagger}. \tag{2.2.18}$$

This formula is due to Wedin [291, 1969].

Theorem 2.2.5 *If* $B = A + E$ *and* rank (B) = rank (A), *then*

$$\|B^{\dagger} - A^{\dagger}\| \leq \mu\|B^{\dagger}\|\,\|A^{\dagger}\|\,\|E\|, \tag{2.2.19}$$

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

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

Proof For the spectral norm, see Wedin [292, 1973]. The result that $\mu = 1$ for the Frobenius norm is due to van der Sluis and Veltkamp [258, 1979]. □

We now give a first-order perturbation analysis for the least squares problem when A has full column rank. The least squares solution x and the corresponding residual $r = b - Ax$ satisfy the augmented system (see (2.1.15))

$$\begin{pmatrix} I & A \\ A^H & 0 \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}. \tag{2.2.20}$$

If rank $(A) = n$, this is a square nonsingular linear system. Hence, the same technique as in Sect. 1.2.7 can be used for the perturbation analysis. Denote the perturbed data by $A + \delta A$, $b + \delta b$, and $c + \delta c$ and assume that δA is sufficiently small so that rank $(A + \delta A) = n$. Let the perturbed solution be $x + \delta x$ and $r + \delta r$. The perturbed solution satisfies the augmented system

$$\begin{pmatrix} I & A + \delta A \\ (A + \delta A)^H & 0 \end{pmatrix} \begin{pmatrix} r + \delta r \\ x + \delta x \end{pmatrix} = \begin{pmatrix} b + \delta b \\ c + \delta c \end{pmatrix}. \tag{2.2.21}$$

Subtracting the unperturbed equations and neglecting second-order quantities gives

$$\begin{pmatrix} I & A \\ A^H & 0 \end{pmatrix} \begin{pmatrix} \delta r \\ \delta x \end{pmatrix} = \begin{pmatrix} \delta b - \delta A x \\ \delta c - \delta A^H r \end{pmatrix}. \tag{2.2.22}$$

From the Schur–Banachiewicz formula (see Sect. 1.1.6) it follows that the inverse of the matrix in this system is

$$\begin{aligned} \begin{pmatrix} I & A \\ A^H & 0 \end{pmatrix}^{-1} &= \begin{pmatrix} (I - A(A^H A)^{-1} A^H) & A(A^H A)^{-1} \\ (A^H A)^{-1} A^H & -(A^H A)^{-1} \end{pmatrix} \\ &= \begin{pmatrix} P_A^{\perp} & (A^{\dagger})^H \\ A^{\dagger} & -A^{\dagger}(A^{\dagger})^H \end{pmatrix}. \end{aligned} \tag{2.2.23}$$

We obtain

$$\delta x = A^{\dagger}(\delta b - \delta A\, x) + A^{\dagger}(A^{\dagger})^H(\delta c - \delta A^H r), \tag{2.2.24}$$

$$\delta r = P_A^{\perp}(\delta b - \delta A x)(A^{\dagger})^H(\delta c - \delta A^H r). \tag{2.2.25}$$

Assuming that the perturbations δA and δb satisfy the componentwise bounds

$$|\delta A| \le \omega E, \qquad |\delta b| \le \omega f, \qquad |\delta c| \le \omega g, \tag{2.2.26}$$

and substituting into (2.2.24)–(2.2.25) yields the componentwise bounds

$$|\delta x| \lessapprox \omega \left(|A^{\dagger}|(f + E|x|) + |A^{\dagger}||(A^{\dagger})^H|\,(g + E^H |r|) \right), \tag{2.2.27}$$

$$|\delta r| \lessapprox \omega \left(|I - AA^{\dagger}|(f + E|x|) + |(A^{\dagger})^H|\,(g + E^H |r|) \right), \tag{2.2.28}$$

where terms of order $O(\omega^2)$ have been neglected.

By setting $g = 0$, componentwise perturbation bounds for the linear least squares problem $\min_x \|Ax - b\|_2$ are obtained. Note that if $Ax = b$ is consistent, i.e., $r = 0$, then the bound for $|\delta x|$ is identical to that obtained for a square nonsingular linear

system in Sect. 1.2.7. A perturbation bound for the minimum-norm solution x of a consistent linear system is obtained by setting $f = 0$. Taking norms in (2.2.24) and (2.2.25) and using

$$\|A^{\dagger}\|_2 = \|(A^{\dagger})^H\|_2 = 1/\sigma_n, \qquad \|(A^H A)^{-1}\|_2 = 1/\sigma_n^2, \qquad \|P_A^{\perp}\|_2 = 1,$$

gives normwise bounds for the least squares problem.

Theorem 2.2.6 *Consider the linear least squares problem* $\min \|Ax - b\|_2$, *with* $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^m$ *and* $\operatorname{rank}(A) = n$. *Let* $A + \delta A$, $b + \delta b$ *be perturbed data and assume that* δA *is sufficiently small so that* $\operatorname{rank}(A+\delta A) = n$. *Denote the perturbed solution by* $x + \delta x$ *and* $r + \delta r$. *If second-order terms can be neglected, then*

$$\|\delta x\|_2 \lessapprox \frac{1}{\sigma_n}\|\delta b\|_2 + \frac{1}{\sigma_n}\|\delta A\|_2\left(\|x\|_2 + \frac{1}{\sigma_n}\|r\|_2\right), \tag{2.2.29}$$

$$\|\delta r\|_2 \lessapprox \|\delta b\|_2 + \|\delta A\|_2\left(\|x\|_2 + \frac{1}{\sigma_n}\|r\|_2\right). \tag{2.2.30}$$

Note that if $r = P_A^{\perp} b \neq 0$, then a term proportional to σ_n^{-2} is present in the bound for $\|\delta x\|_2$. A more refined perturbation analysis (see Wedin [292, 1973]) shows that if

$$\eta = \|A^{\dagger}\|_2\|\delta A\|_2 < 1,$$

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

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \kappa_{\mathrm{LS}}\frac{\|\delta A\|_2}{\|A\|_2}, \quad \kappa_{\mathrm{LS}} = \kappa(A)\Big(1 + \frac{\|r\|_2}{\sigma_n\|x\|_2}\Big), \tag{2.2.31}$$

which is an upper bound for the normwise relative perturbation of the least squares solution. Note that κ_{LS} depends not only on A but also on $r = P_A^{\perp} b$. If $\|r\|_2 \ll \sigma_n\|x\|_2$, then $\kappa_{\mathrm{LS}} \approx \kappa(A)$, but if $\|r\|_2 > \sigma_n\|x\|_2$, then the second term in (2.2.31) dominates. A lower bound given by Malyshev [203, 2003] shows that κ_{LS} overestimates the true condition number at most by a factor of $\sqrt{2}$.

As suggested by van der Sluis [257, 1975], the last term in (2.2.31) can be rewritten as

$$\frac{\|r\|_2}{\sigma_n\|x\|_2} = \tan(\theta)\frac{\|Ax\|_2}{\sigma_n\|x\|_2}, \quad \tan(\theta) = \frac{\|r\|_2}{\|Ax\|_2},$$

where θ is the angle between the right-hand side and the subspace $\mathcal{R}(A)$.

Example 2.2.1 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 $x = (1,\ 0)^T$, $r = (0,\ 0,\ 1)^T$ and

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

For this right-hand side, $\|x\|_2 = \|r\|_2$ and $\kappa_{\mathrm{LS}} = 1/\delta + 1/\delta^2 \approx \kappa^2(A)$. This is reflected in the size of δx. For $\alpha = \delta$, a short calculation shows that $\|r\|_2/\|x\|_2 = \delta$ and $\kappa_{\mathrm{LS}} = 2/\delta$. The same perturbation δA now gives

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

It should be stressed that for the normwise perturbation bounds in Theorem 2.2.6 to be realistic, A and 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 $A = (a_1, \ldots, a_n)$ have widely differing norms. The residual vector r is independent of the column scaling of A, since we can write

$$r = b - Ax = b - AD(D^{-1}x).$$

A much improved error estimate may be obtained by applying (2.2.29) to the scaled problem with D chosen so that the columns of AD have unit length:

$$D = \mathrm{diag}(1/\|a_1\|_2, \ldots, 1/\|a_n\|_2). \tag{2.2.32}$$

From Theorem 1.2.5, p. 56 it follows that if $A \in \mathbb{R}^{m\times n}$ has full column rank, this scaling comes within a factor of $\sqrt{n}$ of achieving the minimum value of $\kappa_2(A)$, i.e.,

$$\kappa_2(A) \le \sqrt{n} \min_{D\in\mathcal{D}} \kappa_2(AD). \tag{2.2.33}$$

If the *rows* in A differ widely in norm, then (2.2.31) may also considerably overestimate the perturbation in x. *But scaling the rows in A is not allowed*, because in the Gauss–Markov model the scaling is determined by the covariance matrix of the error. Methods for solving problems with widely different row norms are discussed in Sect. 2.7.1.

The perturbation theory of pseudoinverses was developed by Wedin [292, 1973] see also Stewart [265, 1977]. Björck [22, 1967] gave nearly optimal normwise perturbation bounds for the solution and residual. componentwise bounds are given in Björck [25, 1991]. As pointed out by Grcar [140, 2010], several other bounds in the literature can overestimate the true condition number by as much asa fac-

tor $\kappa(A)$. Notable recent works on the subject are due to Gratton [139, 1996] and Malyshev [203, 2003].

2.2.3 SVD and Matrix Approximation

In the proof of Theorem 1.1.6 we showed that the largest singular value of A can 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 2.2.7 *Let $A \in \mathbb{C}^{m\times n}$ have singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$, where $p = \min(m, n)$. Then, if $\mathcal{S}$ denotes a linear subspace of $\mathbb{C}^n$, one has that*

$$\sigma_i = \max_{\dim(\mathcal{S})=i} \min_{\substack{x\in\mathcal{S} \\ \|x\|_2=1}} \|Ax\|_2 \tag{2.2.34}$$

$$= \min_{\dim(\mathcal{S})=p-i+1} \max_{\substack{x\in\mathcal{S} \\ \|x\|_2=1}} \|Ax\|_2, \quad i = 1{:}p. \tag{2.2.35}$$

Proof The result follows from a relationship that will be shown in Theorem 3.5.2 and the corresponding result for the Hermitian eigenvalue problem; see Theorem 3.2.7 (Fischer's theorem), p. 443. □

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

$$\max_i |\sigma_i - \tau_i| \leq \|A - B\|_2, \tag{2.2.36}$$

$$\sum_{i=1}^{p} |\sigma_i - \tau_i|^2 \leq \|A - B\|_F^2. \tag{2.2.37}$$

Proof See Stewart [263, 1973], pp. 321–322. □

By inequality (2.2.36), 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 results 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 Sect. 2.4.1.

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 2.2.9 *The ordered singular values σ_i of $A \in \mathbb{C}^{m\times n}$ interlace the ordered singular values $\widehat{\sigma}_i$ of the bordered matrix $\widehat{A} = \begin{pmatrix} A, \\ u^H \end{pmatrix}$ as follows:*

$$\widehat{\sigma}_1 \geq \sigma_1 \geq \widehat{\sigma}_2 \cdots \geq \widehat{\sigma}_m \geq \sigma_m \geq \widehat{\sigma}_{m+1}, \quad m < n,$$
$$\widehat{\sigma}_1 \geq \sigma_1 \geq \widehat{\sigma}_2 \cdots \geq \widehat{\sigma}_{n-1} \geq \sigma_{n-1} \geq \widehat{\sigma}_n \geq \sigma_n, \quad m \geq n.$$

A similar result holds when A is bordered by a column.

Proof The theorem is a consequence of Cauchy's interlacing theorem for Hermitian matrices to be proved later in Chap. 3; see Theorem 3.2.9, p. 444. This says that the eigenvalues of the leading principal minor of order $n-1$ of a Hermitian matrix B interlace those of B. Since

$$\begin{pmatrix} A^H \\ u^H \end{pmatrix} \begin{pmatrix} A & u \end{pmatrix} = \begin{pmatrix} A^H A & A^H u \\ u^H A & u^H u \end{pmatrix},$$
$$\begin{pmatrix} A \\ v^H \end{pmatrix} \begin{pmatrix} A^H & v \end{pmatrix} = \begin{pmatrix} AA^H & Av \\ v^H A^H & v^H v \end{pmatrix},$$

the result follows from the observation that the singular values of A are the positive square roots of the eigenvalues of $A^H A$ and AA^H. □

Lemma 2.2.1 *Let $A \in \mathbb{C}^{m\times n}$ and $B_k = X_k Y_k^H$, where $X_k, Y_k \in \mathbb{C}^{m\times k}$. Then* $\operatorname{rank}(B_k) \leq k < \min\{m, n\}$ *and*

$$\sigma_1(A - B_k) \geq \sigma_{k+1}(A), \tag{2.2.38}$$

where $\sigma_i(\cdot)$ denotes the ith singular value of its argument.

Proof Let v_i, $i = 1{:}n$ be the right singular vectors of A. Since $\operatorname{rank}(Y) = k < n$, there is a vector $v = c_1 v_1 + \cdots + c_{k+1} v_{k+1}$ such that $\|v\|_2^2 = c_1^2 + \cdots + c_{k+1}^2$ and $Y^H v = 0$. It follows that

$$\begin{aligned} \sigma_1^2(A - B_k) &\geq v^H (A - B_k)^H (A - B_k) v = v^H A^H A v \\ &= |c_1|^2 \sigma_1^2 + \cdots + |c_{k+1}|^2 \sigma_{k+1}^2 \geq \sigma_{k+1}^2. \end{aligned}$$

□

Theorem 2.2.10 *Let $A = B + C$, where $B, C \in \mathbb{C}^{m\times n}$, $m \geq n$, have ordered singular values $\sigma_1(B) \geq \cdots \geq \sigma_n(B)$ and $\sigma_1(C) \geq \cdots \geq \sigma_n(C)$, respectively. Then the ordered singular values of A satisfy*

$$\sigma_{i+j-1}(A) \leq \sigma_i(B) + \sigma_j(C). \tag{2.2.39}$$

Proof For $i = j = 1$ we have

$$\sigma_1(A) = u_1^H A v_1 = u_1^H B v_1 + u_1^H C v_1 \leq \sigma_1(B) + \sigma_1(C).$$

Now let B_{i-1} and C_{i-1} denote the SVD expansions truncated to $i-1$ terms. Then $\sigma_1(B - B_{i-1}) = \sigma_i(B)$ and $\sigma_1(C - C_{i-j}) = \sigma_j(C)$. Moreover, $\operatorname{rank}(B_{i-1} + C_{i-1}) \leq i + j - 2$. From these facts and Lemma 2.2.1 it follows that

$$\begin{aligned}\sigma_i(B) + \sigma_j(C) &= \sigma_1(B - B_{i-1}) + \sigma_1(C - C_{j-1}) \\ &\geq \sigma_1(A - B_{i-1} + C_{j-1}) \geq \sigma_{i+j-1}(A). \qquad \square\end{aligned}$$

The best approximation of a matrix $A \in \mathbb{C}^{m\times n}$ by a matrix of lower rank is obtained by truncating the SVD expansion of A. It was proved in 1936 by Eckart and Young [79, 1936] for the Frobenius norm. Mirsky [209, 1960] proved it for all unitarily invariant norms, including the Schatten norms; see (1.1.96). This is one of the most important properties of the SVD and is the basis for numerous applications. For example, in signal processing, the matrix A is derived from data constituting a noisy signal. Rank reduction is used to filter out the noise and reconstruct the true signal.

Theorem 2.2.11 (Eckart–Young–Mirsky theorem) *Let $\mathcal{M}_k^{m\times n}$ denote the set of matrices in $\mathbb{C}^{m\times n}$ of rank k. Assume that $A \in \mathcal{M}_r^{m\times n}$ and consider the problem*

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

where $\|\cdot\|$ is a unitarily invariant norm. Then the SVD *expansion of A truncated to k terms, $X = A_k \equiv \sum_{i=1}^{k} \sigma_i u_i v_i^H$, solves this problem both for the spectral norm and the Frobenius norm. The minimum distance is given by*

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

The solution is unique if and only if $\sigma_k \neq \sigma_{k+1}$

Proof For the spectral norm the result follows directly from Lemma 2.2.1. For the Frobenius norm set $B = A - B_k$, where B_k has rank k. Then $\sigma_{k+1}(B_k) = 0$, and setting $j = k + 1$ in (2.2.39) we obtain

$$\sigma_i(A - B_k) \geq \sigma_{k+1}(A), \quad i = 1, 2, \ldots.$$

From this it follows that $\|A - B_k\|_F^2 \geq \sigma_{k+1}^2(A) + \cdots + \sigma_n^2(A)$. For the general case, see Stewart and Sun [273, 1990], Theorem IV. 4.18. $\square$

The approximation in the above theorem generally differs in all elements of A. Golub et al. [136, 1987] have proved a generalization that shows how to obtain a best approximation when a specified set of columns in the matrix are to remain fixed.

Theorem 2.2.12 *Any matrix $A \in \mathbb{C}^{m\times n}$, $m \geq n$, has a* **polar decomposition**

$$A = PH, \tag{2.2.40}$$

with $P \in \mathbb{C}^{m\times n}$ unitary, $P^H P = I_n$, and $H \in \mathbb{C}^{n\times n}$ Hermitian and positive semidefinite. This decomposition is unique and H is positive definite if and only if rank $(A) = n$.

Proof Let $A = U_1 \Sigma V^H$, $U_1 \in \mathbb{C}^{m\times n}$, be the "thin" SVD and set

$$P = U_1 V^H, \qquad H = V \Sigma V^H. \tag{2.2.41}$$

Then, since $V^H V = I$, it follows that $PH = U_1 V^H V \Sigma V^H = U_1 \Sigma V^H = A$. □

The theorem shows that the polar decomposition can be obtained from the SVD of A. If the polar decomposition $A = PH$ is given, then from a spectral decomposition $H = V \Sigma V^H$ one can construct the SVD $A = (PV)\Sigma V^H$. The polar decomposition is also related to the matrix square root and sign functions; see Sect. 3.8.1.

The significance of the factor P in the polar decomposition is that it is the unitary matrix closest to A. Its applications include factor analysis, satellite tracking, and the Procrustes problem; see Sect. 2.7.8. Perturbation bounds for the polar decomposition have been derived by Barrlund [13, 1990].

Theorem 2.2.13 *Let $A \in \mathbb{C}^{m\times n}$ be a given matrix and $A = PH$ its polar decomposition. Then for any unitary matrix $U \in \mathbb{C}^{m\times n}$,*

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

Proof This theorem was proved for $m = n$ and general unitarily invariant norms by Fan and Hoffman [98, 1955]. The generalization to $m > n$ follows from the additivity property of the Frobenius norm. □

Less well-known are the optimal properties of the Hermitian polar factor H. Let $A \in \mathbb{C}^{n\times n}$ 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 2.2.14 (Higham [161, 1986]) *Let $A \in \mathbb{C}^{n\times n}$ 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 $\|A - B\|_2 \leq \|A - X\|_2$ for any positive semidefinite Hermitian matrix X.

Expositions of the history of the SVD and the related polar decomposition are given by Stewart [270, 1993] and Horn and Johnson [167, 1991], Sect. 3.0. Applications of SVD in signal processing are surveyed in Vaccaro [284, 1991] and De Moor and Moonen [211, 1995]. The polar decomposition for a square nonsingular matrix was first given by Autonne [5, 1902]. The generalization to singular and rectangular matrices appeared in Autonne [6, 1915]. Both Beltrami and Jordan were concerned with diagonalizing a finite-dimensional bilinear form $P = x^T Ay$. Weyl [294, 1911] developed a general perturbation theory and gave a more elegant proof.

2.2.4 Backward Error Analysis

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

$$\min_x \|(A+\delta A)x-(b+\delta b)\|_2, \quad \|\delta A\| \le \epsilon\|A\|, \quad \|\delta b\| \le \epsilon\|b\|, \tag{2.2.42}$$

where ϵ is a small multiple of the unit roundoff $\mathbf{u}$. Algorithms based on orthogonal factorizations have been proved to be backward stable. On the other hand, algorithms in which the normal equations are explicitly formed are not backward stable. Many algorithms used for solving structured problems, such as Toeplitz least squares problems, are also not backward stable.

Any computed solution $\bar{x}$ is called a *stable solution* if it satisfies (2.2.42) for a sufficiently small ϵ. This does not mean that $\bar{x}$ is close to the exact solution x. If the problem is ill-conditioned, then a stable solution can be very different from x. But if ϵ is small compared to the uncertainty in A and b, the solution can be said to be as good as the data deserves. Further, the error $\|x-\bar{x}\|$ can be estimated using the perturbation results given in Sect. 2.2.2.

For a consistent linear system, *a posteriori* bounds for the backward error of a computed solution were derived in Sect. 1.4.5. Such bounds are more difficult to obtain for the linear least squares problem. Given an alleged solution $\widetilde{x}$, we would like to be able to find perturbations E and e of smallest norm such that $\widetilde{x}$ is the *exact* solution to $\min_x \|(b+e)-(A+E)x\|_2$. This could be used to verify numerically the stability of the solution. an algorithm. The following theorem is due to Stewart [266, 1977], Theorem 3.1.

Theorem 2.2.15 *Let $\widetilde{x}$ be an approximate solution to the least squares problem $\min_x \|Ax-b\|_2$ and $\widetilde{r}=b-A\widetilde{x}$ the corresponding residual. Then the vector $\widetilde{x}$ exactly solves $\min_x \|b-(A+E_i)x\|_2$, where*

$$E_1 = -\widetilde{r}(A^T\widetilde{r})^T/\|\widetilde{r}\|_2^2, \qquad E_2 = (\widetilde{r}-r)\widetilde{x}^T/\|\widetilde{x}\|_2^2 \tag{2.2.43}$$

and $r=b-Ax$ the residual corresponding to the exact solution x. The norms of these matrices are equal to

$$\|E_1\|_2 = \|A^T\widetilde{r}\|_2/\|\widetilde{r}\|_2, \tag{2.2.44}$$

$$\|E_2\|_2 = \sqrt{\|\widetilde{r}\|_2^2-\|r\|_2^2}\,\Big/\|\widetilde{x}\|_2 \le \|\widetilde{r}\|_2/\|\widetilde{x}\|_2. \tag{2.2.45}$$

Here $\|E_1\|_2$ is small when $\widetilde{r}$ is almost orthogonal to the column space of A and $\|E_2\|_2$ is small when $\widetilde{r}$ is almost equal to the residual r of the exact solution.

It is possible for $\widetilde{x}$ to have a backward error much smaller than either $\|E_1\|_2$ or $\|E_2\|_2$. Since there is not much loss of generality to assume that $\delta b=0$, we define

the *smallest* backward error bound to be

$$\eta_F(\widetilde{x}) = \min\left\{\|\delta A\|_F \mid \widetilde{x} \text{ solves } \min_x \|b - (A+\delta A)x\|_2\right\}. \tag{2.2.46}$$

It can be found by first characterizing the set of *all* backward perturbations of $\widetilde{x}$ and then finding the optimal bound, which minimizes the Frobenius norm of the perturbation.

Theorem 2.2.16 *Let $\widetilde{x}$ be an approximate solution to the least squares problem $\min_x \|Ax-b\|_2$ and set $\widetilde{r} = b - A\widetilde{x} \neq 0$. If $\widetilde{x} = 0$, then the optimal backward error in the Frobenius norm is $\eta_F(\widetilde{x}) = \|A^T\widetilde{r}\|_2/\|\widetilde{r}\|_2$. Otherwise*

$$\eta_F(\widetilde{x}) = \min\{\eta, \sigma_{\min}(B)\}, \tag{2.2.47}$$

where $B = \begin{pmatrix} A & C \end{pmatrix} \in \mathbb{R}^{m\times(n+m)}$ and

$$\eta = \|\widetilde{r}\|_2/\|\widetilde{x}\|_2, \qquad C = I - (\widetilde{r}\widetilde{r}^T)/\|\widetilde{r}\|_2^2. \tag{2.2.48}$$

Proof See Waldén et al. [290, 1995]. □

Computing $\sigma_{\min}(B)$ is too expensive in practice. If the QR factorization of A is available (see Sect. 2.3), then less costly lower and upper bounds for $\eta_F(\widetilde{x})$ can be computed in only $O(mn)$ operations. Let $r_1 = P_A\widetilde{r}$ be the orthogonal projection of $\widetilde{r}$ onto the range of A. If $\|r_1\|_2 \leq \alpha\|r\|_2$, then

$$\frac{\sqrt{5}-1}{2}\,\widetilde{\sigma}_1 \leq \eta_F(\widetilde{x}) \leq \sqrt{1+\alpha^2}\,\widetilde{\sigma}_1, \tag{2.2.49}$$

where

$$\widetilde{\sigma}_1 = \left\|(A^TA + \eta I)^{-1/2}A^T\widetilde{r}\right\|_2/\|\widetilde{x}\|_2. \tag{2.2.50}$$

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

How to find the optimal backward error for the linear least squares problem was an open problem for many years. It was solved by Waldén et al. [290, 1995]; see also [175, 1997]. Gu [142, 1998] gives several simple estimates of the optimal backward error that deviate at most by a factor 2. Optimal backward perturbation bounds for underdetermined systems are derived in [278, 1997]. The extension of backward error bounds to the case of constrained least squares problems is discussed by Cox and Higham [62, 1999].

2.2.5 Principal Angles Between Subspaces

In many applications the geometric relationship between two given subspaces needs to be investigated. For example, one subspace could be an approximate null space of A that we want to compare with the corresponding exact null space. The **principal**

angles and related principal vectors between two subspaces were first introduced by Jordan [173, 1875]. They are the invariants that best characterize their relative position.

Let $\mathcal{F}$ and $\mathcal{G}$ be subspaces of $\mathbb{C}^n$ and assume that $p = \dim(\mathcal{F}) \geq \dim(\mathcal{G}) = q \geq 1$. The smallest principal angle $\theta_1 = \theta_1(\mathcal{F}, \mathcal{G}) \in [0, \pi/2]$ between $\mathcal{F}$ and $\mathcal{G}$ is

$$\theta_1 = \min_{u \in \mathcal{F}} \min_{v \in cG} \angle(u, v).$$

Assume that the maximum is attained for $u = u_1$ and $v = v_1$. Then the next principal angle θ_2 is 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. This can be continued until one of the subspaces is empty.

Definition 2.2.2 The principal angles $\theta_k \in [0, \pi/2]$ between two subspaces of $\mathbb{C}^n$ are recursively defined for $k = 1{:}q$ by

$$\theta_k = \min_{u \in \mathcal{F}} \min_{v \in \mathcal{G}} \angle(u, v) = \angle(u_k, v_k), \tag{2.2.51}$$

subject to the constraints $u^H u_j = 0$, $v^H v_j = 0$, $j = 1{:}k-1$. The vectors u_k and v_k, $k = 1{:}q$, are called **principal vectors** of the pair of subspaces.

Theorem 2.2.17 (Björck and Golub [30]) *Assume that the columns of $Q_{\mathcal{F}} \in \mathbb{C}^{n\times p}$ and $Q_{\mathcal{G}} \in \mathbb{C}^{n\times q}$, $p \geq q$, form unitary bases for two subspaces of $\mathbb{C}^n$. Let the thin* SVD *of the matrix $M = Q_{\mathcal{F}}^H Q_{\mathcal{G}} \in \mathbb{C}^{p\times q}$ be*

$$M = YCZ^H, \quad C = \operatorname{diag}(\sigma_1, \ldots, \sigma_q), \tag{2.2.52}$$

where $Y^H Y = Z^H Z = ZZ^H = I_q$ 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

$$U = Q_{\mathcal{F}} Y, \qquad V = Q_{\mathcal{G}} Z. \tag{2.2.53}$$

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

$$\sigma_k = \max_{\|y\|_2 = \|z\|_2 = 1} y^H M z = y_k^H M z_k, \tag{2.2.54}$$

subject to $y^H y_j = z^H z_j = 0$, $j = 1{:}k$. If we put $u = Q_{\mathcal{F}} y \in \mathcal{F}$ and $v = Q_{\mathcal{G}} z \in \mathcal{G}$, 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 $y^H M z = y^H Q_{\mathcal{F}}^H Q_{\mathcal{G}} z = u^H v$, (2.2.54) 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 (2.2.53) follows directly from Definition 2.2.2. □

The principal angles are always uniquely defined, but the principal vectors are not. 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 $\mathcal{F}$ and

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

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

Definition 2.2.3 The distance between two subspaces $\mathcal{F}$ and $\mathcal{G}$ of $\mathbb{C}^n$, both of dimension p, is

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

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

$$\theta_{\max}(\mathcal{F}, \mathcal{G}) = \max_{\substack{u \in \mathcal{F} \\ \|u\|_2 = 1}} \min_{\substack{v \in \mathcal{G} \\ \|v\|_2 = 1}} \angle(u, v). \tag{2.2.55}$$

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

A unitary basis for the *intersection of two subspaces* is obtained by taking the vectors u_k that corresponds to $\theta_k = 0$, i.e., $\sigma_k = 1$. Clearly $0 \leq \operatorname{dist}(\mathcal{F}, \mathcal{G}) \leq 1$, and $\operatorname{dist}(\mathcal{F}, \mathcal{G}) = 0$ if and only if $\mathcal{F} = \mathcal{G}$. Since small angles θ_k are not well defined by $\cos \theta_k$, it is preferable to compute $\sin \theta_k$ more directly. Changing notation slightly, we write the SVD in (2.2.52) as $M = Q_{\mathcal{F}}^H Q_{\mathcal{G}} = Y_{\mathcal{F}} C Y_{\mathcal{G}}^H$, and denote the principal vectors by $U_{\mathcal{F}} = Q_{\mathcal{F}} Y_{\mathcal{F}}$, $U_{\mathcal{G}} = Q_{\mathcal{G}} Y_{\mathcal{G}}$. Then $P_{\mathcal{F}} = Q_{\mathcal{F}} Q_{\mathcal{F}}^H$ is the orthogonal projector onto $\mathcal{F}$ and we have

$$P_{\mathcal{F}} Q_{\mathcal{G}} = Q_{\mathcal{F}} Q_{\mathcal{F}}^H Q_{\mathcal{G}} = Q_{\mathcal{F}} M = U_{\mathcal{F}} C Y_{\mathcal{G}}. \tag{2.2.56}$$

Squaring $Q_{\mathcal{G}} = P_{\mathcal{F}} Q_{\mathcal{G}} + (I - P_{\mathcal{F}}) Q_{\mathcal{G}}$, using (2.2.56) and $P_{\mathcal{F}}(I - P_{\mathcal{F}}) = 0$ gives

$$Q_{\mathcal{G}}^H (I - P_{\mathcal{F}})^2 Q_{\mathcal{G}} = Y_{\mathcal{G}} (I - C^2) Y_{\mathcal{G}}^H.$$

This shows that the SVD of $(I - P_{\mathcal{F}}) Q_{\mathcal{G}} = Q_{\mathcal{G}} - Q_{\mathcal{F}} M$ can be written

$$(I - P_{\mathcal{F}}) Q_{\mathcal{G}} = W_{\mathcal{F}} S Y_{\mathcal{G}}^H,$$

where $S^2 = I - C^2$, and thus $S = \pm \operatorname{diag}(\sin \theta_k)$.

The distance between subspaces can also be expressed as

$$\operatorname{dist}(\mathcal{F}, \mathcal{G}) = \| P_{\mathcal{F}} - P_{\mathcal{G}} \|_2, \tag{2.2.57}$$

where $P_{\mathcal{F}}$ and $P_{\mathcal{G}}$ are the orthogonal projectors onto $\mathcal{F}$ and $\mathcal{G}$, respectively; see Golub and Van Loan [133, Theorem 2.6.1].

A remarkable complete analysis of the angles between subspaces was published in 1875 by Jordan [173, 1875]. Mixed stability of the Björck–Golub method is shown by Drmač [75, 2000]. Knyazev and Argentati [179, 2002] survey sine and cosine algorithms and prove basic perturbation theorems for principal angles.

Another applications of principal angles and vectors is in statistical modeling. To measure how "close" two sets A and B of observations are, Hotelling [168, 1936] introduced the **canonical correlations** $\cos\theta_k$, where θ_k are the principal angles between the subspaces spanned by A and B. These are used in a wide variety of applications in econometrics, psychology, and geodesy. A perturbation analysis of canonical correlations of matrix pairs was given by Golub and Zha [134, 1994].

Exercises

2.2.1 (a) Show that the pseudoinverse of a complex vector v is given by

$$v^{\dagger} = \begin{cases} 0 & \text{if } v = 0, \\ v^H/\|v\|_2 & \text{if } v \neq 0. \end{cases}$$

(b) Let $v \in \mathbb{C}^n$, $v \neq 0$, $V_2 \in \mathbb{C}^{n\times(n-1)}$, and $V_2 V_2^H = I - vv^H$. Show that the matrix $V = \begin{pmatrix} v/\|v\|_2 & V_2 \end{pmatrix}$ is unitary.

(c) For $A = \begin{pmatrix} 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 1 & 1 \end{pmatrix}^T$, show that $1 = (AB)^{\dagger} \neq B^{\dagger}A^{\dagger} = 1/2$.

2.2.2 Show that, if A, $B \in \mathbb{R}^{m\times n}$ 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 $\|B - A\|_2 = \epsilon$, $\|B^{\dagger} - A^{\dagger}\|_2 > 1/\epsilon$.

2.2.3 Show that for any matrix $A \in \mathbb{R}^{m\times n}$ it holds that

$$A^{\dagger} = \lim_{\mu\to+0} (A^TA + \mu^2 I)^{-1}A^T = \lim_{\mu\to+0} A^T(AA^T + \mu^2 I)^{-1}. \tag{2.2.58}$$

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

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

$$B_1 = A^{\dagger}AB, \qquad A_1 = AB_1B_1^{\dagger}.$$

Use the Penrose conditions to show that $AB = AB_1 = A_1B_1$ and that $(AB)^{\dagger} = B_1^{\dagger}A_1^{\dagger}$.

2.2.6 (a) Show that the matrix $A \in \mathbb{R}^{m\times n}$ has a left inverse $A^L \in \mathbb{R}^{n\times m}$, i.e., $A^LA = I$, 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.

2.2.7 Prove *Bjerhammar's characterization*: Let $A \in \mathbb{R}^{m\times n}$ have full column rank and B be any matrix such that $A^TB = 0$ and $\begin{pmatrix} A & B \end{pmatrix}$ is nonsingular. Then $A^{\dagger} = X^T$, where

$$\begin{pmatrix} X^T \\ Y^T \end{pmatrix} = \begin{pmatrix} A & B \end{pmatrix}^{-1}.$$

2.2.8 Let the singular values of $A \in \mathbb{R}^{m\times n}$ be $\sigma_1 \geq \cdots \geq \sigma_n$. What relations hold between these and the singular values of $\widetilde{A} = (A, u)$, and $A = \begin{pmatrix} A \\ v^T \end{pmatrix}$?

2.2.9 Give the best approximation of rank $k < n$ of a matrix $A \in \mathbb{R}^{m\times n}$ of rank n in terms of the SVD of A. By "best" we mean that the distance $\|A - B\|$ is minimized for both the Frobenius norm and the 2-norm.

2.2.10 (a) Let $A = (a_1, a_2)$, where $\|a_1\|_2 = \|a_2\|_2 = 1$. Then $a_1^T a_2 = \cos\theta$, where θ 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

$$A^T A = \begin{pmatrix} 1 & \cos\theta \\ \cos\theta & 1 \end{pmatrix}.$$

Then determine the left singular vectors u_1, u_2 from $Av_i = \sigma_i u_i$, $i = 1, 2$.

(b) Show that if $\theta \ll 1$, then $\sigma_1 \approx \sqrt{2}$ and $\sigma_2 \approx \theta/\sqrt{2}$, $u_1 \approx (a_1 + a_2)/2$, and $u_2 \approx (a_1 - a_2)/\theta$.

2.3 Orthogonal Factorizations

> The great stability of unitary transformations in numerical analysis springs from the fact that both the ℓ_2-norm and the Frobenius norm are unitarily invariant. This means in practice that even when rounding errors are made no substantial growth takes place in the norm of the successive transformed matrices.
>
> — J. H. Wilkinson, The Algebraic Eigenvalue Problem [295, 1965]

Orthogonality plays a key role in least squares problems, and a least squares solution x is characterized by the property that the residual $r = b - Ax$ is orthogonal to $\mathcal{R}(A)$; see Theorem 2.1.2. By using methods directly based on orthogonality the squaring of the condition number that results from forming the normal equations can be avoided.

2.3.1 Elementary Orthogonal Matrices

In Sect. 1.2.5 elementary elimination matrices of the form $L_j = I + l_j e_j^T$ (see (1.2.30)) were used to describe the Gaussian elimination. We now introduce **elementary unitary matrices**, which are unitary matrices equal to *the unit matrix modified by a matrix of rank one*. Such transformations are versatile tools for constructing stable algorithms for a variety of matrix problems.

We first consider elementary real orthogonal matrices of the form

$$H(u) = I - 2\frac{uu^T}{u^T u} \in \mathbb{R}^{n\times n}. \tag{2.3.1}$$

Clearly, H is symmetric ($H^T = H$) and with $\beta = 2/u^T u$,

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

shows that H is orthogonal and $H^{-1} = H$. For any vector $x \in \mathbb{R}^n$, we have

$$Hx = (I - \beta uu^T)x = x - \beta(u^T x)u$$

and it follows that $Hx \in \text{span}\{x, u\}$. The effect of the transformation Hx is to reflect x in the $(m-1)$-dimensional hyperplane with normal vector u; see Fig. 2.2. This is equivalent to subtracting *twice* the orthogonal projection of x onto u. In particular, $Hu = -u$, i.e., H reverses u, and if $x \perp u$ then $Hx = x$. Hence, H has one eigenvalue equal to -1 and the remaining eigenvalues are all equal to $+1$. Note that $\|x\|_2 = \|Hx\|_2$, and that the normal u is parallel to the vector $x - Hx$.

The use of elementary reflectors in numerical linear algebra was introduced in matrix computation in 1958 by Householder[3] [169, 1958]. A matrix of the form (2.3.1) is therefore often called a **Householder reflection** and is uniquely determined by the vector u, called the **Householder vector**. Hence, the matrix H need never be explicitly formed.

In applications of Householder reflectors the following standard task is central. Given a nonzero vector $x \in \mathbb{R}^m$, construct a plane reflection such that *multiplication by H zeros all components in x except the first*. That is

$$Hx = s_1 \sigma e_1, \quad \sigma = \|x\|_2, \quad s_1 = \pm 1. \tag{2.3.2}$$

Here e_1 denotes the first unit vector and the second equation is a consequence of the fact that H is orthogonal. Multiplying (2.3.2) from the left by H and using $H^2 = I$, we find that $He_1 = \pm x/\sigma$. Hence, the first column in H is proportional to x. It is easily seen that (2.3.2) is satisfied if we set

$$x = \begin{pmatrix} \xi_1 \\ x_2 \end{pmatrix}, \quad u = \begin{pmatrix} \xi_1 - s_1\sigma \\ x_2 \end{pmatrix}. \tag{2.3.3}$$

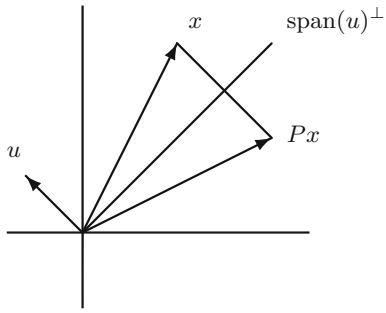


Fig. 2.2 Householder reflection of a vector x

[3] Alston S. Householder (1904–1993), American mathematician at Oak Ridge National Laboratory and University of Tennessee at Knoxville. He pioneered the use of matrix factorization and orthogonal transformations in numerical linear algebra.

Note that u differs from x only in its first component. A short calculation gives

$$u^T u = (x - s_1\sigma e_1)^T(x - s_1\sigma e_1) = (\sigma^2 - 2s_1\sigma\xi_1 + \sigma^2) = 2\sigma(\sigma - s_1\xi_1).$$

If x is close to a multiple of e_1, then $\sigma \approx |\xi_1|$ and cancellation in this formula may lead to a large relative error in β. To avoid cancellation, we take $s_1 = -1$, if $\xi_1 > 0$, and $s_1 = +1$, if $\xi_1 \le 0$, giving

$$u_1 = s_1(\sigma + |\xi_1|), \qquad \beta = \frac{1}{\sigma(\sigma + |\xi_1|)}, \tag{2.3.4}$$

This corresponds to a reflection in the *outer* bisector of the angle between x and $\xi_1 e_1$, not the inner bisector as shown in Fig. 2.2. In particular, the vector $x = \pm e_1$ will be mapped into $\mp e_1$. (Note that the identity matrix I is not a reflector because $\det(I) = +1$.)

A Householder reflector (2.3.1) is invariant under scaling: $H(\alpha u) = H(u)$ for any $\alpha \neq 0$. Since by (2.3.4) $\|u\|_\infty = |u_1|$ we can rescale u so that $u_1 = 1$. Then

$$u_2 = s_1 x_2/\gamma, \quad \gamma = (\sigma + |\xi_1|) \qquad \beta = 1 + \frac{|\xi_1|}{\sigma}. \tag{2.3.5}$$

This has the advantage that β can be stably reconstructed from u_2:

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

The vector u_2 can be stored in the locations for the zeroed entries in x. Note that if $\xi_1 \neq 0$, then $s_1 = -\text{sign}(\xi_1)$, but this relation is not true when $\xi_1 = 0$. This case occurs in the analysis of the modified Gram–Schmidt orthogonalization; see Sect. 2.3.6. In MATLAB `sign(0) = 0`, which can cause errors.

Algorithm 2.3.1 computes a Householder reflector $H = I - \beta u u^T$, with $u^T e_1 = 1$, such that for a given real vector $x \neq 0$, $Hx = \pm(\xi_1)\|x\|_2 e_1$. If $n = 1$ or $x(2:n) = 0$, then $\beta = 0$ is returned.

Algorithm 2.3.1 (*Construct real Householder reflector*)

```
function [u,beta,sigma] = houseg(x)
% HOUSEG generates a real Householder reflector
%   H = I - beta*u*u', with u_1 = 1, such that
%   H*x = sigma*e_1, sigma = ||x||_2.
% ----------------------------------------
u = x;  sigma = norm(x);
u(1) = sigma + abs(x(1));
beta = u(1)/sigma;
if x(1) < 0, u(1) = -u(1);
   else sigma = -sigma;
end
% Normalize the Householder vector
u = u/u(1);
```

The Householder reflector described above is the one commonly used. It is stable because it uses only additions of positive quantities. The choice of reflector in the inner bisector is not stable if the expression (2.3.4) is used because it leads to numerical cancellation when $\xi_1 \approx \sigma$. However, as shown by Parlett [229, 1971], this can be avoided, by rewriting the formula as

$$\sigma - |\xi_1| = \frac{\|x\|_2^2 - \xi_1^2}{\sigma + |\xi_1|} = \frac{\|x_2\|_2^2}{\sigma + |\xi_1|}. \tag{2.3.6}$$

In many algorithms a matrix is to be premultiplied (or postmultiplied) by a sequence of Householder reflectors. It is important to note that in these operations the Householder reflectors are never formed explicitly, but are implicitly represented. When *premultiplying* $A = (a_1, \ldots, a_n) \in \mathbb{R}^{m\times n}$ by a Householder reflector $H \in \mathbb{R}^{m\times m}$, the product $HA = (Ha_1, \ldots, Ha_n)$, is computes as

$$Ha_j = (I - \beta uu^T)a_j = a_j - \beta(u^T a_j)u, \quad j = 1{:}n. \tag{2.3.7}$$

Similarly, in *postmultiplying* A with $H \in \mathbb{R}^{n\times n}$, H acts on the *row* vectors of A. Both operations,

$$HA = A - \beta u(u^T A) \quad \text{and} \quad AH = A - \beta(Au)u^T,$$

require one matrix–vector product followed by a rank-one update and use $4mn$ flops.

In the complex case a Householder reflector has the form (see Wilkinson [296, 1965], pp. 49–50).

$$H = I - \beta uu^H, \quad \beta = \frac{2}{u^H u}, \quad u \in \mathbb{C}^n. \tag{2.3.8}$$

It is easy to check that H is Hermitian and unitary ($H = H^H = H^{-1}$). Given $x \in \mathbb{C}^n$ with $xe_1 = \xi_1 = e^{i\theta_1}|\xi_1|$, we want to determine u such that

$$Hx = \zeta\sigma e_1, \quad |\zeta| = 1, \quad |\sigma| = \|x\|_2.$$

Since H is Hermitian, $x^H Hx = \zeta\sigma x^H e_1 = \zeta\bar{\xi}_1\sigma$ must be real. Hence, unless ξ_1 is real it is not possible to have ζ real. To avoid cancellation in the first component of $u = x - \zeta\sigma e_1$, we take $\zeta = -e^{i\theta_1}$, giving

$$u_1 = \xi_1 - \zeta\sigma = e^{i\theta_1}(|\xi_1| + \sigma).$$

Since $|\zeta|^2 = 1$, we have

$$u^H u = \left(\|x\|_2^2 + \sigma(\bar{\zeta}\xi_1 + \zeta\bar{\xi}_1) + |\zeta|^2\sigma|^2\right) = 2\sigma(\sigma + |\xi_1|). \tag{2.3.9}$$

For the constructed reflector H we have that $-e^{-i\theta_1}Hx = \sigma e_1$ is real and positive.

Another useful class of elementary orthogonal transformations is that of **plane rotations**, also called **Givens rotations**.[4] A plane rotation clockwise through an angle θ in $\mathbb{R}^2$ is represented by the matrix

$$G = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}. \tag{2.3.10}$$

Note that $G^{-1}(\theta) = G(-\theta)$, and $\det G(\theta) = +1$. In $\mathbb{R}^n$ 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:

$$G_{ij}(\theta) = \begin{array}{c} \\ i \\ \\ j \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccc} & & i & & j & & \end{array} \\ \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & & s & & \\ & & & \ddots & & & \\ & & -s & & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{pmatrix} \end{array}. \tag{2.3.11}$$

In practice, neither the angle θ nor the matrix G_{ij} are explicitly constructed, only the values c and s are computed. Once these are known, premultiplying a vector $a = (\alpha_1, \ldots, \alpha_n)^T$ by $G_{ij}(\theta)$ is achieved by

$$\begin{pmatrix} \beta_i \\ \beta_j \end{pmatrix} = G_{ij} \begin{pmatrix} \alpha_i \\ \alpha_j \end{pmatrix} = \begin{pmatrix} c\,\alpha_i + s\,\alpha_j \\ -s\,\alpha_i + c\,\alpha_j \end{pmatrix}. \tag{2.3.12}$$

Only the components α_i and α_j are affected. The cost of multiplying a plane rotation into a vector is four multiplications and two additions or 6 flops.

If $\alpha_j \neq 0$, we can construct $G_{ij}(\theta)$ to make $\beta_j = 0$ by setting

$$c = \alpha_i/\sigma, \qquad s = \alpha_j/\sigma, \quad \sigma = (\alpha_i^2 + \alpha_j^2)^{1/2} > 0. \tag{2.3.13}$$

[4] Named after the American mathematician and pioneer of computer science Wallace Givens (1910–1993), who used them in [123, 1958] to reduce matrices to simpler form. He got his PhD in 1936 at Princeton University under Oscar Veblen, and later worked at University of Tennessee, Knoxville, and Argonne National Laboratories.

While a Householder transformation can be used to zero a large portion of a vector, a Givens rotation zeros just a single entry.

Algorithm 2.3.2 constructs the plane 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 use $-G$. The algorithm requires five flops and one square root.

Algorithm 2.3.2 (*Construct Real Plane Rotation*)

```
function [c,s,r] = givens(a,b)
% GIVENS computes c and s in a real plane rotation
% so that 0 = -s*a + c*b, and r = c*a + s*b
% ------------------------------------------------
if b == 0,
   c = 1.0; s = 0.0; r = a;
   return
end
if abs(b) > abs(a)    % Make |t| <= 1.
   t = a/b; tt = sqrt(1+t*t);
   s = 1/tt; c = t*s; r = tt*b;
else
   t = b/a; tt = sqrt(1+t*t);
   c = 1/tt; s = t*c; r = tt*a;
end
```

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

$$\begin{aligned} a_{ik} &:= c\, a_{ik} + s\, a_{jk}, \\ a_{jk} &:= -s\, a_{ik} + c\, a_{jk}, \end{aligned}$$

$k = 1{:}n$. The product requires $4n$ multiplications and $2n$ additions or $6n$ flops. Postmultiplying A with G_{ij} uses $6m$ flops and only affects *columns* i and j.

An arbitrary nonzero vector $x \in \mathbb{R}^n$ can be transformed into σe_1 with $\sigma = \|x\|_2 > 0$ using a sequence of plane rotations. Let G_{1k}, $k = 2{:}m$ be a sequence of plane rotations, where G_{1k} zeros the kth component in x. Then $G_{1n}\cdots G_{13}G_{12}x = \sigma e_1$. 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 (2.3.10) has determinant equal to $+1$. We could equally well work with plane reflectors of the form

$$H = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} \tag{2.3.14}$$

and determinant equal to -1. The trigonometric identities

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

show the relationship to a 2×2 Householder reflector.

Example 2.3.1 An orthogonal matrix $Q \in \mathbb{R}^{3\times 3}$ in three dimensions is a pure rotation if $\det(Q) = 1$. Such a matrix can be represented as a product of three successive plane rotations or by the angles of these rotations. The classical choice is as a product of the three plane rotations $G_{23}(\phi)G_{12}(\theta)G_{23}(\psi)Q = I$, where ϕ, θ, and ψ are the **Euler angles**:

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

The first rotation $G_{23}(\psi)$ is used to zero the element q_{31}. Next, $G_{12}(\theta)$ zeros the modified element q_{21}. Finally, $G_{23}(\phi)$ is used to zero q_{32}. The angles can always be chosen to make the diagonal elements positive. Since the final product is orthogonal and upper triangular, it must be the unit matrix I_3. By orthogonality, we have

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

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 q_{21}, q_{31}, q_{32}, yields a continuous representation and is to be preferred. This corresponds to the product

$$G_{23}(\phi)G_{13}(\theta)G_{12}(\psi)Q = I_3.$$

For more details, see Hanson and Norris [156, 1981]. □

Complex **unitary** plane rotations have the form

$$G = \begin{pmatrix} \bar{c} & \bar{s} \\ -s & c \end{pmatrix}, \quad c = e^{i\gamma}\cos\theta, \quad s = e^{i\delta}\sin\theta. \tag{2.3.15}$$

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 $z \in \mathbb{C}^2$, we have

$$G\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} \bar{c}z_1 + \bar{s}z_2 \\ -sz_1 + cz_2 \end{pmatrix} = \begin{pmatrix} \sigma \\ 0 \end{pmatrix}, \tag{2.3.16}$$

provided that

$$c = z_1/\sigma, \quad s = z_2/\sigma, \qquad |\sigma|^2 = \|z\|_2^2 = |z_1|^2 + |z_2|^2 > 0.$$

A vector $z \in \mathbb{C}^n$ can be transformed into σe_1 by successive premultiplication with $n-1$ unitary plane rotations in the planes $(1,2), (1,3), \ldots, (1,n)$. The rotations may be chosen so that σ is real and nonnegative.

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 [264, 1976] 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

$$\rho = \begin{cases} s, & \text{if } |s| < |c|, \\ 1/c, & \text{if } |c| \leq |s|. \end{cases} \tag{2.3.17}$$

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

$$\begin{aligned} s = \rho, \quad c = \sqrt{1-s^2}, \quad & \text{if } |\rho| < 1, \\ c = 1/\rho, \quad s = \sqrt{1-c^2}, \quad & \text{if } |\rho| > 1. \end{aligned}$$

It is possible to rearrange the plane rotations so that only two instead of four multiplications per element are used and no square roots are required. These modified transformations, called **fast Givens transformations**, are due to Gentleman [115, 1973] and Hammarling [146, 1974]. The basic idea is to take out a scaling factor and write

$$G = cQ = c \begin{pmatrix} 1 & s/c \\ -s/c & 1 \end{pmatrix} \quad \text{or} \quad G = sQ = s \begin{pmatrix} c/s & 1 \\ -1 & c/s, \end{pmatrix} \tag{2.3.18}$$

depending on whether $|c| > |s|$ or $|c| \leq |s|$. In a product of rotations $G_1 \cdots G_k$ the scaling factors c_i and s_i are accumulated separately. A dynamic scaling has been suggested by Anda and Park [1, 1994]. On modern processors the gain in speed is modest. Because of this and the nontrivial amount of monitoring needed to avoid overflow and underflow, the usefulness of fast Givens transformations appears to be limited and LAPACK does not make use of them.

Plane rotations were used already by Jacobi [172, 1845] to achieve diagonal dominance in systems of normal equations. He then applied a simple iterative scheme that became known as Jacobi's method; see Sect. 3.6.2. The reliable construction of real and complex plane rotations are considered in great detail in Bindel, Demmel, and Kahan [19, 2002].

Wilkinson [296, 1965] proved the backward stability of algorithms based on sequences of Householder reflectors. Parlett [230, 1998] gives stable formulas also for the choice of Householder reflector corresponding to the inner bisector. Dubrulle [76, 2000] shows that the inner reflectors perform better in some eigenvalue algorithms. Different implementations of complex Householder transformations are compared by Lehoucq [191, 1996] and Demmel et al. [71, 2008].

2.3.2 QR Factorization and Least Squares Problems

We first show that any matrix $A \in \mathbb{C}^{m\times n}$ with $m \geq n$ can be factored into the product of a *square* unitary matrix and an upper triangular matrix with real positive diagonal elements.

Theorem 2.3.1 (Full QR Factorization) *Let $A \in \mathbb{C}^{m\times n}$ with* rank $(A) = n$. *Then there is a factorization*

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} R \\ 0 \end{pmatrix} \tag{2.3.19}$$

such that $Q \in \mathbb{C}^{m\times m}$ is a square unitary matrix and $R \in \mathbb{C}^{n\times n}$ is upper triangular with real positive diagonal elements. The matrices R and $Q_1 = AR^{-1}$ are uniquely determined.

Proof The proof is by induction on n. For $A = a_1 \in \mathbb{R}^m$ we set $q_1 = a_1/\rho$, where $\rho = \|a_1\|_2 > 0$. Then q_1 is a unit vector and there is a unitary matrix $U = \begin{pmatrix} q_1 & U_1 \end{pmatrix}$ with q_1 as its first column. Then $U^H a_1 = \begin{pmatrix} \rho \\ 0 \end{pmatrix}$, which shows that the statement is valid for $n = 1$. Assume now that the statement is true for $n - 1$. We will show that it holds for any $A = (a_1\ A_2) \in \mathbb{C}^{m\times n}$. Using the construction for $n = 1$, we have

$$U^H A = \begin{pmatrix} \rho & q_1^H A_2 \\ 0 & U_1^H A_2 \end{pmatrix} = \begin{pmatrix} \rho & r \\ 0 & B \end{pmatrix},$$

where $B \in \mathbb{C}^{(m-1)\times(n-1)}$ and rank $(B) = n - 1$. By the induction hypothesis, there is a unitary matrix $\widetilde{Q}$ such that $\widetilde{U}^H B = \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}$. Hence, if we define

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

then (2.3.19) will hold. □

By (2.3.21), the columns of Q_1 and Q_2 form orthonormal bases for the range space of A and its orthogonal complement, $\mathcal{R}(A) = \mathcal{R}(Q_1)$, $\mathcal{N}(A^H) = \mathcal{R}(Q_2)$. The matrix Q_2 in (2.3.21) is not uniquely determined. The corresponding orthogonal projections are

$$P_A = Q_1 Q_1^H, \qquad P_A^{\perp} = Q_2 Q_2^H. \tag{2.3.20}$$

Note that since Q in (2.3.19) is unitary, it follows that R has the same singular values and right singular vectors as A.

The QR factorization can be written more compactly as

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R, \quad Q_1 \in \mathbb{C}^{m \times n}, \tag{2.3.21}$$

which is the **thin QR factorization**. A QR factorization can be computed also for a rank-deficient matrix A. But then some of the diagonal elements in R must be zero. A simple example is

$$A = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} 0 & s \\ 0 & c \end{pmatrix} \equiv QR,$$

which holds for any s and c such as $s^2 + c^2 = 1$. Here, the columns of Q do not give any information about $\mathcal{R}(A)$. Such a factorization is not useful, until it has been further reduced. The remedy is to use column interchanges; see Sect. 2.4.2.

If $\operatorname{rank}(A) = m < n$, then A has linearly independent rows and from Theorem 2.3.1 it follows that A^H has a unique QR factorization. Equivalently

$$A = \begin{pmatrix} L & 0 \end{pmatrix} Q^H = \begin{pmatrix} L & 0 \end{pmatrix} \begin{pmatrix} Q_1^H \\ Q_2^H \end{pmatrix}, \tag{2.3.22}$$

where $L \in \mathbb{C}^{m \times m}$ is lower triangular with real positive diagonal elements.

Lemma 2.3.1 *Let $A \in \mathbb{C}^{m \times n}$ have the (thin) QR factorization $A = Q_1 R$, where R has positive diagonal elements. Then $R = L^T$, where L is the unique lower triangular Cholesky factor of $A^H A$.*

Proof Since $\operatorname{diag}(R) > 0$, it follows that $\operatorname{rank}(A^H A) = n$. Then $A^H A$ has a unique lower triangular Cholesky factor L with a positive diagonal. From the thin QR factorization it follows that $A^H A = R^H Q_1^H Q_1 R = R^H R$, which shows that R^H is the Cholesky factor. □

The proof of Theorem 2.3.1 gives a way to compute Q and R, provided that we can construct a unitary matrix $U = (y,\ U_1)$, given its first column. Several ways to perform this construction using elementary orthogonal transformations were given in Sect. 2.3.1. The matrix Q is *implicitly* defined as a product of Householder reflectors or Givens rotations.

In developing the following algorithm, we assume that $A \in \mathbb{R}^{m \times n}$ is a real matrix with $\operatorname{rank}(A) = n$. Then in (2.3.19) Q is real orthogonal and R real upper triangular with nonzero diagonal. The QR factorization is computed by premultiplying A by a sequence of n Householder reflectors. In the first step $H_1 = I - \beta_1 u_1 u_1^T$ is determined so as to zero out the elements below the diagonal in the first column of A:

$$H_1 A = H_1 \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{n2} & \dots & a_{nn} \end{pmatrix} = \left(\begin{array}{c|ccc} r_{11} & r_{12} & \dots & r_{1n} \\ \hline 0 & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} \end{array}\right). \tag{2.3.23}$$

With $a_1 = Ae_1$, H_1 is determined so that

$$H_1 a_1 = H_1 \begin{pmatrix} a_{11} \\ \widetilde{a}_1 \end{pmatrix} = \begin{pmatrix} r_{11} \\ 0 \end{pmatrix}, \quad r_{11} = -s_{11}\sigma_1, \quad \sigma_1 = \|a_1\|_2,$$

where $s_{11} = \operatorname{sign}(a_{11})$, if $a_{11} \neq 0$, and $s_{11} = 1$ if $a_{11} = 0$.

After the first step, as indicated by the notation in (2.3.23), the first row is the first row in the final factor R. In the next step, a Householder transformation is chosen to zero elements in the second column of $H_1 A$. This transformation will only affect the $(m-1) \times (n-1)$ block in the lower right corner of $H_1 A$. All remaining steps are similar. After step k, $k < n$, we have computed a matrix of the form

$$A^{(k)} = H_k \cdots H_1 A = \left(\begin{array}{c|c} R_{11} & R_{12} \\ \hline 0 & \widetilde{A}^{(k)} \end{array}\right), \quad k = 1{:}n. \tag{2.3.24}$$

Here $R_{11} \in \mathbb{R}^{k \times k}$ is upper triangular and the first k rows are rows in the final matrix R. The next step is

$$A^{(k+1)} = H_{k+1} A^{(k)}, \qquad H_{k+1} = \operatorname{diag}(I_k,\ \widetilde{H}_{k+1}). \tag{2.3.25}$$

Here the Householder transformation $\widetilde{H}_{k+1}$ is chosen to zero the elements below the main diagonal in column $k+1$ of $A^{(k)}$, $\widetilde{H}_{k+1}\widetilde{A}^{(k)} e_1 = r_{kk} e_1$. This only affects the trailing diagonal block $\widetilde{A}^{(k)} \in \mathbb{R}^{(m-k)\times(n-k)}$. After n steps we have obtained the QR factorization of A

$$H_n \cdots H_2 H_1 A = Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix}. \tag{2.3.26}$$

Here Q is implicitly given in terms as $Q = H_1 H_2 \cdots H_n$. Hence, Q is defined by the Householder vectors $\widehat{u}_k$, which can overwrite the elements in the strictly lower trapezoidal part of A. Thus, all information associated with the factors Q and R can be fitted into the array holding A. The vector $(\beta_1, \dots, \beta_n)$ of length n is usually stored separately, but can be recomputed from $\beta_k = \frac{1}{2}(1 + \|\widehat{u}_k\|_2^2)^{1/2}$.

Algorithm 2.3.3 computes the QR factorization of $A \in \mathbb{C}^{m\times n}$ $(m \geq n)$ using Householder transformations. 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 2.3.3 (*Householder QR Factorization*)

```
function [U,R,beta] = houseqr(A,ifq1)
% HOUSEQR Computes the Householder QR factorization
%   of the m by n matrix A (m >= n). At return
%   U and beta contain the Householder reflections,
% ------------------------------------------------
[m,n] = size(A);
u = zeros(m,1); beta = zeros(n,1);
for k = 1:n
   if k < m,
% Construct and save Householder k:th reflector
      [u(k:m),beta(k),A(k,k)] = houseg(A(k:m,k));
      A(k+1:m,k) = u(k+1:m);
% Apply k:th Householder reflector
      A(k:m,k+1:n) = A(k:m,k+1:n) - ...
            beta(k)*u(k:m)*(u(k:m)'*A(k:m,k+1:n));
   end
end
U = eye(m,n) + tril(A,-1);  R = triu(A(1:n,:));
```

In step k the application of the Householder reflector to the active part of the matrix requires $4(m-k+1)(n-k)$ flops. Hence, the total flop count becomes

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

For $m = n$ this equals $4n^3/3$ flops.

It is usually not necessary to compute explicitly the full square orthogonal factor $Q \in \mathbb{R}^{m\times m}$. But if needed, the product

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} = H_1 H_2 \cdots H_n(e_1, \ldots, e_n, e_{n+1}, \ldots, e_m) \in \mathbb{R}^{m\times m} \tag{2.3.27}$$

can be accumulated from right to left. By (2.3.25) the transformation H_{k+1} leaves the first k rows unchanged. It follows that

$$q_k = H_1 \cdots H_p e_k, \quad k = 1{:}m, \quad p = \min\{k, n\}. \tag{2.3.28}$$

Algorithm 2.3.4 computes the matrix $Q_1 = (q_1, \ldots, q_n) \in \mathbb{R}^{m\times n}$ $(m \geq n)$, giving an orthogonal basis of $\mathcal{R}(A)$. This requires $2(mn^2 - n^3/3)$ flops, or for $m = n$ is $4n^3/3$ flops.

