Chapter 3

Solutions of overdetermined linear problems (Least Square problems)

1. Over-constrained problems

See Chapter 11 from the textbook

1.1. Definition

In the previous chapter, we focused on solving well-defined linear problems defined by m linear equations for m unknowns, put into a compact matrix-vector form $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{A}$ an $m \times m$ square matrix, and $\mathbf{b}$ and $\mathbf{x}$ $m-$long column vectors. We focussed on using *direct methods* to seek *exact* solutions to such well-defined linear systems, which exist whenever $\mathbf{A}$ is nonsingular. We will revisit these problems later this quarter when we learn about iterative methods.

In this chapter, we look at a more general class of problems defined by so-called **overdetermined** systems – systems with a larger numbers of equations (m) than unknowns (n): this time, we have $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{A}$ an $m \times n$ matrix with $m > n$, $\mathbf{x}$ an n-long column vector and $\mathbf{b}$ an $m-$long column vector. This time there generally are no exact solutions to the problem. Rather we now want to find *approximate solutions* to overdetermined linear systems which minimize the residual error

$$E = ||\mathbf{r}|| = ||\mathbf{b} - \mathbf{Ax}|| \tag{3.1}$$

using some norm. The vector $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ is called the **residual** vector.

Any choice of norm would generally work, although, in practice, we prefer to use the Euclidean norm (i.e., the 2-norm) which is more convenient for numerical purposes, as they provide well-established relationships with the inner product and orthogonality, as well as its smoothness and convexity (we shall see later). For this reason, the numerical solution of overdetermined systems is usually called the **Least Square solution**, since it minimizes the sum of the square of the coefficients of the residual vector, $r_i = (\mathbf{b} - \mathbf{Ax})_i$ for $i = 1 \dots m$.

Remark: You will sometimes find references to least squares problems as

$$\mathbf{A}\mathbf{x} \cong \mathbf{b} \tag{3.2}$$

in order to explicitly reflect the fact that $\mathbf{x}$ is *not* the exact solution to the overdetermined system, but rather is an approximate solution that minimizes the Euclidean norm of the residual. □

1.2. Overdetermined System

The question that naturally arises is then "what makes us to consider an overdetermined system?" Let us consider some possible situations.

Example: Suppose we want to know monthly temperature distribution in Santa Cruz. We probably would not make one single measurement for each month and consider it done. Instead, we would need to take temperature readings over many years and average them. From this procedure, what we are going to make is a table of 'typical' temperatures from January to December, based on vast observational data, which often times even include unusual temperature readings that deviate from the 'usual' temperature distributions. This example illustrates a typical overdetermined system: we need to determine one representative meaningful temperature for each month (i.e., one unknown temperature for each month) based on many numbers (thus overdetermined) of collected sample data including sample noises (due to measurement failures/errors, or unusually cold or hot days – data deviations, etc.). □

Example: Early development of the method of least squares was due largely to Gauss, who used it for solving problems in astronomy, particularly determining the orbits of celestial bodies such as asteroids and comets. The least squares method was used to smooth out any observational errors and to obtain more accurate values for the orbital parameters. □

Example: A land surveyor is to determine the heights of three hills above some reference point. Sighting from the reference point, the surveyor measures their respective heights to be

$$h_1 = 1237ft., h_2 = 1942ft., \text{ and } h_3 = 2417ft. \tag{3.3}$$

And the surveyor makes another set of relative measurements of heights

$$h_2 - h_1 = 711ft., h_3 - h_1 = 1177ft., \text{ and } h_3 - h_2 = 475ft. \tag{3.4}$$

The surveyor's observation can be written as

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} \cong \begin{bmatrix} 1237 \\ 1941 \\ 2417 \\ 711 \\ 1177 \\ 475 \end{bmatrix} = \mathbf{b}. \tag{3.5}$$

It turns out that the approximate solution becomes (we will learn how to solve this soon)

$$\mathbf{x}^T = [h_1, h_2, h_3] = [1236, 1943, 2416], \tag{3.6}$$

which differ slightly from the three initial height measurements, representing a compromise that best reconciles the inconsistencies resulting from measurement errors. □

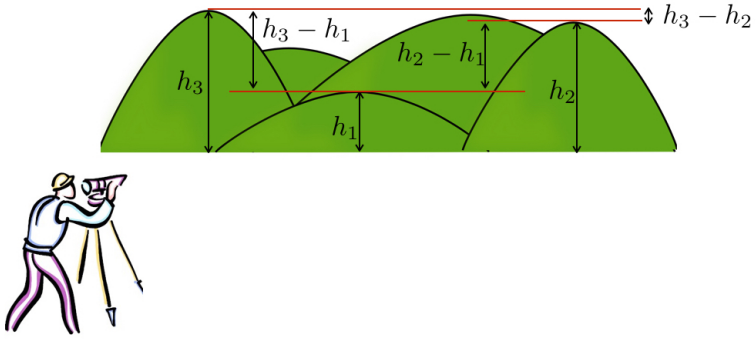

Figure 1. A math-genius land surveyor obtains three hills' height by computing the least squares solution to the overdetermined linear system.

1.3. Examples of applications of overdetermined systems

One of the most common standard applications that give rise to an overdetermined system is that of *data fitting*, or *curve fitting*. The problem can be illustrated quite simply for any function from $\mathbb{R}$ to $\mathbb{R}$ as

- Given m data points $(x_i, y_i), i = 1, \ldots, m$
- Given a function $y = f(x; \mathbf{a})$ where $\mathbf{a} = (a_1, \cdots, a_n)^T$ is a vector of n unknown parameters
- The goal is to find for which vector $\mathbf{a}$ the function f best fits the data in the least square sense, i.e. that minimizes

$$E^2 = \sum_{i=1}^{m} \Big(y_i - f(x_i, \mathbf{a}) \Big)^2. \tag{3.7}$$

The problem can be quite difficult to solve when f is a nonlinear function of the unknown parameters. However, if f uses a linear combination of the coefficients a_i this problem simply reduces to an overdetermined linear problem. These are called **linear data fitting** problems.

Definition: A data fitting problem is **linear** if f is linear in $\mathbf{a} = [a_1, \cdots, a_n]^T$, although f could be nonlinear in x.

Note that the example above can also easily be generalized to multivariate functions. Let's see a few examples of linear fitting problems.

1.3.1. Linear Regression Fitting a linear function through a set of points is a prime example of linear regression.

In the single-variable case, this requires finding the coefficients a and b of the linear function $f(x) = ax+b$ that best fits a set of data points $(x_i, y_i), i = 1, \ldots, m$. This requires solving the overdetermined system

$$\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_m & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \tag{3.8}$$

For multivariate problems, we may for instance want to fit the linear function $y = f(\mathbf{x}) = a_0 + a_1 x_1 + \cdots + a_{n-1} x_{n-1}$ through the m data points with coordinates $(x_1^{(i)}, x_2^{(i)}, \ldots, x_{n-1}^{(i)}, y^{(i)})$ for $i = 1, \ldots, m$. In that case, we need to solve the overdetermined system

$$\begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \ldots & x_{n-1}^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \ldots & x_{n-1}^{(2)} \\ \vdots & & & & \\ 1 & x_1^{(m)} & x_2^{(m)} & \ldots & x_{n-1}^{(m)} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{pmatrix} \tag{3.9}$$

for the n parameters $a_0, \ldots, a_{n-1}$.

1.3.2. Polynomial fitting Polynomial fitting, in which we try to fit a polynomial function

$$f(x; \mathbf{a}) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} \tag{3.10}$$

to a set of m points (x_i, y_i) is a also a linear data fitting problem (since f depends linearly on the coefficients of $\mathbf{a}$). Finding the best fit involves solving the overconstrained problem

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \vdots & & & & \\ 1 & x_m & x_m^2 & \ldots & x_m^{n-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \tag{3.11}$$

for the n parameters $a_0, \ldots, a_{n-1}$. The matrix formed in the process is of a particularly well-known type called a **Vandermonde matrix**.

Example: Consider five given data points, $(t_i, y_i), 1 \leq i \leq 5$, and a data fitting using a quadratic polynomial. This overdetermined system can be written as, using a Vandermonde matrix,

$$\mathbf{Ax} = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ 1 & t_4 & t_4^2 \\ 1 & t_5 & t_5^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \cong \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \mathbf{b}. \tag{3.12}$$

The problem is to find the best possible values of $\mathbf{x} = [x_1, x_2, x_3]^T$ which minimizes the residual $\mathbf{r}$ in l^2-sense:

$$||\mathbf{r}||_2^2 = ||\mathbf{b} - \mathbf{Ax}||_2^2 = \sum_{i=1}^{5} \Big(y_i - (x_1 + x_2 t_i + x_3 t_i^2) \Big)^2 \tag{3.13}$$

Such an approximating quadratic polynomial is plotted as a smooth curve in blue in Fig. 2, together with m given data points denoted as red dots. □

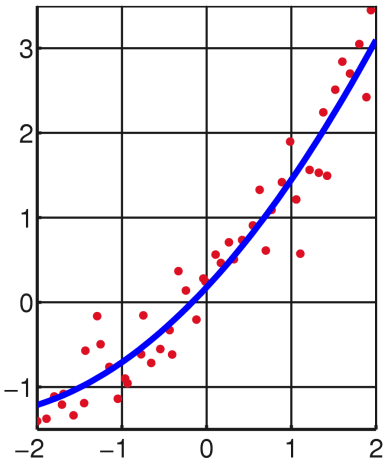

Figure 2. The result of fitting a set of data points (t_i, y_i) with a quadratic function, $f(x, \mathbf{a}) = a_0 + a_1 x + a_2 x^2$. Image source: Wikipedia

Remark: In statistics, the method of least squares is also known as *regression analysis*. □

1.3.3. A nonlinear fitting problem: Fitting a function of the kind

$$f(t, \mathbf{a}) = a_1 e^{a_2 t} + a_2 e^{a_3 t} + \cdots + a_{n-1} e^{a_n t} \tag{3.14}$$

is not a linear data fitting problem. This must be treated using **nonlinear least square** methods, beyond the scope of this course.

Note that in all the examples above, we were able to frame the problem in the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x}$ is the vector of real parameters (called $\mathbf{a}$ above) of size n, $\mathbf{A}$ is a real matrix, of size $m \times n$ and $\mathbf{b}$ is a real vector of size m.

