Section 4: Least squares

March 21, 2019

1 Least squares approximation of functions

We have discussed basic polynomial interpolation now. But as we have mentioned before, there are other means of approximating functions that do not involve any interpolation (at least, directly). One popular approach, due to its ease in solvability, is to find the nearest function within some given basis[1] in the least squares sense.

Let us elaborate on this a bit further. Suppose we are given a bunch of functions $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_n(x)$. The idea is to take linear combinations of these functions to express more complicated functions. That is,

$$f_n(x) = \sum_{k=1}^{n} a_k \phi_k(x)$$

We can use $f_n(x)$ as a means to approximate some "target" function $f(x)$. To express the error between $f_n(x)$ and $f(x)$, the following cost criteria can be used:

$$E(a) = \int_a^b w(x) \left(f(x) - \sum_{k=1}^{n} a_k \phi_k(x) \right)^2 dx \tag{1}$$

where $a = (a_1, a_2, \ldots, a_n)$ are the coefficients that characterize $f_n(x)$. The function $w(x) > 0$ is a weighting function. One can simply choose $w(x) = 1$

[1] Keep in mind that I am using the word basis somewhat loosely here.

where every point in the domain is weighted evenly, or choose to penalize certain parts of the domain more heavily. Regardless, after choosing the weighting, the end goal is to minimize the function in (1) over all possible choices:

$$E = \min_{a \in \mathbb{R}^n} E(a).$$

Matrix re-formulation We are going to rewrite the expression (1) in a convenient matrix notation which will be useful for later.

$$
\begin{aligned}
E &= \int_a^b w(x)\left(f(x) - \sum_{k=1}^{n} a_k \phi_k(x)\right)^2 dx \\
&= \int_a^b w(x)\left(f(x) - \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}\right)^2 dx \\
&= \int_a^b w(x)\left(f(x) - \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}\right)^2 dx \\
&= \int_a^b w(x)\left(f(x) - \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}\right)^T \\
&\quad \left(f(x) - \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}\right) dx \\
&= \int_a^b w(x)\left(f(x) - \begin{bmatrix}a_1 & \cdots & a_n\end{bmatrix}\begin{bmatrix}\phi_1(x)\\ \vdots \\ \phi_n(x)\end{bmatrix}\right) \\
&\quad \left(f(x) - \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}\right) dx \\
&= \int_a^b w(x) f^2(x) dx - 2\int_a^b w(x) f(x) \begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix} dx + \\
&\quad \int_a^b w(x)\begin{bmatrix}a_1 & \cdots & a_n\end{bmatrix}\left(\begin{bmatrix}\phi_1(x)\\ \vdots \\ \phi_n(x)\end{bmatrix}\begin{bmatrix}\phi_1(x) & \cdots & \phi_n(x)\end{bmatrix}\right)\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix} dx \\
&= \int_a^b w(x) f^2(x) dx - 2\begin{bmatrix}\int_a^b w(x)f(x)\phi_1(x)dx & \cdots & \int_a^b w(x)f(x)\phi_n(x)dx\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix} + \\
&\quad \begin{bmatrix}a_1 & \cdots & a_n\end{bmatrix}\begin{bmatrix}\int_a^b w(x)\phi_1(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_1(x)dx \\ \vdots & & \vdots \\ \int_a^b w(x)\phi_n(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_n(x)dx\end{bmatrix}\begin{bmatrix}a_1\\ \vdots \\ a_n\end{bmatrix}
\end{aligned}
$$

We end up with the expression:

$$E(a) = a^T S a - 2d^T a + c \tag{2}$$

where:

$$\begin{aligned} S &= \begin{bmatrix} \int_a^b w(x)\phi_1(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_1(x)dx \\ \vdots & & \vdots \\ \int_a^b w(x)\phi_n(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_n(x)dx \end{bmatrix} \\ d &= \begin{bmatrix} \int_a^b w(x)f(x)\phi_1(x)dx \\ \vdots \\ \int_a^b w(x)f(x)\phi_n(x)dx \end{bmatrix} \\ c &= \int_a^b w(x)f^2(x)dx \end{aligned}$$

This is called a quadratic form.