Algorithm 2.3.4 (*Accumulating Q_1 in Householder QR*)

```
function Q = houseq1(U,beta)
% HOUSEQ1 generates the m by n orthogonal matrix
%   Q from a given Householder QR factorization
% ---------------------------------------------
[m,n] = size(U);
Q = eye(m,n)
for k = n:-1:1
   v = beta(k)*(U(k:m,k)'*Q(k:m,k:n));
   Q(k:m,k:n) = Q(k:m,k:n) - U(k:m,k)*v;
end
```

The matrix $Q_2 = (q_{n+1}, \ldots, q_m)$, which gives an orthogonal basis for $\mathcal{N}(A^T)$, requires $2n(m-n)(2m-n)$ flops to generate. The total work for generating the full matrix $Q = (Q_1,\ Q_2) \in \mathbb{R}^{m\times m}$ is $4(mn(m-n) + n^3/3)$ flops,

For a complex matrix $A \in \mathbb{C}^{m\times n}$ the QR factorization can be computed by using a sequence of unitary Householder reflectors. As remarked in Sect. 2.3.1 this will in general not give a factor R with real positive diagonal elements. This can be remedied by a unitary scaling:

$$A = U\begin{pmatrix} R \\ 0 \end{pmatrix} = (UD^{-1})\begin{pmatrix} DR \\ 0 \end{pmatrix}, \quad D = \operatorname{diag}(e^{i\alpha_1}, \ldots, e^{i\alpha_n}).$$

The following backward error result for Householder QR is due to Higham [162, 2002], Theorem 19.4.

Theorem 2.3.2 *Let $\widehat{R}$ denote the upper triangular matrix computed by the Householder* QR *algorithm for $A \in \mathbb{R}^{m\times n}$. Then there exists an exactly orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ such that*

$$A + \Delta A = Q\begin{pmatrix} \widehat{R} \\ 0 \end{pmatrix}, \qquad \|\Delta a_j\|_2 \le \widetilde{\gamma}_{mn}\|a_j\|_2, \quad j = 1{:}n, \tag{2.3.29}$$

where c is a small constant. The matrix Q is given explicitly by

$$Q = (H_1 H_2 \cdots H_n)^T,$$

where H_k is the Householder matrix that corresponds to the exact application of the kth step to the matrix $\widehat{A}^{(k)}$ computed after $k-1$ steps.

The column-wise bound in Theorem 2.3.2 reflects the invariance of QR factorization under column scaling. Often only the weaker form $\|\Delta A\|_F \le \overline{\gamma}_{mn}\|A\|_F$ is given, which easily follows from the column-wise bound and (1.1.69).

Note that the matrix $\widetilde{Q}$ in Theorem 2.3.2, which is exactly orthogonal, *is not computed by the algorithm*. Denote by $\widehat{Q}$ the matrix computed by (2.3.28). Then

$$\|\widehat{Q} - Q\|_F \le \sqrt{n}\gamma_{mn}, \tag{2.3.30}$$

which shows that $\bar{Q}$ is very close to the exactly orthogonal matrix Q.

When the matrix to be factorized has some structure with zero elements it may be advantageous to use a **Givens QR factorization**. In this algorithm, zero elements below the main diagonal are introduced one at a time from bottom to top and from left to right. An important example is the QR factorization of a Hessenberg matrix

$$H_n = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1,n-1} & h_{1,n} \\ h_{21} & h_{22} & \cdots & h_{2,n-1} & h_{2,n} \\ & h_{32} & \cdots & \vdots & \vdots \\ & & \ddots & h_{n-1,n-1} & h_{n-1,n} \\ & & & h_{n,n-1} & h_{n,n} \end{pmatrix} \in \mathbb{R}^{n\times n}.$$

This occurs as a subproblem in computing the bidiagonal factorization and the SVD (see Sect. 2.3.3) and in the QR algorithm for the unsymmetric eigenvalue problem.

Givens rotations are ubiquitous in matrix algorithms for transforming a matrix to a more compact form. To illustrate the rotation pattern, it is convenient to use a **Wilkinson diagram**. In a Wilkinson diagram $\times$ stands for a (potential) nonzero element and $\otimes$ for a nonzero element that has been zeroed out and $+$ for a nonzero element that has been introduced in the computations (if any). The first two steps of the Givens QR factorization of H_n are illustrated below for $n = 5$. First 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.:

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

In the Wilkinson diagram the arrows point to the rows that took part in the last rotation. After $n - 1$ steps all subdiagonal elements have been zeroed out and the QR factorization

$$Q^T H = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad Q^T = G_{n-1,n}\cdots G_{23}G_{12}, \tag{2.3.31}$$

has been obtained. The first step in the QR factorization takes $6n$ flops. All remaining steps are similar, but work on smaller and smaller matrices. The total work of this QR factorization is only about $\sum_{k=1}^{n-1} 6k \approx 3n^2$ flops. An important special case is when H_n is lower bidiagonal. Then only two diagonals are nonzero and the flop count for the factorization is linear in n; see Sect. 2.6.3.

As for Householder QR factorization, the factor Q is usually not explicitly formed. It suffices to store the rotations in order to be able to perform operations with Q. If the storage scheme described in Sect. 2.3.1 is used, then one (real) rotation can be stored in just one number.

Some applications require the QR factorization of a skinny matrix A with many thousands of rows but with much fewer columns. An example is provided by stationary video background subtraction, where the number of rows can exceed 100,000 and the number of columns is about 100; see Candès et al. [42, 2009].

We now show how to use the QR factorization to solve the linear least squares problem (2.1.2).

Theorem 2.3.3 *Let $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank}(A) = n$, have the* QR *factorization*

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}. \tag{2.3.32}$$

Then the unique solution x to $\min_x \|Ax-b\|_2$ and the corresponding residual vector $r = b - Ax$ are given by

$$\begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = Q^T b, \quad Rx = d_1, \quad r = Q\begin{pmatrix} 0 \\ d_2 \end{pmatrix}, \tag{2.3.33}$$

and hence $\|r\|_2 = \|d_2\|_2$.

Proof Since Q is orthogonal we have

$$\|Ax-b\|_2^2 = \left\|Q^T(Ax-b)\right\|_2^2 = \left\|\begin{pmatrix} Rx \\ 0 \end{pmatrix} - \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}\right\|_2^2 = \|Rx-d_1\|_2^2 + \|d_2\|_2^2.$$

Obviously the right-hand side is minimized if $Rx = d_1$. Using the orthogonality of Q we have $b = Qd = Q_1d_1 + Q_2d_2 = Ax + r$. Since $Q_1d_1 = Q_1Rx = Ax$ it follows that $r = Q_2d_2$. □

By Theorem 2.3.3, when R and the Householder reflectors $H_1, H_2, \ldots, H_n$ have been computed by Algorithm 2.3.3, the least squares solution x and residual r can be computed as follows:

$$\begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = H_n \cdots H_2H_1b, \qquad Rx = d_1, \tag{2.3.34}$$

$$r = H_1 \cdots H_{n-1}H_n\begin{pmatrix} 0 \\ d_2 \end{pmatrix}, \tag{2.3.35}$$

and $\|r\|_2 = \|d_2\|_2$. Note that the matrix Q is not explicitly formed.

Algorithm 2.3.5 computes the least squares solution x and residual r using the Householder QR factorization. The operation count for the Householder QR factorization is $2n^2(m-n/3)$ flops. To compute Q^Tb and solve $Rx = d_1$ requires a further

$4n(m-n/4)$ flops. If not only $\|r\|_2$, but also r is wanted, another $4n(m-n/2)$ flops are needed. 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. For $m=n$ this is the same as for the Householder QR method, but for $m \gg n$ the Householder method is twice as expensive.

Algorithm 2.3.5 (*Least Squares Solution by Householder QR*)

```
function [x,r,rho] = housels(A,b);
% HOUSELS computes the solution x, the residual
%   r and rho = ||r||_2 to the full rank linear
%   least squares problem min||Ax - b||_2
% ------------------------------------------------
[m,n] = size(A);
[U,R,beta] = houseqr(A);
for k = 1:n
   c = beta(k)*(U(k:m,k)'*b(k:m));
   b(k:m) = b(k:m) - c*U(k:m,k);
end
x = R\b(1:n); r = [zeros(n,1); b(n+1:m)];
rho = norm(r);
for k = n:-1:1
   c = beta(k)*(U(k:m,k)'*r(k:m));
   r(k:m) = r(k:m) - c*U(k:m,k);
end
```

The Householder QR algorithm and the resulting method for solving the least squares problem are backward stable and the following result holds.

Theorem 2.3.4 *Let* $\min_x \|Ax-b\|_2$ *be a least squares problem where* $A \in \mathbb{R}^{m\times n}$ *has full column rank. Let* $\widehat{x}$ *be the solution computed using* (2.3.33) *and the Householder* QR *algorithm. Then* $\widehat{x}$ *is the exact least squares solution to a slightly perturbed least squares problem* $\min_x \|(A+\delta A)x-(b+\delta b)\|_2$, *where*

$$\|\delta A\|_F \le n\gamma_{mn}\|A\|_F, \qquad \|\delta b\|_2 \le \gamma_{mn}\|b\|_2. \tag{2.3.36}$$

Proof The result follows from Theorems 19.5 and 20.3 Higham [162, 2002]. □

The backward stability means that the computed residual $\bar{r}$ satisfies

$$(A+E)^T\bar{r}=0, \quad \|E\|_2 \le cu\|A\|_2, \tag{2.3.37}$$

for some constant $c=c(m,n)$. Hence, $A^T\bar{r}=-E^T\bar{r}$, and

$$\|A^T\bar{r}\|_2 \le cu\|\bar{r}\|_2\|A\|_2. \tag{2.3.38}$$

Note that this is a 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. Since $A^T r = 0$, we get from (1.4.10)

$$|A^T\widetilde{r}| < \gamma_{n+1}|A^T|(|b| + |A||x|).$$

From this follows the normwise bound

$$\|A^T\widetilde{r}\|_2 \le n^{1/2}\gamma_{n+1}\|A\|_2(\|b\|_2 + n^{1/2}\|A\|_2\|x\|_2),$$

which is much weaker than (2.3.38), in particular when $\|\bar{r}\|_2 \ll \|b\|_2$.

As shown in Sect. 2.1.2, a more general class of least squares problems are characterized by the augmented system (2.1.15). The algorithm using QR factorization given in Theorem 2.3.3 for the standard least squares problem can easily be generalized to solve the augmented system (2.1.15).

Theorem 2.3.5 *Assume that* $A \in \mathbb{R}^{m\times n}$ *has full column rank and* QR *factorization* (2.3.32). *Then the solution to the augmented system is given by*

$$R^T z_1 = c, \qquad \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = Q^T b, \tag{2.3.39}$$

$$Rx = (d_1 - z_1), \qquad y = Q\begin{pmatrix} z_1 \\ d_2 \end{pmatrix}. \tag{2.3.40}$$

Proof Using the QR factorization, the subsystems $y + Ax = b$ and $A^T y = c$ of the augmented system can be written

$$y + Q\begin{pmatrix} R \\ 0 \end{pmatrix} x = b, \qquad \begin{pmatrix} R^T & 0 \end{pmatrix} Q^T y = c.$$

Multiplying the first system by Q^T and defining $z = Q^T y$ and $d = Q^T b$ gives

$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} + \begin{pmatrix} R \\ 0 \end{pmatrix} x = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}, \qquad R^T z_1 = c.$$

Hence, $z_2 = d_2$, $Rx = d_1 - z_1$, and $y = Qz$. □

Setting $b = 0$ in (2.3.39)–(2.3.40) gives

$$R^T z_1 = c, \qquad y = Q \begin{pmatrix} z_1 \\ 0 \end{pmatrix}, \tag{2.3.41}$$

where y is the solution to the minimum-norm problem

$$\min \|y\|_2, \quad \text{subject to} \quad A^T y = c.$$

Note that either x or y can be computed without the other. Thus the algorithm (2.3.39)–(2.3.40) can be used to solve either the linear least squares problem (2.1.16) or the conditional least squares problem (2.1.17).

The systematic use of orthogonal transformations to reduce matrices to simpler form was initiated in 1958 by Givens [123, 1958] and Householder [169, 1958]. The application of these transformations to the linear least squares problem is due to Golub [126, 1965], who showed how to compute the QR factorization of a rectangular matrix A using Householder reflectors and column pivoting. An Algol implementation of this method is given by Businger and Golub [39, 1965].

2.3.3 Golub–Kahan Bidiagonalization

We now show that any rectangular matrix $A \in \mathbb{C}^{m\times n}$, $m \geq n$, can be reduced to real bidiagonal form

$$U^H A V = \begin{pmatrix} B \\ 0 \end{pmatrix}, \quad B = \begin{pmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \ddots & \ddots & \\ & & & \rho_{n-1} & \theta_n \\ & & & & \rho_n \end{pmatrix}. \tag{2.3.42}$$

by unitary transformations $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ from left and right. This algorithm is due to Golub[5] and Kahan [127, 1965].

In the **Golub–Kahan** Householder (GKH) bidiagonalization algorithm U and V are constructed as products of Householder transformations (see Sect. 2.3.1)

[5] Gene H. Golub (1932–2007), American mathematician and a pioneer in modern matrix computations. Golub studied at the University of Illinois, where he learned to program for the ILLIAC computer from David Wheeler. His thesis on using Chebyshev polynomials for solving linear equations was supervised by Abe Taub. After a postdoc year at Cambridge, England, Golub was recruited in 1962 by George Forsythe to Stanford University, where he remained for the rest of his life. His influential book entitled *Matrix Computations* [133, 1996], coauthored with C. F. Van Loan and now in its fourth edition, has sold more than 50,000 copies. For a more detailed biography of Golub together with reprints of his most important papers, see [48, 2007].

$$U = (u_1, \ldots, u_m) = Q_1 Q_2 \cdots Q_n,$$
$$V = (v_1, \ldots, v_n) = P_0 P_1 \cdots P_{n-2}, \tag{2.3.43}$$

applied alternately from right and left. Here P_0 can be chosen so that $P_0 e_1 = V e_1 = v_1$ is an arbitrary unit vector. In many cases one simply takes $P_0 = I$, i.e., this transformation is skipped. The following transformations are uniquely defined. Q_1 is chosen to zero the elements below the diagonal in the first column of AP_0 and P_1 to zero the last $n-2$ elements in the first row of $Q_1 A P_0$. The key thing to note is that P_1 will leave the first column in $Q_1 A$ unchanged and thus will not affect the zeros introduced by Q_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 the Householder reflector Q_k zeros the last $m-k$ elements in the kth column of $A^{(k)}$ and P_k zeros the last $n-(k+1)$ elements in the kth row of $Q_k A^{(k)}$. This determines the elements ρ_k and θ_k in the kth row of B. The process is continued until there are no more rows or columns to be treated.

The reduction can always be carried through, although some of the elements θ_k, ρ_k in B may vanish. When $m = n$, the zero block in (2.3.42) is empty. Note that from the construction it follows that

$$u_k = Q_1 \cdots Q_k e_k, \qquad v_k = P_0 \cdots P_{k-1} e_k, . \tag{2.3.44}$$

As long as no zero bidiagonal elements occur, the bidiagonal matrix B and the first n columns of U and V are uniquely determined by the choice of v_1. In case $m < n$, the decomposition will instead terminate with a bidiagonal matrix $\begin{pmatrix} B & 0 \end{pmatrix}$, where

$$B = \begin{pmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \ddots & \ddots & \\ & & & \rho_{m-1} & \theta_m \end{pmatrix} \in \mathbb{R}^{(m-1)\times m}$$

is rectangular. This upper bidiagonal matrix can be reduced to a square lower bidiagonal matrix by a sequence Givens rotations from the right, i.e., by flipping; see Sect. 2.4.5.

The bidiagonal form is the closest to diagonal form that can be achieved for a general matrix by a finite number of unitary transformations of A. An important use of the decomposition (2.3.42) is as a preprocessing step in computing the SVD of A; see Sect. 3.5.3. It is also a powerful tool for analyzing and solving various least squares problems.

Algorithm 2.3.6 computes the upper bidiagonal decomposition of $A = U^H B V \in \mathbb{C}^{m\times n}$ $(m \geq n)$. For simplicity it assumes that $m \geq n$ and takes $P_0 = I$. The Householder vectors associated with U are stored in the lower triangular and those

associated with V in the upper triangular part of the array holding B. By applying the algorithm to A^T, A can be reduced to *lower* bidiagonal form.

Algorithm 2.3.6 (*Upper Bidiagonal Decomposition*)

```
function [A,rho,theta] = bidiagu(A)
% BIDIAGU computes the upper bidiagonal decomposition
%   A = U^TBV of the m by n, matrix A (m >= n).
%   The diagonals are returned in rho and theta and the
%   Householder reflectors of U and V stored in A.
% ---------------------------------------------------
[m,n] = size(A);
rho = zeroes(n,1); theta = zeroes(n-1,1);
for k = 1:n
% Apply left transformation
   if k < m,
      [u(k:m), beta, rho(k)] = houseg(A(k:m,k));
       A(k:m,k+1:n) = A(k:m,k+1:n)- ...
                   beta*u(k:m)*(u(k:m)'*A(k:m,k+1:n));
       A(k+1:m,k) = u(k+1:m);  A(k,k) = beta;
    elseif k == m,  rho(m) = A(m,m);
    end
% Apply right transformation
   if k+1 < n,
    [v(k+1:n), gamma, theta(k+1)] = houseg(A(k,k+1:n)');
    A(k+1:m,k+1:n) = A(k+1:m,k+1:n) - ...
                     gamma*(A(k+1:m,k+1:n)*v(k+1:n))*
                          v(k+1:n)';
    A(k,k+2:n) = v(k+2:n)';  A(k,k+1) = gamma;
    elseif k +1 == n,  theta(n) = A(n-1,n);
    end
end
```

The bidiagonal reduction requires approximately $4(mn^2 - \frac{1}{3}n^3)$ flops when $m \geq n$. This is roughly twice as much as for a Householder QR factorization. If $U_1 = (u_1, \ldots, u_n)$ and/or $V = (v_1, \ldots, v_n)$ are explicitly required, then the corresponding products of Householder transformations can be accumulated at a cost of $2(mn^2 - \frac{1}{3}n^3)$ and $\frac{4}{3}n^3$ flops, respectively. If A is square, $m = n$, these counts are $\frac{8}{3}n^3$ for the reduction and $\frac{4}{3}n^3$ for computing each of the matrices U and V.

The GKH algorithm 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

$$\|E\|_F \leq cn^2 u \|A\|_F \tag{2.3.45}$$

and c is a constant of order unity. Moreover, if we use the information stored to generate U and V, the computed matrices are close to the exact matrices U and V that 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.

When $m \gg n$ it is more efficient to use a two-step procedure as originally suggested by Lawson and Hanson [190, 1974] p. 119, and later analyzed by Chan [43, 1982]. In the first step the QR factorization of A is computed (possibly using column pivoting),

$$AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R \in \mathbb{R}^{n \times n},$$

which requires $2n^2(m - \frac{1}{3}n)$ 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 $\frac{8}{3}n^3$ flops. The complete reduction to bidiagonal form then takes a total of $2n^2(m+n)$ flops. The flop counts for the two variants are equal when $m + n = 2m - 2n/3$, or when $m = 5/3n$. When $m/n > 5/3$, Chan's version is more efficient than the original Golub–Kahan algorithm. It is potentially more accurate if column pivoting is used in the initial QR factorization.

If Givens rotations are used, it is possible to take advantage of the zeros in the bidiagonalization of R. The elements are zeroed by diagonals from outside in. In each diagonal zeroes are introduced from top to bottom. An intermediate step is shown below. The element (2,5) is zeroed by a rotation of columns (4,5). This introduces a new nonzero element in position (5,4), which in turn is zeroed by a rotation of rows (4,5). Next, the element (3,6) will be zeroed by a rotation of columns (5,6), etc.:

$$\begin{array}{c} \\ \\ \rightarrow \\ \rightarrow \\ \\ \\ \end{array} \begin{array}{c} \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccc} & & & \downarrow & \downarrow & \\ \end{array} \\ \begin{pmatrix} \times & \times & \times & 0 & 0 & 0 \\ & \times & \times & \times & \otimes & 0 \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & \oplus & \times & \times \\ & & & & & \times \end{pmatrix}. \end{array}$$

This reduction is of more general interest, since it can also be used to reduce the bandwidth of a triangular matrix. The cost of zeroing one element is $6n$ flops and the operation count for the reduction $2n^3$ flops. This is lower than for the Householder algorithm, but if the products of the left or right transformations are to be accumulated, Givens method requires more work.

The upper bidiagonal decomposition can be used to solve the least squares problem $\min \|Ax - b\|_2$, where $A \in \mathbb{R}^{m \times n}$, $m \geq n$. If A has full column rank, then the upper bidiagonal matrix B in (2.3.42) has nonzero diagonal elements. Setting $x = Vy$ and

using the orthogonal invariance of the 2-norm, we have

$$\|Ax - b\|_2^2 = \|U^T AVy - U^T b\|_2^2 = \left\| \begin{pmatrix} B \\ 0 \end{pmatrix} y - \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \right\|_2^2$$
$$= \|By - c_1\|_2^2 + \|c_2\|_2^2,$$

where

$$c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = U^T b = Q_n \cdots Q_2 Q_1 b, \quad c_1 \in \mathbb{R}^n. \tag{2.3.46}$$

Hence, the minimum of $\|Ax - b\|_2^2$ equals $\|c_2\|_2$ and is obtained for $x = Vy$, where y satisfies the bidiagonal system $By = c_1$ and

$$x = P_0 P_1 \cdots P_{n-2} y, \qquad r = Q_1 Q_2 \cdots Q_n \begin{pmatrix} 0 \\ c_2 \end{pmatrix}. \tag{2.3.47}$$

Forming c and x requires $4mn$ flops. Solving $By = c_1$ by back substitution,

$$y_n = c_n/\rho_n, \qquad y_k = (c_k - \theta_{k+1} y_{k-1})/\rho_k, \quad k = n-1:-1:1, \tag{2.3.48}$$

requires only $3n - 2$ flops. If r is wanted this requires an additional $4mn - 2n^2$ flops.

Since U and V are orthogonal, the singular values of R are equal to those of A, and $\kappa_2(A) = \kappa_2(R)$. An estimate of the smallest singular value can be obtained by performing one or more steps of inverse iteration with $B^T B$. Let u be a suitably chosen vector and compute v and w from (cf. (2.3.83))

$$B^T v = u, \qquad Bw = v \tag{2.3.49}$$

by forward and backward substitution. Then $\sigma_n^{-1} \approx \|w\|_\infty / \|v\|_\infty$ will usually be a good estimate of σ_n^{-1} at a cost of less than $6n$ flops.

Barlow et al. [9, 2002] and [11, 2005] give a potentially faster algorithm for computing this decomposition.

2.3.4 Gram–Schmidt QR Factorization

Let $\{a_n\}$ be a linearly independent sequence of elements of a finite- or infinite-dimensional inner-product space. **Gram-Schmidt orthogonalization**[67] is a process that constructs a related orthogonal sequence $\{q_n\}$ by defining q_n inductively as

---

[6] Jørgen Pedersen Gram (1850–1916), Danish mathematician. Gram worked for Hafnia Insurance Company and made contributions to probability and numerical analysis.

[7] Erhard Schmidt (1876–1959) was born in Dorpat, Estonia. He obtained his doctoral degree from the University of Göttingen in 1905 under Hilbert's supervision. After holding positions in Zürich, Erlangen, and Breslau he assumed a position at the University of Berlin in 1917. Here he was

$$q_1 = a_1, \qquad q_n = a_n - \sum_{k=1}^{n-1} \frac{(q_k, a_n)}{\|q_k\|_2^2} q_k, \quad n \geq 2. \tag{2.3.50}$$

Replacing each q_n by $q_n/\|q_n\|_2^2$ gives an orthonormal sequence. By construction,

$$\text{span}\{q_1, \ldots, q_k\} = \text{span}\{a_1, \ldots, a_k\}, \quad k \geq 1. \tag{2.3.51}$$

Having an orthogonal basis for this nested sequence of subspaces simplifies many operations and applications of the Gram-Schmidt process are ubiquitous in mathematics.

Given a matrix $A \in \mathbb{C}^{m\times n}$ with linearly independent columns $a_1, a_2, \ldots, a_n$, the Gram–Schmidt process computes an orthonormal basis $q_1, q_2, \ldots, q_n$ for the column space of A. Then each column vector a_k, $k = 1{:}n$, in A can be expressed as

$$a_k = r_{1k}q_1 + r_{2k}q_2 + \cdots + r_{kk}q_k, \quad r_{kk} \neq 0, \quad k = 1{:}n. \tag{2.3.52}$$

Assume that $q_1, q_2, \ldots, q_{k-1}$ have been determined. Multiplying (2.3.52) by q_j^H from the left and using orthogonality it follows that $q_j^H a_k = r_{jk}$, $j = 1{:}k-1$. If we set

$$\widehat{q}_k \equiv r_{kk}q_k = a_k - \sum_{j=1}^{k-1} r_{jk}q_j \tag{2.3.53}$$

then $\widehat{q}_k \neq 0$, since otherwise a_k would be a linear combination of $a_1, \ldots, a_{k-1}$, which contradicts our assumption. Hence,

$$q_k = \widehat{q}_k/r_{kk}, \quad r_{kk} = \|\widehat{q}_k\|_2 = (\widehat{q}_k^H \widehat{q}_k)^{1/2} \tag{2.3.54}$$

is the desired vector.Note that the term $r_{jk}q_j$ subtracted from a_k is equal to $P_j a_k$, where $P_j = q_j q_j^H$ is the orthogonal projector onto the subspace spanned by q_j. It follows that $P_j^2 = P_j$ and $P_j^{\perp} = I - q_j q_j^H$ is the orthogonal projector onto the orthogonal complement.

Theorem 2.3.6 *Let the matrix $A = (a_1, a_2, \ldots, a_n) \in \mathbb{C}^{m\times n}$ have linearly independent columns. Then the Gram–Schmidt algorithm computes a matrix $Q_1 \in \mathbb{C}^{m\times n}$ with orthonormal columns $Q_1^H Q_1 = I_n$ and an upper triangular matrix $R \in \mathbb{C}^{n\times n}$ with real positive diagonal elements, such that*

---

instrumental in setting up the Institute of Applied Mathematics and establishing Berlin as a leading center for applied mathematics.

$$A = (q_1, q_2, \ldots, q_n) \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \vdots \\ & & & r_{nn} \end{pmatrix} \equiv Q_1 R. \tag{2.3.55}$$

Proof Combining (2.3.52) and (2.3.54) 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 (2.3.55). Since the vectors q_k are mutually orthogonal by construction, the theorem follows. □

In matrix terms the Gram–Schmidt process uses elementary column operations to transform the matrix A into an orthogonal matrix Q. The matrix Q in the thin QR factorization is formed explicitly. This is in contrast to the Householder QR factorization, where A is premultiplied by a sequence of elementary orthogonal transformations to produce R and Q (in product form) in the full QR factorization.

There are two mathematically equivalent variants of the Gram–Schmidt process. (We say that two formulas or algorithms are mathematically equivalent if they produce the same result in exact arithmetic.) Although these variants differ only in the order in which the operations are carried out, their numerical stability properties differ greatly.

In the Classical Gram–Schmidt (CGS) algorithm, we set $q_1 = a_1/r_{11}$, where $r_{11} = \|a_1\|_2$, and for $k = 2{:}n$, orthogonalize a_k against previous vectors $q_1, \ldots, q_{k-1}$:

$$\widehat{q}_k = a_k - \sum_{i=1}^{k-1} r_{ik} q_i, \quad r_{ik} = q_i^H a_k. \tag{2.3.56}$$

Here $\widehat{q}_k$ is the orthogonal projection of a_k onto the complement of $\operatorname{span}\{a_1, \ldots, a_{k-1}\}$. If $r_{kk} \neq 0$, we set $q_k = \widehat{q}_k / r_{kk}$, where $r_{kk} = \|\widehat{q}_k\|_2$. The elements in R are generated column by column. Algorithm 2.3.7 computes the factorization $A = Q_1 R$ by CGS, provided $\operatorname{rank}(A) = n$.

Algorithm 2.3.7 (*Classical Gram–Schmidt*)

```
function [Q,R] = cgs(A);
% CGS computes the thin QR factorization
%   of A using the CGS algorithm
% -------------------------------------
[m,n] = size(A);
Q = A; R = zeros(n);
for k = 1:n
   R(1:k-1,k) = Q(:,1:k-1)'*A(:,k);
   Q(:,k) = A(:,k)- Q(:,1:k-1)*R(1:k-1,k);
   R(k,k) = norm(Q(:,k));
   Q(:,k) = Q(:,k)/R(k,k);
end
```

In the Modified Gram–Schmidt (MGS) algorithm, we set $A^{(1)} = A$, and for $k = 1{:}n$ compute

$$q_k = a_k^{(k)}/r_{kk}, \quad r_{kk} = \|a_k^{(k)}\|_2.$$

We then orthogonalize $a_j^{(k)}$, $j > k$, against q_k: $a_j^{(k+1)} = (I - q_k q_k^T)a_j^{(k)}$, where $P_k = (I - q_k q_k^T)$ is an elementary orthogonal projector:

$$a_j^{(k+1)} = a_j^{(k)} - r_{kj} q_k, \quad r_{kj} = q_k^T a_j^{(k)}, \quad j = k+1{:}n. \tag{2.3.57}$$

After k steps we have computed

$$A^{(k)} = \left(q_1, \ldots, q_k, a_k^{(k+1)}, \ldots, a_n^{(k+)}\right),$$

where $a_k^{(k+1)}, \ldots, a_n^{(k+1)}$ are orthogonal to $q_1, \ldots, q_k$. As described, the elements in R are computed row by row. Given $A \in \mathbb{R}^{m\times n}$ with rank $(A) = n$, Algorithm 2.3.8 computes the factorization $A = Q_1 R$, by MGS.

Algorithm 2.3.8 (*Row-Wise Modified Gram–Schmidt*)

```
function [Q,R] = mgs(A);
% MGS computes the thin QR factorization
%   using the MGS algorithm
% ----------------------------------------
[m,n] = size(A);
Q = A; R = zeros(n);
for k = 1:n
   R(k,k) = norm(Q(:,k));
   Q(:,k) = Q(:,k)/R(k,k);
   R(k,k+1:n) = Q(:,k)'*Q(:,k+1:n);
   Q(:,k+1:n) = Q(:,k+1:n) - Q(:,k)*R(k,k+1:n);
end
```

Table 2.1 Condition number and loss of orthogonality in 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

The CGS and MGS algorithms both require approximately $2mn^2$ flops. This is $2n^3/3$ flops more than for Householder QR factorization if Q is kept in product form. The Gram–Schmidt algorithms also need extra storage for R. When $m \gg n$, the extra work and storage are negligible. Such a matrix is often called "skinny".

The difference in numerical stability between CGS and MGS is due to the fact that in MGS *the orthogonalizations are carried out using a product of elementary orthogonal projectors*:

$$r_{kk} q_k = (I - q_{k-1} q_{k-1}^T) \cdots (I - q_1 q_1^T) a_k. \tag{2.3.58}$$

The projections $r_{ik} q_i$ $(i = 1{:}k-1)$ are subtracted from a_k as soon as they are computed, whereas in CGS

$$r_{kk} q_k = (I - Q_{k-1} Q_{k-1}^T) a_k, \quad Q_{k-1} = (q_1, \ldots, q_{k-1}). \tag{2.3.59}$$

For $k > 2$ the formulas (2.3.58) and (2.3.59) are identical only provided that $q_1, \ldots, q_{k-1}$ are exactly orthogonal. In floating point arithmetic the rounding errors are different and MGS has superior numerical properties compared to CGS.

To illustrate the difference in stability between MGS and CGS, a matrix $A \in \mathbb{R}^{50\times 10}$ with singular values $\sigma_i = 10^{-i+1}$, $i = 1{:}10$, was generated by computing

$$A = U\, DV^T, \quad D = \mathrm{diag}(1, 10^{-1}, \ldots, 10^{-9})$$

with U and V orthonormal matrices. Table 2.1 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 $\|I_k - Q_k^T Q_k\|_2$. For MGS the loss of orthogonality is more gradual than for CGS and proportional to $\kappa(A_k)$. The loss of orthogonality in Gram–Schmidt orthogonalization is studied in more detail in Sect. 2.3.5.

An important property of all Gram–Schmidt algorithms is their invariance under column scaling. The Gram–Schmidt algorithms applied to the scaled matrix $\widetilde{A} = AD$ yield the factors $\widetilde{Q} = Q$ and $\widetilde{R} = RD$ for any positive diagonal matrix D. This is true even in finite precision arithmetic, provided the scaling is done without error.

Let $\bar{Q}_1$ and $\bar{R}$ be the computed factors from MGS. Then, by an elementary error analysis, the following bound for the backward error can be established:

$$A + E = \bar{Q}_1 \bar{R}, \quad \|E\|_2 \leq c_0 u \|A\|_2, \tag{2.3.60}$$

where the factor c_0 roughly equals $2(mn)^2$. This ensures that the product $\bar{Q}_1 \bar{R}$ represents A to working accuracy.

In Gram–Schmidt QR factorization one works with vectors of constant length, which is not the case for Householder QR factorization. This is sometimes an advantage for parallel implementation The implementation of MGS and CGS for a complex matrix $A \in \mathbb{C}^{m\times n}$ is straightforward, whereas the representation of complex Householder reflectors is less obvious; see Lehoucq [191, 1996].

The row-wise generation of R in MGS has the important advantage that it allows for column interchanges (see Sect. 2.4.2). However, it cannot be used in applications, where the columns a_k are generated one at a time. Algorithm 2.3.9 implements a column-wise version of MGS that is numerically equivalent to the row-wise version. Although the sequencing of the operations is different, the rounding errors are the same.

Algorithm 2.3.9 (*Column-Wise Modified Gram–Schmidt*) *Given $A \in \mathbb{R}^{m\times n}$ with rank$(A) = n$ the following algorithm computes the factorization $A = Q_1 R$, where R is generated by columns.*

```
function [Q,R] = mgsc(A);
% MGSC computes the thin QR factorization
% of A using the column-wise MGS algorithm
% ---------------------------------------
[m,n] = size(A);
Q = A; R = zeros(n);
for k = 1:n
   for i = 1:k-1
      R(i,k) = Q(:,i)'*Q(:,k);
      Q(:,k) = Q(:,k) - Q(:,i)*R(i,k);
   end
   R(k,k) = norm(Q(:,k));
   Q(:,k) = Q(:,k)/R(k,k);
end
```

When MGS is used correctly it gives backwards stable solutions and residuals to linear least squares problem. However, unless it is used correctly, the loss of orthogonality in the computed Q_1 can spoil the accuracy. An algorithm seen in some

textbooks, computes $c = Q_1^T b$ and then solves $Rx = c$. *This procedure should not be used.* Instead b should be treated as an $(n+1)$st column appended to A. Set $b^{(1)} = b$, and for $k = 1{:}n$, use elementary orthogonal projectors to compute $c = (c_1, \ldots, n)^T$:

$$b^{(k+1)} = (I - q_k q_k^T) b^{(k)} = b^{(k)} - c_k q_k, \quad c_k = q_k^T b^{(k)}, \tag{2.3.61}$$

and $r = b^{(n)}$. This will give the factorization

$$\begin{pmatrix} A & b \end{pmatrix} = \begin{pmatrix} Q_1 & r \end{pmatrix} \begin{pmatrix} R & c \\ 0 & 1 \end{pmatrix}. \tag{2.3.62}$$

By (2.3.60) the product of the computed factors accurately reproduces the matrix $(A,\ b)$. It follows that

$$\|Ax - b\|_2 = \left\| \begin{pmatrix} A & b \end{pmatrix} \begin{pmatrix} x \\ -1 \end{pmatrix} \right\|_2 = \|Q_1(Rx - c) - r\|_2.$$

If $Q_1^T r = 0$, the minimum of the last expression occurs when $Rx - c = 0$. It is not necessary to require that Q_1 is accurately orthogonal for this conclusion to hold; see Björck [26, 1994].

Algorithm 2.3.10 (*Linear Least Squares Solution by MGS*)

```
function [x,r,rho] = mgsls(A,b);
% MGSLS uses MGS QR factorization of A to solve
%   the least squares problem min||Ax - b||_2.
%   It returns x, r, and rho = \|r\|_2.
% -----------------------------------------------
[m,n] = size(A); d = zeros(n,1);
[Q,R] = mgs(A);      % Apply MGS to A.
for k = 1:n
   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);
```

Algorithm 2.3.10 computes the solution x to the linear least squares problem $\min_x \|Ax - b\|_2$, the residual r, and its Euclidean norm. It is assumed that $A \in \mathbb{C}^{m\times n}$ has full column rank and one uses MGS to compute Q and R. In the last loop the computed residual is reorthogonalized against the vectors q_k. Why this is done in backward order is explained in Sect. 2.3.6. If only x and the residual norm $\|r\|$ are

needed, then the last loop can be skipped and only $2n(m+n)$ flops are needed for each right-hand side.

Like the corresponding Householder algorithm, Algorithm 2.3.10 is backward stable also for computing the residual r. This means that the computed residual $\bar{r}$ satisfies

$$\|A^T\bar{r}\|_2 \le \gamma_{mn}\|\bar{r}\|_2\|A\|_2. \tag{2.3.63}$$

which is much better than if the residual is computed from its definition $r = b - Ax$ using the computed solution x. The proof of backward stability depends on a remarkable connection between MGS and Householder QR factorization, which is described in Sect. 2.3.6.

The different computational the variants of Gram–Schmidt procedure have an interesting history. What is now called the "classical" Gram–Schmidt algorithm first appeared explicitly in papers by Gram [138, 1879] and Schmidt [251, 1907]. Schmidt treats the solution of linear systems with infinitely many unknowns and uses the orthogonalization as a theoretical tool rather than a computational procedure. The "modified" Gram–Schmidt algorithm is related to an algorithm used by Laplace in 1816. But Laplace did not interpret his algorithm in terms of orthogonalization, nor did he use it for computing least squares solutions. In 1853 Bienaymé gave a similar derivation of a slightly more general algorithm.

In the 1950s, algorithms based on Gram–Schmidt orthogonalization were frequently used, although their numerical properties were not well understood at the time. The superior properties of MGS compared to CGS were experimentally established by Rice [243, 1966]. A roundoff analysis by Björck [22, 1967] proved the forward stability of MGS for solving linear least squares problems.

2.3.5 Loss of Orthogonality and Reorthogonalization

We now analyze the loss of orthogonality in the Gram–Schmidt process when it is used to orthogonalize two linearly independent vectors a_1 and a_2 in $\mathbb{R}^n$. Since rounding errors in the normalization of vectors have a negligible effect on the the loss of orthogonality we assume, without loss of generality, that a_1 and a_2 have unit length.

By this assumption, $q_1 = a_1, r_{11} = 1$. Using the standard model for floating point computation and the basic results in Sect. 1.4.2, an upper bound for the error in the *computed* scalar product $\bar{r}_{12} = fl(q_1^T a_2)$ is

$$|\bar{r}_{12} - r_{12}| < \gamma_m\|a_2\|_2, \quad \gamma_m = \frac{m\mathbf{u}}{1 - m\mathbf{u}/2},$$

where $\mathbf{u}$ is the unit roundoff. For the error in the computed unnormalized vector $\bar{q}_2 = fl(a_2 - fl(\bar{r}_{12}q_1))$ we obtain

$$\|\bar{q}_2 - \widehat{q}_2\|_2 < \gamma_{m+2}|\bar{r}_{12}| < \gamma_{m+2}\|a_2\|_2 = \gamma_{m+2}$$

where $\bar{q}_2$ denotes the true result). Since $q_1^T \widehat{q}_2 = 0$, it follows that $|q_1^T \bar{q}_2| = |q_1^T (\bar{q}_2 - \widehat{q}_2)| < \gamma_{m+2}$. Hence, if we set $r_{22} = \|\bar{q}_2\|_2$ the loss of orthogonality can be bounded by

$$\frac{|q_1^T \bar{q}_2|}{\|\bar{q}_2\|_2} < \frac{\gamma_{m+2}}{r_{22}} \tag{2.3.64}$$

and can be severe when $r_{22} \ll 1$. Since $r_{22} = \sin \angle(a_1, a_2)$, this will be the case when the angle between a_1 and a_2 is small. In general, we conclude that when

$$r_{kk} = \|a_k^{k)}\|_2 \ll \|a_k\|_2. \tag{2.3.65}$$

a severe loss of orthogonality may have occured in the Gram–Schmidt process.

Example 2.3.2 Consider the extremely ill-conditioned matrix in Example 1.4.1:

$$A = (a_1, a_2) = \begin{pmatrix} 1.2969 & 0.8648 \\ 0.2161 & 0.1441 \end{pmatrix}.$$

Applying the Gram–Schmidt algorithm in IEEE double precision to A gives

$$q_1 = \begin{pmatrix} 0.98640009002732 \\ 0.16436198585466 \end{pmatrix}, \qquad r_{12} = q_1^T a_2 = 0.87672336001729.$$

Subtracting the orthogonal projection onto q_1 we get $\bar{q}_2 = a_2 - r_{12} q_1 =$

$\begin{pmatrix} -0.12501091273265 \\ 0.75023914025696 \end{pmatrix} 10^{-8}$. Normalizing this vector gives

$$q_2 = \begin{pmatrix} -0.1643619607147 \\ 0.9864000942164 \end{pmatrix}, \quad R = \begin{pmatrix} 1.3147809018996 & 0.8767233600173 \\ 0 & 0.0000000076058 \end{pmatrix}.$$

Massive cancellation has taken place in computing $\bar{q}_2$, leading to a serious loss of orthogonality between q_1 and q_2: $q_1^T q_2 = 2.5486557 \cdot 10^{-8}$, which should be compared with the unit roundoff $\mathbf{u} \approx 1.11 \cdot 10^{-16}$. Note that 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 2.3.7 *Let $\bar{Q}$ and $\bar{R}$ denote the factors computed by the* MGS *algorithm. Then for some $c_1 = c_1(m, n)$,*

$$\|I - \bar{Q}_1^T \bar{Q}_1\|_2 \leq \frac{c_1 u \kappa_2(A)}{1 - c_1 u \kappa_2(A)}, \tag{2.3.66}$$

provided that $c_1\kappa_2(A)u < 1$,

Proof See Björck [22, 1967]. □

No similar bound for the loss of orthogonality exists for the CGS algorithm given above. Even computing $Q_1 = AR^{-1}$, where R is determined by the Cholesky factorization of A^TA, often gives better orthogonality than CGS. For a slightly altered version of CGS an upper bound proportional to κ^2 for the loss of orthogonality has recently been proved. Usually, the diagonal entry r_{kk} in the kth step of CGS is computed as

$$\bar{q}_k = a_k - \sum_{i=1}^{k-1} r_{ik}q_i, \quad r_{kk} = \|\bar{q}_k\|_2.$$

From the Pythagorean theorem it follows that $r_{kk}^2 + p_k^2 = s_k^2$, where

$$s_k = \|a_k\|_2, \qquad p_k = (r_{1k}^2 + \cdots + r_{k-1,k}^2)^{1/2}.$$

In the altered version the diagonal entry r_{kk} is computed as

$$r_{kk} = (s_k^2 - p_k^2)^{1/2} = (s_k - p_k)^{1/2}(s_k + p_k)^{1/2}. \tag{2.3.67}$$

Under the assumption that A^TA is not too ill-conditioned, the bound

$$\|I - \bar{Q}^T\bar{Q}\|_2 \le c_2(m,n)\kappa(A)_2^2 \tag{2.3.68}$$

was established by Smoktunowicz et al. [259, 2006] for this "Pythagorean variant".

Using the invariance of the Gram–Schmidt algorithm under column scaling a sharper upper bound for the loss of orthogonality is obtained. Let $\mathcal{D}$ is the set of all positive diagonal matrices. Then in (2.3.66) we can replace $\kappa_2(A)$ by

$$\widetilde{\kappa}_2 = \min_{D\in\mathcal{D}} \kappa_2(AD). \tag{2.3.69}$$

By (2.2.33), scaling A so that all column norms in A are equal will approximately minimize $\kappa_2(AD)$.

In some applications it may be essential that the computed columns of Q are orthogonal to working accuracy. For example, this is the case case in algorithms for solving unsymmetric eigenproblems, such as simultaneous iteration and Arnoldi methods. As we have seen, both the CGS and MGS algorithms fail to achieve this. A remedy to this is to enforce orthogonality by **reorthogonalization**. In selective reorthogonalization, a test is performed at each step to see whether or not it is necessary to reorthogonalize. An indication that cancellation has occurred is that

$$\|\bar{q}_k\|_2 < \alpha\|a_k\|_2 \tag{2.3.70}$$

for some chosen tolerance α. Typically α is chosen in the range $0.1 \le \alpha \le 1/\sqrt{2}$. 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. Rutishauser [247, 1967] was the first to use a condition of the form (2.3.70). He used $\alpha = 0.1$, i.e., when at least one decimal digit of accuracy has been lost due to cancellation reorthogonalization is applied. The choice $\alpha = 1/\sqrt{2}$ used by Daniel et al. [64, 1976] (see also Reichel and Gragg [240, 1990]) is most often recommended.

In principle, reorthogonalization can be applied several times. But if A has full numerical column rank, then one reorthogonalization step suffices to achieve orthogonality to unit roundoff level. An analysis for the case $n = 2$ due to Kahan and published by Parlett [230, 1998] shows that *"twice is enough"*. That is, unless the vectors are linearly dependent to machine precision, full orthogonality will be achieved. A similar result for the general case $n > 2$ is shown by Giraud et al. [122, 2005]. As an example, consider reorthogonalizing the computed vector $a_2^{(2)}$ in Example 2.3.2 against q_1. This gives $q_1^T q_2 = 2.5486557 \cdot 10^{-8}$ and

$$\widetilde{q}_2 = \begin{pmatrix} -0.16436198585466 \\ 0.98640009002732 \end{pmatrix},$$

where the vector $\widetilde{q}_2$ now is exactly orthogonal to q_1.

The simplest option if full orthogonality is desired is to *always perform a reorthogonalization*, even if that doubles the cost. In the two-iteration CGS2 algorithm applied to $A^{(0)} = A$, the vectors $a_k^{(0)}$, $k \ge 2$, are orthogonalized against the computed basis vectors $Q_{k-1} = (q_1, \ldots, q_{k-1})$ as follows: For $i = 1, 2$,

$$a_k^{(i)} = (I - Q_{k-1} Q_{k-1}^T) a_k^{(i-1)} = a_k^{(i-1)} - Q_{k-1}(Q_{k-1}^T a_k^{(i-1)}).$$

The new basis vector is then given as $q_k = a_k^{(2)} / \|a_k^{(2)}\|_2$.

Algorithm 2.3.11 (*CGS2*)

```
function [Q,R] = cgs2(A);
% CGS2 computes the thin QR factorization of
%   A using CGS with reorthogonalization.
% ----------------------------------------
[m,n] = size(A);
 Q = A; R = zeros(n);
 for k = 1:n
    for i = 1:2
       V =  Q(:,1:k-1)'*Q(:,k);
       Q(:,k) = Q(:,k) - Q(:,1:k-1)*V;
       R(1:k-1,k) = R(1:k-1,k) + V;
    end
    R(k,k) = norm(Q(:,k));
    Q(:,k) = Q(:,k)/R(k,k);
 end
```

A rounding error analysis of CGS2 by Giraud et al. [122, 2005] shows that if the matrix A has full numerical rank, then CGS2 will guarantee that the orthogonality of the computed basis vectors is close to the unit roundoff level. A similar algorithm for column-wise MGS with reorthogonalization has the same operation count, and also produces basis vectors with orthogonality close to unit roundoff. For column-wise MGS2 the inner loop is a vector operation whereas in CGS2 it is a matrix-vector operation. This means that CGS2 executes faster than MGS2 and it is therefore usually the preferred choice.

If failure occurs in step k of CGS2, this means that to within machine precision a_k is a linear combination of $q_1, , \ldots, q_{n-1}$, with coefficients given by the computed $r_{1k}, \ldots, r_{k-1,k}$. How to recover the orthogonalization is problem dependent. One option is not to generate a new vector q_k in this step, set $r_{kk} = 0$, and proceed to the next column. After a suitable permutation of columns this will generate a QR factorization where Q is $m \times (n-p)$ and R is $(n-p) \times n$ upper trapezoidal with nonzero diagonal entries. This factorization can be used to compute a pseudoinverse solution to a least squares problem; see Sect. 2.4.2. Other options are discussed in Daniel et al. [64, 1978] and Stewart [271, 1994].

Hoffman [165, 1989] reports extensive experiments with selective reorthogonalization using a range of α values, specifically, $\alpha = 1/2, 0.1, \ldots, 10^{-10}$. It is the Pythagorean variant of CGS was used in 1962 by Davis [66, 1962].

2.3.6 MGS as a Householder Method

Evidence is accumulating that the modified Gram–Schmidt method gives better results than Householder. The reasons for this phenomenon appear not to have been elucidated yet.

—G. Peters and J. H. Wilkinson [234, 1970]

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 extended with a square matrix of zero elements on top.[8] This is true not only in theory, but in the presence of rounding errors as well. We first look at the theoretical result.

Let $A \in \mathbb{R}^{m\times n}$ have rank n and consider the two QR factorizations

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad \widetilde{A} = \begin{pmatrix} O \\ A \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}, \tag{2.3.71}$$

where $Q \in \mathbb{R}^{m\times m}$ and $P \in \mathbb{R}^{(n+m)\times(n+m)}$ are orthogonal matrices. If the upper

[8] This observation was made by Charles Sheffield, apparently when comparing FORTRAN code for Householder and MGS QR factorization.

triangular matrices R and $\widetilde{R}$ are chosen to have positive diagonal elements, then by uniqueness $R = \widetilde{R}$. Hence, 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 QR applied to the extended matrix. To see this, recall that the Householder reflector $Px = \sigma e_1$ uses

$$P = I - 2vv^T/\|v\|_2^2, \quad v = x - \sigma e_1, \quad \sigma = \pm\|x\|_2.$$

If the second factorization in (2.3.71) is obtained using Householder reflectors, then

$$P^T = P_n \cdots P_2 P_1, \quad P_k = I - 2\widehat{v}_k\widehat{v}_k^T/\|\widehat{v}_k\|_2^2, \quad k = 1{:}n, \tag{2.3.72}$$

where the vectors $\widehat{v}_k$ are described below. Now, from MGS applied to $A^{(1)} = A$, $r_{11} = \|a_1^{(1)}\|_2$, and $a_1^{(1)} = \widehat{q}_1 = q_1 r_{11}$. Hence, for the first Householder reflector applied to the extended matrix

$$\widetilde{A}^{(1)} \equiv \begin{pmatrix} O \\ A^{(1)} \end{pmatrix}, \qquad \widetilde{a}_1^{(1)} = \begin{pmatrix} 0 \\ a_1^{(1)} \end{pmatrix},$$

the Householder vector is

$$\widehat{v}_1 \equiv \begin{pmatrix} -r_{11}e_1 \\ \widehat{q}_1 \end{pmatrix} = r_{11}v_1, \qquad v_1 = \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix}.$$

Since $\|v_1\|_2^2 = \|e_1\|_2^2 + \|q_1\|_2^2 = 2$, we have $P_1 = I - 2\widehat{v}_1\widehat{v}_1^T/\|\widehat{v}_1\|_2^2 = I - v_1 v_1^T$ and

$$P_1\widetilde{a}_j^{(1)} = \widetilde{a}_j^{(1)} - v_1 v_1^T \widetilde{a}_j^{(1)} = \begin{pmatrix} 0 \\ a_j^{(1)} \end{pmatrix} - \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix} q_1^T a_j^{(1)} = \begin{pmatrix} r_{1j}e_1 \\ a_j^{(2)} \end{pmatrix},$$

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 $a_j^{(2)}$, $j = 2{:}n$, are the same. The next Householder reflector produces the second row of R and $a_j^{(3)}$, $j = 3 : n$, just as in MGS. All remaining steps are similar and we conclude that this Householder QR is *numerically equivalent* to MGS applied to A. Note that every P_k is effectively

defined by the columns of Q_1, since

$$P_k = I - v_k v_k^T, \qquad v_k = \begin{pmatrix} -e_k \\ q_k \end{pmatrix}, \quad k = 1{:}n. \tag{2.3.73}$$

From the numerical equivalence it follows that the backward error analysis for the Householder QR factorization of the extended matrix can also be applied to the modified Gram–Schmidt algorithm on A. From the error analysis for Householder QR factorization (see Theorem 2.3.4, p. 261) it follows that for $\widehat{R}$ computed by MGS,

$$\begin{pmatrix} E_1 \\ A + E_2 \end{pmatrix} = \widetilde{P} \begin{pmatrix} \widehat{R} \\ 0 \end{pmatrix}, \quad \widehat{P} = \widetilde{P} + E',$$

where

$$\|E_i\|_2 \le c_i u \|A\|_2, \quad i = 1, 2, \quad \|E'\|_2 \le c_3 u. \tag{2.3.74}$$

Here 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 [31, 1992]) that there exists an *exactly orthonormal matrix* $\widehat{Q}_1$ and E such that

$$A + E = \widehat{Q}_1 \widehat{R}, \qquad \widehat{Q}_1^T \widehat{Q}_1 = I, \quad \|E\|_2 \le c_1 u \|A\|_2. \tag{2.3.75}$$

If $\bar{\sigma}_1 \ge \cdots \ge \bar{\sigma}_n$ are the singular values of $\widehat{R}$ 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 (2.3.75) shows that $\widehat{R}$ *computed by MGS is comparable in accuracy to the upper triangular matrix from the Householder QR factorization applied to* A.

The relationship between MGS and Householder QR can be used to develop algorithms for solving least squares problems with MGS. These will give results comparable in accuracy with those obtained by the Householder QR algorithms. We first derive the MGS algorithm for solving least squares problems. Clearly, the problem $\min_x \|Ax - b\|_2$ and the extended problem

$$\min_x \left\| \begin{pmatrix} 0 \\ A \end{pmatrix} x - \begin{pmatrix} 0 \\ b \end{pmatrix} \right\|_2$$

have the same solution. We apply the Householder algorithm (2.3.34)–(2.3.35) to the extended problem, where the Householder reflector P_k, $1{:}n$, is defined as in (2.3.73) by the vector q_k from MGS. To compute

$$\begin{pmatrix} d \\ h \end{pmatrix} = P^T \begin{pmatrix} 0 \\ b \end{pmatrix}, \quad P^T = P_n \cdots P_2 P_1,$$

set $d_1 = 0$, $h_1 = b$, and for $k = 1{:}n$, compute

$$\begin{pmatrix} d_{k+1} \\ h_{k+1} \end{pmatrix} = P_k \begin{pmatrix} d_k \\ h_k \end{pmatrix} = \begin{pmatrix} d_k \\ h_k \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} \begin{pmatrix} -e_k^T & q_k^T \end{pmatrix} \begin{pmatrix} d_k \\ h_k \end{pmatrix}.$$

Note that only the first $k-1$ elements in d_k are nonzero. Further, $h = h^{(n+1)}$ and $d = d^{(n+1)} = (\delta_1, \ldots, \delta_n)^T$, where

$$\delta_k := q_k^T h_k; \quad h_{k+1} := h_k - q_k \delta_k, \quad k = 1{:}n.$$

The recursion for d and h is exactly the same for MGS applied to $\min_x \|Ax - b\|_2$. This shows that the MGS algorithm is backward stable for computing x.

A backward stable approximation of the residual vector r is obtained from the Householder algorithm by 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 (2.3.73). More generally, $P \begin{pmatrix} z \\ h \end{pmatrix}$ is computed by setting $\begin{pmatrix} w_n \\ y_n \end{pmatrix} = \begin{pmatrix} z \\ h \end{pmatrix}$, and for $k = n{:}-1{:}1$,

$$\begin{pmatrix} w_{k-1} \\ y_{k-1} \end{pmatrix} = P_k \begin{pmatrix} w_k \\ y_k \end{pmatrix} = \begin{pmatrix} w_k \\ y_k \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} \left(-e_k^T w^{(k)} + q_k^T y_k \right).$$

Hence, in this step only the kth element of w_k is changed from $\zeta_k = e_k^T z$ to $\omega_k = q_k^T y_k$. The recurrence can be written as

$$y_{k-1} := y_k - q_k(\omega_k - \zeta_k), \quad \omega_k := q_k^T y_k, \quad k = n{:}-1{:}1, \tag{2.3.76}$$

so $y = y_0$, $w = (\omega_1, \ldots, \omega_n)^T$. 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 $w = (\omega_1, \ldots, \omega_n)^T$ is ideally zero, but can be significant when $\kappa(A)$ is large. The computation of y can be seen as reorthogonalization of h_{n+1} against the vectors q_k. It is interesting to note that this is to be done in backward order.

A backward stable MGS algorithm for solving the minimum-norm problem

$$\min \|y\|_2 \quad \text{subject to} \quad A^T y = c$$

can be developed using the same technique as above. Using the interpretation as a Householder QR factorization the solution is obtained from

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

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 (2.3.76) to compute $y = y_0$.

Assuming that MGS has been applied to $A \in \mathbb{R}^{m\times n}$, rank $(A) = n$, to compute Q and R, Algorithm 2.3.12 computes the solution minimum-norm solution y to the linear system $A^T y = c$.

Algorithm 2.3.12 (*Minimum-Norm Solution by MGS*)

```
function [y,rho] = mgsmn(Q,R,c)
% MGSMN uses the MGS thin QR factorization
%   of A to solve the minimum-norm problem and
%   returns the solution y, and its norm rho.
% ------------------------------------------
[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);
```

No derivation of this algorithm without using the interpretation as a Householder QR factorization seems possible. A backward stable MGS algorithm can also be developed for solving the augmented system (2.1.15), based on the Householder QR algorithm given in Theorem 2.3.5.

2.3.7 Partitioned and Recursive QR Factorization

To obtain near-peak performance for large dense matrix computations on current computing architectures requires code dominated by matrix-matrix operations, since these involve less data movement per floating point operation. To achieve this, the QR factorization can be organized in partitioned or blocked form, where the operations are reordered and grouped into matrix operations.

Assume that the matrix $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$ is partitioned as

$$A = (A_1,\ A_2), \quad A_1 \in \mathbb{R}^{m\times n_1}, \tag{2.3.77}$$

where $n_1 \ll n$ is a suitable block size. In the first step, we compute the QR factorization

$$Q_1^T A_1 = \begin{pmatrix} R_1 \\ 0 \end{pmatrix}, \quad Q_1 = P_1 P_2 \cdots P_{n_1}, \tag{2.3.78}$$

using Algorithm 2.3.3. Here $P_i = I - u_i u_i^T$, $i = 1{:}n_1$, are Householder reflectors. Next, the remaining columns A_2 are updated:

$$Q_1^T A_2 = P_{n_1} \cdots P_2 P_1 \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} = \begin{pmatrix} R_{12} \\ B \end{pmatrix}. \tag{2.3.79}$$

where $R_{12} \in \mathbb{R}^{n_1 \times (m-n_1)}$ is part of the final factor R. It now remains to compute the QR factorization of B. In the next step the columns of B are partitioned so that

$$B = (B_1,\ B_2), \quad B_1 \in \mathbb{R}^{(m-n_1)\times n_2}.$$

Then, as in the first step, the QR factorization of B_1 is computed and B_2 is updated. This process is continued until the columns in A are exhausted.

In partitioned QR factorization the major part of the computation is spent in the updating steps (2.3.79). As described, these steps are slowed down because they do not use BLAS 3. To achieve high performance, it is essential to speed up this part. This can be done by aggregating the Householder reflectors so that the updating can be expressed as matrix-matrix operations. Since each Householder reflector performs a rank-one modification, it should be possible to express the product of n_1 Householder reflectors as a rank-n_1 modification. The following lemma shows how to generate the latter representation, which is the one used in LAPACK.

Lemma 2.3.2 *Let $P_1, P_2, \ldots, P_r$ be a sequence of Householder reflectors. 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 $T_1, T_2 \in \mathbb{R}^{r\times r}$ are upper triangular matrices. Then, the product $Q_1 Q_2$ can be written as

$$Q = Q_1 Q_2 = (I - Y_1 T_1 Y_1^T)(I - Y_2 T_2 Y_2^T) = (I - Y T Y^T), \tag{2.3.80}$$

where

$$Y = (Y_1,\ Y_2), \qquad T = \begin{pmatrix} T_1 & -T_1 (Y_1^T Y_2) T_2) \\ 0 & T_2 \end{pmatrix}. \tag{2.3.81}$$

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

For the special case that $r_2 = 1$ and

$$I - Y_k T_k Y_k^T = (I - Y_{k-1} T_{k-1} Y_{k-1}^T)(I - \tau_k u_k u_k^T),$$

Lemma 2.3.2 gives the recursion

$$Y_k = (Y_{k-1},\ u_k), \qquad T_k = \begin{pmatrix} T_{k-1} & -\tau_k T_{k-1}(Y_{k-1}^T u_k), \\ 0 & \tau_k \end{pmatrix}. \quad k = 2{:}n_1. \quad (2.3.82)$$

This is used to aggregate the Householder reflectors for each processed block. The updating of A_2 in (2.3.79) can then be written

$$(I - Y_{n_1} T_{n_1}^T Y_{n_1}^T) A_2 = A_2 - Y_{n_1}(T_{n_1}^T (Y_{n_1}^T A_2)),$$

which involves only matrix-matrix operations. (Note the order of the operations on the right-hand side is important.). The partitioned algorithm requires more storage and operations than the point algorithm, namely those needed to compute and store the T matrices. Using a fixed number p of columns in the partitioned algorithm requires n/p T-matrices of size p to be formed and stored, giving a storage overhead of $\frac{1}{2}12np$. For large matrices this is more than offset by the increased rate of execution.

As mentioned in Sect. 1.6.4, recursive algorithms can execute efficiently on high performance computers and are a viable alternative to partitioned algorithms. The reason is that recursion leads to automatic variable blocking that dynamically adjusts to an arbitrary number of levels of memory hierarchy. To develop a recursive QR algorithm, let

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

be a partitioned QR factorization, where A_1 consists of the first $\lfloor n/2 \rfloor$ columns of A. The QR factorization of A_1 is computed and the remaining part A_2 of the matrix is updated:

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

Next, the QR factorization of B is recursively computed, giving Q_2, R_{22}, and $Q = Q_1 Q_2$. Algorithm 2.3.13 performs a recursive QR factorization of $A \in \mathbb{C}^{m \times n}$ ($m \geq n$). The matrix $Q = I - YTY'$ is given in aggregated form, where $Y \in \mathbb{C}^{m \times n}$ and unit lower trapezoidal and $T \in \mathbb{C}^{n \times n}$ is upper triangular.