2. Solution of Least Squares Problems using the Cholesky decomposition

As discussed earlier, a Least Squares problem can be viewed as a minimization problem on the norm of the residual $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ over all possible values of $\mathbf{x}$. This is essentially a problem in multivariate calculus! To solve it, first note that minimizing the norm or its square is the same thing. Then we write the square of the Euclidean norm of $\mathbf{r}$ as an inner product:

$$E^2 = ||\mathbf{r}||^2 = \mathbf{r}^T\mathbf{r} = (\mathbf{b} - \mathbf{Ax})^T(\mathbf{b} - \mathbf{Ax}) \tag{3.15}$$

Expanding this, we get

$$E^2 = \mathbf{b}^T\mathbf{b} - (\mathbf{Ax})^T\mathbf{b} - \mathbf{b}^T(\mathbf{Ax}) + (\mathbf{Ax})^T(\mathbf{Ax}) \tag{3.16}$$

$$= \mathbf{b}^T\mathbf{b} - \mathbf{x}^T\mathbf{A}^T\mathbf{b} - \mathbf{b}^T\mathbf{Ax} + \mathbf{x}^T\mathbf{A}^T\mathbf{Ax} \tag{3.17}$$

Minimizing the residual implies we need to find the solution $\mathbf{x}$ that satisfies

$$\frac{\partial E^2}{\partial x_i} = 0, \text{ for } i = 1, \ldots, n. \tag{3.18}$$

Writing E^2 in component form, we have

$$E^2 = \sum_i b_i^2 - \sum_{i,j} x_i a_{ji} b_j - \sum_{i,j} b_i a_{ij} x_j + \sum_{ijk} x_i a_{ji} a_{jk} x_k \tag{3.19}$$

$$= \sum_i b_i^2 - 2\sum_{i,j} x_i a_{ji} b_j + \sum_{ijk} x_i a_{ji} a_{jk} x_k \tag{3.20}$$

It is fairly easy to verify that

$$\frac{\partial E^2}{\partial x_i} = -2\sum_j a_{ji} b_j + 2\sum_{jk} a_{ji} a_{jk} x_k = 2(\mathbf{A}^T\mathbf{Ax})_i - 2(\mathbf{A}^T\mathbf{b})_i \tag{3.21}$$

so setting this to zero for all i requires

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}. \tag{3.22}$$

These new equations are usually referred to as the **normal equations** associated with the overdetermined problem $\mathbf{Ax} = \mathbf{b}$. The reason for this nomenclature will be clarified shortly.

This time, since $\mathbf{A}$ is an $m \times n$ matrix, $\mathbf{A}^T\mathbf{A}$ is a square $n \times n$ matrix which is usually much smaller than the original matrix $\mathbf{A}$. Similarly, $\mathbf{A}^T\mathbf{b}$ is now a

$n-$long vector, much smaller than the original $\mathbf{b}$. In addition, it is easy to show that as long as $\mathbf{A}$ is full rank (i.e. rank n), $\mathbf{A}^T\mathbf{A}$ is positive definite. Indeed, let $\tilde{\mathbf{A}} = \mathbf{A}^T\mathbf{A}$, and let's compute

$$\mathbf{y}^T\tilde{\mathbf{A}}\mathbf{y} = \mathbf{y}^T\mathbf{A}^T\mathbf{A}\mathbf{y} = (\mathbf{A}\mathbf{y})^T(\mathbf{A}\mathbf{y}) = ||\mathbf{A}\mathbf{y}||^2 \tag{3.23}$$

which is always greater than zero for any input vector $\mathbf{y}$ (unless $\mathbf{y} = 0$) since $\mathbf{A}$ is assumed to be full rank.

Positive definiteness is good news because it means that we can use the Cholesky decomposition to solve the problem. Since we saw that the Cholesky decomposition is stable, we therefore know that we can stably find solutions to Least Square problems using the following steps. Given the overconstrained problem $\mathbf{A}\mathbf{x} = \mathbf{b}$,

- Form the normal equation $\tilde{\mathbf{A}}\mathbf{x} = \tilde{\mathbf{b}}$ where $\tilde{\mathbf{A}} = \mathbf{A}^T\mathbf{A}$ and $\tilde{\mathbf{b}} = \mathbf{A}^T\mathbf{b}$.
- Solve the normal equation using the Cholesky decompositon and backsubstitution, for the unknown vector $\mathbf{x}$.

and voila! Examples of application of this method for data fitting will be explored in Homework 3.

Among all the methods for the solution on unconstrained linear systems (see later for other ones) this is definitely the fastest. However, it is also the least accurate of all methods (even though it is stable). That's because

$$\text{cond}(\mathbf{A}^T\mathbf{A}) = \text{cond}(\mathbf{A})^2, \tag{3.24}$$

so if $\mathbf{A}$ is relatively poorly conditioned (large cond($\mathbf{A}$)) then $\tilde{\mathbf{A}} = \mathbf{A}^T\mathbf{A}$ is *very* poorly conditioned, and from a numerical point of view can even be singular even if the original problem is not.

Example: Consider

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}, \tag{3.25}$$

where $0 < \epsilon < \sqrt{\epsilon_{\text{mach}}}$. Forming $\tilde{\mathbf{A}}$ we get

$$\tilde{\mathbf{A}} = \mathbf{A}^T\mathbf{A} = \begin{bmatrix} 1+\epsilon^2 & 1 \\ 1 & 1+\epsilon^2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \tag{3.26}$$

in floating point arightmetic, which is a singular matrix. □

Even though this example is rather extreme, poor conditioning is unfortunately very common in data sets that involve points with vastly varying values of x, *especially* for polynomial fitting and their associated Vandermonde matrices. For this reason, other methods were later developed that do not involve forming the normal equations in the first place, but instead, working directly with the original matrix $\mathbf{A}$.

3. Towards better algorithms for Least Square problems

3.1. A geometric interpretation of the Least Square Problem

To see how one may go about constructing such methods, let's look back at the normal equations and re-write them as

$$\mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{x}) = 0 \Leftrightarrow \mathbf{A}^T\mathbf{r} = 0 \tag{3.27}$$

which is also equivalent to saying that

$$\mathbf{a}_i^T\mathbf{r} = 0 \text{ for all } i = 1, \ldots, n, \tag{3.28}$$

where $\mathbf{a}_i$ are column vectors of $\mathbf{A}$ (or $\mathbf{a}_i^T$ are row vectors of $\mathbf{A}^T$). In other words, the normal equations simply state that the solution that minimizes the error E is the one for which the residual $\mathbf{r}$ is orthogonal to the column vectors of $\mathbf{A}$.

While this may seem to be just an odd coincidence at first, of course it isn't. This result actually has a very simple geometric interpretation. Consider indeed that, in solving the original problem $\mathbf{A}\mathbf{x} \cong \mathbf{b}$, the vector $\mathbf{b}$ has dimension m, while the range of the matrix $\mathbf{A}$ only has dimension $n < m$. The vector $\mathbf{y} = \mathbf{A}\mathbf{x}$, by definition, is a linear combination of the column vectors of $\mathbf{A}$, and therefore lies in the span of $\mathbf{A}$. But in general $\mathbf{b}$ does not lie in span($\mathbf{A}$) because its dimension is m and is larger than the dimension of span($\mathbf{A}$). We therefore have the situation illustrated in the figure below. In this figure, we see that the vector

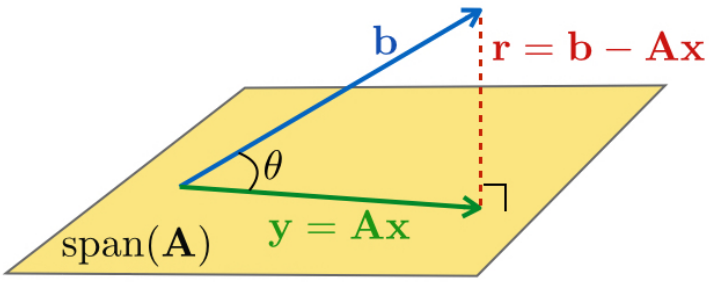

Figure 3. Geometric interpretation of linear squares problem.

$\mathbf{r}$ joins the tip of $\mathbf{b}$ to the tip of $\mathbf{y}$, and so it is shortest when $\mathbf{r}$ is perpendicular to the plane spanned by $\mathbf{A}$. This picture can be expanded to more than 3 dimensions, and shows that to find the vector $\mathbf{x}$ for which $||\mathbf{r}|| = ||\mathbf{b} - \mathbf{A}\mathbf{x}||$ is minimal, we simply have to ensure that $\mathbf{r}$ is orthogonal to the span of $\mathbf{A}$, which requires that it must be orthogonal to every single column vector of $\mathbf{A}$.

More importantly, this diagram also illustrates that finding the vector $\mathbf{A}\mathbf{x}$ for which $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$ is orthogonal to the span of $\mathbf{A}$ is equivalent to finding the

orthogonal projection of **b** *onto* the span of **A**. If that projection operator, $\mathbf{P}_A$, is known, then we just have to solve

$$\mathbf{A}\mathbf{x} = \mathbf{P}_A\mathbf{b} \tag{3.29}$$

to find **x**. Note that this time, $\mathbf{A}\mathbf{x}$ and $\mathbf{P}_A\mathbf{b}$ are both vectors of length equal to the rank of **A** (n, if **A** is full rank) so this equation forms an exact linear system. This idea leads to a new method of solution of Least Square problems, namely that of orthogonal projection. To construct it we first need to make a little detour to learn about orthogonal projectors.

3.2. Orthogonal Projectors

See Chapter 6 of the textbook

Definition: A square matrix **P** (of size $m \times m$) is said to be a **projector** if it is **idempotent**, that is,

$$\mathbf{P}^2 = \mathbf{P}. \tag{3.30}$$

Definition: If a projector **P** is also symmetric, $\mathbf{P}^T = \mathbf{P}$, then it is called an **orthogonal projector**. Note that an orthogonal projector is not necessarily an orthogonal matrix (this can be confusing!)

Orthogonal projectors behave exactly as you would imagine from their names: they project vectors onto some subspace, orthogonally to that subspace. And the subspace in question is simply the space spanned by the column vectors of **P**. To see all of this, first note that because $\mathbf{P}^2 = \mathbf{P}$, then

$$\mathbf{P}\mathbf{p}_i = \mathbf{p}_i \text{ for } i = 1, \ldots, k \tag{3.31}$$

where $\mathbf{p}_i$ are the column vectors of **P**, and $\text{rank}(\mathbf{P}) = k \leq m$. Hence, any vector **x** that lies in the subspace spanned by the columns of **P**, which can be written as $\mathbf{x} = \sum_i \alpha_i \mathbf{p}_i$, is invariant by application of **P** since $\mathbf{P}\mathbf{x} = \sum_i \alpha_i \mathbf{P}\mathbf{p}_i = \sum_i \alpha_i \mathbf{p}_i = \mathbf{x}$. Meanwhile, suppose instead that we have a vector **x** that is *orthogonal* to the span of **P**, then by definition we have $\mathbf{x}^T\mathbf{p}_i = \mathbf{p}_i^T\mathbf{x} = 0$ for all i. If we now calculate $\mathbf{P}\mathbf{x}$ we find that

$$(\mathbf{P}\mathbf{x})_i = (\mathbf{P}^T\mathbf{x})_i = \mathbf{p}_i^T\mathbf{x} = 0 \text{ for all } i = 1, \ldots, k \tag{3.32}$$

as required. Hence, the projector **P** behaves as expected, leaving any vector in the span of **P** invariant, but projecting any vector orthogonal to the span of **P** onto 0.