Quadratic forms and positive definite matrices. Remember, our end goals was to find $a \in \mathbb{R}^n$ which minimizes the quadratic form (2). For that purpose let us apply some tricks we have learned in Gaussian elimination.

$$\begin{aligned} E(a) &= a^T S a - 2d^T a + c \\ &= \begin{bmatrix} a^T & 1 \end{bmatrix} \begin{bmatrix} S & -d \\ -d^T & c \end{bmatrix} \begin{bmatrix} a^T \\ 1 \end{bmatrix} \\ &= \left(\begin{bmatrix} a^T & 1 \end{bmatrix} \begin{bmatrix} I & 0 \\ -d^T S^{-1} & 1 \end{bmatrix} \right) \left(\begin{bmatrix} I & 0 \\ d^T S^{-1} & 1 \end{bmatrix} \begin{bmatrix} S & -d \\ -d^T & c \end{bmatrix} \right) \begin{bmatrix} a \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} a^T - d^T S^{-1} & 1 \end{bmatrix} \begin{bmatrix} S & -d \\ 0 & c - d^T S^{-1} d \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} a^T - d^T S^{-1} & 1 \end{bmatrix} \left(\begin{bmatrix} S & -d \\ 0 & c - d^T S^{-1} d \end{bmatrix} \begin{bmatrix} I & S^{-1} d \\ & 1 \end{bmatrix} \right) \left(\begin{bmatrix} I & -S^{-1} d \\ & 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} \right) \\ &= \begin{bmatrix} (a - S^{-1} d)^T & 1 \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & c - d^T S^{-1} d \end{bmatrix} \begin{bmatrix} a - S^{-1} d \\ 1 \end{bmatrix} \\ &= (a - S^{-1} d)^T S (a - S^{-1} d) + c - d^T S^{-1} d \end{aligned}$$

We may have done something potentially in the above derivation. For instance, does the inverse of S even exist!? Obviously, the answer is yes,

otherwise we would not make you go through the pain of this derivation. We have the following result.

Theorem 1. *Let $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_n(x)$ be a set of linearly independent functions. Then the inverse of S exists, and*

$$E = \min_{a \in \mathbb{R}^n} E(a)$$

is uniquely solved by $a = S^{-1}d$.

Proof. If S is not invertible, then there must exist some vector $a \neq 0$ for which $Sa = 0$, hence also $a^T Sa = 0$. We will show that this is not possible if $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_n(x)$ are linearly independent. First of all, observe that for any $a \in \mathbb{R}^n$:

$$\begin{aligned}
a^T Sa &= \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \begin{bmatrix} \int_a^b w(x)\phi_1(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_1(x)dx \\ \vdots & & \vdots \\ \int_a^b w(x)\phi_n(x)\phi_1(x)dx & \cdots & \int_a^b w(x)\phi_n(x)\phi_n(x)dx \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \\
&= \int_a^b w(x) \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \left(\begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_n(x) \end{bmatrix} \begin{bmatrix} \phi_1(x) & \cdots & \phi_n(x) \end{bmatrix} \right) \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} dx \\
&= \int_a^b w(x) \left(\begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_n(x) \end{bmatrix} \right) \left(\begin{bmatrix} \phi_1(x) & \cdots & \phi_n(x) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \right) dx \\
&= \int_a^b w(x) \left(\sum_{k=1}^n a_k \phi_k(x) \right) \left(\sum_{k=1}^n a_k \phi_k(x) \right) dx \\
&= \int_a^b w(x) \left(\sum_{k=1}^n a_k \phi_k(x) \right)^2 dx \\
&\geq 0.
\end{aligned}$$