A disadvantage of this algorithm is the overhead in storage and operations caused by the T matrices. At the end of the recursive QR factorization a T-matrix of size $n \times n$ is formed and stored. This is a much larger storage overhead than for the partitioned QR algorithm. A better solution would be to use a hybrid of the partitioned and the recursive algorithm, where the recursive QR algorithm is used to factorize the blocks in the partitioned algorithm; see Elmroth and Gustavson [94, 2004].

Algorithm 2.3.13 (*Recursive QR Factorization*)

```
function [Y,T,R] = recqr(A)
% RECQR computes recursively the QR factorization
%   of the m by n matrix A (m >= n). Output is the
%   n by n R and Q = (I - YTY') in aggregated form.
% ------------------------------------------------
[m,n] = size(A);
if n == 1
   [Y,T,R] = houseg(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
```

Two different schemes have been proposed for aggregating products of Householder transformations: the WY representation of Bischof and Van Loan [20, 1987] and a more storage-efficient version by Schreiber and Van Loan [253, 1989]. Algorithms for QR factorization on parallel processing machines have been studied by many authors. O'Leary and Whitman [218, 1990] consider algorithms for Householder and row-wise MGS on distributed MIMD machines using row-wise partitioning schemes. Oliveira et al. [219, 2000] analyze pipelined implementations using different partitioning schemes including block and block-cyclic column-wise schemes. A parallel implementation of CGS with reorthogonalization is given by Hernandez et al. [160, 2006]. Communication-avoiding parallel and sequential algorithms for QR factorization are developed by Demmel et al. [69, 2008].

2.3.8 Condition Estimation and Iterative Refinement

A condition estimator based on inverse iteration and similar to that described in Sect. 1.4.4 can be developed for the least squares problem. Let R be the upper triangular factor in the QR factorization of A or alternatively the Cholesky factor of A^TA. Let u be a given vector, and compute v and w from

$$R^T v = u, \qquad Rw = v. \tag{2.3.83}$$

This requires about $2n^2$ flops and since $w = R^{-1}(R^{-T}u) = (A^TA)^{-1}u$, it is equivalent to one step of inverse iteration with A^TA (see Sect. 3.3.3). Provided that u is suitably chosen,

$$\sigma_{\min}^{-1} \approx \|w\|_2/\|v\|_2$$

will usually be a good estimate. If A is ill-conditioned, then w is usually a good approximation of the right singular vector corresponding to σ_n. If u is chosen as a random vector, two or three steps of inverse iteration usually suffice.

Example 2.3.3 Inverse iteration will often detect near rank-deficiency even when it is not revealed by a small diagonal element in R. The $n \times n$ upper triangular matrix

$$W = \begin{pmatrix} 1 & -1 & \cdots & -1 & -1 \\ & 1 & \cdots & -1 & -1 \\ & & \ddots & \vdots & \vdots \\ & & & 1 & -1 \\ & & & & 1 \end{pmatrix}. \tag{2.3.84}$$

has numerical rank $n-1$ when n is large. If $n = 50$ and W is perturbed by changing the $w_{50,1}$ entry to -2^{-48}, then the new matrix $\widehat{W}$ will be exactly singular. If σ_{48} is the smallest singular value of W, then

$$\sigma_{50} \le \|W - \widehat{W}\|_F = \frac{1}{2^{48}} \approx 7.105 \cdot 10^{-15}.$$

The next smallest singular value is $\sigma_{49} \approx 1.5$, so there is a well defined gap between σ_{49} and σ_{50}. But the computed QR factorization $Q = I$ and $R = W$ (which is exact) gives no indication of the numerical rank-deficiency. (If column interchanges are employed, the diagonals elements in R indicate rank 49.) Doing a single inverse iteration on W^TW using the MATLAB commands

```
n = 50;  W = eye(n) - triu(ones(n,n),1);
z = ones(n,1);  x = W\(W'\z);
s = 1/sqrt(max(abs(x)));
```

gives an approximate smallest singular value $s = 1.9323 \cdot 10^{-15}$. A second inverse iteration gives a value of $2.3666 \cdot 10^{-30}$. □

Reliable estimates can be based on the componentwise error bounds (2.2.27)–(2.2.28). In particular, if $E = |A|$, $f = |b|$, we obtain taking norms the estimates

$$\|\delta x\| \lessapprox \omega \left(\| \, |A^\dagger|(|b| + |A||x|) \| + \| \, |(A^TA)^{-1}||A|^T|r| \, \| \right), \tag{2.3.85}$$

$$\|\delta r\| \lessapprox \omega \left(\| \, |I - AA^\dagger|(|A||x| + |b|) \| + \| \, |(A^\dagger)^T||A|^T|r| \, \| \right). \tag{2.3.86}$$

For maximum norm the estimate for $\|\delta x\|$ can be written as

$$\|\delta x\|_\infty \lesssim \omega(\| \, |B_1|g_1\|_\infty + \| \, |B_2|g_2\|_\infty), \tag{2.3.87}$$

where

$$B_1 = A^\dagger, \quad g_1 = |b| + |A||x|, \quad B_2 = (A^T A)^{-1}, \quad g_2 = |A^T||r|. \tag{2.3.88}$$

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 $D = \mathrm{diag}(d)$ and $e = (1, 1, \ldots, 1)$, we have

$$\| \, |B^{-1}|d\|_\infty = \| \, |B^{-1}|De\|_\infty = \| \, |B^{-1}D|e\|_\infty = \| \, |B^{-1}D|\|_\infty = \|B^{-1}D\|_\infty.$$

Using Hager's 1-norm estimator (see Algorithm 1.4.2, p. 104), a reliable order-of-magnitude estimate can be obtained of $\|C^T\|_1 = \|C\|_\infty$, where $C = B^{-1}D$, at a cost of a few matrix-vector products Cx and $C^T y$ for some carefully selected vectors x and y. If A has full rank and a QR factorization of A is known, then

$$A^\dagger = R^{-1}Q^T, \qquad (A^\dagger)^T = QR^{-T}$$

are known and the required matrix-vector products can be computed inexpensively. For details we refer to Higham [162, 2002], Chap. 15.

In Sect. 1.4.6 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 $A\delta = \bar{r}$ is solved 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. A similar process can be devised to compute highly accurate solutions to the linear least squares problems $\min_x \|Ax - b\|_2$. 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. But it turns out that this naive approach is satisfactory only if the residual vector r is sufficiently small. In general, iterative refinement should be applied to the augmented system (2.1.15) and both the solution x and the residual r refined simultaneously. The process of iterative refinement in floating point arithmetic with base β is described in Algorithm 2.3.14.

Algorithm 2.3.14 (*Iterative Refinement with QR Factorization*)

$s := 0;\ \ x^{(0)} := 0;\ r^{(0)} := b;$
repeat
$f^{(s)} := b - r^{(s)} - Ax^{(s)};$
$g^{(s)} := c - A^T r^{(s)};$ (in precision $\mathbf{u}_2 = \beta^{-t_2}$)
solve augmented system (in precision $\mathbf{u}_1 = \beta^{-t_1}$)
$x^{(s+1)} := x^{(s)} + \delta x^{(s)};$
$r^{(s+1)} := r^{(s)} + \delta r^{(s)};$
$s := s + 1;$
end

To solve for the corrections in the algorithm, Theorem 2.3.5 is used with the computed factors $\bar{Q}$ and $\bar{R}$:

$$z^{(s)} = \bar{R}^{-T} g^{(s)}, \qquad \begin{pmatrix} d^{(s)} \\ e^{(s)} \end{pmatrix} = \bar{Q}^T f^{(s)}, \tag{2.3.89}$$

$$\delta r^{(s)} = \bar{Q} \begin{pmatrix} z^{(s)} \\ e^{(s)} \end{pmatrix}, \qquad \delta x^{(s)} = \bar{R}^{-1}(d^{(s)} - z^{(s)}). \tag{2.3.90}$$

Computing the residuals and corrections takes $4mn$ flops in high precision. Computing the solution from (2.3.89)–(2.3.90) 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.

It can be shown (see Björck [23, 1967]) that initially the convergence rate of iterative refinement is linear with rate

$$\rho = c_1 \mathbf{u} \min_{D>0} \kappa_2(AD),$$

where c_1 is of modest size. This rate is independent of the right-hand side and hence true also for large residual problems. Amazingly, the process converges if $\rho < 1$, even though the first approximation may have no significant digits; see Björck and Golub [29, 1967]. A portable and parallelizable implementation of the Björck–Golub refinement algorithm using the extended precision BLAS is now being made available in LAPACK; see Demmel et al. [70, 2009].

In contrast, when iterative refinement is applied to the normal equations (see Algorithm 2.1.1), the rate of convergence is proportional to $c_2 \mathbf{u} \min_{D>0} \kappa_2^2(AD)$. This makes a huge difference for ill-conditioned problems.

Exercises

2.3.1 The matrix H is an orthogonal reflector if H is Hermitian and $H^2 = I$. Show that $P = (I - H)/2$ is an orthogonal projector. Conversely, if P is an orthogonal projector show that $H = I - 2P$ is a reflector.

2.3.2 Show that 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

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

2.3.3 Modify Algorithm 2.3.1 so that it works also for constructing a complex Householder transformation P such that for a given complex vector x, $Px = \gamma\|x\|_2 e_1$ with $|\gamma| = 1$.

2.3.4 Show that the plane rotation (2.3.12) can be applied using three additions and three multiplications, by setting $p = s/(1 + c)$ and computing

$$\begin{aligned} \beta_i &= c\,\alpha_i + s\,\alpha_j, \\ \beta_j &= p\,(\alpha_i + \beta_i) - \alpha_j. \end{aligned}$$

These formulas can be used when multiplication is more expensive than addition.

2.3.5 Specialize the formulas in (2.3.5) for a Householder reflector P to the case $n = 2$. What is the relation between this and the corresponding plane rotation? How many flops are needed to apply the reflector P to a matrix of dimension 2 by n?

2.3.6 Show that if $S \in \mathbb{R}^{n\times n}$ is skew-symmetric ($S^T = -S$), then $I - S$ is nonsingular and the matrix

$$Q = (I - S)^{-1}(I + S) \tag{2.3.91}$$

is orthogonal. This is known as the **Cayley transform**, (b) Verify the special 2×2 case

$$S = \begin{pmatrix} 0 & \tan\frac{\theta}{2} \\ -\tan\frac{\theta}{2} & 0 \end{pmatrix}, \qquad Q = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

where $0 \leq \theta < \pi$.

2.3.7 Let $a_j = Ae_j$, $j = 1{:}n$, be the jth column of a matrix $A \in \mathbb{R}^{n\times n}$. Use the QR factorization to show Hadamard's determinant inequality

$$|\det(A)| \leq \prod_{j=1}^{n} \|a_j\|_2, \tag{2.3.92}$$

where equality holds only if A^TA is a diagonal matrix or A has a zero column.
Hint: Use that $\det(A) = \det(Q)\det(R)$, where $\det(Q) = \pm 1$.

2.3.8 Modify the MATLAB code in Algorithm 2.3.3 for Householder QR factorization so it computes and returns the matrix $Q_1 \in \mathbb{R}^{m\times n}$.

2.3.9 (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) Modify Algorithm 2.3.5 for computing the least squares solution and residual when the square root free version of modified Gram–Schmidt is used.

Comment: There is no square root free version of the Householder QR factorization.

2.3.10 Let $Q = Q_1 = (q_1, q_2, \ldots, q_n) \in \mathbb{R}^{n\times n}$ be a real orthogonal matrix.

(a) Determine a reflector $P_1 = I - 2v_1v_1^T$ such that $P_1q_1 = e_1 = (1, 0, \ldots, 0)^T$, and show that $P_1Q_1 = Q_2$ has the form

$$Q_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & \widetilde{Q}_2 & \\ 0 & & & \end{pmatrix},$$

where $\widetilde{Q}_2 = (\widetilde{q}_1, \widetilde{q}_2, \ldots, \widetilde{q}_n) \in \mathbb{R}^{(n-1)\times(n-1)}$ 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

$$P_{n-1} \cdots P_2 P_1 Q = \operatorname{diag}(1, \ldots, 1, \pm 1).$$

2.3.11 Let $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \in \mathbb{R}^{m\times m}$ be the orthogonal factor in the Householder QR factorization of a real matrix $A \in \mathbb{R}^{m\times n}$. Verify the operation counts for the explicit computations of the submatrices Q_1 and Q_2 given in Sect. 2.3.2: $2(mn^2 - n^3/3)$ flops for Q_1 and $2n(m-n)(2m-n)$ flops for Q_2.

2.3.12 (a) Show that a 3×3 upper triangular matrix can be brought to bidiagonal form using two plane rotations. The element r_{13} is zeroed by a rotation from the left in the (1,2)-plane, which introduces a new nonzero element in position (2,1). This is then zeroed by a rotation from the right in the (1,2) plane.

(b) Use the idea in (a) to develop an algorithm using plane rotations to transform an upper triangular matrix $R \in \mathbb{R}^{n\times n}$ to bidiagonal form. How many flops does this require?

2.3.13 Trefethen and Bau [281, 1997], pp. 237–238, suggest a blend of the methods of Golub–Kahan and Chan for bidiagonal reduction, which is more efficient 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 $(m-k)/(n-k) = 2$.

2.3.14 Consider the over-determined linear system $Ax = b$ in Example 2.1.5. Assume that $\epsilon^2 \le \mathbf{u}$ 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

$$\text{CGS}: \ |q_3^T q_2| = 1/2, \qquad \text{MGS}: \ |q_3^T q_1| = \frac{\epsilon}{\sqrt{6}} \le \frac{\kappa \mathbf{u}}{3\sqrt{3}}.$$

Note that for CGS orthogonality has been completely lost!

2.3.15 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 $Q_2^T A_2 A_1 = R_2 R_1$.

2.3.16 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

$$\begin{pmatrix} A & b \\ I & 0 \end{pmatrix} \longrightarrow \begin{pmatrix} Q_1 & r \\ R^{-1} & -x \end{pmatrix}.$$

Hence, the MGS algorithm can be made to provide in a single sweep operation the solution x, the residual r, and the matrix R^{-1}, which is required for computing the covariance matrix.

2.3.17 Suppose n_1 steps of MGS are performed on the least squares problem $\min_x \|Ax - b\|_2$, yielding the partial factorization

$$\begin{pmatrix} A & b \end{pmatrix} = \begin{pmatrix} Q_1 & \widetilde{A}_2 & \widetilde{b} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} & z_1 \\ 0 & I & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

where $x_1 \in \mathbb{R}^{n_1}$ and $R_{11} \in \mathbb{R}^{n_1 \times n_1}$ is nonsingular. Show that x_2 is the solution to the reduced least squares problem $\min_{x_2} \|\widetilde{b} - \widetilde{A}x_2\|_2$, and that with x_2 known x_1 can be computed by back substitution from $R_{11}x_1 = z_1 - R_{12}x_2$.
Hint: Show that $r_1 \perp r_2$, where $r_1 = Q_1(z_1 - R_{12}x_2 - R_{11}x_1)$ and $r_2 = \widetilde{b}_1 - \widetilde{A}_2x_2$.

2.3.18 (a) Test the recursive QR algorithm `recqr(A)` on some test matrices of your choice. Do you obtain the same result as from the built-in function `qr(A)`?
(b) Write a recursive algorithm for computing the QR factorization by MGS.
Hint: Model it after Algorithm 2.3.13.

2.4 Rank-Deficient Problems

Rank-deficiency in least squares problems can arise in several different ways. In statistics one often has one set of variables called the factors that are used to control, explain, or predict another variable. The set of factors correspond to the columns of a matrix $A = (a_1, a_2, \ldots, a_n)$. If these are highly collinear, then the numerical rank of A is less than n and the least squares problem $\min_x \|Ax - b\|_2$ does not have a unique solution. Often the rank is not known in advance and one wants the computed factorization to reveal the rank. Another typical case occurs in discrete approximations to ill-posed problems, where the numerical rank is not well determined and is usually much less than n. Problems of this type require special treatment.

2.4.1 Numerical Rank

The *mathematical* notion of rank is no longer appropriate when the ideal matrix A is subject to inaccuracy of data and rounding errors made during computation. Suppose $A \in \mathbb{R}^{m \times n}$ is a matrix of rank $r < n$, whose elements are perturbed by a matrix E of small random errors, e.g.,

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \qquad E = \begin{pmatrix} \epsilon_{11} & \epsilon_{12} \\ \epsilon_{21} & \epsilon_{22} \end{pmatrix},$$

where $|\epsilon_{ij}| \ll 1$. Then it is most likely that the perturbed matrix $A + E$ has full rank. But since $A + E$ is close to a rank-deficient matrix, it should be considered as **numerically rank-deficient**. Failure to detect this when solving linear systems and linear least squares problems can lead to a meaningless solution of very large norm, or even to breakdown of the numerical algorithm.

The **numerical rank** assigned to a matrix should depend on some tolerance δ, which reflects the error level in the data and/or the precision of the floating point arithmetic used. A useful definition can be given in terms of the singular values of A.

Definition 2.4.1 A matrix $A \in \mathbb{R}^{m \times n}$ has numerical δ-rank equal to k ($k \leq \min\{m, n\}$) if

$$\sigma_1 \geq \cdots \geq \sigma_k > \delta \geq \sigma_{k+1} \geq \cdots \geq \sigma_n,$$

where σ_i, $i = 1{:}n$, are the singular values of A. If we write

$$A = U\Sigma V^T = U_1\Sigma_1 V_1^T + U_2\Sigma_2 V_2^T,$$

where $\Sigma_2 = \mathrm{diag}(\sigma_{k+1}, \ldots, \sigma_n)$, then $\mathcal{R}(V_2) = \mathrm{span}\{v_{k+1}, \ldots, v_n\}$ is called the **numerical null space** of A.

It follows from Theorem 2.2.8 that if the numerical δ-rank of A equals k, then rank $(A + E) \geq k$ for all perturbations such that $\|E\|_2 \leq \delta$, i.e., such perturbations cannot *lower* the rank. Definition 2.4.1 is only useful when there is a well defined gap between σ_{k+1} and σ_k. This should be the case if the exact matrix A is rank-deficient but well-conditioned. But it may happen that there does not exist a gap for any k, e.g., if $\sigma_k = 1/k$. In such a case the numerical rank of A is not well defined.

The choice of the parameter δ in Definition 2.4.1 of numerical rank is not always an easy matter. If the errors in a_{ij} satisfy $|e_{ij}| \leq \epsilon$, for all i, j, an appropriate choice is $\delta = (mn)^{1/2}\epsilon$. On the other hand, if the absolute size of the errors e_{ij} differs widely, then Definition 2.4.1 is not appropriate. One could then scale the rows and columns of A so that the magnitude of the errors become nearly equal. (Note that any such diagonal scaling $D_r A D_c$ will induce the same scaling $D_r E D_c$ of the error matrix.)

2.4.2 Pivoted QR Factorizations

A QR factorization of a rank-deficient matrix may not serve any useful purpose unless column pivoting is employed. In **pivoted QR factorization**, column interchanges yield a QR factorization of $A\Pi$ for some permutation matrix Π.

Assume that in the Gram–Schmidt QR factorization of a rank-deficient matrix A the first $k-1$ columns $a_1, \ldots, a_{k-1}$ are linearly independent. Then the orthogonal vectors $q_1, \ldots, q_{k-1}$ can be computed without breakdown. If the column a_k is a linear combination of $q_1, \ldots, q_{k-1}$, then $a_k^{(k)} = 0$, and without pivoting the process stops. However, if rank $(A) \geq k$, then there must be a vector a_p, for some $p > k$, that is linearly independent on $q_1, \ldots, q_{k-1}$. After columns k and p are interchanged, the process can proceed.

We now describe the row-wise MGS algorithm with column interchanges. One reason MGS was used in practice long before its superior numerical stability was appreciated is that it is more suitable for pivoting and solving rank-deficient problems. The standard pivoting strategy is to choose at step k the column that maximizes the diagonal element $|r_{kk}|$. Let p be the smallest index such that

$$\|a_p^{(k)}\|_2 = \max_{k \leq j \leq n} \|a_j^{(k)}\|_2.$$

If this maximum is zero, then all remaining columns $a_k, \ldots, a_n$ are linearly dependent on $q_1, \ldots, q_{k-1}$ and the factorization is complete. Otherwise, interchange columns p

and k and proceed. If rank $(A) = r$, then in exact arithmetic, pivoted MGS computes a factorization $A\Pi = QR$, where

$$Q = (q_1, \ldots, q_r) \in \mathbb{R}^{m\times r}, \quad R = \begin{pmatrix} R_{11} & R_{12} \end{pmatrix} \in \mathbb{R}^{r\times n}, \tag{2.4.1}$$

and Π is a permutation matrix and R_{11} upper triangular and nonsingular. In exact arithmetic the computed R-factor corresponds to that obtained from pivoted Cholesky factorization.

If the column norms $\|\tilde{a}^{(k)}\|_2$, $j = k : n$ are recomputed at each stage, then column interchanges will increase the operation count of the QR factorization by 50 %. It suffices to compute the initial column norms and then update these as the factorization proceeds using the recursion

$$\|a_j^{(k+1)}\|_2 = \left(\|a_j^{(k)}\|_2^2 - r_{kj}^2\right)^{1/2} = \|a_j^{(k)}\|_2 \left[1 - \left(r_{kj} \Big/ \|a_j^{(k)}\|_2\right)^2\right]^{1/2}, \tag{2.4.2}$$

$j = k + 1{:}n$. To avoid overflow the last expression should be used. This reduces the overhead of pivoting to $O(mn)$ operations. Cancellation in the subtraction can cause this to fail and therefore the new column norms are recomputed from scratch if

$$\|a_j^{(k+1)}\|_2 = \mathbf{u}^{1/2}\|a_j^{(1)}\|_2.$$

As shown by Golub [126, 1965], the same pivoting strategy can be used in Householder QR factorization. Assume that the first $k-1$ steps have yielded the partial QR factorization

$$A^{(k)} = P_{k-1}\cdots P_1\, A\, \Pi_1 \cdots \Pi_{k-1} = \begin{pmatrix} R_{11}^{(k)} & R_{12}^{(k)} \\ 0 & \widetilde{A}^{(k)} \end{pmatrix}. \tag{2.4.3}$$

Then the pivot column in the next step is chosen as a column of largest norm in the submatrix $\widetilde{A}^{(k)} = (\tilde{a}_k^{(k)}, \ldots, \tilde{a}_n^{(k)})$. Let p be the smallest index such that

$$s_p^{(k)} = \max_{k\le j\le n} s_j^{(k)}, \quad s_j^{(k)} = \|\tilde{a}_j^{(k)}\|_2, \quad j = k : n. \tag{2.4.4}$$

If $s_p^{(k)} = 0$, the algorithm terminates. Otherwise, columns p and k are interchanged. It is easy to show that this pivoting rule is equivalent to maximizing the diagonal element r_{kk}. Since the column lengths are invariant under orthogonal transformations, the quantities $s_j^{(k)}$ can be updated using

$$s_j^{(k+1)} = s_j^{(k)}\left[1 - \left(r_{jk}/s_j^{(k)})\right)^2\right]^{1/2}, \quad j = k+1{:}n. \tag{2.4.5}$$

Without pivoting, the QR factorization is numerically invariant under column scaling, provided the scaling does can be done exactly. This is no longer true for pivoted QR factorization with the standard pivoting strategy described here because scaling will influence the choice of pivots.

In the upper triangular factor R obtained from a pivoted QR factorization, certain relations must hold between its entries.

Theorem 2.4.1 *Suppose that R is computed by pivoted* QR *factorization. Then the elements in R satisfy the inequalities*

$$r_{kk}^2 \geq \sum_{i=k}^{j} r_{ij}^2, \quad j = k+1{:}n. \tag{2.4.6}$$

In particular, the diagonal elements form a non-increasing sequence

$$|r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}|. \tag{2.4.7}$$

If rank $(A) = r < n$, then in exact arithmetic Householder pivoted QR factorization yields a factorization of $A\Pi$ of the form

$$A\Pi = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix}, \tag{2.4.8}$$

where $R_{11} \in \mathbb{R}^{r\times r}$ is upper triangular with positive diagonal elements. 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 (2.4.8) is not unique, since there are many ways to select r linearly independent columns of A.

If floating point arithmetic is used, then pivoted QR factorization yields a factorization

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

with $R_{22} \neq 0$, but $\|R_{22}\| \leq \epsilon \|A\|$ for some user specified tolerance ϵ. If ϵ is chosen sufficiently small, deleting the block R_{22} corresponds to a small backward error. Since only orthogonal transformations have been used, there is a perturbation E with $\|E\|_2 \leq \epsilon \|A\|_2$ such that $A + E$ has rank r.

Example 2.4.1 Let $A \in \mathbb{R}^{n\times n}$, $n = 100$, be an ill-conditioned matrix obtained from the discretization of an ill-posed integral equation. Figure 2.3 shows the singular values $\sigma_k(A)$ together with the diagonal elements r_{kk} of R in the pivoted QR factorization. The singular values are sufficiently well approximated to reveal the rank. □

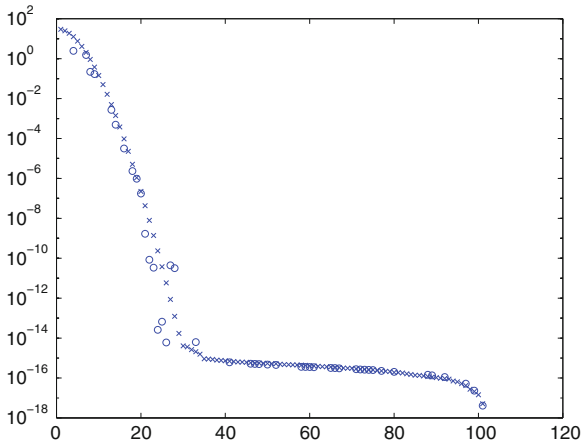


Fig. 2.3 Diagonal elements of R (○) in pivoted QR factorization compared with singular values (×) of the matrix A

For any QR factorization it holds that

$$\sigma_1 = \max_{\|x\|_2=1} \|Rx\|_2 \geq \|Re_1\|_2 = |r_{11}|.$$

Hence, $|r_{11}|$ is a *lower bound for the largest singular value* σ_1 of A. Since R and R^T have the same singular values, we also have

$$\sigma_n = \min_{\|x\|_2=1} \|R^T x\|_2 \leq \|R^T e_n\|_2 = |r_{nn}|,$$

which gives an upper bound for σ_n. For a triangular matrix satisfying (2.4.6) we also have the upper bound

$$\sigma_1(R) = \|R\|_2 \leq \|R\|_F = \Big(\sum_{i\leq j} r_{ij}^2\Big)^{1/2} \leq \sqrt{n}|r_{11}|.$$

From the interlacing property of singular values (Theorem 2.2.9), a similar argument gives the upper bounds

$$\sigma_k(R) \leq \sqrt{n-k+1}\,|r_{kk}|, \quad 1 \leq k \leq n. \tag{2.4.9}$$

Hence, after k steps in the pivoted QR factorization, if $|r_{kk}| \leq (n-k+1)^{-1/2}\delta$, then $\sigma_k(A) = \sigma_k(R) \leq \delta$. Then A has numerical rank less than k, and the algorithm can be terminated. The converse is not true, i.e., the rank may not always be revealed by a small element $|r_{kk}|$, $k \leq n$. The best known lower bounds for the singular value of R whose elements satisfy (2.4.6) are

$$2^{1-k}|r_{kk}| \le 3|r_{kk}| \Big/ \sqrt{4^k + 6k - 1} \le \sigma_k, \quad 1 \le k \le n. \tag{2.4.10}$$

(These bounds were stated without proof in Faddeev et al. [97, 1968]. A proof is given in Lawson and Hanson [190, 1974], Chap. 6.) The lower bound in (2.4.10) for $k = n$ can almost be attained, as shown in the example below. Then the pivoted QR factorization may not reveal the rank of A.

Example 2.4.2 Pivoted QR factorization will detect the rank-deficiency of the matrix in Example 2.3.3. The computed value (using MATLAB) of $r_{100,100}$ is $5.5511e{-}017$. Since this value is smaller than the tolerance $\tau = 100u\|W\| \approx 1.4e{-}012$, we conclude correctly that W has numerical rank 99. However, the computed value of $r_{100,100}$ is far greater than the exact value of $\sigma_{100} = 3.1554e{-}030$.

Even with column interchanges, QR factorization may not reveal rank deficiency. The upper triangular Kahan matrix (see Kahan [174, 1966])

$$A_n = \operatorname{diag}(1, s, \ldots, s^{n-1}) \begin{pmatrix} 1 & -c & \cdots & -c & -c \\ & 1 & \cdots & -c & -c \\ & & \ddots & \vdots & \vdots \\ & & & 1 & -c \\ & & & & 1 \end{pmatrix}, \quad s = \sqrt{1 - c^3}, \tag{2.4.11}$$

has been chosen so that no column interchanges will occur. Therefore, $R_n = A_n$ for pivoted QR factorization. For $n = 100$ and $c = 0.2$, the last diagonal element of R is $r_{nn} = s^{n-1} = 0.820$. This is a large overestimate of the smallest singular value $\sigma_n = 0.368 \cdot 10^{-8}$. QR factorization will reveal the correct rank if the columns are reordered as $(n, 1, 2, \ldots, n-1)$. □

To simplify notation, we assume in the following that $\Pi = I$. (This is no restriction, because the column permutation of A can be assumed to have been applied in advance.) Using (2.4.8), we reduce the least squares problem $\min_x \|Ax - b\|_2$ to

$$\min_x \left\| \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \right\|_2, \tag{2.4.12}$$

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 satisfy

$$R_{11}x_1 = c_1 - R_{12}x_2, \tag{2.4.13}$$

where x_2 can be chosen arbitrarily. By setting $x_2 = 0$ and solving $R_{11}x_1 = c_1$, we obtain a particular solution $x_1 = R_{11}^{-1}c_1$ for which at most $r = \operatorname{rank}(A)$ components are nonzero. Any least squares solution x such that Ax involves at most r columns of A is called a **basic solution**. Such a solution is appropriate in applications where it is required to fit the vector b using *as few columns of* A as possible.

For an arbitrary vector x_2, we have

$$x_1 = d - Cx_2, \quad R_{11}d = c_1, \quad R_{11}C = R_{12}. \tag{2.4.14}$$

The solution of minimum norm, i.e., the pseudoinverse solution, is obtained by solving

$$\min_x \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_2 = \min_{x_2} \left\| \begin{pmatrix} d \\ 0 \end{pmatrix} - Wx_2 \right\|_2, \quad W = \begin{pmatrix} C \\ -I_{n-r} \end{pmatrix}. \tag{2.4.15}$$

This least squares problem for x_2 always has full rank, and hence a unique solution. To compute x_2 we could form and solve the normal equations

$$(C^T C + I)x_2 = C^T d.$$

It is preferable to compute the Householder QR factorization

$$Q_C^T W = \begin{pmatrix} R_C \\ 0 \end{pmatrix}, \qquad Q_C^T \begin{pmatrix} d \\ 0 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix},$$

taking into account the special structure of the matrix W. Since the pseudoinverse solution $x = A^\dagger b$ is *the residual* of the problem (2.4.15), we get

$$x = A^\dagger b = Q_C \begin{pmatrix} 0 \\ d_2 \end{pmatrix}.$$

For any $z \in \mathbb{R}^{n-r}$ 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 $\mathcal{N}(A) = \mathcal{R}(W)$.

2.4.3 Rank-Revealing Permutations

From Example 2.4.2 it follows that using pivoted QR factorization and setting rank $(A) = \max_{|r_{kk}|>\tau} k$ for some tolerance τ may not yield reliable results. Assume that A has a well defined numerical rank $k < n$, i.e.,

$$\sigma_1 \geq \cdots \geq \sigma_k \gg \tau > \sigma_{k+1} \geq \cdots \geq \sigma_n.$$

Consider now the partial QR factorization

$$A\Pi_k = Q_k \begin{pmatrix} R_k & S_k \\ O & B_k \end{pmatrix}, \tag{2.4.16}$$

where Π_k is a permutation matrix and $R_k \in \mathbb{R}^{k\times k}$ is an upper triangular matrix. By induction and the interlacing properties of singular values (Theorem 2.2.9), it follows that for any factorization of the form (2.4.16)

$$\sigma_k(R_k) \le \sigma_k(A) \quad \text{and} \quad \sigma_1(B_k) \ge \sigma_{k+1}(A). \tag{2.4.17}$$

The factorization (2.4.16) is called a rank-revealing QR factorization if it satisfies

$$\sigma_{\min}(R_k) \ge \sigma_k(A)/p(k,n) \quad \text{and} \quad \sigma_1(B_k) \le \sigma_{k+1}(A)p(k,n), \tag{2.4.18}$$

where $p(k,n)$ is a function bounded by a low-order polynomial in k and n. A QR factorization such that (2.4.18) is satisfied will detect a rank-deficiency if σ_k and σ_{k+1} straddle a gap containing the tolerance τ used to judge the numerical rank and is called a **rank-revealing QR** factorization. The term "rank-revealing" was first used by Chan [44, 1987]. It is known (Hong and Pan [166, 1992]) that every matrix $A \in \mathbb{R}^{m\times n}$, $m \ge n$, has an rank-revealing QR factorization with

$$p(k,n) = \sqrt{k(n-k) + \min(k, n-k)}\,. \tag{2.4.19}$$

Although their proof is constructive, their algorithm is not computationally efficient and is primarily of theoretical importance. The naive solution, to try all possible permutations, is not feasible because the cost is prohibitive—it is exponential in the dimension n. Indeed, to find such a permutation is an NP-hard problem. But there are heuristic algorithms that almost always succeeds in practice.

There are two primary approaches for finding a pivoting strategy that gives a rank-revealing QR factorization:

- **Strategy 1.** Find a pivoting strategy to maximize $\sigma_k(R_k)$.
- **Strategy 2.** Find a pivoting strategy to minimize $\sigma_1(B_k)$.

These two strategies are in a certain sense dual; cf. Problem 2.4.1. Strategy 1 could also be stated in terms of minimizing $\|R_k^{-1}\|_2$. Note that in the Strategy 2 approach we are minimizing $\|R_{22}\|_2$ rather than $\|R_{22}\|_{(1,2)}$, as with column interchanges.

The early rank-revealing algorithms (Foster [103, 1986], Chan [44, 1987]) follow Strategy 2. The basic idea behind Chan's algorithm is as follows: Let v_n be the right singular vector belonging to the smallest singular value σ_n. Then the index of the largest component in v_n indicates which column to permute into position n.

Theorem 2.4.2 *Let $v \in \mathbb{R}^n$ be a vector with $\|v\|_2 = 1$ such that $\|Av\|_2 = \epsilon$, and let Π be a permutation such that if $\Pi^T v = w$, then $|w_n| = \|w\|_\infty$. If $A\Pi = QR$ is the* QR *factorization of $A\Pi$, then $|r_{nn}| \le n^{1/2}\epsilon$.*

Proof (Chan [44, 1987], Theorem 2.1) First we note that since $|w_n| = \|w\|_\infty$ and $\|v\|_2 = \|w\|_2 = 1$, we have $|w_n| = \|w\|_\infty \ge n^{-1/2}\|w\|_2$. Next we have

$$Q^T Av = Q^T A\Pi\Pi^T v = Rw = \begin{pmatrix} \vdots \\ r_{nn}w_n \end{pmatrix}.$$

Therefore $\epsilon = \|Av\|_2 = \|Q^T Av\|_2 = \|Rw\|_2 \geq |r_{nn} w_n|$, from which the result follows. □

In particular, if $v = v_n$, the right singular vector corresponding to the smallest singular value $\sigma_n(A)$,

$$|r_{nn}| \leq \sigma_n(A) \leq n^{-1/2} |r_{nn}|.$$

The algorithm starts with $k = n$, and $R_k = R$, where R is obtained from a pivoted QR factorization of A. Estimates δ_n of the smallest singular value and the corresponding right singular vector are obtained using the LINPACK condition estimator. These estimates are then improved by inverse iteration (see Sect. 3.3.3) applied to $R^T R$. If $\delta_k = \delta_n > \tau$, then the numerical rank is $k = n$. Otherwise, the columns of R_k are permuted and P, Q, and Rf are updated. This process is repeated on the leading $(n-1) \times (n-1)$ principal submatrix of R, and so on. It stops when one of the computed δ_k's is greater than the tolerance τ.

The **column subset selection** problem is closely related to rank-revealing QR factorization. In this problem we are given a matrix $A \in \mathbb{R}^{m \times n}$ and want to determine a subset A_1 of $k < n$ columns such that

$$\|A - (A_1 A_1^{\dagger}) A\|_2$$

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 is maximized.

A comprehensive study of algorithms for rank-revealing QR factorizations is found in Chandrasekaran and Ipsen [49, 1994]. They suggest hybrid algorithms that in practice compute a rank-revealing QR factorization using a small number of iterations even if the worst case is exponential in n. A survey of the use of RRQR for solving discrete ill-posed problems is given by Hansen [150, 1998].

An efficient algorithm for solving rank-deficient problems of low rank $r \ll n$ using UTV and QR factorizations is given by Foster [104, 2004] and Foster and Kommu [105, 2006]. This uses a truncated pivoted QR factorization where the rank of the trailing diagonal block is estimated by a condition estimator.

2.4.4 Complete QR Factorizations

In some applications, e.g., signal processing, it is required to determine the rank of A as well as the range $\mathcal{R}(A)$ (signal subspace) and the null space $\mathcal{N}(A)$. Moreover, the data to be analyzed arrives in real time and these quantities have to be updated at each time step. For such applications the SVD has the disadvantage that it cannot be accurately updated in less than $O(n^3)$ operations, when a row and/or column is modified. Rank-revealing QR factorizations are cheaper to update, but less suitable

when the null space $\mathcal{N}(A)$ of A is needed. The matrix W in (2.4.15) may be ill-conditioned and cannot be easily updated.

Applications of this kind motivate the use of the **complete QR factorization**. This factorization, due to Hanson and Lawson [155, 1969], is of the form

$$A = U \begin{pmatrix} T & 0 \\ 0 & 0 \end{pmatrix} V^T = U_1 T V_1^T, \tag{2.4.20}$$

where $T \in \mathbb{R}^{r\times r}$, $r = \text{rank}\,(A)$, is a triangular matrix with positive diagonal elements, and

$$U = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \in \mathbb{R}^{m\times m}, \qquad V = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \in \mathbb{R}^{n\times n},$$

are square orthogonal matrices partitioned so that $U_1 \in \mathbb{R}^{m\times r}$ and $V_1 = \in \mathbb{R}^{n\times r}$. An advantage of this decomposition is that, like the SVD, it gives orthogonal bases for all four fundamental subspaces of A. In particular, U_1 and V_2 give orthogonal bases for $\mathcal{R}(A)$ and $\mathcal{N}(A)$, respectively. From the orthogonal invariance of the spectral norm it follows that the pseudoinverse of A is

$$A^\dagger = V \begin{pmatrix} T^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^T = V_1 T^{-1} U_1^T. \tag{2.4.21}$$

The factorization (2.4.20) can be computed starting from a rank-revealing QR factorization (2.4.8)

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix}.$$

Next, Householder reflectors P_k, $k = r : (-1) : 1$, are constructed such that

$$\begin{pmatrix} R_{11} & R_{12} \end{pmatrix} P_r \cdots P_1 = \begin{pmatrix} T & 0 \end{pmatrix}.$$

Here P_k zeros elements in row k and only affects columns $k, r+1{:}n$. Then (2.4.20) holds with T upper triangular and $U = Q$ and $V = \Pi P_r \cdots P_1$. 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 reflector P_2 to zero the elements in position (2,5) and (2,6). (Recall that when applied from the right the Householder reflector will act on rows.) In step

k a full matrix of size $k \times (n-r+1)$ is transformed by a Householder reflector. The transformations require a total of $2r^2(n-r+1)$ flops.

In practice, we work with a sequence of matrices whose numerical rank may change at updates. Then it is convenient to use the more general **URV decomposition**,

$$A = URV^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}, \tag{2.4.22}$$

where

$$\left(\|R_{12}\|_F^2 + \|R_{22}\|_F^2 \right)^{1/2} \leq c\sigma_{r+1}(A). \tag{2.4.23}$$

Note that both submatrices R_{12} and R_{22} are required to have small elements. From (2.4.22) we have

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

so that the orthogonal matrix V_2 can be taken as an approximation to the numerical null space $\mathcal{N}_r$.

Related to the URV factorization is the **ULV decomposition**, which is of the form

$$A = U \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} V^T, \tag{2.4.24}$$

where the middle matrix is lower triangular. For this factorization

$$\|AV_2\|_2 = \|L_{22}\|_F,$$

and hence the size of $\|L_{21}\|$ does not adversely affect the null space approximation. Therefore, this factorization is more satisfactory for applications where an accurate approximate null space is needed. On the other hand, the URV decomposition usually gives a superior approximation for the numerical range space and is much simpler to update.

Algorithms for computing an URV decomposition may start with a standard pivoted QR factorization. Next 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 $P^T RQ = \widehat{R}$, where $\widehat{R}$ is upper triangular. Then

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

which shows that the entire last column in $\widehat{R}$ is small. Given w, the matrices P and Q can be constructed as a sequence of plane rotations; see Stewart [268, 1992]. Efficient algorithms can be given for updating an URV decomposition when a new row is appended to A.

Like the rank-revealing QR factorizations, the URV decomposition yields approximations to the singular values. Mathias and Stewart [206, 1993] give the following bounds:

$$f\sigma_i \le \sigma_i(R_{11}) \le \sigma_i, \quad i = 1{:}r,$$
$$\sigma_i \le \sigma_{i-k}(R_{22}) \le \sigma_i/f, \quad i = r+1{:}n,$$

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.

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 Chan and Hansen [46, 1994].

2.4.5 The QLP Factorization

Let the pivoted QR factorization of $A \in \mathbb{R}^{m\times n}$ be

$$A\Pi = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R \in \mathbb{R}^{n\times n}. \tag{2.4.25}$$

Take the transpose R^T of the R-factor and compute its QR factorization without column interchanges,

$$R^T = PL^T, \quad L \in \mathbb{R}^{n\times n}, \tag{2.4.26}$$

giving $R = LP^T$, where L is lower triangular. This is equivalent to postmultiplying the matrix R by orthogonal transformations to get a lower triangular matrix L. Combining these two factorizations gives

$$A\Pi = Q\begin{pmatrix} L \\ 0 \end{pmatrix} P^T. \tag{2.4.27}$$

This factorization was introduced by Stewart [272, 1999] and called the **QLP factorization** of A. If Householder reflectors are used in the the second decomposition (2.4.26), then no advantage can be taken of the triangular form of R^T. If Givens rotations are used, the triangular form can be exploited as follows. In the first step a sequence of plane rotations is used to zero elements in the first column of R^T by rotating rows $(1, 2), (1, 3), \ldots, (1, n)$ in this order. This uses $2n^2$ flops and fills out

the first row of the matrix, but preserves the other zeros. The elements in the second column can now be zeroed similarly by rotations in the planes $(2, i)$, $i = 3 : n$, etc. In a typical intermediate step the matrix will have the form

$$\begin{array}{l} \\ \rightarrow \\ \rightarrow \\ \\ \\ \end{array} \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & + & & \\ & & \otimes & \times & & \\ & & \times & \times & \times & \\ & & \times & \times & \times & \times \end{pmatrix}.$$

This transformation, which requires a total of $2n^3/3$ flops, is called **flipping** the triangular matrix R^T. It can be considered as the first step in an iterative algorithm outlined in Sect. 3.5.3 for computing the SVD of R. The QLP factorization is used as a preprocessing step in the Jacobi SVD method; see Sect. 3.6.3.

Example 2.4.3 The diagonal elements of L often are quite good approximations to the singular values of A, and can be used to estimate condition numbers. Figure 2.4 shows a plot of the singular values of the matrix in Example 2.4.1 together with the diagonal elements of L in the QLP factorization. In the plot these values virtually coincide. In this example the correct numerical rank is revealed in both the QR and QLP factorizations. But the diagonal elements of L in the QLP factorization track the singular values much better and more smoothly. □

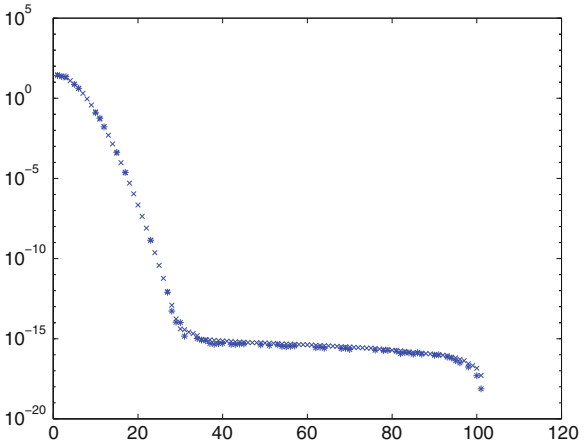


Fig. 2.4 Diagonal elements of L $(*)$ in the QLP factorization compared with singular values (x) of the matrix K

Suppose that after k steps of the Householder Algorithm 2.3.3 we have computed the partial QR factorization

$$A^{(k+1)} = (H_k \cdots H_1) A (\Pi_1 \cdots \Pi_k) = \begin{pmatrix} R_{11} & R_{12} \\ 0 & \widetilde{A}_{22} \end{pmatrix},$$

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

$$\begin{pmatrix} R_{11} & R_{12} \end{pmatrix} H_1 \cdots H_k = \begin{pmatrix} L_{11} & 0 \end{pmatrix},$$

where L_{11} is the first k rows of L in the QLP factorization. Hence, to determine the first k diagonal elements of L, which give the QLP approximations to the first k singular values of A, it is only necessary to perform k steps in each of the two factorizations.

The above observation shows that the two factorizations can be interleaved. That is, in the kth step first the kth row of R is computed and then the kth row of L. This is advantageous when, as in Example 2.4.1, the numerical rank is much less than n. In particular, if $\begin{pmatrix} r_{11} & r_{12} \end{pmatrix}$ is the first row of R, then a good estimate of $\sigma_1 = \|A\|_2$ is obtained in $O(n^2)$ operations from

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

For a lower or upper bidiagonal matrix B, flipping can be performed using a sequence of $n-1$ Givens rotations; the first two steps of which are shown below:

$$Q_1^T B = \begin{array}{c} \\ \rightarrow \\ \rightarrow \\ \\ \\ \\ \end{array} \begin{pmatrix} \times & + & & & & \\ \otimes & \times & & & & \\ & \times & \times & & & \\ & & \times & \times & & \\ & & & \times & \times & \end{pmatrix},$$

$$Q_2 Q_1^T B = \begin{array}{c} \\ \\ \rightarrow \\ \rightarrow \\ \\ \\ \end{array} \begin{pmatrix} \times & \times & & & & \\ & \times & + & & & \\ & \otimes & \times & & & \\ & & \times & \times & & \\ & & & \times & \times & \end{pmatrix},$$

The cost is only $2n$ multiplications and the generation of $n-1$ Givens rotations.

The use of flipping a triangular matrix is mentioned already by Faddeev et al. [96, 1968]. The convergence of an iterated QLP algorithm for computing the SVD is analyzed by Huckaby and Chan [171, 2003].

2.4.6 Modifying QR Factorizations

Suppose that we have computed the solution to a least squares problem

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

It may often be required to solve a related least squares problem, where simple modifications of A and b have been performed. For example, a problem already considered by Gauss is that one may want to add new observations or discard observations with unacceptably large residuals, without starting from scratch. Such modifications are often referred to as **updating** when (new) data are added and **downdating** when (old) data are removed.

In various time-series problems, data are arriving sequentially and a related least squares solution has to be updated at each time step. Applications in signal processing often require real-time solutions, so efficiency is critical. Other applications arise in active set methods for solving least squares problems with inequality constraints and in optimization and statistics. In linear regression, efficient and stable procedures for adding and/or deleting observations are often required. In stepwise regression, different models are examined by adding or deleting variables.

The bidiagonal decomposition and hence the SVD cannot be cheaply updated when A is modified by a rank-one matrix; see Bunch and Nielsen [38, 1978]. Therefore, we consider algorithms for updating the full QR factorization of $A \in \mathbb{R}^{m\times n}$, where the square orthogonal factor $Q \in \mathbb{R}^{m\times m}$ is *explicitly* known. We assume that A and the modified matrix $\widetilde{A}$ have full column rank, so that the factorizations are uniquely determined. These algorithms can be used also for updating a least squares solution by considering the QR factorization of the matrix

$$\begin{pmatrix} A & b \end{pmatrix} = Q \begin{pmatrix} R & z \\ 0 & \rho e_1 \end{pmatrix}. \tag{2.4.28}$$

From this the solution and residual norm of the least squares problem $\min_x \|Ax-b\|_2$ can be obtained by

$$Rx = z, \qquad \|r\|_2 = \rho. \tag{2.4.29}$$

In the following we derive algorithms for a general rank-one change of A as well as the special cases of adding or deleting a row or a column of A.

By the interlacing property of singular values (Theorem 2.2.9) it follows that when a row is added to A the smallest singular value of the modified matrix will not decrease. When a row is deleted, the smallest singular value and the rank may decrease and the problem can become singular. Similarly, when a column is deleted the smallest singular value will not decrease, but when a column is added the modified matrix may become singular. This observation indicates that adding a column or deleting a row are more sensitive operations. In some applications a **sliding window method** is used, where at each time step a new row of data is added and the oldest

row of data deleted. Then, adding a new row should be performed before an old row is deleted.

The updating algorithms given below are minor modifications of those given in LINPACK; see Dongarra et al. [74, 1979], p. 10.2.

2.4.6.1 Appending a Row

It is no loss of generality to assume that the new row v^T is appended as the last row. Since

$$\begin{pmatrix} Q^T & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} A \\ v^T \end{pmatrix} = \begin{pmatrix} R \\ 0 \\ v^T \end{pmatrix} = I_{n+1,m+1} \begin{pmatrix} R \\ v^T \\ 0 \end{pmatrix},$$

where $I_{n+1,m+1}$ interchanges rows $n+1$ and $m+1$, this problem can be reduced to appending v^T as an $(n+1)$st row to R. Determine Givens rotations $G_{k,n+1}$, $k = 1{:}n$, that annihilate the kth element in v^T, giving

$$G_{n,n+1} \cdots G_{1,n+1} \begin{pmatrix} R \\ v^T \end{pmatrix} = \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}.$$

This requires $3n^2$ flops. Note that R can be updated without Q being available. Updating Q using

$$\widetilde{Q} = \begin{pmatrix} Q & 0 \\ 0 & 1 \end{pmatrix} I_{n+1,m+1} G_{1,n+1}^T \cdots G_{n,n+1}^T,$$

requires $6mn$ flops.

2.4.6.2 Rank-One Change

Given $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$, it is required to compute the modified QR factorization

$$\widetilde{A} = A + uv^T = \widetilde{Q} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}. \tag{2.4.30}$$

The following updating algorithm is mixed backward stable:

1. Compute the vector $w = Q^T u \in \mathbb{R}^m$, so that

$$A + uv^T = Q \left[\begin{pmatrix} R \\ 0 \end{pmatrix} + wv^T \right]. \tag{2.4.31}$$

2. Determine an orthogonal transformation P such that

$$P\begin{pmatrix} R \\ 0 \end{pmatrix} + (Pw)v^T = \begin{pmatrix} R \\ z^T \\ 0 \end{pmatrix} + \beta e_{n+1} v^T, \quad \beta = \pm\|w\|_2. \tag{2.4.32}$$

This is done in two steps. First, a Householder transformation is used to zero the elements in w below row $n+1$. This will not change R. Next a sequence of Givens transformations $G_{k,n+1}$, $k = n : (-1) : 1$, is used to zero the first n elements in W from bottom up. These transformations will create a nonzero row z^T below the matrix R. Adding the second term in (2.4.32) will also just add to the same row. The process is illustrated in the following Wilkinson diagram ($m = 7$, $n = 4$):

$$\left(\begin{array}{ccc|c} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ \hline 0 & 0 & 0 & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & \times \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ \hline 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} \times & \times & \times & 0 \\ \times & \times & \times & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & 0 \\ \hline \times & \times & \times & \times \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0. \end{array}\right)$$

3. Determine a sequence of plane rotations $\widetilde{G}_{k,k+1}(\phi_k)$ to zero the elements in row $n+1$ as described in the algorithm for adding a row. This gives the factor $\widetilde{R}$.
4. Apply the transformations from previous steps to Q to obtain $\widetilde{Q}$,

The work needed for of R is $6n^2$ flops. Applying the Householder transformations to Q takes $4m(m-n)$ flops and applying the Givens transformations from steps 2 and 3 takes $12mn$ flops. This gives a total of $4m^2 + 8nm + 4n^2$ flops.

Remark 2.4.1 In an earlier (LINPACK) version of this algorithm the matrix R was modified into a Hessenberg matrix. The version given here is easier to implement since the modified row can be held in a vector. This becomes even more important for large sparse problems.

2.4.6.3 Deleting a Column

We first observe that deleting the last column of A is trivial. Assume that

$$A = \begin{pmatrix} A_1 & a_n \end{pmatrix} = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad R = \begin{pmatrix} R_{11} & u \\ 0 & r_{nn} \end{pmatrix},$$

where $A_1 = (a_1, \ldots, a_{n-1})$. Then the QR factorization of A_1 is obtained simply by deleting the trailing column from the decomposition. Suppose now that we want to compute the QR factorization

$$\widetilde{A} = (a_1, \ldots, a_{k-1}, a_{k+1}, \ldots, a_n),$$

where the kth column of A is deleted, $k < n$. From the above observation it follows that this decomposition can readily be obtained from the QR factorization of the matrix

$$AP_L = (a_1, \ldots, a_{k-1}, a_{k+1}, \ldots, a_n, a_k) \tag{2.4.33}$$

where P_L is a permutation matrix that performs a *left circular shift* of the columns $a_k, \ldots, a_n$. The matrix RP_L is upper Hessenberg, but the matrix $P_L^T RP_L$ is upper triangular except in its last row. For example, if $k = 3$, $n = 6$, then it has the structure

$$P_L^T RP_L = \left(\begin{array}{cc|ccc|c} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ \hline 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & 0 \\ \hline 0 & 0 & \times & \times & \times & \times \end{array}\right).$$

The task is now been reduced to constructing a sequence of Givens rotations $G_{i,n}$, $i = k : n-1$, to zero out the off-diagonal elements in the last row. Only the trailing principal submatrix of order $n-k+1$ in $P_L^T RP_L$, which has the form

$$\begin{pmatrix} R_{22} & 0 \\ v^T & r_{nn} \end{pmatrix},$$

participates in this transformation. Here the last column can be deleted. This remaining update of R_{22} is precisely the same as already described for adding a row. Finally, the updated Q factor is

$$\widetilde{Q} = QP_L G_{k,n}^T \cdots G_{n-1,n}^T.$$

By an obvious extension of the above algorithm, we obtain the QR factorization of the matrix resulting from a left circular shift applied to a set of columns $(a_1, \ldots, a_{k-1}, a_{k+1}, \ldots, a_p, a_k, a_{p+1}, \ldots a_n)$.

2.4.6.4 Appending a Column

Assume that the QR factorization

$$A = (a_1, \ldots, a_{k-1}, a_{k+1}, \ldots, a_n) = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

is known. We want to insert the column a_k, as the kth column, $k \neq n$, and compute the QR factorization of $\widetilde{A} = (a_1, \ldots, a_n)$. We start by appending a_k as the last column, which is straightforward. We first compute the vector

$$w = Q^T a_k = \begin{pmatrix} u \\ v \end{pmatrix} \begin{matrix} \} & n \\ \} & m-n \end{matrix}.$$

If $\gamma = \|v\|_2 = 0$, then $\widetilde{A}$ is singular. Otherwise, a Householder reflector H_n is constructed so that $H_n^T v = \gamma e_1$. We have now obtained the QR factorization

$$\begin{pmatrix} A & a_k \end{pmatrix} = \widetilde{Q} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}, \quad \widetilde{Q} = Q \begin{pmatrix} I_n & 0 \\ 0 & H_n \end{pmatrix}, \quad \widetilde{R} = \begin{pmatrix} R & u \\ 0 & \gamma \end{pmatrix}.$$

Let P_R be the permutation matrix that performs a *right circular shift* on the columns $a_{k+1}, \ldots, a_n, a_k$, so that

$$\widetilde{A} = \begin{pmatrix} A & a_k \end{pmatrix} P_R = \widetilde{Q} \begin{pmatrix} \widetilde{R} P_R \\ 0 \end{pmatrix}, \quad \widetilde{R} P_R = \begin{pmatrix} R_{11} & u_1 & R_{12} \\ 0 & u_2 & R_{22} \\ 0 & \gamma & 0 \end{pmatrix},$$

where $R_{11} \in \mathbb{R}^{(k-1)\times(k-1)}$ and $R_{22} \in \mathbb{R}^{(n-k)\times(n-k)}$ are upper triangular, e.g., for $k = 4, n = 6$:

$$\widetilde{R} P_R = \left(\begin{array}{ccc|c|cc} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ \hline 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & 0 & \times \\ \hline 0 & 0 & 0 & \times & 0 & 0 \end{array}\right).$$

Next, Givens rotations $G_{i-1,i}$, $i = n\!:-1\!:\!k$, are determined to zero the last $n-k+1$ elements in the kth column of $\widetilde{R} P_R$. Then

$$G_{k-1,k} \cdots G_{n-1,n} \begin{pmatrix} u_2 & R_{22} \\ \gamma & 0 \end{pmatrix} = \widetilde{R}_{22}$$

is upper triangular and the updated factors are

$$\overline{R} = \begin{pmatrix} R_{11} & \widetilde{R}_{12} \\ 0 & \widetilde{R}_{22} \end{pmatrix}, \quad \widetilde{R}_{12} = (u_1, R_{12}) \tag{2.4.34}$$

and $\overline{Q} = \widetilde{Q} G_{n-1,n}^T \cdots G_{k-1,k}$.

The above method easily generalizes to computing the QR factorization of

$$(a_1, \ldots, a_{k-1}, a_p, a_k, \ldots, a_{p-1}, a_{p+1}, \ldots, a_{n+1}),$$

i.e., of the matrix resulting from a right circular shift of the columns $a_k, \ldots, a_p$. Note that when a column is *deleted*, the new R-factor can be computed without Q being available. But when a column is *added*, it is essential that Q be known.

The algorithms given for appending and deleting a column correspond to the MATLAB functions `qrinsert(Q,R,k,ak)` and `qrdelete(Q,R,k)`.

2.4.6.5 Deleting a Row

Modifying the QR factorization when a row is *deleted* is a more difficult task than when a row is added. With no loss of generality we assume that it is the *first row* of A which is to be deleted. Thus, we wish to obtain the QR factorization of the matrix $\widetilde{A} \in \mathbb{R}^{(m-1)\times n}$ when the factorization

$$A = \begin{pmatrix} a_1^T \\ \widetilde{A} \end{pmatrix} = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{2.4.35}$$

is known. This is a special case of the rank-one change algorithm, when we take $u = -e_1$, $v^T = a_1^T$ in (2.4.30).

Consider the QR transformation of $(e_1,\ A)$, where a dummy column $e_1 = (1, 0, \ldots, 0)^T$ has been added. From (2.4.35) it follows that

$$Q^T(e_1,\ A) = \begin{pmatrix} q_1 & R \\ q_2 & 0 \end{pmatrix}, \tag{2.4.36}$$

where $q^T = (q_1^T,\ q_2^T) \in \mathbb{R}^m$ is the *first row* of Q. First determine a Householder transformation H such that $Hq_2 = \beta e_1$. Then Givens rotations $G_{k,n+1}, k = n{:}-1{:}1$, can be determined so that

$$G_{1,n+1} \cdots G_{n,n+1} \begin{pmatrix} q_1 \\ \beta \end{pmatrix} = \alpha e_{n+1}, \quad \alpha = \pm 1.$$

Applying these transformations to (2.4.36) gives

$$G_{1,n+1} \cdots G_{n,n+1} H \begin{pmatrix} q_1 & R \\ q_2 & 0 \end{pmatrix} = \begin{pmatrix} 0 & \widetilde{R} \\ \alpha & v^T \\ 0 & 0 \end{pmatrix}, \tag{2.4.37}$$

where the matrix $\widetilde{R} \in \mathbb{R}^{n\times n}$ is upper triangular and the row v^T has been added. Forming $\overline{Q} = QG_{n,n+1}^T \cdots JG_{1,n+1}^T$, it follows from (2.4.36) that the last row will be αe_{n+1}^T. But since $\overline{Q}$ is orthogonal, it must have the form

$$\overline{Q} = \begin{pmatrix} \widetilde{Q} & 0 \\ 0 & \alpha \end{pmatrix}$$

with $|\alpha| = 1$ and $\widetilde{Q} \in \mathbb{R}^{(m-1)\times(m-1)}$ orthogonal. Hence, we have obtained the factorization

$$\begin{pmatrix} 1 & a_1^T \\ 0 & \widetilde{A} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \widetilde{Q} \end{pmatrix} \begin{pmatrix} \alpha & v^T \\ 0 & \widetilde{R} \\ 0 & 0 \end{pmatrix}.$$

Deleting the first row and column in this equation gives $\widetilde{A} = \widetilde{Q} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}$, which is the desired factorization. Note the essential role played by the first row of Q in this algorithm.

The algorithms given above update the full QR factorization with $Q \in \mathbb{R}^{m\times m}$. When $m \gg n$ it is more economical to update the thin QR factorization $A = Q_1 R$, $Q_1 \in \mathbb{R}^{m\times n}$. Such algorithms, which use the Gram–Schmidt method with reorthogonalization combined with plane rotations, are given in Daniel [64, 1976]. Fortran subroutines implementing these algorithms are available in Reichel and Gragg [240, 1990]. Sometimes only the factor R is known, e.g., when the initial problem is large and sparse or has been solved by the normal equations. For deleting a row, Saunders [248, 1972] has given an algorithm that often is referred to as the LINPACK algorithm. This algorithm, which computes the first row q_1^T of Q_1 from $R^T q_1 = A^T e_1$ and takes $\|q_2^T\|_2 = (1 - \|q_1^T\|_2^2)^{1/2}$, is less stable the more expensive algorithm given above.

It is straightforward to extend the above updating algorithms to cases where a block of rows/columns are added or deleted. Such block algorithms are more amenable to efficient implementation on modern computers. Block methods for downdating have been studied by Eldén and Park [90, 1994] and Olszansky et al. [220, 1994].

Updating algorithms for the URV and ULV decompositions are given by Stewart in [268, 1992] and [269, 1993]. For recent work on updating UTV decompositions, see [12, 2005] and [10, 2008]. MATLAB templates for computing UTV decompositions are given by Fierro and Hansen [100, 2005]. Symmetric rank-revealing decompositions are studied by Hansen and Yalamov [153, 2001].