Example:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.33}$$

is an orthogonal projector that maps all vectors in $\mathbb{R}^3$ onto the x-y plane, while keeping those vectors in the x-y plane unchanged:

$$\mathbf{P}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}, \tag{3.34}$$

and

$$\mathbf{P}\begin{bmatrix} x \\ y \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}, \text{ while } \mathbf{P}\begin{bmatrix} 0 \\ 0 \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.35}$$

It is easy to check $\mathbf{P}^2 = \mathbf{P}$:

$$\mathbf{P}^2\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{P}\begin{bmatrix} x \\ y \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}. \tag{3.36}$$

□

Note: Given an orthogonal projector $\mathbf{P}$ we can also define

$$\mathbf{P}_\perp = \mathbf{I} - \mathbf{P} \tag{3.37}$$

which can be shown to be an orthogonal projector onto $\text{span}(\mathbf{P})^\perp$, the orthogonal complement of $\text{span}(\mathbf{P})$. Then, we can express any vector $\mathbf{x} \in \mathbb{R}^m$ as a sum

$$\mathbf{x} = \Big(\mathbf{P} + (\mathbf{I} - \mathbf{P})\Big)\mathbf{x} = \mathbf{P}\mathbf{x} + \mathbf{P}_\perp\mathbf{x}. \tag{3.38}$$

□

We will now use these definitions to go back to the problem discussed at the end of the previous section, namely how to construct $\mathbf{P}_A$, an orthogonal projector onto the span of any given $m \times n$ matrix $\mathbf{A}$.

Let us now consider how we can make use of the concept of the orthogonal projector $\mathbf{P}_A$ onto $\text{span}(\mathbf{A})$ to help understanding in solving the overdetermined system $\mathbf{A}\mathbf{x} \cong \mathbf{b}$. For notational simplicity, let us denote $\mathbf{P} = \mathbf{P}_A$ for now. First, by definition, we get

$$\mathbf{P}\mathbf{A} = \mathbf{A}, \mathbf{P}_\perp\mathbf{A} = \mathbf{0}. \tag{3.39}$$

Then we have

$$\begin{aligned} ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 &= ||\mathbf{P}(\mathbf{b} - \mathbf{A}\mathbf{x}) + \mathbf{P}_\perp(\mathbf{b} - \mathbf{A}\mathbf{x})||_2^2 \\ &= ||\mathbf{P}(\mathbf{b} - \mathbf{A}\mathbf{x})||_2^2 + ||\mathbf{P}_\perp(\mathbf{b} - \mathbf{A}\mathbf{x})||_2^2 \text{ (by Pythagorean Theorem)} \\ &= ||\mathbf{P}\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + ||\mathbf{P}_\perp\mathbf{b}||_2^2. \end{aligned} \tag{3.40}$$

Therefore, we see that the least squares solution is given by the solution $\mathbf{x}$ that satisfies the first term in the last relation, which is the solution to the overdetermined linear system,

$$\mathbf{Ax} = \mathbf{Pb}. \tag{3.41}$$

This is an intuitively clear result and is shown in Fig. 3 that the orthogonal projection $\mathbf{Pb}$ of $\mathbf{b}$ gives $\mathbf{y} \in \text{span}(\mathbf{A})$, the closest vector to $\mathbf{b}$.

Remember that we wish to transform our given overdetermined $m \times n$ system into an $n \times n$ square system, so that we can use the techniques we learned in Chapter 2. Assuming $\text{rank}(\mathbf{A}) = n$, there are two ways to construct orthogonal projectors (i.e., symmetric and idempotent) *explicitly*, which allow us to have transformation into the square system:

- $\mathbf{P} = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$.
- $\mathbf{P} = \mathbf{QQ}^T$, where $\mathbf{Q}$ is an $m \times n$ matrix whose columns form an orthonormal bases (such $\mathbf{Q}$ is said to be *orthogonal* and satisfies $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$) for $\text{span}(\mathbf{A})$. Obviously, this gives $\text{span}(\mathbf{Q}) = \text{span}(\mathbf{A})$.

First of all, we see that both choices of $\mathbf{P}$ easily satisfy $\mathbf{P}^T = \mathbf{P}$ and $\mathbf{P}^2 = \mathbf{P}$. Also, after substituting $\mathbf{P}$ into Eq. 3.41, one can respectively obtain the following square systems:

- $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$ (note here we used $\mathbf{A}^T\mathbf{P} = \mathbf{A}^T\mathbf{P}^T = (\mathbf{PA})^T = \mathbf{A}^T$)
- $\mathbf{Q}^T\mathbf{Ax} = \mathbf{Q}^T\mathbf{b}$ (note here we used $\mathbf{Q}^T\mathbf{P} = \mathbf{Q}^T\mathbf{QQ}^T = \mathbf{Q}^T$)

Notice that the first transformation – the easier construction of the two – results in the system of normal equations which we already showed there are some associated numerical issues. The second orthogonal transformation, however, will provide us with a very useful idea on accomplishing so called the *QR factorization* as will be shown in Section 5.

4. Invariant Transformations

We will now focus on examining several methods for transforming an overdetermined $m \times n$ linear least squares problem $\mathbf{Ax} \cong \mathbf{b}$ into an $n \times n$ square linear system $\mathbf{A}'\mathbf{x} = \mathbf{b}'$ which leaves $\mathbf{x}$ unchanged and which we already know how to solve using the methods of Chapter 2.

In seeking for invariant transformations we keep in mind that the sequence of problem transformation we would like to establish is:

$$\text{rectangular} \rightarrow \text{square} \rightarrow \text{triangular}$$

Note: The second transformation (square to triangular) is what we already have learned in Chapter 2; while we now try to learn the first transformation (rectangular to square) in this chapter. □

4.1. Normal Equations

With rank($\mathbf{A}$) = n we already have seen several times the $n \times n$ symmetric positive definite system of normal equations

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b} \tag{3.42}$$

has the invariant property of preserving the same solution $\mathbf{x}$ as the $m \times n$ least squares problem $\mathbf{A}\mathbf{x} \cong \mathbf{b}$.

As discussed, we could theoretically pursuit to use the Cholesky factorization,

$$\mathbf{A}^T\mathbf{A} = \mathbf{L}\mathbf{L}^T, \tag{3.43}$$

followed by solving the forward-substitution first $\mathbf{L}\mathbf{y} = \mathbf{A}^T\mathbf{b}$, and then the backward-substitution later $\mathbf{L}^T\mathbf{x} = \mathbf{y}$. But we've seen there are numerical accuracy and stability issues that are related to floating-point arithmetics as well as condition number squaring effects. In this reason, we do not use the normal equations in practice.

4.2. Orthogonal Transformations

In view of the potential numerical difficulties with the normal equations approach, we need an alternative that does not require $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{b}$. In this alternative, we expect a more numerically robust type of transformation.

Recall that in Chapter 2 we did use a similar trick to introduce transformations to a simpler system which was a triangular system. Can we use the same triangular form for the current purpose? The answer is no simply because such a triangular transformation does not preserve what we want to preserve in the least squares problems now, the l^2-norm.

What kind other transformation then preserve the norm, at the same time, without changing the solution $\mathbf{x}$? Hinted by the previous section, we see that an orthogonal transformation given by $\mathbf{Q}$, where $\mathbf{Q}$ is a orthogonal real square matrix, i.e., $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$ (in other words, each column of $\mathbf{Q}$ is orthonormal basis) would be a good candidate.

The norm-preserving property of $\mathbf{Q}$ can be easily shown:

$$||\mathbf{Q}\mathbf{x}||_2^2 = (\mathbf{Q}\mathbf{x})^T\mathbf{Q}\mathbf{x} = \mathbf{x}^T\mathbf{Q}^T\mathbf{Q}\mathbf{x} = \mathbf{x}^T\mathbf{x} = ||\mathbf{x}||_2^2. \tag{3.44}$$

Similarly, we also have

$$||\mathbf{Q}^T\mathbf{x}||_2^2 = (\mathbf{Q}^T\mathbf{x})^T\mathbf{Q}\mathbf{x} = \mathbf{x}^T\mathbf{Q}\mathbf{Q}^T\mathbf{x} = \mathbf{x}^T\mathbf{x} = ||\mathbf{x}||_2^2. \tag{3.45}$$

Remark: Orthogonal matrices are of great importance in many areas of numerical computation because of their norm-preserving property. With this property, the magnitude of errors will remain the same without any amplification. Thus, for example, one can use orthogonal transformations to solve square linear systems which will *not* require the need for pivoting for numerical stability. Although it looks very attractive the orthogonalization process is significantly more expensive computationally then the standard Gaussian elimination, so their superior numerical properties come at a price. □

4.3. Triangular Least Squares

As we now prepared ourselves with a couple of transformations that preserves the least squares solution, we are further motivated to seek for a more simplified system where a least squares problem can be solved in an easier way. As seen in the square linear systems in Chapter 2, we consider least squares problems with an upper triangular matrix of the form:

$$\begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{x} \cong \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = \mathbf{c}, \tag{3.46}$$

where $\mathbf{R}$ is an $n \times n$ upper triangular matrix, $\mathbf{x}$ an n-vector. The right hand side vector $\mathbf{c}$ is partitioned accordingly into an n-vector $\mathbf{c}_1$ and an $(m-n)$-vector $\mathbf{c}_2$.

We see that the least squares residual is given by

$$||\mathbf{r}||_2^2 = ||\mathbf{c}_1 - \mathbf{R}\mathbf{x}||_2^2 + ||\mathbf{c}_2||_2^2, \tag{3.47}$$

which tells us that the the least squares solution $\mathbf{x}$ satisfies

$$\mathbf{R}\mathbf{x} = \mathbf{c}_1, \tag{3.48}$$

which is solvable by back-substitution. The minimum residual then becomes

$$||\mathbf{r}||_2^2 = ||\mathbf{c}_2||_2^2. \tag{3.49}$$

4.4. QR Factorization

Let us now combine the two nice techniques, the orthogonal transformation and the triangular least squares, into one, and we call this method the *QR factorization*. The QR factorization method writes $m \times n$ $(m > n)$ matrix $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \tag{3.50}$$

where $\mathbf{Q}$ is an $m \times m$ orthogonal matrix and $\mathbf{R}$ is an $n \times n$ upper triangular matrix.

This QR factorization transforms the *linear squares problem* $\mathbf{A}\mathbf{x} \cong \mathbf{b}$ into a *triangular least squares problem*. Do they both have the same solution? To see this, we check:

$$\begin{aligned} ||\mathbf{r}||_2^2 &= ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 && (3.51)\\ &= ||\mathbf{b} - \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{x}||_2^2 && (3.52)\\ &= ||\mathbf{Q}^T(\mathbf{b} - \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{x})||_2^2 && (3.53)\\ &= ||\mathbf{Q}^T\mathbf{b} - \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{x}||_2^2 && (3.54)\\ &= ||\mathbf{c}_1 - \mathbf{R}\mathbf{x}||_2^2 + ||\mathbf{c}_2||_2^2, && (3.55) \end{aligned}$$

where the transformed right hand side

$$\mathbf{Q}^T\mathbf{b} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix}, \tag{3.56}$$

with n-vector $\mathbf{c}_1$ and an $(m-n)$-vector $\mathbf{c}_2$. As in the previous section, we make the same conclusion on $\mathbf{x}$ that satisfies:

$$\mathbf{R}\mathbf{x} = \mathbf{c}_1, \tag{3.57}$$

which is solvable by back-substitution, and the minimum residual is

$$||\mathbf{r}||_2^2 = ||\mathbf{c}_2||_2^2. \tag{3.58}$$

We will study how to compute this QR factorization in the next section.

