CHAPTER 5

Column Spaces and QR

CONTENTS

ONE way to interpret the linear problem $A\vec{x} = \vec{b}$ for $\vec{x}$ is that we wish to write $\vec{b}$ as a linear combination of the columns of A with weights given in $\vec{x}$. This perspective does not change when we allow $A \in \mathbb{R}^{m\times n}$ to be non-square, but the solution may not exist or be unique depending on the structure of the column space of A. For these reasons, some techniques for factoring matrices and analyzing linear systems seek simpler representations of the column space of A to address questions regarding solvability and span more explicitly than row-based factorizations like LU.

5.1 THE STRUCTURE OF THE NORMAL EQUATIONS

As shown in §4.1.2, a necessary and sufficient condition for $\vec{x}$ to be a solution of the least-squares problem $A\vec{x} \approx \vec{b}$ is that $\vec{x}$ must satisfy the normal equations $(A^\top A)\vec{x} = A^\top \vec{b}$. This equation shows that least-squares problems can be solved using linear techniques on the matrix $A^\top A$. Methods like Cholesky factorization use the special structure of this matrix to the solver's advantage.

There is one large problem limiting the use of the normal equations, however. For now, suppose A is square; then we can write:

$$\begin{aligned}
\text{cond } A^\top A &= \|A^\top A\|\|(A^\top A)^{-1}\| \\
&\approx \|A^\top\|\|A\|\|A^{-1}\|\|(A^\top)^{-1}\| \text{ for many choices of } \|\cdot\| \\
&= \|A\|^2\|A^{-1}\|^2 \\
&= (\text{cond } A)^2.
\end{aligned}$$

That is, the condition number of $A^\top A$ is approximately the **square** of the condition number of A! Thus, while generic linear strategies might work on $A^\top A$ when the least-squares problem is "easy," when the columns of A are nearly linearly dependent these strategies are likely to exhibit considerable error since they do not deal with A directly.

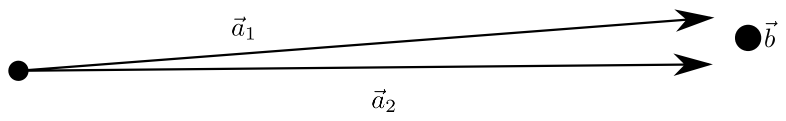

Figure 5.1 The vectors $\vec{a}_1$ and $\vec{a}_2$ nearly coincide; hence, writing $\vec{b}$ in the span of these vectors is difficult since $\vec{v}_1$ can be replaced with $\vec{v}_2$ or vice versa in a linear combination without incurring much error.

Intuitively, a primary reason that cond $A^\top A$ can be large is that columns of A might look "similar," as illustrated in Figure 5.1. Think of each column of A as a vector in $\mathbb{R}^m$. If two columns $\vec{a}_i$ and $\vec{a}_j$ satisfy $\vec{a}_i \approx \vec{a}_j$, then the least-squares residual length $\|\vec{b} - A\vec{x}\|_2$ will not suffer much if we replace multiples of $\vec{a}_i$ with multiples of $\vec{a}_j$ or vice versa. This wide range of nearly—but not completely—equivalent solutions yields poor conditioning. While the resulting vector $\vec{x}$ is unstable, however, the product $A\vec{x}$ remains nearly unchanged. If our goal is to write $\vec{b}$ in the column space of A, either approximate solution suffices. In other words, the backward error of multiple near-optimal $\vec{x}$'s is similar.

To solve such poorly conditioned problems, we will employ an alternative technique with closer attention to the column space of A rather than employing row operations as in Gaussian elimination. This strategy identifies and deals with such near-dependencies *explicitly*, bringing about greater numerical stability.

5.2 ORTHOGONALITY

We have identified why a least-squares problem might be difficult, but we might also ask when it is possible to perform least-squares without suffering from conditioning issues. If we can reduce a system to the straightforward case without inducing conditioning problems along the way, we will have found a stable way around the drawbacks explained in §5.1.

The easiest linear system to solve is $I_{n\times n}\vec{x} = \vec{b}$, where $I_{n\times n}$ is the $n \times n$ identity matrix: The solution is $\vec{x} = \vec{b}$! We are unlikely to bother using a linear solver to invert this particular linear system on purpose, but we may do so accidentally while solving least-squares. Even when $A \neq I_{n\times n}$—A may not even be square—we may, in particularly lucky circumstances, find that the Gram matrix $A^\top A$ satisfies $A^\top A = I_{n\times n}$, making least-squares trivial. To avoid confusion with the general case, we will use the variable Q to represent such a matrix satisfying $Q^\top Q = I_{n\times n}$.

While simply praying that $Q^\top Q = I_{n\times n}$ unlikely will yield a useful algorithm, we can examine this case to see how it becomes so favorable. Write the columns of Q as vectors $\vec{q}_1, \cdots, \vec{q}_n \in \mathbb{R}^m$. Then, the product $Q^\top Q$ has the following structure:

$$Q^\top Q = \begin{pmatrix} - & \vec{q}_1^\top & - \\ - & \vec{q}_2^\top & - \\ & \vdots & \\ - & \vec{q}_n^\top & \end{pmatrix} \begin{pmatrix} | & | & & | \\ \vec{q}_1 & \vec{q}_2 & \cdots & \vec{q}_n \\ | & | & & | \end{pmatrix} = \begin{pmatrix} \vec{q}_1 \cdot \vec{q}_1 & \vec{q}_1 \cdot \vec{q}_2 & \cdots & \vec{q}_1 \cdot \vec{q}_n \\ \vec{q}_2 \cdot \vec{q}_1 & \vec{q}_2 \cdot \vec{q}_2 & \cdots & \vec{q}_2 \cdot \vec{q}_n \\ \vdots & \vdots & \cdots & \vdots \\ \vec{q}_n \cdot \vec{q}_1 & \vec{q}_n \cdot \vec{q}_2 & \cdots & \vec{q}_n \cdot \vec{q}_n \end{pmatrix}.$$

Setting the expression on the right equal to $I_{n\times n}$ yields the following relationship:

$$\vec{q}_i \cdot \vec{q}_j = \begin{cases} 1 & \text{when } i = j \\ 0 & \text{when } i \neq j. \end{cases}$$

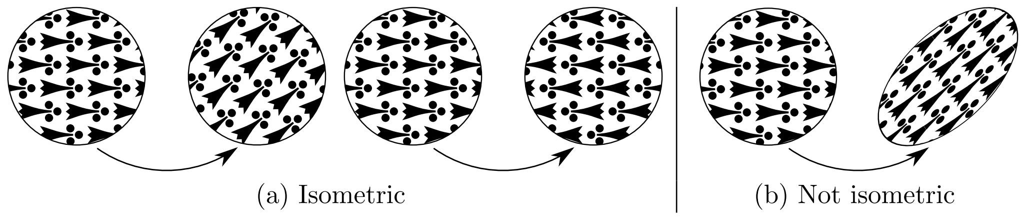

Figure 5.2 Isometries can (a) rotate and flip vectors but cannot (b) stretch or shear them.