2.4.7 Stepwise Variable Regression

Consider a linear regression model

$$Ax = (a_1, a_2, \ldots, a_n)x = b, \quad A \in \mathbb{R}^{m\times n}.$$

In exploratory data analysis the number of variables may be large and some of them closely correlated. It is not unusual for there to be more variables than data, $n > m$. Limiting the model to a smaller number of variables gives a simpler model, which may fit almost as well. Models using few variables are often preferred for the sake

of simplicity and scientific insight. Stepwise regression is a greedy technique for selecting a suitable subset of variables.

In forward **stepwise regression** the model is built sequentially by adding one variable at a time. Initially, we set $x^{(0)} = 0$, and $r^{(0)} = b$. At each step, the variable added is chosen so that the residual norm is maximally decreased. Assume that at the current step k variables have entered the regression. Let the current least squares solution be $x^{(k)}$. In the next step, a column a_p is added that makes the smallest acute angle with the residual $r^{(k)} = b - Ax^{(k)}$. That is,

$$\cos(a_j, r^{(k)}) = \frac{\left|a_j^T r^{(k)}\right|}{\|a_j\|_2 \|r^{k)}\|_2},$$

is the maximized for $j = p$ over all variables not yet in the model. This amounts to using a different pivoting strategy in the QR factorization of A. Note that the standard pivoting strategy for computing a rank-revealing QR factorization makes no reference to b and therefore is not appropriate. Even when the given vector b is a multiple of one column in A, the full pivoted QR factorization of A may be computed before this is revealed.

Efroymson [82, 1960] gave an algorithm for stepwise regression based on Gauss–Jordan elimination on the normal equations. Methods based on orthogonal transformations show better stability. We describe here an algorithm by Eldén [84, 1972] that uses Householder QR factorization.

Assume that the data have been preprocessed by subtracting the mean values from b and the columns of A so that the transformed data satisfy $e^T A = 0$ and $e^T b = 0$, where $e = (1, \ldots, 1)^T$. To simplify notation, we further assume that the columns are ordered initially so that it is the first k variables that have entered the regression. At this step we have computed the partial QR factorization

$$Q_k^T(A,\ b) = (A^{(k)},\ b^{(k)}) = \left(\begin{array}{cc|c} R_{11}^{(k)} & R_{12}^{(k)} & c^{(k)} \\ 0 & \widetilde{A}_k^{(k)} & d^{(k)} \end{array}\right), \tag{2.4.38}$$

where $R_{11}^{(k)} \in \mathbb{R}^{k\times k}$ is upper triangular. We do not assume that the matrix Q_k is saved, since if $n \gg m$, this could require too much storage space. The regression coefficients are then obtained from the triangular system $R_{11}^{(k)} x_k = c^{(k)}$. The sums of squares due to the regression and the residual are $\|c^{(k)}\|_2^2$ and $\|d^{(k)}\|_2^2$, respectively, and

$$\|b\|_2^2 = \|c^{(k)}\|_2^2 + \|d^{(k)}\|_2^2.$$

Hence, all information is available to perform partial F-tests for the significance of a given variable in the regression.

Assume that in the next step column j, $k < j \le n$, is to be included. Then we proceed as follows. Consider the submatrix $\widetilde{A}^{(k)} = (\tilde{a}_k, \ldots, \tilde{a}_n)$ in (2.4.38) and construct a Householder transformation $\widetilde{H}_{k+1}$ such that

$$\widetilde{H}_{k+1}\tilde{a}_j^{(k)} = \sigma_j e_1, \quad \sigma_j = \|\tilde{a}_j^{(k)}\|_2 = r_{k+1,k+1}. \tag{2.4.39}$$

This determines the new column in $R_{11}^{(k+1)}$. The same orthogonal transformation is applied to $d^{(k)}$, giving

$$\widetilde{H}_{k+1}d^{(k)} = \begin{pmatrix} c_{k+1}^{(k+1)} \\ d^{(k+1)} \end{pmatrix}.$$

The variable to enter the regression should be chosen to decrease the norm of the residual as much as possible. By the Pythagorean theorem, the decrease in the squared residual norm is $\left(c_{k+1}^{(k+1)}\right)^2$. Using (2.4.39) to eliminate $\widetilde{H}_{k+1}$ gives

$$c_{k+1}^{(k+1)} = e_1^T \widetilde{H}_{k+1} d^{(k)} = \tilde{a}_j^T d^{(k)} / \|\tilde{a}_j\|_2, \tag{2.4.40}$$

is the quantity to maximize. This can be interpreted as minimizing the angle between the current residual and the reduced columns in the remaining part of A.

In regression analysis the covariance matrix of the regression coefficients $V_x = (R^T R)^{-1}$ is also required. If the lower triangular matrix $S = R^{-T}$ is known, then $V_x = S^T S$ is readily available. We now show how S is updated during the regression. Assume that $R^T S = I$ and that a new variable is added to the model. Then the updated matrices are determined from

$$\begin{pmatrix} R^T & 0 \\ r^T & \rho \end{pmatrix} \begin{pmatrix} S & 0 \\ s^T & \sigma \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & 1 \end{pmatrix}.$$

We obtain $\rho\sigma = 1$ and $r^T S + \rho s^T = 0$, giving $\sigma = 1/\rho$ and $s = -\sigma S^T r$. The cost of this updating is just $2k^2$ flops.

The following lemma shows that when R is multiplied from left and right by orthogonal transformations, the matrix $S = R^{-T}$ is transformed similarly. Therefore, it is convenient to store S in the lower triangular part of the array used for storing R.

Lemma 2.4.1 *Let R be a nonsingular matrix and $\widetilde{R} = Q_1 R Q_2$, where Q_1 and Q_2 are orthogonal. If $R^T S = I$, then $\widetilde{R}^T \widetilde{S} = I$, where $\widetilde{S} = Q_1 S Q_2$.*

Proof Using the orthogonality of Q_1 and Q_2, we have $Q_2^T R^T Q_1^T Q_1 S Q_2 = Q_2^T R^T S Q_2 = Q_2^T Q_2 = I$. □

After a new variable has entered the regression, it may be that the contribution of some other variable included in the regression is no longer significant. We now consider *deleting the jth column* from the regression using a technique slightly different from that described in Sect. 2.4.6. A product of $k-j$ Givens transformations Q are applied to $R_{11}^{(k)}$ so that when excluding the jth column, the rest of the matrix has triangular form. The rows $j{:}k$ in $R^{(k)}$ and $b^{(k)}$ will be affected. In the Wilkinson diagram below, the transformations of $R_{11}^{(k)}$ and $c^{(k)}$ are illustrated for the case $k = 5$ and $j = 3$. Premultiplying with $Q = G_{4,5}G_{3,4}$, we obtain

$$G_{4,5}G_{3,4}\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & + & \otimes & \times & \times \\ & & + & & \otimes & * \end{pmatrix}$$

A circular permutation of columns $j : k$ will bring the jth column into place k and a repartitioning of rows and columns gives the updated submatrix $R_{11}^{(k-1)}$ of order $k-1$. The same orthogonal transformations and repartitioning of rows are applied to $c^{(k)}$. The increase in the norm of the residual will come from the last element in $Q_k c^{(k)}$ (marked $*$ in the diagram), i.e., $e_k^T Q_k c^{(k)}$.

By Lemma 2.4.1, when a variable is deleted by the process just outlined, the lower triangular matrix $S = R^{-T}$ is transformed just like R:

$$G_{4,5}G_{3,4}\begin{pmatrix} \times & & & & \\ \times & \times & & & \\ \times & \times & \times & & \\ \times & \times & \times & \times & \\ \times & \times & \times & \times & \times \end{pmatrix} = \begin{pmatrix} \times & & & & \\ \times & \times & & & \\ \times & \times & \otimes & + & \\ \times & \times & \otimes & \times & + \\ \times & \times & \times & \times & \times \end{pmatrix}$$

By Lemma 2.4.1, permuting the jth column to the last position must give a lower triangular matrix. It follows that zeros are introduced in the jth columns as indicated. Denote the jth column of $S = R_{11}^{-T}$ by s_j. Then, in the general case, the orthogonal transformation Q_k is such that

$$Q_k s_j = \tau_j e_k, \quad \tau_j = \|s_j\|_2.$$

Hence, $s_j = \tau_j Q_k^T e_k$, and the increase in residual norm caused by deleting the jth variable, $1 \le j \le k$ in (2.4.38) equals

$$e_k^T Q_k c^{(k)} = s_j^T Q_k^T Q_k c^{(k)}/\tau_j = s_j^T c^{(k)}/\tau_j. \tag{2.4.41}$$

A potential variable to delete is determined by finding the minimum value of

$$|s_j^T c^{(k)}|/\tau_j, \quad j = 1{:}k.$$

If the increase in the residual norm for this j is not significant, then the jth variable is deleted from the regression.

It may be desirable to add or delete rows to $(A,\ b)$ when new information becomes available without recomputing the regression from scratch. It is possible to add this possibility to the stepwise algorithm described here using the tools described in Sect. 2.4.6.

Stepwise regression will in general not find the subsets of size $k, k = 1, 2, \ldots, p$, that give the smallest residual sum of squares. In certain cases, it might make the wrong choice in the first few steps and then waste much time in correcting this. There

are no guarantees that the process will not cycle. Trying all possible combinations is only feasible for small values of k. Furnival and Wilson [107, 1974] have developed an algorithm based on "branch and bound" techniques that can be used for p as large as 30.

Miller [207, 1982] surveys subset selection in regression and emphasizes the computational and conceptual advantages of using methods based on QR factorization rather than normal equations. Another approach based on the modified Gram–Schmidt process with reorthogonalization is described in Gragg et al. [137, 1979].

Exercises

2.4.1 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 the permutation Π to *maximize* the $(1, 1)$ element in the QR factorization of $R\Pi$. Show that this is achieved by taking

$$\Pi = \begin{cases} I_{1,2} & \text{if } |a| < \sqrt{b^2 + c^2}, \\ I & \text{otherwise,} \end{cases}$$

where $I_{1,2}$ 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

$$\Pi = \begin{cases} I & \text{if } |c^{-1}| \geq \sqrt{a^{-2} + b^2(ac)^{-2}}, \\ \Pi_{12} & \text{otherwise.} \end{cases}$$

Hence, the test compares *row* norms in R^{-1} instead of *column* norms in R.

2.4.2 (a) The general solution to a rank-deficient least squares problem is obtained by solving (see 2.4.13)

$$R_{11}x_1 = c_1 - R_{12}x_2,$$

where x_2 is arbitrary. To minimize $\|x\|_2$ is not always the best way to resolve the rank-deficiency and the following approach is often more appropriate.
For a given matrix $B \in \mathbb{R}^{p\times n}$ consider the problem

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

Show that with $R_{11}C = R_{12}$, $R_{11}d = c_1$, this amounts to solving

$$\min_{x_2} \|(BC)x_2 - (Bd)\|_2.$$

2.4.3 A rank-revealing LU factorization of $A \in \mathbb{R}^{m\times n}$ with rank $(A) = r$ has 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 suitable permutation matrices and $L_{11}, U_{11} \in \mathbb{R}^{r\times r}$ are triangular and nonsingular. Such a factorization can also be used to compute the pseudoinverse solution $x = A^{\dagger}b$. Show, using Theorem 2.2.3, 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 $r = m$.) *Remark:* To obtain a rank-revealing LU factorization, complete or rook pivoting must be used.

2.4.4 The QLP factorization of $A \in \mathbb{R}^{m\times n}$ is the two-sided orthogonal factorization

$$Q_1^T A \Pi\, Q_2 = \begin{pmatrix} R \\ 0 \end{pmatrix} Q_2 = \begin{pmatrix} L \\ 0 \end{pmatrix}, \tag{2.4.42}$$

where Q_1 and Q_2 are orthogonal matrices and L is lower triangular. Show that the transformations from the left and right can be interleaved. What is the operation count then for performing the first k steps?

2.4.5 (a) The normal equations for the least squares problem $\min_x \|Ax-b\|_2$ are $A^TAx = A^Tb$, with covariance matrix $V = (A^TA)^{-1}$. If an equation $w^Tx = \beta$ is added, then the updated solution $\widetilde{x}$ satisfies $(A^TA + ww^T)\widetilde{x} = A^Tb + \beta w$. Show that the updated covariance matrix is

$$\widetilde{V} = V - \frac{1}{1 + w^Tu} uu^T, \quad u = Vw.$$

(b) Show that the modified least squares solution satisfies

$$(A^TA + ww^T)(\widetilde{x} - x) = (\beta - w^Tx)w.$$

Use this and the result from (a) to show that the updated solution is

$$\widetilde{x} = x + (\beta - w^Tx)\widetilde{u}, \quad \widetilde{u} = \widetilde{C}w.$$

Comment: The simplicity and recursive nature of this updating algorithm has made it popular for many applications, notably Kalman filtering. The main disadvantage of the algorithm is its serious sensitivity to roundoff errors.

2.5 Structured and Sparse Least Squares

Kronecker structures 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 A and B are often large, resulting in models involving several hundred thousand equations and unknowns, such savings may be essential; see Fausett and Fulton [99, 1994].

A useful technique called **substructuring** or dissection gives rise to problems of block angular form. The idea is similar to (but preceded) the nested dissection method presented in Sect. 1.7.4. It dates back to 1880, when Helmert [159, 1980] used it for breaking down geodetic problems into geographically defined subproblems.

Another frequently occurring structure is when in each row all nonzero elements in A are contained in a narrow band. Banded and block angular least squares problems

are the simplest examples of least squares problems where A is sparse. Often the sparsity pattern of A is irregular and techniques introduced in Sect. 1.7 have to be used.

2.5.1 Kronecker Products

Sometimes least squares problems have a highly regular block structure. Here we consider problems of the form

$$\min_x \|(A \otimes B)x - c\|_2, \quad c = \mathrm{vec}C, \tag{2.5.1}$$

where $A \otimes B$ is the **Kronecker product** of $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$. This product is the $mp \times nq$ block matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}.$$

We recall from Sect. 1.8.1 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 pseudoinverses.

Lemma 2.5.1 *Let $A^\dagger$ and $B^\dagger$ be the pseudoinverses of A and B. Then*

$$(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger.$$

Proof We can verify that $X = A^\dagger \otimes B^\dagger$ satisfies the four Penrose conditions in (2.2.8)–(2.2.9). □

By Lemmas 1.8.1 and 2.5.1, the solution to the Kronecker least squares problem (2.5.1) can be written

$$x = (A \otimes B)^\dagger c = (A^\dagger \otimes B^\dagger)\mathrm{vec}C = \mathrm{vec}(B^\dagger C\,(A^\dagger)^T). \tag{2.5.2}$$

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 solving the least squares problems (2.5.1) is to use the normal equations. If we assume that both A and B have full column rank, we can use in (2.5.2)

$$A^\dagger = (A^T A)^{-1} A^T, \qquad B^\dagger = (B^T B)^{-1} B^T.$$

But in general an approach based on orthogonal factorizations should be preferred. If we have computed the complete QR factors of A and B,

$$A\Pi_A = Q_A \begin{pmatrix} R_A & 0 \\ 0 & 0 \end{pmatrix} V_A^T, \qquad B\Pi_B = Q_B \begin{pmatrix} R_B & 0 \\ 0 & 0 \end{pmatrix} V_B^T,$$

with R_A, R_B upper triangular and nonsingular, then (see Sect. 2.7.3) we have

$$A^\dagger = \Pi_A V_A \begin{pmatrix} R_A^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q_A^T, \qquad B^\dagger = \Pi_B V_B \begin{pmatrix} R_B^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q_B^T.$$

These expressions can be used in (2.5.2) to compute the pseudoinverse solution of problem (2.5.1) even in the rank-deficient case.

The SVD of a Kronecker product $A \otimes B$ can be simply expressed in terms of the SVDs of A and B, say

$$A = U_A \Sigma_A V_A^T, \qquad B = U_B \Sigma_B V_B^T.$$

From Lemma 2.5.1 it follows that

$$A \otimes B = (U_A \otimes U_B)(\Sigma_A \otimes \Sigma_B)(V_A \otimes V_B)^T. \tag{2.5.3}$$

This is the SVD of $A \otimes B$, except that the singular values in the diagonal matrix $\Sigma_A \otimes \Sigma_B$ are not necessarily in nondecreasing order. With $c = \mathrm{vec}(C)$, the solution can be written as

$$x = \mathrm{vec}(X), \quad X = (B^\dagger C)(A^\dagger)^T. \tag{2.5.4}$$

2.5.2 Tensor Computations

In many applications the data structures encountered have more than two dimensions and are represented by a multidimensional **tensor** or its coordinate representation, i.e., a **hypermatrix**. Tensors will be denoted by calligraphic letters, e.g., we refer to

$$\mathcal{A} = (a_{i_1,\ldots,i_d}) \in \mathbf{R}^{n_1 \times \cdots \times n_d} \tag{2.5.5}$$

as a d-mode tensor $d > 2$. The case $d = 2$ corresponds to matrices. In the following discussion we emphasize the case $d = 3$, because the main difference between matrices and hypermatrices comes from the transition from $d = 2$ to $d = 3$.

Tensor decompositions originated with Hitchcock [164, 1927] and were much later taken up and used successfully to analyze data in psychometrics (Tucker [283, 1966]). In the last decades the use of tensor methods has spread to other fields such as chemometrics (Bro [36, 1997]), signal and image processing, data mining and pattern recognition (Eldén [89, 2007]). Mathematical theory and new computational methods are rapidly being developed. Here we can only give short introduction to concepts. We caution the reader that notation is still in its infancy and varies between different papers.

Subarrays are formed by keeping a subset of the indices constant. A 3-mode tensor (2.5.5) can be thought of as being built up by matrix *slices* in three ways by fixing one of the indices, e.g.,

$$(a_{:,:,j}) \in \mathbb{R}^{n_1 \times n_2}, \quad j = 1{:}n_3.$$

Similarly, fixing any two indices we get a vector, or *fiber*

$$(a_{:,j,k}) \in \mathbb{R}^{n_1}, \quad j = 1{:}n_2 \quad k = 1{:}n_3.$$

A tensor is said to be **symmetric** if its elements are equal under any permutations of the indices, i.e., for a 3-mode tensor

$$a_{i,j,k} = a_{i,k,j} = a_{j,k,i} = a_{j,i,k} = a_{k,i,j} = a_{k,j,i}, \quad \forall\, i, j, k.$$

see Comon et al. [57, 2008]. A tensor is **diagonal** if $a_{i_1,i_2,\ldots,i_d} \neq 0$ only if $i_1 = i_2 = \cdots = i_d$.

Elementwise addition and scalar multiplication trivially extends to hypermatrices of arbitrary order. The tensor or **outer product** is denoted by $\circ$ (not to be confused with the Hadamard product of matrices). For example, if $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ and $B = (b_{kl}) \in \mathbb{R}^{p \times q}$ are matrices, then

$$\mathcal{C} = A \circ B = (a_{i,j,k,l})$$

is a 4-mode tensor. The 1-mode **contraction product** of two 3-mode hypermatrices $\mathcal{A} = (a_{i,j,k}) \in \mathbf{R}^{n \times n_2 \times n_3}$ and $\mathcal{B} = (b_{i,l,m}) \in \mathbf{R}^{n \times m_2 \times m_3}$ with conforming first dimension is the 4-mode tensor $\mathcal{C} \in \mathbf{R}^{n_2 \times n_3 \times m_2 \times m_3}$ defined as

$$\mathcal{C} = \langle \mathcal{A}, \mathcal{B} \rangle_1, \quad c_{j,k,l,m} = \sum_{i=1}^{n} a_{i,j,k} b_{i,l,m}. \tag{2.5.6}$$

Contractions need not be restricted to one pair of indices at a time. The inner product of two 3-mode tensors of the same size and the Frobenius norm of a tensor are defined as

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i,j,k} a_{ijk} b_{ijk}, \qquad \|\mathcal{A}\|_F^2 = \langle \mathcal{A}, \mathcal{A} \rangle^{1/2} = \sum_{i,j,k} a_{ijk}^2. \tag{2.5.7}$$

The matrix Hölder norm for $p = 1, \infty$ is similarly generalized.

The columns of the matrix A can be stacked or **unfolded** into a column vector, the operation vec(A). A second way would be to unfold its rows into a row vector. Similarly, a 3-mode tensor $\mathcal{A}$ can be unfolded or **matricized** by stacking in some order the matrix slices obtained by fixing one of its three modes. Following Eldén and Savas [92, 2009], we use the notation

$$\begin{aligned} A_{(1)} &= (A_{:,1,:}, A_{:,2,:}, \ldots, A_{:,n_2,:}) \in \mathbf{R}^{n_1 \times n_2 n_3}, \\ A_{(2)} &= (A_{:,:,1}^T, A_{:,:,2}^T, \ldots, A_{:,:,n_3}^T) \in \mathbf{R}^{n_2 \times n_1 n_3}, \\ A_{(3)} &= (A_{1,:,:}^T, A_{2,:,:}^T, \ldots, A_{n_1,:,:}^T) \in \mathbf{R}^{n_3 \times n_1 n_2}, \end{aligned} \tag{2.5.8}$$

where a colon is used to indicate all elements of a mode. Different papers sometimes use different orderings of the columns. The specific permutation is not important as long as it is consistent.

A matrix $C \in \mathbb{R}^{p \times q}$ can be multiplied from the left and right by other matrices $X \in \mathbb{R}^{m \times p}$ and $Y \in \mathbb{R}^{n \times q}$, and we write

$$A = XCY^T, \qquad a_{ij} = \sum_{\alpha=1}^{p} \sum_{\beta=1}^{q} x_{i\alpha} y_{j\beta} c_{\alpha\beta}.$$

The corresponding tensor-matrix multiplication of a 3-mode tensor $\mathcal{C} \in \mathbb{R}^{p \times q \times r}$ by *three* matrices $X \in \mathbb{R}^{l \times p}$, $Y \in \mathbb{R}^{m \times q}$, and $Z \in \mathbb{R}^{n \times r}$ transforms $\mathcal{C}$ into the 3-mode tensor $\mathcal{A} \in \mathbb{R}^{l \times m \times n}$ with entries

$$a_{ijk} = \sum_{\alpha=1}^{p} \sum_{\beta=1}^{q} \sum_{\gamma=1}^{r} x_{i,\alpha} y_{j,\beta} z_{k\gamma} c_{\alpha\beta\gamma}, \tag{2.5.9}$$

A convenient notation for this operation, suggested by Silva and Lim [256, 2008], is

$$\mathcal{C} = (X, Y, Z) \cdot \mathcal{A}, \tag{2.5.10}$$

where the mode of each multiplication is understood from the ordering of the matrices.

For a matrix $A \in \mathbb{R}^{m \times n}$ there are three ways to define the rank r, which all yield the same value. The rank is equal to the dimension of the subspace of $\mathbb{R}^m$ spanned by its columns. It also equals the dimension of the subspace of $\mathbb{R}^n$ spanned by its rows. Also, the minimum number of terms in the expansion of A as a sum of rank one matrices is equal to r; cf. the SVD expansion. For a tensor of mode $d > 2$ these three definitions yield different results.

The column- and row-rank of a matrix are generalized as follows. For a 3-mode tensor $\mathcal{A} =\in \mathbf{R}^{n_1 \times n_2 \times n_3}$, let r_1 be the dimension of the subspace of $\mathbb{R}^{n_1}$ spanned by

the $n_2 n_3$ vectors with entries

$$a_{:,i_2,i_3}, \quad i_2 = 1{:}n_2, \quad i_3 = 1{:}n_3.$$

In other words, $r_1(\mathcal{A}) = \text{rank}\,(A_{(1)})$, with similar interpretations for r_2 and r_3. The triple (r_1, r_2, r_3) is called the **multirank** of $\mathcal{A}$, and r_1, r_2, r_3 can all be different.

The outer product of vectors $x \in \mathbb{R}^l$, $y \in \mathbb{R}^m$, and $z \in \mathbb{R}^n$ is the 3-mode hypermatrix

$$\mathcal{T} = x \circ y \circ z \in \mathbb{R}^{l \times m \times n}, \quad t_{i_1 i_2 i_3} = x_{i_1} y_{i_2} z_{i_3}. \tag{2.5.11}$$

If nonzero, we call this a rank-one tensor. The **tensor rank** of $\mathcal{A}$ is the smallest number r such that $\mathcal{A}$ may be written as a sum of rank-one hypermatrices,

$$\mathcal{A} = \sum_{p=1}^{r} x_p \circ y_p \circ z_p. \tag{2.5.12}$$

When $d = 2$ this definition agrees with the usual definition of the rank of a matrix. Generalization of this rank definition to higher order tensors is straightforward. However, for $d \geq 3$ there is no algorithm for determining the rank of a given tensor and this problem is NP-hard. Furthermore, de Silva and Lim [256, 2008] show that the problem of finding the best rank-p approximation in general has no solution even for $d = 3$.

In many applications one would like to approximate a given tensor $\mathcal{A}$ with a sum of rank-one tensors so that

$$\left\| \mathcal{A} - \sum_{i=1}^{p} \lambda_i x_i \circ y_i \circ z_i \right\|_F \tag{2.5.13}$$

is minimized. Here weights λ_i are introduced so that we can assume that the vectors x_i, y_i, and z_i are normalized to have length one. Since the problem of determining the rank of a tensor is NP-hard, we assume that the number $p < r$ factors is fixed. A frequently used algorithms for computing such an approximate CP decomposition is the alternating least squares (ALS) method. In the ALS method the vectors y_i and z_i are first fixed and x_i determined to minimize (2.5.13). Next, x_i, z_i are fixed and one solves for y_i, and then x_i, y_i are fixed and one solves for z_i. Define the matrices

$$X = (x_1, \ldots, x_p) \in \mathbb{R}^{n_1 \times p}, \quad Y = (y_1, \ldots, y_p) \in \mathbb{R}^{n_2 \times p},$$
$$Z = (z_1, \ldots, z_p) \in \mathbb{R}^{n_3 \times p}.$$

With y_i, z_i fixed, the minimizing problem can be written in matrix form as

$$\min_X \|A_{(1)} - \widehat{X}(Z \odot Y)^T\|_F.$$

Here $A_{(1)} \in \mathbb{R}^{n_1 \times n_2 n_3}$ is the matrix obtained by unfolding $\mathcal{A}$ along the first mode, and

$$Z \odot Y = (z_1 \otimes y_1, \ldots, z_p \otimes y_p) \in \mathbb{R}^{n_2 n_3 \times p}$$

is the matching column-wise Kronecker product, also called the Khatri–Rao product, of Z and Y. The solution can be written

$$\widehat{X} = A_{(1)}[(Z \odot Y)^T]^\dagger,$$

and then the columns of $\widehat{X}$ are normalized to give $\widehat{X} = X\mathrm{diag}(\lambda_i)$. Because of the special form of the Khatri–Rao product, the solution can also be written as

$$\widehat{X} = X_{(1)}(Z \odot Y)(Z^T Z .* Y^T Y)^\dagger,$$

where $.*$ is the Hadamard (elementwise) matrix product. This version is not always suitable due to the squaring of the condition number.

Similar formulas for the two other modes are easy to derive. At each inner iteration a pseudoinverse must be computed. The ALS method can take many iterations to converge and is not guaranteed to converge to a global minimum. The solution obtained depends on the starting guess as well.

The idea of expressing a tensor as a sum of rank-one tensors has been proposed by several authors under different names. In psychometrics the it was called CANDECOMP (canonical decomposition) and PARAFAC (parallel factors); see Kolda and Bader [180, 2009]. Here we call it the CP (CANDECOMP/PARAFAC) decomposition (Leurgans et al. [192, 1993]).

In matrix computations an important role is played by the SVD

$$A = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T \in \mathbb{R}^{m \times n}, \quad r \le \min\{m, n\}. \tag{2.5.14}$$

This expresses a matrix A of rank r as the weighted sum of rank-one matrices $u_i v_i^T$, where $u_i \in \mathbb{R}^m$ and $v_i \in \mathbb{R}^n$, $i = 1{:}r$, are mutually orthogonal. The SVD expansion has the desirable property that for any unitarily invariant norm, the best approximation of A by a matrix of rank $r < n$ is obtained by truncating the expansion; see the Eckart–Young Theorem 2.2.11.

The high order SVD (HOSVD) is a generalization of the SVD decomposition to 3-mode hypermatrices

$$\mathcal{A} = (U, V, W) \cdot \mathcal{C},$$

where the matrices U, V, and W are square and orthogonal and $\mathcal{C}$ has the same size as $\mathcal{A}$. Furthermore, the different matrix slices of $\mathcal{C}$ are mutually orthogonal (with respect to the standard inner product on matrix spaces) and with decreasing Frobenius norm. Due to the imposed orthogonality conditions, the HOSVD of $\mathcal{A}$ is essentially unique. It is rank-revealing in the sense that if $\mathcal{A}$ has multirank (r_1, r_2, r_3), then the last $n_1 - r_1, n_2 - r_2$, and $n_3 - r_3$ slices along the different modes of the core tensor $\mathcal{C}$ are

zero matrices. Algorithms for computing the HOSVD are described by Lauthouwer et al. [187, 2000]). The matrix U is obtained from the SVD of the $l \times mn$ matrix obtained from unfolding $\mathcal{A}$. V and W are obtained in the same way. Since U, V, and W are orthogonal, $\mathcal{C} = (c_{ijk})$ is then easily computed from $\mathcal{C} = (U^T, V^T, W^T) \cdot \mathcal{A}$.

Suppose we want to approximate the tensor $\mathcal{A}$ by another tensor $\mathcal{B}$ of lower multirank. Then we want to solve

$$\min_{\text{rank}\,(\mathcal{B})=(p,q,r)} \|\mathcal{A} - \mathcal{B}\|_F, \tag{2.5.15}$$

where the Frobenius tensor norm is defined in (2.5.7). This is the basis of the Tucker model [283, 1966]. Unlike the matrix case, this problem can not be solved by truncating the HOSVD of $\mathcal{A}$. It is no restriction to assume that $\mathcal{B} = (U, V, W) \cdot \mathcal{C}$, where $U \in \mathbb{R}^{\ell \times p}$, $V \in \mathbb{R}^{m \times q}$, and $W \in \mathbb{R}^{\ell \times p}$ are orthogonal matrices. Due to the orthogonal invariance of the Frobenius norm, U, V, and W are only determined up to a rotation. Eliminating the core tensor $\mathcal{C}$, problem (2.5.15) can be rewritten as a maximization problem with the objective function

$$\Phi(U, V, W) = \frac{1}{2}\|(U^T, V^T, W^T) \cdot \mathcal{A}\|_F^2,$$

subject to $U^T U = I$, $V^T V = I$, and $W^T W = I$ (compare with the corresponding matrix problem for $d = 2$). It can be solved as an optimization problem on a Stiefel manifold; see Eldén and Savas [92, 2009] and Savas and Lim [250, 2010]. A framework of Newton algorithms with orthogonality constraints is given by Edelman et al. [80, 1999].

An extensive survey of tensor methods is given by Kolda and Bader [180, 2009]. The theory of tensors and hypermatrices is surveyed by Lim [195, 2013]. Tensor rank problems are studied by De Silva and Lim [256, 2008] and Comen et al. [56, 2009]. A tutorial on CP decomposition and its applications is given by Bro [36, 1997]). The N-way Toolbox for MATLAB (Andersson and Bro [3, 2000]) for analysis of multiway data can be downloaded from http://www.models.kvl.dk/source/. Tools for tensor computations in MATLAB have also been developed by Bader and Kolda [7, 2006] and [8, 2007]; see also the MATLAB Tensor Toolbox http://www.sandia.gov/~tgkolda/TensorToolbox/index-2.5.html.

2.5.3 Block Angular Least Squares Problems

Consider a geodetic network consisting of geodetic stations connected through observations. To each station corresponds a set of unknown coordinates to be determined. In substructuring a set of stations $\mathcal{B}$ are chosen that separates the other stations into two regional blocks $\mathcal{A}_1$ and $\mathcal{A}_2$ such that station variables in $\mathcal{A}_1$ are not connected by observations to those in $\mathcal{A}_2$. The variables are then ordered so that those in $\mathcal{A}_1$ appear

first, $\mathcal{A}_2$ second, in $\mathcal{B}$ last. The equations are then ordered so that those including only $\mathcal{A}_1$ come first, $\mathcal{A}_2$ next, and those only involving variables in $\mathcal{B}$ come last. The dissection can be continued by dissecting each of the regions $\mathcal{A}_1$ and $\mathcal{A}_2$ 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 Fig. 2.5. The corresponding block structure induced in the matrix are

$$A = \left(\begin{array}{cc|c} A_1 & & B_1 \\ & A_2 & B_2 \end{array}\right), \qquad A = \left(\begin{array}{cccc|cc} A_1 & & & & C_1 & B_1 \\ & A_2 & & & C_2 & B_2 \\ & & A_3 & & D_3 & B_3 \\ & & & A_4 & D_4 & B_4 \end{array}\right).$$

The block of rows corresponding to $\mathcal{A}_i, i = 1, 2, \ldots$, can be processed independently. The remaining variables are then eliminated, etc. 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 the number of variables in the two partitions is roughly the same. Also, the number of variables in the separator nodes should be as small as possible. Nested dissection and orthogonal factorizations in geodetic survey problems are studied by Golub and Plemmons [129, 1980].

We consider now least squares problems of the following bordered block diagonal or **block angular form**:

$$A = \left(\begin{array}{ccc|c} A_1 & & & B_1 \\ & A_2 & & B_2 \\ & & \ddots & \vdots \\ & & A_M & B_M \end{array}\right), \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \\ x_{M+1} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{pmatrix}, \tag{2.5.16}$$

where $A_i \in \mathbb{R}^{m_i \times n_i}$, $B_i \in \mathbb{R}^{m_i \times n_{M+1}}$, $i = 1{:}M$, and

$$m = m_1 + m_2 + \cdots + m_M, \qquad n = n_1 + n_2 + \cdots + n_{M+1}.$$

This is a special instance of the two-block form (2.1.20), where the first block has a special structure. 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 prob-

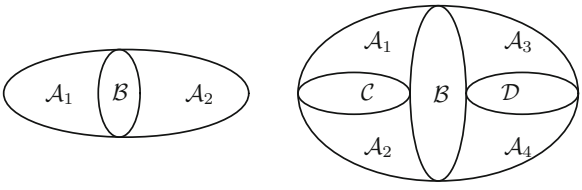


Fig. 2.5 One and two levels of dissection of a region

lem. Applications where the form (2.5.16) arises naturally include photogrammetry, Doppler radar, 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 [60, 1990].

The normal equations of the least squares problem where A and b have the form (2.5.16) has the doubly bordered block diagonal form:

$$A^TA = \left(\begin{array}{cccc|c} A_1^TA_1 & & & & A_1^TB_1 \\ & A_2^TA_2 & & & A_2^TB_2 \\ & & \ddots & & \vdots \\ & & & A_M^TA_M & A_M^TB_M \\ \hline B_1^TA_1 & B_2^TA_2 & \cdots & B_M^TA_M & C \end{array}\right), \quad C = \sum_{k=1}^{M} B_k^TB_k. \tag{2.5.17}$$

If rank $(A) = n$, then the Cholesky factor of A^TA is nonsingular and has a block structure similar to that of A:

$$R = \left(\begin{array}{cccc|c} R_1 & & & & S_1 \\ & R_2 & & & S_2 \\ & & \ddots & & \vdots \\ & & & R_M & S_M \\ \hline & & & & R_{M+1} \end{array}\right). \tag{2.5.18}$$

Identifying the blocks in the relation $R^TR = A^TA$, we get

$$R_i^TR_i = A_i^TA_i, \qquad R_i^TS_i = A_i^TB_i,$$
$$C = R_{M+1}^TR_{M+1} + \sum_{i=1}^{M} S_i^TS_i, \quad i = 1{:}M.$$

We start by computing the Cholesky factors $R_i \in \mathbb{R}^{n_i \times n_i}$, of $A_i^TA_i$ and solving the triangular systems $R_i^TS_i = A_i^TB_i$, for S_i, $i = 1{:}M$. Next, we form

$$\widetilde{C} = C - \sum_{i=1}^{M} S_i^TS_i$$

and compute its Cholesky factor, which is R_{M+1}. The right hand side of the normal equations is $f = A^Tb$, where

$$f_i = A_i^Tb_i, \quad i = 1{:}M, \qquad f_{M+1} = A_{M+1}^Tb_{M+1} + \sum_{i=1}^{M} B_i^Tb_i.$$

The solution is then obtained by solving $R^T y = f$ and $Rx = y$:

$$R_i^T y_i = f_i, \quad i = 1{:}M, \qquad R_{M+1}^T y_{M+1} = f_{M+1} - \sum_{i=1}^{M} S_i^T y_i, \quad (2.5.19)$$

$$R_{M+1} x_{M+1} = y_{M+1}, \qquad R_i x_{=} y_i - S_i x_{M+1}, \quad i = M{:}-1{:}1, \quad (2.5.20)$$

using block forward and back substitution. Note that much of the computations can be performed in parallel on M subsystems.

It is usually preferable to solve block angular least squares problems using QR factorizations. This algorithm 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

   $$Q_i^T (A_i,\ B_i) = \begin{pmatrix} R_i & S_i \\ 0 & T_i \end{pmatrix}, \qquad Q_i^T b_i = \begin{pmatrix} c_i \\ d_i \end{pmatrix}, \quad (2.5.21)$$

   where Q_i and R_i are the QR factors of A_i.

   (b) Apply orthogonal transformations P_i to update

   $$\begin{pmatrix} R_{M+1} & c_{M+1} \\ 0 & f_i \end{pmatrix} := P_i \begin{pmatrix} R_{M+1} & c_{M+1} \\ T_i & d_i \end{pmatrix}. \quad (2.5.22)$$

   (c) Update the residual norm $\rho = (\rho^2 + \| f_i \|_2^2)^{1/2}$.

3. Solve by back substitution the triangular systems

$$R_{M+1} x_{M+1} = c_{M+1}, \qquad R_i x_i = c_i - S_i x_{M+1}, \quad i = 1{:}M. \quad (2.5.23)$$

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, provided that 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 marginally increase the operation count.

In order to estimate the accuracy of the results, it is often required to estimate elements of the covariance matrix

$$c_{ij} = e_i^T C e_j = e_i^T R^{-1} R^{-T} e_j.$$

Then $c_{ij} = u_i^T u_j$, where u_i and u_j are the solutions of the triangular systems $R^T u_k = e_k$, $k = i, j$, and from (2.5.18),

$$R^T = \begin{pmatrix} R_1^T & & & & \\ & R_2^T & & & \\ & & \ddots & & \\ & & & R_M^T & \\ S_1^T & S_2^T & \cdots & S_M^T & R_{M+1}^t \end{pmatrix}.$$

2.5.4 Banded Least Squares Problems

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.

Definition 2.5.1 The **row bandwidth** of a matrix $A \in \mathbb{R}^{m\times n}$ is

$$w = \max_{1\le i\le m} (l_i - f_i + 1), \tag{2.5.24}$$

where

$$f_i = \min\{j \mid a_{ij} \neq 0\}, \qquad l_i = \max\{j \mid a_{ij} \neq 0\}, \tag{2.5.25}$$

are the column subscripts of the first and last nonzero in the ith row.

For a well-conditioned least squares problem the method of normal equations may give sufficiently accurate results. In this approach the matrix $C = A^T A$ is formed and its Cholesky factorization $C = LL^T$ computed. The Cholesky factor is independent of the row ordering of A. For if $B = PA$, where P is a permutation matrix, then

$$B^T B = A^T P^T PA = A^T A.$$

The least squares solution is then obtained by solving the triangular systems $Ly = A^T b$ and $L^T x = y$.

We now prove a relation between the row bandwidth of the matrix A and the bandwidth of the corresponding symmetric nonnegative definite matrix $A^T A$.

Theorem 2.5.1 *Let $A \in \mathbb{R}^{m\times n}$ have row bandwidth w. Then the symmetric matrix $A^T A$ has lower (and upper) bandwidth $r \le w - 1$.*

Proof From Definition 2.5.1 it follows that

$$|j-k| \geq w \quad \Rightarrow \quad a_{ij}a_{ik} = 0 \quad \forall\, i = 1{:}m, \tag{2.5.26}$$

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

Another proof of the lemma is obtained by using the expression

$$A^TA = \sum_{i=1}^{m} \widetilde{a}_i \widetilde{a}_i^T,$$

where $\widetilde{a}_i^T$, $i = 1{:}m$, is the ith row of A. Here A^TA is expressed as the sum of m symmetric matrices of rank one, each of which has lower (upper) bandwidth at most equal to $r = w - 1$. Then the lower (upper) bandwidth of the sum is also bounded by $w - 1$. Therefore, the normal equations of banded least squares problems can be solved very efficiently using the band Cholesky Algorithm 1.5.2.

Unless A is well-conditioned, a method based on the QR factorization of A should be preferred. Since the factor R equals the (unique) Cholesky factor of A^TA, it follows from Theorem 2.5.1 that only $r = w - 1$ diagonals in R will be nonzero. This indicates that it should be possible to take advantage of the band structure also in the QR factorization of A. This is indeed true, but less straightforward than for the normal equations. Let $A = Q_1R$ be the thin QR factorization of a banded matrix $A \in \mathbb{R}^{m\times n}$ of full column rank. Then R and $Q_1 = AR^{-1}$ are uniquely defined. But the inverse R^{-1} of a banded upper triangular matrix is a full upper triangular matrix. Therefore, Q_1 will be less sparse than A and R. *This rules out methods like Gram–Schmidt orthogonalization that explicitly compute* Q.

Householder or Givens QR factorizations represent Q implicitly and can still be used. However, the required work and intermediate storage requirement can differ considerably for different row orderings. A suitable initial row ordering is to sort the rows of A by leading entry order, i.e., so that

$$i \leq k \Rightarrow f_i \leq f_k.$$

Such a band matrix is said to be in **standard form**. Note that such an ordering is not unique.

We first consider Givens QR factorization of a matrix in standard form. The orthogonalization proceeds row-by-row. In step i the row a_i^T is merged with the triangular matrix R_{i-1} produced in earlier steps to give a triangular matrix R_i. In Fig. 2.6 we show the situation before the elimination of the ith row. The basic step is to merge a *full triangular* $w \times w$ *matrix* formed by rows and columns $f_i = f_i(A)$ to $l_i = l_i(A)$ of R with a row of elements in columns f_i to l_i. Note that only the indicated $w \times w$ upper triangular part of R_{i-1} is involved in this step. If A has constant bandwidth and is in standard form, then the last $n - l_i$ columns of R have not been touched and are still zero as initialized. Further, the first $f_i - 1$ rows of R are already finished at this stage and can be read out to secondary storage. Very

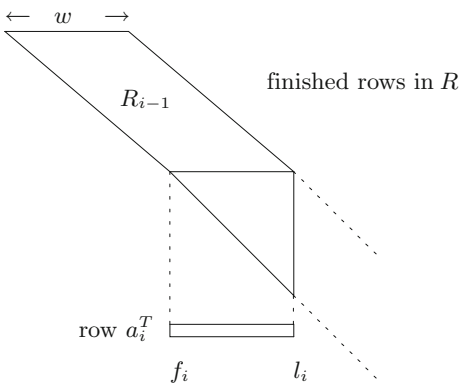


Fig. 2.6 The ith step of the QR factorization of a banded matrix

large problems can be handled because primary storage is needed only for the active triangular part. The following two cases can occur when merging a row ($w = 4$):

$$\text{case (i)} \quad \begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \\ \otimes & \otimes & \otimes & \otimes \end{bmatrix}; \qquad \text{case (ii)} \quad \begin{bmatrix} \times & \times & \times & \times & \\ & \times & \times & \times & + \\ & & \times & \times & + \\ & & & \times & + \\ & \otimes & \otimes & \otimes & \times \end{bmatrix}.$$

In case (ii) the first row does not participate and the active triangular matrix is shifted one step down.

If R is initialized as an $n \times n$ upper triangular band matrix of zeros, the description above is also valid for the processing of the initial n rows of A. Note that if at some stage $r_{jj} = 0$, then the whole jth row in R_{i-1} must be zero and the remaining part of the current row a_i^T can be inserted in row j of R_{i-1}. (This is a special case of a Givens rotation with $c = 0$, $s = 1$.) The number of rotations needed to process row a_i^T is at most $\min(i-1, w)$. A matrix $A \in \mathbb{R}^{m\times n}$ in standard form of bandwidth w can conveniently be stored in an $m \times w$ array, together with a vector p of pointers, where p_i points to the first row in A with $f_i(A) = i$, $i = 1\colon n$. The factor $R \in \mathbb{R}^{n\times n}$ can be stored in an $n \times w$ array. For a more detailed discussion; see Cox [59, 1981].

It is clear from the above that the processing of row a_i^T requires at most $3w^2$ flops if Givens rotations are used. Hence, the complete orthogonalization requires about $3mw^2$ flops, and can be performed in $\frac{1}{2}w(w+3)$ locations of primary storage. We remark that if the rows of A are processed in random order, then we can only bound the operation count by $3mnw$ flops, which is a factor of n/w worse. Thus, it almost invariably pays to sort the rows. The algorithm can be modified to handle problems with variable row bandwidth w_i. In this case an envelope data structure (see Definition 1.5.2) is used in which the factor R will fit.

If the Givens rotations are saved, they can be applied later to extra right-hand sides b to produce

$$c_i = Q^T b_i = \begin{pmatrix} c_i \\ d_i \end{pmatrix}, \quad c_i \in \mathbf{R}^n.$$

The least squares solution X_i is then obtained from $Rx_i = c_i$ by back substitution. If only the residual norm is needed, the vector d_i need not be stored, but used to accumulate the residual sum of squares $\|r_i\|_2^2 = \|d_i\|_2^2$. Each Givens rotation can be represented by a single floating point number as in (2.3.17). Since at most w rotations are needed to process each row, Q can be stored in no more space than allocated for A.

If Householder QR factorization is applied to an overdetermined banded matrix $A \in \mathbb{R}^{m\times n}$ with $m > n$, the Householder vectors tend to fill-in just as in MGS. An efficient Householder QR algorithm for banded least squares problems was first given by Reid [241, 1967]. To avoid unnecessary fill-in, the Householder reflections are split as follows. Assume that the matrix A is in standard form and partition A into blocks of rows as

$$A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_q \end{pmatrix}, \quad q \le n, \tag{2.5.27}$$

where in A_k each row has its first nonzero element in column k. In the QR factorization the blocks are processed sequentially in q major steps. In the first step the Householder QR factorization of the first block A_1 is computed, giving an upper trapezoidal matrix R_1. Next, R_{k-1}, $k = 2{:}q$, is merged with the block of rows A_k, yielding

$$\begin{pmatrix} R_k \\ 0 \end{pmatrix} = Q_k^T \begin{pmatrix} R_{k-1} \\ A_k \end{pmatrix}, \quad k = 2{:}q.$$

Since the rows of block A_k have their first nonzero elements in column k, the first $k-1$ rows of R_{k-1} will not be affected in this and later steps. The matrix Q can be implicitly represented in terms of Householder vectors of the factorization of the sub-blocks. This sequential Householder algorithm requires $(2m + 3n)w(w + 1)$ flops, or about twice the work of the less stable Cholesky approach. In order to understand how the algorithm proceeds, the reader is encouraged to work through the following example. A detailed description of the algorithm is given in Lawson and Hanson [190, 1974], Chap. 11.

Example 2.5.1 Consider the least squares approximation of a discrete set of data (y_i, t_i), $i = 1{:}m$, by a linear combination $s(t) = \sum_{j=1}^{n} x_j B_j(t)$, where $B_j(t)$, $j = 1{:}n$ are normalized cubic B-splines, with support on the interval $[t_j, t_{j+4}]$ (see Dahlquist and Björck [63, 2008], Sect. 4.4.3). This leads to the least squares problem to minimize

$$\sum_{i=1}^{m}(s(t_i)-y_i)^2 = \|Ax-y\|_2^2.$$

Since $B_j(t) = 0$ for $t \notin [t_j, t_{j+4}]$, the matrix A will be banded with $w = 4$. After the first three blocks have been reduced by Householder reflectors $P_1, \ldots, P_9$, the matrix has the form

$$\left[\begin{array}{ccc|ccccc}
\times & \times & \times & \times & & & & \\
1 & \times & \times & \times & + & & & \\
1 & 2 & \times & \times & + & + & & \\
\hline
 & 3 & 4 & \times & \times & + & & \\
 & 3 & 4 & 5 & \times & + & & \\
 & & 6 & 7 & 8 & \times & & \\
 & & 6 & 7 & 8 & 9 & & \\
 & & 6 & 7 & 8 & 9 & & \\
 & & & \times & \times & \times & \times & \\
 & & & \times & \times & \times & \times & \\
 & & & & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times
\end{array}\right].$$

Elements annihilated by P_j are denoted by j and fill elements by $+$. In later steps only the lower right part of the matrix is involved. □

The algorithms given in this section can easily be modified to work also for augmented band matrices of the form $A = \begin{pmatrix} A_1 & A_2 \end{pmatrix}$, where A_1 is a band matrix and A_2 is a set of full columns. These columns could correspond to multiple right-hand sides. For the standard linear model the covariance matrix is

$$V_x = (R^T R)^{-1} = R^{-1}R^{-T}. \tag{2.5.28}$$

The inverse matrix R^{-T} will be a full lower triangular matrix even when R is banded, and explicitly computing V_x should be avoided. The covariance of two linear functionals $f^T x$ and $g^T x$,

$$\operatorname{cov}(f^T x, g^T x) = \sigma^2 f^T V_x g = \sigma^2 (f^T R^{-1})(R^{-T} g) = \sigma^2 u^T v,$$

can be calculated from the lower triangular systems $R^T u = f$ and $R^T v = g$ by forward substitution. Here full advantage can be taken of the band structure R^T. The covariance of the components $x_i = e_i^T x$ and $x_j = e_j^T x$ is obtained by taking $f = e_i$ and $g = e_j$. Setting $i = j$ gives the variance of x_i.

2.5.5 Sparse Least Squares Problems

We now consider methods for least squares problems where the sparsity pattern of A is more irregular. If the method of normal equations is to be used, then many well developed techniques (see Sect. 1.7.4) for solving sparse symmetric positive definite systems can be applied to $A^TAx = A^Tb$.

The first step in computing the Cholesky factorization of $C = A^TA$ is a symbolic analyze phase. In this, the nonzero pattern of C is used to determine a fill reducing symmetric permutation P of C. Simultaneously, a storage scheme for the Cholesky factor of PCP^T is set up. To compute the nonzero pattern of A^TA, the matrix A is partitioned by rows. Let $a_i^T = e_i^T A$ be the ith row of A, so that

$$A^TA = \sum_{i=1}^{m} a_i a_i^T. \tag{2.5.29}$$

This expresses A^TA as the sum of m rank-one matrices. Invoking the no-cancellation assumption, this shows that the nonzero pattern of A^TA is the direct sum of the patterns of $a_i a_i^T$, $i = 1{:}m$. Note that the nonzeros in any row a_i^T will generate a full submatrix in A^TA. In the graph $G(A^TA)$ this corresponds to a subgraph where all pairs of nodes are connected. Such a subgraph is called a **clique**. Also note that the nonzero pattern of A^TA is not changed by dropping any row of A whose nonzero pattern is a subset of another row. This observation can often speed up the algorithm considerably.

It is possible to perform the symbolic factorization of A^TA operating directly on the structure of A. This algorithm due to George and Ng [119, 1987] removes the need for determining the structure of A^TA.

For ill-conditioned or stiff problems, methods based on the QR factorization should be preferred. Since the factor R in the QR factorization of A is mathematically equivalent to the upper triangular Cholesky factor R of A^TA, the nonzero structure is the same. But, because of the no-cancellation assumption, predicting the structure of R by performing the Cholesky factor symbolically may be too generous in allocating space for nonzeros in R. To see this, consider the structure of the left matrix

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \times & \\ & & & & & \times \end{bmatrix}, \qquad \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & & & & \\ & \times & & & \\ & & \times & & \\ & & & \times & \\ & & & & \times \end{bmatrix}. \tag{2.5.30}$$

For this matrix $R = A$, since A is already upper triangular. But, because the first row of A is full, A^TA will be full and the algorithm above will predict R to be full. In the Cholesky factorization of A^TA cancellation will occur 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. Structural cancellation cannot predicted from the nonzero structure of A^TA alone.

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 2.5.2 A matrix $A \in \mathbb{R}^{m\times n}$, $m \geq n$, is said to have the strong **Hall property** if for every subset of k columns, $0 < k < m$, the corresponding submatrix has nonzero elements in at least $k + 1$ rows. (Thus, when $m > n$, every subset of $k \leq n$ columns has the required property, and when $m = n$, every subset of $k < n$ columns has the property.)

For matrices with the strong Hall property 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 on the left in (2.5.30) does not have the strong Hall property since the first column has only one nonzero element. But the matrix on the right in (2.5.30), obtained by deleting the first column, does have 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

$$(P_{\mathrm{r}}A)^T(P_{\mathrm{r}}A) = A^T(P_{\mathrm{r}}^T P_{\mathrm{r}})A = A^TA,$$

it follows that the resulting factor R is independent of the row ordering. But the 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. 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 used for banded sparse matrices.

Algorithm 2.5.1 (*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 by increasing $f_i(A)$, so that $f_i(A) \leq f_k(A)$ 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 by increasing $l_i(A)$.*

An alternative row ordering, that has been found to work well is obtained by ordering the rows by increasing values of $l_i(A)$. With this ordering only the columns $f_i(A)$ to $l_i(A)$ of R_{i-1} will be involved when row a_i^T is being processed, since all previous rows only have nonzero elements 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 reflectors. 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. But as shown in Sect. 2.5.4, 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 plane rotations. Initialize R_0 to have 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 (see Fig. 2.7): we uncompress this row into a full vector and scan the nonzero elements from left to right. For each $a_{kj} \neq 0$, a plane rotation involving row j in R_{k-1} is used to annihilate a_{kj}. This may create new nonzero elements 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 [118, 1983]. Assume that the first k rows of A have been processed to generate $R^{(k)}$. In Fig. 2.7 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

Fig. 2.7 Row-sequential sparse Givens QR factorization; *circled* elements $\otimes$ in R_{k-1} are involved in the elimination of a_k^T; fill elements are denoted by $\oplus$

$$\begin{pmatrix} R_{k-1} \\ a_k^T \end{pmatrix} = \begin{bmatrix}
\times & 0 & \times & 0 & 0 & \times & 0 & 0 & 0 & 0 \\
 & \otimes & 0 & \oplus & \otimes & 0 & 0 & 0 & 0 & 0 \\
 & & \times & 0 & \times & 0 & 0 & 0 & \times & 0 \\
 & & & \otimes & \oplus & 0 & \otimes & 0 & 0 & 0 \\
 & & & & \otimes & \oplus & 0 & 0 & 0 & 0 \\
 & & & & & \times & 0 & 0 & \times & 0 \\
 & & & & & & \otimes & \otimes & 0 & 0 \\
 & & & & & & & \otimes & 0 & 0 \\
 & & & & & & & & \times & \times \\
 & & & & & & & & 0 & \times \\
0 & \otimes & 0 & \otimes & \oplus & 0 & \oplus & \oplus & 0 & 0
\end{bmatrix}$$

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.

Note that, unlike in the Householder method, intermediate fill now only takes place in the row 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 Fig. 2.7, but the plane rotations should be applied simultaneously to b to form Q^Tb. In the implementation by George and Heath [116, 1980], the plane 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 row sequential sparse QR algorithm employing plane rotations is due to George and Heath [116, 1980]. Liu [196, 1986] introduces the notion of *row merge tree* for sparse QR factorization by plane rotations. This row merging scheme can be viewed as implicitly using the multifrontal method on A^TA. George and Liu [117, 1987] give a modified version of Liu's algorithm using Householder reflectors instead of plane rotations. Recent work is surveyed by Davis [67, 2006].

The explicit orthogonal factor Q is often much less sparse than R. Therefore, Q is often discarded in sparse QR factorization. This creates a problem if additional right-hand sides have to be treated at a later stage, since we cannot form Q^Tb. Saving the plane rotations separately requires far less storage and fewer operations than computing and storing Q explicitly. The analysis of sparsity of the factor Q in sparse QR factorizations includes some subtle considerations; see Hare et al. [157, 1993] and Gilbert, Ng and Peyton [120, 1997].

If A is available, another method to deal with this problem is to use the **seminormal equations**

$$R^TRx = A^Tb, \tag{2.5.31}$$

with R from the QR factorization. The accuracy of the solution $\bar{x}$ computed from (2.5.31) is not much better than for the method of normal equations. Rounding errors committed when computing A^Tb will lead to an error δx bounded in magnitude by

$$\|\delta x\|_2 \le m\mathbf{u}\|(A^TA)^{-1}\|_2\|A\|_2\|b\|_2 \le m\mathbf{u}\,\kappa(A)^2\|b\|_2/\|A\|_2.$$

In the corrected seminormal equations (CSNE), a corrected solution $x_C = \bar{x} + \delta x$ is computed from

$$\bar{r} = b - A\bar{x}, \quad R^TR\delta x = A^T\bar{r}. \tag{2.5.32}$$

This is similar to one step of iterative refinement for the normal equations (see Algorithm 2.1.1), except that here R from the QR factorization is used. The error bound for the x_C from CSNE is usually no worse and often better than that for a

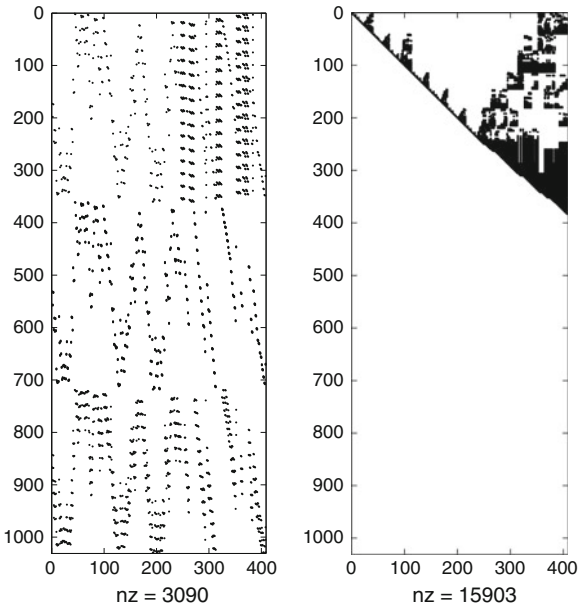


Fig. 2.8 Nonzero pattern of a sparse matrix A and the factor R in its QR factorization using MATLAB's colamd reordering

backward stable method; see [27, 1996], Sect. 6.6.5. No extra precision for computing $\bar{r}$ is needed for this to be true.

For solving the minimum-norm problem

$$\min \|y\|_2, \quad \text{subject to} \quad A^T y = c.$$

the algorithm using Q given in (2.3.41) is solve $R^T z_1 = c$, and set $y = Q_1 z_1$. If Q is not available, an approach suggested by Saunders [248, 1972] is to compute

$$R^T R w = c, \quad y = A w. \tag{2.5.33}$$

As proved by Paige [225, 1973], this algorithm is quite satisfactory without adding a correction step, and the bound on the error is proportional to κ.

Example 2.5.2 To illustrate the effect of different column orderings we use a model by Elfving and Skoglund [93, 2007] for substance transport in rivers. In this time series data

$$L_{ij}, \quad i = 1{:}n, \quad j = 1{:}m,$$

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 into two sets $x^T = (x_1, x_2)$ and a regularization matrix is added. Figures 2.9 and 2.8

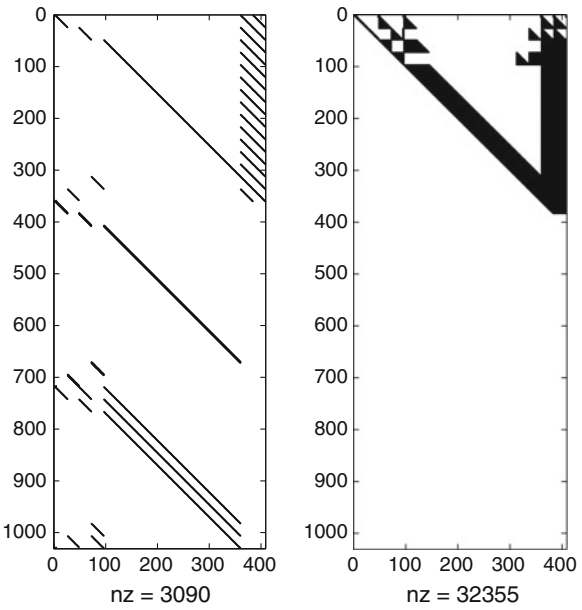


Fig. 2.9 Nonzero pattern of a sparse matrix A and the factor R in its QR factorization using MATLAB's colperm reordering

show 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. For this $\text{nnz}(R) = 32\,355$. The second (`colamd` is an approximate minimum degree ordering for A^TA. For this $\text{nnz}(R) = 15\,903$, a greatimprovement. □

In multifrontal methods the QR factorization is reorganized into a sequence of independent partial QR factorizations of small dense matrices. Since these subproblems can be can solved in parallel, this can lead to a significant reduction in factorization time at a modest increase in working storage. The good data locality of the multifrontal method gives fewer page faults on paging systems, and out -of-core versions can be developed.

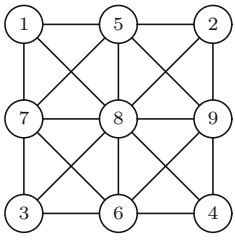


Fig. 2.10 The graph $G(A^TA)$ of a matrix arising from a 3×3 mesh problem and a nested dissection ordering

Example 2.5.3 We illustrate the multiple front idea on the QR factorization of a 12×9 matrix A taken from Liu [196], shown below before and after the first elimination stage in the QR factorization:

$$A = \left(\begin{array}{ccccccccc}
\times & & & \times & & \times & \times & & \\
\times & & & \times & & \times & \times & & \\
\times & & & \times & & \times & \times & & \\
\hline
 & \times & & \times & & & \times & \times & \\
 & \times & & \times & & & \times & \times & \\
 & \times & & \times & & & \times & \times & \\
\hline
 & & \times & & & \times & \times & \times & \\
 & & \times & & & \times & \times & \times & \\
 & & \times & & & \times & \times & \times & \\
\hline
 & & & \times & & \times & & \times & \times \\
 & & & \times & & \times & & \times & \times \\
 & & & \times & & \times & & \times & \times
\end{array}\right)$$

This matrix arises from a 3×3 mesh problem using a nested dissection ordering, and its graph $G(A^TA)$ is shown in Fig. 2.10.