5. The QR factorization for Least Square problems

5.1. Preliminaries

See Chapter 7 of the textbook

Definition: A reduced QR factorization of a full rank $m \times n$ $(m > n)$ matrix $\mathbf{A}$ expresses $\mathbf{A}$ as

$$\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}} \tag{3.59}$$

where $\hat{\mathbf{Q}}$ is an $m \times n$ matrix whose column vectors are orthonormal, and $\hat{\mathbf{R}}$ is an $n \times n$ upper triangular matrix.

Definition: A full QR factorization of a full rank $m \times n$ $(m > n)$ matrix $\mathbf{A}$ expresses $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{Q}\mathbf{R} \tag{3.60}$$

where $\mathbf{Q}$ is an $m \times m$ orthogonal matrix (unitary, if $\mathbf{A}$ is complex) and $\mathbf{R}$ is an $m \times n$ upper triangular matrix (i.e. a matrix whose entries r_{ij} are 0 if $i > j$).

The two definitions are related in the sense that $\hat{\mathbf{Q}}$ forms the first n column-vectors of $\mathbf{Q}$, and $\hat{\mathbf{R}}$ forms the first n rows of $\mathbf{R}$. The last $m-n$ columns of $\mathbf{Q}$ are filled with vectors that are orthogonal to each other and to the span of $\hat{\mathbf{Q}}$, while the last $m-n$ rows of $\mathbf{R}$ are just filled with zeros. We therefore have

$$\mathbf{Q} = \left(\hat{\mathbf{Q}}\middle|\mathbf{q}_{n+1} \quad \dots \quad \mathbf{q}_m\right) \text{ and } \mathbf{R} = \begin{pmatrix} \hat{\mathbf{R}} \\ \hline \mathbf{0} \end{pmatrix} \tag{3.61}$$

As it turns out, knowing $\hat{\mathbf{Q}}$ is all we need to construct the projector $\mathbf{P}_A$. That's because it can be shown that $\mathbf{P} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T$ is indeed a projector on the subspace spanned by $\mathbf{A}$, hence $\mathbf{P}_A = \mathbf{P}$. To do this we must show that

- The span of $\mathbf{P}$ is the same as the span of $\mathbf{A}$

- $\mathbf{P}^2 = \mathbf{P}$
- $\mathbf{P}^T = \mathbf{P}$.

Proving the second and third statements is very easy:

$$\mathbf{P}^T = (\hat{\mathbf{Q}}\hat{\mathbf{Q}}^T)^T = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T \tag{3.62}$$

$$\mathbf{P}^2 = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T\hat{\mathbf{Q}}\hat{\mathbf{Q}}^T = \hat{\mathbf{Q}}\mathbf{I}\hat{\mathbf{Q}}^T = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T = \mathbf{P} \tag{3.63}$$

because $\hat{\mathbf{Q}}^T\hat{\mathbf{Q}}$ is an $n \times n$ identity matrix by the orthonormality of the columns of $\hat{\mathbf{Q}}$.

To show that the span of $\mathbf{P}$ is the same as a span of $\mathbf{A}$ is a little trickier, but not much. We can first show that the span of $\hat{\mathbf{Q}}$ lies in the span of $\mathbf{A}$ by noting that $\hat{\mathbf{Q}} = \mathbf{A}\hat{\mathbf{R}}^{-1}$, so the column vectors of $\hat{\mathbf{Q}}$, $\mathbf{q}_i = \mathbf{A}(\hat{\mathbf{R}}^{-1})_i$ (where $(\hat{\mathbf{R}}^{-1})_i$ are the column vectors of $\hat{\mathbf{R}}^{-1}$), are necessarily a linear combination of the column vectors of $\mathbf{A}$ (see first lecture). Then, since the $\mathbf{q}_i$ vectors are mutually orthogonal, they are also linearly independent. Finally, since there are n of them, the rank of $\hat{\mathbf{Q}}$ is n, which is the same as the rank of $\mathbf{A}$, so the span of $\hat{\mathbf{Q}}$ is *equal* to the span of $\mathbf{A}$. A series of very similar arguments can then be applied to show that the span of $\mathbf{P}$ is equal to the span of $\hat{\mathbf{Q}}$, and therefore to the span of $\mathbf{A}$.

This shows that $\mathbf{P} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T$ is the orthogonal projector onto the span of $\mathbf{A}$, namely $\mathbf{P}_A$, that we were looking for earlier. We then write

$$\mathbf{A}\mathbf{x} = \mathbf{P}_A\mathbf{b} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T\mathbf{b} \rightarrow \hat{\mathbf{Q}}^T\mathbf{A}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b} \rightarrow \hat{\mathbf{Q}}^T\hat{\mathbf{Q}}\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b} \rightarrow \hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b} \tag{3.64}$$

where we repeatedly used the fact that $\hat{\mathbf{Q}}^T\hat{\mathbf{Q}} = \mathbf{I}$. In other words, an alternative algorithm for solving the overdetermined system $\mathbf{A}\mathbf{x} = \mathbf{b}$ involves

- Finding the reduced QR factorization of $\mathbf{A}$, such that $\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$
- Solving the exact system $\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$

Note that $\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$ is an exact system since $\hat{\mathbf{R}}$ is $n \times n$ and $\mathbf{x}$ and $\hat{\mathbf{Q}}^T\mathbf{b}$ are both $n-$long vectors. Also, since $\hat{\mathbf{R}}$ is upper triangular, the second step boils down to a very basic back-substitution step! The crux of the method is therefore not step 2, but step 1, the QR factorization.

To understand how to construct a QR factorization, it is worth trying to interpret its meaning geometrically. A simple way of seing what the reduced QR factorization does is to note that it constructs an orthonormal basis for the space spanned by the column vectors of $\mathbf{A}$, namely the basis formed by the column vectors of $\hat{\mathbf{Q}}$. Furthermore, since

$$\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}} \rightarrow \mathbf{a}_j = \hat{\mathbf{Q}}\mathbf{r}_j \tag{3.65}$$

This explicitly writes each vector $\mathbf{a}_j$ in this new orthonormal basis, and the components of the vector $\mathbf{r}_j$ are the coordinates of $\mathbf{a}_j$ in the new basis.

There are a number of commonly used methods to construct the reduced QR factorization of the matrix $\mathbf{A}$. The two we will see here are

- Gram-Schmidt orthogonalization,
- Householder transformation (based on elementary reflectors).

There is another popular method that uses so-called Givens transformations, which is based on plane rotations. This method can be found in many numerical linear algebra text books but we will not cover it here. Finally, in all that follows we now assume that $\mathbf{A}$ is real, although the various methods are very easy to generalize for complex matrices using unitary transformations instead of orthogonal ones.

5.2. QR decomposition using Gram-Schmidt Orthogonalization

See Chapters 8 and 11

The most straightforward method for computing the QR factorization is *Gram-Schmidt orthogonalization*. It works directly with the expression $\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$, expressing it in vector form, and progressively solving for the columns of $\hat{\mathbf{Q}}$ and the elements of $\hat{\mathbf{R}}$. To see how, note first that $\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$ is equivalent to the system of vector equations

$$\begin{aligned} \mathbf{a}_1 &= r_{11}\mathbf{q}_1 \\ \mathbf{a}_2 &= r_{12}\mathbf{q}_1 + r_{22}\mathbf{q}_2 \\ &\vdots \\ \mathbf{a}_i &= \sum_{k=1}^{i} r_{ki}\mathbf{q}_k \end{aligned} \tag{3.66}$$

This shows that

$$\mathbf{q}_1 = \frac{\mathbf{a}_1}{||\mathbf{a}_1||} \text{ and } r_{11} = ||\mathbf{a}_1|| \tag{3.67}$$

Next, we have

$$\mathbf{q}_2 = \frac{\mathbf{a}_2 - r_{12}\mathbf{q}_1}{r_{22}} \tag{3.68}$$

Once $\mathbf{a}_2 - r_{12}\mathbf{q}_1$ is known we can calculate $r_{22} = ||\mathbf{a}_2 - r_{12}\mathbf{q}_1||$ to normalize $\mathbf{q}_2$. The value of r_{12} if found by requiring the orthogonality of $\mathbf{q}_2$ with $\mathbf{q}_1$: we need $\mathbf{q}_1^T\mathbf{q}_2 = 0$ so $r_{12} = \mathbf{q}_1^T\mathbf{a}_2$ since $\mathbf{q}_1$ is normalized. This shows that

$$\mathbf{q}_2 = \frac{\mathbf{a}_2 - (\mathbf{q}_1^T\mathbf{a}_2)\mathbf{q}_1}{r_{22}} = \frac{\mathbf{a}_2 - \mathbf{q}_1\mathbf{q}_1^T\mathbf{a}_2}{r_{22}} = \frac{(\mathbf{I} - \mathbf{q}_1\mathbf{q}_1^T)\mathbf{a}_2}{r_{22}} \tag{3.69}$$

Note that in the last expression above, it is easy to show, based on what we learned about projectors earlier, that $\mathbf{q}_1\mathbf{q}_1^T$ is an orthogonal projector onto the vector $\mathbf{q}_1$, that we shall call $\mathbf{P}_1$. This will come in handy later when interpreting

the process geometrically.

At the i-th step, we then have

$$\mathbf{q}_i = \frac{\mathbf{a}_i - \sum_{k=1}^{i-1} r_{ki}\mathbf{q}_k}{r_{ii}} \tag{3.70}$$

and requiring as before orthogonality of $\mathbf{q}_i$ with all previously determined vectors we get $r_{ki} = \mathbf{q}_k^T \mathbf{a}_i$. This can be used to form $\mathbf{a}_i - \sum_{k=1}^{i-1} r_{ki}\mathbf{q}_k$, and then $\mathbf{q}_i$ after normalization. And as before, we can also interpret

$$\mathbf{a}_i - \sum_{k=1}^{i-1} r_{ki}\mathbf{q}_k = (\mathbf{I} - \sum_{k=1}^{i-1} \mathbf{P}_k)\mathbf{a}_i \tag{3.71}$$

where $\mathbf{P}_k = \mathbf{q}_k \mathbf{q}_k^T$ is a projector onto $\mathbf{q}_k$.