In other words, the columns of Q are unit-length and orthogonal to one another. We say that they form an *orthonormal basis* for the column space of Q:

Definition 5.1 (Orthonormal; orthogonal matrix)**.** A set of vectors $\{\vec{v}_1, \cdots, \vec{v}_k\}$ is *orthonormal* if $\|\vec{v}_i\|_2 = 1$ for all i and $\vec{v}_i \cdot \vec{v}_j = 0$ for all $i \neq j$. A square matrix whose columns are orthonormal is called an *orthogonal* matrix.

The standard basis $\{\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n\}$ is an example of an orthonormal basis, and since the columns of the identity matrix $I_{n\times n}$ are these vectors, $I_{n\times n}$ is an orthogonal matrix.

We motivated our discussion by asking when we can expect $Q^\top Q = I_{n\times n}$. Now we know that this condition occurs exactly when the columns of Q are orthonormal. Furthermore, if Q is square and invertible with $Q^\top Q = I_{n\times n}$, then by multiplying both sides of the expression $Q^\top Q = I_{n\times n}$ by Q^{-1} shows $Q^{-1} = Q^\top$. Hence, $Q\vec{x} = \vec{b}$ is equivalent to $\vec{x} = Q^\top \vec{b}$ after multiplying both sides by the transpose $Q^\top$.

Orthonormality has a strong geometric interpretation. Recall from Chapter 1 that we can regard two orthogonal vectors $\vec{a}$ and $\vec{b}$ as being *perpendicular*. So, an orthonormal set of vectors is a set of mutually perpendicular unit vectors in $\mathbb{R}^n$. Furthermore, if Q is orthogonal, then its action does not affect the length of vectors:

$$\|Q\vec{x}\|_2^2 = \vec{x}^\top Q^\top Q\vec{x} = \vec{x}^\top I_{n\times n}\vec{x} = \vec{x}\cdot\vec{x} = \|\vec{x}\|_2^2.$$

Similarly, Q cannot affect the angle between two vectors, since:

$$(Q\vec{x})\cdot(Q\vec{y}) = \vec{x}^\top Q^\top Q\vec{y} = \vec{x}^\top I_{n\times n}\vec{y} = \vec{x}\cdot\vec{y}.$$

From this standpoint, if Q is orthogonal, then the operation $\vec{x} \mapsto Q\vec{x}$ is an *isometry* of $\mathbb{R}^n$, that is, it preserves lengths and angles. As illustrated in Figure 5.2, Q can rotate or reflect vectors but cannot scale or shear them. From a high level, the linear algebra of orthogonal matrices is easier because their actions do not affect the geometry of the underlying space.

5.3 STRATEGY FOR NON-ORTHOGONAL MATRICES

Most matrices A encountered when solving $A\vec{x} = \vec{b}$ or the least-squares problem $A\vec{x} \approx \vec{b}$ will not be orthogonal, so the machinery of §5.2 does not apply directly. For this reason, we must do some additional computations to connect the general case to the orthogonal one. To this end, we will derive an alternative to LU factorization using orthogonal rather than substitution matrices.

Take a matrix $A \in \mathbb{R}^{m\times n}$, and denote its column space as col A; col A is the span of the columns of A. Now, suppose a matrix $B \in \mathbb{R}^{n\times n}$ is invertible. We make the following observation about the column space of AB relative to that of A:

Proposition 5.1 (Column space invariance)**.** For any $A \in \mathbb{R}^{m\times n}$ and invertible $B \in \mathbb{R}^{n\times n}$, col $A =$ col AB.

Proof. Suppose $\vec{b} \in$ col A. By definition, there exists $\vec{x}$ with $A\vec{x} = \vec{b}$. If we take $\vec{y} = B^{-1}\vec{x}$, then $AB\vec{y} = (AB)\cdot(B^{-1}\vec{x}) = A\vec{x} = \vec{b}$, so $\vec{b} \in$ col AB. Conversely, take $\vec{c} \in$ col AB, so there exists $\vec{y}$ with $(AB)\vec{y} = \vec{c}$. In this case, $A\cdot(B\vec{y}) = \vec{c}$, showing that $\vec{c} \in$ col A. □

Recall the "elimination matrix" description of Gaussian elimination: We started with a matrix A and applied row operation matrices E_i such that the sequence $A, E_1A, E_2E_1A, \ldots$ eventually reduced to more easily solved triangular systems. The proposition above suggests an alternative strategy for situations like least-squares in which we care about the column space of A: Apply *column* operations to A by *post*-multiplication until the columns are orthonormal. So long as these operations are invertible, Proposition 5.1 shows that the column spaces of the modified matrices will be the same as the column space of A.

In the end, we will attempt to find a product $Q = AE_1E_2\cdots E_k$ starting from A and applying invertible operation matrices E_i such that Q is orthonormal. As we have argued above, the proposition shows that col $Q =$ col A. Inverting these operations yields a factorization $A = QR$ for $R = E_k^{-1}E_{k-1}^{-1}\cdots E_1^{-1}$. The columns of the matrix Q contain an orthonormal basis for the column space of A, and with careful design we can once again make R upper triangular.

When $A = QR$, by orthogonality of Q we have $A^\top A = R^\top Q^\top QR = R^\top R$. Making this substitution, the normal equations $A^\top A\vec{x} = A^\top\vec{b}$ imply $R^\top R\vec{x} = R^\top Q^\top\vec{b}$, or equivalently $R\vec{x} = Q^\top\vec{b}$. If we design R to be a triangular matrix, then solving the least-squares system $A^\top A\vec{x} = A^\top\vec{b}$ can be carried out efficiently by back-substitution via $R\vec{x} = Q^\top\vec{b}$.

5.4 GRAM-SCHMIDT ORTHOGONALIZATION

Our first algorithm for QR factorization follows naturally from our discussion above but may suffer from numerical issues. We use it here as an initial example of orthogonalization and then will improve upon it with better operations.