First a QR factorization of rows 1–3 is performed. Since these rows have nonzero elements only in columns $\{1, 5, 7, 8\}$, this operation can be carried out as a QR factorization 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 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 step the matrix $A^{(2)}$ has the form shown below. 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.

$$A^{(2)} = \left(\begin{array}{ccccccccc}
\times & & & \times & & \times & \times & & \\
 & & & \times & & \times & \times & & \\
 & & & & & \times & \times & & \\
\hline
 & \times & & \times & & & \times & \times & \\
 & & & \times & & & \times & \times & \\
 & & & & & & \times & \times & \\
\hline
 & & \times & & & \times & \times & \times & \\
 & & & & & \times & \times & \times & \\
 & & & & & & \times & \times & \\
\hline
 & & & \times & & \times & & \times & \times \\
 & & & & & \times & & \times & \times \\
 & & & & & & & \times & \times
\end{array}\right).$$

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 by the index of the first nonzero element, and then perform a QR factorization:

$$Q^T \begin{pmatrix} \times & \times & \times & \\ \times & & \times & \times \\ & \times & \times & \\ & & \times & \times \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \end{pmatrix}.$$

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, working on columns 7, 8, and 9, we merge the remaining two upper trapezoidal (here triangular) matrices into a triangular matrix. □

The organization of the multifrontal method is based on the elimination tree. Nodes in the tree are visited in turn following a postordering, i.e., a topological ordering in which a parent node j always has node $j-1$ as one of its children. 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, which is stored in a stack until needed. An important advantage of using a postordering is that data management is simplified, since the update matrices can be managed in a stack on a last-in-first-out basis.

A formal outline of the multifrontal sparse QR algorithm goes as follows: 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.

In many implementations of multifrontal algorithms the orthogonal transformations are not stored, and the seminormal equations are used for treating additional right-hand sides. If Q is needed, it should not be stored explicitly, but 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 $\text{nnz}(R) = O(n \log n)$, but $\text{nnz}(Q) = O(n\sqrt{n})$. Lu and Barlow [200] show that the frontal Householder vectors only require $O(n \log n)$ storage.

Multifrontal algorithms for QR factorization were first developed by Liu [196, 1986] and George and Liu [117, 1987]. Supernodes and other modifications of the multifrontal method are discussed by Liu [197, 1990]. The latest sparse QR algorithm included in MATLAB 7.9 is the multithreaded multifrontal QR in SuiteSparse by Davis [68, 2011]; see also http://www.cise.ufl.edu/research/sparse/SuiteSparse.

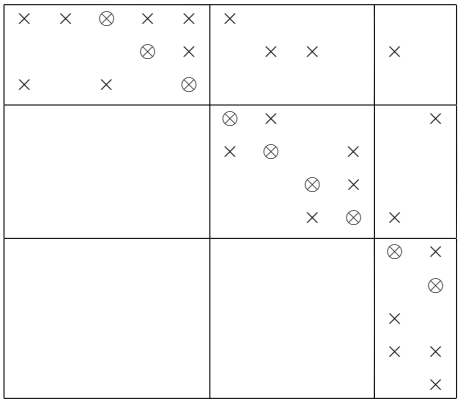


Fig. 2.11 The coarse block triangular decomposition of a rectangular matrix

2.5.6 Block Triangular Form

As shown in Sect. 1.7.6, it can be advantageous to permute a square matrix A into block triangular form (1.7.13) before solving the linear system $Ax = b$. An arbitrary rectangular matrix $A \in \mathbb{R}^{m\times n}$ can be permuted into a similar block triangular form, called the **Dulmage–Mendelsohn form**

$$PAQ = \begin{pmatrix} A_h & U_{hs} & U_{hv} \\ & A_s & U_{sv} \\ & & A_v \end{pmatrix}. \tag{2.5.34}$$

The first diagonal block A_h may have more columns than rows, the middle block A_s is square, and the last A_v may have more rows than columns. These blocks both have the strong Hall property. The middle diagonal block is square with nonzero diagonal entries. One or two of the blocks in (2.5.34) may be absent. The off-diagonal blocks are possibly nonzero matrices of appropriate dimensions. An example of the coarse block triangular decomposition of a rectangular matrix is given in Fig. 2.11.

It may be possible to further decompose some of the diagonal blocks in (2.5.34) to obtain a finer decomposition. Each of the blocks A_h and A_v may be further decomposable into block diagonal form, where the blocks $A_{h1}, \ldots, A_{hp}$ are underdetermined and the blocks $A_{v1}, \ldots, A_{vq}$ are overdetermined. The square submatrix A_s may be further decomposable into block upper triangular form. The resulting decomposition can be shown to be essentially unique. Any such 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.

The block triangular form is called the **Dulmage–Mendelsohn** form, because it is based on a canonical decomposition of a **bipartite graph** discovered by Dulmage and Mendelsohn. The bipartite graph of a rectangular matrix $A \in \mathbb{R}^{m\times n}$ is denoted

by $G(A) = \{R, C, E\}$. Here $R = (r_1, \ldots, r_m)$ is a set of vertices corresponding to the rows of A and $C = (c_1, \ldots, c_n)$ is a set of vertices corresponding to the columns of A. E is the set of edges, where $\{r_i, c_j\} \in E$ if and only if $a_{ij} \neq 0$. A bipartite matching in $G(A)$ is a subset of its edges with no common end points (Fig. 2.11).

The algorithm by Pothen and Fan [237, 1990] for computing the Dulmage–Mendelsohn decomposition 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 submatrices 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 from columns to rows.

The algorithm by Pothen and Fan is available in MATLAB through the function `[p,q,r,s,cc,rr] = dmperm(A)`. The result is row and column permutations vectors p and q, respectively, such that $A(p, q)$ has Dulmage–Mendelsohn block triangular form. The vectors r and s are index vectors indicating the block boundaries for the fine decomposition, while the vectors cc and rr indicates the boundaries of the coarse decomposition.

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 (2.5.34) 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

$$\min_{\widetilde{x}_v} \|A_v \widetilde{x}_v - \widetilde{b}_v\|_2, \tag{2.5.35}$$

where $\widetilde{x} = Q^T x$ and $\widetilde{b} = Pb$ have been partitioned conformally with PAQ in (2.5.34). The remaining part of the solution $\widetilde{x}_k, \ldots, \widetilde{x}_1$ is then determined by

$$A_{si}\widetilde{x}_i = \widetilde{b}_i - \sum_{j=i+1}^{k} U_{ij}\widetilde{x}_j, \quad i = k{:}G - 1{:}1. \tag{2.5.36}$$

Finally, we have $x = Q\widetilde{x}$. The subproblems in (2.5.35) and (2.5.36) can be solved 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.

The block triangular form 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 [78, 1963].

Exercises

2.5.1 A frequent task in multifrontal methods is 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 reflectors that do 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

$$\begin{bmatrix} * & * & * & * \\ & * & * & * \\ & & \times & \times \\ & & & \times \\ \otimes & \otimes & * & * \\ & \otimes & * & * \\ & & \times & \times \\ & & & \times \end{bmatrix}.$$

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.

2.5.2 Assume that the matrix $A \in \mathbb{R}^{m\times n}$ of bandwidth w is in standard form and stored in an $m \times w$ array, together with a vector p of pointers, where p_i points to the first row of A with $f_i(A) = i$, $i = 1 : n$. Write a MATLAB program that computes $R \in \mathbb{R}^{n\times n}$ in the QR factorization and stores it in an $n \times w$ array. The rows of A are to be merged into R one at a time using Givens rotations.

2.6 Regularization of Ill-Posed Linear Systems

A Fredholm[9] integral equation of the first kind has the form

$$\int_0^1 k(s,t)\,f(s)\,ds = g(t), \qquad -1 \le t \le 1. \tag{2.6.1}$$

When the kernel $k(s,t)$ is smooth, this equation is known to be **ill-posed** in the sense that the solution f does not depend continuously on the right-hand side g. This is related to the fact that there are rapidly oscillating functions $f(t)$ that come arbitrarily close to being annihilated by the integral operator.

In order to solve the Eq. (2.6.1) numerically it must first be discretized. Introducing a uniform mesh for s and t on $[-1, 1]$ with step size $h = 2/(n+1)$, $s_i = -1 + ih$, $t_j = -1 + jh$, $i, j = 0 : n+1$ and approximating the integral with the trapezoidal rule gives

$$h\sum_{i=0}^{n} w_i k(s_i, t_j) f(t_i) = g(t_j), \quad j = 0 : n+1, \tag{2.6.2}$$

[9] Erik Ivar Fredholm (1866–1927), a Swedish mathematician and a student of Mittag-Leffler, got his doctorate from the University of Uppsala in 1898. Much of his main contributions on integral equations were accomplished during a visit to Paris in 1899. Fredholms work was extended by Hilbert and led to the theory of Hilbert spaces.

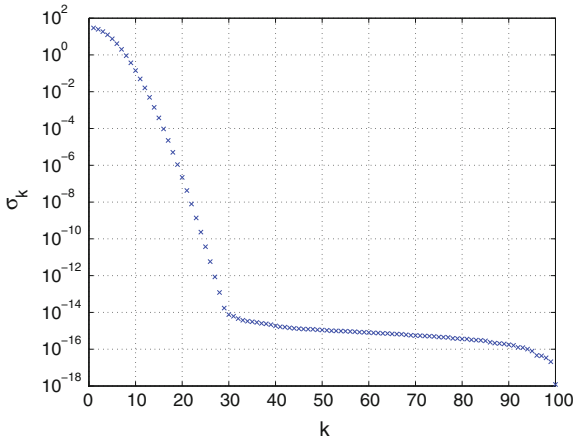


Fig. 2.12 Singular values of a matrix with ill-defined rank

where $w_i = 1, i \neq 0, n$ and $w_0 = w_n = 1/2$. These equations form a linear system $Kf = g$, $K \in \mathbb{R}^{n\times n}$, and $f, g \in \mathbb{R}^n$.

The discretized problem is not ill-conditioned in the original sense of Hadamard. However, the inherent difficulty in solving the continuous ill-posed problem carries over to the discrete problem. This becomes evident by looking at the singular values σ_i of K. For $k(s,t) = e^{-(s-t)^2}$, and $n = 100$, these were computed using IEEE double precision. The result is displayed in logarithmic scale in Fig. 2.12. For $i > 30$, σ_i are close to roundoff level and the *numerical* rank of K is certainly smaller than 30. This means that the linear system $Kf = g$ has a meaningful solution only for special right-hand sides g. Following Hansen [148, 1990] we call such problems **discrete ill-posed problems**.

If the right-hand side g is restricted to lie in a subspace spanned by the left singular vectors corresponding to a small set of the largest singular values, the linear system is *effectively well-conditioned*; see Varah [287, 1973]. This concept is made more precise by Chan and Foulser [45, 1988].

2.6.1 TSVD and Tikhonov Regularization

The solution to a discrete ill-posed linear system $Kf = g$ (or, more generally, least squares problem $\min_x \|Kf - g\|_2$), can be expressed in terms of the SVD of $K = \sum_{i=1}^{n} u_i \sigma_i v_i$ as

$$f = \sum_{i=1}^{n} \frac{c_i}{\sigma_i} v_i, \quad c_i = u_i^T g. \tag{2.6.3}$$

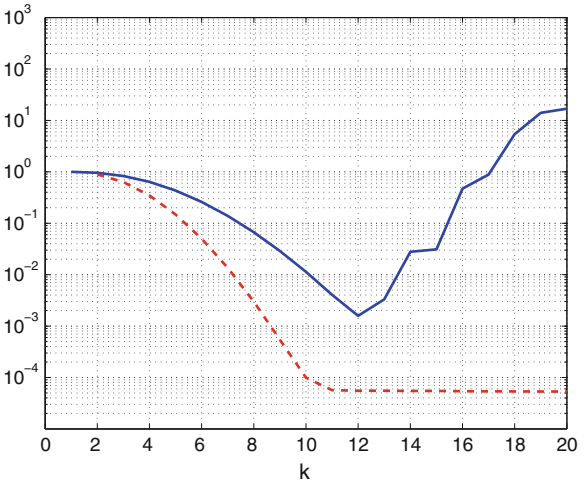


Fig. 2.13 Relative error $\|f_k - f\|_2/\|f\|_2$ (*solid line*) and relative residual $\|Kf_k - g\|_2/\|g\|_2$ (*dashed line*) for TSVD solutions truncated after k steps

In the continuous case, (2.6.1) can be written $Kf = g$, where K is a compact operator with singular value expansion $\sum_{i=1}^{\infty} u_i \sigma_i v_i$. For this to have a square integrable solution f, it is necessary and sufficient that the right-hand side g satisfies the **Picard condition** (Groetsch [141, 1984], Theorem 1.2.6).

$$\sum_{i=1}^{\infty} |u_i^T g/\sigma_i|^2 < \infty \tag{2.6.4}$$

A consequence of this is that for the *exact* right-hand side g the coefficients c_i in the SVD expansion must eventually decrease faster than σ_i. However, if g is contaminated by a random noise vector η, *all coefficients* c_i are affected more or less equally. This will cause the computed solution for a perturbed right-hand side $\widehat{g} = g + \eta$ to blow up.

To obtain a stable and accurate approximate solution to the discretized problem, the SVD expansion (2.6.3) can be truncated after a small number $k \ll n$ of terms. This restricts the solution to lie in a low-dimensional subspace spanned by the right singular vectors corresponding to the large singular values. Equivalently, we seek a **regularized solution** as a linear combination of the first k left singular vectors,

$$f_k = V_k z_k, \quad V_k = (v_1, \ldots, v_k),$$

giving $z_k = \Sigma_k^{-1}(U_k^T g)$. This is known as a **truncated SVD** (TSVD) solution. The value of the truncation index k is chosen so that a large reduction in the norm of the residual is achieved without causing the norm of the approximate solution to become

too large. In statistics this approach is known as **principal components regression** (PCR).

Example 2.6.1 For $n = 100$ and a given solution f, a perturbed right-hand side $g = Kf + \eta$ for (2.6.2) was computed with η normally distributed with zero mean and variance 10^{-4}. Figure 2.13 shows the relative error $\|f_k - f\|_2/\|f\|_2$ and the relative residual $\|Kf_k - g\|_2/\|g\|_2$ for the TSVD solutions as a function of k. The smallest error occurs for $k = 12$. For larger values of k the error increases very rapidly, although the residual norm is almost constant. In practice, the error is unknown and only the residual can be observed. □

In TSVD the solution is orthogonally projected onto a lower dimensional subspace spanned by $k < n$ of the right singular vectors. Another method for the regularization of discrete ill-posed problems is due to Tikhonov [280, 1963][10] and called **Tikhonov regularization**. In this approach the linear system $Ax = b$ is replacedby the minimization problem

$$\min_x \|Ax - b\|_2^2 + \mu^2 \|Lx\|_2^2. \tag{2.6.5}$$

Here μ is a regularization parameter, which governs the balance between a small residual norm $\|b - Ax(\mu)\|_2$ and a smooth solution as measured by $\|Lx(\mu)\|_2$. A similar technique was used already by Levenberg [193, 1944] and Marquardt [205, 1963] to stabilize solutions to nonlinear least squares problems. In statistics, Tikhonov regularization is known as **ridge regression** and used to stabilize regression estimates.

Typically L in (2.6.5) is taken to be a discrete approximation of some derivative operator. For example, except for a scaling factor,

$$L = \begin{pmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{pmatrix} \in \mathbb{R}^{(n-1)\times n} \tag{2.6.6}$$

approximates the first derivative operator.

It is easy to show that the solution to (2.6.5) is the solution to the generalized normal equations

$$(A^T A + \mu^2 L^T L)x = A^T b. \tag{2.6.7}$$

These equations have a unique solution if rank $\begin{pmatrix} A \\ L \end{pmatrix} = n$ or, equivalently, if the null spaces of A and L intersect only trivially, i.e., $\mathcal{N}(A) \cap \mathcal{N}(L) = \{0\}$. Forming the normal equations requires computing the cross-product matrices $A^T A$ and $L^T L$ and

[10] Andrei Nicholaevich Tikhonov (1906–1993), Russian mathematician, who made deep contributions in topology and functional analysis. In the 1960's he introduced the concept of "regularizing operator" for ill-posed problems, for which he was awarded the Lenin medal.

will square the condition number. This can be avoided by noticing that (2.6.7) are the normal equations for the least squares problem

$$\min_x \left\| \begin{pmatrix} A \\ \mu L \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2. \tag{2.6.8}$$

Hence, for any given value of $\mu > 0$, (2.6.7) can be solved by QR factorization of the augmented matrix in (2.6.8).

In the **standard case** of Tikhonov regularization $L = I_n$. Then the singular values of the augmented matrix in (2.6.8) are $\widetilde{\sigma}_i = \sqrt{\sigma_i^2 + \mu^2}$, $i = 1{:}n$, and the regularized solution becomes

$$x(\mu) = \sum_{i=1}^{n} \frac{c_i}{\widetilde{\sigma}_i} v_i = \sum_{i=1}^{n} f_i \frac{c_i}{\sigma_i} v_i, \quad f_i = \frac{\sigma_i}{\sqrt{\sigma_i^2 + \mu^2}}. \tag{2.6.9}$$

The quantities f_i are called **filter factors** or, in statistical applications, **shrinkage factors**. As long as $\mu \ll \sigma_i$, we have $f_i \approx 1$, and if $\mu \gg \sigma_i$, then $f_i \ll 1$. This establishes a relation to the TSVD solution, where the filter factors are step functions: $f_i = 1$ if $\sigma_i > \delta$ and $f_i = 0$ otherwise. In practice, the solutions obtained via Tikhonov regularization with $L = I_n$ and an appropriate value of μ is very close to the TSVD solution.

For a given value of μ, the solution $x(\mu)$ can in the standard case be computed using the Cholesky factorization of $A^T A + \mu^2 I$, or more reliably from the QR factorization

$$Q(\mu)^T \begin{pmatrix} A \\ \mu I \end{pmatrix} = \begin{pmatrix} R(\mu) \\ 0 \end{pmatrix}, \qquad \begin{pmatrix} c(\mu) \\ d(\mu) \end{pmatrix} = Q(\mu)^T \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{2.6.10}$$

Then $x(\mu)$ is obtained from the triangular systems

$$R(\mu) x(\mu) = c(\mu). \tag{2.6.11}$$

Problem LSQI with $L \neq I$ can be transformed to standard form. If L is square and nonsingular, this is achieved by the change of variables $Lx = y$, giving the problem

$$\min_y \left\| \begin{pmatrix} AL^{-1} \\ \mu I \end{pmatrix} y - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2. \tag{2.6.12}$$

The matrix $\widetilde{A} = AL^{-1}$ can be formed by solving the upper triangular matrix equation $L^T \widetilde{A}^T = A^T$. In practice it is often the case that $L \in \mathbb{R}^{(n-t)\times n}$ with full row rank. Let the QR factorization of L^T be

$$L^T = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} R_L \\ 0 \end{pmatrix} \in \mathbb{R}^{n\times(n-t)}.$$

Then $L = R_L^T V_1$, where R_L nonsingular V_1 and V_2 span $\mathcal{R}(L^T)$ and $\mathcal{N}(L)$, respectively. For example, if L as in (2.6.6) approximates the first derivative operator, then the dimension of $\mathcal{N}(L)$ is $t = 1$. The transformation to standard form can be achieved using the pseudoinverse $L^\dagger = V_1 R_L^{-T}$. The solution x is split into two orthogonal components $x = L^\dagger y + V_2 w$, so that

$$Ax = AV_1 R_L^{-T} y + AV_2 w = \widetilde{A} y + AV_2 w. \tag{2.6.13}$$

For details we refer to Björck [24, 1988]. The general case with no restrictions on L has been treated by Eldén [86, 1982].

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 criterion is known as the **discrepancy principle** and is due to Morozov [212, 1984]. In Example 2.6.1 the noise was normally distributed with variance 10^{-4}. In Fig. 2.13 the relative residual norm touches this value for $k = 10$, which is close to $k = 12$ for which value the minimum error norms occurs. However, it gives a slightly oversmoothed solution, which means that all the information present in the data has not been recovered. This behavior is typical for the discrepancy principle.

When no a priori information about the noise level is available, the determination of a suitable value of μ is a major difficulty. In the **generalized cross-validation** (GCV) of Golub et al. [135, 1979], the basic idea is as follows: Let $x_{\mu,i}$ be the solution of the regularized problem when the ith equation is left out. If this solution is a good approximation, then the error in the prediction of the ith component of the right-hand side should be small. This is repeated for all equations $i = 1{:}m$. Assume that the regularized solution is a linear function of the right-hand side $x(\mu) = A^\dagger(\mu) b$. Then the GCV function is

$$\mathcal{G}(\mu) = \frac{n^{-1}\|b - Ax(\mu)\|_2^2}{(n^{-1}\mathrm{trace}(I - AA^\dagger(\mu)))^2}. \tag{2.6.14}$$

Note that the GCV function is invariant under orthogonal transformations. For the standard Tikhonov regularization $I - AA(\mu)^\dagger = I - A(A^TA + \mu^2 I)^{-1}A^T$, and using the SVD $A = U\Sigma V^T$ we get

$$\frac{1}{n}\mathrm{trace}(I - AA(\mu)^\dagger) = \frac{1}{n}\sum_{i=1}^{n} \frac{\mu^2}{\sigma_i^2 + \mu^2}.$$

For some problems the GCV function can have a very flat minimum and hence be difficult to localize numerically; see Varah [288, 1983]. Another popular method, when the norm of the error is not explicitly known, is based on the **L-curve**

$$\mathcal{L} = \left\{\left(\log\|b - Ax(\mu)\|, \log\|x(\mu)\|\right)\right\} \tag{2.6.15}$$

For a large class of problems this curve is shaped as the letter L. Such a plot is used by Lawson and Hansen [190, 1974], Chap. 26, to analyze an ill-conditioned least squares problem. Hansen and O'Leary [152, 1993] propose to choose μ as the vertex point on the L-curve, i.e., the point where the curvature has the largest magnitude. For advantages and shortcomings of this method, see Hansen [149, 1992] and [150, 1998]. For large problems it may be too expensive to compute sufficiently many points on the L-curve. Calvetti et al. [40, 2002] show how to compute cheap upper and lower bounds in this case.

Regularization methods for linear and nonlinear ill-posed problems are admirably surveyed by Engl et al. [95, 1996]. A MATLAB regularization toolbox for analysis and solution of discrete ill-posed problems has been developed by P. C. Hansen. The latest version 4.0 for MATLAB 7.3 is described in [151, 2007]. The toolbox can be downloaded from Netlib at http://ftp.irisa.fr/pub/netlib/numeralgo/.

2.6.2 Least Squares with Quadratic Constraints

Closely related to Tikhonov regularization is the least squares problem subject to a quadratic inequality constraint:

Problem LSQI: Given $A \in \mathbb{R}^{m\times n}$, $L \in \mathbb{R}^{p\times n}$, and $\gamma > 0$, solve

$$\min_x \|Ax - b\|_2^2 \quad \text{subject to} \quad \|Lx - d\|_2^2 \le \gamma^2. \tag{2.6.16}$$

Conditions for existence and uniqueness and properties of solutions to problem LSQI are given by Gander [108, 1981]. Let an L-generalized solution $x_{A,L}$ of $\min_x \|Ax - b\|_2$ be defined as the solution to

$$\min_{x\in S} \|Lx - d\|_2, \qquad S = \{x \in \mathbb{R}^n \mid \|Ax - b\|_2 = \min\}. \tag{2.6.17}$$

Then either $x_{A,L}$ solves problem LSQI, or $\|Lx_{A,L} - d\|_2^2 > \gamma^2$ and the constraint is binding and the solution occurs on the boundary of the constraint region.

Theorem 2.6.1 *Assume that the solution x of problem* LSQI *occurs on the boundary of the constraint region. Let $x(\mu)$ be the solution to the generalized normal equations*

$$(A^TA + \mu^2 L^TL)x = A^Tb + \mu^2 L^Td. \tag{2.6.18}$$

Then $x = x(\mu)$, where the parameter $\mu > 0$ is determined by the so called **secular equation**[11]

$$\|Lx(\mu) - d\|_2^2 = \gamma^2. \tag{2.6.19}$$

[11] This terms comes from celestial mechanics, where a similar equation appears in the computation of secular, i.e., long-term perturbations of orbiting bodies.

Proof Using the method of Lagrange multipliers we consider the function

$$\psi(x,\mu) = \|Ax - b\|_2^2 + \mu^2\left(\|Lx - d\|_2^2 - \gamma^2\right). \tag{2.6.20}$$

where μ^2 is a Lagrange multiplier. Setting the gradient of $\psi(x,\mu)$ with respect to x equal to zero gives (2.6.18) and μ is obtained by solving the secular equation. □

Note that (2.6.18) are the normal equations for

$$\min_x \left\| \begin{pmatrix} A \\ \mu L \end{pmatrix} x - \begin{pmatrix} b \\ \mu d \end{pmatrix} \right\|_2, \quad \mu > 0. \tag{2.6.21}$$

The solution to problem LSQI will be unique if the constraint in (2.6.16) is binding and rank $\begin{pmatrix} A \\ L \end{pmatrix} = n$.

In the standard case $L = I$ and $d = 0$, the secular equation can be written as

$$f(\mu) = \|x(\mu)\|_2 - \gamma = 0, \quad \gamma > 0, \tag{2.6.22}$$

where $x(\mu) = (A^TA + \mu^2 I)^{-1}A^Tb$ solves the least squares problem (2.6.8). Newton's method, which approximates $f(\mu)$ with a linear function, is not suitable for solving (2.6.22), because $f(\mu)$ can have a singularity for $\mu = 0$. A rational approximation can be used, but a similar effect is achieved by applying Newton's method to the equation

$$g(\mu) = \frac{1}{\|x(\mu)\|_2} - \frac{1}{\gamma} = 0. \tag{2.6.23}$$

Taking the derivative with respect to μ of $\|x(\mu)\|_2^{-1} = (x(\mu)^Tx(\mu))^{-1/2}$ gives

$$\frac{dg(\mu)}{d\mu} = -\frac{x^T(\mu)}{\|x(\mu)\|_2^3}\frac{dx(\mu)}{d\mu}.$$

From the formula for the derivative of an inverse matrix we obtain

$$x(\mu)^T\frac{dx(\mu)}{d\mu} = -x(\mu)^T(A^TA + \mu^2 I)^{-1}x(\mu) \equiv -\|z(\mu)\|_2^2. \tag{2.6.24}$$

This gives the iteration method due to Reinsch [242, 1971][12]:

$$\mu_{k+1} = \mu_k + \left(\frac{\|x(\mu_k)\|_2}{\gamma} - 1\right)\frac{\|x(\mu_k)\|_2^2}{\|z(\mu_k)\|_2^2}. \tag{2.6.25}$$

[12] In optimization literature this method is usually credited to Hebden [158, 1973].

The asymptotic rate of convergence for this method is quadratic. Furthermore, if the initial approximation satisfies $0 < \mu < \mu^*$, where μ^* is the solution, then *the iterates μ_k converge monotonically from below*.

Algorithm 2.6.1 (*Reinsch's Algorithm*)

```
function [x,nx] = reinsch(A,b,gamma)
% REINSCH performs <= p iterations to solve
%   min_x||A x - b||_2 subject to ||x||_2 = gamma
% ------------------------------------------------
[m,n] = size(A);
mu = m*eps*norm(A,1);
for k = 1:p
% Compute thin QR.
   [Q,R] = qr([A; mu*eye(n)], 0);
   c = Q'*b;
   x = R\c; nx = norm(x);
   if nx <= gamma, break end
% Perform Newton step.
   z = R'\x; nz = norm(z);
   dmu = (nx/gamma - 1)*(nx/nz)^2;
   mu = mu + dmu;
end
```

The main cost in this method is for computing the QR factorizations for solving (2.6.21) in each iteration step. Then $x(\mu)$ and $z(\mu)$ are obtained from the triangular systems

$$R(\mu)x(\mu) = c(\mu), \qquad R(\mu)^T z(\mu) = x(\mu). \tag{2.6.26}$$

Hence, computing the derivative (2.6.26) costs only one more triangular solve.

Example 2.6.2 When computing the Householder QR factorization one can take advantage of the special structure of the matrix. The shape of the transformed matrix after $k = 2$ steps for $m = n = 5$ is shown below

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & + & + & + \\ & 0 & + & + & + \\ & & \times & & \\ & & & \times & \\ & & & & \times \end{bmatrix}.$$

Note that only the first two rows of D have filled in, and the remaining rows of μI are still untouched. In each step of the QR factorization there are m elements in the current column to annihilate, and the operation count is $2mn^2$ flops. If $A = R$ already is in upper triangular form the flop count for the QR factorization is reduced to $2n^3$. (Show this!) Hence, unless A is sparse, it is more efficient to start by computing the QR factorizations of A at a cost of $2(mn^2 - n^3/3)$ flops. In practice ≈ 6 iterations usually suffice to achieve full accuracy. Further savings are possible by initially transforming A to bidiagonal form; see Eldén [87, 1984] and Problem 2.6.2.

2.6.3 Bidiagonalization and Partial Least Squares

The **partial least squares** (PLS) method is due to Wold [297, 1966] and originated in statistical applications, specifically economics. It is also an alternative technique for regularization of linear least squares problems. The PLS method generates a sequence of approximations, which consists of orthogonal projections of the pseudoinverse solution $A^\dagger b$ onto low dimensional Krylov subspaces.

Definition 2.6.1 The PLS approximation x_k, $k = 1, 2, \ldots$ to the pseudoinverse solution of $\min_x \|Ax - b\|_2$ are the solutions to the subproblem

$$\min_{x_k} \|Ax_k - b\|_2, \quad \text{subject to } x_k \in \mathcal{K}_k(A^TA, A^Tb), \quad k = 1, 2, \ldots, \tag{2.6.27}$$

where $\mathcal{K}_k(A^TA, A^Tb)$ is the Krylov subspace

$$\text{span}\{A^Tb, (A^TA)A^Tb, \ldots, (A^TA)^{k-1}A^Tb\}. \tag{2.6.28}$$

Since the Krylov subspaces are nested, $\mathcal{K}_k(A^TA, A^Tb) \subseteq \mathcal{K}_{k+1}(A^TA, A^Tb)$, the sequence of residual norms $\|r_k\|_2 = \|b - Ax_k\|$ of the PLS approximations is nonincreasing. The PLS method terminates for $k = p$, where p is the grade of A^Tb with respect to A^TA. Then $x_k = x_p$, for $k > p$, is the pseudoinverse solution $x^\dagger$. Using the SVD of A, the Krylov vector $(A^TA)^{k-1}A^Tb$ can be written

$$y_k = (A^TA)^{k-1}A^Tb = \sum_{i=1}^{p} c_i \sigma_i^{2k-1} v_i, \quad c_i = u_i^Tb, \quad k \geq 1. \tag{2.6.29}$$

Let $\sigma_1 > \sigma_2 > \cdots > \sigma_n$ be the singular values and u_i, v_i the left and right singular vectors. If σ_i is a simple singular value, then u_i and v_i are uniquely determined and we set $c_i = u_i^Tb$. For a multiple singular value c_i is the norm of the projection of b onto the left singular subspace corresponding to σ_i. In this case the left and right singular vectors can be chosen as an arbitrary basis for the singular subspaces. It is therefore no restriction to assume that the right-hand side b has a nonzero projection onto only *one* particular singular vector u_i in the invariant subspace. Denote the unique corresponding right singular vector by v_i. Deleting the terms in (2.6.29) for

which $c_i = 0$ and renumbering the singular values accordingly, it follows that the Krylov vectors $y_k, k \geq 1$, are linear combinations of $v_1, \ldots, v_s$. Therefore, the grade of A^TA with respect to A^TA is at most equal to s.

We now show a relation between the Krylov vectors (2.6.28) and the subset of the right singular vectors v_i chosen as described above, that is fundamental for the understanding of the PLS method.

Theorem 2.6.2 *Let $\sigma_1 > \sigma_2 > \cdots > \sigma_p$ be the distinct nonzero singular values of A. Let c_i be the norm of the orthogonal projection of b onto the corresponding left singular subspaces. Then the grade of A^Tb with respect to A^TA equals the number $s \leq p$ of nonzero coefficients c_i.*

Proof Setting $z_i = c_i\sigma_i v_i$, and using (2.6.29), we have $(y_1, y_2, \ldots, y_s) = (z_1, z_2, \ldots, z_s)W$, where

$$W = \begin{pmatrix} 1 & \sigma_1^2 & \cdots & \sigma_1^{2(s-1)} \\ 1 & \sigma_2^2 & \cdots & \sigma_2^{2(s-1)} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & \sigma_s^2 & \cdots & \sigma_s^{2(s-1)} \end{pmatrix} \in \mathbb{R}^{s\times s}. \tag{2.6.30}$$

Since $\sigma_i \neq \sigma_j, i \neq j$, the Vandermonde matrix W is nonsingular. It follows that the Krylov vectors $(y_1, y_2, \ldots, y_s)$ are linearly independent and the grade is s. □

Setting $k = 0$ in (2.6.29) gives the pseudoinverse solution

$$x^\dagger = \sum_{i=1}^{s} c_i\sigma_i^{-1} v_i \in \mathcal{K}_s(A^TA, A^Tb). \tag{2.6.31}$$

It follows that the PLS method always terminates with the pseudoinverse solution.

The PLS approximations can be computed using the GKH algorithm for upper bidiagonal reduction of A, with P_0 chosen so that

$$P_0(A^Tb) = \theta_1 e_1 \in \mathbb{R}^n, \tag{2.6.32}$$

or equivalently $\theta_1 P_0 e_1 = A^Tb$. We assume that $A^Tb \neq 0$, since otherwise $x^\dagger = 0$. After k steps of the bidiagonalization algorithm we have

$$Q_k \cdots Q_2 Q_1 A V_k = \begin{pmatrix} B_k \\ 0 \end{pmatrix}, \quad V_k = P_0 P_1 \cdots P_{k-1} \begin{pmatrix} I_k \\ 0 \end{pmatrix}, \tag{2.6.33}$$

where

$$B_k = \begin{pmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \ddots & \ddots & \\ & & & \rho_{k-1} & \theta_k \\ & & & & \rho_k \end{pmatrix} \in \mathbb{R}^{k\times k} \tag{2.6.34}$$

is upper bidiagonal. After P_k is applied, the first k rows of A are in upper bidiagonal form and

$$U_k^T A\, P_0 P_1 \cdots P_k = \begin{pmatrix} \widehat{B}_k & 0 \end{pmatrix}, \quad U_k = Q_1 Q_2 \cdots Q_k \begin{pmatrix} I_k \\ 0 \end{pmatrix}, \tag{2.6.35}$$

where $\widehat{B}_k = \begin{pmatrix} B_k & \theta_{k+1} e_k \end{pmatrix}$. From (2.6.33) and (2.6.35), we get the two fundamental relations

$$AV_k = U_k B_k, \tag{2.6.36}$$
$$A^T U_k = V_{k+1} \widehat{B}_k^T = V_k B_k^T + \theta_{k+1} v_{k+1} e_k^T. \tag{2.6.37}$$

Lemma 2.6.1 *Assume that the matrix B_k in (2.6.34) has nonzero bidiagonal elements. Then all its singular values are simple.*

Proof The singular values of B_k are the positive square roots of the eigenvalues of the symmetric tridiagonal matrix $T_k = B_k^T B_k$. The matrix T_k is unreduced if and only if B_k has nonzero bidiagonal elements. The lemma now follows from the result of Lemma 3.5.1 that an unreduced symmetric tridiagonal matrix has simple eigenvalues. □

From the choice of P_0 it follows that $\theta_1 v_1 = A^T b$. Equating the jth columns in Eqs. (2.6.36) and (2.6.37) yields the Lanczos-type recurrence relations $\rho_1 u_1 = Av_1$, and

$$A^T u_j = \rho_j v_j + \theta_{j+1} v_{j+1} \quad j = 1, 2, \ldots, \tag{2.6.38}$$
$$Av_{j+1} = \theta_j u_j + \rho_{j+1} u_{j+1}, \quad j = 1, 2, \ldots. \tag{2.6.39}$$

These equations yield the recurrence relations

$$\theta_{j+1} v_{j+1} = A^T u_j - \rho_j v_j, \tag{2.6.40}$$
$$\rho_{j+1} u_{j+1} = Av_{j+1} - \theta_{j+1} u_j. \tag{2.6.41}$$

for computing the vectors v_{j+1}, u_{j+1}. The elements θ_{j+1} and ρ_{j+1} in B_n are obtained as normalization conditions $\|u_{j+1}\|_2 = \|v_{j+1}\|_2 = 1$. The resulting algorithm is also due to Golub and Kahan [127, 1965]. Its numerical properties are further studied in Sect. 4.5.4.

Theorem 2.6.3 *As long as no bidiagonal element in B_k is zero, the matrices $U_k = (u_1, \ldots, u_k)$ and $V_k = (v_1, \ldots, v_k)$ are unique orthonormal bases for the*

two sequences of Krylov subspaces

$$\mathcal{R}(V_k) = \mathcal{K}_k(A^TA, A^Tb), \qquad \mathcal{R}(U_k) = \mathcal{K}_k(AA^T, AA^Tb), \tag{2.6.42}$$

generated by the symmetric matrices A^TA and AA^T.

Proof The columns of the matrices U_k and V_k are orthonormal by construction. Since $\theta_1 v_1 = A^Tb$ and $\rho_1 u_1 = Av_1 = AA^Tb/\theta_1$, the theorem is true for $k = 1$. The proof proceeds by induction in k. From (2.6.38)–(2.6.39) it follows that

$$\theta_{k+1}v_{k+1} \in A^T\mathcal{K}_k(AA^T, AA^Tb) \subset \mathcal{K}_{k+1}(A^TA, A^Tb).$$
$$\rho_{k+1}u_{k+1} \in A\mathcal{K}_{k+1}(A^TA, A^Tb) = \mathcal{K}_{k+1}(AA^T, AA^Tb).$$

The bases can also be obtained by applying the Gram–Schmidt orthogonalization to the respective sequence of Krylov vectors. The uniqueness of the bases is a consequence of the uniqueness (up to a diagonal scaling with elements ± 1) of the QR factorization of a real matrix of full rank. □

From Theorem 2.6.3 it follows that any vector $x_k \in \mathcal{K}_k(A^TA, A^Tb)$ can be written as

$$x_k = P_1 \cdots P_k \begin{pmatrix} y_k \\ 0 \end{pmatrix} = V_k y_k, \tag{2.6.43}$$

and the PLS approximation is obtained by solving $\min_{y_k} \|AV_k y_k - b\|_2^2$. From the orthogonal invariance of the ℓ_2-norm we obtain

$$\|AV_k y_k - b\|_2^2 = \|Q_k \cdots Q_1 (AV_k y_k - b)\|_2^2 = \|B_k y_k - c_k\|_2^2 + \|d_k\|_2^2,$$

where

$$\begin{pmatrix} c_k \\ d_k \end{pmatrix} = Q_k \cdots Q_1 b = Q_k \begin{pmatrix} c_{k-1} \\ d_{k-1} \end{pmatrix} \tag{2.6.44}$$

and the first $k-1$ elements in c_k are c_{k-1}. Hence, the minimizing y_k is the solution to the upper bidiagonal system $B_k y_k = c_k \in \mathbb{R}^k$, and the residual norm is $\|d_k\|_2$. Since the matrices U_k and V_k are never explicitly formed, they are orthogonal by construction. In step k the arithmetic cost for applying the transformations to the active part of A is $8(m-k)(n-k)$ flops. The flop counts for the additional scalar products and final back substitution are negligible in comparison.

The algorithm will terminate with $\theta_{k+1} = 0$ for some $k \le \operatorname{rank}(A)$ when the subspaces $\mathcal{K}_k(A^TA, A^Tb)$ have reached maximum rank. Then the subspaces $\mathcal{K}_k(AA^T, AA^Tb) = A\mathcal{K}_k(A^TA, A^Tb)$ have also reached maximal rank and $\rho_{k+1} = 0$. Hence,

$$x_k = A^\dagger b = P_1 P_2 \cdots P_k \begin{pmatrix} y_k \\ 0 \end{pmatrix} = V_k y_k.$$

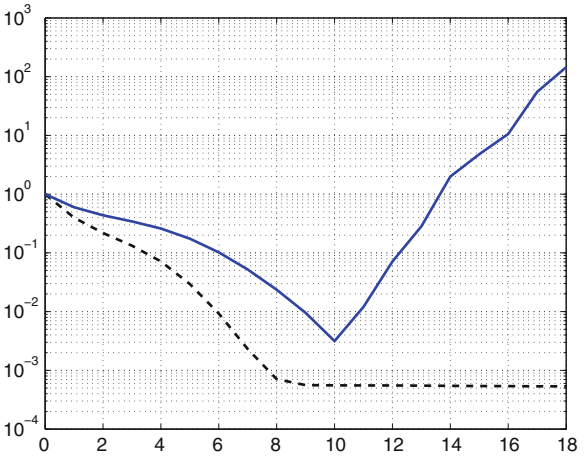


Fig. 2.14 Relative error norm $\|x_k - x\|_2/\|x\|_2$ (*solid line*) and residual norm $\|Ax_k - b\|_2/\|b\|_2$ (*dashed line*) after k steps of PLS

is the pseudoinverse solution.

Example 2.6.3 Like the TSVD method, PLS can be used as a regularization method. Both methods work by orthogonally projecting the least squares solution onto a sequence of subspaces of dimension $k \ll n$. They differ in the way the subspaces are chosen. In TSVD the subspaces are spanned by the first k right singular vectors. For PLS the truncated subspaces depend (nonlinearly) on the right-hand side b.

In Example 2.6.1, the TSVD method was used to compute a sequence of approximate solutions to the ill-posed linear system (2.6.2). In Fig. 2.13 the relative errors $\|x_k - x\|_2/\|x\|_2$ and $\|Ax_k - b\|_2/\|b\|_2$ are shown as functions of the number k of terms used in the SVD expansion. Similar results for GKH are shown in Fig. 2.14, where k now is the number of steps in GKH. The results are almost identical for PLS and TSVD although PLS only requires a partial bidiagonalization of A and generation of V_k. □

2.6.4 The NIPALS Algorithm

The NIPALS (Nonlinear Iterative PArtial Least Squares) PLS algorithm, due to Wold et al. [299, 1984], is frequently used in statistical computing, in particular in chemometrics. It uses a sequence of elementary orthogonal projections and generates the orthogonal basis vectors explicitly.

Set $A_0 = A$ and $b_0 = b$ and for $i = 1{:}n$,

(a) Generate unit vectors u_i, v_i by

$$\widehat{v}_i = A^T{}_{(i-1)}b_{i-1}, \qquad \widehat{u}_i = A_{i-1}v_i. \tag{2.6.45}$$

If $\|\widehat{v}_i\|_2 = 0$ or $\|\widehat{u}_i\|_2 = 0$, terminate the process. Otherwise, normalize these vectors

$$v_i = \widehat{v}_i/\|\widehat{v}_i\|_2, \qquad u_i = \widehat{u}_i/\|\widehat{u}_i\|_2. \tag{2.6.46}$$

(b) Deflate the data matrix A_{i-1} and right-hand side b_{i-1} by subtracting the orthogonal projections onto u_i:

$$(A_i,\ b_i) = (I - u_i u_i^T)(A_{i-1},\ b_{i-1}) = (A_{i-1},\ b_{i-1}) - u_i(p_i^T,\ \zeta_i), \tag{2.6.47}$$

$$p_i = A_{i-1}^T u_i, \quad \zeta_i = u_i^T b_{i-1}. \tag{2.6.48}$$

Summing the equations in (2.6.47) for $i = 1{:}k$ gives

$$A = U_k P_k^T + A_k, \qquad b = U_k z_k^T + b_k, \tag{2.6.49}$$

where $U_k = (u_1, \ldots, u_k)$, $P_k = (p_1, \ldots, p_k)$, $z_k = (\zeta_1, \ldots, \zeta_k)^T$, and $U_k P_k^T = \sum_{i=1}^k u_i p_i^T$ is a rank k approximation to the data matrix A.

Lemma 2.6.2 *The vectors $\{v_1, \ldots, v_k\}$ and $\{u_1, \ldots, u_k\}$ generated in exact arithmetic by* PLS *are orthonormal.*

Proof Assume that $\{u_1, \ldots, u_i\}$ and $\{v_1, \ldots, v_i\}$ are orthogonal. Since $u_1^T u_1 = 1$, using exact arithmetic in (2.6.47) gives

$$u_1^T u_2 = c_1 u_1^T A_1 v_2 = c_1 u_1^T (I - u_1 u_1^T) A v_2 = 0,$$
$$v_1^T v_2 = c_3 v_1^T A_1^T b_1 = c_3 v_1^T A^T (I - u_1 u_1^T) b = c_4 u_1^T (I - u_1 u_1^T) b = 0.$$

(Here and in the following $c_1, c_2, \ldots$ are generic constants.) This shows that the assumptions hold for $i = 2$. Using (2.6.47)) and the induction assumptions again we obtain

$$u_j^T u_{i+1} = c_2 u_j^T A_i v_{i+1} = c_2 u_j^T (I - u_i u_i^T) \cdots (I - u_j u_j^T) A_{j-1} v_{i+1} = 0,$$

$0 \le j \le i$. Similarly

$$\begin{aligned} v_j^T v_{i+1} &= c_5 v_j^T A_i^T b_i = c_5 v_j^T A_{j-1}^T (I - u_j u_j^T) \cdots (I - u_i u_i^T) b_i \\ &= c_6 u_j^T (I - u_j u_j^T) \cdots (I - u_i u_i^T) b_i = 0. \end{aligned}$$

□

The equivalence of the MGS and the Householder algorithms for PLS follows from the following result.

Theorem 2.6.4 *The vectors $\{v_1, \ldots, v_k\}$ and $\{u_1, \ldots, u_k\}$ generated by* PLS *form orthonormal bases for $\mathcal{K}_k(A^T A, A^T b)$ and $\mathcal{K}_k(AA^T, AA^T b)$, respectively.*

In exact arithmetic these vectors are the same as those computed by the upper bidiagonal GKH *algorithm with $v_1 = A^T b/\|A^T b\|_2$. The same holds for the sequence of approximate solutions $x_1, \ldots, x_k$.*

Proof Assume now that $v_i \in \mathcal{K}_i(A^TA, A^Tb)$, $u_i \in \mathcal{K}_i(AA^T, AA^Tb)$, and $p_i \in \mathcal{K}_i(A^TA, (A^TA)A^Tb)$. Clearly, this is true for $i = 1$. Now,

$$v_{i+1} = c_7 A_i^T b_i = c_7(A_{i-1}^T b_i - (u_i^T b)p_i) = c_8 v_i - c_9 p_i.$$

From the induction assumptions, it follows that $v_{i+1} \in \mathcal{K}_{i+1}(A^TA, A^Tb)$. Similarly,

$$u_{i+1} = c_{10} A_i v_{i+1} = c_{10}\Big(A - \sum_{j=1}^{i} u_j p_j^T\Big)v_{j+1} = c_{10}\Big(Av_{i+1} - \sum_{j=1}^{i}(p_j^T v_{j+1})u_j\Big),$$

and from the induction assumptions, we find that $u_{i+1} \in \mathcal{K}_{i+1}(AA^T, AA^Tb)$. □

The two algorithms generate orthonormal bases for the same sequences of Krylov subspaces. Hence, the equivalence of the two algorithms follows from the uniqueness of such bases. The second statement follows from the uniqueness of the solution to the full rank least squares subproblems. □

It is important to remember that, as in Gram–Schmidt orthogonalization, there will be a gradual loss of orthogonality in the u and v vectors, if floating point arithmetic is used. Therefore, the implememtation of the NIPALS algorithm is more delicate than for the PLS algorithm using Householder transformations, where the basis vectors are orthogonal by construction. However, relations (2.6.49) do not rely on orthogonality and will hold to working precision.

The kth approximation of the solution is of the form $x_k = V_k y_k$, where $V_k = (v_1, \ldots, v_k)$. By (2.6.49), the residual can be written as

$$b - AV_k y_k = r_1 + r_2, \quad r_1 = U_k(z_k - P_k^T V_k y_k), \quad r_2 = b_k - A_k V_k y_k.$$

The first term r_1 lies in $\mathcal{R}(U_k)$ and vanishes if y_k satisfies the linear system $(P_k^T V_k)y_k = z_k$. In exact computation $B_k = P_k^T V_k$ is upper bidiagonal and, by uniqueness, is the matrix in (2.6.34). Hence, the solution y_k can be computed by back substitution as for GKH. Assuming orthogonality of U_k, it follows that

Table 2.2 Condition number of P_k and loss of orthogonality in MGS-PLS: $\gamma(V_k) = \|I_k - V_k^T V_k\|_2$ and $\gamma(U_k) = \|I_k - U_k^T U_k\|_2$ *left*: with deflation of b; *right*: without deflation

k	$\kappa(P_k^T V_k)$	$\gamma(U_k)$	$\gamma(V_k)$	$\gamma(U_k)$	$\gamma(V_k)$
1	1.000+00	6.661-16	2.222-16	6.661-16	0
2	1.000+01	1.256-15	2.222-16	1.254-15	7.200-14
3	1.000+02	1.258-15	5.562-15	1.255-15	7.187-12
4	1.000+03	2.746-14	4.576-14	2.772-14	7.186-10
5	1.000+04	2.746-14	2.871-13	2.772-14	7.186-08
6	1.000+05	1.667-13	1.024-12	1.674-13	7.186-06
7	1.000+06	1.775-13	8.975-12	5.172-13	7.186-04
8	1.000+07	6.000-11	6.541-11	5.158-10	7.167-02

$$b_k - A_k V_k y_k = b_k - (I - U_k U_k^T) A V_k y_k = b_k.$$

Example 2.6.4 To study the loss of orthogonality in V_k and U_k, we take A to be an ill-conditioned matrix with singular values $\sigma_i = 10^{-i+1}$, $i = 1{:}8$, and set

$$A = UDV^T \in \mathbb{R}^{50\times 8}, \quad D = 1, 0.1, \ldots, 10^{-7},$$

where U and V are random orthogonal matrices.[13] The right-hand side is chosen as $b = Ae$, where $e = (1, \ldots, 1)^T$.

The NIPALS algorithm uses three matrix-vector products and one rank-one deflation, which together require $8mn$ flops per PLS factor. The flop counts for the additional scalar products and final back substitution are negligible in comparison. This is the same number of flops per step as required by the GKH algorithm as long as the number of factors $k \ll \min(m, n)$.

The numerical results were obtained using Algorithm 2.6.2. Table 2.2 shows the condition number $\kappa(P_k)$ and the loss of orthogonality in U_k and V_k measured by $\|I_k - U_k^T U_k\|_2$ and $\|I_k - V_k^T V_k\|_2$. With deflation of b the loss of orthogonality is proportional to κ_k in both U and V. The norm of the error $\|\bar{x} - x\|_2$ in the computed solution $\bar{x}$ is $1.149 \cdot 10^{-10}$ for $k = 8$. This is of the same magnitude as the loss of orthogonality in V_k and U_k. The corresponding error norm for the Householder algorithm is $2.181 \cdot 10^{-10}$. This strongly suggests forward stability of the MGS-PLS algorithm.

It has been suggested that the deflation of b can be omitted, since in exact arithmetic

$$u_i^T \Big(b - \sum_{j=1}^{i-1} \zeta_j u_j \Big) = u_i^T b.$$

The columns to the right in Table 2.2 show the effect of omitting the deflation of b. Although the loss of orthogonality in U_k is nearly unchanged, the loss of orthogonality in V_k is now proportional to κ_k^2. The norm of the error in the computed solution is also of the same magnitude and equals $0.724 \cdot 10^{-1}$. This loss of accuracy is similar to that when MGS is incorrectly used to solve a least squares problem by computing the right-hand side as $c = Q^T b$. We conclude that omitting the deflation of b destroys the otherwise excellent numerical accuracy of the MGS-PLS. □

Algorithm 2.6.2 (*NIPALS Algorithm*)

```
function [x,U,P,V] = nipls(A,b,k)
% NIPLS computes at most k PLS factors for the
%   least squares problem min||A x - b||_2.
% ----------------------------------------------
[m,n] = size(A); x = zeros(n,1);
for i = 1:k
```

[13] How to generate a random orthogonal matrix is described in Stewart [267, 1980].

```
% Determine i'th column of B
   v = A'*b; nv = norm(v);
   if nv == 0, k = i-1; break end
   v = v/nv;  u = A*v;
   rho(i) = norm(u);  u = u/rho(i);
   if i > 1, theta(i) = p'*v; end
% Deflate A and b
   p = A'*u;  z(i) = u'*b;
   A = A - u*p';  b = b - u*z(i);
   V(:,i) = v; U(:,i) = u; P(:,i) = p;
end
% Solve the bidiagonal system.
y(k) = z(k)/rho(k);
for i = k-1:(-1):1
   y(i) = (z(i) - theta(i+1)*y(i+1))/rho(i);
end
x = V*y;
```

Several generalizations of the basic PLS algorithm have been devised; see Wold et al. [298, 2001]. The relationship between the original PLS algorithm and the GKH algorithm is analyzed by Eldén [88, 1984] and Bro and Eldén [37, 2008].

2.6.5 Least Angle Regression and l_1 Constraints

A problem related to the standard case of LSQI is the least squares problem with a bound on the sum of the absolute values of the coefficients

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad \|x\|_1 = e^T x \le \mu, \tag{2.6.50}$$

where $e = (1, \ldots, 1)^T$. The use of (2.6.50) for variable selection in least squares problems is due to Tibshirani [279, 1996]. He gave this procedure the colorful name **LASSO**, which stands for "least absolute shrinkage and selection operator". For a fixed value of the regularization parameter μ (2.6.50) is an optimization problem, whose objective function is strictly convex over a convex feasible region. Such a problem has a unique minimizer. Let x_{LS} be the unconstrained least squares solution and set $\mu_{\mathrm{LS}} = \|x_{\mathrm{LS}}\|_1$. We shall see that for $\mu \in [0, \mu_{\mathrm{LS}}]$, the trajectory of the solution is piecewise linear.

The use of the ℓ_1-norm instead of the the ℓ_2-norm norm for the regularization term has a great influence on the character of the solution. For the spectral norm the feasible region $\|x\|_2 \le \mu$ is a hyper-sphere. For the 1-norm the feasible region $\|x\| \le \mu$ is a diamond-shaped polyhedron, which unlike the sphere has many sharp corners, edges and faces at which one or several parameters are zero. This structure of the ℓ_1-norm favors solutions with few nonzero coefficients.

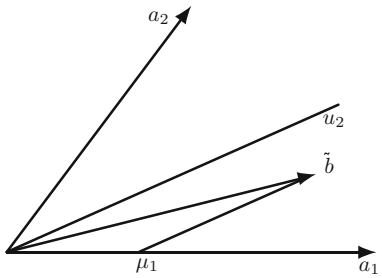


Fig. 2.15 Least angle regression for $n = 2$ variables. Here $\widetilde{b}$ is the orthogonal projection of b onto span$\{a_1, a_2\}$

Osborne et al. [224, 2000] proposed an algorithm for computing this trajectory for the ℓ_1 constrained least squares problem based on standard methods for convex programming. We describe here a more intuitive algorithm by Efron et al. [81, 2004], derived by modifying a variable selection algorithm called **least angle regression** (LAR).

Assume that $A = (a_1, \ldots, a_n)$ has linearly independent columns, normalized so that $\|a_j\|_2 = 1$, $j = 1{:}n$. We start with $x_j = 0$, $j = 1{:}n$ and enter one variable at each step. The first variable to enter the regression is chosen as the one making the smallest angle with b. Since the columns of A are normalized to have unit length, this is the variable x_j with the largest correlation $|a_j^T b|$. This maximizes the decrease in residual norm for a fixed value of $\|x\|_2$. It is the same choice as in stepwise regression. Let this variable be x_1, so that

$$|a_1^T b| > |a_j^T b|, \quad j \neq 1.$$

In stepwise regression the next iterate $x_1 = a_1^T b$ is the least squares solution, for which the residual $r = b - (a_1^T b)x_1$ is orthogonal to a_1. In LAR a smaller step

$$x_1 = \gamma(a_1^T b), \quad 0 < \gamma \leq 1,$$

is taken. As γ increases, the correlation between a_1 and $r(t) = b - \gamma s_1 a_1$ becomes smaller. For some intermediate value $\gamma = \gamma_1$, there is another column a_j, such that $|a_1^T r(\gamma)| = |a_j^T r(\gamma)|$. At this point the residual $r(1) = r(\gamma_1)$ bisects the angle between a_1 and a_j, and the variable x_j joins the active set.

In the next step the solution moves towards the unconstrained solution for $A = (a_1, a_2)$ as illustrated in Fig. 2.15. It is apparent that the residual vector will change in the direction of the bisector of a_1 and a_2, but a full step is not taken. From the lemma below it follows that this will keep the correlations of the residual and the active variables tied and decreasing.

Lemma 2.6.3 *Consider a least squares problem* $\min_x \|Ax - b\|_2$, *where the columns of A have unit length* $\|a_j\|_2 = 1$. *Assume that b makes the same acute angle with each column* a_j, $j = 1{:}n$. *Set* $x(t) = tx_{LS}$, *where* $t \in [0, 1]$ *and* x_{LS} *is the least*

squares solution. Then the residual $r(t) = b - Ax(t)$ also makes equal angle to the columns of A.

Proof Since b makes equal angles with each column of A, the correlations are equal, i.e., $|A^T b| = ce$, where $e = (1, \ldots, 1)^T$. Then the residual vector is $r(t) = b - tA(A^T A)^{-1} A^T b$, and hence

$$A^T r(t) = A^T b - tA^T b = (1-t)A^T b.$$

It follows that $|A^T r(t)| = (1-t)|A^T b| = c(1-t)e$. □

The subsequent steps are similar. Let σ be the index set pointing to the current nonzero components of x and $\sigma \cup \sigma^C = \{1, 2, \ldots, n\}$. Assume that a new variable has just become active and is zero. Denote the current solution by $x^{(k)}$ and the residual by $r^{(k)} = b - Ax^{(k)}$. The correlations $|a_j^T r^{(k)}|$, $j \in \sigma$, are all equal. The solution moves towards the least squares solution in the subspace corresponding to the active set, which is $x = x^{(k)} + u^{(k)}$, where $u = u^{(k)}$ solves $\min_u \|A^{(k)} u - r^{(k)}\|_2$. By Lemma 2.6.3, this direction of change makes equal angles in the residual space with all $a_\cdot$, $j \in \sigma$. The maximal reduction in residual norm is obtained by making the residual orthogonal to a_j, $j \in \sigma$, but a smaller step is taken, say

$$x = x^{(k)} + \gamma u^{(k)}, \quad \gamma \in (0, 1]. \tag{2.6.51}$$

At the current point $x^{(k)}$ the correlations are

$$a_j^T r^{(k)} = \begin{cases} \widehat{c} & \text{if } j \in \sigma, \\ c_j & \text{otherwise.} \end{cases}$$

When the solution moves towards the least squares solution according to (2.6.51), the correlations of the variables in the active set will change according to $(1-\gamma)\widehat{c}$. The breakpoint will occur for the smallest value of γ for which

$$(1-\gamma)\widehat{c} = |c_j - \gamma \beta_j|, \quad \beta_j = a_j^T A u^{(k)}, \quad j \notin \sigma^C.$$

It follows that LAR will use the stepsize

$$\widehat{\gamma} = \min_{j \in \sigma^C}{}^{+} \left\{ \frac{\widehat{c} - c_j}{\widehat{c} - \beta_j}, \frac{\widehat{c} + c_j}{\widehat{c} + \beta_j} \right\}, \tag{2.6.52}$$

where $\min^+$ indicates that the minimum is taken only over positive components for each j. The sum of squares of the residuals in the above algorithm is monotonically decreasing as μ increases. After n steps LAR will terminate with the full least squares solution. In this discussion, we have assumed that only a single variable becomes active at a time.

By construction, the trajectory of the solution in LAR is piecewise linear with n breakpoints. In many cases this trajectory coincides with that of the ℓ_1 constrained least squares problem if we set $\mu_k = \sum_{j \in \mathcal{A}_k} |x|$. Indeed, with the following small modification the LAR algorithm can be used for solving the ℓ_1 constrained least squares problem: When a nonzero variable becomes zero and is about to change sign, the variable is removed from the active set and the least squares direction of change recomputed.

Theorem 2.6.5 *Let $x(\mu)$ be the solution of the ℓ_1 constrained least squares problem. Then there exists a finite set of break points $0 = \mu_0 \le \mu_1 \le \cdots \le \mu_p = \mu_{\mathrm{LS}}$, such that $x(\mu)$ is a piecewise linear function*

$$x(\mu) = x(\mu_k) + (\mu - \mu_k)(x(\mu_{k+1}) - x(\mu_k)), \quad \mu_k \le \mu \le \mu_{k+1}. \tag{2.6.53}$$

The ℓ_1 constrained least squares solution can be computed with about the same arithmetic cost as a single least squares problem. As in stepwise regression, the QR factorization of the columns in the active set is modified in each step. When a new variable is added, the factorization is updated by adding the new column. In case a variable becomes zero, the factorization is downdated. Although unlikely, the possibility of a multiple change in the active set cannot be excluded. A brute force method to cope with this is to make a small change in the right-hand side. It follows from continuity that no index set σ can be repeated in the algorithm. There are only a finite number of steps in the algorithm, usually not much greater than $\min\{m, n\}$.

If the linear system $Ax = b$ is consistent, then problem (2.6.50) simplifies to

$$\min_x \|x\|_1 \quad \text{subject to} \quad Ax = b. \tag{2.6.54}$$

This formulation is used in Basis Pursuit, which is a principle for decomposing a signal using a superposition of basis functions from some "dictionary" of, e.g. wavelet packages; see Chen et al. [51, 2001]. Problem (2.6.54) is connected to an LP program in standard form

$$\min_{u,v} c^T y \quad \text{subject to} \quad By = b, \quad y \ge 0.$$

If we take

$$c = \begin{pmatrix} e \\ -e \end{pmatrix}, \quad y = \begin{pmatrix} u \\ v \end{pmatrix}, \quad B = \begin{pmatrix} A & -A \end{pmatrix},$$

and y solves the LP program, then $x = u - v$ solves (2.6.54).

An important application of ℓ_1 regularization is in signal processing. If the true signal is known to be sparse, i.e., has few nonzero coefficients, then ℓ_1 regularization will identify the correct predictor with high probability. This property is the basis for **compressive sensing** in signal processing. To minimize the number of nonzero entries in a signal x is equivalent to minimizing the ℓ_0 "norm". This problem is

known to be computationally intractable. The ℓ_1 norm can be seen as the closest "convex" approximation. A general mathematical framework has been developed in which the ℓ_1 approximation can be *proved to recover sparse solutions exactly*; see Candès [41, 2006]. A matrix analogue of recovering a sparse vector is to minimize the nuclear norm of a matrix, i.e., the sum of its singular values. As in the case of a vector, this can be be viewed as a convex approximation of the rank of the matrix. The literature on compressive sensing is huge and growing fast.

Exercises

2.6.1 Consider the (thin) QR factorization

$$\begin{pmatrix} A \\ \mu I_n \end{pmatrix} = \begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix} R, \quad \mu > 0,$$

where $A, \in \mathbb{R}^{m\times n}$ and $Q_2 \in \mathbb{R}^{n\times n}$. Show that $A^TA + \mu^2 I_n = R^T R$ and that

$$A(A^TA + \mu^2 I_n)^{-1} = \frac{1}{\mu} Q_1 Q_2^T. \tag{2.6.55}$$

2.6.2 Describe an efficient algorithm using Givens rotations for computing the QR factorization of a matrix

$$\begin{pmatrix} B \\ \mu D \end{pmatrix}, \quad R \in \mathbb{R}^{n\times n},$$

where B is upper triangular and D diagonal.
Hint: The number of flops required for the factorization is about $11n$.