The last conclusion follows from the fact that $w(x) \geq 0$ and:

$$\left(\sum_{k=1}^n a_k \phi_k(x) \right)^2 \geq 0.$$

In fact $a^T S a = 0$, only if:

$$\sum_{k=1}^{n} a_k \phi_k(x) = 0$$

Since $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_n(x)$ are linearly independent, this can happen only when $a = 0$, thereby confirming that S has to be invertible. Now to show that $a = S^{-1}d$, notice in the formula that in the expression

$$E(a) = (a - S^{-1}d)^T S(a - S^{-1}d) + c - d^T S^{-1} d$$

the term $(a - S^{-1}d)^T S(a - S^{-1}d) \geq 0$ and only equal to zero when $a - S^{-1}d = 0$, the other is a constant for which we dont have any control over. □

Discrete least squares Most likely, in your linear algebra classes you looked at a different type of least squares problem. Something of the form:

$$E = \min_{x \in \mathbb{R}^{\kappa}} \|b - Ax\|_2^2 \tag{3}$$

where $A \in \mathbb{R}^{m \times n}$, where typically $m >> n$ (i.e. an overdetermined system). The expression (1) can be reduced to this problem.

Dirac delta function:

$$w(x) = \sum_{k=1}^{m} \delta(x - x_i)$$

$$
\begin{aligned}
E(a) &= \int_a^b w(x)\left(f(x)-\sum_{k=1}^{n} a_k\phi_k(x)\right)^2 dx \\
&= \int_a^b \sum_{k=1}^{m}\delta(x-x_i)\left(f(x)-\sum_{k=1}^{n} a_k\phi_k(x)\right)^2 dx \\
&= \int_a^b \sum_{k=1}^{m}\delta(x-x_i)\left(f(x)-\sum_{k=1}^{n} a_k\phi_k(x)\right)^2 dx \\
&= \sum_{k=1}^{m}\left(\int_a^b \delta(x-x_i)\left(f(x)-\sum_{k=1}^{n} a_k\phi_k(x)\right)^2 dx\right) \\
&= \sum_{k=1}^{m}\left(f(x_i)-\sum_{k=1}^{n} a_k\phi_k(x_i)\right)^2 \\
&= \left\| \begin{bmatrix} f(x_1)-\sum_{k=1}^{n} a_k\phi_k(x_1) \\ \vdots \\ f(x_n)-\sum_{k=1}^{n} a_k\phi_k(x_n) \end{bmatrix} \right\|^2 \\
&= \left\| \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} - \begin{bmatrix} \phi_1(x_1) & \cdots & \phi_n(x_1) \\ \vdots & & \vdots \\ \phi_1(x_n) & \cdots & \phi_n(x_n) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \right\|^2
\end{aligned}
$$

From here one can derive the quadratic form again and repeat the same procedure as before. The familiar $x = (A^TA)^{-1}Ab$ would have been obtained. There is another way of solving the least squares which is numerically more preferred. This is what we will discuss next.

2 QR factorization

Let $A \in \mathbb{R}^{m\times n}$ be a matrix with $m > n$ (i.e. a tall matrix), then:

$$A = Q\begin{bmatrix} R \\ 0 \end{bmatrix} \tag{4}$$

where $Q \in \mathbb{R}^{m\times m}$ is a orthonormal matrix ($Q^TQ = QQ^T = I$) and $\mathbb{R}^{n\times n}$ is an upper triangular matrix. Like a lot linear algebra, this factorization of a matrix is named after the symbols which are typically used to express them: the QR factorization. Sometimes, the matrix $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$, where $Q_1 \in \mathbb{R}^{n\times n}$

