The *least squares (LS) problem* consists in, given an $m \times n$ matrix A and an m-vector b over the reals, finding an n-vector x_{LS} over the reals that minimizes the quantity

$$\|A\,x - b\|_2. \tag{5.2}$$

In other terms, the LS problem seeks the linear combination of the columns of A

$$A\,x = \sum_{j=1}^{n} x_j \,\mathrm{col}_j(A) \in \mathbb{R}^m$$

that best approaches b with respect to the 2-norm.

In data analysis, the pair (A, b) arises from measuments on a series of samples. The data obtained from the samples is arranged in the rows of A, whereas the columns of A correspond to the different attibutes or variables measured and b represents a further variable that purportedly depends on the other ones. Since the n-vector x_{LS} gives the linear combination of the variables in A that best approaches b, it can be used as a predictor to extrapolate the measurements on these samples to other cases.

Example 5.4.1. Suppose that we are doing medical research on the effect of a drug on the level of a certain toxin in the blood. To this aim, we extract the following data from a set of m patients:

age, weight, initial toxin level, given amount of drug, final toxin level.

Then we consider the LS problem for the $m \times 4$ matrix A whose (i, j)-entry is the measurement for the i-th patient of the j-th variable, and together with the m-vector b whose i-th entry is the measurement for the i-th patient of the last variable.

For this dataset, the solution x_{LS} gives the best linear approximation for final toxin level in terms of the other variables. For a further patient with age, weight, initial toxin level and amount of drug stored in a vector $p \in \mathbb{R}^4$, his/her final toxin level q would be predicted as

$$q = p^T\, x_{\mathrm{LS}}.$$

A similar situation arises when fitting a 2-dimensional plot with polynomials of a certain degree.

Example 5.4.2. Suppose that we have a sequence of points in the plane

$$(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_m, y_m)$$

and we want to find the polynomial of degree $\leq d$ that best fits the y_i's as a function of the x_i's.

This problem boils down to computing a $d+1$-vector α such that the polynomial

$$p_\alpha(x) = \sum_{j=0}^{d} \alpha_{j-1}\, x^j$$

minimizes the *residuals* $p(x_i) - y_i$, $i = 1, \ldots, m$. Minimizing the 2-norm of the vectors of these residuals is an LS problem for the data

$$A = \begin{bmatrix} 1 & x_1 & \cdots & x_1^d \\ \vdots & \vdots & & \vdots \\ 1 & x_m & \cdots & x_m^d \end{bmatrix} \in \mathbb{R}^{m\times(d+1)} \quad \text{and} \quad b = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m,$$

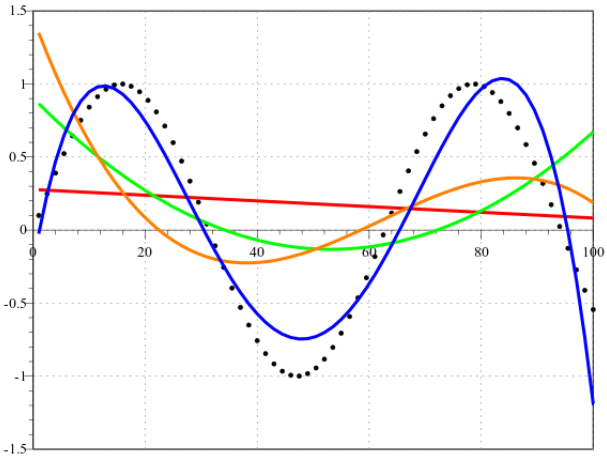

FIGURE 5.4.1. Curve fitting by polynomials of degree ≤ 4

since

$$\|A\,\alpha - b\|_2 = \Big(\sum_{i=1}^{m}(p_\alpha(x_i) - y_i)^2\Big)^{1/2}.$$

If $m = n$ and A is nonsingular then x_{LS} is the unique solution of $A\,x = b$. However, if $m > n$ then typically this linear equation has no solution, and the minimum in (5.2) is positive. It should be clear from the previous applications that this is our case of interest and as a matter of fact, in these application the number of rows m is usually much larger than the number of columns n.

The choice of the 2-norm for measuring the residual $A\,x - b$ is due to mathematical reasons: it places the problem within the realm of Euclidean geometry, which allows to use notions and tools like angles and inner products, orthogonal projections, orthonormal basis and so on. However, other norms might be equally valid and interesting from the point of view of applications. In particular, the minimization of the residual vector with respect to the 1-norm is also very relevant for applications like compressed sensing for sparse signal reconstruction.

We will study three methods for solving the LS problem:

- normal equations,
- QR factorization,
- singular value decomposition.