2.6.3 (Eldén [87, 1984]) An important special case of Tikhonov regularization is when $A = K$ and L are upper triangular Toeplitz matrices, i.e.,

$$K = \begin{pmatrix} k_1 & k_2 & \dots & k_{n-1} & k_n \\ & k_1 & k_2 & \dots & k_{n-1} \\ & & \ddots & \ddots & \vdots \\ & & & k_1 & k_2 \\ & & & & k_1 \end{pmatrix}.$$

Such systems arise when convolution-type Volterra integral equations of the first kind are discretized. Develop a method using Givens rotations for computing the QR factorization of $\begin{pmatrix} K \\ \mu L \end{pmatrix}$, which for a fixed value of μ only uses about $3n^2$ flops. Is the final triangular matrix Toeplitz?
Hint: In the first step use Givens rotations in the planes $(1, n+1), (2, n+1), \dots, (n, 2n)$ to zero the main diagonal in L. Notice that the rotation angle is the same in all rotations and that the Toeplitz structure is preserved.

2.6.4 Work out the details of the transformation to standard for Tikhonov regularization, when L is the discrete approximation to the first derivative operator in (2.6.6). What is the null space of L in this case?

2.6.5 Show that transforming A to lower bidiagonal form using an initial transformation $Q_0 b = \beta e_1$ gives the same result as transforming the matrix $(b,\ A)$ to upper bidiagonal form using $P_0 = I$.

2.6.6 (a) Develop an algorithm using Givens rotations for transforming a lower bidiagonal matrix $B \in \mathbb{R}^{(n+1)\times n}$ into an upper bidiagonal matrix $R \in \mathbb{R}^{n\times n}$.

(b) Extend the algorithm in (a) for solving a lower bidiagonal least squares problem $\min_y \|By - \beta_1 e_1\|_2$.

2.6.7 For a matrix $A \in \mathbb{R}^{m\times n}$ of full column rank, the matrix $B \in \mathbb{R}^{n\times n}$ in the bidiagonal decomposition

$$A = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} B \\ 0 \end{pmatrix} V^T = U_1 B V^T$$

is nonsingular and the pseudoinverse is $A^\dagger = VB^{-1}U_1^T$. This suggests the following method to compute $A^\dagger$. Compute the bidiagonal factorization using Householder reflectors, and form U_1 by accumulating the left Householder reflectors. Solve the bidiagonal matrix equation $BY = U_1^T$ and compute $A^\dagger = VY$. Determine the number of flops needed by this method.

2.6.8 (a) Consider the least squares problem $\min_x \|b - Ax\|_2$ where $A \in \mathbb{R}^{m\times n}$, $m > n$ and $b \notin \mathcal{R}(A)$. Show that Householder QR factorization applied to the extended matrix $\begin{pmatrix} b & A \end{pmatrix}$ gives the factorization

$$\begin{pmatrix} b & A \end{pmatrix} = Q \begin{pmatrix} \beta e_1 & H \\ 0 & 0 \end{pmatrix},$$

where e_1 is the first unit vector and $H \in \mathbb{R}^{(n+1)\times n}$ a Hessenberg matrix with $h_{k+1,k} \neq 0$, $k = 1{:}n$. Conclude that the least squares solution x is obtained by solving $\min_x \|Hx - \beta_1 e_1\|_2$.

(b) Assume that standard pivoting is used in the QR factorization of $\begin{pmatrix} b & A \end{pmatrix}$, except that b is fixed as first column. Show that b is a linear combination of a subset of k columns in A if and only if $h_{k+1,k} = 0$.

2.7 Some Special Least Squares Problems

In this section we treat a selection of least squares problems, that do not fit the standard Gauss–Markov model and require special methods for their solution. Included are problems involving linear equality or inequality constraints, as well as problems with indefinite or more general covariance structure. In the "total least squares model" errors are allowed in both the right-hand side b and in A. A special case of this is orthogonal regression.

2.7.1 Weighted Least Squares Problems

For the standard Gauss–Markov model to be valid, the errors in the right-hand side must be independently and identically distributed with covariance matrix $\sigma^2 I$. Consider a least squares problem $\min_x \|Ax - b\|_2$, where the covariance matrix is the positive diagonal matrix,

$$V = \sigma^2 \text{diag}(v_{11}, v_{22}, \ldots, v_{mm}) > 0.$$

Here the equations should be weighted so that the errors have equal variance. This is achieved by a rescaling

$$\min_x \|D(Ax - b)\|_2 = \min_x \|(DA)x - Db\|_2. \tag{2.7.1}$$

where $D = \mathrm{diag}(v_{ii}^{-1/2})$. Note that the *smaller* the variance v_{ii}, the *larger* the weight given to a particular equation.

Problems for which the error variances $\sigma^2 v_{ii}$ have widely different magnitude, i.e., $d_{\max}/d_{\min} \gg 1$, are called **stiff**. For such problems DA in (2.7.1) can be ill-conditioned even when A is well-conditioned. An example is Läuchli's problem (see Example 2.1.5, p. 225) in which the first equation $x_1 + x_2 + x_3 = 1$ has a much larger weight than the rest. Stiff least squares problems arise, e.g., in geodetic problems, electrical networks, certain classes of finite element problems, and in interior point methods for constrained optimization.

Special care may be needed when solving stiff least squares problems. We assume in the following that the matrix A is row equilibrated, i.e.,

$$\max_{1 \le j \le n} |a_{ij}| = 1, \quad i = 1{:}m,$$

and that the weights $D = \mathrm{diag}(d_1, d_2, \ldots, d_m)$ have been ordered so that $\infty > d_1 \ge d_2 \ge \cdots \ge d_m > 0$. Note that only the *relative size* of the weights influences the solution.

The method of normal equations is not well suited for solving stiff problems. To illustrate this, we consider the special case where only the first $p < n$ equations are weighted,

$$\min_x \left\| \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} x - \begin{pmatrix} \gamma b_1 \\ b_2 \end{pmatrix} \right\|_2^2, \tag{2.7.2}$$

$A_1 \in \mathbb{R}^{p \times n}$ and $A_2 \in \mathbb{R}^{(m-p) \times n}$. For problem (2.7.2) the matrix of normal equations becomes

$$B = \begin{pmatrix} \gamma A_1^T & A_2^T \end{pmatrix} \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} = \gamma^2 A_1^T A_1 + A_2^T A_2.$$

If $\gamma > \mathbf{u}^{-1/2}$ ($\mathbf{u}$ 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, this is a disaster, since the solution depends critically on the less precise data in A_2.

For a stiff problem $\kappa(DA)$ is large. An upper bound 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 $\{D_r, A, b\}$ is ill-conditioned when $\gamma \gg 1$.* Note that when $\gamma \to \infty$ (2.7.2) becomes the least squares problem $\min_x \|A_2 x - b_2\|_2$ subject to linear constraints $A_1 x = b_1$; see Sect. 2.7.2.

QR factorization can be used to stably solve weighted problems. However, the example

$$A = \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix}, \quad A_1 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \end{pmatrix},$$

where A_2 arbitrary and $\gamma \gg 1$, shows that *it is essential to use column pivoting*. Since the first two columns of A_1 are linearly dependent, stability will be lost in QR factorization without interchanging the second and third columns. As the following example shows, the ordering of the rows is also important.

Example 2.7.1 (Powell and Reid [238, 1969]) The pivoted Householder QR method can also give poor accuracy for weighted problems. 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 $x = (1, 1, 1)$. With exact arithmetic, after the first step of Householder QR factorization of A, we obtain 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. But 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. (But compared to the method of normal equations, which fails already when $\gamma > u^{-1/2}$, this is an improvement!) □

Cox and Higham [61, 1998] show that provided an initial **row sorting** is performed, the pivoted Householder QR method has very good stability properties for weighted problems. The rows of $\widetilde{A} = DA$ and $\widetilde{b} = Db$ should be sorted to give decreasing ℓ_∞-norm:

$$\max_j |\widetilde{a}_{1j}| \geq \max_j |\widetilde{a}_{2j}| \geq \cdots \geq \max_j |\widetilde{a}_{mj}|. \tag{2.7.3}$$

(In Example 2.7.1 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 pivoted QR factorization can be used.

We now consider some hybrid methods that are suitable for solving stiff problems. In a first step Gaussian elimination is applied to reduce $A \in \mathbb{R}^{m\times n}$ to upper

trapezoidal form U. In general, column interchanges are needed to ensure numerical stability. Usually it will be sufficient to use partial pivoting combined with a check for linear independence. After p steps, let $\widetilde{a}_{q,p+1}$ be the element of largest magnitude among the entries $p+1{:}m$ of column $p+1$. If $|\widetilde{a}_{q,p+1}| < tol$, then 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. If the rank $(A) = n$, the resulting LDU factorization becomes

$$\Pi_1 A \Pi_2 = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = LDU = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} DU, \tag{2.7.4}$$

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

Setting $\widetilde{x} = \Pi_2^T x$ and $\widetilde{b} = \Pi_1 b$, the least squares problem $\min_x \|Ax - b\|_2$ becomes

$$\min_y \|Ly - \widetilde{b}\|_2, \qquad DU\widetilde{x} = y. \tag{2.7.5}$$

If the initial LU factorization is computed with row and column interchanges, the resulting factor L tends to be well-conditioned. and any ill-conditioning is usually reflected in D. The least squares problem in (2.7.5) can then be solved using the normal equations

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

without substantial loss of accuracy. This is known as the **Peters–Wilkinson** method; see [234, 1970]. Forming the symmetric matrix $L^T L = L_1^T L_1 + L_2^T L_2$ requires $n^2(m - \frac{2}{3}n)$ flops and the Cholesky factorization $n^3/3$ flops. Hence, neglecting terms of order n^2, a total of $2n^2(m - \frac{1}{3}n)$ flops is required. Although this is always more than required by the standard method of normal equations, it is a more stable method. In a variant of the Peters–Wilkinson method, problem (2.7.5) is solved by computing an orthogonal factorization of the lower trapezoidal matrix L,

$$Q^T L = \begin{pmatrix} \widehat{L} \\ 0 \end{pmatrix}, \qquad \widehat{L} y = Q^T \widetilde{b}, \tag{2.7.6}$$

where $\widehat{L}$ is square and lower triangular; see Cline [54, 1973].

Example 2.7.2 (Noble [215, 1976]) Consider the matrix A and the corresponding (exact) normal equations matrix

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

If $\epsilon \le \sqrt{u}$, then in floating point computation $fl(3+2\epsilon^2) = 3$, and the computed matrix $fl(A^T A)$ has rank one. But the LDU 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. Since $L^\dagger = (L^T L)^{-1} L^T$, Theorem 2.2.3 shows that the pseudoinverse is obtained from

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

Here there is no cancellation. □

A more complete treatment of weighted least squares and the general linear model is given in Björck [27, 1996], Chap. 3. Methods for solving least squares problems based on Gaussian elimination were used by Noble [215, 1976]. Sautter [249, 1979] gives a detailed analysis of stability and rounding errors in the LU algorithm for computing pseudoinverse solutions.

2.7.2 Linear Equality Constraints

In some least squares problems 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.

Given matrices $A \in \mathbb{R}^{m \times n}$ and $C \in \mathbb{R}^{p \times n}$, problem **LSE** is to find $x \in \mathbb{R}^n$ such that

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad Cx = d. \tag{2.7.7}$$

A solution exists if and only if the linear system $Cx = d$ is consistent. A robust algorithm should check for possible inconsistency of the constraints $Cx = d$. If rank $(C) = p$, then $Cx = d$ is consistent for any right-hand side d. In the inconsistent case, problem LSE may be reformulated as a **sequential least squares problem**

$$\min_{x \in S} \|Ax - b\|_2, \qquad S = \{x \mid \|Cx - d\|_2 = \min\}. \tag{2.7.8}$$

A solution to problem (2.7.7) is unique if and only if the null spaces of A and C intersect trivially, i.e., $\mathcal{N}(A) \cap \mathcal{N}(C) = \{0\}$, or equivalently

$$\operatorname{rank} \begin{pmatrix} C \\ A \end{pmatrix} = n. \tag{2.7.9}$$

If not, there is a vector $z \neq 0$ such that $Az = Cz = 0$, and if x solves (2.7.8), $x + z$ is a different solution. In the following we therefore assume that rank $(C) = p$ and that (2.7.9) is satisfied.

The most efficient way to solve problem LSE is to derive an equivalent unconstrained least squares problem of lower dimension. There are 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 C to upper trapezoidal form. It is essential that column interchanges be used in this step. In order to be able to solve also the more general problem (2.7.8) we compute a QR factorization of C such that

$$Q_C^T C \Pi_C = \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix}, \qquad \bar{d} = Q_C^T d = \begin{pmatrix} \bar{d}_1 \\ \bar{d}_2 \end{pmatrix}. \tag{2.7.10}$$

Here $R_{11} \in \mathbb{R}^{r\times r}$ is upper triangular and nonsingular, where $r = \operatorname{rank}(C) \le p$. Further, $\bar{d}_2 = 0$ if and only if the constraints are consistent. With this factorization and $\bar{x} = \Pi_C^T x$, the constraints become

$$\left(R_{11}, R_{12}\right)\bar{x} = R_{11}\bar{x}_1 + R_{12}\bar{x}_2 = \bar{d}_1. \tag{2.7.11}$$

The permutation Π_C is also applied to the columns of A. Partitioning the resulting matrix conformally with (2.7.10) gives $A\Pi_C = \begin{pmatrix} \bar{A}_1 & \bar{A}_2 \end{pmatrix}$. Solving $R_{11}\bar{x}_1 = (\bar{d}_1 - R_{12}\bar{x}_2)$ in (2.7.11) and substituting in $Ax - b = \bar{A}_1\bar{x}_1 + \bar{A}_2\bar{x}_2 - b$, it follows that the original problem LSE is equivalent to the unconstrained least squares problem

$$\min_{\bar{x}_2} \|\widehat{A}_2\bar{x}_2 - \widehat{b}\|_2, \tag{2.7.12}$$

where

$$\widehat{A}_2 = \bar{A}_2 - \bar{A}_1 R_{11}^{-1} R_{12}, \qquad \widehat{b} = b - \bar{A}_1 R_{11}^{-1}\bar{d}_1.$$

Note that $\widehat{A}_2 \in \mathbb{R}^{m\times(n-r)}$ 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 condition (2.7.9) 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 $\widehat{A}_2$. The resulting algorithm can be kept remarkably compact, as exemplified by the Algol program of Björck and Golub [29, 1967].

In the null space method the LQ factorization $C = LQ^T$ is computed, with L lower triangular and Q orthogonal. (This is equivalent to the QR factorization of C^T.) If A is also postmultiplied by Q, we obtain

$$\begin{pmatrix} C \\ A \end{pmatrix} Q = \begin{pmatrix} C \\ A \end{pmatrix} \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} = \begin{pmatrix} L & 0 \\ AQ_1 & AQ_2 \end{pmatrix}, \quad L \in \mathbb{R}^{p\times p}. \tag{2.7.13}$$

Here Q_2 is an orthogonal basis for the null space of C. The matrix Q can be constructed as a product of Householder reflectors. Set $x = Qy$ and split the solution into the sum of two orthogonal components by setting

$$x = x_1 + x_2 = Q_1 y_1 + Q_2 y_2, \quad y_1 \in \mathbb{R}^p, \quad y_2 \in \mathbb{R}^{(n-p)}. \tag{2.7.14}$$

Here $Cx_2 = CQ_2 y_2 = 0$, i.e., x_2 lies in the null space of C. From the assumption that rank $(C) = p$, it follows that L is nonsingular and the constraints now give $Ly_1 = d$. The residual vector can be written

$$r = b - AQy = c - AQ_2 y_2, \quad c = b - (AQ_1)y_1.$$

Hence, y_2 is the solution to the unconstrained least squares problem

$$\min_{y_2} \|(AQ_2)y_2 - c\|_2. \tag{2.7.15}$$

This reduced problem can be solved by computing the QR factorization of AQ_2. If (2.7.9) is satisfied, then rank $(AQ_2) = n - p$ and the solution to (2.7.15) is unique. Let y_2 be that unique solution. Since

$$\|x\|_2^2 = \|x_1\|_2^2 + \|Q_2 y_2\|_2^2 = \|x_1\|_2^2 + \|y_2\|_2^2,$$

it follows that $x = Qy$ is the minimum-norm solution to problem LSE.

The representation in (2.7.14) of the solution x can be used as a basis for a perturbation theory for problem LSE. A strict analysis is given by Eldén [86, 1982], but the result is too complicated to be given here. If the matrix C is well-conditioned, then the sensitivity is governed by $\kappa(AQ_2)$, for which $\kappa(A)$ is an upper bound.

Both the direct elimination and the null space method 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.

2.7.3 Linear Inequality Constraints

Often linear least squares problems arise where the solution is subject to linear inequality constraints. In this section we discuss a few simple special cases.

PROBLEM LSI:

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad l \le Cx \le u, \tag{2.7.16}$$

where $A \in \mathbf{R}^{m\times n}$ and $C \in \mathbf{R}^{p\times n}$ and the inequalities are to be interpreted componentwise. If c_i^T denotes the ith row of the constraint matrix C then the constraints can also be written

$$l_i \le c_i^T x \le u_i, \qquad i = 1{:}p.$$

The existence, uniqueness, and boundedness of solutions to problem LSI are treated in Fletcher [101, 1987] and Lötstedt [198, 1983].

Theorem 2.7.1 (Lötstedt [198, Theorem 1]) *Let the solution to Problem* LSI *be split into mutually orthogonal components*

$$x = x_R + x_N, \quad x_R \in \mathcal{R}(A^T), \quad x_N \in \mathcal{N}(A). \tag{2.7.17}$$

If the set $\mathcal{M} = \{l \le Cx \le u\}$ *is not empty, then there exists a bounded solution* x^* *to* (2.7.16). *Further,* Ax *and* $x_R = A^{\dagger}Ax$ *are uniquely determined. In particular, if* rank $(A) = n$, *then* $\mathcal{N}(A)$ *is empty and the solution* x *is unique*

Proof The existence of a bounded solution follows from the fact that the objective function $\|Ax - b\|_2$ is bounded below by 0 and the constraint set $l \le Cx \le u$ is convex and polyhedral. □

The case when the inequalities in problem LSI are simple bounds deserves separate treatment:

PROBLEM BLS:

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad l \le x \le u. \tag{2.7.18}$$

Such bound-constrained least squares problems arise in many practical applications, e.g., reconstruction problems in geodesy and tomography, contact problems for mechanical systems, and modeling of ocean circulation. Sometimes it can be argued that the linear model is only realistic when the variables are constrained within meaningful intervals. For reasons of computational efficiency such constraints should be considered separately from general constraints.

In the special case when only one-sided bounds on x are specified in problem BLS it is no restriction to assume that these are nonnegativity constraints.

PROBLEM NNLS:

$$\min_x \|Ax - b\|_2 \quad \text{subject to} \quad x \ge 0. \tag{2.7.19}$$

Stoer [277, 1971] gives an active set algorithm for problem LSI. At the solution of (2.7.16) a certain subset of constraints $l \le Cx \le u$ will be active, i.e., satisfied with equality. If this subset is known, the LSI problem reduces to a problem with *equality constraints* only and can be solved efficiently. In an active set algorithm a sequence of equality-constrained problems are solved corresponding to a prediction of the correct

active set, called the working set. The working set includes only constraints that are satisfied at the current approximation, but not necessarily all such constraints.

Problem LSI is equivalent to the quadratic programming problem

$$\min_x \left(\tfrac{1}{2}x^T Qx + c^T x\right) \quad \text{subject to} \quad l \le Cx \le u, \tag{2.7.20}$$

where $Q = A^T A$, $c = -2A^T b$. Since problem (2.7.20) arises as a subproblem in general nonlinear programming algorithms, it has been studied extensively, and many algorithms are available to solve it. For problem LSI the matrix Q in (2.7.20) is by definition positive definite or semidefinite, and hence (2.7.20) is a convex program. When A is ill-conditioned, the computed cross-product matrix Q may become indefinite due to rounding errors and cause slow and erratic convergence. Therefore, methods for quadratic programming should preferably be adapted to work directly with A.

In the case when A has full rank, problem LSI always has a unique solution. Otherwise there may be an infinite manifold M of optimal solutions with a unique optimal value. In this case we can seek the unique solution of minimum norm, which satisfies $\min_{x\in M} \|x\|_2$. This is a least distance problem (LSD).

PROBLEM LSD:

$$\min_x \|x\|_2 \text{ subject to } g \le Gx \le h. \tag{2.7.21}$$

Several implementations of varying generality of active set methods for problem BLS have been developed. Lötstedt [199] has developed a two-stage algorithm to solve problem BLS. Lawson and Hanson [190] give a Fortran implementation of an algorithm for problem NNLS. For large-scale BLS problems the classical approach has two main disadvantages. One is that constraints are added/deleted one at a time to the working set. The other is that the exact minimizer with the current working set is required. To resolve these problems, methods based on gradient projection combined with the CG method have been devised; see Friedlander et al. [106, 1995]. Similar features are implemented in the software package BCLS by M. Friedlander, available at http://www.cs.ubc.ca/~mpf/bcls/.

Cline [55, 1975] showed how problem LSI can be transformed into a least distance problem. Let

$$Q^T APV = \begin{pmatrix} T & 0 \\ 0 & 0 \end{pmatrix}$$

be a complete orthogonal factorization of A with T triangular and nonsingular. Then (2.7.16) can be written

$$\min_y \|Ty_1 - c_1\|_2 \quad \text{subject to} \quad l \le Ey \le u,$$

where

$$E = (E_1, E_2) = CPV, \qquad y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = V^T Px$$

are conformally partitioned, and $y_1 \in \mathbf{R}^r$. Making the further change of variables $z_1 = Ty_1 - c_1$, $z_2 = y_2$, and substituting $y_1 = T^{-1}(z_1 + c_1)$ in the constraints, we arrive at an equivalent least distance problem:

$$\min_z \|z_1\|_2 \ \text{ subject to } \ \widetilde{l} \le G_1 z_1 + G_2 z_2 \le \widetilde{u}, \tag{2.7.22}$$

where $G_1 = E_1 T^{-1}$, $G_2 = E_2$, $\widetilde{l} = l - G_1 c_1$, and $\widetilde{u} = u - G_1 c_1$. Note that if A has full column rank, then $r = n$ and $z = z_1$, so we get a least distance problem of the form (2.7.21). Lawson and Hanson [190, 1974], Chap. 23, give a Fortran subroutine for problem LSD with lower bounds only based on their active set method for NNLS.

LSSOL is a set of Fortran subroutines for convex quadratic programming and problem LSI. A mixture of simple bounds and general linear constraints can be handled; see Gill et al. [121, 1986]. LSSOL uses a two-phase active set method, and a linear term can be added to the objective function.

2.7.4 Generalized Least Squares Problems

The **generalized QR** (GQR) factorization was introduced by Hammarling [147, 1987] and Paige [227, 1990]. 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. Then the GQR factorization is

$$A = QR, \qquad B = QTZ, \tag{2.7.23}$$

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

$$R = \begin{pmatrix} R_{11} \\ 0 \end{pmatrix} \quad (m \ge n), \qquad R = \begin{pmatrix} R_{11} & R_{12} \end{pmatrix} \quad (m < n), \tag{2.7.24}$$

and

$$T = \begin{pmatrix} 0 & T_{12} \end{pmatrix} \quad (m \le p), \qquad T = \begin{pmatrix} T_{11} \\ T_{21} \end{pmatrix} \quad (m > p). \tag{2.7.25}$$

If B is square and nonsingular, GQR implicitly gives the QR factorization of $B^{-1}A$. Explicitly forming $B^{-1}A$ may result in a loss of precision when B is ill-conditioned. There is also a similar generalized factorization related to the QR factorization of AB^{-1}. Routines for computing a GQR factorization are included in LAPACK. These factorizations as well as the GSVD allow the solution of very general formulations of least squares problems.

The systematic use of GQR as a basic conceptual and computational tool is explored by Paige [227, 1990]. These generalized decompositions and their applications are discussed in Anderssen et al. [2, 1992].

It is straightforward to generalize the Gauss–Markov model (see Definition 2.1.1) to the case when the error vector e has a positive definite covariance matrix $\sigma^2 V$.

Theorem 2.7.2 *Consider a Gauss–Markov linear model with symmetric positive definite error covariance matrix $\mathcal{V}(e) = \sigma^2 V$. If $A \in \mathbb{R}^{m\times n}$ has full column rank, then the best unbiased linear estimate $\widehat{x}$ minimizes $(Ax-b)^T V^{-1}(Ax-b)$, and satisfies the* **generalized normal equations**

$$A^T V^{-1} A x = A^T V^{-1} b. \tag{2.7.26}$$

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

$$\mathcal{V}(\widehat{x}) = \sigma^2 (A^T V^{-1} A)^{-1} \tag{2.7.27}$$

and an unbiased estimate of σ^2 is $s^2 = \widehat{r}^T V^{-1} \widehat{r}/(m-n)$.

Proof Since V is positive definite, it has a unique Cholesky factorization $V = LL^T$, with nonsingular $L \in \mathbb{R}^{m\times m}$. The transformed Gauss–Markov model is

$$(L^{-1}A)x = L^{-1}b + f, \quad f = L^{-1}e, \tag{2.7.28}$$

where f has covariance matrix $\sigma^2 L^{-1} V L^{-T} = \sigma^2 I$. The proof now follows with $\widetilde{A} = L^{-1}A$ and $\widetilde{b} = L^{-1}b$ replacing A and b in Theorem 2.1.1. □

The assumptions about the rank of A and V can be dropped. It is only necessary to assume that A and V have the same number of rows. If $V = LL^T$, where $L \in \mathbb{R}^{m\times k}$ and $k \le m$, then the Gauss–Markov model can be replaced by the equivalent model

$$Ax = b + Lv, \qquad \mathcal{V}(v) = \sigma^2 I, \tag{2.7.29}$$

which allows for rank-deficiency in both A and V. It must be required that the consistency condition $b \in \mathcal{R}(A) \cup \mathcal{R}(L)$ is satisfied, because otherwise b could not have come from the linear model (2.7.29). The best unbiased linear estimate of x is a solution to the constrained linear least squares problem

$$\min_{x,v} v^T v \quad \text{subject to} \quad Ax = b + Lv. \tag{2.7.30}$$

If the solution $\widehat{x}$ to (2.7.30) is not unique, then $\widehat{x}$ is chosen as the solution of minimum norm. For simplicity, we consider here only the case when V is positive definite and $A \in \mathbb{R}^{m\times n}$ has full column rank.

A special case of the GQR factorization is used in Paige's method for the general linear model (2.7.29). In the first step the QR factorization

$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} n \\ m-n \end{matrix}, \quad Q^T b = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}, \tag{2.7.31}$$

with $R \in \mathbb{R}^{n\times n}$ nonsingular, is computed. The same orthogonal transformation is applied also to $L \in \mathbb{R}^{m\times m}$, giving

$$Q^T L = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \begin{matrix} \} & n \\ \} & m-n \end{matrix}.$$

From the nonsingularity of L it follows that rank $(C_2) = m - n$. The constraints in (2.7.29) can now be written in the partitioned form

$$\begin{pmatrix} R \\ 0 \end{pmatrix} x = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} v. \tag{2.7.32}$$

For any vector v, we can determine x so that $Rx = c_1 + C_1 v$. Next, an orthogonal matrix $P \in R^{m\times m}$ is determined so that $C_2 P = \begin{pmatrix} 0 & S \end{pmatrix} \in \mathbb{R}^{(m-n)\times m}$, with S upper triangular and nonsingular. With $v = Pu$, the second block of the constraints in (2.7.32) becomes

$$c_2 + \begin{pmatrix} 0 & S \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = 0.$$

Since P is orthogonal, $\|v\|_2 = \|u\|_2$ and the minimum in (2.7.30) is found from

$$Su_2 = -c_2, \quad v = P \begin{pmatrix} 0 \\ u_2 \end{pmatrix}. \tag{2.7.33}$$

Finally, x is obtained by solving 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 (2.7.30) with covariance matrix

$$\sigma^2 R^{-1} B^T B R^{-T}, \quad B^T = C_1^T P_1. \tag{2.7.34}$$

The algorithm can be generalized in a straightforward way to rank-deficient A and L. The general case is analyzed by Paige [226, 1979] and Korouklis and Paige [182, 1981]. A perturbation and rounding error analysis shows that the algorithm is numerically stable. If no advantage is taken of any special structure of A and L, Paige's method requires a total of about $4m^3/3 + 2m^2 n$ flops.

A linear system of the form

$$Mz = \begin{pmatrix} V & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = d = \begin{pmatrix} b \\ c \end{pmatrix}, \tag{2.7.35}$$

where $V \in \mathbb{R}^{m\times m}$ is positive semidefinite and $A \in \mathbb{R}^{m\times n}$, is called a **saddle-point system**. The augmented system for the standard least squares problem introduced in Sect. 2.1.2 is the special case when $V = I$ in (2.7.35).

Lemma 2.7.1 *Let $V \in \mathbb{R}^{m\times m}$ be positive semidefinite. Then the system* (2.7.35) *is nonsingular if and only if* rank $(A) = n$ *and* rank $\begin{pmatrix} V \\ A^T \end{pmatrix} = m$.

Proof The two conditions are necessary, because if they were not satisfied M would have linearly dependent columns. We next show that if the two conditions are satisfied, then the null space of M is empty and thus M is nonsingular. Suppose that $Vy + Ax = 0$ and $A^T y = 0$. Then $Vy \in \mathcal{R}(A)$ and $y \perp \mathcal{R}(A)$, and thus $y^T Vy = 0$. Since V is positive semidefinite, this implies that $Vy = 0$ and hence $Ax = 0$. But since rank $(A) = n$, it follows that $x = 0$. Finally, using the second condition shows that $Vy = 0$ and $A^T y = 0$ implies that $y = 0$. □

When V is positive definite the system gives the conditions for the solution of the generalized linear least squares (GLLS) problem

$$\min_x (Ax - b)^T V^{-1}(Ax - b) + 2c^T x. \tag{2.7.36}$$

Problem GLLS is the general univariate linear model with covariance matrix V. Eliminating y in (2.7.35) gives the generalized normal equations,

$$A^T V^{-1} Ax = A^T V^{-1} b - c, \quad y = V^{-1}(b - Ax), \tag{2.7.37}$$

also called the range space equations.

The system (2.7.35) also gives necessary conditions for y to solve the *equality constrained quadratic optimization* (ECQO) problem (cf. Theorem 2.1.3)

$$\min_y \frac{1}{2} y^T Vy - b^T y \quad \text{subject to} \quad A^T y = c. \tag{2.7.38}$$

Any solution $(x,\ y)$ is a **saddle point** for the Lagrangian function

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

i.e., $\min_y \max_x \mathcal{L}(x, y) = \max_x \min_y \mathcal{L}(x, y)$.

Problem ECQO occurs as a subproblem in constrained optimization, where y is a search direction and $\lambda = -x$ a vector of Lagrange multipliers. The system (2.7.38) represents the equilibrium of a physical system and occurs in numerous applications. In the null space method for solving problem ECQO the solution y is split as $y = y_1 + y_2$, where $y_1 \in \mathcal{R}(A)$ and $y_2 \in \mathcal{N}(A^T)$. Let the QR factorization of A be

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}.$$

Then Q_2 is an orthogonal basis for $\mathcal{N}(A^T)$ and we set

$$y = Qz = Q_1 z_1 + Q_2 z_2, \quad z_1 \in \mathbb{R}^{n\times n}, \quad z_2 \in \mathbb{R}^{m-n}.$$

The solution can now be obtained as follows:

1. Compute the minimum-norm solution y_1 of $A^T y = c$ from $R^T z_1 = c$, $y_1 = Q_1 z_1$.
2. Find z_2 by solving the projected system $Q_2^T V Q_2 z_2 = Q_2^T (b - V y_1)$.
3. Compute $y = Q \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$ and solve $Rx = Q_1^T (b - Vy)$ for x.

If the QR factorization is computed by Householder reflectors, then Q_1 and Q_2 can be represented by the Householder vectors and need not be explicitly formed. If only y in problem ECQO is wanted, then x need not be formed. The null space method is advantageous to use for solving *a sequence* of saddle-point systems where A remains fixed but with varying $V = V_k$, $k = 1, 2, \ldots$. In this case the null space matrix Q_2 needs only be computed once. In many applications V and A are large and sparse and iterative methods are to be preferred.

Direct and iterative solution methods for saddle-point systems are described in the comprehensive survey of Benzi et al. [18, 2005]. Arioli [4, 2000] gives an error analysis of the null space method for problem ECQO.

2.7.5 *Indefinite Least Squares*

A matrix $Q \in \mathbb{R}^{n\times n}$ is said to be J-orthogonal if

$$Q^T J Q = J, \tag{2.7.39}$$

where the matrix $J = \text{diag}(\pm 1)$ is the **signature matrix**. This implies that Q is non-singular. Multiplying (2.7.39) with QJ and using $J^2 = I$ it follows that $QJQ^{TJ} = I$, and hence $QJQ^T = J$. If Q_1 and Q_2 are J-orthogonal, then

$$Q_2^T Q_1^T J Q_1 Q_2 = Q_2^T J Q_2 = J,$$

i.e., a product of J-orthogonal matrices is J-orthogonal.

J-orthogonal matrices are useful in the treatment of problems with an underlying indefinite inner product. To construct J-orthogonal matrices we consider the block 2×2 system

$$Qx = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}. \tag{2.7.40}$$

Solving the first equation for x_1 and substituting in the second equation will exchange x_1 and y_1. This can be written as $\begin{pmatrix} x_1 \\ y_2 \end{pmatrix} = \text{exc}\,(Q) \begin{pmatrix} y_1 \\ x_2 \end{pmatrix}$, where

$$\operatorname{exc}(Q) = \begin{pmatrix} Q_{11}^{-1} & -Q_{11}^{-1}Q_{12} \\ Q_{21}Q_{11}^{-1} & Q_{22} - Q_{21}Q_{11}^{-1}Q_{12} \end{pmatrix}, \tag{2.7.41}$$

is the **exchange operator**. The (2, 2) block is the Schur complement of Q_{11} in Q.

Theorem 2.7.3 *Let $Q \in \mathbb{R}^{n\times n}$ be partitioned as in (2.7.40). If Q is orthogonal and Q_{11} nonsingular, then* exc (Q) *is J-orthogonal. If Q is J-orthogonal, then* exc (Q) *is orthogonal.*

Proof See Higham [163, 2003], Theorem 2.2. □

Consider the 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 2.7.3 it follows that

$$H = \operatorname{exc}(G) = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix} \tag{2.7.42}$$

is J-orthogonal: $H^T J H = I$, $J = \operatorname{diag}(1, -1)$. The matrix H is called a hyperbolic plane rotation, because it can be written as

$$H = \begin{pmatrix} \breve{c} & -\breve{s} \\ -\breve{s} & \breve{c} \end{pmatrix}, \quad \breve{c}^2 - \breve{s}^2 = 1,$$

where $\breve{s} = \sinh\theta$, $\breve{c} = \cosh\theta$ for some θ. A hyperbolic rotation H can be used to zero a selected component in a vector. Provided that $|\alpha| > |\beta|$, we have

$$H\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \sigma \\ 0 \end{pmatrix},$$

where

$$s = \beta/\alpha, \qquad \sigma = |\alpha|\sqrt{1 - s^2}, \qquad c = \sigma/\alpha. \tag{2.7.43}$$

The elements of a hyperbolic rotation H are unbounded and such transformations must be used with care. The direct computation of $y = Hx$ is not stable. Instead, as first shown by Chambers [50, 1971], a mixed form should be used based on the equivalence

$$H\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \quad \Leftrightarrow \quad G\begin{pmatrix} y_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ y_2 \end{pmatrix},$$

where $H = \operatorname{exc}(G)$. First y_1 is determined from the hyperbolic rotation and then y_2 from the Givens rotation, i.e.,

$$y_1 = (x_1 - sx_2)/c, \qquad y_2 = cx_2 - sy_1. \tag{2.7.44}$$

An error analysis of Chambers' algorithm is given by Bojanczyk et al. [34, 1987].

Given $A \in \mathbb{R}^{m\times n}$, $m \geq n$, and $b \in \mathbb{R}^m$, the indefinite least squares (ILS) problem is

$$\min_x (b - Ax)^T J(b - Ax). \tag{2.7.45}$$

A necessary condition for x to solve this problem is that the gradient be zero: $A^T J(b - Ax) = 0$. Equivalently, the residual vector $r = b - Ax$ should be J-orthogonal to the column space $\mathcal{R}(A)$. If $A^T JA$ is positive definite, then there is a unique solution. This implies that $m_1 \geq n$ and that A has full rank. The solution can be computed from the normal equations $A^T JAx = A^T Jb$. It is no restriction to assume that

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad J = \begin{pmatrix} I_{m_1} & 0 \\ 0 & -I_{m_2} \end{pmatrix}, \tag{2.7.46}$$

where $m_1 + m_2 = m$, $m_1 m_2 \neq 0$. Then the normal equations are

$$(A_1^T A_1 - A_2^T A_2)x = A_1^T b_1 - A_2^T b_2.$$

If the problem is ill-conditioned, then the explicit formation of $A^T A$ should be avoided. Assume that the **hyperbolic QR factorization**

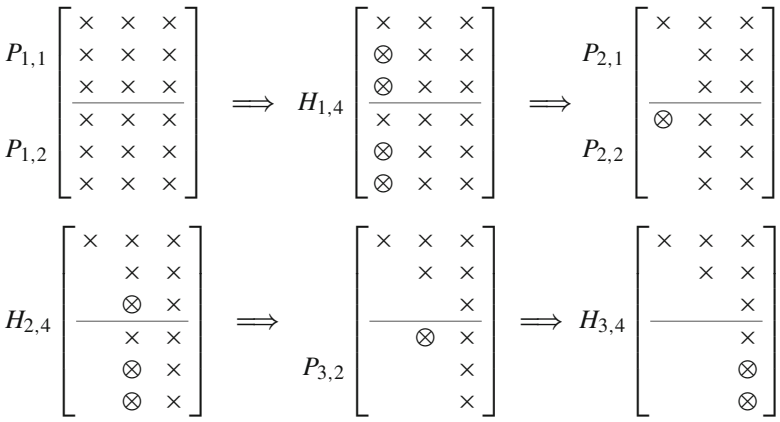
$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad Q^T b = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \tag{2.7.47}$$

where $Q^T JQ = J$ exists. Then

$$\begin{aligned}(b - Ax)^T J(b - Ax) &= (b - Ax)^T QJQ^T(b - Ax) \\ &= \begin{pmatrix} c_1 - Rx \\ c_2 \end{pmatrix}^T J \begin{pmatrix} c_1 - Rx \\ c_2 \end{pmatrix} = \|c_1 - Rx\|_2^2 - \|c_2\|_2^2\end{aligned}$$

and the ILS solution is obtained by solving $Rx = c_1$.

We now describe a hyperbolic QR algorithm due to Bojanczyk, Higham, and Patel [35, 2009]. The algorithm combines Householder reflectors and hyperbolic rotations. In the first step two Householder reflectors are used. The first, $P_{1,1}$, zeros the elements $2{:}m_1$ in the first column of of A_1. The second, $P_{1,2}$, zeros the elements $1{:}m_2$ in A_2. If the problem is positive definite, the remaining element in the first column of A_2 can be zeroed by a hyperbolic rotation in the plane $(1, m_1+1)$. The steps in this reduction to triangular form are shown below for the case $n = m_1 = m_2 = 3$:



The remaining steps are similar. In step k the last $m_1 - k$ elements in the kth column of A_1 are zeroed and the last $m_2 - 1$ elements in the kth column of A_2. A hyperbolic rotation in the plane (k, m_1) is then used to zero the remaining element in the kth column of A_2. If the process does not break down, an upper triangular matrix R is obtained after n steps. In this case the problem must be positive definite. Note that this can be combined with column interchanges so that at each step the diagonal element in R is maximized. It suffices to consider the first step; all remaining steps are similar. If in (2.7.46) $A_1 = (a_1, \ldots, a_n)$, $A_2 = (c_1, \ldots, c_n)$, we use the following modified column pivoting: Let p be the smallest index for which

$$s_p \geq s_j, \qquad s_j = \|a_j\|_2^2 - \|c_j\|_2^2, \quad \forall j = 1{:}n,$$

and interchange columns 1 and p in A and B.

The algorithm uses n hyperbolic rotations for the reduction. Since the operation count for these is $O(n^2)$ flops, the total cost is about the same as for the usual Householder QR factorization. The algorithm has been shown to be forward stable.

A perturbation analysis of the indefinite least squares problem can be obtained as follows. The normal equations can be written in symmetric augmented form as

$$\begin{pmatrix} J & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{2.7.48}$$

The inverse of the augmented matrix is

$$\begin{pmatrix} J & A \\ A^T & 0 \end{pmatrix}^{-1} = \begin{pmatrix} J - JAM^{-1}A^T J & JAM^{-1} \\ M^{-1}A^T J & -M^{-1} \end{pmatrix}, \quad M = A^{TJA}. \tag{2.7.49}$$

This generalizes the corresponding result (2.2.23) for the standard least squares problem. It can be used to show the components-wise perturbation bound

$$|\delta x| \leq \omega\big(|M^{-1}A^T|(f + E|x|) + |M^{-1}|E^T|r|\big), \tag{2.7.50}$$

where $|\delta A| \le \omega E$ and $|\delta b| \le \omega f$.

An early reference to the exchange operator is in network analysis; see the survey by Tsatsomeros [282, 2000]. J-orthogonal matrices also play a role in the solution of certain structured eigenvalue problems; see Sect. 3.7. A systematic study of J-orthogonal matrices and their many applications is given in Higham [163, 2003]. Linear algebra with an indefinite inner product and applications thereof are treated by Gohberg et al. [124, 2005].

2.7.6 Total Least Squares Problems

In the standard linear model (2.1.3) it is assumed that the vector $b \in \mathbb{R}^m$ is related to the unknown parameter vector $x \in \mathbb{R}^n$ by a linear relation $Ax = b + e$, where $A \in \mathbb{R}^{m\times n}$ is an exactly known matrix and e a vector of random errors. If the components of e are uncorrelated and have zero means and the same variance, then by the Gauss–Markov theorem (Theorem 2.1.1) the best linear unbiased estimate of x is given by the solution of the least squares problem

$$\min_x \|r\|_2 \quad \text{subject to} \quad Ax = b + r. \tag{2.7.51}$$

The assumption that all errors are confined to b is frequently unrealistic and sampling or modeling errors will often affect A as well. In the **errors-in-variables model** it is assumed that a linear relation of the form

$$(A + E)x = b + r \tag{2.7.52}$$

holds, where the rows of the error matrix $\begin{pmatrix} E & r \end{pmatrix}$ are *independently and identically distributed with zero mean and the same variance*. If this assumption is not satisfied, it might be possible to find diagonal scaling matrices D_1 and D_2 such that $D_1 \begin{pmatrix} A & b \end{pmatrix} D_2$ satisfies the assumption.

Optimal estimates of the unknown parameters x in this model can satisfy a **total least squares**[14] (TLS) problem

$$\min_{E,\,r} \| \begin{pmatrix} r & E \end{pmatrix} \|_F \quad \text{subject to} \quad (A + E)x = b + r, \tag{2.7.53}$$

where $\|\cdot\|_F$ denotes the Frobenius matrix norm. The constraint in (2.7.53) implies that $b + r \in \mathcal{R}(A + E)$. Thus, 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 $\begin{pmatrix} E & r \end{pmatrix}$ has been found for the problem (2.7.53), then any x satisfying (2.7.52) is said to solve the TLS problem.

[14] The term "total least squares problem" was coined by Golub and Van Loan [132, 1980]. The model has independently been developed in statistics, where it is known as "latent root regression".

The TLS solution will depend on the scaling of the data A and b. In the following we assume that this scaling has been carried out in advance, so that any statistical knowledge of the errors 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**

$$\min_{E,\,r} \| \begin{pmatrix} \gamma r & E \end{pmatrix} \|_F \quad \text{subject to} \quad (A+E)x = b + r.$$

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

$$\min_{E} \|E\|_F \quad \text{subject to} \quad (A+E)x = b, \tag{2.7.54}$$

where it is assumed that the errors in the data are confined to the matrix A.

The constraint in (2.7.53) can be written as

$$\begin{pmatrix} b + r & A + E \end{pmatrix} \begin{pmatrix} -1 \\ x \end{pmatrix} = 0. \tag{2.7.55}$$

This is satisfied if the matrix $\begin{pmatrix} b + r & A + E \end{pmatrix}$ is rank-deficient and $\begin{pmatrix} -1 & x \end{pmatrix}^T$ lies in its null space. Hence, the TLS problem involves finding a perturbation of minimal Frobenius norm that lowers the rank of the matrix $\begin{pmatrix} b & A \end{pmatrix}$.

The TLS problem can be analyzed in terms of the SVD

$$\begin{pmatrix} b & A \end{pmatrix} = U \Sigma V^T = \sum_{i=1}^{n+1} \sigma_i u_i v_i^T, \tag{2.7.56}$$

where $\sigma_1 \geq \cdots \geq \sigma_n \geq \sigma_{n+1} \geq 0$ are the singular values of $\begin{pmatrix} b & A \end{pmatrix}$. If $\sigma_{n+1} = 0$, then the linear system $Ax = b$ is consistent. Otherwise, by Theorem 2.2.11, the unique perturbation of minimum Frobenius norm $\| \begin{pmatrix} r & E \end{pmatrix} \|_F = \sigma_{n+1}$ that makes $(A + E)x = b + r$ consistent is the rank-one perturbation

$$\begin{pmatrix} r & E \end{pmatrix} = -\sigma_{n+1} u_{n+1} v_{n+1}^T, \tag{2.7.57}$$

Multiplying this from the right with v_{n+1} and using (2.7.56) gives

$$\begin{pmatrix} r & E \end{pmatrix} v_{n+1} = -\sigma_{n+1} u_{n+1} = -\begin{pmatrix} b & A \end{pmatrix} v_{n+1}. \tag{2.7.58}$$

It follows that $\begin{pmatrix} b + r & A + E \end{pmatrix} v_{n+1} = 0$ and hence the TLS solution can be expressed in terms of the right singular vector v_{n+1} as

$$v_{n+1} = \begin{pmatrix} \omega \\ y \end{pmatrix}, \quad x = -\omega^{-1} y. \tag{2.7.59}$$

If $\omega = 0$, there is no solution. From (2.7.56) it follows that $\begin{pmatrix} b & A \end{pmatrix}^T U = V \Sigma^T$, and equating the $(n+1)$st columns in the two sides gives

$$\begin{pmatrix} b^T \\ A^T \end{pmatrix} u_{n+1} = \sigma_{n+1} v_{n+1}. \tag{2.7.60}$$

Hence, if $\sigma_{n+1} > 0$, then $\omega = 0$ if and only if $b \perp u_{n+1}$. In this case the TLS problem is called **nongeneric**.

By the minimax characterization of singular values (Theorem 2.2.9), the singular values of $\widehat{\sigma}_i$ of A interlace those of $\begin{pmatrix} b & A \end{pmatrix}$, i.e.,

$$\sigma_1 \geq \widehat{\sigma}_1 \geq \sigma_2 > \cdots \geq \sigma_n \geq \widehat{\sigma}_n \geq \sigma_{n+1}. \tag{2.7.61}$$

If $\widehat{\sigma}_n > \sigma_{n+1}$, then rank $(A) = n$ and by (2.7.61) $\sigma_n > \sigma_{n+1}$. The nongeneric case can only occur when $\widehat{\sigma}_n = \sigma_{n+1}$, since otherwise the TLS problem has a unique solution. The nongeneric case can be treated by adding constraints on the solution; see the discussion in Van Huffel and Vandewalle [286, 1991].

Example 2.7.3 For

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \qquad E = \begin{pmatrix} 0 & 0 \\ 0 & \epsilon \\ 0 & 0 \end{pmatrix}, \tag{2.7.62}$$

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, $\widehat{\sigma}_2 = \sigma_3 = 0$, and $b \perp u_3 = e_3$. □

Let σ_{n+1} be a multiple singular value,

$$\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+1}, \quad p < n,$$

and let $V_2 z$, $z \in \mathbb{R}^{n-p+1}$, be any unit vector in the column subspace of $V_2 = (v_{p+1}, \ldots, v_{n+1})$. Then any vector

$$x = -\omega^{-1} y, \quad V_2 z = \begin{pmatrix} \omega \\ y \end{pmatrix},$$

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 $\omega = e_1^T V_2 z$. Set $z = Q e_1$, where Q is a Householder reflector such that

$$V_2 Q = \begin{pmatrix} \omega & 0 \\ y & \widehat{V}_2 \end{pmatrix}.$$

Then a TLS solution of minimum norm is given by (2.7.59). If $\omega \neq 0$, then there is no solution and the problem is nongeneric. By an argument similar to the case when $p = n$, this can only happen if $b \perp u_j$, $j = p : n$.

We now consider the conditioning of the TLS problem and its relation to the least squares problem. We denote those solutions by x_{TLS} and x_{LS}, respectively. In Sect. 7.1.6 we showed that the SVD of a matrix A is related to the eigenvalue problem for the symmetric matrix $A^T A$. In the generic case the TLS solution can also be characterized by

$$\begin{pmatrix} b^T \\ A^T \end{pmatrix} \begin{pmatrix} b & A \end{pmatrix} \begin{pmatrix} -1 \\ x \end{pmatrix} = \sigma_{n+1}^2 \begin{pmatrix} -1 \\ x \end{pmatrix}, \tag{2.7.63}$$

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" $\begin{pmatrix} b & A \end{pmatrix}$. From the properties of the Rayleigh quotient of symmetric matrices (see Theorem 3.2.12, p. 465) it follows that x_{TLS} minimizes

$$\rho(x) = \frac{(b - Ax)^T (b - Ax)}{x^T x + 1} = \frac{\|b - Ax\|_2^2}{\|x\|_2^2 + 1}, \tag{2.7.64}$$

Thus, whereas the LS solution minimizes $\|b - Ax\|_2^2$, the TLS solution minimizes the "orthogonal distance" function $\rho(x)$ in (2.7.64).

From the last block row of (2.7.63) it follows that

$$(A^T A - \sigma_{n+1}^2 I) x_{\mathrm{TLS}} = A^T b. \tag{2.7.65}$$

The matrix $A^T A - \sigma_{n+1}^2 I$ is symmetric positive definite if $\widehat{\sigma}_n > \sigma_{n+1}$, so this condition ensures that the TLS problem has a unique solution. The system (2.7.65) can be compared to the normal equations

$$A^T A x_{\mathrm{LS}} = A^T b \tag{2.7.66}$$

for the corresponding least squares problem. In (2.7.65) a positive multiple of the unit matrix is *subtracted* from $A^T A$. Thus, TLS can be considered as a *deregularizing* procedure for the least squares problem. (Compare with Tikhonov regularization (see Sect. 2.2.3), where a multiple of the unit matrix is *added* to improve the conditioning of the normal equations.) We conclude that the TLS problem is *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 $A^T A$.

Example 2.7.4 Consider the overdetermined system

$$\begin{pmatrix} \hat{\sigma}_1 & 0 \\ 0 & \hat{\sigma}_2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \beta \end{pmatrix}. \tag{2.7.67}$$

Trivially, the LS solution is $x_{\mathrm{LS}} = (c_1/\hat{\sigma}_1, c_2/\hat{\sigma}_2)^T$, $\|r_{\mathrm{LS}}\|_2 = |\beta|$. If we take $\hat{\sigma}_1 = c_1 = 1$, $\hat{\sigma}_2 = c_2 = 10^{-6}$, then $x_{\mathrm{LS}} = (1,\ 1)^T$ is independent of β, and hence does not reflect the ill-conditioning of A. But the condition number

$$\kappa_{\mathrm{LS}}(A, b) = \kappa(A) \left(1 + \frac{\|r_{\mathrm{LS}}\|_2}{\|\hat{\sigma}_1 x_{\mathrm{LS}}\|_2}\right)$$

will increase proportionally to β. The TLS solution is similar in size to the LS solution as long as $|\beta| \leq \hat{\sigma}_2$. For $\beta > \hat{\sigma}_2$ the condition number κ_{TLS} and $\|x_{\mathrm{TLS}}\|_2$ grow proportionally to β^2. Setting $c_1 = c_2 = 0$ gives $x_{\mathrm{LS}} = 0$. If $|\beta| \geq \sigma_2(A)$, then $\sigma_2(A) = \sigma_3(A, b)$ and the TLS problem is nongeneric. □

The TLS problem can also be posed as an indefinite least squares problem and minimizes the function

$$\|b - Ax\|_2^2 - \sigma_{n+1}^2 \|x\|_2^2 = s(x)^T \begin{pmatrix} I_m & 0 \\ 0 & -I_n \end{pmatrix} s(x), \tag{2.7.68}$$

where

$$s(x) = \begin{pmatrix} b \\ 0 \end{pmatrix} - \begin{pmatrix} A \\ \sigma_{n+1} I_n \end{pmatrix} x. \tag{2.7.69}$$

The first step toward the solution of the TLS problem is to use the Golub–Kahan algorithm to reduce the matrix $\begin{pmatrix} b & A \end{pmatrix}$ to bidiagonal form. This determines orthogonal matrices U and V such that

$$U^T \begin{pmatrix} b & AV \end{pmatrix} = \begin{pmatrix} \beta_1 e_1 & B_n \\ 0 & 0 \end{pmatrix}, \tag{2.7.70}$$

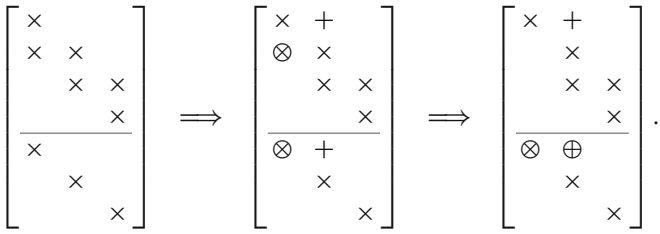
where $B_n \in \mathbb{R}^{(n+1)\times n}$ is lower bidiagonal. For simplicity, we assume that a solution exists and is unique. The solution is then obtained from the right singular vector corresponding to the smallest singular value σ_{n+1} of the bidiagonal matrix $\begin{pmatrix} \beta_1 e_1 & B_n \end{pmatrix}$. There are several ways to compute this singular vector. The high cost of computing the full SVD should be avoided. Van Huffel [285, 1990] gives an algorithm for computing a partial SVD for solving TLS problems. Another possibility is to apply inverse iteration to the bidiagonal matrix. The cost of one inverse iteration is only $6n$ flops; see Sect. 2.3.3.

Example 2.7.5 If σ_{n+1} has been determined, it remains to solve the indefinite least squares problem (2.7.68)–(2.7.69). Setting $x = Vy$ and using the invariance of the

Euclidean norm, we see that this is equivalent to minimizing

$$\|\beta_1 e_1 - By\|_2^2 - \sigma_{n+1}^2 \|y\|_2^2.$$

Because of the special structure of this problem, the method using hyperbolic rotation simplifies. The first step in the transformation to upper bidiagonal form is pictured here:



In the first step a Givens rotation of rows 1 and 2 is used to zero element $(2,1)$, and then a hyperbolic rotation of rows 1 and 5 to zero element $(5,1)$. Next, a Givens rotation in rows 5 and 6 is used to zero the fill-in element $(5,2)$. The reduction can then continue on a reduced problem of the same structure.

The transformations of the upper part are also applied to the right-hand side. The solution is finally obtained by solving an upper bidiagonal linear system. The cost of the reduction and solution is about $21n$ flops, which is the same as for a similar regularization algorithm given by Eldén [85, 1977]. The total cost for obtaining the TLS solution is dominated by the cost of the initial bidiagonalization, which is $4(mn^2 - n^3/3)$ flops. Note that the bidiagonalization (2.7.70) can terminate prematurely. In particular, this step will reveal if the system $Ax = b$ is consistent; cf. the PLS algorithm in Sect. 2.6.3. □

We now consider the more general TLS problem with $d > 1$ right-hand sides:

$$\min_{E,\ F} \|(E,\ F)\|_F, \qquad (A+E)X = B+F, \tag{2.7.71}$$

where $B \in \mathbb{R}^{m\times d}$. The consistency relations can be written

$$(B+F,\ A+E)\begin{pmatrix} -I_d \\ X \end{pmatrix} = 0.$$

We now seek perturbations $(F,\ E)$ that reduce the rank of the matrix $(B,\ A)$ by d. We call this a **multidimensional** TLS problem. As remarked before, for this problem to be meaningful the rows of the matrix $(B+F,\ A+E)$ 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 improved predictive power of the TLS solution.

The solution to the TLS problem with multiple right-hand sides can be expressed in terms of the SVD

$$\begin{pmatrix} B & A \end{pmatrix} = U\Sigma V^T = U_1\Sigma_1 V_1^T + U_2\Sigma_2 V_2^T, \tag{2.7.72}$$

where $\Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$, $\Sigma_2 = \operatorname{diag}(\sigma_{n+1}, \ldots, \sigma_{n+d})$, and U and V partitioned conformally with $\begin{pmatrix} B & A \end{pmatrix}$. If $\sigma_n > \sigma_{n+1}$, the minimizing perturbation is unique and given by the rank-d matrix

$$\begin{pmatrix} F & E \end{pmatrix} = -U_2\Sigma_2 V_2^T = -\begin{pmatrix} B & A \end{pmatrix} V_2 V_2^T,$$

for which $\| \begin{pmatrix} F & E \end{pmatrix} \|_F = \sum_{j=1}^{d} \sigma_{n+j}^2$ and $\begin{pmatrix} B+F & A+E \end{pmatrix} V_2 = 0$. Assume that

$$V_2 = \begin{pmatrix} V_{12} \\ V_{22} \end{pmatrix}$$

with $V_{12} \in \mathbb{R}^{d\times d}$ nonsingular. Then the solution to the TLS problem is unique:

$$X = -V_{22}V_{12}^{-1} \in \mathbb{R}^{n\times d}.$$

We show that if $\sigma_n(A) > \sigma_{n+1}\begin{pmatrix} B & A \end{pmatrix}$, then V_{12} is nonsingular. From (2.7.72) 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 and hence $U_2\Sigma_2 x = AV_{12}x$. 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 $\|V_{22}x\|_2 = 1$. But then

$$\sigma_{n+1}\begin{pmatrix} B & A \end{pmatrix} \geq \|U_2\Sigma_2 x\|_2 = \|AV_{12}x\|_2 \geq \sigma_n(A),$$

a contradiction. Hence, V_{12} is nonsingular.

In many parameter estimation problems, some of the columns are known exactly. It is no loss of generality 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

$$\begin{pmatrix} A_1 & A_2 + E_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = B + F, \qquad A_1 \in \mathbb{R}^{m\times n_1},$$

where $A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \in \mathbb{R}^{m\times n}$, $n = n_1 + n_2$. It is assumed that the rows of the errors $\begin{pmatrix} E_2 & F \end{pmatrix}$ are independently and identically distributed with zero mean and the same variance. The mixed LS–TLS problem can then be expressed as

$$\min_{E_2, F} \| \begin{pmatrix} E_2 & F \end{pmatrix} \|_F, \qquad \begin{pmatrix} A_1 & A_2 + E_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = B + F. \tag{2.7.73}$$

When A_2 is empty, this reduces to solving an ordinary least squares problem with multiple right-hand sides. When A_1 is empty, this is the standard TLS problem. Hence, this mixed problem includes both extreme cases.

Let $A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \in \mathbb{R}^{m\times n}$, $n = n_1 + n_2$, $m \geq n$, and $B \in \mathbb{R}^{m\times d}$. Assume that the columns of A_1 are linearly independent. Then the mixed LS–TLS problem can be solved as follows. First compute the QR factorization

$$\begin{pmatrix} A_1 & A_2 \end{pmatrix} = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \qquad B = Q \begin{pmatrix} C_1 \\ C_2 \end{pmatrix},$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal, $R_{11} \in \mathbb{R}^{n_1\times n_1}$ is upper triangular, and $R_{22} \in \mathbb{R}^{(m-n_1)\times n_2}$. If $n_2 = 0$, then the solution is obtained from $R_{11}X = C_1$. Otherwise, compute X_2 as the solution to the TLS problem.

$$\min_{E,G} \| \begin{pmatrix} E & G \end{pmatrix} \|_F, \qquad (R_{22} + E)X = C_2 + G. \tag{2.7.74}$$

Finally, X_1 is obtained by solving the triangular system

$$R_{11}X_1 = C_1 - R_{12}X_2. \tag{2.7.75}$$

For a full discussion of the details in the algorithm, see Van Huffel and Vandewalle [286, 1991], Sect. 3.6.3.

The term "total least squares problem" coined by Golub and Van Loan [132, 1980] renewed the interest in the "errors in variable model". A rigorous treatment of the TLS problem is given by Van Huffel and Vandewalle [286, 1991]. They outline the partial SVD (PSVD) algorithm for computing the left/right singular subspaces associated with smallest singular values. A Fortran 77 implementation of this algorithm is available from Netlib. The important role of the core problem for weighted TLS problems was discovered by Paige and Strakoš [228, 2006].

2.7.7 Linear Orthogonal Regression

Let P_i, $i = 1{:}m$, be a set of given points in $\mathbb{R}^n$. In the linear **orthogonal regression** problem we want to fit a hyperplane 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

$$c_1 x + c_2 y + d = 0, \tag{2.7.76}$$

where $c_1^2 + c_2^2 = 1$. Then the orthogonal distance from the point $P_i = (x_i, y_i)$ to the line is $r_i = c_1 x_i + c_2 y_i + d$. Thus, we want to minimize

$$\sum_{i=1}^{m}(c_1x_i + c_2y_i + d)^2, \tag{2.7.77}$$

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

$$\begin{pmatrix} e & Y \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \vdots & \vdots & \\ 1 & x_m & y_m \end{pmatrix}, \quad c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}.$$

By computing the QR factorization of $\begin{pmatrix} e & Y \end{pmatrix}$ and using the invariance of the Euclidean norm, we can reduce the problem 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, d can always be chosen so that $r_{11}d + r_{12}c_1 + r_{13}c_2 = 0$. 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 2.2.7), the solution is the right singular vector corresponding to the smallest singular value of B. Let the SVD be

$$B = \begin{pmatrix} r_{21} & r_{22} \\ 0 & r_{33} \end{pmatrix} = (u_1\ u_2) \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \end{pmatrix},$$

where $\sigma_1 \geq \sigma_2 \geq 0$. (A stable algorithm for computing the SVD of a two by two upper triangular matrix is given in Sect. 3.6.3.) 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 Gander and Hřebiček [109, 2004], Chap. 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 $P_i \in \mathbb{R}^n$ to a hyperplane M so that the sum of squares of the orthogonal distances is minimized. The equation for the hyperplane can be written

$$c^T z = d, \quad z, c \in \mathbb{R}^n, \quad \|c\|_2 = 1,$$

where $c \in \mathbb{R}^n$ is the unit normal vector of M, and $|d|$ is the orthogonal distance from the origin to the plane. Then the orthogonal projection z_i of the point y_i onto M is given by

$$z_i = y_i - (c^T y_i - d)c. \tag{2.7.78}$$

It is readily verified that z_i lies on M and that 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 $Y^T = (y_1, \ldots, y_m) \in \mathbb{R}^{n\times m}$ and $e = (1, \ldots, 1)^T \in \mathbb{R}^m$, this problem can be written in matrix form as

$$\min_{c,d} \left\| \begin{pmatrix} -e & Y \end{pmatrix} \begin{pmatrix} d \\ c \end{pmatrix} \right\|_2 \quad \text{subject to} \quad \|c\|_2 = 1. \tag{2.7.79}$$

For a fixed c, this expression is minimized when the residual vector $Yc - de$ is orthogonal to e, i.e., when $e^T(Yc - de) = e^T Yc - de^T e = 0$. Since $e^T e = m$, it follows that

$$d = \frac{1}{m} c^T Y^T e = c^T \bar{y}, \qquad \bar{y} = \frac{1}{m} Y^T e, \tag{2.7.80}$$

where $\bar{y}$ is the mean of the given points y_i. Hence, d is determined by the condition that the mean $\bar{y}$ lies on the optimal plane M. Note that this property is shared by the solution to the usual linear regression problem.