5.4.1 Projections

Suppose we have two vectors $\vec{a}$ and $\vec{b}$, with $\vec{a} \neq \vec{0}$. Then, we could easily ask, "Which multiple of $\vec{a}$ is closest to $\vec{b}$?" Mathematically, this task is equivalent to minimizing $\|c\vec{a} - \vec{b}\|_2^2$ over all possible $c \in \mathbb{R}$. If we think of $\vec{a}$ and $\vec{b}$ as $n\times 1$ matrices and c as a 1×1 matrix, then this is nothing more than an unconventional least-squares problem $\vec{a}\cdot c \approx \vec{b}$. In this formulation, the normal equations show $\vec{a}^\top\vec{a}\cdot c = \vec{a}^\top\vec{b}$, or

$$c = \frac{\vec{a}\cdot\vec{b}}{\vec{a}\cdot\vec{a}} = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|_2^2}.$$

We denote the resulting *projection* of $\vec{b}$ onto $\vec{a}$ as:

$$\text{proj}_{\vec{a}}\,\vec{b} \equiv c\vec{a} = \frac{\vec{a}\cdot\vec{b}}{\vec{a}\cdot\vec{a}}\vec{a} = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|_2^2}\vec{a}.$$

By design, $\text{proj}_{\vec{a}}\,\vec{b}$ is parallel to $\vec{a}$. What about the remainder $\vec{b} - \text{proj}_{\vec{a}}\vec{b}$? We can do the following computation to find out:

$$\vec{a}\cdot(\vec{b} - \text{proj}_{\vec{a}}\,\vec{b}) = \vec{a}\cdot\vec{b} - \vec{a}\cdot\left(\frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|_2^2}\vec{a}\right) \text{ by definition of } \text{proj}_{\vec{a}}\,\vec{b}$$

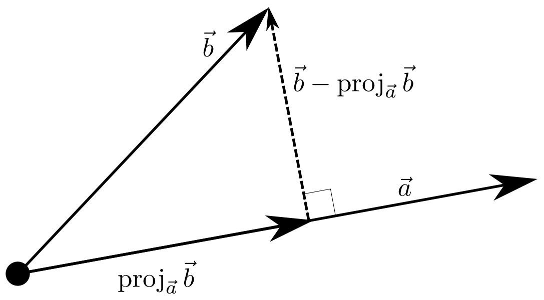
$\vec{b}$
$\vec{b} - \mathrm{proj}_{\vec{a}}\,\vec{b}$
$\vec{a}$
$\mathrm{proj}_{\vec{a}}\,\vec{b}$

Figure 5.3 The projection $\mathrm{proj}_{\vec{a}}\,\vec{b}$ is parallel to $\vec{a}$, while the remainder $\vec{b} - \mathrm{proj}_{\vec{a}}\,\vec{b}$ is perpendicular to $\vec{a}$.

$$\begin{aligned}
&= \vec{a}\cdot\vec{b} - \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|_2^2}(\vec{a}\cdot\vec{a}) \text{ by moving the constant outside the dot product}\\
&= \vec{a}\cdot\vec{b} - \vec{a}\cdot\vec{b} \text{ since } \vec{a}\cdot\vec{a} = \|\vec{a}\|_2^2\\
&= 0.
\end{aligned}$$

This simplification shows we have decomposed $\vec{b}$ into a component $\mathrm{proj}_{\vec{a}}\,\vec{b}$ parallel to $\vec{a}$ and another component $\vec{b} - \mathrm{proj}_{\vec{a}}\,\vec{b}$ orthogonal to $\vec{a}$, as illustrated in Figure 5.3.

Now, suppose that $\hat{a}_1, \hat{a}_2, \cdots, \hat{a}_k$ are orthonormal; for clarity, in this section we will put hats over vectors with unit length. For any single i, by the projection formula above

$$\mathrm{proj}_{\hat{a}_i}\,\vec{b} = (\hat{a}_i\cdot\vec{b})\hat{a}_i.$$

The denominator does not appear because $\|\hat{a}_i\|_2 = 1$ by definition. More generally, however, we can project $\vec{b}$ onto span $\{\hat{a}_1, \cdots, \hat{a}_k\}$ by minimizing the following energy function E over $c_1, \ldots, c_k \in \mathbb{R}$:

$$\begin{aligned}
E(c_1, c_2, \ldots, c_k) &\equiv \|c_1\hat{a}_1 + c_2\hat{a}_2 + \cdots + c_k\hat{a}_k - \vec{b}\|_2^2\\
&= \left(\sum_{i=1}^k\sum_{j=1}^k c_i c_j(\hat{a}_i\cdot\hat{a}_j)\right) - 2\vec{b}\cdot\left(\sum_{i=1}^k c_i\hat{a}_i\right) + \vec{b}\cdot\vec{b}\\
&\qquad \text{by applying and expanding } \|\vec{v}\|_2^2 = \vec{v}\cdot\vec{v}\\
&= \sum_{i=1}^k\left(c_i^2 - 2c_i\vec{b}\cdot\hat{a}_i\right) + \|\vec{b}\|_2^2 \text{ since the } \hat{a}_i\text{'s are orthonormal.}
\end{aligned}$$

The second step here is *only* valid because of orthonormality of the $\hat{a}_i$'s. At a minimum, the derivative of this energy with respect to c_i is zero for every i, yielding the relationship

$$0 = \frac{\partial E}{\partial c_i} = 2c_i - 2\vec{b}\cdot\hat{a}_i \implies c_i = \hat{a}_i\cdot\vec{b}.$$

This argument shows that when $\hat{a}_1, \cdots, \hat{a}_k$ are orthonormal,

$$\mathrm{proj}_{\mathrm{span}\,\{\hat{a}_1,\cdots,\hat{a}_k\}}\,\vec{b} = (\hat{a}_1\cdot\vec{b})\hat{a}_1 + \cdots + (\hat{a}_k\cdot\vec{b})\hat{a}_k.$$

This formula extends the formula for $\mathrm{proj}_{\vec{a}}\,\vec{b}$, and by a proof identical to the one above for single-vector projections, we must have

$$\hat{a}_i\cdot(\vec{b} - \mathrm{proj}_{\mathrm{span}\,\{\hat{a}_1,\cdots,\hat{a}_k\}}\,\vec{b}) = 0.$$

```
function Gram-Schmidt($\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_k$)
  ▷ Computes an orthonormal basis $\hat{a}_1, \ldots, \hat{a}_k$ for span $\{\vec{v}_1, \ldots, \vec{v}_k\}$
  ▷ Assumes $\vec{v}_1, \ldots, \vec{v}_k$ are linearly independent.
  $\hat{a}_1 \leftarrow \vec{v}_1/\|\vec{v}_1\|_2$                 ▷ Nothing to project out of the first vector
  for $i \leftarrow 2, 3, \ldots, k$
    $\vec{p} \leftarrow \vec{0}$                  ▷ Projection of $\vec{v}_i$ onto span $\{\hat{a}_1, \ldots, \hat{a}_{i-1}\}$
    for $j \leftarrow 1, 2, \ldots, i-1$
      $\vec{p} \leftarrow \vec{p} + (\vec{v}_i \cdot \hat{a}_j)\hat{a}_j$          ▷ Projecting onto orthonormal basis
    $\vec{r} \leftarrow \vec{v}_i - \vec{p}$               ▷ Residual is orthogonal to current basis
    $\hat{a}_i \leftarrow \vec{r}/\|\vec{r}\|_2$          ▷ Normalize this residual and add it to the basis
  return $\{\hat{a}_1, \ldots, \hat{a}_k\}$
```

Figure 5.4 The Gram-Schmidt algorithm for orthogonalization. This implementation assumes that the input vectors are linearly independent; in practice, linear dependence can be detected by checking for division by zero.