QR factorizaton and least squares. The QR factorization is useful for solving least squares problems. Consider again the discrete least squares problem:

$$E = \min_{x\in\mathbb{R}^\kappa} \|b - Ax\|_2$$

If Q denotes the orthogonal matrix associated with QR factorization of A, i.e. as in (4), then we can perform the following manipulations:

$$\begin{aligned}
E &= \min_{x\in\mathbb{R}^\kappa} \|b - Ax\|_2^2 \\
&= \min_{x\in\mathbb{R}^\kappa} (b - Ax)^T(b - Ax) \\
&= \min_{x\in\mathbb{R}^\kappa} (b - Ax)^T QQ^T(b - Ax) \\
&= \min_{x\in\mathbb{R}^\kappa} (Q^T(b - Ax))^T(Q^T(b - Ax)) \\
&= \min_{x\in\mathbb{R}^\kappa} \left(Q^Tb - \begin{bmatrix} R \\ 0 \end{bmatrix} x\right)^T \left(Q^Tb - \begin{bmatrix} R \\ 0 \end{bmatrix} x\right) \\
&= \min_{x\in\mathbb{R}^\kappa} \left(Q^Tb - \begin{bmatrix} Rx \\ 0 \end{bmatrix}\right)^T \left(Q^Tb - \begin{bmatrix} Rx \\ 0 \end{bmatrix}\right)
\end{aligned}$$

Now split:

$$Q^Tb = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \end{bmatrix}$$

We may continue:

$$\begin{aligned}
E &= \min_{x\in\mathbb{R}^\kappa} \left(\begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \end{bmatrix} - \begin{bmatrix} Rx \\ 0 \end{bmatrix}\right)^T \left(\begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \end{bmatrix} - \begin{bmatrix} Rx \\ 0 \end{bmatrix}\right) \\
&= \left(\hat{b}_1 - Rx\right)^T \left(\hat{b}_1 - Rx\right) + \hat{b}_2^T\hat{b}_2 \\
&= \left\|Rx - \hat{b}_1\right\|^2 + \left\|\hat{b}_2\right\|^2
\end{aligned}$$

Now the last expression is interesting! Through the QR factorization, we were able to transform the square of the cost function to:

$$E = \left\| Rx - \hat{b}_1 \right\|^2 + \left\| \hat{b}_2 \right\|^2$$

This is something we can solve. Under very mild conditions on the matrix A, R will turn out to be an invertible matrix. The first term can therefore be set to zero and this our solution the least squares probem.

The Gram schmidt process. To someone who is already familiar to the Gram schmidt process (and if you are not, you will be learning it now!), the QR factorization should not appear as some new concept. Let $\vec{a}_1, \ldots, \vec{a}_m \in \mathbb{R}^n$ denote the columns of $A \in \mathbb{R}^{n \times m}$. We see that:

$$\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \cdots & \vec{a}_m \end{bmatrix} = \left[\begin{array}{ccc|ccc} \vec{q}_1 & \cdots & \vec{q}_m & \vec{q}_{m+1} & \cdots & \vec{q}_n \end{array}\right] \left[\begin{array}{ccc} r_{11} & \cdots & r_{1m} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & r_{mm} \\ \hline 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{array}\right]$$

where $\vec{q}_1, \ldots, \vec{q}_n \in \mathbb{R}^n$ are the orthonormal columns of Q. The above is equivalent to:

$$\begin{aligned} \vec{a}_1 &= r_{11}\vec{q}_1 \\ \vec{a}_2 &= r_{12}\vec{q}_1 + r_{22}\vec{q}_2 \\ &\vdots \\ \vec{a}_m &= r_{1m}\vec{q}_1 + r_{2m}\vec{q}_2 + \ldots + r_{mm}\vec{q}_m \end{aligned}$$

In other words, the QR factorization just re-expresses the columns of A in terms of an orthonormal basis. Such a basis can be obtained from the Gram-Schmidt process.