Each of them has its own interest and range of applicability, indicated by its time and space complexities, and its numerical stability. We will concentrate on the full rank case, but we will also explain how to treat the rank deficient case.

CHAPTER 6

The QR factorization

6.1. Normal equations

Let $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, and consider the LS problem consisting in finding a vector $x_{\rm LS} \in \mathbb{R}^n$ minimizing the 2-norm of the residual, that is

$$\|A\,x - b\|_2. \tag{6.1}$$

For the moment we concentrate on the *full rank case*, which occurs when $m \geq n$ and $\mathrm{rank}(A) = n$.

Figure 6.1 illustrates this problem and its solution. For $x \in \mathbb{R}^n$ the point $p = Ax$ lies in the image of the linear map L_A, and $e = b - p$ is the residual. The norm of this vector is minimal when p is the orthogonal projection of b into $\mathrm{Im}(L_A)$, in which case e is orthogonal to this linear subspace.

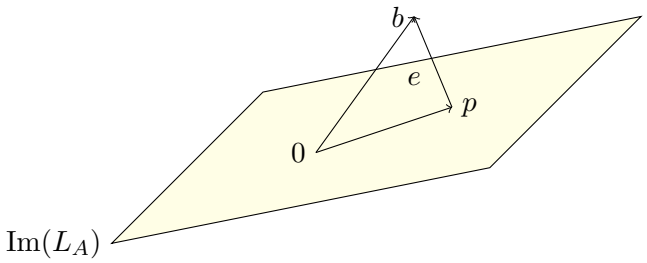

FIGURE 6.1.1. The least squares problem

Hence for $x \in \mathbb{R}^n$ we have that

$$0 = \langle Ax, e\rangle = (A\,x)^T e = (A\,x)^T(b - A\,x_{\rm LS}) = x^T A^T(b - A\,x_{\rm LS}).$$

Since this holds for all x, it implies that

$$A^T A\,x_{\rm LS} = A^T b. \tag{6.2}$$

The $n \times n$ matrix $A^T A$ is symmetric and positive definite *(check it!)*. In particular, it is nonsingular and so the solution of the *normal equations* (6.2) is unique. By Pythagoras' theorem, the norm of the residual is

$$\|e\|_2 = (\|b\|_2^2 - \|A\,x_{\rm LS}\|_2^2)^{1/2}. \tag{6.3}$$

Since $A^T A$ is an SPD matrix, we can use its Cholesky factorization to solve the normal equations.

This procedure is convenient since it relies on standard algorithms. Its complexity is

$$\left(m + \frac{n}{3}\right) n^2 + O(n^2) \text{ flops.}$$

Algorithm 6.1.1 (Normal equations)

Compute (the lower triangular part of) $C \leftarrow A^T A$,
Compute $d \leftarrow A^T b$,
Compute the Cholesky factorization $C = G\,G^T$,
Solve $G\,y = d$ and $G^T x_{\rm LS} = y$.

For $m \gg n$ the complexity $m\,n^2$ of computing the lower triangular part of the product $A^T A$ dominates. The procedure is reliable when A is far from rank deficient, but unstable otherwise.

6.2. The QR factorization

Let $a_j = \mathrm{col}_j(A) \in \mathbb{R}^m$, $j = 1, \ldots, n$, be the columns of $m \times n$ matrix A. If these vectors happen to be orthonormal, that is if

$$\langle a_i, a_k \rangle = \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{otherwise,} \end{cases}$$

then $A^T A = \mathbb{1}_n$ and so by (6.2) the solution of the LS problem is easily computed as $x_{\rm LS} = A^T b$.

In general, the columns of A are linearly independent because this matrix has full rank, but typically they are not orthogonal. However these vectors can be orthogonalized with the *Gram-Schmidt orthogonalization (GSO) process*. This algorithm produces an orthonormal family of m-vectors q_j, $j = 1, \ldots, n$, such that

$$\mathrm{span}(q_1, \ldots, q_j) = \mathrm{span}(a_1, \ldots, a_j) \quad \text{for } j = 1, \ldots, n. \tag{6.4}$$

Thus these vectors give orthonormal bases for the linear subspaces incrementally generated by the columns of A.

It runs as follows:

First step

$$r_{1,1} \leftarrow \|a_1\|_2, \quad q_1 \leftarrow \frac{a_1}{r_{1,1}},$$

so that this latter is a unit vector in the direction of the first column of A. Next:

Second step

$$r_{1,2} \leftarrow \langle q_1, a_2 \rangle, \quad \widetilde{a}_2 \leftarrow a_2 - r_{1,2}\,q_1, \quad r_{2,2} \leftarrow \|\widetilde{a}_2\|_2, \quad q_2 \leftarrow \frac{\widetilde{a}_2}{r_{2,2}}.$$

Indeed, $a_2 - r_{1,2}\,q_1$ is orthogonal to q_1 *(prove it!)*. Hence q_2 is a unit vector that is orthogonal to q_1, and so these two vectors form an orthonormal basis of $\mathrm{span}(a_1, a_2)$ (Figure 6.2).