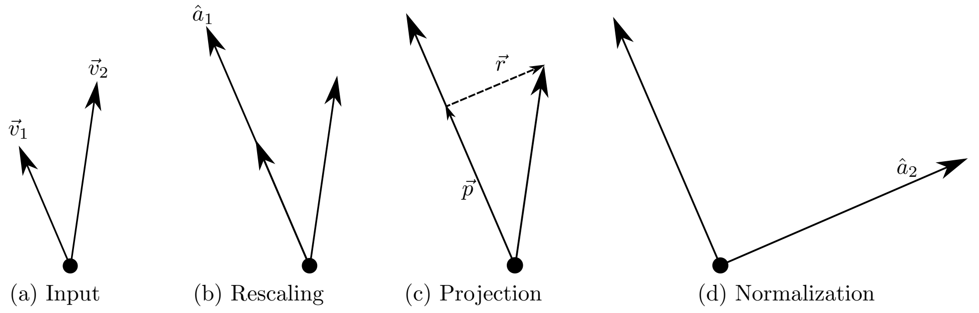

Figure 5.5 Steps of the Gram-Schmidt algorithm on (a) two vectors $\vec{v}_1$ and $\vec{v}_2$: (b) $\hat{a}_1$ is a rescaled version of $\vec{v}_1$; (c) $\vec{v}_2$ is decomposed into a parallel component $\vec{p}$ and a residual $\vec{r}$; (d) $\vec{r}$ is normalized to obtain $\hat{a}_2$.

Once again, we separated $\vec{b}$ into a component parallel to the span of the $\hat{a}_i$'s and a perpendicular residual.

5.4.2 Gram-Schmidt Algorithm

Our observations above lead to an algorithm for *orthogonalization*, or building an orthogonal basis $\{\hat{a}_1, \cdots, \hat{a}_k\}$ whose span is the same as that of a set of linearly independent but not necessarily orthogonal input vectors $\{\vec{v}_1, \cdots, \vec{v}_k\}$.

We add one vector at a time to the basis, starting with $\vec{v}_1$, then $\vec{v}_2$, and so on. When $\vec{v}_i$ is added to the current basis $\{\hat{a}_1, \ldots, \hat{a}_{i-1}\}$, we project out the span of $\hat{a}_1, \ldots, \hat{a}_{i-1}$. By the discussion in §5.4.1 the remaining residual must be orthogonal to the current basis, so we divide this residual by its norm to make it unit-length and add it to the basis. This technique, known as *Gram-Schmidt orthogonalization*, is detailed in Figure 5.4 and illustrated in Figure 5.5.

Example 5.1 (Gram-Schmidt orthogonalization)**.** Suppose we are given $\vec{v}_1 = (1, 0, 0)$, $\vec{v}_2 = (1, 1, 1)$, and $\vec{v}_3 = (1, 1, 0)$. The Gram-Schmidt algorithm proceeds as follows:

1. The first vector $\vec{v}_1$ is already unit-length, so we take $\hat{a}_1 = \vec{v}_1 = (1,0,0)$.
2. Now, we remove the span of $\hat{a}_1$ from the second vector $\vec{v}_2$:
$$\vec{v}_2 - \mathrm{proj}_{\hat{a}_1}\vec{v}_2 = \begin{pmatrix}1\\1\\1\end{pmatrix} - \left[\begin{pmatrix}1\\0\\0\end{pmatrix}\cdot\begin{pmatrix}1\\1\\1\end{pmatrix}\right]\begin{pmatrix}1\\0\\0\end{pmatrix} = \begin{pmatrix}0\\1\\1\end{pmatrix}.$$
Dividing this vector by its norm, we take $\hat{a}_2 = (0, 1/\sqrt{2}, 1/\sqrt{2})$.
3. Finally, we remove span $\{\hat{a}_1, \hat{a}_2\}$ from $\vec{v}_3$:
$$\begin{aligned}&\vec{v}_3 - \mathrm{proj}_{\mathrm{span}\,\{\hat{a}_1,\hat{a}_2\}}\vec{v}_3\\ &= \begin{pmatrix}1\\1\\0\end{pmatrix} - \left[\begin{pmatrix}1\\0\\0\end{pmatrix}\cdot\begin{pmatrix}1\\1\\0\end{pmatrix}\right]\begin{pmatrix}1\\0\\0\end{pmatrix} - \left[\begin{pmatrix}0\\1/\sqrt{2}\\1/\sqrt{2}\end{pmatrix}\cdot\begin{pmatrix}1\\1\\0\end{pmatrix}\right]\begin{pmatrix}0\\1/\sqrt{2}\\1/\sqrt{2}\end{pmatrix}\\ &= \begin{pmatrix}0\\1/2\\-1/2\end{pmatrix}.\end{aligned}$$
Normalizing this vector yields $\hat{a}_3 = (0, 1/\sqrt{2}, -1/\sqrt{2})$.

If we start with a matrix $A \in \mathbb{R}^{m\times n}$ whose columns are $\vec{v}_1, \cdots, \vec{v}_k$, then we can implement Gram-Schmidt using a series of column operations on A. Dividing column i of A by its norm is equivalent to post-multiplying A by a $k \times k$ diagonal matrix. The projection step for column i involves subtracting only multiples of columns j with $j < i$, and thus this operation can be implemented with an upper-triangular elimination matrix. Thus, our discussion in §5.3 applies, and we can use Gram-Schmidt to obtain a factorization $A = QR$. When the columns of A are linearly independent, one way to find R is as a product $R = Q^\top A$; a more stable approach is to keep track of operations as we did for Gaussian elimination.