To introduce the Gram-Schmidt process, define the inner product as:

$$\langle \vec{x}, \vec{y} \rangle := \vec{x}^T \vec{y} = \sum_{k=1}^{n} x_k y_k \tag{5}$$

and observe that:

$$\|x\|^2 = \langle \vec{x}, \vec{x} \rangle = \vec{x}^T \vec{x} = \sum_{k=1}^{n} x_k^2 \tag{6}$$

One must understand that the inner product generalizes the notion of angles between vectors. Indeed, when the dimension of the space is 3, we have that $\langle \vec{x}, \vec{y} \rangle \equiv x \cdot y$, i.e. the familiar dot product. You may know from vector calculus that:

$$x \cdot y = \|x\| \, \|y\| \cos\theta$$

Hence,

$$\frac{\langle \vec{x}, \vec{y} \rangle}{\|x\| \, \|y\|}$$

quantifies angles between the vectors $\vec{x}$ and $\vec{y}$. We are digressing a bit, but this generalization is extremely valuable, because the definition of inner products can also be extended also to functions. That is, you can interpret functions as vectors and this allows us to construct, for instance, orthogonal polynomials[2]

Anyway, to explain Gram schmidt in the simplest way possible, suppose we have three vectors $\vec{a}_1, \vec{a}_2, \vec{a}_3 \in \mathbb{R}^3$ that span the entire space. We can use these three vectors to construct an orthonormal basis $\vec{q}_1, \vec{q}_2, \vec{q}_3 \in \mathbb{R}^3$. Simply call, $\vec{q}_1 = \frac{1}{\|\vec{a}_1\|} \vec{a}_1$. To construct $\vec{q}_2$, note that it has to be orthogonal $\vec{q}_1$. First, recognize that:

$$\langle \vec{a}_2, \vec{q}_1 \rangle \, \vec{q}_1 = \text{Proj}_{\vec{q}_1} \vec{a}_2$$

is the orthogonal projection of $\vec{a}_2$ onto the one dimensional subspace spanned by $\vec{a}_1$. and:

$$\tilde{q}_2 = \vec{a}_2 - \langle \vec{a}_2, \vec{q}_1 \rangle \, \vec{q}_1$$

is a vector which is orthogonal to $\vec{q}_1$. To normalize it, we simply do:

$$\vec{q}_2 = \frac{\vec{q}_2}{\|\tilde{q}_2\|}.$$

This gives us the second vector we need. To find the third, we apply the same ideas:

$$\begin{aligned} \tilde{q}_3 &= \vec{a}_3 - \langle \vec{a}_3, \vec{q}_1 \rangle \, \vec{q}_1 - \langle \vec{a}_3, \vec{q}_2 \rangle \, \vec{q}_2 \\ &= \vec{a}_3 - \text{Proj}_{\vec{q}_1} \vec{a}_3 - \text{Proj}_{\vec{q}_2} \vec{a}_3 \end{aligned}$$

[2]We have already seen one type of orthogonal polynomials in our lectues, the Chebyshev polynomials.

Normalizing: $\vec{q}_3 = \tilde{q}_3/\|\tilde{q}_3\|$ provides us the third orthogonal vector.

Let us generalize this. Suppose we are now given a bunch of vectors $\vec{a}_1, \ldots, \vec{a}_m \in \mathbb{R}^n$ that span some subspace in $\mathbb{R}^n$. The Gram Schmidt process for m vectors is descibed by:

$$\begin{aligned}
\vec{q}_1 &= \frac{\tilde{q}_1}{\|\tilde{q}_1\|}, \quad \tilde{q}_1 = \vec{a}_1 \\
\vec{q}_2 &= \frac{\tilde{q}_2}{\|\tilde{q}_2\|}, \quad \tilde{q}_2 = \vec{a}_2 - \langle \vec{q}_1, \vec{a}_2 \rangle \, \vec{q}_1 \\
&\vdots \\
\vec{q}_m &= \frac{\tilde{q}_m}{\|\tilde{q}_m\|}, \quad \tilde{q}_m = \vec{a}_m - \sum_{k=1}^{m-1} \langle \vec{q}_k, \vec{a}_m \rangle \, \vec{q}_k
\end{aligned}$$