When $n > 2$ this process continues similarly:

Third step

$$r_{1,3} \leftarrow \langle q_1, a_3 \rangle,\ r_{2,3} \leftarrow \langle q_2, a_3 \rangle,\ \widetilde{a}_3 \leftarrow a_3 - r_{1,3}\,q_1 - r_{2,3}\,q_2,\ r_{3,3} \leftarrow \|\widetilde{a}_3\|_2,\ q_3 \leftarrow \frac{\widetilde{a}_3}{r_{3,3}}$$

and so on.

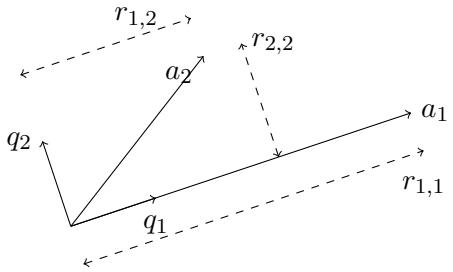

FIGURE 6.2.1. The Gram-Schmidt orthogonalization process

We can express the columns of A in terms of this orthonormal family:

$$\begin{aligned} a_1 &= r_{1,1}\, q_1, \\ a_2 &= r_{1,2}\, q_1 + r_{2,2}\, q_2, \\ a_3 &= r_{1,3}\, q_1 + r_{2,3}\, q_2 + r_{3,3}\, q_3, \ \ldots \end{aligned}$$

This can be written in matrix form as

$$\begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix} \begin{bmatrix} r_{1,1} & \cdots & \cdots & r_{1,n} \\ 0 & r_{2,2} & \cdots & r_{2,n} \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & 0 & r_{n,n} \end{bmatrix} \tag{6.5}$$

with $r_{i,j} = \langle q_i, a_j \rangle$ for $1 \le i \le j \le n$, which leads us directly to the all-important QR factorization.

Recall that a matrix $Q \in \mathbb{R}^{m\times n}$ is *orthogonal* if

$$Q^T Q = \mathbb{1}_n$$

or equivalently, if its columns form an orthonormal family of vectors of $\mathbb{R}^m$. Then the *(thin) QR factorization* of our full rank $m \times n$ matrix A is

$$A = Q\, R \tag{6.6}$$

with $Q \in \mathbb{R}^{m\times n}$ orthogonal and $R \in \mathbb{R}^{n\times n}$ upper triangular with positive diagonal entries. The *(full) QR factorization* of A is a factorization

$$A = \widetilde{Q}\, \widetilde{R} \tag{6.7}$$

with $\widetilde{Q} \in \mathbb{R}^{m\times m}$ orthogonal and $\widetilde{R} \in \mathbb{R}^{m\times n}$ of the form $\widetilde{R} = \begin{bmatrix} R \\ \mathbb{0} \end{bmatrix} \begin{matrix} n \\ m-n \end{matrix}$ where $R \in \mathbb{R}^{n\times n}$ is an upper triangular matrix having positive diagonal entries.

The thin QR factorization of A concides with that in (6.5), and so the GSO algorithm both shows its existence and gives a method to compute it. Moreover, this factorization is unique because it is equivalent to the conditions in (6.4).

The full QR factorization of A can be derived from the thin by extending Q to an $m \times m$ orthogonal matrix or equivalently, by extending the orthonormal family of vectors q_j, $j = 1, \ldots, n$, to an orthonormal basis of $\mathbb{R}^m$. Computationally, this can be done extending A to a nonsingular $m \times m$ matrix $\widetilde{A}$ by adding $m - n$ further columns that are linearly independent from those of A, and applying the GSO algorithm to $\widetilde{A}$.

EXAMPLE 6.2.1. Set

$$A = \begin{bmatrix} 1 & -3 \\ 0 & 2 \\ -1 & -1 \end{bmatrix} \in \mathbb{R}^{3\times 2}.$$