Example 5.2 (QR factorization)**.** Suppose we construct a matrix whose columns are $\vec{v}_1$, $\vec{v}_2$, and $\vec{v}_3$ from Example 5.1:
$$A \equiv \begin{pmatrix}1&1&1\\0&1&1\\0&1&0\end{pmatrix}.$$
The output of Gram-Schmidt orthogonalization can be encoded in the matrix
$$Q \equiv \begin{pmatrix}1&0&0\\0&1/\sqrt{2}&1/\sqrt{2}\\0&1/\sqrt{2}&-1/\sqrt{2}\end{pmatrix}.$$
We can obtain the upper-triangular matrix R in the QR factorization two different ways. First, we can compute R after the fact using a product:
$$R = Q^\top A = \begin{pmatrix}1&0&0\\0&1/\sqrt{2}&1/\sqrt{2}\\0&1/\sqrt{2}&-1/\sqrt{2}\end{pmatrix}^\top \begin{pmatrix}1&1&1\\0&1&1\\0&1&0\end{pmatrix} = \begin{pmatrix}1&1&1\\0&\sqrt{2}&1/\sqrt{2}\\0&0&1/\sqrt{2}\end{pmatrix}.$$
As expected, R is upper triangular.

```
function MODIFIED-GRAM-SCHMIDT($\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_k$)
  ▷ Computes an orthonormal basis $\hat{a}_1, \ldots, \hat{a}_k$ for span $\{\vec{v}_1, \ldots, \vec{v}_k\}$
  ▷ Assumes $\vec{v}_1, \ldots, \vec{v}_k$ are linearly independent.

  for $i \leftarrow 1, 2, \ldots, k$
    $\hat{a}_i \leftarrow \vec{v}_i / \|\vec{v}_i\|_2$          ▷ Normalize the current vector and store in the basis
    for $j \leftarrow i+1, i+2, \ldots, k$
      $\vec{v}_j \leftarrow \vec{v}_j - (\vec{v}_j \cdot \hat{a}_i)\hat{a}_i$          ▷ Project $\hat{a}_i$ out of the remaining vectors
  return $\{\hat{a}_1, \ldots, \hat{a}_k\}$
```

Figure 5.6 The modified Gram-Schmidt algorithm.

We can also return to the steps of Gram-Schmidt orthogonalization to obtain R from the sequence of elimination matrices. A compact way to write the steps of Gram-Schmidt from Example 5.1 is as follows:

$$\text{Step 1: } Q_0 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\text{Step 2: } Q_1 = Q_0 E_1 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -1/\sqrt{2} & 0 \\ 0 & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1/\sqrt{2} & 1 \\ 0 & 1/\sqrt{2} & 0 \end{pmatrix}$$

$$\text{Step 3: } Q_2 = Q_1 E_2 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1/\sqrt{2} & 1 \\ 0 & 1/\sqrt{2} & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & -\sqrt{2} \\ 0 & 1 & -1 \\ 0 & 0 & \sqrt{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/\sqrt{2} & 1/\sqrt{2} \\ 0 & 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}.$$

These steps show $Q = AE_1E_2$, or equivalently $A = QE_2^{-1}E_1^{-1}$. This gives a second way to compute R:

$$R = E_2^{-1}E_1^{-1} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1/\sqrt{2} \\ 0 & 0 & 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & \sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & \sqrt{2} & 1/\sqrt{2} \\ 0 & 0 & 1/\sqrt{2} \end{pmatrix}.$$

The Gram-Schmidt algorithm is well known to be numerically unstable. There are many reasons for this instability that may or may not appear depending on the particular application. For instance, thanks to rounding and other issues, it might be the case that the $\hat{a}_i$'s are not completely orthogonal after the projection step. Our projection formula for finding $\vec{p}$ within the algorithm in Figure 5.4, however, only works when the $\hat{a}_i$'s are orthogonal. For this reason, in the presence of rounding, the projection $\vec{p}$ of $\vec{v}_i$ becomes less accurate.

One way around this issue is the "modified Gram-Schmidt" (MGS) algorithm in Figure 5.6, which has similar running time but makes a subtle change in the way projections are computed. Rather than computing the projection $\vec{p}$ in each iteration i onto span $\{\hat{a}_1, \ldots, \hat{a}_{i-1}\}$, as soon as $\hat{a}_i$ is computed it is projected out of $\vec{v}_{i+1}, \ldots, \vec{v}_k$; subsequently we never have to consider $\hat{a}_i$ again. This way, even if the basis globally is not completely orthogonal due to rounding, the projection step is valid since it only projects onto one $\hat{a}_i$ at a time. In the absence of rounding, modified Gram-Schmidt and classical Gram-Schmidt generate identical output.

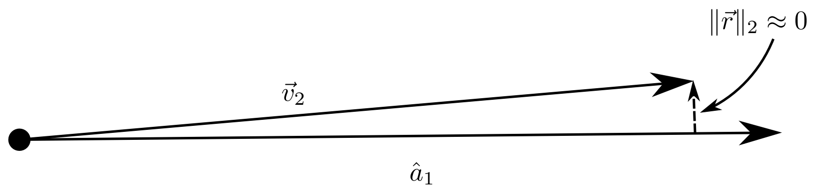

Figure 5.7 A failure mode of the basic and modified Gram-Schmidt algorithms; here $\hat{a}_1$ is nearly parallel to $\vec{v}_2$ and hence the residual $\vec{r}$ is vanishingly small.

A more subtle instability in the Gram-Schmidt algorithm is *not* resolved by MGS and can introduce serious numerical instabilities during the subtraction step. Suppose we provide the vectors $\vec{v}_1 = (1, 1)$ and $\vec{v}_2 = (1+\varepsilon, 1)$ as input to Gram-Schmidt for some $0 < \varepsilon \ll 1$. A reasonable basis for span $\{\vec{v}_1, \vec{v}_2\}$ might be $\{(1, 0), (0, 1)\}$. But, if we apply Gram-Schmidt, we obtain:

$$\begin{aligned}
\hat{a}_1 &= \frac{\vec{v}_1}{\|\vec{v}_1\|} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
\vec{p} &= \frac{2+\varepsilon}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
\vec{r} = \vec{v}_2 - \vec{p} &= \begin{pmatrix} 1+\varepsilon \\ 1 \end{pmatrix} - \frac{2+\varepsilon}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
&= \frac{1}{2}\begin{pmatrix} \varepsilon \\ -\varepsilon \end{pmatrix}.
\end{aligned}$$

Taking the norm, $\|\vec{v}_2 - \vec{p}\|_2 = (\sqrt{2}/2) \cdot \varepsilon$, so computing $\hat{a}_2 = (1/\sqrt{2}, -1/\sqrt{2})$ will require division by a scalar on the order of ε. Division by small numbers is an unstable numerical operation that generally should be avoided. A geometric interpretation of this case is shown in Figure 5.7.

5.5 HOUSEHOLDER TRANSFORMATIONS

In §5.3, we motivated the construction of QR factorization through the use of column operations. This construction is reasonable in the context of analyzing column spaces, but as we saw in our derivation of the Gram-Schmidt algorithm, the resulting numerical techniques can be unstable.