The vectors q_1, …, q_m are the first m columns of Q. You may ask, what are the entries of R? Notice that:

$$\begin{aligned}
\vec{q}_1 &= \frac{1}{r_{11}} \vec{a}_1 \\
\vec{q}_2 &= \frac{1}{r_{22}} \left(\vec{a}_2 - r_{12} \vec{q}_1 \right) \\
&\vdots \\
\vec{q}_m &= \frac{1}{r_{mm}} \left(\vec{a}_m - \left(r_{1m} \vec{q}_1 + r_{2m} \vec{q}_2 + \ldots + r_{(m-1)m} \vec{q}_{m-1} \right) \right.
\end{aligned}$$

We see that:

$$r_{ij} = \begin{cases} \|\tilde{q}_i\| & i = j \\ \langle \vec{q}_i, \vec{a}_j \rangle & i \neq j \end{cases} \tag{7}$$

In matrix notation, the Gram-Schmidt process can be described as follows:

$$
\begin{aligned}
A &= \begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \cdots & \vec{a}_m \end{bmatrix} \\
&= \left(\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \cdots & \vec{a}_m \end{bmatrix} \begin{bmatrix} \frac{1}{r_{11}} & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \right) \begin{bmatrix} r_{11} & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \\
&= \begin{bmatrix} \vec{q}_1 & \vec{a}_2 & \cdots & \vec{a}_m \end{bmatrix} \begin{bmatrix} r_{11} & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \\
&= \left(\begin{bmatrix} \vec{q}_1 & \vec{a}_2 & \cdots & \vec{a}_m \end{bmatrix} \begin{bmatrix} 1 & -\frac{r_{21}}{r_{22}} & & \\ & \frac{1}{r_{22}} & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \right) \left(\left(\begin{bmatrix} 1 & r_{21} & & \\ & r_{22} & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \begin{bmatrix} r_{11} & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \right) \right) \\
&= \begin{bmatrix} \vec{q}_1 & \vec{q}_2 & \cdots & \vec{a}_m \end{bmatrix} \begin{bmatrix} r_{11} & r_{21} & & \\ & r_{22} & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}
\end{aligned}
$$

Etcetera. The matrix A is slowly transformed into a matrix with orthonormal columns through multiplication by upper triangular "Gauss"-like transforms from the right. A direct implementation of this process is however numerically unstable. There are ways to stabilize it (also known as modified Gram Schmidt) though, but we will look into a different method of obtaining the QR factorization.

Householder's algorithm. As we have observed, the Gram Schmidt algorithm obtains the QR factorization by a sequence triangular transformations from the right. This converts the A matrix slowly into an orthogonal matrix. Alternatively, we may choose to convert A matrix into an upper triangular matrix, by a sequence of orthonormal transformations from the left:

$$Q_n \cdots Q_1 A = R.$$

This fundamental shift in perspective gives rise to Householder's algorithm.

The nice thing about this algorithm is that the columns of A are multiplied by orthonormal matrices. This is numerically desirable (although not completely free of pit-falls!) because the multiplication by orthonormal matrices are well conditioned operations (the condition number is in fact equal to one). To build a mental model of what the Householder algorithm does, assume:

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

The sequence of transformation introduce the following zeros:

$$Q_1 A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}, \quad Q_2 Q_1 A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix}, \quad Q_3 Q_2 Q_1 A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

What is left for us to figure out how to pick the matrices Q_1, Q_2, Q_3. This is where we need the concept of a Householder transform, or a reflection matrix. But before that, recall that multiplying a vector by an orthonormal matrix $Q \in \mathbb{R}^{n \times n}$ always preserves length of that vector. Indeed,

$$\|Qx\|^2 = x^T Q^T Q x = x^T x = \|x\|^2 .$$

A specific geometric operation on a vector which preserves length is a reflection. For example, consider the vector $x \in \mathbb{R}^n$, and suppose we would like to reflect it accross the hyper-plane:

$$H_v := \left\{ x \in \mathbb{R}^n \quad | \quad x^T v = 0 \right\}, \quad \|v\| = 1 \tag{8}$$