GSO applied this matrix gives that

$$q_1 = \frac{a_1}{\|a_1\|_2} = (0.71, 0, -0.71)$$

whereas $\widetilde{a}_2 = a_2 - \langle a_2, q_1\rangle\, q_1 = (-2, 2, -2)$ and so

$$q_2 = \frac{\widetilde{a}_2}{\|\widetilde{a}_2\|_2} = (-0.58, 0.58, -0.58).$$

Thus q_1 and q_1, q_2 are orthonormal bases of $\mathrm{span}(a_1)$ and of $\mathrm{span}(a_1, a_2)$, respectively. We also have that

$$\langle q_1, a_1\rangle = 1.41, \quad \langle q_1, a_2\rangle = -1.41, \quad \langle q_2, a_2\rangle = 3.49$$

and so the (thin) QR factorization of A is

$$A = Q\,R = \begin{bmatrix} 0.71 & 0.58 \\ 0 & -0.58 \\ -0.71 & 0.58 \end{bmatrix} \begin{bmatrix} 1.41 & -1.41 \\ 0 & 3.49 \end{bmatrix}.$$

Once the QR factorization is computed, we can easily solve the normal equations (6.2) and so the LS problem:

$$x_{\mathrm{LS}} = (A^T A)^{-1} A^T b = ((QR)^T Q\,R)^{-1} (Q\,R)^T b = (R^T R)^{-1} R^T Q\, b = R^{-1} Q^T\, b. \tag{6.8}$$

The GSO algorithm is not numerically stable when the columns of A are nearly linearly dependent or equivalently, when A is close to rank deficient.

6.3. Householder reflections

The key to a numerically stable algorithm for computing the QR factorization is to apply a sequence of basic elementary orthogonal transformations that approach the matrix to desired upper triangular form. These basic orthogonal transformations of $\mathbb{R}^m$ can be:

- symmetries with respect to a linear subspace,
- rotations around a linear subspace of codimension 2.

In this section we will consider *Householder reflections*, which are the symmetries of $\mathbb{R}^m$ with respect to a hyperplane. Given a unit vector $u \in \mathbb{R}^m$, its the reflection with respect to the orthogonal hyperplane $u^\perp$ is given by the $m \times m$ matrix

$$P_u = \mathbb{1}_m - 2\, u\, u^T \in \mathbb{R}^{m\times m}$$

This matrix is symmetric ($P^T = P$) and orthogonal ($P^T P = \mathbb{1}_m$) *(prove it!)*. The hyperplane $u^\perp$ acts like a mirror, since this linear map sends each vector to its opposite with respect to this hyperplane.

We claim that for any given nonzero m-vector there is a reflection that zeroes all but the first entry. That is, given $y \in \mathbb{R}^m \setminus \{0\}$ we want to find a unit vector u such

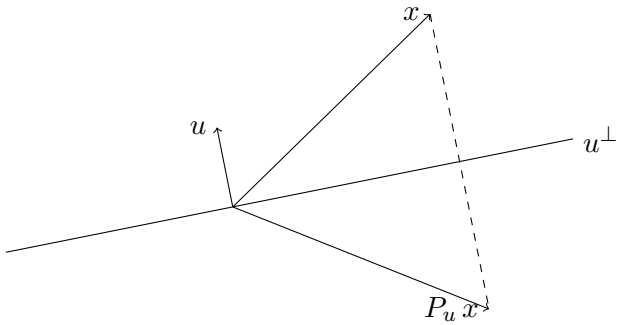

FIGURE 6.3.1. Reflection on $\mathbb{R}^2$ with respect to the line $u^\perp$

that

$$P_u\, y = \begin{bmatrix} c \\ 0 \\ \vdots \\ 0 \end{bmatrix} = c\, e_1 \tag{6.9}$$

where $e_1 = (1, 0, \ldots, 0)$ is the first vector in the standard basis of $\mathbb{R}^m$. To compute this unit vector, note that

$$P_u\, y = (\mathbb{1}_m - 2\, u\, u^T)\, y = y - 2\, \langle u, y \rangle\, u = c\, e_1. \tag{6.10}$$

Hence $2\, \langle u, y \rangle\, u = y - c\, e_1$. Since P_u is orthogonal, necessarily $|c| = \|P\, y\|_2 = \|y\|_2$ and so $c = \pm \|y\|_2$. To avoid an undesired cancellation, we choose $c = -\operatorname{sign}(y_1)\, \|y_1\|_2$, which implies that u is a scalar multiple of

$$\widetilde{u} \coloneqq y + \operatorname{sign}(y_1) \|y\|_2\, e_1 = \begin{bmatrix} y_1 + \operatorname{sign}(y_1)\, \|y\|_2 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}.$$

Thus we set

$$u = \operatorname{House}(y) = \frac{\widetilde{u}}{\|\widetilde{u}\|_2}$$

which satisfies (6.9) for $c = -\operatorname{sign}(y_1)$.

The principles for computing the QR factorization via Householder reflections can be already illustrated in the 4×3 case. Let then

$$A = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

where each symbol $\times$ indicates an unespecified scalar.