Rather than starting with A and post-multiplying by column operations to obtain $Q = AE_1 \cdots E_k$, however, we can also start with A and *pre*-multiply by orthogonal matrices Q_i to obtain $Q_k \cdots Q_1 A = R$. These Q's will act like row operations, eliminating elements of A until the resulting product R is upper triangular. Thanks to orthogonality of the Q's, we can write $A = (Q_1^\top \cdots Q_k^\top)R$, obtaining the QR factorization since products and transposes of orthogonal matrices are orthogonal.

The row operation matrices we used in Gaussian elimination and LU will not suffice for QR factorization since they are not orthogonal. Several alternatives have been suggested; we will introduce a common orthogonal row operation introduced in 1958 by Alston Scott Householder [65].

The space of orthogonal $n \times n$ matrices is very large, so we seek a smaller set of possible Q_i's that is easier to work with while still powerful enough to implement elimination

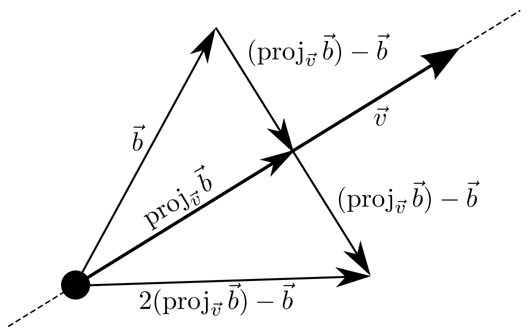
$(\mathrm{proj}_{\vec v}\,\vec b) - \vec b$
$\vec v$
$\vec b$
$\mathrm{proj}_{\vec v}\,\vec b$
$(\mathrm{proj}_{\vec v}\,\vec b) - \vec b$
$2(\mathrm{proj}_{\vec v}\,\vec b) - \vec b$

Figure 5.8 Reflecting $\vec b$ over $\vec v$.

operations. To develop some intuition, from our geometric discussions in §5.2 we know that orthogonal matrices must preserve angles and lengths, so intuitively they only can rotate and reflect vectors. Householder proposed using only reflection operations to reduce A to upper-triangular form. A well-known alternative by Givens uses only rotations to accomplish the same task [48] and is explored in Exercise 5.11.

One way to write an orthogonal reflection matrix is in terms of projections, as illustrated in Figure 5.8. Suppose we have a vector $\vec b$ that we wish to reflect over a vector $\vec v$. We have shown that the residual $\vec r \equiv \vec b - \mathrm{proj}_{\vec v}\,\vec b$ is perpendicular to $\vec v$. Following the reverse of this direction twice shows that the difference $2\mathrm{proj}_{\vec v}\,\vec b - \vec b$ reflects $\vec b$ *over* $\vec v$.

We can expand our reflection formula as follows:

$$\begin{aligned}
2\mathrm{proj}_{\vec v}\,\vec b - \vec b &= 2\frac{\vec v \cdot \vec b}{\vec v \cdot \vec v}\vec v - \vec b \text{ by definition of projection}\\
&= 2\vec v \cdot \frac{\vec v^\top \vec b}{\vec v^\top \vec v} - \vec b \text{ using matrix notation}\\
&= \left(\frac{2\vec v \vec v^\top}{\vec v^\top \vec v} - I_{n\times n}\right)\vec b\\
&\equiv -H_{\vec v}\vec b, \text{ where we define } H_{\vec v} \equiv I_{n\times n} - \frac{2\vec v\vec v^\top}{\vec v^\top \vec v}.
\end{aligned}$$

By this factorization, we can think of reflecting $\vec b$ over $\vec v$ as applying a matrix $-H_{\vec v}$ to $\vec b$; $-H_{\vec v}$ has no dependence on $\vec b$. $H_{\vec v}$ without the negative is still orthogonal, and by convention we will use it from now on. Our derivation will parallel that in [58].

Like in forward-substitution, in our first step we wish to pre-multiply A by a matrix that takes the first column of A, which we will denote $\vec a$, to some multiple of the first identity vector $\vec e_1$. Using reflections rather than forward-substitutions, however, we now need to find some $\vec v, c$ such that $H_{\vec v}\vec a = c\vec e_1$. Expanding this relationship,

$$\begin{aligned}
c\vec e_1 &= H_{\vec v}\vec a, \text{ as explained above}\\
&= \left(I_{n\times n} - \frac{2\vec v\vec v^\top}{\vec v^\top \vec v}\right)\vec a, \text{ by definition of } H_{\vec v}\\
&= \vec a - 2\vec v\frac{\vec v^\top \vec a}{\vec v^\top \vec v}.
\end{aligned}$$

Moving terms around shows

$$\vec v = (\vec a - c\vec e_1)\cdot \frac{\vec v^\top \vec v}{2\vec v^\top \vec a}.$$

In other words, if $H_{\vec{v}}$ accomplishes the desired reflection, then $\vec{v}$ must be parallel to the difference $\vec{a} - c\vec{e}_1$. Scaling $\vec{v}$ does not affect the formula for $H_{\vec{v}}$, so for now assuming such an $H_{\vec{v}}$ exists we can attempt to *choose* $\vec{v} = \vec{a} - c\vec{e}_1$.

If this choice is valid, then substituting $\vec{v} = \vec{a} - c\vec{e}_1$ into the simplified expression shows

$$\vec{v} = \vec{v} \cdot \frac{\vec{v}^\top \vec{v}}{2\vec{v}^\top \vec{a}}.$$

Assuming $\vec{v} \neq \vec{0}$, the coefficient next to $\vec{v}$ on the right-hand side must be 1, showing:

$$\begin{aligned} 1 &= \frac{\vec{v}^\top \vec{v}}{2\vec{v}^\top \vec{a}} \\ &= \frac{\|\vec{a}\|_2^2 - 2c\vec{e}_1 \cdot \vec{a} + c^2}{2(\vec{a}\cdot\vec{a} - c\vec{e}_1 \cdot \vec{a})} \\ \text{or, } 0 &= \|\vec{a}\|_2^2 - c^2 \implies c = \pm\|\vec{a}\|_2. \end{aligned}$$

After choosing $c = \pm\|\vec{a}\|_2$, our steps above are all reversible. We originally set out to find $\vec{v}$ such that $H_{\vec{v}}\vec{a} = c\vec{e}_1$. By taking $\vec{v} = \vec{a} - c\vec{e}_1$ with $c = \pm\|\vec{a}\|_2$, the steps above show:

$$H_{\vec{v}}A = \begin{pmatrix} c & \times & \times & \times \\ 0 & \times & \times & \times \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \times & \times & \times \end{pmatrix}.$$

We have just accomplished a step similar to forward elimination using orthogonal matrices!