We can write the so-called basic Gram-Schmidt orthogonalization algorithm as follows:

Algorithm: Basic Gram-Schmidt orthogonalization:

```
do i = 1 to n
    [!loop over columns of A and Q̂]
    q_i = a_i
    do k = 1 to i − 1
        r_ki = q_k^T a_i
        [!Form the coefficients of R̂]
        q_i = q_i − r_ki q_k  [!Compute q_i]
    enddo
    r_ii = ||q_i||
    if r_ii = 0 stop
    q_i = q_i / r_ii  [!Normalize q_i]
enddo
```

In this algorithm we treated $\mathbf{a}_i$ and $\mathbf{q}_i$ separately for clear exposition purpose, but they can be shared in the same storage, with new $\mathbf{q}_i$ gradually replacing the $\mathbf{a}_i$ $\square$

While we derived the algorithm from mathematical considerations, it has a very simple geometrical interpretation, illustrated in Figure 4.

Indeed, the first $\mathbf{q}_1$ is selected to be the unit vector in the direction of $\mathbf{a}_1$, and successive vectors $\mathbf{q}_i$ are constructed using the orthogonal projectors $\mathbf{P}_k$ to be orthogonal to the previous ones. For instance, we see that $(\mathbf{I} - \mathbf{P}_1)\mathbf{a}_2$ constructs a vector that is perpendicular to $\mathbf{q}_1$ while lying in the plane spanned by $\mathbf{a}_1$ and $\mathbf{a}_2$, and by successive application of the algorithm, $(\mathbf{I} - \mathbf{P}_1 - \mathbf{P}_2 - \cdots - \mathbf{P}_{i-1})\mathbf{a}_i$ constructs a vector that is perpendicular to the span of $\{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{i-1}\}$ while lying in the span of $\{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{i-1}, \mathbf{q}_i\}$

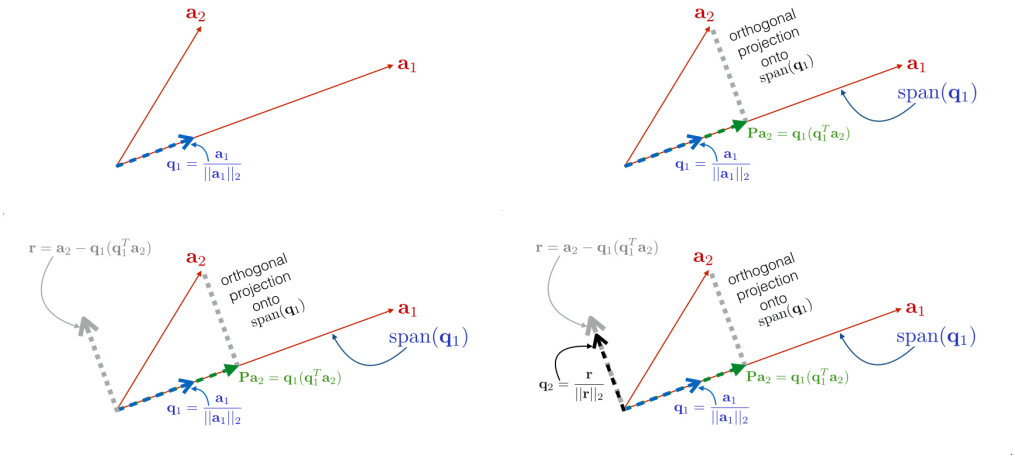

Figure 4. Geometrical interpretation of Gram-Schmidt orthogonalization

As it turns out, it can be shown that this Gram-Schmidt algorithm is unfortunately numerically unstable (see below for more on this). However there is a remarkably simple fix to that problem – simply switch the order of the operations, leading to the **Modified Gram-Schmidt algorithm**:

Algorithm: Modified Gram-Schmidt orthogonalization:

```
do i = 1 to n
    [!First initialize temporary vectors v_i as the a_i]
    v_i = a_i
enddo
do i = 1 to n
    r_ii = ||v_i||
    q_i = v_i / r_ii  [!Create q_i normalizing v_i ]
    do k = i + 1 to n
    [!Apply simple projector to all the vectors v_k for k > i]

        r_ik = q_i^T v_k
        v_k = v_k − r_ik q_i
    enddo
enddo
```

As before, we can alternatively store the $\mathbf{q}$ vectors into the columns of $\mathbf{A}$ as the algorithm proceeds. With a bit of work, one can show that the operations performed end up being *exactly* the same as in the basic Gram-Schmidt algo-

rithm, but their order is different, and the new ordering stabilizes the algorithm. The modified algorithm has therefore become the standard in commercial packages.

It is also interesting to note that this modified algorithm can be viewed as the successive multiplication of the matrix $\mathbf{A}$ from the right by upper triangular matrices $\mathbf{R}_i$, as

$$\mathbf{A}\mathbf{R}_1\mathbf{R}_2\dots\mathbf{R}_n = \hat{\mathbf{Q}} \text{ where } \mathbf{R}_i = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \frac{1}{r_{ii}} & -\frac{r_{i,i+1}}{r_{ii}} & \cdots & -\frac{r_{i,n}}{r_{ii}} & \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \end{pmatrix} \tag{3.72}$$

where the coefficients r_{ij} were defined earlier. Since the multiplication of upper triangular matrices is another upper triangular one, and since the inverse of an upper triangular matrix is also upper triangular, we then have

$$\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}} \text{ where } \hat{\mathbf{R}} = (\mathbf{R}_1\mathbf{R}_2\dots\mathbf{R}_n)^{-1} \tag{3.73}$$

For this reason, the procedure is sometimes referred to as **triangular orthogonalization** (i.e. multiply by triangular matrices to form an orthogonal one).

Note that in practice, however, it is rather difficult to find examples of commonly occurring matrices for which an instability genuinely manifests itself in the basic algorithm. Recall that for instability to occur an algorithm must return an answer whose error does *not* scale with machine accuracy. See the textbook Chapter 9 for such an example. What is much more common, however, is to note that *both* Gram-Schmidt-based QR decomposition algorithms (standard and modified) suffer from the accumulation of round-off errors, and this manifests itself in two ways when applied to poorly conditioned matrices:

- $||\mathbf{A} - \hat{\mathbf{Q}}\hat{\mathbf{R}}|| \gg \epsilon_{\text{mach}}$ (loss of accuracy)
- $||\hat{\mathbf{Q}}^T\hat{\mathbf{Q}} - \mathbf{I}|| \gg \epsilon_{\text{mach}}$ (loss of orthogonality)

Let us see a simple example of this.

Example: Consider the general 2×2 matrix

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \tag{3.74}$$

Applying either of the Gram-Schmidt algorithm (whose steps are exactly the same for a 2×2 matrix), we have

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \to r_{11} = \sqrt{a^2 + c^2} \to \mathbf{A} = \begin{pmatrix} \frac{a}{\sqrt{a^2+c^2}} & b \\ \frac{c}{\sqrt{a^2+c^2}} & d \end{pmatrix} \tag{3.75}$$

then

$$r_{12} = \frac{ab+cd}{\sqrt{a^2+c^2}} \rightarrow \mathbf{A} = \begin{pmatrix} \frac{a}{\sqrt{a^2+c^2}} & \frac{c(bc-ad)}{a^2+c^2} \\ \frac{c}{\sqrt{a^2+c^2}} & \frac{a(ad-bc)}{a^2+c^2} \end{pmatrix} \tag{3.76}$$

We easily recognize $D = ad - bc$ appear in the second column vector. Let us assume, for the sake of simplicity, that $D > 0$ (a similar line of argument applies if $D < 0$). We then finally normalize the second vector to get

$$r_{22} = \frac{D}{\sqrt{a^2+c^2}} \rightarrow \mathbf{A} = \frac{1}{\sqrt{a^2+c^2}} \begin{pmatrix} a & -c \\ c & a \end{pmatrix} \tag{3.77}$$

It is trivial to verify that, in this exact case, the column vectors of $\mathbf{A}$ are indeed orthogonal. In addition, we can also check easily that $\mathbf{A} - \mathbf{QR} = 0$.

However, what happens when the matrix is nearly singular? In that case, recall that $D \simeq 0$ even when a, b, c and d themselves are of order unity. We also saw that, when performing floating point operations, the absolute error on the subtraction of two numbers is equal to machine error times the size of the largest number. In the calculations of the two components of the second column vector, the error on $D = ad - bc$ can therefore be as large as D if the latter is close to zero, and so the result would be an approximate matrix

$$\mathbf{A} = \begin{pmatrix} \frac{a}{\sqrt{a^2+c^2}} & \frac{c(-D+\epsilon_1)}{a^2+c^2} \\ \frac{c}{\sqrt{a^2+c^2}} & \frac{a(D+\epsilon_2))}{a^2+c^2} \end{pmatrix} \tag{3.78}$$

where ϵ_1 and ϵ_2 are order machine precision (if a, b, c and d are order unity). If $D = O(\epsilon_1) = O(\epsilon_2)$ as well, the second column can be quite far from being orthogonal to the first. In addition, r_{22} will be of the same order as D, ϵ_1 and ϵ_2, while r_{11} is of order a and/or c. The matrix $\mathbf{R}$ will then be very poorly conditioned, with a condition number $O(D^{-1})$. The relative error on the matrix multiplication $\mathbf{QR}$ is therefore of order $D^{-1}\epsilon_{\text{mach}} \gg \epsilon_{\text{mach}}$. $\square$

The lack of both accuracy and orthogonality in the calculation of the QR decomposition naturally carry over to the original Least Square problem we were trying to solve. Recall that, if $\hat{\mathbf{Q}}\hat{\mathbf{R}}$ is known, then we can solve the Least Square problem $\mathbf{Ax} = \mathbf{b}$ by the two-step method $\hat{\mathbf{Q}}^T\hat{\mathbf{Q}}\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$, and then solve the upper-triangular $n \times n$ problem $\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$. The first step uses the fact that $\hat{\mathbf{Q}}^T\hat{\mathbf{Q}} = \mathbf{I}$, so large errors may appear if this is no longer true because of loss of orthogonality. Compounding on this, the relative error of the second step is proportional to the condition number of $\hat{\mathbf{R}}$, which is large if the problem is poorly conditioned.

Luckily, the orthogonality problem of step 1 can be solved, even if the accuracy problem of step 2 cannot (for poorly conditioned matrices). Doing so requires a completely different approach to orthogonalization, using so-called Householder transformations.