(1) Choose $P_1 \in \mathbb{R}^{4\times 4}$ such that all entries of $P_1\,[a_{1,1}\,a_{2,1}\,a_{3,1}\,a_{4,1}]^T$ are zero except the first one, and set

$$A^{(1)} \leftarrow P_1\,A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}.$$

(2) Choose $P_2' \in \mathbb{R}^{3\times 3}$ such that the second and third entries of the product $P_2'\,[a_{2,2}^{(1)}\,a_{3,2}^{(1)}\,a_{4,2}^{(1)}]^T$ are zero, set $P_2 = \begin{bmatrix} 1 & \mathbb{0} \\ \mathbb{0} & P_2' \end{bmatrix} \in \mathbb{R}^{4\times 4}$ and

$$A^{(2)} \leftarrow P_2\,A^{(1)} = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix}.$$

(3) Choose $P_3' \in \mathbb{R}^{2\times 2}$ such that the second entry of $P_3'\,[a_{3,3}^{(2)}\,a_{3,4}^{(2)}]^T$ is zero, set $P = \begin{bmatrix} \mathbb{1}_2 & \mathbb{0} \\ \mathbb{0} & P_3' \end{bmatrix} \in \mathbb{R}^{4\times 4}$ and

$$A^{(3)} \leftarrow P_3\,A^{(2)} = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix}.$$

The product $A^{(3)} \coloneqq P_3\,P_2\,P_1\,A$ is upper triangular, and we can obtain the full QR factorization

$$A = \widetilde{Q}\,\widetilde{R}$$

by setting $\widetilde{Q} = P_1^T P_2^T P_3^T \Lambda$ and $\widetilde{R} = \Lambda A^{(3)}$ with $\Lambda = \operatorname{diag}(\pm 1, \ldots, \pm 1) \in \mathbb{R}^{4\times 4}$ that is chosen so that the diagonal entries of $\widetilde{R}$ as positive, as required.

The thin QR factorization $A = Q\,R$ is obtained by suitably cutting these matrices, taking Q as the left 4×3 submatrix of $\widetilde{Q}$ and R as the upper 3×3 submatrix of $\widetilde{R}$.

Example 6.3.1. Consider the 3×2 matrix

$$A = \begin{bmatrix} 1 & -3 \\ 0 & 2 \\ -1 & -1 \end{bmatrix}.$$

Set $\widetilde{u}_1 = \begin{bmatrix} 1 + 2^{1/2} \\ 0 \\ -1 \end{bmatrix}$ so that $u_1 = \text{House}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} = \frac{\widetilde{u}_1}{\|\widetilde{u}_1\|_2} = \begin{bmatrix} 0.92 \\ 0 \\ -0.38 \end{bmatrix}$. Then

$$P_1 = \mathbb{1}_3 - 2\,u_1\,u_1^T = \begin{bmatrix} -0.71 & 0 & 0.71 \\ 0 & 1 & 0 \\ 0.71 & 0 & 0.71 \end{bmatrix} \quad \text{and} \quad A^{(1)} = P_1\,A = \begin{bmatrix} -1.41 & 1 - 41 \\ 0 & 2 \\ 0 & -2.83 \end{bmatrix}.$$

Set then $\widetilde{u}_2 = \begin{bmatrix} 2 + (2^2 + (-2.83)^2)^{1/2} \\ -2.83 \end{bmatrix}$ so that $u_2 = \text{House}\begin{bmatrix} 2 \\ -2.83 \end{bmatrix} = \frac{\widetilde{u}_2}{\|\widetilde{u}_2\|_2} = \begin{bmatrix} 0.89 \\ -0.46 \end{bmatrix}$. Then

$$P_2 = \begin{bmatrix} 1 & \mathbb{0} \\ \mathbb{0} & \mathbb{1}_2 - 2\,u_2\,u_2^T \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.58 & 0.82 \\ 0 & 0.82 & 0.58 \end{bmatrix} \text{ and } A^{(2)} = P_2\,A^{(1)} = \begin{bmatrix} -1.41 & 1.41 \\ 0 & -3.49 \\ 0 & 0 \end{bmatrix}.$$

Then setting

$$\widetilde{Q} = -P_1^T P_2^T = \begin{bmatrix} 0.71 & -0.58 & -0.41 \\ 0 & 0.58 & -0.82 \\ -0.71 & -0.58 & -0.41 \end{bmatrix} \quad \text{and} \quad \widetilde{R} = -A^{(2)} = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 3.49 \\ 0 & 0 \end{bmatrix}$$

we obtain the full QR factorization $A = \widetilde{Q}\,\widetilde{R}$. The thin QR factorization can be obtained from this by taking the appropriate submatrices, as explained before.