Example 5.3 (Householder transformation)**.** Suppose

$$A = \begin{pmatrix} 2 & -1 & 5 \\ 2 & 1 & 2 \\ 1 & 0 & -2 \end{pmatrix}.$$

The first column of A has norm $\sqrt{2^2+2^2+1^2} = 3$, so if we take $c = 3$ we can write:

$$\vec{v} = \vec{a} - c\vec{e}_1 = \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix} - 3\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.$$

This choice of $\vec{v}$ gives elimination matrix

$$H_{\vec{v}} = I_{3\times 3} - \frac{2\vec{v}\vec{v}^\top}{\vec{v}^\top\vec{v}} = \frac{1}{3}\begin{pmatrix} 2 & 2 & 1 \\ 2 & -1 & -2 \\ 1 & -2 & 2 \end{pmatrix}.$$

As expected, $H_{\vec{v}}^\top H_{\vec{v}} = I_{3\times 3}$. Furthermore, $H_{\vec{v}}$ eliminates the first column of A:

$$H_{\vec{v}}A = \frac{1}{3}\begin{pmatrix} 2 & 2 & 1 \\ 2 & -1 & -2 \\ 1 & -2 & 2 \end{pmatrix}\begin{pmatrix} 2 & -1 & 5 \\ 2 & 1 & 2 \\ 1 & 0 & -2 \end{pmatrix} = \begin{pmatrix} 3 & 0 & 4 \\ 0 & -1 & 4 \\ 0 & -1 & -1 \end{pmatrix}.$$

To fully reduce A to upper-triangular form, we must repeat the steps above to eliminate all elements of A below the diagonal. During the k-th step of triangularization, we can take

```
function HOUSEHOLDER-QR($A$)
  ▷ Factors $A \in \mathbb{R}^{m\times n}$ as $A = QR$.
  ▷ $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R \in \mathbb{R}^{m\times n}$ is upper triangular

  $Q \leftarrow I_{m\times m}$
  $R \leftarrow A$
  for $k \leftarrow 1, 2, \ldots, m$
    $\vec{a} \leftarrow R\vec{e}_k$                    ▷ Isolate column $k$ of $R$ and store it in $\vec{a}$
    $(\vec{a}_1, \vec{a}_2) \leftarrow$ SPLIT($\vec{a}$,$k-1$)       ▷ Separate off the first $k-1$ elements of $\vec{a}$

    $c \leftarrow \|\vec{a}_2\|_2$          ▷ Find reflection vector $\vec{v}$ for the Householder matrix $H_{\vec{v}}$
    $\vec{v} \leftarrow \begin{pmatrix} \vec{0} \\ \vec{a}_2 \end{pmatrix} - c\vec{e}_k$

    $R \leftarrow H_{\vec{v}}R$         ▷ Eliminate elements below the diagonal of the $k$-th column
    $Q \leftarrow QH_{\vec{v}}^\top$
  return $Q, R$
```

Figure 5.9 Householder QR factorization; the products with $H_{\vec{v}}$ can be carried out in quadratic time after expanding the formula for $H_{\vec{v}}$ in terms of $\vec{v}$ (see Exercise 5.2).

$\vec{a}$ to be the k-th column of $Q_{k-1}Q_{k-2}\cdots Q_1A$, where the Q_i's are reflection matrices like the one derived above. We can split $\vec{a}$ into two components:

$$\vec{a} = \begin{pmatrix} \vec{a}_1 \\ \vec{a}_2 \end{pmatrix}.$$

Here, $\vec{a}_1 \in \mathbb{R}^{k-1}$ and $\vec{a}_2 \in \mathbb{R}^{m-k+1}$. We wish to find $\vec{v}$ such that

$$H_{\vec{v}}\vec{a} = \begin{pmatrix} \vec{a}_1 \\ c \\ \vec{0} \end{pmatrix}.$$

Following a parallel derivation to the one above for the case $k = 1$ shows that

$$\vec{v} = \begin{pmatrix} \vec{0} \\ \vec{a}_2 \end{pmatrix} - c\vec{e}_k$$

accomplishes exactly this transformation when $c = \pm\|\vec{a}_2\|_2$.

The algorithm for Householder QR, illustrated in Figure 5.9, applies these formulas iteratively, reducing to triangular form in a manner similar to Gaussian elimination. For each column of A, we compute $\vec{v}$ annihilating the bottom elements of the column and apply $H_{\vec{v}}$ to A. The end result is an upper-triangular matrix $R = H_{\vec{v}_n}\cdots H_{\vec{v}_1}A$. Q is given by the product $H_{\vec{v}_1}^\top \cdots H_{\vec{v}_n}^\top$. When $m < n$, it may be preferable to store Q implicitly as a list of vectors $\vec{v}$, which fits in the lower triangle that otherwise would be empty in R.

Example 5.4 (Householder QR)**.** Continuing Example 5.3, we split the second column of $H_{\vec{v}}A$ as $\vec{a}_1 = (0) \in \mathbb{R}^1$ and $\vec{a}_2 = (-1, -1) \in \mathbb{R}^2$. We now take $c' = -\|\vec{a}_2\|_2 = -\sqrt{2}$, yielding

$$\vec{v}' = \begin{pmatrix} \vec{0} \\ \vec{a}_2 \end{pmatrix} - c'\vec{e}_2 = \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix} + \sqrt{2}\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -1+\sqrt{2} \\ -1 \end{pmatrix}$$

$$\implies H_{\vec{v}'} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/\sqrt{2} & 1/\sqrt{2} \\ 0 & 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}.$$

Applying the two Householder steps reveals an upper-triangular matrix:

$$R = H_{\vec{v}'} H_{\vec{v}} A = \begin{pmatrix} 3 & 0 & 4 \\ 0 & -\sqrt{2} & 3/\sqrt{2} \\ 0 & 0 & 5/\sqrt{2} \end{pmatrix}.$$

The corresponding Q is given by $Q = H_{\vec{v}'}^\top H_{\vec{v}}^\top$.

5.6 REDUCED QR FACTORIZATION

We conclude our discussion by returning to the least-squares problem $A\vec{x} \approx \vec{b}$ when $A \in \mathbb{R}^{m\times n}$ is not square. Both algorithms we have discussed in this chapter can factor non-square matrices A into products QR, but the sizes of Q and R are different depending on the approach:

- When applying Gram-Schmidt, we do column operations on A to obtain Q by orthogonalization. For this reason, the dimension of A is that of Q, yielding $Q \in \mathbb{R}^{m\times n}$ and $R \in \mathbb{R}^{n\times n}$ as a product of elimination matrices.
- When using Householder reflections, we obtain Q as the product of $m \times m$ reflection matrices, leaving $R \in \mathbb{R}^{m\times n}$.