We now subtract $\bar{y}$ 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 (2.7.80)

$$\begin{pmatrix} -e & Y \end{pmatrix} \begin{pmatrix} d \\ c \end{pmatrix} = Yc - e\bar{y}^T c = (Y - e\bar{y}^T)c = \bar{Y}c,$$

problem (2.7.79) is equivalent to $\min_c \|\overline{Y}c\|_2$ subject to $\|c\|_2 = 1$. By the min-max characterization of the singular values (Theorem 2.2.7), a solution is $c = v_n$, where

v_n is a right singular vector of $\overline{Y}$ 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 is σ_n^2. Further, $\sigma_n = 0$ if and only if the given points $y_i, i = 1{:}m$, all lie on the hyperplane M. In the extreme case, all points coincide. Then $\bar{Y} = 0$, and any plane going through $\bar{y}$ is a solution.

The above method solves the problem of fitting an $(n-1)$-dimensional linear manifold to a given set of points in $\mathbb{R}^n$. It is readily generalized to the fitting of an $(n-p)$-dimensional linear manifold by orthogonalizing the shifted points y against the p right singular vectors of Y corresponding to p smallest singular values.

2.7.8 The Orthogonal Procrustes Problem

Let A and B be given matrices in $\mathbb{R}^{m\times n}$. The **orthogonal Procrustes problem**[15] is

$$\min_{Q} \|A - BQ\|_F \quad \text{subject to} \quad Q^T Q = I. \tag{2.7.81}$$

The solution to this problem can be computed from the polar decomposition of $B^T A$ (see Theorem 2.2.12, p. 239) as shown by the following generalization of Theorem 2.2.13.

Theorem 2.7.4 (Schönemann [252, 1966]) *Let $\mathcal{M}_{m\times n}$ denote the set of all matrices in $\mathbb{R}^{m\times n}$ with orthogonal columns. Let A and B be given matrices in $\mathbb{R}^{m\times n}$ such that* rank $(B^T A) = n$. *Then*

$$\|A - BQ\|_F \geq \|A - BP\|_F$$

for any matrix $Q \in \mathcal{M}_{m\times n}$, where $B^T A = PH$ is the polar decomposition.

Proof Recall from (1.1.67) that $\|A\|_F^2 = \text{trace}(A^T A)$ and that $\text{trace}(X^T Y) = \text{trace}(YX^T)$. Using this and the orthogonality of Q, we find that

[15] Procrustes was a giant of Attica in Greece, who seized travelers, tied them to an iron bedstead, and either stretched them or chopped off their legs to make them fit it.

$$\|A - BQ\|_F^2 = \operatorname{trace}(A^T A) + \operatorname{trace}(B^T B) - 2\operatorname{trace}(Q^T B^T A).$$

It follows that the problem (2.7.81) is equivalent to maximizing $\operatorname{trace}(Q^T B^T A)$. Let the SVD of $B^T A$ be $B^T A = U\Sigma V^T$ and set $Q = UZV^T$, where Z is orthogonal. Then $\|Z\|_2 = 1$ and hence the diagonal elements of Z must satisfy $|z_{ii}| \le 1, i = 1{:}n$. Hence,

$$\begin{aligned}\operatorname{trace}(Q^T B^T A) &= \operatorname{trace}(VZ^T U^T B^T A) = \operatorname{trace}(Z^T U^T B^T AV) \\ &= \operatorname{trace}(Z^T \Sigma) = \sum_{i=1}^{n} z_{ii}\sigma_i \le \sum_{i=1}^{n} \sigma_i,\end{aligned}$$

where $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$. The upper bound is obtained for $Q = UV^T$. If $\operatorname{rank}(A) = n$, this solution is unique. □

In many applications it is important that Q corresponds to a pure rotation, i.e., $\det(Q) = 1$. If $\det(UV^T) = 1$, the optimal is $Q = UV^T$ as before. Otherwise, if $\det(UV^T) = -1$, the optimal solution can be shown to be (see [156, 1981])

$$Q = UZV^T, \quad Z = \operatorname{diag}(1, \ldots, 1, -1),$$

with $\det(Q) = +1$. For this choice,

$$\sum_{i=1}^{n} z_{ii}\sigma_i = \operatorname{trace}(\Sigma) - 2\sigma_n.$$

In both cases the optimal solution can be written as

$$Q = UZV^T, \quad Z = \operatorname{diag}(1, \ldots, 1, \det(UV^T)).$$

The analysis of rigid body motions involves also a translation vector $c \in \mathbb{R}^n$. Then, we have the model $A = BQ + ec^T$, $e = (1, 1, \ldots, 1)^T \in \mathbb{R}^m$. To estimate also $c \in \mathbb{R}^n$ we solve the problem

$$\min_{Q,c} \|A - BQ - ec^T\|_F \quad \text{subject to} \quad Q^T Q = I, \quad \det(Q) = 1. \tag{2.7.82}$$

For any Q, including the optimal Q not yet known, the best least squares estimate of c is characterized by the condition that the residual be 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 is

$$c = \frac{1}{m}((B^T e)Q - A^T e). \tag{2.7.83}$$

Substituting this expression into (2.7.82), we can eliminate c and the problem becomes $\min_Q \|\widetilde{A} - \widetilde{B}Q\|_F$, where

$$\widetilde{A} = A - \frac{1}{m}ee^T A, \qquad \widetilde{B} = B - \frac{1}{m}ee^T B.$$

This is now a standard orthogonal Procrustes problem and the solution is obtained from the SVD of $\widetilde{A}^T\widetilde{B}$.

If A is close to an orthogonal matrix, an iterative method for computing the polar decomposition can be used. Such methods are developed in Sect. 3.8.1. A perturbation analysis of the orthogonal Procrustes problem is given by Söderkvist [260, 1993].

The orthogonal Procrustes problem arises, e.g., in factor analysis in statistics. A large-scale application occurs in calculations of subspace alignment in molecular dynamics simulation of electronic structure. Another application is in determining rigid body movements, which has important applications in radio-stereometric analysis; see Söderkvist and Wedin [261, 1993]. Let $A = (a_1, \ldots, a_m)$ be measured positions of $m \geq n$ landmarks of a rigid body in $\mathbb{R}^n$ and $B = (b_1, \ldots, b_m)$ be the measured positions after the body has been rotated. An orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ is desired, which represents the rotation of the body; see Söderkvist and Wedin [262, 1994]. Eldén and Park [91, 1999] study a more general unbalanced Procrustes problem, where the number of columns in A and B differs and Q is rectangular.

Exercises

2.7.1 Work out the details of Cline's variant of the Peters-Wilkinson method. Use Householder transformations for the orthogonal factorization in (2.7.6). Compare the operation count and storage requirement with the original method.

2.7.2 Prove that the exchange operator satisfies $\mathrm{exc}\,(\mathrm{exc}\,(Q)) = Q$.

2.7.3 (Stewart and Stewart [274, 1998]) If properly implemented, hyperbolic Householder reflectors have the same good stability as the mixed scheme of hyperbolic rotations.

(a) Show that the hyperbolic rotations H can be rewritten as

$$H = \frac{1}{c}\begin{pmatrix} 1 & -s \\ -s & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + \frac{1}{c}\begin{pmatrix} t \\ -s/t \end{pmatrix}\begin{pmatrix} t & -s/t \end{pmatrix}, \quad t = \sqrt{1-c},$$

which now has the form of a hyperbolic Householder reflector. If H is J-orthogonal, so is

$$JH = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \frac{1}{c}\begin{pmatrix} t \\ s/t \end{pmatrix}\begin{pmatrix} t & -s/t \end{pmatrix}, \quad t = \sqrt{1-c}.$$

(b) Show that the transformation can be computed from

$$SH\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \gamma\begin{pmatrix} 1 \\ s/(1-c) \end{pmatrix}, \qquad \gamma = \frac{1}{c}((1-c)x - sy).$$

2.7.4 Consider a TLS problem where $n = 1$ and

$$C = (A,\ b) = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}.$$

The unique ΔC lowering the rank satisfies

$$C + \Delta C = (A + E, b + r) = \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix},$$

so the perturbed system is not compatible. Show that an arbitrary small perturbation ϵ in the (2,1) element will give a compatible system with solution $x = 2/\epsilon$.

2.7.5 (Gander and Hřebiček [109, 2004])

(a) Write a MATLAB program for fitting a straight line $c_1 x + c_2 y = d$ to given points $(x_i, y_i) \in \mathbb{R}^2$, $i = 1{:}m$, so that the sum of orthogonal distances is minimized. The program should handle all exceptional cases, such as $c_1 = 0$ and/or $c_2 = 0$.

(b) Suppose we want to fit two set of points $(x_i, y_i) \in \mathbb{R}^2$, $i = 1{:}p$ and $i = p + 1{:}m$, 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 is 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.

2.8 Nonlinear Least Squares Problems

Nonlinear least squares problems are common in science and engineering. A classical problem is fitting a sum of real exponential functions $c_j e^{\lambda_j t}$ with c_j and λ_j unknown. This arises, e.g., in radioactive decay, compartment models, and atmospheric transfer functions. A recent large-scale application is the construction of three-dimensional models from photographs as in Google's street view sensor fusion. For this Google developed their own nonlinear least squares Ceres solver.

The nonlinear least squares problem is a simple special case of the optimization problem to minimize a convex[16] **objective function** ϕ, $\mathbb{R}^n \to \mathbb{R}$. In practice the parameters may be restricted to lie in some convex subset of $\mathbb{R}^n$, but only methods for unconstrained problems will be considered here. In the nonlinear least squares problem the objective function has the special form

$$\min_{x \in \mathbf{R}^n} \phi(x), \quad \phi(x) = \tfrac{1}{2}\|r(x)\|_2^2 = \tfrac{1}{2} r(x)^T r(x), \tag{2.8.1}$$

and $\phi\colon \mathbb{R}^n \to \mathbb{R}^m$, $m \geq n$. When fitting observations (y_i, t_i), $i = 1{:}m$, to a model function $y = h(x, t)$, the error in the model prediction for the ith observation is

$$r_i(x) = y_i - h(x, t_i), \quad i = 1{:}m.$$

The choice of the least squares measure is justified, as in the linear case, by statistical considerations. If the observations have equal weight, this leads to the minimization problem (2.8.1).

[16] A function $\phi(x)$, $x \in \mathbb{R}^n$, is convex if $\phi(\theta x + (1-\theta)y) \leq \theta\phi(x) + (1-\theta)\phi(y)$, $0 \leq \theta \leq 1$.

2.8.1 Conditions for a Local Minimum

A point x^* is said to be a **local minimizer** of ϕ if $\phi(x^*) \leq \phi(y)$ for all y in a sufficiently small neighborhood of x^*. If $\phi(x^*) < \phi(y)$ for $y \neq x^*$, then x^* is a **strong** local minimizer. In the following we assume that the ϕ is twice continuously differentiable. The **gradient** of ϕ at x is the row vector

$$g(x) = (g_1(x), \ldots, g_n(x)), \quad g_i(x) = \frac{\partial \phi}{\partial x_i}, \tag{2.8.2}$$

and is normal to the tangent hyperplane of $\phi(x)$. A *necessary condition* for x^* to be a local minimizer is $g(x^*) = 0$. Then x^* is called a **stationary point** of ϕ.

It is possible for a stationary point to be neither a maximizer nor a minimizer. Such a point is called a **saddle point**. To determine if a stationary point is a local minimizer, information about the second-order partial derivatives of $\phi(x)$ is needed. These form an $n \times n$ matrix called the **Hessian**:

$$H(x) = \begin{pmatrix} h_{11} & \ldots & h_{1n} \\ \vdots & & \vdots \\ h_{n1} & \ldots & h_{nn} \end{pmatrix} \in \mathbb{R}^{n\times n}, \quad h_{ij} = \frac{\partial^2 \phi}{\partial x_i \partial x_j}. \tag{2.8.3}$$

If the Hessian exists and is continuous, then it is symmetric. The Hessian can be looked upon as the derivative of the gradient.

Theorem 2.8.1 *Necessary conditions for x^* to be a local minimizer of ϕ is that x^* is a stationary point, i.e., $g(x^*) = 0$, and that $H(x^*)$ is positive semidefinite. If $g(x^*) = 0$ and $H(x^*)$ is positive definite, then x^* is a strong local minimizer.*

Proof The Taylor-series expansion of ϕ about x^* is

$$\phi(x^* + \epsilon d) = \phi(x^*) + \epsilon d^T g(x^*) + \frac{1}{2}\epsilon^2 d^T H(x^* + \epsilon\theta d)d,$$

where $0 \leq \theta \leq 1$, ϵ is a scalar, and d a vector. Assume that $g(x^*) \neq 0$ and choose d so that $d^T g(x^*) < 0$. Then for sufficiently small $\epsilon > 0$ the last term is negligible and $\phi(x^* + \epsilon d) < \phi(x^*)$. □

For the nonlinear least squares problems, assume that $r(x)$ is twice continuously differentiable. Then it is easily shown that the gradient of $\phi(x) = \frac{1}{2} r^T(x) r(x)$ is given by

$$g(x)^T = \nabla \phi(x) = J(x)^T r(x), \tag{2.8.4}$$

where $J(x) \in \mathbb{R}^{m\times n}$ is the **Jacobian**, with elements

$$J(x)_{ij} = \frac{\partial r_i(x)}{\partial x_j} \quad i = 1{:}m, \quad j = 1{:}n. \tag{2.8.5}$$

A necessary condition for x^* to be a local minimizer of $\phi(x)$ is that $g(x^*)^T = J(x^*)^T r(x^*) = 0$ (cf. the normal equations). The Hessian matrix is

$$H(x) = \nabla^2 \phi(x) = J(x)^T J(x) + Q(x), \quad Q(x) = \sum_{i=1}^{m} r_i(x) G_i(x), \tag{2.8.6}$$

where $G_i(x) \in \mathbb{R}^{n\times n}$ is the Hessian of $r_i(x)$, with elements

$$G_i(x)_{jk} = \frac{\partial^2 r_i(x)}{\partial x_j \partial x_k}, \quad i = 1{:}m, \quad j,k = 1{:}n. \tag{2.8.7}$$

The special forms of the gradient $g(x)$ and Hessian $H(x)$ can be exploited by methods for the nonlinear least squares problem. This is the main reason for studying such problems separately from more general minimization problems.

2.8.2 Newton and Gauss–Newton Methods

There are two main ways to view problem (2.8.1). In **Gauss–Newton** methods one thinks of the problem as arising from an overdetermined system of nonlinear equations $r(x) = 0$. It is then natural to use a linear model

$$\widetilde{r}(x) = r(x_k) + J(x_k)(x - x_k) \tag{2.8.8}$$

around a given approximate solution $x_k \in \mathbb{R}^n$. The solution p_k to the linear least squares problem

$$\min_p \|r(x_k) + J(x_k)p_k\|_2 \tag{2.8.9}$$

is used to derive a new (ideally improved) solution $x_{k+1} = x_k + p_k$. As the name implies, the Gauss–Newton method was used by Gauss. This method has in general only linear rate of convergence.

In the second approach, (2.8.1) is viewed as a special case of unconstrained optimization in $\mathbb{R}^n$. At a point x_k, a quadratic model of ϕ is used:

$$\widetilde{\phi}_{\mathrm{q}}(x) = \phi(x_k) + g(x_k)^T(x - x_k) + \frac{1}{2}(x - x_k)^T H(x_k)(x - x_k). \tag{2.8.10}$$

The gradient and Hessian of $\phi(x) = \frac{1}{2} r^T(x) r(x)$ are given by (2.8.4) and (2.8.6). The minimizer of $\widetilde{\phi}_{\mathrm{q}}(x)$ is given by $x_{k+1} = x_k + p_k$, where

$$H(x_k)p_k = -J(x_k)^T r(x_k). \tag{2.8.11}$$

This method is equivalent to Newton's method applied to (2.8.1). It is quadratically convergent to a local minimizer x^* as long as $H(x)$ is Lipschitz continuous around x_k and $H(x^*)$ is positive definite (see Dennis and Schnabel [72, 1983], p. 229).

The Gauss–Newton method can be thought of as arising from neglecting the second-derivative term $Q(x)$ in the Hessian (2.8.6). Note that $Q(x_k)$ will be small close to the solution x^* if either the residual norm $\|r(x^*)\|$ is small, or $r(x)$ is only mildly nonlinear. The behavior of the Gauss–Newton method can then be expected to be similar to that of Newton's method. In particular, for a consistent problem where $r(x^*) = 0$, the local convergence will be the same for both methods. But for moderate to large residual problems, the local convergence rate for the Gauss–Newton method can be much inferior to that of Newton's method.

Let x_k denote the current approximation in Gauss–Newton's method. (Note that here and in the following, k denotes the iteration index and not a component of a vector.) Then d_k is a solution to the linear least squares problem

$$\min_{d_k} \|r(x_k) + J(x_k)d_k\|_2, \quad d_k \in \mathbb{R}^n, \tag{2.8.12}$$

and the new approximation is $x_{k+1} = x_k + d_k$. The solution d_k is unique if $\text{rank}(J(x_k)) = n$. Since $J(x_k)$ may be ill-conditioned or singular, d_k should be computed by a stable method using, e.g., the QR factorization or SVD of $J(x_k)$. The Gauss–Newton step $d_k = -J(x_k)^\dagger r(x_k)$ has the following important properties:

(i) d_k is invariant under linear transformations of the independent variable x, i.e., if $\widetilde{x} = Sx$, S nonsingular, then $\widetilde{d}_k = Sd_k$.
(ii) if $J(x_k)^T r(x_k) \neq 0$, then d_k is a descent direction for $\phi(x) = \frac{1}{2} r(x)^T r(x)$.

The first property is easily verified. To prove the second property, we note that

$$d_k^T g(x_k) = -r(x_k)^T J^\dagger(x_k)^T J(x_k)^T r(x_k) = -\|P_{J_k} r(x_k)\|_2^2, \tag{2.8.13}$$

where $P_{J_k} = J(x_k)J^\dagger(x_k) = P_{J_k}^2$ is the orthogonal projection onto the range space of $J(x_k)$. Further, if $J(x_k)^T r(x_k) \neq 0$, then $r(x_k)$ is not in the null space of $J(x_k)^T$ and it follows that $P_{J_k} r(x_k) \neq 0$. This proves (ii).

The Gauss–Newton method can fail at an intermediate point where the Jacobian is rank-deficient or ill-conditioned. Formally, we can take d_k to be the minimum-norm solution

$$d_k = -J(x_k)^\dagger r(x_k).$$

In practice it is necessary to include some strategy to estimate the numerical rank of $J(x_k)$, cf. Sects. 2.4.1 and 2.2.3. The assigned rank can have a decisive influence. Usually it is preferable to *underestimate* the rank, except when $\phi(x)$ is close to an ill-conditioned quadratic function. One could also switch to a search direction along the negative gradient.

The geometrical interpretation of the nonlinear least squares problem (2.8.1) is to find a point on the surface $\{r(x) \mid x \in \mathbb{R}^n\}$ in $\mathbb{R}^m$ closest to the origin. In case of

data fitting $r_i(x) = y_i - h(x, t_i)$, it is more illustrative to consider the surface

$$z(x) = (h(x, t_1), \ldots, h(x, t_m))^T \in \mathbb{R}^m.$$

The solution (if it exists) is given by an orthogonal projection of y onto the surface $z(x)$. The rate of convergence of Gauss–Newton type methods can be analyzed using differential-geometric concepts, as first suggested by Wedin [293, 1974]. Let $J^\dagger(x)$ denote the pseudoinverse of $J(x)$ and assume that rank $(J(x)) = n$. Then $J^\dagger(x)J(x) = I$, and the Hessian can be written in the form

$$H(x) = J(x)^T(I - \gamma K(x))J(x), \quad K(x) = J^\dagger(x)^T G_w(x) J^\dagger(x), \tag{2.8.14}$$

where $\gamma = \|r(x)\|_2 \neq 0$ and

$$G_w(x) = \sum_{i=1}^{m} w_i G_i(x), \quad w(x) = -\frac{1}{\gamma} r(x). \tag{2.8.15}$$

The matrix $K(x)$ is symmetric and has a geometric interpretation. It is the **normal curvature matrix** of the n-dimensional surface $z(x)$ in $\mathbb{R}^m$ with respect to the unit normal vector $w(x)$. The quantities $\rho_i = 1/\kappa_i$, where

$$\kappa_1 \geq \kappa_2 \geq \cdots \geq \kappa_n,$$

are the nonzero eigenvalues of $K(x)$, are called the **principal radii of curvature** of the surface.

The Hessian $H(x^*)$ is positive definite and x^* a local minimizer if and only if $u^T H(x^*)u > 0$ for all $u \in \mathbb{R}^n$. It follows that $u \neq 0 \Rightarrow J(x^*)u \neq 0$, and hence $H(x^*)$ is positive definite when $I - \gamma K(x^*)$ is positive definite, i.e., when $1 - \gamma\kappa_1 > 0$. It can be shown that the asymptotic rate of convergence of the Gauss–Newton method in the neighborhood of a stationary point x^* is

$$\rho = \gamma \max(\kappa_1, -\kappa_n), \tag{2.8.16}$$

where κ_i are the eigenvalues of the normal curvature matrix $K(x)$ in (2.8.14) evaluated at x^*, and $\gamma = \|r(x^*)\|_2$. In general convergence is linear, but if $\gamma = 0$, then convergence becomes superlinear. Hence, the asymptotic rate of convergence of the Gauss–Newton method is fast when either

(i) the residual norm $\gamma = \|r(x^*)\|_2$ is small, or
(ii) $r(x)$ is mildly nonlinear, i.e., $|\kappa_i|$, $i = 1{:}n$, are small.

If $1 - \gamma\kappa_1 \leq 0$, then the least squares problem has a saddle point at x^*, or if also $1 - \gamma\kappa_n < 0$, even a local maximum at x^*. If x^* is a saddle point, then the Gauss–Newton method is repelled from a saddle point. This is an excellent property, because saddle points are not at all uncommon for nonlinear least squares problems.

The rate of convergence for the Gauss–Newton method can be estimated during the iterations by

$$\rho_{\text{est}} = \|P_J(x_{k+1})r_{k+1}\|_2 / \|P_J(x_k)r_k\|_2 = \rho + O(\|x_k - x^*\|_2^2). \tag{2.8.17}$$

Since $P_J(x_k)r_k = J(x_k)J(x_k)^\dagger r_k = -J(x_k)p_k$, the cost of computing this estimate is only one matrix-vector multiplication. When $\rho_{\text{est}} > 0.5$ (say), one should consider switching to a method using second derivative information, or perhaps evaluate the quality of the underlying model.

2.8.3 Modifications for Global Convergence

The Gauss–Newton method can be modified for global convergence by using the Gauss–Newton direction d_k as a search direction. The next iteration is then taken to be $x_{k+1} = x_k + t_k d_k$, where t_k solves the one-dimensional minimization problem

$$\min_t \|r(x_k + td_k)\|_2^2.$$

In general, it is not worthwhile solving this minimization accurately. It suffices taking t_k can be taken to be the largest number in the sequence $1, \frac{1}{2}, \frac{1}{4}, \ldots$ for which

$$\|r(x_k)\|_2^2 - \|r(x_k + t_k d_k)\|_2^2 \geq \frac{1}{2} t_k \|P_{J_k} r(x_k)\|_2^2.$$

Here $t = 1$ corresponds to the full Gauss–Newton step. Since d_k is a descent direction, this modification of the Gauss–Newton method is locally convergent in almost all nonlinear least squares problems. In fact, it is usually even globally convergent. For large residual or very nonlinear problems, convergence may still be slow.

The rate of convergence for the Gauss–Newton method with *exact* line search can be shown to be

$$\widetilde{\rho} = \gamma(\kappa_1 - \kappa_n)/(2 - \gamma(\kappa_1 + \kappa_n)).$$

We have $\widetilde{\rho} = \rho$ if $\kappa_n = -\kappa_1$, and $\widetilde{\rho} < \rho$ otherwise. Since $\gamma\kappa_1 < 1$ implies $\widetilde{\rho} < 1$, we always get convergence close to a local minimizer. This contrasts to the unmodified Gauss–Newton method, which may fail to converge to a local minimizer.

Even with line search the Gauss–Newton method can have difficulties getting around an intermediate point where the Jacobian matrix is rank-deficient. This can be avoided either by taking second derivatives into account, or by further stabilizing the Gauss–Newton method to overcome this possibility of failure. Methods using the latter approach were first suggested by Levenberg [193, 1944] and Marquardt [205, 1963]. Here the search direction d_k is computed from the problem (cf. Tikhonov regularization)

$$\min_{d_k} \left\{ \|r(x_k) + J(x_k)d_k\|_2^2 + \mu_k \|d_k\|_2^2 \right\}, \tag{2.8.18}$$

where the parameter $\mu_k \geq 0$ controls the iterations and limits the size of d_k. Note that if $\mu_k > 0$, then d_k is well defined even when $J(x_k)$ is rank-deficient. As $\mu_k \to \infty$, $\|d_k\|_2 \to 0$ and d_k becomes parallel to the steepest descent direction. It can be shown that d_k is the solution to the least squares problem with quadratic constraint

$$\min_{d_k} \|r(x_k) + J(x_k)d_k\|_2, \quad \text{subject to} \quad \|d_k\|_2 \leq \delta_k, \tag{2.8.19}$$

where $\mu_k = 0$ if the constraint in (2.8.19) is not binding and $\mu_k > 0$ otherwise. The set of feasible vectors d_k, $\|d_k\|_2 \leq \delta_k$, can be thought of as a trust region for the linear model $r(x) \approx r(x_k) + J(x_k)(x - x_k)$. Hence, these methods are known as **trust region methods**.

The following trust region strategy has proved very successful in practice:

Let x_0, D_0 and δ_0 be given and choose $\beta \in (0, 1)$. For $k = 0, 1, 2, \ldots$ do

(a) Compute $r(x_k)$, $J(x_k)$, and determine d_k as a solution to the subproblem

$$\min_{d_k} \|r(x_k) + J(x_k)d_k\|_2 \quad \text{subject to} \quad \|D_k d_k\|_2 \leq \delta_k,$$

where D_k is a diagonal scaling matrix.

(b) Compute the ratio $\rho_k = \left(\|r(x_k)\|_2^2 - \|r(x_k + d_k)\|_2^2\right)/\psi_k(d_k)$, where

$$\psi_k(d_k) = \|r(x_k)\|_2^2 - \|r(x_k) + J(x_k)d_k\|_2^2$$

is the model prediction of the decrease in $\|r(x_k)\|_2^2$.

(c) If $\rho_k > \beta$ the step is successful and we set $x_{k+1} = x_k + d_k$ and $\delta_{k+1} = \delta_k$; otherwise, set $x_{k+1} = x_k$ and $\delta_{k+1} = \beta\delta_k$. Update the scaling matrix D_k.

The ratio ρ_k measures the agreement between the linear model and the nonlinear function. After an unsuccessful iteration δ_k is reduced. The scaling D_k can be chosen such that the algorithm is scale invariant, i.e., the algorithm generates the same iterations if applied to $r(Dx)$ for any nonsingular diagonal matrix D. It can be proved that if $r(x)$ is continuously differentiable, $r'(x)$ uniformly continuous and $J(x_k)$ bounded, then this algorithm will converge to a stationary point.

A trust region implementation of the Levenberg–Marquardt method will give a Gauss–Newton step close to the solution of a regular problem. Its convergence will therefore often be slow for large residual or highly nonlinear problems. Methods using second derivative information (see Sect. 2.8.4) are somewhat more efficient, but also more complex.

2.8.4 Quasi-Newton Methods

When the Gauss–Newton method converges slowly or has problems with a rank-deficient Jacobian, Newton's method can be tried. To ensure global convergence a line search algorithm can be used, where the search direction d_k is determined by the quadratic model (2.8.10) and satisfies the linear system

$$H(x_k)d_k = -J(x_k)^T r(x_k). \tag{2.8.20}$$

Note that the Hessian $H(x_k)$ must be positive definite in order for the Newton direction d_k to be a descent direction.

Newton's method is not often used because the second derivative term $Q(x_k)$ in the Hessian is rarely available at a reasonable cost. However, for curve fitting problems, $r_i(x) = y_i - h(x, t_i)$ and the derivatives $\partial^2 r_i(x)/\partial x_j \partial x_k$ can be obtained from the single function $h(x, t)$. If $h(x, t)$ is composed of, e.g., simple exponential and trigonometric functions, then the Hessian can in some cases be computed cheaply. Another case when it may be practical to store approximations to all $G_i(x)$, $i = 1{:}m$, is when every function $r_i(x)$ depends on a small subset of the n variables. Then both the Jacobian $J(x)$ and the Hessians $G_i(x)$ will be *sparse* and special methods may be applied.

Several methods have been suggested that partly take the second derivatives into account, either explicitly or implicitly. An implicit way to obtain second derivative information is to use a general **quasi-Newton** optimization routine, which builds up a sequence of approximations of the Hessians $H(x_k)$. Let B_{k-1} be a symmetric approximation to the Hessian at step k. It is required that the updated approximation B_k approximates the curvature along $s_k = x_k - x_{k-1}$. This gives the quasi-Newton conditions

$$B_k s_k = y_k, \quad y_k = g(x_k) - g(x_{k-1}), \tag{2.8.21}$$

and $g(x_k) = J(x_k)^T r(x_k)$. As starting value, $B_0 = J(x_0)^T J(x_0)$ is recommended. The search directions d_k are then computed from

$$B_k d_k = -g(x_k). \tag{2.8.22}$$

The direct application of the quasi-Newton method to the nonlinear least squares problem has not been so efficient in practice. A more successful approach takes advantage of the special form $J(x_k)^T J(x_k) + Q_k$ of the Hessian and uses a quasi-Newton approximation S_k only for the term $Q(x_k)$. With $S_0 = 0$ the quasi-Newton relations (2.8.21) can then be written as

$$S_k s_k = z_k, \quad z_k = \big(J(x_k) - J(x_{k-1})\big)^T r(x_k), \tag{2.8.23}$$

where S_k is required to be symmetric. An update formula due to Dennis et al. [73, 1981] and used in their subroutine NL2SOL is

$$S_k = S_{k-1} + \frac{w_k y_k^T + y_k w_k^T}{y_k^T s_k} - \frac{(w_k^T s_k) y_k y_k^T}{(y_k^T s_k)^2}, \quad w_k = z_k - S_{k-1} s_k. \tag{2.8.24}$$

It can be shown (see Dennis and Schnabel [72, 1983], pp. 231–232) that this solution to (2.8.23) minimizes the change from S_{k-1} in a certain weighted Frobenius norm. In some cases the update (2.8.24) gives inadequate results. This motivates the inclusion of "sizing" in which the matrix S_k is replaced by $\rho_k S_k$, where

$$\rho_k = \min\{|s_k^T z_k|/|s_k^T S_k s_k|, 1\}.$$

This heuristic choice ensures that S_k converges to zero for zero residual problems, which improves the convergence behavior.

Another way to obtain second derivative information is developed by Ruhe [244, 1979]. It uses a nonlinear conjugate gradient (CG) acceleration of the Gauss–Newton method with exact line searches. This method achieves quadratic convergence and gives much improved results compared to the standard Gauss-Newton method on difficult problems. When exact line searches are used, the CG acceleration amounts to a negligible amount of extra work. However, for small residual problems exact line search is a waste of time.

Outstanding textbooks on numerical methods for unconstrained optimization, nonlinear systems, and nonlinear least squares are Dennis and Schnabel [72, 1983] and Nocedal and Wright [216, 2006]). The much used quasi-Newton algorithm NL2SOL for nonlinear least squares is due to Dennis et al. [73, 1981]. Trust region methods are discussed by Conn et al. [58, 2000]. Methods for solving constrained nonlinear least squares problems are treated by Gulliksson et al. [143, 1997].

2.8.5 Separable Least Squares Problems

In some structured nonlinear least squares problems it is advantageous to separate the parameters into two sets. For example, suppose that we want to fit a linear combination of functions $\phi_j(z; t)$, nonlinear in the parameters $z \in \mathbb{R}^q$, to given data (g_k, t_k), $k = 1{:}m$, by minimizing $\|r(y, z)\|_2^2$, where

$$r_k(y, z) = g_k - \sum_{j=1}^{p} y_j \phi_j(z; t_k), \quad k = 1{:}m. \tag{2.8.25}$$

In this least squares problem, $y \in \mathbb{R}^p$ are linear and $z \in \mathbb{R}^q$ nonlinear parameters. The full least squares problem is

$$\min_{y,z} \|g - \Phi(z)y\|^2, \tag{2.8.26}$$

where $\Phi(z)$ is a matrix whose jth column has elements $\phi_j(z; t_k)$, $k = 1{:}m$. For any fixed value of z the problem (2.8.26) is an easily solved *linear* least squares problem in y. A particularly simple case is when $r(y, z)$ is linear in both y and z. Then we also have

$$r(y, z) = g(y) - \Psi(y)z, \quad \Psi(y) \in \mathbb{R}^{m\times q}.$$

We now describe a method for solving general separable least squares problems. We first observe that the solution of (2.8.26) can be expressed as

$$y(z) = \Phi^{\dagger}(z)g(z), \tag{2.8.27}$$

where $\Phi^{\dagger}(z)$ is the pseudoinverse of $\Phi(z)$. If the linear parameters are eliminated in (2.8.26), the original minimization problem can be cast in the form

$$\min_z \|(I - P_{\Phi(z)})g\|^2, \quad P_{\Phi(z)} = \Phi(z)\Phi(z)^{\dagger}, \tag{2.8.28}$$

where $P_{\Phi(z)}$ is the orthogonal projector onto the column space of $\Phi(z)$. This is a pure nonlinear problem of reduced dimension. The **variable projection method** of Golub and Pereyra [130, 1973] consists of solving (2.8.28), e.g., by a Gauss–Newton method, obtaining the optimal vector z. The linear parameters are then computed from $y = \Phi(z)^{\dagger}g$. To use the Gauss–Newton method, we need a formula for the derivative of an orthogonal projection $P_{\Phi(z)} = \Phi(z)\Phi(z)^{\dagger}$. The following formula, due to Golub and Pereyra holds for any symmetric generalized inverse Φ^{-}. Its proof is a good exercise in differentiating matrix expressions.

Lemma 2.8.1 *Let $\Phi = \Phi(\zeta) \in \mathbb{R}^{m\times n}$ be a matrix of local constant rank and $\Phi^{\dagger}$ its pseudoinverse. Then*

$$\frac{d}{d\zeta}(P_\Phi) = P_\Phi^{\perp}\frac{d\Phi}{d\zeta}\Phi^{\dagger} + (\Phi^{\dagger})^T\frac{d\Phi^T}{d\zeta}P_\Phi^{\perp}, \tag{2.8.29}$$

where $P_\Phi = \Phi\Phi^{\dagger}$ is the orthogonal projector onto $\mathcal{R}(\Phi)$ and $P_\Phi^{\perp} = I - P_\Phi$.

We describe a version of the variable projection algorithm due to Kaufman [176, 1975]. The jth column of the Jacobian of the reduced problem (2.8.28) can be written as

$$J = -\left[P_\Phi^{\perp}\frac{d\Phi}{dz_j}\Phi^{\dagger} + (\Phi^{\dagger})^T\frac{d\Phi^T}{dz_j}P_\Phi^{\perp}\right]y.$$

Kaufman's simplification consists in using an approximate Jacobian obtained by dropping the second term in this formula. The effect is to reduce the work per iteration at the cost of marginally increasing the number of iterations.

The algorithm contains two steps merged into one. Let $x_k = (y_k, z_k)^T$ be the current approximate solution. The next approximation is determined as follows:

(i) Compute the solution δy_k to the linear subproblem

$$\min_{\delta y_k} \left\| f(z_k)\delta y_k - \big(g(z_k) - f(z_k)y_k\big) \right\|_2, \tag{2.8.30}$$

and put $y_{k+1/2} = y_k + \delta_k$ and $x_{k+1/2} = (y_{k+1/2}, z_k)^T$.

(ii) Compute d_k as the Gauss–Newton step at $x_{k+1/2}$ from

$$\min_{d_k} \left\| C(x_{k+1/2})d_k + r(y_{k+1/2}, z_k) \right\|_2, \tag{2.8.31}$$

where the Jacobian is $C(x_{k+1/2}) = \big(f(z_k), r_z(y_{k+1/2}, z_k)\big)$. Take $x_{k+1} = x_k + \lambda_k d_k$ and go to (i).

In (2.8.31) we have used that, by (2.8.26), the first derivative of r with respect to y is given by $r_y(y_{k+1/2}, z_k) = f(z_k)$. The derivatives with respect to z are

$$r_z(y_{k+1/2}, z_k) = B(z_k)y_{k+1/2} - g'(z_k), \qquad B(z)y = \left(\frac{\partial F}{\partial z_1}y, \ldots, \frac{\partial F}{\partial z_q}y\right),$$

where $B(z)y \in \mathbb{R}^{m\times q}$. Note that in case $r(y, z)$ is linear also in y it follows from (2.8.7) that $C(x_{k+1/2}) = (f(z_k), H(y_{k+1/2}))$. To be robust the algorithms for separable problems must employ a line search or trust region approach for the Gauss–Newton steps as described in Sect. 2.8.3.

It can be shown that the Gauss–Newton algorithm applied to (2.8.28) has the same asymptotic convergence rate as the ordinary Gauss–Newton method. Both converge quadratically for zero residual problem, in contrast to the naive algorithm of alternatively minimizing $\|r(y, z)\|_2$ over y and z, which *always* converges linearly.

The variable projection approach not only reduces the dimension of the parameter space, but also leads to a better conditioned problem. One advantage of Kaufman's algorithm is that *no starting values for the linear parameters have to be provided*. We can, e.g., take $y_0 = 0$ and determine $y_1 = \delta y_1$, in the first step of (2.8.30). This seems to make a difference in the first steps of the iterations. Several problems, for which methods not using separability fail, have been solved by the variable projection algorithm.

An important example of a separable problem is fitting a linear combination of exponential functions

$$g(t) = \sum_{j=1}^{p} c_j e^{\lambda_j t}, \tag{2.8.32}$$

to observations $g_k = g(t_k) + \epsilon_k$, made at equidistant times $t_k = kh$, $k = 0{:}m$. Since $g(t)$ in (2.8.32) depends on p linear parameters c_j and p nonlinear parameters λ_j, at least $m = 2p$ observations are needed. If we set $v_j = e^{\lambda_j h}$, then $e^{\lambda_j t_k} = e^{\lambda_j hk} = v_j^k$, and the model (2.8.32) becomes

$$g(t_k) = \sum_{j=1}^{p} c_j v_j^k, \quad k = 0:m. \tag{2.8.33}$$

For given $v = (v_1, \ldots, v_p)$, this is a linear least squares problem for c, which in matrix form can be written as

$$\min_c \|M(v)c - g\|_2^2, \tag{2.8.34}$$

where $(m \geq 2p)$

$$M(v) = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ v_1 & v_2 & \cdots & v_p \\ \vdots & \vdots & \cdots & \vdots \\ v_1^m & v_2^m & \cdots & v_p^m \end{pmatrix}, \quad c = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_p \end{pmatrix}, \quad g = \begin{pmatrix} g_0 \\ g_1 \\ \vdots \\ g_m \end{pmatrix}.$$

Here the observed values g_k have been substituted for $g(t_k)$. Note that $M(v)$ is a Vandermonde matrix.

Prony's method[17] uses the fact that $g(t_k)$ satisfies a homogeneous linear difference equation of order p with constant coefficients. (For properties of linear difference equations, see [63, 2008] Sect. 3.3.5).) The nonlinear parameters $v_1, \ldots, v_p$ are the roots of the corresponding characteristic polynomial

$$P(v) = (v - v_1)(v - v_2)\cdots(v - v_p) = v^p + s_1 v^{p-1} + \cdots + s_p.$$

The coefficients $s_1, s_2, \ldots, s_p$ can be obtained as follows. Multiplying the first $p+1$ equations in $Mc = g$ in turn by $s_p, s_{p-1}, \ldots, s_1, s_0 = 1$ and adding, we obtain

$$\sum_{j=1}^{p} P(v_j)c_j = \sum_{k=0}^{p} s_{p-k} g_k = 0,$$

because $P(v_j) = 0$, $j = 1{:}p$. Repeating this with rows $k, k+1, \ldots, k+p$, $k = 2{:}m - p$, we obtain $(m - p)$ equations

$$\begin{pmatrix} g_{p-1} & g_{p-2} & \cdots & g_0 \\ g_p & g_{p-1} & \cdots & g_1 \\ \vdots & \vdots & \vdots & \vdots \\ g_{m-1} & g_{m-2} & \cdots & g_{m-p} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_p \end{pmatrix} = - \begin{pmatrix} g_p \\ g_{p+1} \\ \vdots \\ g_m \end{pmatrix},$$

If $m > 2p$, this is a *linear* overdetermined Toeplitz system for the coefficients $s_1, \ldots, s_p$ of the characteristic polynomial $P(v)$. This can be solved by the method of linear least squares. (Special methods for solving Toeplitz least squares problems

[17] Developed by the French mathematician and engineer Gaspard de Prony in 1795; see [239, 1795].

are discussed in Sect. 8.4, [27, 1996]; see also Nagy [213, 1993].) The roots $v_j = e^{\lambda_j h}$ of the polynomial $P(v)$ are then determined, e.g., as the eigenvalues of the companion matrix of $P(v)$; see Example 3.1.4. (This is how the MATLAB functions `v = roots([1,s])` works.) From this the $\lambda_j = h^{-1} \log v_j$ exponents in (2.8.32) are obtained. Finally, c_j are computed from the linear least squares problem (2.8.34).

The solution obtained by Prony's method can be complex exponentials, damped and undamped sinusoids, and real exponentials, depending on the roots of the polynomial $P(v)$. If the data vector g is real, then complex exponents must occur in complex conjugate pairs.

Example 2.8.1 Fitting real exponentials occurs, e.g., in radioactive decay, compartment models, and atmospheric transfer functions. This is a particularly difficult and ill-conditioned problem, because the same data can be well approximated by different exponential sums. Lanczos [183, 1956], Chapter IV.23, shows that the three exponential sums

$$\begin{aligned} f_1(t) &= 0.0951e^{-t} + 0.8607e^{-3t} + 1.5576e^{-5t}, \\ f_2(t) &= 0.305e^{-1.58t} + 2.202e^{-4.45t}, \\ f_3(t) &= 0.041e^{-0.5t} + 0.79e^{-2.73t} + 1.68e^{-4.96t}, \end{aligned}$$

approximate the same data to two decimals for $0 \le t \le 1.2$. □

Osborne [221, 1975] shows how the linear parameters can be eliminated in separable problems. Prony's method is known to perform poorly when the signal is noisy, and the method has been shown to be inconsistent. Osborne and Smyth [223, 1995] develop a modified Prony's method that does estimate exponentials, which best fits the available data. Numerically stable variants of Prony's method are discussed by Pereyra and Scherer [233, 2010]. In many applications it is natural to restrict the coefficients to be positive. A convex cone characterization is then possible and Ruhe [245, 1980] proposes a special algorithm for this case.

Models and algorithms for general nonlinear parameter estimation are discussed in Schwetlick [255, 1992]. An early analysis of the convergence of the Gauss-Newton method is given by Pereyra [232, 1967]. The variable projection method is an extension of an idea in Guttman et al. [144, 1973].

The carefully written and documented program VARPRO implements the variable projection algorithm, with the modification due to Kaufman. It also calculates the covariance matrix. A version called VARP2 by LeVeque handles multiple right-hand sides. Both VARPRO and VARP2 are available in the public domain at http://www.netlib.org/opt/. Golub and LeVeque [128, 1979] extend the VARPRO algorithm to the case when several data sets are to be fitted to the model with the same nonlinear parameter vector; see also Kaufman and Sylvester [178, 1992]. Kaufman and Pereyra [177, 1978] consider problems with nonlinear equality constraints. Ruhe and Wedin [246, 1980] analyze several different algorithms for a more general class of separable problems. A review of developments and applications of the variable

projection approach for separable nonlinear least squares problems is given by Golub and Pereyra [131, 2003].

2.8.6 Iteratively Reweighted Least Squares

In some applications it might be more adequate to consider the minimization problem

$$\min_x \|Ax - b\|_p^p \qquad 1 \le p < \infty, \tag{2.8.35}$$

where $p \neq 2$. An illustrative example is estimating a scalar γ from a vector of m observations $y_1 \le y_2 \le \cdots \le y_m$. The ℓ_p-norm estimates for $p = 1, 2$, and ∞ correspond to the median, mean, and midrange, respectively. Whereas the mean value uses all data y_i, $i = 1{:}m$, the median does not depend on the extreme values of y_1 and y_m. On the other hand, the midrange $(y_1 + y_m)/2$ depends *only* on the extreme data points. These observations are valid more generally. The ℓ_1 solution to (2.8.35) is more **robust** than the least squares solution, i.e., a small number of isolated large errors will have a large effect on the solution. The same holds for solutions corresponding to values of p such that $(1 < p < 2)$.

For solving problem (2.8.35) when $p \neq 2$, the **iteratively reweighted least squares** (IRLS) method (see Osborne [222, 1985]) is widely used. This approach reduces the problem to the solution of a sequence of weighted least squares problems. This is attractive since methods and software for weighted least squares are available. Provided that $|r_i(x)| = |b - Ax|_i > 0, i = 1, \ldots, m$, problem (2.8.35) can be restated in the form

$$\min_x \psi(x), \qquad \psi(x) = \sum_{i=1}^m |r_i(x)|^p = \sum_{i=1}^m |r_i(x)|^{p-2} r_i(x)^2. \tag{2.8.36}$$

This can be interpreted as a weighted least squares problem

$$\min_x \|D(r)^{(p-2)/2}(b - Ax)\|_2, \qquad D(r) = \operatorname{diag}(|r(x)|). \tag{2.8.37}$$

Here and in the following the notation $\operatorname{diag}(|r|), r \in \mathbf{R}^m$, denotes the diagonal matrix with ith component $|r_i|$.

The diagonal weight matrix $D(r)^{(p-2)/2}$ in (2.8.37) depends on the unknown solution x. In Algorithm 2.8.1 $x^{(0)}$ is an initial approximation, e.g., equal to the unweighted least squares solution. It is assumed that $r_i^{(0)} = (b - Ax^{(0)})_i \neq 0$, $i = 1{:}m$. Such an IRLS method was first used by Lawson [189, 1961]. Since $r^{(k)} = b - Ax^{(k)}$, $x^{(k+1)}$ solves the problem $\min_x \|D_k(b - Ax)\|_2$, but the implementation above is to be preferred. The linear subproblem in each step of IRLS can be solved by computing the QR factorization of $D_k A$ or, if the normal equations are to be used, the Cholesky factorization of $A^T D_k^2 A$; see Sect. 2.7.1.

Algorithm 2.8.1 (*Iteratively Reweighted Least Squares*)

$$
\begin{aligned}
&\text{for } k = 0, 1, 2, \ldots \\
&\quad r^{(k)} = (b - Ax^{(k)}); \\
&\quad D_k = \operatorname{diag}\left((|r^{(k)}|)^{(p-2)/2}\right); \\
&\quad \text{solve } \min_{\delta x} \left\| D_k(r^{(k)} - A\delta x)\right\|_2; \\
&\quad x^{(k+1)} = x^{(k)} + \delta x^{(k)}; \\
&\text{end}
\end{aligned}
$$

Cline [53, 1972] proved that the rate of convergence of IRLS is linear. Osborne [222, 1985] shows that the IRLS method is convergent for $1 < p < 3$, and that any fixed point of the IRLS iteration satisfies the necessary conditions for a minimum of $\psi(x)$ in (2.8.36). IRLS converges for $p = 1$ provided that the problem has a unique nondegenerate solution. If $p < 2$ and $r_i^{(k)} = 0$, then the corresponding weight is infinite. If this occurs for a component that does not have a zero residual in the solution, there will be misconvergence. Li [194, 1993] suggests that zero residuals be perturbed by a small amount, e.g., $100ue$, where u is the machine precision.

We now compare IRLS with Newton's method for the problem

$$\min_x \psi(x) = \phi(r(x)), \quad r(x) = b - Ax,$$

where $\psi(x)$ is the function in (2.8.36). The gradient of $\phi(r)$ is

$$g^T = -A^T y, \quad y_i = p\,\operatorname{sign}(r_i)(|r_i|)^{p-1}, \quad i = 1{:}m.$$

The Hessian matrix of second derivatives of $\psi(x)$ is $H = A^T W A$, where

$$W = \operatorname{diag}(w_i), \quad w_i = p(p-1)(|r_i|)^{p-2}, \quad i = 1{:}m.$$

Hence the Newton step s satisfies $Hs = -g$. Apart from the factor $p - 1$ this is just the normal equations for the weighted least squares problem (2.8.37). Hence, the Newton step for minimizing $\psi(x)$ is related to the IRLS step by

$$s_k = (1/(p-1))\delta x^{(k)}.$$

Since the Newton step is always a descent direction for the objective function $\psi(x)$ it follows that the same is true for the step obtained from IRLS. Hence, global convergence and local quadratic convergence can be achieved by using a line search procedure together with IRLS.

In robust linear regression, possible "outsiders" among the data points are identified and given less weight. Huber's M-estimator [170, 1981] can be viewed as a compromise between ℓ_2 and ℓ_1 approximation. It uses the least squares estimator for

"normal" data but the ℓ_1 norm estimator for data points that disagree too much with the normal picture. More precisely, the Huber M-estimate minimizes the objective function

$$\psi(x) = \sum_{i=1}^{m} \rho(r_j(x)/\sigma), \quad \rho(t) = \begin{cases} \frac{1}{2}t^2 & \text{if } |t| \le \gamma, \\ \gamma|t| - \frac{1}{2}\gamma^2 & \text{if } |t| > \gamma, \end{cases} \tag{2.8.38}$$

where γ is a tuning parameter and σ a scaling factor that depends on the data to be estimated. If σ is a constant, then it is no restriction to take $\sigma = 1$. Since the Huber function is smooth near zero residuals, it can be expected that it is easier to minimize than the ℓ_1 norm of the residual.

Methods for computing the Huber M-estimator are given by Clark and Osborne [52, 1986], Ekblom [83, 1988], and Madsen and Nielsen [201, 1990]. O'Leary [217, 1990] studies different implementations of Newton-like methods. The Newton step s for minimizing $\psi(x)$ in (2.8.38) ($\sigma = 1$) is given by the solution of $A^T DAs = A^T y$, where

$$y_i = \rho'(r_i), \qquad D = \operatorname{diag}(\rho''(r_i)), \quad i = 1{:}m.$$

This is similar to the Newton method for ℓ_p approximation. O'Leary recommends that at the initial iterations the cutoff value γ in the Huber function (2.8.38) 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.

For $p = 1$ convergence of IRLS can be slow. Madsen and Nielsen [202, 1993] give a finite more efficient algorithm for this case. At each iteration the non-differentiable function $\psi(x)$, $p = 1$, in (2.8.36) is approximated by a Huber function (2.8.38) with some threshold parameter γ. This parameter is successively reduced until, when γ is small enough, the ℓ_1 solution can be detected. A similar strategy is used by Daubechies et al. [65, 2010] in IRLS for sparse recovery.

Linear programming methods for solving $\min_x \|Ax - b\|_1$ are given by Barrodale and Roberts [14, 1973]. Bartels et al. [15, 1978] use a projected gradient technique, where a descent methods is used to find the correct subset of zero residuals.

IRLS can be used to solve other nonlinear regression problems. An important case is **logistic regression**, used in binary classification. Let $X \in \mathbb{R}^{m\times n}$, $y \in \mathbb{R}^m$, be a data set, where y_i is a Bernoulli random variable that takes the value $y_i = 1$ with probability $\mu(x_i)$ and the value $y_i = 0$ with probability $1 - \mu(x_i)$. Then the variance of y_i equals $\mu(x_i)(1 - \mu(x_i))$. It is important to notice that the variance is not constant, but depends on the experiment x_i. The relation between the experiment x_i and the expected value of its outcome is modeled by the logistic function

$$\mu(x, \beta) = 1/(e^{-\beta^T x} + 1). \tag{2.8.39}$$

If z varies from $-\infty$ to ∞, then $\mu(z)$ varies from zero to one. Hence, $\mu(z)$ can be interpreted as a probability. Thus the regression model is

$$y = \mu(x, \beta) + \epsilon, \tag{2.8.40}$$

where ϵ is the error term and $\beta \in \mathbb{R}^n$ is the parameter vector to be predicted. Using the **logit function** $g(\mu) = \mu/(1-\mu)$, this model can be transformed into the model

$$g(y) = g(\mu(x)) + \widetilde{\epsilon} = \beta^T + \widetilde{\epsilon},$$

which is linear in the parameter vector β. Since the variance of the error $\widetilde{\epsilon}$ is not independent of the mean, this is not a Gauss–Markov model and the least squares method is not valid. Instead, the parameters are estimated using the **maximum likelihood** method.

The outcome y is a Bernoulli random variable with mean $\mu(x, \beta)$. Therefore, the probability of the outcome of an experiment can be written as

$$\begin{aligned} P(x, y \mid \beta) &= \begin{cases} \mu(x, \beta) & \text{if } y = 1, \\ 1 - \mu(x, \beta) & \text{if } y = 0, \end{cases} \\ &= \mu(x, \beta)^y (1 - \mu(x, \beta))^{1-y}. \end{aligned}$$

It follows that the log-likelyhood function $L(X, y, \beta)$ of the data X, y under the model with parameter β is

$$\log(L(X, y, \beta)) = \sum_{i=1}^{n} \Big(y_i \log(\mu(x_i, \beta)) + (1 - y_i) \log(1 - \mu(x_i, \beta)) \Big).$$

The estimate of β is chosen as the value $\widehat{\beta}$ that maximizes this log-likelihood function. Setting the gradient vector equal to zero gives the system of nonlinear equations

$$h(\beta) = X^T \big(y - \mu(X, \beta) \big),$$

where $\mu(X, \beta) = \big(\mu(\beta^T x_1), \ldots, \mu(\beta^T x_m) \big)^T$. Newton's method for solving this system is $x^{(k+1)} = x^{(k)} + \delta x^{(k)}$, where $\delta x^{(k)}$ solves the weighted least squares problem

$$\min_{\delta x} \big\| D_k \big(\mu(A, x^{(k)}) - b \big) \big\|_2, \quad D_k = \operatorname{diag}(\sqrt{w_1}, \ldots, \sqrt{w_m}), \tag{2.8.41}$$

and $w_i = \mu(a_i, x^{(k)})(1 - \mu(a_i, x^{(k)}))$. The problem (2.8.41) can be solved by IRLS.

Logistic regression is often applied to large-scale data sets. Then iterative methods may have to be used to solve the weighted linear least squares subproblems (2.8.41); see Komarek [181, 2004].

2.8.7 Nonlinear Orthogonal Regression

In Sect. 2.7.7 we considered the *linear* orthogonal regression problem of fitting a hyperplane M to a set of given points $P_i \in \mathbb{R}^n$, $i = 1{:}m$. The solution to this problem was obtained from the SVD of a related matrix. In this section we consider the case when a nonlinear model is to be fitted by minimizing the *sum of squares of the orthogonal distances* from observations (y_i, t_i), $i = 1{:}m$, to the curve

$$y = f(p, t), \tag{2.8.42}$$

as illustrated in Fig. 2.16. In (2.8.42) f is a scalar nonlinear function, t a scalar variable, and $p \in \mathbb{R}^n$ are parameters to be determined.

Assume that y_i and t_i are subject to errors $\bar{\epsilon}_i$ and $\bar{\delta}_i$, respectively, so that

$$y_i + \bar{\epsilon}_i = f(p, t_i + \bar{\delta}_i), \quad i = 1{:}m,$$

where $\bar{\epsilon}_i$ and $\bar{\delta}_i$ are independent random variables with zero mean and variance σ^2. Then the parameters p should be chosen so that the sum of squares of the **orthogonal distances** from the observations (y_i, t_i) to the curve in (2.8.42) is minimized, cf. Fig. 2.16. Hence, p should be chosen as the solution to

$$\min_{p,\epsilon,\delta} \sum_{i=1}^{m} (\epsilon_i^2 + \delta_i^2) \quad \text{subject to} \quad y_i + \epsilon_i = f(p, t_i + \delta_i), \quad i = 1{:}m.$$

Eliminating ϵ_i using the constraints we obtain the **orthogonal distance problem**

$$\min_{p,\delta} \sum_{i=1}^{m} \left[\left(f(p, t_i + \delta_i) - y_i \right)^2 + \delta_i^2 \right]. \tag{2.8.43}$$

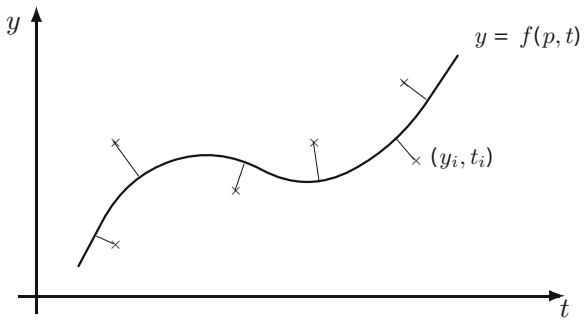


Fig. 2.16 Orthogonal distance regression

Note that this is a nonlinear least squares problem even if $f(p,t)$ is linear in p.

Problem (2.8.43) has $m+n$ unknowns p and δ. In applications usually $m \gg n$ and accounting for the errors in t_i will considerably increase the size of the problem. Therefore, the use of standard methods will not be efficient unless the special structure is taken into account to reduce the work. If we define the residual vector $r(\delta, p) = (r_1^T(\delta, p), r_2^T(\delta))$ by

$$r_1^T(\delta, p)_i = f(p, t_i + \delta_i) - y_i, \qquad r_2^T(\delta)_i = \delta_i, \quad i = 1{:}m,$$

the Jacobian for problem (2.8.43) can be written in block form as

$$\widetilde{J} = \begin{pmatrix} D_1 & J \\ I_m & 0 \end{pmatrix} \in \mathbb{R}^{2m\times(m+n)}, \tag{2.8.44}$$

where $D_1 = \operatorname{diag}(d_1, \ldots, d_m)$ and

$$d_i = \left(\frac{\partial f}{\partial t}\right)_{t=t_i+\delta_i}, \quad J_{ij} = \left(\frac{\partial f}{\partial p_j}\right)_{t=t_i+\delta_i}, i = 1{:}m, \quad j = 1{:}n. \tag{2.8.45}$$

Note that $\widetilde{J}$ is sparse and highly structured. In the Gauss–Newton method we compute search directions $(\Delta\delta_k, \Delta p_k)$ from the linear least squares problem

$$\min_{\Delta\delta, \Delta p} \left\| \widetilde{J} \begin{pmatrix} \Delta\delta \\ \Delta p \end{pmatrix} - \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \right\|_2, \tag{2.8.46}$$

where $\widetilde{J}, r_1$, and r_2 are evaluated at the current estimates of δ and p. To solve this problem, we need the QR factorization of $\widetilde{J}$. This can be computed in two steps. First, we apply a sequence of Givens rotations $Q_1 = G_m \cdots G_2 G_1$, where $G_i = R_{i,i+m}$, $i = 1{:}m$, to zero the (2, 1) block of $\widetilde{J}$:

$$Q_1 \widetilde{J} = \begin{pmatrix} D_2 & K \\ 0 & L \end{pmatrix}, \qquad Q_2 \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix},$$

where D_2 is again a diagonal matrix. The problem (2.8.46) now decouples, and Δp_k is determined as the solution to $\min_{\Delta p} \|L\Delta p - s_2\|_2$. Here $L \in \mathbb{R}^{m\times n}$, so this problem is of the same size as for the Gauss–Newton correction in the standard nonlinear least squares problem. We then have

$$\Delta\delta_k = D_2^{-1}(s_1 - K\Delta p_k).$$

So far we have assumed that y and t are scalar variables. More generally, if $y \in \mathbb{R}^{n_y}$ and $t \in \mathbb{R}^{n_t}$ the problem becomes

$$\min_{p,\delta} \sum_{i=1}^{m} \Big(\| f(p, t_i + \delta_i) - y_i \|_2^2 + \| \delta_i \|_2^2 \Big).$$

The structure in this more general problem can be taken advantage of in a similar manner.

The general orthogonal distance problem has not received the same attention as the standard nonlinear regression problem, except for the case when f is linear in x. One reason is that, if the errors in the independent variables are small, then ignoring these errors will not seriously degrade the estimates of x. The algorithm by Boggs et al. [32, 1987] and [33, 1989] uses a stabilized Gauss–Newton method and incorporates a full trust region strategy. Schwetlick and Tiller [254, 1985] use a partial Marquardt-type regularization where only the Δx part of $\widetilde{J}$ is regularized.

2.8.8 *Fitting Circles and Ellipses*

A special nonlinear least squares problem that arises in many applications is to fit given data points to a geometrical element, which may be defined in implicit form. We have already discussed fitting data to an affine linear manifold such as a line or a plane. The problem of fitting circles, ellipses, spheres, and cylinders arises in applications such as computer graphics, coordinate meteorology, and statistics.

Least squares algorithms to fit an implicitly defined curve in the x-y plane can be divided into two classes. In **algebraic fitting**, a least squares functional is used that directly involves the function $f(x, y, p) = 0$ to be fitted. If (x_i, y_i), $i = 1{:}n$, are given data points, the functional

$$\Phi(p) = \sum_{i=1}^{m} f^2(x_i, y_i, p)$$

is minimized. The second method, **geometric fitting**, minimizes a least squares functional involving the geometric distances from the data points to the curve; cf. orthogonal distance regression. Often algebraic fitting leads to a simpler problem, in particular when f is linear in the parameters p.