As in other previous algorithms, the entries of A are no longer needed after each computation, and so this matrix can be safely overwritten. Here is the algorithm for the QR factorization using Householder reflections, using overwritting and the Matlab notation.

Algorithm 6.3.1 (QR factorization *via* Householder reflections)

for $i = 1$ to $\min(m-1, n)$
  $u_i \leftarrow \text{House}(A(i:m, i))$
  $P_i' \leftarrow \mathbb{1}_{m-i+1} - 2\,u_i\,u_i^T$
  $A_i(i:m, i:n) \leftarrow P_i'\,A(i:m, i:n)$

There are some further details that can speed up its implementation. We do not need to form the $(m-i+1) \times (m-i+1)$ matrix P_i' but just need to compute the multiplication

$$(\mathbb{1}_{m-i+1} - 2\,u_i\,u_i^T)\,A(i:m, i:n) = A(i:m, i:n) - 2\,u_i\,(u_i^T A(i:m, i:n)), \tag{6.11}$$

which costs less.

Moreover, we do not need to form the product $\prod_{j=1}^{\min(m-1,n)} P_j$, and instead we can code it with the sequence of Householder vectors u_j, $j = 1, \ldots, \min(m-1, n)$. These can be stored in the lower part of the matrix plus an extra array of length $\min(m-1, n)$ for their first entries.

Recall also that to sove the LS problem $\min_x \|A\,x - b\|_2$ we need to compute the product

$$Q^T b = P_n \cdots P_1 b.$$

This can be done without explicitly forming these orthogonal matrices applying iteratively the formula in (6.11).

The complexity of this algorithm is

$$2\,n^2\,m - \frac{2}{3}\,n^2 \text{ flops}$$

if the product $Q = P_1^T \cdots P_2^T$ is not required. When $m \gg n$ this is about twice the complexity of solving the normal equations via Cholesky's algorithm, but is more numerically stable.

6.4. Givens rotations

A variant of the previous approach for computing the QR factorization can be obtained by considering rotations instead of symmetries. A rotation on the plane of angle θ is the linear map $R(\theta)\colon \mathbb{R}^2 \to \mathbb{R}^2$ given by the orthogonal matrix

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

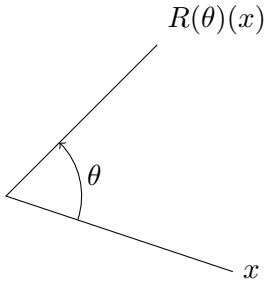

FIGURE 6.4.1. Rotation of angle θ

A *Givens rotation* on $\mathbb{R}^m$ is an linear map that rotates the vectors in one of the standard planes of $\mathbb{R}^m$ with a certain angle, and that fixes the vectors in the standard complementary subspace. Precisely, the Givens rotation associated to a pair of indexes $1 \leq i < j \leq m$ and an angle $\theta \in \mathbb{R}$ is the linear map on $\mathbb{R}^m$ corresponding to the matrix

$$R(i,j,\theta) = \begin{array}{c} \\ i \\ \\ j \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} \scriptstyle i & & & \scriptstyle j & \end{array} \\ \begin{bmatrix} \mathbb{1}_{i-1} & & & & \\ & \cos(\theta) & & -\sin(\theta) & \\ & & \mathbb{1}_{j-i-1} & & \\ & \sin(\theta) & & \cos(\theta) & \\ & & & & \mathbb{1}_{n-j-1} \end{bmatrix} \end{array}$$

Given a vector $y \in \mathbb{R}^m$ and indexes $i > j$, we can apply a Givens rotation to zero the j-th entry of y by choosing θ such that

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} y_i \\ y_j \end{bmatrix} = \begin{bmatrix} (y_i^2 + y_j^2)^{1/2} \\ 0 \end{bmatrix}.$$

Indeed, we do not need to compute the angle but just its cosinus and sinus, which are given by the formulae

$$\cos(\theta) = \frac{y_i}{(y_i^2 + y_j^2)^{1/2}} \quad \text{and} \quad \sin(\theta) = \frac{-y_j}{(y_i^2 + y_j^2)^{1/2}}.$$

Thus the QR factorization of an $m \times n$ matrix A can be computed with Givens rotations similarly as it is done with Householder reflections, but zeroing one entry of this matrix at each step.