Example: Let us consider to solve the same least squares problem of the land surveyors example given in Eq. 3.5, using Gram-Schmidt orthogonalization. The first step is to normalize the first column of $\mathbf{A}$:

$$r_{11} = ||\mathbf{a}_1||_2 = 1.7321, \;\; \mathbf{q}_1 = \frac{\mathbf{a}_1}{r_{11}} = \begin{bmatrix} 0.5774 \\ 0 \\ 0 \\ -0.5774 \\ -0.5774 \\ 0 \end{bmatrix}. \tag{3.79}$$

Calculating orthogonalization processes and subtractions in **Step 2** and **Step 3**, we first obtain

$$r_{12} = \mathbf{q}_1^T \mathbf{a}_2 = -0.5774, \;\; r_{13} = \mathbf{q}_1^T \mathbf{a}_3 = -0.5774, \tag{3.80}$$

where $\mathbf{a}_2$ and $\mathbf{a}_3$ are the second and the third column of $\mathbf{A}$. Continuing the remaining procedures in **Step 2** and **Step 3** we get:

$$\mathbf{r}_2 = \mathbf{a}_2 - r_{12}\mathbf{q}_1 = \begin{bmatrix} 0.3333 \\ 1 \\ 0 \\ 0.6667 \\ -0.3333 \\ -1 \end{bmatrix}, \;\; \mathbf{r}_3 = \mathbf{a}_3 - r_{13}\mathbf{q}_1 = \begin{bmatrix} 0.3333 \\ 0 \\ 1 \\ -0.3333 \\ 0.6667 \\ 1 \end{bmatrix}. \tag{3.81}$$

The resulting transformed matrix now has $\mathbf{q}_1$ for its first column, together with $\mathbf{r}_2$ and $\mathbf{r}_3$ for the unnormalized second and the third columns:

$$\begin{bmatrix} 0.5774 & 0.3333 & 0.3333 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.5774 & 0.6667 & -0.3333 \\ -0.5774 & -0.3333 & 0.6667 \\ 0 & -1 & 1 \end{bmatrix} \tag{3.82}$$

Let us abuse our naming strategy and let $\mathbf{a}_2$ and $\mathbf{a}_3$ be the second and the third columns of the new matrix now. Normalizing the second column gives

$$r_{22} = ||\mathbf{a}_2||_2 = 1.6330, \;\; \mathbf{q}_2 = \frac{\mathbf{a}_2}{r_{22}} = \begin{bmatrix} 0.2041 \\ 0.6124 \\ 0 \\ 0.4082 \\ -0.2041 \\ -0.6124 \end{bmatrix}. \tag{3.83}$$

If we evaluate the orthogonalization of the second column against the third column, we get:

$$r_{23} = \mathbf{q}_2^T \mathbf{a}_3 = -0.8165, \tag{3.84}$$

and hence we further obtain yet another residual vector

$$\mathbf{r}_3 = \mathbf{a}_3 - r_{23}\mathbf{q}_2 = \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \\ 0 \\ 0.5 \\ 0.5 \end{bmatrix}. \tag{3.85}$$

We now form another transformed matrix which has $\mathbf{q}_2$ for its second orthonormal column and $\mathbf{r}_3$ for its unnormalized third column:

$$\begin{bmatrix} 0.5774 & 0.2041 & 0.5 \\ 0 & 0.6124 & 0.5 \\ 0 & 0 & 1 \\ -0.5774 & 0.4082 & 0 \\ -0.5774 & -0.2041 & 0.5 \\ 0 & -0.6124 & 0.5 \end{bmatrix}. \tag{3.86}$$

Finally we normalize the last column, again abusing our naming strategy and let the third column be $\mathbf{a}_3$:

$$r_{33} = ||\mathbf{a}_3||_2 = 1.4142, \quad \mathbf{q}_3 = \frac{\mathbf{a}_3}{r_{33}} = \begin{bmatrix} 0.3536 \\ 0.3536 \\ 0.7071 \\ 0 \\ 0.3536 \\ 0.3536 \end{bmatrix}, \tag{3.87}$$

and by replacing the last column of the last transformed matrix with $\mathbf{q}_3$ results in the final transformed matrix

$$\begin{bmatrix} 0.5774 & 0.2041 & 0.3536 \\ 0 & 0.6124 & 0.3536 \\ 0 & 0 & 0.7071 \\ -0.5774 & 0.4082 & 0 \\ -0.5774 & -0.2041 & 0.3536 \\ 0 & -0.6124 & 0.3536 \end{bmatrix}. \tag{3.88}$$

Now collecting entries r_{ij} of $\mathbf{R}$, we form

$$\mathbf{R} = \begin{bmatrix} 1.7321 & -0.5774 & -0.5774 \\ & 1.6330 & -0.8165 \\ & & 1.4142 \end{bmatrix}, \tag{3.89}$$

which altogether provides the QR factorization:

$$\mathbf{A} = \mathbf{QR} = \begin{bmatrix} 0.5774 & 0.2041 & 0.3536 \\ 0 & 0.6124 & 0.3536 \\ 0 & 0 & 0.7071 \\ -0.5774 & 0.4082 & 0 \\ -0.5774 & -0.2041 & 0.3536 \\ 0 & -0.6124 & 0.3536 \end{bmatrix} \begin{bmatrix} 1.7321 & -0.5774 & -0.5774 \\ & 1.6330 & -0.8165 \\ & & 1.4142 \end{bmatrix}. \tag{3.90}$$

Computing the right hand side transformation, $\mathbf{Q}^T\mathbf{b}$, we obtain

$$\mathbf{Q}^T\mathbf{b} = \begin{bmatrix} -376 \\ 1200 \\ 3417 \end{bmatrix} = \mathbf{c}_1. \tag{3.91}$$

This allows us to solve the upper triangular system $\mathbf{Rx} = \mathbf{c}_1$ by back-substitution, resulting in the same solution as before:

$$\mathbf{x}^T = [1236, 1943, 2416]. \tag{3.92}$$

□

5.3. QR decomposition using Householder Transformations

See Chapter 10 of the textbook

5.3.1. Householder matrices Given a vector $\mathbf{v}$ with $||\mathbf{v}|| = 1$, its corresponding Householder matrix $\mathbf{H}$ is

$$\mathbf{H} = \mathbf{I} - 2\mathbf{v}\mathbf{v}^T \tag{3.93}$$

We see that from the definition, $\mathbf{H}$ is both

- symmetric, i.e., $\mathbf{H}^T = \mathbf{H}$, and
- orthogonal, i.e., $\mathbf{H}^T\mathbf{H} = \mathbf{I}$, hence $\mathbf{H}^T = \mathbf{H}^{-1}$.

The geometrical interpretation of matrix $\mathbf{H}$ is actually quite simple: it is a **reflection** across the plane perpendicular to $\mathbf{v}$. To see this, note that for any vector parallel to $\mathbf{v}$ (namely $\mathbf{w} = \alpha\mathbf{v}$),

$$\mathbf{H}\mathbf{w} = \alpha\mathbf{H}\mathbf{v} = \alpha(\mathbf{v} - 2\mathbf{v}\mathbf{v}^T\mathbf{v}) = -\alpha\mathbf{v} = -\mathbf{w} \tag{3.94}$$

while for any vector $\mathbf{w}$ perpendicular to $\mathbf{v}$, we have

$$\mathbf{H}\mathbf{w} = \mathbf{w} - 2\mathbf{v}\mathbf{v}^T\mathbf{w} = \mathbf{w} \tag{3.95}$$

These two properties define an orthogonal reflection about a plane: any vector parallel to the plane exactly reflected ($\mathbf{w} \to -\mathbf{w}$), while vectors perpendicular to that plane remain invariant .

Remark: First recall that the orthogonal projector onto span($\mathbf{v}$) is given by

$$\mathbf{P} = \mathbf{v}(\mathbf{v}^T\mathbf{v})^{-1}\mathbf{v}^T = \frac{\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}}. \tag{3.96}$$

Also the orthogonal complement projector onto span$(\mathbf{v})^\perp$ is given by

$$\mathbf{P}_\perp = \mathbf{I} - \mathbf{P} = \mathbf{I} - \frac{\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}}. \tag{3.97}$$

This projector $\mathbf{P}_\perp$ gives the projection of $\mathbf{a}$ onto the hyperplane,

$$\mathbf{P}_\perp \mathbf{a} = (\mathbf{I} - \mathbf{P})\mathbf{a} = \mathbf{a} - \mathbf{v}\frac{\mathbf{v}^T\mathbf{a}}{\mathbf{v}^T\mathbf{v}}, \tag{3.98}$$

which is only the half way through to the desired location, the first coordinate axis in the current example.

In order to reach the first coordinate axis we therefore need to go twice as far, which gives us our final form of $\mathbf{H}$:

$$\mathbf{H} = \mathbf{I} - 2\frac{\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}}. \tag{3.99}$$

The full design construction is illustrated in Fig. 5, where the hyperplane is given by $\text{span}(\mathbf{v})^\perp = \{\mathbf{x} : \mathbf{v}^T\mathbf{x} = \mathbf{0}\}$, for some $\mathbf{v} \neq \mathbf{0}$.

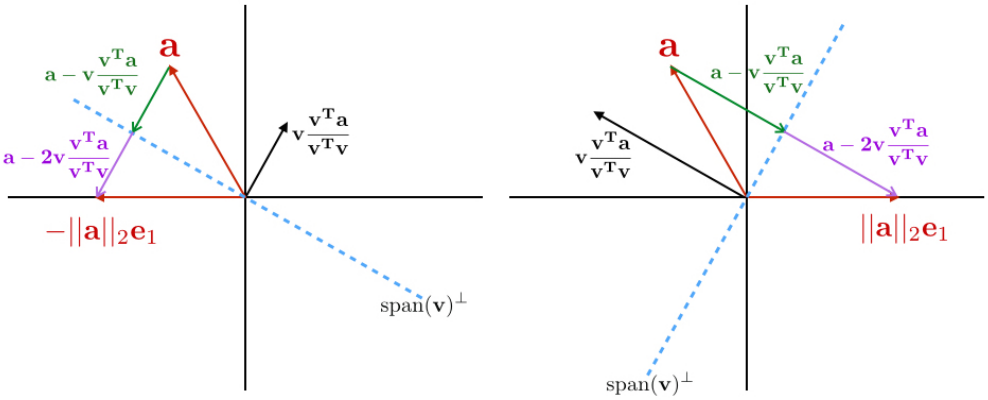

Figure 5. Geometric interpretation of Householder transformation as reflection.

Note: We should make a quick comment on the choice of signs of $\alpha = \pm||\mathbf{a}||_2$ now. Depending the choice of the sign, we get the closer transformation $-||\mathbf{a}||_2\mathbf{e}_1$ from $\mathbf{a}$ (shown in the left panel in Fig. 5), or the farther one $||\mathbf{a}||_2\mathbf{e}_1$ from $\mathbf{a}$ (shown in the right panel in Fig. 5). Both choices should work just fine in principle, however, we know from experience that dealing with subtraction that results in small in magnitude, i.e., $\mathbf{v} = \mathbf{a} - \alpha\mathbf{e}_1$, is prone to numerical errors due to finite-precision arithmetic. In this reason, we prefer to choose the sign for α that yields the point on the first coordinate axis as farther away as possible from $\mathbf{a}$. □

5.3.2. Relationship between Householder transformations and QR decomposition. A QR factorization based on Householder transformations is sometimes known as the **orthogonal triangularization** of $\mathbf{A}$, by contrast with the Gram-Schmidt algorithm which performs a triangular orthogonalization (see above). The terminology implies that $\mathbf{A}$ is now gradually transformed into an upper

triangular matrix $\mathbf{R}$ by successive application of orthogonal transformations, of the kind

$$\mathbf{Q}_n\mathbf{Q}_{n-1}\dots\mathbf{Q}_1\mathbf{A} = \mathbf{R} \tag{3.100}$$

The product of orthogonal matrices $\mathbf{Q}_n\mathbf{Q}_{n-1}\dots\mathbf{Q}_1$ is also orthogonal, so if we call it $\mathbf{Q}^T$, we then have

$$\mathbf{A} = \mathbf{QR} \text{ where } \mathbf{Q}^T = \mathbf{Q}_n\mathbf{Q}_{n-1}\dots\mathbf{Q}_1 \tag{3.101}$$

Note that the Householder QR decomposition can only perform *full QR decompositions* which is why we have now begun to refer to the matrices as $\mathbf{Q}$ and $\mathbf{R}$ as defined in the last lecture, with $\mathbf{Q}$ an $m\times m$ matrix, and $\mathbf{R}$ an $m\times n$ matrix.