The appropriate matrix which does that job is given by:

$$Q(v) = I - 2vv^T \tag{9}$$

This is called a Householder transform. Now let us look at the first column of A, i.e. $\vec{a}_1$. For our first Householder transform, we clearly want to choose

our unit vector $v \in \mathbb{R}^n$ in such a way so that $Q(v)\vec{a}_1 = \|\vec{a}_1\| e_1$, where $e_1 = \begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}^T$. This is achieved by simply setting:

$$v = \frac{\tilde{v}}{v}, \quad \tilde{v} = \|\vec{a}_1\| e_1 - \vec{a}_1.$$

Our first Gauss transform is $Q_1 = Q(\|\vec{a}_1\| e_1 - \vec{a}_1 / \|\|\vec{a}_1\| e_1 - \vec{a}_1|)$. But what about all the others? Let us look at the process general.

In general, we start with:

$$A = \left[\begin{array}{c|cccc} a_{11} & a_{12} & a_{13} & \cdots & a_{1m} \\ \hline a_{21} & a_{22} & a_{23} & \cdots & a_{2m} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nm} \end{array}\right]$$

After the first Householder transform $Q_1 = Q(\|\vec{a}_1\| e_1 - \vec{a}_1 / \|\|\vec{a}_1\| e_1 - \vec{a}_1|)$, we see that:

$$Q_1 A = \left[\begin{array}{c|cccc} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ \hline 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2m}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & \cdots & a_{3m}^{(1)} \\ 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & a_{n3}^{(1)} & \cdots & a_{nm}^{(1)} \end{array}\right]$$

Define: $\vec{a}_1^{(1)} = \begin{bmatrix} a_{22}^{(1)} & a_{22}^{(1)} & \cdots & a_{n2}^{(1)} \end{bmatrix}^T$ The second householder transform is:

$$Q_2 = \begin{bmatrix} 1 & & 0 \\ 0 & Q(\left\|\vec{a}_1^{(1)}\right\| e_1 - \vec{a}_1^{(1)} / \left\|\left\|\vec{a}_1^{(1)}\right\| e_1 - \vec{a}_1^{(1)}\right\|) \end{bmatrix}$$

Which gives:

$$Q_2 Q_1 A = \left[\begin{array}{cc|ccc} r_{11} & r_{12} & r_{13} & \cdots & r_{1m} \\ 0 & r_{22} & r_{23} & \cdots & r_{2m} \\ \hline 0 & 0 & a_{33}^{(2)} & \cdots & a_{3m}^{(2)} \\ 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & a_{n3}^{(2)} & \cdots & a_{nm}^{(2)} \end{array}\right]$$

I believe it should be clear how to continue the process untill A is turned into a triangular matrix.

Making Householder's algorithm stable. What we left out in the previous discussion, is that there are two ways of using Householder reflections. For our first Householder transform (the same applies however to all of them), we can either reflect the vector $\vec{a}_1$ (the first column of the matrix) to the position $\|\vec{a}_1\| e_1$ or $-\|\vec{a}_1\| e_1$. Both are mathematically equivalent. One of them is however better numerically. If the vector $\vec{a}_1$ is close to $\|\vec{a}_1\| e_1$, then $v = \|\vec{a}_1\| e_1 - \vec{a}_1$ may end up suffering from catastrophic cancellations. In that situation, it is better to reflect it to the other direction. A simple mechanism to avoid such things from happening is to apply following rule. Let $\vec{a}$ be a vector which want to reflect into $\pm e_1$ direction. Then:

- if the first entry of $\vec{a}$ is greater then zero, reflect to $-e_1$
- if the first entry of $\vec{a}$ is less then zero, reflect to $+e_1$

solving least-squares problems using householder When solving least squares problems using there is no need to explicitly constuct the Q matrix. Simply apply the transformations Q_1 to Q_n directly.

$$\begin{aligned} Q_n \cdots Q_2 Q_1 A x &= Q_n \cdots Q_2 Q_1 b \\ \begin{bmatrix} R \\ 0 \end{bmatrix} x &= \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{bmatrix} \end{aligned}$$

And after that, it should be clear what to do next.