EXAMPLE 6.4.1. Let $m = 3$, $n = 2$ and

$$A = \begin{bmatrix} 1 & -3 \\ 0 & 2 \\ -1 & -1 \end{bmatrix}.$$

Set $R_1 = \begin{bmatrix} 0.71 & 0 & -0.71 \\ 0 & 1 & 0 \\ 0.71 & 0 & 0.71 \end{bmatrix}$ so that

$$A^{(1)} = R_1\, A = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 2 \\ 0 & -2.82 \end{bmatrix}.$$

Then set $R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.58 & -0.82 \\ 0 & 0.82 & 0.58 \end{bmatrix}$ so that

$$A_2 = R_2\, A_1 = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 3.49 \\ 0 & 0 \end{bmatrix} = \widetilde{R}.$$

Setting

$$\widetilde{Q} = R_1^T R_2^T = \begin{bmatrix} 0.71 & -0.58 & 0.41 \\ 0 & 0.58 & 0.82 \\ -0.71 & -0.58 & 0.41 \end{bmatrix} \quad \text{and} \quad \widetilde{R} = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 3.49 \\ 0 & 0 \end{bmatrix}$$

we obtain the full QR factorization $A = \widetilde{Q}\,\widetilde{R}$.

The complexity of the QR factorization using Givens rotations is about twice that of doing this task with Householder reflections. However, it is useful in special situations where the input matrix has already many zeros, like in the Hessenberg case:

$$A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$

6.5. The numerical stability of the LS problem

Let A be an $m \times n$ matrix with $m \geq n$ and $\operatorname{rank}(A) = n$. In the setting of the LS problem, its *condition number* is defined as

$$\kappa_{\mathrm{LS}}(A) = \kappa_2(A^TA)^{1/2} = \big(\|A^TA\|_2\, \|(A^TA)^{-1}\|_2\big)^{1/2}. \tag{6.12}$$

Since A^TA is an $n \times n$ matrix, we can deduce from §2.3(3) that

$$\|A^TA\|_2\, \|(A^TA)^{-1}\|_2 = \left(\frac{\lambda_{\max}((A^TA)^T(A^TA))}{\lambda_{\min}((A^TA)^T(A^TA))}\right)^{1/2} = \frac{\lambda_{\max}(A^TA)}{\lambda_{\min}(A^TA)},$$

where $\lambda_{\max}(A^TA)$ and $\lambda_{\min}(A^TA)$ denote respectively the largest and the smallest eigenvalue of this SPD matrix. Hence this condition number can also be expressed as

$$\kappa_{\mathrm{LS}}(A) = \left(\frac{\lambda_{\max}(A^TA)}{\lambda_{\min}(A^TA)}\right)^{1/2}.$$

In the square case, it coincides with the condition number for linear equation solving with respect to the 2-norm: when $m = n$ we have that

$$\kappa_2(A) = \left(\frac{\lambda_{\max}(A^TA)}{\lambda_{\min}(A^TA)}\right)^{1/2} = \kappa_{\mathrm{LS}}(A).$$

This parameter controls forward error analysis of the LS problem, that is, its sensitivity to small perturbations: let $0 < \varepsilon < \kappa_2(A)^{-1}$ and suppose that

$$\frac{\|\delta A\|_2}{\|A\|_2}, \frac{\|\delta b\|_2}{\|b\|_2} \leq \varepsilon.$$

Let x and $x + \delta x$ be the solution of the LS problem for the pairs (A, b) and $(A + \delta A, b + \delta b)$, respectively. Then ([**Hig02**, Theorem 20.1])

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \frac{\kappa_{\mathrm{LS}}(A)}{1 - \varepsilon\, \kappa_{\mathrm{LS}}(A)} \Big(2 + (\kappa_{\mathrm{LS}}(A) + 1) \frac{\|Ax - b\|_2}{\|A\|_2 \|x\|_2} \Big), \tag{6.13}$$

This bound is usually interpreted as saying that the sensitivity of the LS problem in measured by $\kappa_{\mathrm{LS}}(A)$ when the residual $\|Ax - b\|_2$ is small or zero, and by $\kappa_{\mathrm{LS}}(A)^2$ otherwise.

The computation of the QR factorization $A = Q\,R$ *via* either Householder reflections or Givens rotations is backwards stable: the computed factors $Q + \delta\, Q$ and $R + \delta\, R$ are the exact factors of a perturbation $A + \delta\, A$, that is

$$A + \delta\, A = (Q + \delta\, Q)\,(R + \delta\, R),$$

whose relative error is bounded by

$$\frac{\|\delta\, A\|_2}{\|A\|_2} \leq O(n\, \varepsilon)$$

where ε denotes the machine epsilon, see [**Dem97**, §3.4.3] for details.

We can combine this with the forward error analysis in (6.13) to get error bounds for the solution of the LS problem, much as we did for linear equation solving. Hence when the residual $\|Axb\|_2$ is small and zero, the QR factorization *via* Householder reflections or Givens rotations solves the LS problem with a loss of precision of

$$\approx \log_\beta \kappa_{\mathrm{LS}}(A) \text{ digits}$$

where β denotes the base of our floating-point system, using respectively $2\, n^2\, m$ flops or $4\, n^2\, m$ flops when $m \gg n$.