Based on this idea, we want to find a way of gradually zeroing out the subdiagonal elements of $\mathbf{A}$, which is superficially similar to what Gaussian elimination or LU decomposition do. The important difference however is that these operations must be done by orthogonal transformations! So how do we construct them? Here again, having a geometric mind can really help.

In what follows, it will be useful to remember that orthogonal operations on vectors are norm-preserving, since $||\mathbf{Qx}|| = \sqrt{(\mathbf{Qx})^T\mathbf{Qx}} = \sqrt{\mathbf{x}^T\mathbf{x}} = ||\mathbf{x}||$, using the fact that $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$. In the first step of transforming $\mathbf{A}$ into an upper triangular matrix, we want to zero out all the entries below the first one. Geometrically speaking, this involves creating an orthogonal transformation that takes the first column vector $\mathbf{a}_1$, and returns a vector $\mathbf{Q}_1\mathbf{a}_1 = \pm||\mathbf{a}_1||\mathbf{e}_1$, i.e. with the same norm but pointing in the $\mathbf{e}_1$ direction. Since orthogonal transformations are either rotations or reflections, we see that a simple way of constructing $\mathbf{Q}_1$ is to construct the relevant plane that reflects $\mathbf{a}_1$ onto $\pm||\mathbf{a}_1||\mathbf{e}_1$, as shown in Figure 6.

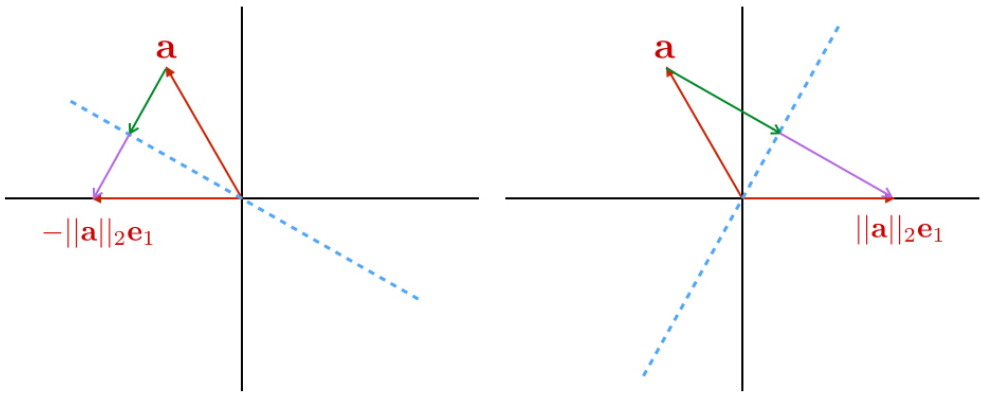

Figure 6. Geometric interpretation of Householder transformation $\mathbf{H}$ as reflection. The transformation operator $\mathbf{H}$ is represented in two successive transformations denoted as the green arrow, followed by the purple arrow, across the hyperplane in dashed pale-blue line.

The left panel in Fig. 6 shows the principle of the method that reflects the given vector $\mathbf{a}_1$ to produce $-||\mathbf{a}_1||\mathbf{e}_1$. The reflection is established by bisecting the angle between $\mathbf{a}_1$ and the the first coordinate axis. Obviously, the norm is well preserved in this reflection.

Clearly, another transformation is also available, as indicated on the right panel in Fig. 6, where in this case the vector $\mathbf{a}_1$ is reflected onto $||\mathbf{a}_1||\mathbf{e}_1$.

Both choices of reflection should work just fine in principle, but the first can be shown to be more prone to numerical errors due to finite-precision arithmetic than the second. For this reason, we prefer to choose the sign $\pm$ that is equal to **minus** the sign of $\mathbf{a}_1^T\mathbf{e}_1$ (i.e. $-\text{sign}(a_{11})$).

Geometrically speaking, we therefore see that we need a Householder transformation (an orthogonal reflection across a plane) $\mathbf{H}_1$, such that

$$\mathbf{H}_1\mathbf{a}_1 = -s_1\mathbf{e}_1 \tag{3.102}$$

where $s_1 = \text{sign}(a_{11})||\mathbf{a}_1||$ is a signed norm of $\mathbf{a}_1$. The correct vector $\mathbf{v}_1$ that can perform this transformation is found by solving

$$(\mathbf{I} - 2\mathbf{v}_1\mathbf{v}_1^T)\mathbf{a}_1 = -s_1\mathbf{e}_1 \tag{3.103}$$

and requiring that $\mathbf{v}_1$ be normalized. The first condition implies that

$$\mathbf{v}_1 = \frac{\mathbf{a}_1 + s_1\mathbf{e}_1}{2\mathbf{v}_1^T\mathbf{a}_1} \tag{3.104}$$

which looks awkward because of the term in the denominator. But since this term is just a constant, and since we require $\mathbf{v}_1$ to be normalized, we then simply have

$$\mathbf{v}_1 = \frac{\mathbf{a}_1 + s_1\mathbf{e}_1}{||\mathbf{a}_1 + s_1\mathbf{e}_1||} \tag{3.105}$$

which can now be constructed easily once $\mathbf{a}_1$ is known. Note that, effectively, we have

$$\mathbf{v}_1 = (a_{11} + s_1, a_{21}, a_{31}, \ldots, a_{m1})^T/||\mathbf{v}_1|| \tag{3.106}$$

To summarize, we can zero out the elements below the diagonal in the first column of $\mathbf{a}_1$ simply by constructing $\mathbf{v}_1$, then applying the corresponding Householder reflection $\mathbf{H}_1$ to $\mathbf{A}$. The effect of $\mathbf{H}_1$ on the other columns of $\mathbf{A}$ on the other hand is benign (i.e. does not do anything special).

Next, we look at the second column. This time, we effectively want to transform $\mathbf{a}_2$ into a vector that lies in the plane formed by $\mathbf{e}_1$ and $\mathbf{e}_2$, *without moving the transformed vector* $\mathbf{a}_1$ *away from* $\mathbf{e}_1$. For $\mathbf{e}_1$ to be invariant by transformation $\mathbf{H}_2$, $\mathbf{e}_1$ must lie in the plane of reflection, and therefore be perpendicular to the vector $\mathbf{v}_2$. Hence $\mathbf{v}_2$ must have a null first entry. By analogy with the construction of $\mathbf{v}_1$, and following this last remark, we therefore try

$$\mathbf{v}_2 = (0, a_{22} + s_2, a_{32}, \ldots, a_{m2})^T/||\mathbf{v}_2|| \tag{3.107}$$

and $\mathbf{H}_2 = \mathbf{I} - 2\mathbf{v}_2\mathbf{v}_2^T$, where

$$s_2 = \text{sign}(a_{22}) \left(\sum_{k=2}^{m} a_{k2}^2 \right)^{1/2} \tag{3.108}$$

It can be verified that $\mathbf{H}_2$ indeed zeroes out the subdiagonal elements of the second column of $\mathbf{A}$, without modifying the first column. More generally, we then have at the j-th step, $\mathbf{H}_j = \mathbf{I} - 2\mathbf{v}_j\mathbf{v}_j^T$ where

$$\mathbf{v}_j = (0, \ldots, 0, a_{jj} + s_j, a_{j+1,j}, \ldots, a_{mj})^T / ||\mathbf{v}_j|| \tag{3.109}$$

and

$$s_j = \text{sign}(a_{jj}) \left(\sum_{k=j}^{m} a_{kj}^2 \right)^{1/2} \tag{3.110}$$

After n applications of the algorithm, we have

$$\mathbf{H}_n \ldots \mathbf{H}_1 \mathbf{A} = \mathbf{R} \tag{3.111}$$

where each $\mathbf{H}_j$ is orthogonal by construction, as required, and $\mathbf{Q}^T = \mathbf{H}_n \ldots \mathbf{H}_1$.

This then suggests the following QR decomposition algorithm:

Algorithm: Householder QR factorization algorithm:

```
do j = 1 to n
    ![loop over columns]
    s_j = sign(a_jj) sqrt(sum_{i=j}^{m} a_ij^2)
    !  [compute signed norm]
    v_j = [0, ... 0, a_jj + s_j, a_{j+1,j}, ... , a_mj]^T
    v_j = v_j / ||v_j||
    !  [compute Householder vector and normalize it]
    A = A - 2 v_j v_j^T A  ![Update A]
enddo
```

This algorithm gradually transforms $\mathbf{A}$ into $\mathbf{R}$, but does not compute nor save $\mathbf{Q}$. This turns out not to be needed as long as the $\mathbf{v}_j$ vectors are saved. Indeed, should we ever want to compute the action of $\mathbf{Q}$ or $\mathbf{Q}^T$ onto a vector $\mathbf{x}$, we simply have to remember their definitions, and the fact that each $\mathbf{H}_j$ matrix is symmetric and orthogonal at the same time. This implies

$$\begin{aligned} \mathbf{Q}^T\mathbf{x} = \mathbf{H}_n \ldots \mathbf{H}_1 \mathbf{x} &= (\mathbf{I} - 2\mathbf{v}_n\mathbf{v}_n^T) \ldots (\mathbf{I} - 2\mathbf{v}_1\mathbf{v}_1^T)\mathbf{x} \\ \mathbf{Q}\mathbf{x} = (\mathbf{H}_n \ldots \mathbf{H}_1)^{-1}\mathbf{x} = \mathbf{H}_1^{-1} \ldots \mathbf{H}_n^{-1}\mathbf{x} &= \mathbf{H}_1 \ldots \mathbf{H}_n \mathbf{x} \\ &= (\mathbf{I} - 2\mathbf{v}_1\mathbf{v}_1^T) \ldots (\mathbf{I} - 2\mathbf{v}_n\mathbf{v}_n^T)\mathbf{x} \end{aligned} \tag{3.112}$$

In each case, this implies repeated application of $\mathbf{H}_j$ to a vector, which can easily done as

$$\mathbf{H}_j\mathbf{x} = \mathbf{x} - 2\mathbf{v}_j(\mathbf{v}_j^T\mathbf{x}). \tag{3.113}$$

There are several options to save the vectors $\mathbf{v}_j$. One can, as in the example above, save them in a separate matrix $\mathbf{V}$ whose columns are $\mathbf{v}_j$. This is very simple, but not very memory efficient nor computationally efficient. A much more efficient way it to save the elements of $\mathbf{v}_j$ whose indices range from $j+1$ to m, in the column elements of the matrix $\mathbf{A}$ that have just been zeroed out. We then simply have to decide what to do with the diagonal elements of $\mathbf{A}$: we can either (i) choose to put all the elements of $\mathbf{R}$ including the diagonal elements in the corresponding upper triangular region of $\mathbf{A}$ and save the j-th elements of the $\mathbf{v}_j$, i.e., $\mathbf{v}_{jj} = a_{jj} + s_j$ vectors in a separate array, or (ii) the converse, namely to put the element $a_{jj} + s_j$ in diag($\mathbf{A}$) and return the diagonal elements of $\mathbf{R}$ (namely each $-s_j$) as a separate array. Some codes choose the first option (c.f. LAPACK) some choose the second (c.f. Numerical recipes), so it's really important to read the code description!