We first discuss algebraic fitting of circles. A circle has three degrees of freedom and can be represented algebraically by

$$f(x, y, p) = a \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} b_1 & b_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + c = 0.$$

We define a parameter vector p and an $m \times 4$ matrix S with rows s_i^T by

$$p = (a, b_1, b_2, c)^T, \qquad s_i^T = (x_i^2 + y_i^2, x_i, y_i, 1).$$

The problem can now be formulated as

$$\min_p \|Sp\|_2^2 \quad \text{subject to} \quad \|p\|_2 = 1. \tag{2.8.47}$$

Note that p is defined only up to a constant multiple, which is why the constraint is required. The solution is the right singular vector corresponding to the smallest singular value of S. When p is known, the center z and radius ρ of the circle can be obtained from

$$z = -\frac{1}{2a}\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \qquad \rho = \frac{1}{2a}\sqrt{\|b\|_2^2 - 4ac}\,. \tag{2.8.48}$$

We now discuss the algebraic fitting of ellipses. An ellipse in the (x, y)-plane can be represented algebraically by

$$f(x, y, p) = (x\ y)\begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + (b_1\ b_2)\begin{pmatrix} x \\ y \end{pmatrix} + c = 0. \tag{2.8.49}$$

If we define

$$p = (a_{11}, a_{12}, a_{22}, b_1, b_2, c)^T, \qquad s_i^T = (x_i^2, 2x_i y_i, y_i^2, x_i, y_i, 1), \tag{2.8.50}$$

then we have $\Phi(p) = \|Sp\|_2^2$, where S is an $m \times 6$ matrix with rows s_i^T. Obviously the parameter vector is only determined up to a constant factor. Hence, we must complete the problem formulation by including some constraint on p. Three such constraints have been considered for fitting ellipses.
(a) **SVD constraint**:

$$\min_p \|Sp\|_2^2 \quad \text{subject to} \quad \|p\|_2 = 1. \tag{2.8.51}$$

Again, the solution of this constrained problem is the right singular vector corresponding to the smallest singular value of S.

(b) **Linear constraint**:

$$\min_p \|Sp\|_2^2 \quad \text{subject to} \quad d^T p = 1, \tag{2.8.52}$$

where d is a fixed vector such that $\|d\|_2 = 1$. Let H be an orthogonal matrix such that $Hd = e_1$. Then the constraint becomes $e_1^T Hp = 1$ and we can write

$$Sp = SH^T Hp = SH^T \begin{pmatrix} 1 \\ q \end{pmatrix} = s + S_2 q,$$

where $SH^T = (s,\ S_2)$. We have transformed (2.8.52) into the standard linear least squares problem $\min_q \|S_2 q + s\|_2^2$.

(c) **Quadratic constraint**:

$$\min_p \|Sp\|_2^2 \quad \text{subject to} \quad \|Bp\|_2 = 1. \tag{2.8.53}$$

The different constraints can lead to very different solutions, unless the errors in the fit are small; see Varah [289, 1996]. Of particular interest is the choice $B = (0,\ I)$ in (c). With $p^T = (p_1,\ p_2)$, the constraint becomes $\|p_2\|_2^2 = 1$, and the problem is a generalized total least squares problem. Let

$$S = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

be the QR factorization of S. Then p_2 is determined from the SVD of R_{22} and p_1 is obtained from $R_{11}p_1 = -R_{12}p_2$.

One desirable property of a fitting algorithm is that when the data are translated and rotated, the fitted ellipse should be transformed in the same way. It can be seen that to lead to this kind of invariance the constraint must involve only symmetric functions of the eigenvalues of the matrix A. A disadvantage of the SVD constraint is its non-invariance under translation and rotations. In case of a linear constraint the choice $d = (1,\ 0,\ 1,\ 0,\ 0,\ 0)^T$, which corresponds to

$$\text{trace}(A) = a_{11} + a_{22} = \lambda_1 + \lambda_2 = 1, \tag{2.8.54}$$

gives the desired invariance. The constraint

$$\|A\|_F^2 = a_{11}^2 + 2a_{12}^2 + a_{22}^2 = \lambda_1^2 + \lambda_2^2 = 1, \tag{2.8.55}$$

attributed to Bookstein, also leads to this kind of invariance. Note that the Bookstein constraint can be put in the form $\begin{pmatrix} 0 & I \end{pmatrix}$ by permuting the variables and scaling by $\sqrt{2}$.

To construct and plot the ellipse, it is convenient to convert the algebraic form (2.8.49) to the parametric form

$$\begin{pmatrix} x(\theta) \\ y(\theta) \end{pmatrix} = \begin{pmatrix} x_c \\ y_c \end{pmatrix} + Q(\alpha) \begin{pmatrix} a\,\cos(\theta) \\ b\,\sin(\theta) \end{pmatrix}, \quad Q(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}. \tag{2.8.56}$$

The new parameters (x_c, y_c, a, b, α) can be obtained from the algebraic parameter vector p in (2.8.50). Let $A = Q\Lambda Q^T$ be the eigenvalue decomposition of the real symmetric 2×2 matrix A in (2.8.49). An algorithm for computing this accurately is given in Sect. 3.6.2. If new coordinates $z = Q\widetilde{z} + s$ are introduced in the algebraic form (2.8.49), this equation becomes

$$\widetilde{z}^T \Lambda \widetilde{z} + (2As + b)^T Q\widetilde{z} + (As + b)^T s + c = 0.$$

Here s can be chosen so that this equation reduces to $\lambda_1\widetilde{x}^2 + \lambda_2\widetilde{y}^2 + \widetilde{c} = 0$. Hence, the center s is

$$s = \begin{pmatrix} x_c \\ y_c \end{pmatrix} = -\frac{1}{2}A^{-1}b = -\frac{1}{2}A^{-1}Q\Lambda^{-1}(Q^T b), \tag{2.8.57}$$

and the axes $(a,\ b)$ of the ellipse are given by

$$\begin{pmatrix} a \\ b \end{pmatrix} = \sqrt{\widetilde{c}}\,\mathrm{diag}\Lambda^{-1/2}, \qquad \widetilde{c} = -c - \frac{1}{2}b^T s = \frac{1}{2}\widetilde{b}^T\Lambda^{-1}\widetilde{b}. \tag{2.8.58}$$

We now consider the geometric fitting of data (x_i, y_i), $i = 1{:}m$, to a curve given in the implicit form $f(x, y, p) = 0$. This problem is

$$\min_p \sum_{i=1}^{m} d_i^2(p), \quad d_i^2(p) = \min_{f(x,y,p)=0} \left((x - x_i)^2 + (y - y_i)^2\right), \tag{2.8.59}$$

where $d_i(p)$ is the orthogonal distance from each data point to the curve. For implicitly defined functions the calculation of the distance function $d_i(p)$ is more complicated than for explicit functions. When the curve admits a parametrization, as in the case of the ellipse, the minimization problem for each point is only one-dimensional.

We consider first the orthogonal distance fitting of a circle written in parametric form

$$f(x, y, p) = \begin{pmatrix} x - x_c - r\cos\phi \\ y - y_c - r\sin\phi \end{pmatrix} = 0, \tag{2.8.60}$$

where $p = (x_c, y_c, r)^T$. The problem can be written as a nonlinear least squares problem

$$\min_{p,\phi_i} \|r(p,\ \phi)\|_2^2, \quad \phi = (\phi_1, \ldots, \phi_m), \tag{2.8.61}$$

where

$$r = \begin{pmatrix} r_1 \\ \vdots \\ r_m \end{pmatrix} \in \mathbb{R}^{2m}, \quad r_i = \begin{pmatrix} x_i - x_c - r\cos\phi_i \\ y_i - y_c - r\sin\phi_i \end{pmatrix}.$$

We have $2m$ nonlinear equations for $m+3$ unknowns $\phi_1, \ldots, \phi_m$ and x_c, y_c, r. (Note that at least 3 points are needed to define a circle.)

We now show how to construct the Jacobian of $r(p,\ \phi)$, which should be evaluated at the current approximations to the $m+3$ parameters. We need the partial derivatives

$$\frac{\partial r_i}{\partial \phi_i} = r\begin{pmatrix} \sin\phi_i \\ -\cos\phi_i \end{pmatrix}, \qquad \frac{\partial r_i}{\partial r} = -\begin{pmatrix} \cos\phi_i \\ \sin\phi_i \end{pmatrix},$$

and

$$\frac{\partial r_i}{\partial x_c} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, \qquad \frac{\partial r_i}{\partial y_c} = \begin{pmatrix} 0 \\ -1 \end{pmatrix}.$$

After reordering of its rows, the Jacobian has the form

$$J = \begin{pmatrix} rS & A \\ -rC & B \end{pmatrix}, \quad A, B \in \mathbb{R}^{m\times 3}.$$

where

$$S = \operatorname{diag}(\sin\phi_i), \qquad C = \operatorname{diag}(\cos\phi_i) \tag{2.8.62}$$

are two $m \times m$ diagonal matrices. Here the first block column, which corresponds to the m parameters ϕ_i, is orthogonal. Multiplying from the left by an orthogonal matrix, we obtain

$$Q^T J = \begin{pmatrix} rI & SA - CB \\ 0 & CA + SB \end{pmatrix}, \qquad Q = \begin{pmatrix} S & C \\ -C & S \end{pmatrix}. \tag{2.8.63}$$

To obtain the QR factorization of J, we compute the QR factorization of the $m \times 3$ matrix $CA + SB$. Then a Gauss–Newton type method for nonlinear least squares problems can be used to solve the problem. Good starting values for the parameters may often be obtained using an algebraic fit, as described in the previously. Experience shows that the amount of computation involved in a geometric fit is at least one order of magnitude larger than for an algebraic fit.

For the geometric fit of an ellipse, the parametric form

$$f(x, y, p) = \begin{pmatrix} x - x_c \\ y - y_c \end{pmatrix} - Q(\alpha) \begin{pmatrix} a\cos\phi \\ b\sin\phi \end{pmatrix} = 0 \tag{2.8.64}$$

can be used, where $p = (x_c, y_c, a, b, \alpha)^T$ and $Q(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$. The problem can be formulated as a nonlinear least squares problem of the form (2.8.61); see Gander et al. [110, 1994].

The fitting of a sphere or an ellipsoid can be treated analogously. The sphere can be represented in parametric form as

$$f(x, y, z, p) = \begin{pmatrix} x - x_c - r\cos\theta\cos\phi \\ y - y_c - r\cos\theta\sin\phi \\ z - z_c - r\sin\theta \end{pmatrix} = 0, \tag{2.8.65}$$

where $p = (x_c, y_c, z_c, r)^T$. We get $3m$ nonlinear equations for $2m + 4$ unknowns. The first $2m$ columns of the Jacobian can easily be brought into upper triangular form.

When the data cover only a small arc of the circle or a small patch of the sphere, the fitting problem can be ill-conditioned. An important application involving this type of data is the fitting of a spherical lens. Likewise, the fitting of a sphere or an ellipsoid to near planar data will lead to an ill-conditioned problem.

Exercises

2.8.1 (a) The general form for a quadratic function is

$$\phi(x) = \frac{1}{2}x^T Gx - b^T x + c,$$

where $G \in \mathbb{R}^{n\times n}$ is a symmetric matrix and $b \in \mathbb{R}^n$. Show that the gradient of ϕ is $g = Gx - b$ and the Hessian is G. Also, show that if $g(x^*) = 0$, then

$$\phi(x) = \phi(x^*) + \frac{1}{2}(x - x^*)^T H(x - x^*).$$

(b) Suppose that G is symmetric and nonsingular. Using the result in (a), show that Newton's method will find a stationary point of ϕ in one step from an arbitrary starting point x_0. Under what condition is this a minimizer?

2.8.2 Let $\phi(x)$ be a quadratic function with Hessian G, which need not be positive definite.

(a) Let $\psi(\lambda) = \phi(x_0 - \lambda d)$. Show using Taylor's formula that

$$\psi(\lambda) = \psi(0) - \lambda g^T d + \frac{1}{2}\lambda^2 d^T Gd.$$

Conclude that if $d^T Gd > 0$ for a certain vector d, then $\psi(\lambda)$ is minimized when $\lambda = g^T d/d^T Gd$, and

$$\min_{\lambda} \psi(\lambda) = \psi(0) - \frac{1}{2}\frac{(d^T g)^2}{d^T Gd}.$$

(b) Using the result from (a), show that if $g^T Gg > 0$ and $g^T G^{-1} g > 0$, then the steepest descent method $d = g$ with optimal λ gives a smaller reduction of ψ than Newton's method if $g^T G^{-1} g > (g^T g)^2/g^T Gg$.

(c) Suppose that G is symmetric and nonsingular. Using the result from (b), show that Newton's method will find a stationary point of ϕ in one step from an arbitrary starting point x_0. Under what condition is this a minimizer?

2.8.3 One wants to fit a circle with radius r and center (x_0, y_0) to given data (x_i, y_i), $i = 1{:}m$. The orthogonal distance from (x_i, y_i) to the circle,

$$d_i(x_0, y_0, r) = r_i - r, \qquad r_i = \left((x_i - x_0)^2 + (y_i - y_0)^2\right)^{1/2},$$

depends nonlinearly on the parameters x_0, y_0. Thus, the problem

$$\min_{x_0, y_0, r} \sum_{i=1}^{m} d_i^2(x_0, y_0, r)$$

is a nonlinear least squares problem. An approximate linear model is obtained by writing the equation of the circle $(x - x_0)^2 + (y - y_0)^2 = r^2$ in the form

$$\delta(x_0, y_0, c) = 2xx_0 + 2yy_0 + c = x^2 + y^2,$$

which depends linearly on the parameters x_0, y_0 and $c = r^2 - x_0^2 - y_0^2$. If these parameters are known, then the radius of the circle can be determined by $r = (c + x_0^2 + y_0^2)^{1/2}$.

(a) Write the overdetermined linear system $\delta_i(x_0, y_0, c) = x^2 + y^2$ corresponding to the data $(x, y) = (x_i, y_i)$, where

$$\begin{array}{cccccccc} x_i & 0.7 & 3.3 & 5.6 & 7.5 & 0.3 & -1.1 \\ y_i & 4.0 & 4.7 & 4.0 & 1.3 & -2.5 & 1.3 \end{array}$$

(b) Describe, preferably in the form of a MATLAB program, a suitable algorithm to calculate x_0, y_0, c with the linearized model. The program should function for all possible cases, e.g., even when $m < 3$.

2.8.4* Develop an algorithm for the orthogonal distance fitting of a sphere to three-dimensional data (x_i, y_i, z_i), $i = 1{:}m$. Model the algorithm after the algorithm for a circle described by (2.8.60)–(2.8.63).

References

1. Anda, A., Park, H.: Fast plane rotations with dynamic scaling. BIT **15**(3), 162–174 (1994)
2. Anderssen, E., Bai, Z., Dongarra, J.J.: Generalized QR factorization and its applications. Linear Algebra Appl. **162–164**, 243–271 (1992)
3. Andersson, C.A., Bro, R.: The *N*-way Toolbox for MATLAB. Chemom. Intell. Lab. Syst. **52**, 1–4 (2000)
4. Arioli, M.: The use of QR factorization in sparse quadratic programming and backward error issues. SIAM J. Matrix Anal. Appl. **21**(3), 825–839 (2000)
5. Autonne, L.: Sur les groupes linéaires réels et orthogonaux. Bulletin de la Société Mathématique de France **30**, 121–134 (1902)
6. Autonne, L.: Sur les matrices hypohermitiennes et sur les matrices unitaires. Univ. Lyon, Nouvelle Sér. **38**, 1–77 (1915)
7. Bader, B.W., Kolda, T.G.: Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. ACM Trans. Math. Softw. **32**(4), 455–500 (2006)
8. Bader, B.W., Kolda, T.G.: Efficient MATLAB computations with sparse and factored tensors. SIAM J. Sci. Comput. **30**(1), 205–231 (2007)
9. Barlow, J.L.: More accurate bidiagonal reduction algorithm for computing the singular value decomposition. SIAM J. Matrix Anal. Appl. **23**(3), 761–798 (2002)
10. Barlow, J.L., Erbay, H.: A modifiable low-rank approximation of a matrix. Numer. Linear Algebra Appl. **16**(10), 833–860 (2009)
11. Barlow, J.L., Bosner, N., Drmač, Z.: A new stable bidiagonal reduction algorithm. Linear Algebra Appl. **397**, 35–84 (2005)
12. Barlow, J.L., Erbay, H., Slapničar, I.: An alternative algorithm for the refinement of ULV decompositions. SIAM J. Matrix Anal. Appl. **27**(1), 198–214 (2005)
13. Barrlund, A.: Perturbation bounds on the polar decomposition. BIT **20**(1), 101–113 (1990)
14. Barrodale, I., Roberts, F.D.K.: An improved algorithm for discrete ℓ_1 linear approximation. SIAM J. Numer. Anal. **10**, 839–848 (1973)
15. Bartels, R.H., Conn, A.R., Sinclair, J.W.: Minimization techniques for piecewise differentiable functions: The l_1 solution to an overdetermined linear system. SIAM J. Numer. Anal. **15**, 224–241 (1978)

16. Ben-Israel, A., Greville, T.N.E.: Some topics in generalized inverses of matrices. In : Zuhair Nashed, M. (ed.) Generalized Inverses and Applications, pp. 125–147. Academic Press, New York (1976)
17. Ben-Israel, A., Greville, T.N.E.: Generalized Inverses: Theory and Applications. Springer, Berlin (2003)
18. Benzi, M., Golub, G.H., Liesen, J.: Numerical solution of saddle point problems. Acta Numerica **14**, 1–138 (2005)
19. Bindel, D., Demmel, J.W., Kahan, W.M., Marques, O.: On computing givens rotations reliably and efficiently. ACM Trans. Math. Softw. **28**(2), 206–238 (2002)
20. Bischof, C.H., Van Loan, C.F.: The WY representation for products of Householder matrices. SIAM J. Sci. Stat. Comput. **8**(1), 2–13 (1987)
21. Bjerhammar, A.: Rectangular reciprocal matrices with special reference to geodetic calculations. Bull. Géodésique **52**, 118–220 (1951)
22. Björck, Å.: Solving linear least squares problems by Gram-Schmidt orthogonalization. BIT **7**(1), 1–21 (1967)
23. Björck, Å.: Iterative refinement of linear least squares solutions I. BIT **7**, 257–278 (1967)
24. Björck, Å.: A bidiagonalization algorithm for solving ill-posed systems of linear equations. BIT **28**, 659–670 (1988)
25. Björck, Å.: Component-wise perturbation analysis and error bounds for linear least squares solutions. BIT **31**, 238–244 (1991)
26. Björck, Å.: Numerics of Gram-Schmidt orthogonalization. Linear Algebra Appl. **197–198**, 297–316 (1994)
27. Björck, Å.: Numerical Methods for Least Squares Problems. SIAM, Philadelphia (1996)
28. Björck, Å.: The calculation of linear least squares problems. Acta Numerica **13**, 1–54 (2004)
29. Björck, Å., Golub, G.H.: Iterative refinement of linear least squares solution by Householder transformation. BIT **7**(4), 322–337 (1967)
30. Björck, Å., Golub, G.H.: Numerical methods for computing angles between subspaces. Math. Comp. **27**(123), 579–594 (1973)
31. Björck, Å., Paige, C.C.: Loss and recapture of orthogonality in the modified Gram-Schmidt algorithm. SIAM J. Matrix Anal. Appl. **13**(1), 176–190 (1992)
32. Boggs, P.T., Byrd, R.H., Schnabel, R.B.: A stable and efficient algorithm for nonlinear orthogonal regression. SIAM J. Sci. Stat. Comput. **8**, 1052–1078 (1987)
33. Boggs, P.T., Donaldson, J.R., Byrd, R. H., Schnabel R.B.: Algorithm 676. ODRPACK.: software for weighted orthogonal distance regression. ACM Trans. Math. Softw. **15**, 348–364 (1989)
34. Bojanczyk, A.W., Brent, R.P., Dooren, P.V., de Hoog, F.: A note on downdating the Cholesky factorization. SIAM J. Sci. Stat. Comput. **8**, 210–221 (1987)
35. Bojanczyk, A.W., Higham, N.J., Patel, H.: Solving the indefinite least squares problem by hyperbolic QR factorization. SIAM J. Matrix Anal. Appl. **24**(4), 914–931 (2003)
36. Bro, R.: PARAFAC. Tutorial and applications. Chemom. Intell. Lab. Syst. **38**, 149–171 (1997). Special Issue 2nd Internet Conf. in Chemometrics (INCINC'96)
37. Bro, R., Eldén, L.: PLS works. J. Chemometrics **23**, 69–71 (2009)
38. Bunch, J.R., Nielsen, C.P.: Updating the singular value decomposition. Numer. Math. **31**, 111–129 (1978)
39. Businger, P., Golub, G.H.: Linear least squares solutions by Householder transformations. Numer. Math. **7**, 269–276 (1965). Also published as Contribution I/8 in the Handbook
40. Calvetti, D., Hansen, P.C., Reichel, L.: L-curve curvature bounds via Lanczos bidiagonalization. Electron. Trans. Numer. Anal. **14**, 134–149 (2002)
41. Candès, E.J.: Compressive sampling. In Sanz-Solé, M., Soria, J., Varona, J.L., Verdera, J. (eds.) Proceedings of the International Congress of Mathematicians, Madrid 2006. Vol. III., pp. 1433–1452. European Mathematical Society, Zürich (2006)
42. Candès, E.J., Li, X., Ma, Y., Wright, J.: Robust Principal Component Analysis. Stanford University, Stanford (2009)

43. Chan, T.: An improved algorithm for computing the singular value decomposition. ACM Trans. Math. Softw. **8**(1), 72–83 (1982)
44. Chan, T.F.: Rank revealing QR factorizations. Linear Algebra Appl. **88**(89), 67–82 (1987)
45. Chan, T.F., Foulser, D.E.: Effectively well-conditioned linear systems. SIAM J. Sci. Stat. Comput. **9**, 963–969 (1988)
46. Chan, T.F., Hansen, P.C.: Low-rank revealing QR factorizations. Numer. Linear Algebra Appl. **1**, 33–44 (1994)
47. Chan, T.F., Golub, G.H., LeVeque, R.J.: Algorithms for computing sample variances: analysis and recommendations. Am. Statistician. **37**(3), 241–247 (1983)
48. Chan, R.H., Greif, C., O'Leary, D.P. (eds.): Milestones in Matrix Computations: The Selected Works of Gene H. Golub, with Commentaries. Oxford University Press, Oxford (2007)
49. Chandrasekaran, S., Ipsen, I.C.F.: On rank-revealing factorizations. SIAM J. Matrix Anal. Appl. **15**, 592–622 (1994)
50. Chambers, J.M.: Regression updating. J. Amer. Statist. Assoc. **66**, 744–748 (1971)
51. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM Review **43**, 129–159 (2001)
52. Clark, D.I., Osborne, M.R.: Finite algorithms for Hubers's M-estimator. SIAM J. Sci. Stat. Comput., **7**:72–85 (1986)
53. Cline, A.K.: Rate of convergence of Lawson's algorithm. Math. Comp. **26**(2), 167–176 (1972)
54. Cline, A.K.: An elimination method for the solution of linear least squares problems. SIAM J. Numer. Anal. **10**(2), 283–289 (1973)
55. Cline, A.K.: The transformation of a quadratic programming problem into solvable form. Tech. Report ICASE 75–14, NASA, (Langley Research Center, Hampton, VA, 1975)
56. Comon, P., ten Berge, J.M.F., De Lathauwer, L., Castaing, J.: Generic and typical ranks of multiway arrays. Linear Algebra Applic. **430**, 2997–3007 (2009)
57. Comon, P., Golub, G.H., Lim, L.-H., Mourrain, B.: Symmetric tensors and symmetric rank. SIAM J. Matrix Anal. Appl. **30**, 1254–1279 (2008)
58. Conn, A.R., Gould, N.I.M., Toint, P.L.: Trust Region Methods. SIAM, Philadelphia, PA (2000)
59. Cox, M.G.: The least squares solution of overdetermined linear systems having band or augmented band structure. IMA J. Numer. Anal. **1**(2), 3–22 (1981)
60. Cox, M.G.: The least-squares solution of linear equations with block-angular observation matrix. In: Cox, M.G., Hammarling, S.J. (eds.) Reliable Numerical Computation, pp. 227–240. Oxford University Press, UK (1990)
61. Cox, A.J. Higham, N.J.: Stability of Householder QR factorization for weighted least squares problems. In Griffiths, D.F., Higham, D.J., Watson, G.A. (eds.), Numerical Analysis 1997: Proceedings of the 17th Dundee Biennial Conference, Pitman Research Notes in mathematics, vol. 380, pp. 57–73. Longman Scientific and Technical, Harlow, Essex, UK (1998)
62. Cox, A.J., Higham, N.J.: Backward error bounds for constrained least squares problems. BIT **39**(2), 210–227 (1999)
63. Dahlquist, G., Björck, Å.: Numerical Methods in Scientific Computing. vol. 1. SIAM, Philadelphia (2008)
64. Daniel, J.W., Gragg, W.B., Kaufman, L., Stewart, G.W.: Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. Math. Comp. **30**, 772–795 (1976)
65. Daubechies, I., DeVore, R., Fornasier, M., Güntürk, C.S.: Iteratively re-weighted least squares minimization for sparse recovery. Comm. Pure Appl. Math., **63**(1):1–38, 2010.
66. Davis, P.J.: Orthonormalizing codes in numerical analysis. In: Todd, J. (ed.) Survey of Numerical Analysis. pp. 558–584. McGraw-Hill, New York (1962)
67. Davis, T.A.: Direct methods for sparse linear systems, Fundamental of Algorithms, vol 2. SIAM, Philadelphia, PA (2006)
68. Davis, T.A.: Algorithm 915, SuiteSparseQR: multifrontal multithreaded rank-revealing sparse QR factorizatiom. ACM Trans. Math. Softw., **38**(1):8:1–8:22 (2011)
69. Demmel, J., Grigori, L., Hoemmen, M., Langou, J.: Communication-avoiding parallel and sequential QR factorizations. Technical Report UCB/EECS-2008-74, EECS Department, University of California, Berkeley (2008)

70. Demmel, J.W., Hida, Y., Riedy, E.J., Li, X.S.: Extra-precise iterative refinement for overdetermined least squares problems. ACM Trans. Math. Softw. **35**(4), 29:1–32 (2009)
71. Demmel, J.W., Hoemmen, M., Hida, Y., Riedy, E.J.: Non-negative diagonals and high performance on low-profile matrices from Householder QR. LAPACK Working Note 203 Technical Report UCB/EECS-2008-76, UCB/EECS, Berkeley (2008)
72. Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewood Cliffs, NJ, 1983. Reprinted in 1995 by SIAM, Philadelphia, PA
73. Dennis, J.E., Gay, D.M., Welsh, R.E.: An adaptive nonlinear least-squares algorithm. ACM. Trans. Math. Softw., **7**:348–368 (1981)
74. Dongarra, J.J., Bunch, J.R., Moler, C.B., Stewart, G.W.: LINPACK Users' Guide. SIAM, Philadelphia (1979)
75. Drmač, Z.: On principal angles between subspaces of euclidean space. SIAM J. Matrix Anal. Appl. **22**(1), 173–194 (2000)
76. Dubrulle, A.A.: Householder transformations revisited. SIAM J. Matrix Anal. Appl. **22**(1), 33–40 (2000)
77. Duff, I.S., Gratton, S.: The parallel algorithms team at CERFACS. SIAM News **39**, 10 (2006)
78. Dulmage, A.L., Mendelsohn, N.S.: Two algorithms for bipartite graphs. J. Soc. Indust. Appl. Math. **11**, 183–194 (1963)
79. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika **1**, 211–218 (1936)
80. Edelman, A., Arias, T., Smith, S.T.: The geometry of algorithms with orthogonality constraints. SIAM J. Matrix. Anal. Appl. **20**(2), 303–353 (1999)
81. Efron, B., Hastie, T., Johnston, I., Tibshirani, R.: Least angle regression. The Annals of Statistics **32**(2), 407–499 (2004)
82. Efroymson, M.A.: Multiple regression analysis. In: Ralston, A., Wilf, H.S. (eds.) Mathematical Methods for Digital Computers. Vol. 1, pp. 191–203. Wiley, New York (1960)
83. Ekblom, H.: A new algorithm for the Huber estimator in linear models. BIT **28**, 60–76 (1988)
84. Eldén, L.: Stepwise regression analysis with orthogonal transformations. Technical Report LiTH-MAT-R-1972-2, Department of Mathematics, Linköping University, Sweden (1972)
85. Eldén, L.: Algorithms for the regularization of ill-conditioned least squares problems. BIT **17**, 134–145 (1977)
86. Eldén, L.: A weighted pseudoinverse, generalized singular values, and constrained least squares problems. BIT **22**, 487–502 (1982)
87. Eldén, L.: An efficient algorithm for the regularization of ill-conditioned least squares problems with a triangular Toeplitz matrix. SIAM J. Sci. Stat. Comput. **5**(1), 229–236 (1984)
88. Eldén, L.: Partial least-squares vs. Lanczos bidiagonalization–I: Analysis of a projection method for multiple regression. Comput. Statist. Data Anal. **46**, 11–31 (2004)
89. Eldén, L.: Matrix Methods in Data Mining and Pattern Recognition. SIAM, Philadelphia (2007)
90. Eldén, L., Park, H.: Block downdating of least squares solutions. SIAM J. Matrix. Anal. Appl. **15**(3), 1018–1034 (1994)
91. Eldén, L., Park, H.: A Procrustes problem on the Stiefel manifold. Numer. Math. **82**, 599–619 (1999)
92. Eldén, L., Savas, B.: A Newton-Grassman method for computing the best multi-linear rank-(r_1, r_2, r_3) approximation of a tensor. SIAM J. Matrix Anal. Appl. **31**(2), 248–271 (2009)
93. Elfving, T., Skoglund, I.: A direct method for a regularized least-squares problem. Numer. Linear Algebra Appl. **16**, 649–675 (2009)
94. Elmroth, E., Gustavson, F.G.: Applying recursion to serial and parallel QR factorization leads to better performance. IBM J. Res. Develop. **44**(4), 605–624 (2004)
95. Engl, H.W., Hanke, M., Neubauer, A.: Regularization of Inverse Problems. Kluwer Academic Press, Dordrecht, The Netherlands (1995)
96. Faddeev, D.K., Kublanovskaya, V.N., Faddeeva. V.N.: Sur les systèmes linéaires algébriques de matrices rectangularies et malconditionées. In: Programmation en Mathématiques Numériques, pp. 161–170. Centre National de la Recherche Scientifique, Paris VII (1968)

97. Faddeev, D.K., Kublanovskaya, V.N., Faddeeva, V.N.: Solution of linear algebraic systems with rectangular matrices. Proc. Steklov Inst. Math. **96**, 93–111 (1968)
98. Fan, K.Y., Hoffman, A.J.: Some metric inequalities in the space of matrices. Proc. Amer. Math. Soc. **6**, 111–116 (1955)
99. Fausett, D.W., Fulton, C.T.: Large least squares problems involving Kronecker products. SIAM J. Matrix. Anal Appl. **15**, 219–227 (1994)
100. Fierro, R.D., Hansen, P.C.: UTV Expansion Pack: special-purpose rank-revealing algorithms. Numer. Algorithms **40**, 47–66 (2005)
101. Fletcher, R.: Practical Methods of Optimization. 2nd edn. Wiley, New York (1987)
102. Forsythe, G.E.: Generation and use of orthogonal polynomials for data-fitting with a digital computer. J. SIAM **5**, 74–88 (1957)
103. Foster, L.: Rank and null space calculations using matrix decomposition without column interchanges. Linear Algebra Appl. **74**, 47–91 (1986)
104. Foster, L.: Solving rank-deficient and ill-posed problems using UTV and QR factorizations. SIAM J. Matrix Anal. Appl. **25**(2), 560–582 (2003)
105. Foster, L., Kommu, R.: Algorithm 853. An efficient algorithm for solving rank-deficient least squares problems. ACM Trans. Math. Softw. **32**(1), 157–165 (2006)
106. Friedlander, A., Mart'ınez, J.M., Raydan, M.: A new method for large-scale box constrained convex quadratic minimization problems. Optim. Meth. Software, **5**:57–74 (1995)
107. Furnival, G.M., Wilson, R.W.: Regression by leaps and bounds. Technometrics **16**(4), 499–511 (1974)
108. Gander, W.: Least squares with a quadratic constraint. Numer. Math. **36**, 291–307 (1981)
109. Gander, W., Hřebiček, J.: Solving Problems in Scientific Computing using Maple and Matlab. Springer, Berlin, fourth edition (2004)
110. Gander, W., Golub, G.H., Strebel, R.: Least squares fitting of circles and ellipses. BIT **34**(4), 558–578 (1994)
111. Gauss, C.F.: Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections. Republished in 1963 by Dover, Mineola, NY (1809). C.H. Davis, Translation
112. Gauss, C.F.: The Theory of the Combination of Observations Least Subject to Errors. Part 1, Part 2. SIAM, Philadelphia, PA, G. W. Stewart, Translation 1995 (1821)
113. Gauss, C.F.: Theoria combinationis observationum erroribus minimis obnoxiae, pars prior. In: Werke, IV (ed.) pp. 1–26. Königlichen Gesellschaft der Wissenschaften zu Göttingen (1880). First published in 1821
114. Gauss, C.F.: Theoria combinationis observationum erroribus minimis obnoxiae, pars posterior. In: Werke, IV (ed.) pp. 27–53. Königlichen Gesellschaft der Wissenschaften zu Göttingen (1880). First published in 1823
115. Gentleman, W.M.: Least squares computations by givens transformations without square roots. J. Inst Math. Applics. **12**, 329–336 (1973)
116. George, A., Heath, M.T.: Solution of sparse linear least squares problems using Givens rotations. Linear Algebra Appl. **34**, 69–83 (1980)
117. George, A., Liu, J.W.H.: Householder reflections versus givens rotations in sparse orthogonal decomposition. Linear Algebra Appl. **88**(89), 223–238 (1987)
118. George, A., Ng, E.: On row and column orderings for sparse least squares problems. SIAM J. Numer. Anal. **20**, 326–344 (1983)
119. George, A.J., Ng, E.G.: Symbolic factorization for sparse Gaussian elimination with partial pivoting. SIAM J. Sci. Stat. Comput. **8**(6), 877–898 (1987)
120. Gilbert, J.R., Ng, E., Peyton, B.W.: Separators and structure prediction in sparse orthogonal factorization. Linear Algebra Appl. **262**, 83–97 (1997)
121. Gill, P.E., Hammarling, S.J., Murray, W., Saunders, M.A., Wright, M.H.: User's guide for LSSOL (version 1.0): a Fortran package for constrained linear least-squares and convex quadratic programming. Report SOL, Department of Operations Research, Stanford University, CA (1986)
122. Giraud, L., Langou, J., Rozložnık, M.: The loss of orthogonality in the Gram-Schmidt orthogonalization process. Comput. Math. Applics. **50**, 1069–1075 (2005)

123. Givens, W.G.: Computation of plane unitary rotations transforming a general matrix to triangular form. SIAM J. Appl. Math. **6**(1), 26–50 (1958)
124. Gohberg, I., Lancaster, P., Rodman, L.: Indefinite Linear Algebra and Applications. Birkhäuser, Boston (2005)
125. Goldstine, H.H.: A History of Numerical Analysis from the 16th through the 19th Century. Stud. Hist. Math. Phys. Sci. vol. 2. Springer, New York (1977)
126. Golub, G.H.: Numerical methods for solving least squares problems. Numer. Math. **7**, 206–216 (1965)
127. Golub, G.H., Kahan, W.: Calculating the singular values and pseudoinverse of a matrix. SIAM J. Numer. Anal. Ser. B **2**, 205–224 (1965)
128. Golub, G.H., LeVeque, R.J.: Extensions and uses of the variable projection algorithm for solving nonlinear least squares problems. In Proceedings of the 1979 Army Numerical Analysis and Computers Conf., pp. 1–12. White Sands Missile Range, White Sands, NM, ARO Report 79–3 (1979)
129. Golub, G.H., Plemmons, R.J.: Large-scale geodetic least-squares adjustment by dissection and orthogonal decomposition. Linear Algebra Appl. **34**, 3–28 (1980)
130. Golub, G.H., Pereyra, V.: The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. SIAM J. Numer. Anal. **10**(2), 413–432 (1973)
131. Golub, G.H., Pereyra, V.: Separable nonlinear least squares: the variable projection method and its application. Inverse Problems **19**, R1–R26 (2003)
132. Golub, G.H., Van Loan, C.F.: An analysis of the total least squares problem. SIAM J. Numer. Anal. **17**(6), 883–893 (1980)
133. Golub, G.H., Van Loan, C.F.: Matrix Computations. 3rd edn. Johns Hopkins University Press, Baltimore (1983)
134. Golub, G.H., Zha, H.: Perturbation analysis of the canonical correlation of matrix pairs. Linear Algebra Appl. **210**, 3–28 (1994)
135. Golub, G.H., Heath, M.T., Wahba, G.: Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics **21**, 215–223 (1979)
136. Golub, G.H., Hoffman, A., Stewart, G.W.: A generalization of the Eckart-Young matrix approximation theorem. Linear Algebra Appl. **88**(89), 317–327 (1987)
137. Gragg, W.B., Leveque, R.J., Trangenstein, J.A.: Numerically stable methods for updating regressions. J. Amer. Statist. Org. **74**(365), 161–168 (1979)
138. Gram, J.P.: Om Raekkenudviklinger bestemte ved Hjaelp af de mindste Kvadraters Methode. Andr. Fred. Host & Son, Copenhagen (1879)
139. Gratton, S.: On the condition number of the least squares problem in a weighted Frobenius norm. BIT **36**(3), 523–530 (1996)
140. Grcar, J.F.: Spectral condition numbers of orthogonal projections and full rank linear least squares residuals. SIAM J. Matrix Anal. Appl. **31**(5), 2934–2949 (2010)
141. Groetsch, C.W.: The Theory of Tikhonov Regularization for Fredholm Integral Equations of the First Kind. Pitman, Boston, MA, 1984.
142. Gu, M.: Backward perturbation bounds for linear least squares problems. SIAM J. Matrix. Anal. Appl. **20**(2), 363–372 (1998)
143. Gulliksson, M.E., Söderkvist, I., Wedin, P.-Å.: Algorithms for constrained and weighted nonlinear least squares. SIAM J. Optim. **7**, 208–224 (1997)
144. Guttman, I., Pereyra, V., Scolnik, H.D.: Least squares estimation for a class of nonlinear models. Technometrics **15**(2), 309–318 (1973)
145. Hald, A.: Statistical Theory with Engineering Applications. Wiley, New York (1952). Transl. by G. Seidelin
146. Hammarling, S.J.: A note of modifications to the Givens plane rotations. J. Inst. Math. Applics. **13**, 215–218 (1974)
147. Hammarling, S.J.: The numerical solution of the general Gauss-Markov linear model. In: Durrani, T.S., Abbiss, J.B., Hudson, J.E., (eds) Mathematics in Signal Processing, pp. 451–456. Oxford University Press (1987)

148. Hansen, P.C.: The discrete Picard condition for discrete ill-posed problems. BIT **30**(4), 658–672 (1990)
149. Hansen, P.C.: Analysis of discrete ill-posed problems by means of the L-curve. SIAM Review **34**(4), 561–580 (1992)
150. Hansen, P.C.: Rank-Deficient and Discrete Ill-Posed Problems. Numerical Aspects of Linear Inversion. SIAM, Philadelphia (1998)
151. Hansen, P.C.: Regularization tools version 4.0 for matlab 7.3. Numerical Algorithms **46**, 189–194 (2007)
152. Hansen, P.C., O'Leary, D.P.: The use of the L-curve in the regularization of discrete ill-posed problems. SIAM J. Sci. Stat. Comput. **14**(6), 1487–1503 (1993)
153. Hansen, P.C., Yalamov, P.Y.: Computing symmetric rank-relieving decompositions via triangular factorizations. SIAM J. Matrix Anal. Appl. **23**(2), 443–458 (2001)
154. Hansen, P.C., Pereyra, V., Scherer, G.: Least Squares Data Fitting with Applications. The Johns Hopkins University Press, Baltimore (2013)
155. Hanson, R.J., Lawson, C.L.: Extensions and applications of the Householder algorithm for solving linear least squares problems. Math. Comp. **23**, 787–812 (1969)
156. Hanson, R.J., Norris, M.J.: Analysis of measurements based on the singular value decomposition. SIAM J. Sci. Stat. Comput. **2**(3), 363–373 (1981)
157. Hare, D.P., Johnson, C.R., Olesky, D.D., Van Der Driessche, P.: Sparsity analysis of the QR factorization. SIAM J. Matrix. Anal. Appl. **14**(2), 655–659 (1993)
158. Hebden, M.D.: An algorithm for minimization using exact second derivatives. Tech. Report T. P. 515, Atomic Energy Research Establishment, Harwell, England, 1973.
159. Helmert, F.R.: Die Mathematischen und Physikalischen Theorien der höheren Geodäsie, 1 Teil. B. G. Teubner, Leipzig (1880)
160. Hernandez, V., Román, J.E., Tomás, A.: A parallel variant of the Gram-Schmidt process with reorthogonaliztion. In: Joubert, G.R., Nagel, W.E., Peters, F.J., Plata, O.G., Zapata, E.L. (eds.) Parallel Computing: Current & Future Issues in High-End Computing. John von Neumann Institute for Computing Series, vol. 33, pp. 221–228. Central Institute for Applied Mathematics, Jülich, Germany (2006)
161. Higham, N.J.: Computing the polar decomposition–with applications. SIAM J. Sci. Stat. Comput. **7**(4), 1160–1174 (1986)
162. Higham, N.J.: Accuracy and Stability of Numerical Algorithms. 2nd edn. SIAM, Philadelphia (2002)
163. Higham, N.J.: J-Orthogonal matrices: Properties and generation. SIAM Review **45**(3), 504–519 (2003)
164. Hitchcock, F.L.: The expression of a tensor or a polyadic as a sum of products. J. Math. Phys. **7**, 164–189 (1927)
165. Hoffmann, W.: Iterative algorithms for Gram-Schmidt orthogonalization. Computing **41**, 335–348 (1989)
166. Hong, Y.T., Pan, C.T.: Rank-revealing QR decompositions and the singular value decomposition. Math. Comp. **58**, 213–232 (1992)
167. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis. Cambridge University Press, Cambridge (1991)
168. Hotelling, H.: Relation between two sets of variables. Biometrica **28**, 322–377 (1936)
169. Householder, A.S.: Unitary triangularization of a nonsymmetric matrix. J. Assoc. Comput. Mach. **5**, 339–342 (1958)
170. Huber, P.J.: Robust Statistics. Wiley, New York (1981)
171. Huckaby, D.A., Chan, T.F.: On the convergence of Stewart's QLP algorithm for approximating the SVD. Numer. Algorithms **32**(2–4), 387–316 (2003)
172. Jacobi, C.G.J.: Über eine neue Auflösungsart der bei der Methode der kleinsten Quadrate vorkommenden linearen Gleichungen. Astronomische Nachrichten **22**(523), 297–306 (1845)
173. Jordan, C.: Essai sur la géométrie à n dimensions. Bull. Soc. Math. France **3**, 103–174 (1875)
174. Kahan, W.M.: Numerical linear algebra. Canad. Math. Bull. **9**, 757–801 (1966)

175. Karlsson, R., Waldén, B.: Estimation of optimal backward perturbation bounds for the linear least squares problem. BIT **37**(4), 862–869 (1997)
176. Kaufman, L.: Variable projection methods for solving separable nonlinear least squares problems. BIT **15**, 49–57 (1975)
177. Kaufman, L., Pereyra, V.: A method for separable nonlinear least squares problems with separable nonlinear equality constraints. SIAM J. Numer. Anal. **15**(1), 12–20 (1978)
178. Kaufman, L., Sylvester, G.: Separable nonlinear least squares problems with multiple right hand sides. SIAM J. Matrix Anal. Appl. **13**, 68–89 (1992)
179. Knyazev, A.V., Argentati, M.E.: Principal angles between subspaces in an *A*-based scalar product: Algorithms and perturbation estimates. SIAM J. Sci. Comput. **23**(6), 2008–2040 (2002)
180. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)
181. Komarek, P.: Logistic Regression for Data Mining and High-Dimensional Classification. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2004.
182. Kourouklis, S., Paige, C.C.: A constrained approach to the general Gauss-Markov linear model. J. Amer. Statist. Assoc., 76:620–625, 1981.
183. Lanczos, C.: Analysis applied. Prentice-Hall, Englewood Cliffs, NJ (1956). Reprinted by Dover. Mineola, NY, 1988
184. Lanczos, C.: Linear systems in self-adjoint form. Amer. Math. Monthly **65**, 665–671 (1958)
185. Langou, J.: Translation and modern interpretation of Laplace's Théorie Analytique des Probabilités, pp. 505–512, 516–520. Technical Report 280, UC Denver CCM (2009)
186. Laplace, P.-S.: Théorie analytique des probabilités. Premier supplément. 3rd edn.. Courcier, Paris (1820). With an introduction and three supplements
187. Lathauwer, L.D., Moor, B.D., Vandewalle, J.: A multilinear singular value decomposition. SIAM J. Matrix Anal. Appl. **21**(4), 1253–1278 (2000)
188. Läuchli, P.: Jordan-Elimination und Ausgleichung nach kleinsten Quadraten. Numer. Math. **3**, 226–240 (1961)
189. Lawson, C.L.: Contributions to the theory of linear least maximum approximation. PhD thesis, University of California, Los Angeles, (1961)
190. Lawson, C.L., Hanson, R.J.: Solving Least Squares Problems. Prentice-Hall, Englewood Cliffs (1974). xii+337 pp. Revised republication by SIAM, Philadelphia, PA, 1995
191. Lehoucq, R.B.: The computations of elementary unitary matrices. ACM Trans. Math. Softw. **22**, 393–400 (1996)
192. Leurgans, S.E., Ross, R.T., Abel, R.B.: A decomposition for three-way arrays. SIAM J. Matrix Anal. Appl. **14**(4), 1064–1083 (1993)
193. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Quart. Appl. Math. **2**, 164–168 (1944)
194. Li, Y.: A globally convergent method for l_p problems. SIAM J. Optim. **3**, 609–629 (1993)
195. Lim, L.-H.: Tensor and hypermatrices. In: Hogben, L. (ed.) Handbook of Linear Algebra, 2nd edn, pp. 15.1–15.30. Chapman & Hall/CRC Press, Boca Raton (2013)
196. Liu, J.W.H.: On general row merging schemes for sparse Givens transformations. SIAM J. Sci. Stat. Comput. **7**, 1190–1211 (1986)
197. Liu, J.W.H.: The role of elimination trees in sparse matrix factorization. SIAM J. Matrix Anal. Appl. **11**(1), 134–172 (1990)
198. Lötstedt, P.: Perturbation bounds for the linear least squares problem subject to linear inequality constraints. BIT **23**, 500–519 (1983)
199. Lötstedt, P.: Solving the minimal least squares problem subject to bounds on the variables. BIT **24**, 206–224 (1984)
200. Lu, S.-M., Barlow, J.L.: Multifrontal computation with the orthogonal factors of sparse matrices. SIAM J. Matrix Anal. Appl., **17**:658–679 (1996)
201. Madsen, K., Nielsen, H.B.: Finite algorithms for robust linear regression. BIT **30**(4), 682–699 (1990)

202. Madsen, K., Nielsen, H.B.: A finite smoothing algorithm for linear ℓ_1 estimation. SIAM J. Opt. **3**(2), 223–235 (1993)
203. Malyshev, A.N.: A unified theory of conditioning for linear least squares and Tikhonov regularization solutions. SIAM J. Matrix Anal. Appl. **24**(4), 1186–1196 (2003)
204. Markov, A.A.: Wahrscheinlichheitsrechnung. 2nd edn. Leipzig, Liebmann (1912)
205. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Indust. Appl. Math. **11**, 431–441 (1963)
206. Mathias, R., Stewart, G.W.: A block QR algorithm and the singular value decompositions. Linear Algebra Appl. **182**, 91–100 (1993)
207. Miller, A.J.: Subset Selection in Regression. Monograph on Statistics and Applied Probability. 2nd edn. Chapman & Hall/CRC Press, Boca Raton (2002)
208. Miller, K.S.: Complex least squares. SIAM Review **15**(4), 706–726 (1973)
209. Mirsky, L.: Symmetric gauge functions and unitarily invariant norms. Quart. J. Math. Oxford **11**, 50–59 (1960)
210. Moore, E.H.: On the reciprocal of the general algebraic matrix. Bull. Amer. Math. Soc. **26**, 394–395 (1920)
211. Moor, B.D., Moonen, M. (eds.): SVD and Signal Processing, III: Algorithms. Analysis and Architectures. Elsevier Science Publishers, North Holland, Amsterdam (1995)
212. Morozov, V.A.: Methods for Solving Incorrectly Posed Problems. Springer, New York (1984)
213. Nagy, J.G.: Fast inverse QR factorization for Toeplitz matrices. SIAM J. Sci. Comput. **14**, 1174–1193 (1993)
214. Nashed, M.Z.: Generalized Inverses and Applications. Publication of the Mathematics Research Center, The University of Wisconsin-Madison, No. 32. Academic Press, New York (1976)
215. Noble, B.: Methods for computing the Moore-Penrose generalized inverse and related matters. In M. Zuhair Nashed, editor, Generalized Inverses and Applications, pages 245–302. Academic Press, New York, 1976.
216. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research. 2nd edn. Springer, New York (2006)
217. O'Leary, D.P.: Robust regression computation using iteratively reweighted least squares. SIAM J. Matrix Anal. Appl. **11**, 466–480 (1990)
218. O'Leary, D.P, Whitman, P.: Parallel QR factorization by Householder and modified Gram-Schmidt algorithms. Parallel Comput. **16**(1), 99–112 (1990)
219. Oliveira, S., Borges, L., Holzrichter, M., Soma, T.: Analysis of different partitioning schemes for parallel Gram-Schmidt algorithms. Internat. J. Parallel, Emergent Distr. Syst. **14**(4), 293–320 (2000)
220. Olszanskyj, S.J., Lebak, J.M., Bojanczyk, A.W.: Rank-*k* modification methods for recursive least squares problems. Numer. Algorithms **7**, 325–354 (1994)
221. Osborne, M.R.: Some special nonlinear least squares problems. SIAM J. Numer. Anal. **12**(5), 571–592 (1975)
222. Osborne, M.R.: Finite Algorithms in Optimization and Data Analysis. Wiley, New York (1985)
223. Osborne, M.R., Smyth, G.K.: A modified Prony algorithm for exponential function fitting. SIAM J. Sci. Comput. **16**, 119–138 (1995)
224. Osborne, M.R., Presnell, B., Turlach, B.A.: A new approach to variable selection in least squares problems. IMA J. Numer. Anal. **20**, 389–404 (2000)
225. Paige, C.C.: An error analysis of a method for solving matrix equations. Math. Comp. **27**(122), 355–359 (1973)
226. Paige, C.C.: Computer solution and perturbation analysis of generalized linear least squares problems. Math. Comp. **33**, 171–184 (1979)
227. Paige, C.C.: Some aspects of generalized QR factorizations. In M. G. Cox and Sven J. Hammarling, editors, Reliable Numerical Computation, pp. 71–91. Clarendon Press, Oxford, UK (1990)
228. Paige, C.C., Strakoš, Z.: Core problems in linear algebraic systems. SIAM J. Matrix. Anal. Appl. **27**(2), 861–875 (2006)

229. Parlett, B.N.: Analysis of algorithms for reflections in bisectors. SIAM Rev. **13**, 197–208 (1971)
230. Parlett, B.N.: The Symmetric Eigenvalue Problem. SIAM, Philadelphia (1980). Republished amended version of original published by Prentice-Hall. Englewood Cliffs
231. Penrose, R.: A generalized inverse for matrices. Proc. Cambridge Philos. Soc. **51**, 406–413 (1955)
232. Pereyra, V.: Iterative methods for solving nonlinear least squares problems. SIAM J. Numer. Anal. **4**(1), 27–36 (1967)
233. Pereyra, V., Scherer, G.: Exponential data fitting. In: Pereyra, Victor, Scherer, Godela (eds.) Exponential Data Fitting and its Applications. Bentham Books, Oak Park, IL (2010)
234. Peters, G., Wilkinson, J.H.: The least squares problem and pseudo-inverses. Comput. J. **13**, 309–316 (1970)
235. Placket, R.L.: A historical note on the method of least squares. Biometrika **36**, 456–460 (1949)
236. Placket, R.L.: The discovery of the method of least squares. Biometrika **59**, 239–251 (1972)
237. Pothen, A., Fan, C.J.: Computing the block triangular form of a sparse matrix. ACM Trans. Math. Softw. **16**, 303–324 (1990)
238. Powell, M.J.D., Reid, J.K.: On applying Householder's method to linear least squares problems. In: Morell, A.J.H. (ed.), Information Processing 68. Proceedings of the IFIP Congress 68, pp. 122–126. North-Holland, Amsterdam (1969)
239. Prony, G.R.d.: Essai éxperimental et analytique: sur les lois de al dilatabilité de fluids élastique et sur celles de la force expansive de la vapeur de l'alkool à différentes temperatures. J. de l'École Polytechnique, 1:24–76, 1795.
240. Reichel, L., Gragg, W.B.: FORTRAN subroutines for updating the QR decomposition. ACM Trans. Math. Softw. **16**, 369–377 (1990)
241. Reid, J.K.: A note on the least squares solution of a band system of linear equations by Householder reductions. Comput J. **10**, 188–189 (1967)
242. Reinsch, C.H.: Smoothing by spline functions. Numer. Math. **16**, 451–454 (1971)
243. Rice, J.R.: A theory of condition. SIAM J. Numer. Anal. **3**(2), 287–310 (1966)
244. Ruhe, A.: Accelerated Gauss-Newton algorithms for nonlinear least squares problems. BIT **19**, 356–367 (1979)
245. Ruhe, A.: Fitting empirical data by positive sums of exponentials. SIAM J. Sci. Stat. Comput. **1**, 481–498 (1980)
246. Ruhe, A., Wedin, P.-Å.: Algorithms for separable nonlinear least squares problems. SIAM Review **22**(3), 318–337 (1980)
247. Rutishauser, H.: Description of Algol 60. Handbook for Automatic Computation, vol. 1a. Springer, Berlin (1967)
248. Saunders, M.A.: Large-scale linear programming using the Cholesky factorization. Technical Report CS252, Computer Science Department, Stanford University, CA (1972)
249. Sautter, W.: Fehleranalyse für die Gauss-Elimination zur Berechnung der Lösung minimaler Länge. Numer. Math. **30**, 165–184 (1978)
250. Savas, B., Lim, L.-H.: Quasi-Newton methods on Grassmannians and multilinear approximations of tensors. SIAM J. Sci. Comput. **32**(6), 3352–3393 (2010)
251. Schmidt, E.: Zur Theorie der linearen und nichtlinearen Integralgleichungen. 1 Teil. Entwicklung willkürlichen Funktionen nach Systemen vorgeschribener. Math. Ann. **63**, 433–476 (1907)
252. Schönemann, W.: A generalized solution of the orthogonal Procrustes problem. Psychometrica. **31**, 1–10 (1966)
253. Schreiber, R., Van Loan, C.: A storage efficient WY representation for products of Householder transformations. SIAM J. Sci. Stat. Comput. **10**(1), 53–57 (1989)
254. Schwetlick, H., Tiller, V.: Numerical methods for estimating parameters in nonlinear models with errors in the variables. Technometrics **27**, 17–24 (1985)
255. Schwetlick, H.: Nonlinear parameter estimation: Models, criteria and estimation. In: Griffiths, D.F., Watson, G.A. (eds.) Numerical Analysis 1991. Proceedings of the 14th Dundee Conference on Numerical Analysis, Pitman Research Notes in mathematics, vol. 260, pp. 164–193. Longman Scientific and Technical, Harlow, Essex, UK (1992)

256. Silva, V.D., Lim, L.H.: Tensor rank and the ill-posedness of the best low rank approximation. SIAM J. Matrix Anal. Appl. **30**(3), 1084–1127 (2008)
257. van der Sluis, A.: Stability of the solutions of linear least squares problems. Numer. Math. **23**, 241–254 (1975)
258. van der Sluis, A., Veltkamp, G.: Restoring rank and consistency by orthogonal projection. Linear Algebra Appl. **28**, 257–278 (1979)
259. Smoktunowicz, A., Barlow, J.L., Langou, J.: A note on the error analysis of the classical Gram-Schmidt. Numer. Math. **105**, 299–313 (2006)
260. Söderkvist, I.: Perturbation analysis of the orthogonal Procrustes problem. BIT **33**(4), 687–694 (1993)
261. Söderkvist, I., Wedin, P.-Å.: Determining the movements of the skeleton using well-configured markers. J. Biomech. **26**(12), 1473–1477 (1993)
262. Söderkvist, I., Wedin, P.-Å.: On condition numbers and algorithms for determining a rigid body movements. BIT **34**(3), 424–436 (1994)
263. Stewart, G.W.: Introduction to Matrix Computations. Academic Press, New York (1973)
264. Stewart, G.W.: The economical storage of plane rotations. Numer. Math. **25**, 137–138 (1976)
265. Stewart, G.W.: On the perturbation of pseudoinverses, projections and linear least squares problems. SIAM Rev. **19**(4), 634–662 (1977)
266. Stewart, G.W.: Research, development, and LINPACK. In: Rice, J.R. (ed.) Mathematical Software III, pp. 1–14. Academic Press, New York (1977)
267. Stewart, G.W.: On the efficient generation of random orthogonal matrices with an application to condition estimators. SIAM J. Numer. Anal. **17**(3), 403–409 (1980)
268. Stewart, G.W.: An updating algorithm for subspace tracking. IEEE Trans. Signal Process. **40**, 1535–1541 (1992)
269. Stewart, G.W.: Updating a rank-revealing ULV decomposition. SIAM J. Matrix Anal. Appl. **14**, 494–499 (1993)
270. Stewart, G.W.: On the early history of the singular value decomposition. SIAM Rev. **35**(4), 551–556 (1993)
271. Stewart. G.W.: Gauss, statistics, and Gaussian elimination. In: Computing Science and Statistics: Computational intensive Statistical Methods, editors, Handbook for Automatic Computation. vol. 2, pp. 1–7. Interface Foundation of America, Fairfax Station (1994)
272. Stewart, G.W.: The QLP approximation to the singular value decomposition. SIAM J. Sci. Comput. **20**(4), 1336–1348 (1999)
273. Stewart, G.W., Sun, J.-G.: Matrix Perturbation Theory. Academic Press, New York (1990)
274. Stewart, M., Stewart, G.W.: On hyperbolic triangularization: Stability and pivoting. SIAM J. Matrix Anal. Appl. **19**(4), 8471–860 (1998)
275. Stigler, S.M.: Gauss and the invention of least squares. Ann. Statist. **9**, 465–474 (1981)
276. Stigler, S.M.: The History of Statistics. The Measurement of Uncertainty Before 1900. The Belknap Press of Harward University Press, Cambridge (1986)
277. Stoer, J.: On the numerical solution of constrained least squares problems. SIAM J. Numer. Anal. **8**, 382–411 (1971)
278. Sun, J.-G., Sun, Z.: Optimal backward perturbation bounds for underdetermined systems. SIAM J. Matrix Anal. Appl. **18**(2), 393–402 (1997)
279. Tibshirani, R.: Regression shrinkage and selection via the LASSO. Royal Statist. Soc. B **58**(1), 267–288 (1996)
280. Tikhonov, A.N.: Solution of incorrectly formulated problems and the regularization method. Soviet Math. Dokl. **4**, 1035–1038 (1963)
281. Trefethen, L.N., Bau, III, D.: Numerical Linear Algebra. SIAM, Philadelphia (1997)
282. Tsatsomeros, M.J.: Principal pivot transforms. Linear Algebra Appl. **307**, 151–165 (2000)
283. Tucker, L.R.: Some mathematical notes on three-mode factor analysis. Psychometrica **31**, 279–311 (1966)
284. Vaccaro, R. (ed.): SVD and Signal Processing, II: Algorithms. Analysis and Applications. Elsevier Science Publishers, North Holland, Amsterdam (1991)

285. Van Huffel, S.: Partial singular value decomposition algorithm. J. Comp. Appl. Math. **33**(1), 105–112 (1990)
286. Van Huffel, S., Vandewalle, J.: The Total Least Squares Problem. Computational Aspects and Analysis. SIAM, Philadelphia, PA (1991)
287. Varah, J.M.: On the numerical solution of ill-conditioned linear systems with application to ill-posed problems. SIAM J. Numer. Anal. **10**(2), 257–267 (1973)
288. Varah, J.M.: Pitfalls in the numerical solution of linear ill-posed problems. SIAM J. Sci. Stat. Comput. **4**, 164–176 (1983)
289. Varah, J.M.: Least squares data fitting with implicit functions. BIT **36**(4), 842–854 (1996)
290. Waldén, B., Karlsson, R., Sun, J.-G.: Optimal backward perturbation bounds for the linear least squares problem. Numer. Linear Algebra Appl. **2**, 271–286 (1995)
291. Wedin, P.-Å.: On pseudo-inverses of perturbed matrices. Technical Report, Department of Computer Science. Lund University, Sweden (1969)
292. Wedin, P.-Å.: Perturbation theory for pseudo-inverses. BIT **13**, 217–232 (1973)
293. Wedin, P.-Å.: On the Gauss-Newton method for the nonlinear least squares problem. Working Paper 24, Institute for Applied Mathematics, Stockholm, Sweden (1974)
294. Weyl, H.: Das asymtotische Verteilungsgesetz der Eigenwerte linearer partieller Differnstialgleichungen. Math. Ann. **71**, 441–469 (1911)
295. Wilkinson, J.H.: Error analysis of transformations based on the use of matrices of the form $I - 2ww^{\mathrm{H}}$. In: Rall, L.B. (ed.) Error in Digital Computation. vol 2, pp. 77–101. Wiley, New York (1965)
296. Wilkinson, J.H.: The Algebraic Eigenvalue Problem. Clarendon Press, Oxford (1965)
297. Wold, H.: Estimation of principal components and related models by iterative least squares. In: Krishnaiah, P.R. (ed.) Multivariate Analysis, pp. 391–420. Academic Press, New York (1966)
298. Wold, S., Sjöström, M., Eriksson, L.: PLS-regression: A basic tool of chemometrics. Chemom. Intell. Lab. Syst. **58**, 109–130 (2001)
299. Wold, S., Ruhe, A., Wold, H., Dunn, W.J.: The collinearity problem in linear regression, the partial least squares (PLS) approach to generalized inverses. SIAM J. Sci. Stat. Comput., 5:735–743 (1984)

http://www.springer.com/978-3-319-05088-1

Numerical Methods in Matrix Computations
Björck, Å.
2015, XVI, 800 p. 45 illus., 26 illus. in color., Hardcover
ISBN: 978-3-319-05088-1