On the other hand, solving the normal equations via Cholesky's algorithm solves the LS problem with a loss of precision of

$$\approx \log_b \kappa_L(A)^2 = 2 \log_b \kappa_{\mathrm{LS}}(A) \text{ digits}$$

using $\approx n^2 m$ flops when $m \gg n$.

When A is not well-conditionned, the method of choice for solving the LS problem is either by computing its QR factorization via Householder reflections or applying the SVD *(to be discussed later)*. On the other hand, if the matrix is well-conditionned then we might solve it with Cholesky's algorithm.

6.6. Rank defficient LS problem *via* the QR factorization

So far we have assumed so that A full rank when minimizing $\|Ax - b\|_2$. What happens when A is rank deficient?

Precisely, let A be an $m \times n$ matrix with $r = \mathrm{rank}(A) < n$ and $b \in \mathbb{R}^m$. The solution of the associated LS problem is not unique: for a particular solution x_0 and every $x \in \mathrm{Ker}(L_A) \simeq \mathbb{R}^{n-r}$ we have that

$$A\,(x_0 + x) = A\,x_0 + A\,x = A\,x_0,$$

and so $x_0 + x$ also solves it.

In spite of this, in practice we need to pick a single solution $x_{\mathrm{LS}} = x_0 + x$ having a reasonable size. Such a solution is typically used as a predictor for a certain variable, and predictor that is too large might give a bad result when applied to an instance outside the dataset.

EXAMPLE 6.6.1. Suppose that we are doing medical research on the effect of a drug on the level of a certain toxin in the blood as in Example 5.4.1. Then for a set of m patients we want to consider the data therein but for some reason we measure their respective weights over five consecutive days, instead of just once Then there are chances that these weights coincides, so the corresponding dataset will be encoded by a pair (A, b) where A is an $m \times 8$ matrix of rank 4.

As explained in Example 5.4.1, the solution x_{LS} would be used to predict the final toxin level

$$q = p^T x_{\mathrm{LS}}$$

for a further patient with known age, weights, initial toxin level and given amount of drug stored in a vector $p \in \mathbb{R}^8$. In might happen though that this patient does have a variation in the measurement of his weight, and an abnormally large choice of x_{LS} might given him/her an irrealistic prediction.

We can solve a rank deficient LS problem applying the QR factorization: if A has rank $r < n$ and its first r columns are linearly independent, its thin QR factorization would look like

$$A = Q\,R = \begin{bmatrix} \overset{r}{Q_1} & \overset{n-r}{Q_2} \end{bmatrix} \begin{bmatrix} \overset{r}{R_{1,1}} & \overset{n-r}{R_{1,2}} \\ 0 & 0 \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$$

with Q_1 and Q_2 orthogonal and $R_{1,1}$ nonsingular.

We can minimize the quantity $\|A\,x - b\|_2$ as follows: write $x = (x_1, x_2)$ with $x_1 \in \mathbb{R}^r$ and $x_2 \in \mathbb{R}^{n-r}$ and complete Q to an orthogonal $m \times m$ matrix $\widetilde{Q} = [Q_1\,Q_2\,Q']$ so that

$$\|A\,x - b\|_2^2 = \|Q^T(A\,x - b)\|_2^2 = \|R_{1,1}\,x_1 + R_{1,2}\,x_2 - Q_1^T b - Q_2^T b - Q'^T b\|_2^2 \tag{6.14}$$

where the second equality comes from the fact that multiplying by an orthogonal $m \times m$ matrix does not change the 2-norm.

The first three terms in the right-hand side of (6.14) lie in $\mathbb{R}^r \oplus \mathbb{0}_{n-r} \oplus \mathbb{0}_{m-n}$, whereas the fourth and the fifth lie in $\mathbb{0}_r \oplus \mathbb{R}^{n-r} \oplus \mathbb{0}_{m-n}$ and in $\mathbb{0}_r \oplus \mathbb{0}_{n-r} \oplus \mathbb{R}^{m-n}$, respectively. We deduce that

$$\|A\,x - b\|_2^2 = \|R_{1,1}\,x_1 + R_{1,2}\,x_2 - Q_1^T b\|_2^2 + \|Q_2^T b\|_2^2 + \|Q'^T b\|_2^2. \tag{6.15}$$

Hence this expression is minimized by

$$x = \begin{bmatrix} R_{1,1}^{-1}(Q_1^T b - R_{1,2}\,x_2) \\ x_2 \end{bmatrix}$$

for any $(n-r)$-vector x_2. The typical choice is $x_2 = \mathbb{0}$.