Finally, it is worth noting that the Householder QR algorithm is backward stable (see Chapter 16), with all the properties that this implies. The orthogonality of $\mathbf{Q}$ is guaranteed to be close to machine accuracy, but roundoff errors in $\mathbf{Q}$ and $\mathbf{R}$ mean that the reconstruction $\mathbf{QR}$ is not necessarily that close to $\mathbf{A}$ for poorly conditioned problem.

5.3.3. Solving a Least-Square problem using the QR Householder decomposition. While the Gram-Schmidt algorithm returns the reduced QR decomposition $\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$, which can then easily be used to solve the $n \times n$ exact problem $\hat{\mathbf{R}} = \hat{\mathbf{Q}}^T\mathbf{b}$, the Householder method returns the full QR decompostion $\mathbf{A} = \mathbf{QR}$. While the equality $\mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$ is well-defined in the sense that it equates two $m-$long vectors, the problem appears ill-posed because there are only n unknowns (the n components of $\mathbf{x}$). This mismatch, however, does not matter as long as we remember that $\mathbf{R}$ contains $\hat{\mathbf{R}}$ in its first n rows, and only zeros afterwards, while the matrix $\mathbf{Q}$ contains $\hat{\mathbf{Q}}$ in its first columns, and other orthogonal vectors after that. This means that we can recover the original $\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$ problem by looking *only* at the first n lines of the problem $\mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$ and ignoring the other lines.

Because the QR method using Householder transformation guarantees the orthogonality of $\mathbf{Q}$ to within machine accuracy, it is vastly preferred over Gram-Schmidt orthogonalization when used in the context of Least-Square problems. So one may wonder why ever use the Gram-Schmidt method? The answer to that question lies in the fact that the Householder algorithm is inherently serial and cannot be parallelized: each column j must be zeroed out before work can be done on column $j+1$. This is not true of *modified* Gram-Schmidt algorithm, where the entire matrix $\mathbf{A}$ is orthogonalized at the same time by the iterative process, rather than doing so one vector at a time. Some parallel QR algorithms therefore work with the Gram-Schmidt method instead, when it is perceived that the loss of accuracy is an acceptable price to pay for a vast gain in time.

Example: Let us now solve the land surveyor's least squares problem given by the system in Eq. 3.5 using Householder QR factorization:

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} \cong \begin{bmatrix} 1237 \\ 1941 \\ 2417 \\ 711 \\ 1177 \\ 475 \end{bmatrix} = \mathbf{b}. \tag{3.114}$$

Recall that the solution we assumed to know was given by

$$\mathbf{x}^T = [h_1, h_2, h_3] = [1236, 1943, 2416], \tag{3.115}$$

and let's see if we get this solution indeed.

The first Householder step is to construct the Householder vector $\mathbf{v}_1$ that annihilates the subdiagonal entries of the first column $\mathbf{a}_1$ of $\mathbf{A}$ with, in this case, $s_1 = ||\mathbf{a}_1||_2 = \sqrt{3} \cong 1.7321$,

$$\mathbf{v}_1 = \mathbf{a}_1 + s_1\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \\ -1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1.7321 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2.7321 \\ 0 \\ 0 \\ -1 \\ -1 \\ 0 \end{bmatrix}. \tag{3.116}$$

Applying the resulting $\mathbf{H}_1$ to the first column gives

$$\mathbf{H}_1\mathbf{a}_1 = \mathbf{a}_1 - 2\mathbf{v}_1\frac{\mathbf{v}_1^T\mathbf{a}_1}{\mathbf{v}_1^T\mathbf{v}_1} = \begin{bmatrix} -1.7321 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.117}$$

Applying $\mathbf{H}_1$ to the second and third columns and also the right hand side vector $\mathbf{b}$ in a similar way gives, respectively:

$$\mathbf{H}_1\mathbf{a}_2 = \mathbf{a}_2 - 2\mathbf{v}_1\frac{\mathbf{v}_1^T\mathbf{a}_2}{\mathbf{v}_1^T\mathbf{v}_1} = \begin{bmatrix} 0.5774 \\ 1 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix}, \tag{3.118}$$

$$\mathbf{H}_1\mathbf{a}_3 = \mathbf{a}_3 - 2\mathbf{v}_1\frac{\mathbf{v}_1^T\mathbf{a}_3}{\mathbf{v}_1^T\mathbf{v}_1} = \begin{bmatrix} 0.5774 \\ 0 \\ 1 \\ -0.2113 \\ 0.7887 \\ 1 \end{bmatrix}, \tag{3.119}$$

and

$$\mathbf{H}_1\mathbf{b} = \mathbf{b} - 2\mathbf{v}_1\frac{\mathbf{v}_1^T\mathbf{b}}{\mathbf{v}_1^T\mathbf{v}_1} = \begin{bmatrix} 376 \\ 1941 \\ 2417 \\ 1026 \\ 1492 \\ 475 \end{bmatrix}. \tag{3.120}$$

Putting all things together, we get

$$\mathbf{H}_1\mathbf{A} = \begin{bmatrix} -1.7321 & 0.5774 & 0.5774 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0.7887 & -0.2113 \\ 0 & -0.2113 & 0.7887 \\ 0 & -1 & 1 \end{bmatrix}, \quad \mathbf{H}_1\mathbf{b} = \begin{bmatrix} 376 \\ 1941 \\ 2417 \\ 1026 \\ 1492 \\ 475 \end{bmatrix}. \tag{3.121}$$

Next, we compute the second Householder vector $\mathbf{v}_2$ that annihilates the subdiagonal entries of the second column of $\mathbf{H}_1\mathbf{A}$ and leave the first entry 0.5774 unchanged (i.e., we choose all entries of $\mathbf{a}_2$ except for the first entry 0.5774):

$$\mathbf{a}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix}. \tag{3.122}$$

Then, with $s_2 = ||\mathbf{a}_2||_2 = 1.6330$, we get

$$\mathbf{v}_2 = \mathbf{a}_2 + s_2\mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1.6330 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 2.6330 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix}. \tag{3.123}$$

Now we apply $\mathbf{H}_2 = \mathbf{I} - 2\mathbf{v}_2\frac{\mathbf{v}_2^T}{\mathbf{v}_2^T\mathbf{v}_2}$ onto the second and the third columns of $\mathbf{H}_1\mathbf{A}$ as well as $\mathbf{H}_1\mathbf{b}$, where the actual operation begins from the second entry in both columns, keeping the first entries unchanged. Writing these out explicitly, we get:

$$\mathbf{H}_2\begin{bmatrix} 0.5774 \\ 1 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix} = \left(\mathbf{I} - 2\mathbf{v}_2\frac{\mathbf{v}_2^T}{\mathbf{v}_2^T\mathbf{v}_2}\right)\begin{bmatrix} 0.5774 \\ 1 \\ 0 \\ 0.7887 \\ -0.2113 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.5774 \\ -1.6330 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{3.124}$$

$$\mathbf{H}_2 \begin{bmatrix} 0.5774 \\ 0 \\ 1 \\ -0.2113 \\ 0.7887 \\ 1 \end{bmatrix} = \left(\mathbf{I} - 2\mathbf{v}_2 \frac{\mathbf{v}_2^T}{\mathbf{v}_2^T \mathbf{v}_2}\right) \begin{bmatrix} 0.5774 \\ 0 \\ 1 \\ -0.2113 \\ 0.7887 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.5774 \\ 0.8165 \\ 1 \\ 0.0333 \\ 0.7232 \\ 0.6899 \end{bmatrix}, \tag{3.125}$$

so that we have:

$$\mathbf{H}_2\mathbf{H}_1\mathbf{A} = \begin{bmatrix} -1.7321 & 0.5774 & 0.5774 \\ 0 & -1.6330 & 0.8165 \\ 0 & 0 & 1 \\ 0 & 0 & 0.0333 \\ 0 & 0 & 0.7232 \\ 0 & 0 & 0.6899 \end{bmatrix}, \quad \mathbf{H}_2\mathbf{H}_1\mathbf{b} = \begin{bmatrix} 376 \\ -1200 \\ 2417 \\ 85 \\ 1744 \\ 1668 \end{bmatrix}. \tag{3.126}$$

The last step now uses the third Householder vector $\mathbf{v}_3$ for annihilating the subdiagonal entries of the third column of $\mathbf{H}_2\mathbf{H}_1\mathbf{A}$ by considering

$$\mathbf{a}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0.0333 \\ 0.7232 \\ 0.6899 \end{bmatrix}. \tag{3.127}$$

This gives $s_3 = ||\mathbf{a}_3||_2 = 1.4142$ and hence we get

$$\mathbf{v}_3 = \mathbf{a}_3 + s_3\mathbf{e}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0.0333 \\ 0.7232 \\ 0.6899 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1.4142 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2.4142 \\ 0.0332 \\ 0.7231 \\ -0.6899 \end{bmatrix}. \tag{3.128}$$

Applying the final Householder transformation with $\mathbf{H}_3 = \mathbf{I} - 2\mathbf{v}_3 \frac{\mathbf{v}_3^T}{\mathbf{v}_3^T \mathbf{v}_3}$ to the third column of $\mathbf{H}_2\mathbf{H}_1\mathbf{A}$ gives

$$\mathbf{H}_3\mathbf{H}_2\mathbf{H}_1\mathbf{A} = \begin{bmatrix} -1.7321 & 0.5774 & 0.5774 \\ 0 & -1.6330 & 0.8165 \\ 0 & 0 & -1.4142 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \tag{3.129}$$

and

$$\mathbf{H}_3\mathbf{H}_2\mathbf{H}_1\mathbf{b} = \begin{bmatrix} 376 \\ -1200 \\ -3417 \\ 5 \\ 3 \\ 1 \end{bmatrix} = \mathbf{Q}^T\mathbf{b} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix}. \tag{3.130}$$

We now can solve the upper triangular system $\mathbf{Rx} = \mathbf{c}_1$ by back-substitution to obtain

$$\mathbf{x}^T = [h_1, h_2, h_3] = [1236, 1943, 2416]. \tag{3.131}$$

Finally, the minimum residual is given by $||\mathbf{r}||_2^2 = ||\mathbf{c}_2||_2^2 = 35$. $\square$