Suppose we are in the typical case for least-squares, for which $m \gg n$. We still prefer to use the Householder method due to its numerical stability, but now the $m \times m$ matrix Q might be too large to store. To save space, we can use the upper-triangular structure of R to our advantage. For instance, consider the structure of a 5×3 matrix R:

$$R = \left(\begin{array}{ccc} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right).$$

Anything below the upper $n \times n$ square of R must be zero, yielding a simplification:

$$A = QR = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_1 \\ 0 \end{pmatrix} = Q_1 R_1.$$

Here, $Q_1 \in \mathbb{R}^{m\times n}$ and $R_1 \in \mathbb{R}^{n\times n}$ still contains the upper triangle of R. This is called the "reduced" QR factorization of A, since the columns of Q_1 contain a basis for the column space of A rather than for all of $\mathbb{R}^m$; it takes up far less space. The discussion in §5.3 still applies, so the reduced QR factorization can be used for least-squares in a similar fashion.

5.7 EXERCISES

5.1 Use Householder reflections to obtain a QR factorization of the matrix A from Example 5.2. Do you obtain the same QR factorization as the Gram-Schmidt approach?

5.2 Suppose $A \in \mathbb{R}^{n\times n}$ and $\vec{v} \in \mathbb{R}^n$. Provide pseudocode for computing the product $H_{\vec{v}}A$ in $O(n^2)$ time. Explain where this method might be used in implementations of Householder QR factorization.

5.3 (Adapted from Stanford CS 205A, 2012) Suppose $A \in \mathbb{R}^{m\times n}$ is factored $A = QR$. Show that $P_0 = I_{m\times m} - QQ^\top$ is the projection matrix onto the null space of $A^\top$.

5.4 (Adapted from Stanford CS 205A, 2012) Suppose we consider $\vec{a} \in \mathbb{R}^n$ as an $n \times 1$ matrix. Write out its "reduced" QR factorization explicitly.

5.5 Show that the Householder matrix $H_{\vec{v}}$ is *involutary*, meaning $H_{\vec{v}}^2 = I_{n\times n}$. What are the eigenvalues of $H_{\vec{v}}$?

5.6 Propose a method for finding the least-norm projection of a vector $\vec{v}$ onto the column space of $A \in \mathbb{R}^{m\times n}$ with $m > n$.

5.7 Alternatives to the QR factorization:

(a) Can a matrix $A \in \mathbb{R}^{m\times n}$ be factored into $A = RQ$ where R is upper triangular and Q is orthogonal? How?

(b) Can a matrix $A \in \mathbb{R}^{m\times n}$ be factored into $A = QL$ where L is lower triangular?

5.8 Relating QR and Cholesky factorizations:

(a) Take $A \in \mathbb{R}^{m\times n}$ and suppose we apply the Cholesky factorization to obtain $A^\top A = LL^\top$. Define $Q \equiv A(L^\top)^{-1}$. Show that the columns of Q are orthogonal.

(b) Based on the previous part, suggest a relationship between the Cholesky factorization of $A^\top A$ and QR factorization of A.

5.9 Suppose $A \in \mathbb{R}^{m\times n}$ is rank m with $m < n$. Suppose we factor

$$A^\top = Q\begin{pmatrix} R_1 \\ 0 \end{pmatrix}.$$

Provide a solution $\vec{x}$ to the underdetermined system $A\vec{x} = \vec{b}$ in terms of Q and R_1.

Hint: Try the square case $A \in \mathbb{R}^{n\times n}$ first, and use the result to guess a form for $\vec{x}$. Be careful that you multiply matrices of proper size.

5.10 ("Generalized QR," [2]) One way to generalize the QR factorization of a matrix is to consider the possibility of factorizing multiple matrices *simultaneously.*

(a) Suppose $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{n\times p}$, with $m \le n \le p$. Show that there are orthogonal matrices $Q \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{p\times p}$ as well as a matrix $R \in \mathbb{R}^{n\times m}$ such that the following conditions hold:

- $Q^\top A = R$.
- $Q^\top BV = S$, where S can be written

$$S = \begin{pmatrix} 0 & \bar{S} \end{pmatrix},$$

for upper-triangular $\bar{S} \in \mathbb{R}^{n\times n}$.

- R can be written
$$R = \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix},$$
for upper-triangular $\bar{R} \in \mathbb{R}^{m\times m}$.

Hint: Take $\bar{R}$ to be R_1 from the reduced QR factorization of A. Apply RQ factorization to $Q^\top B$; see Exercise 5.7a.

(b) Show how to solve the following optimization problem for $\vec{x}$ and $\vec{u}$ using the generalized QR factorization:
$$\begin{array}{rl} \min_{\vec{x},\vec{u}} & \|\vec{u}\|_2 \\ \text{subject to} & A\vec{x} + B\vec{u} = \vec{c}. \end{array}$$
You can assume $\bar{S}$ and $\bar{R}$ are invertible.

5.11 An alternative algorithm for QR factorization uses *Givens rotations* rather than Householder reflections.

(a) The 2×2 rotation matrix by angle $\theta \in [0, 2\pi)$ is given by
$$R_\theta \equiv \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$
Show that for a given $\vec{x} \in \mathbb{R}^2$, a θ always exists such that $R_\theta \vec{x} = r\vec{e}_1$, where $r \in \mathbb{R}$ and $\vec{e}_1 = (1, 0)$. Give formulas for $\cos\theta$ and $\sin\theta$ that do not require trigonometric functions.

(b) The *Givens rotation matrix* of rows i and j about angle θ is given by
$$G(i,j,\theta) \equiv \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix},$$
where $c \equiv \cos\theta$ and $s \equiv \sin\theta$. In this formula, the c's appear in positions (i,i) and (j,j) while the s's appear in positions (i,j) and (j,i). Provide an $O(n)$ method for finding the product $G(i,j,\theta)A$ for $A \in \mathbb{R}^{n\times n}$; the matrix A can be modified in the process.

(c) Give an $O(n^3)$ time algorithm for overwriting $A \in \mathbb{R}^{n\times n}$ with $Q^\top A = R$, where $Q \in \mathbb{R}^{n\times n}$ is orthogonal and $R \in \mathbb{R}^{n\times n}$ is upper triangular. You do not need to store Q.

(d) Suggest how you might store Q implicitly if you use the QR method you developed in the previous part.

(e) Suggest an $O(n^3)$ method for recovering the matrix Q given A and R.

5.12 (Adapted from [50], §5.1) If $\vec{x}, \vec{y} \in \mathbb{R}^m$ with $\|\vec{x}\|_2 = \|\vec{y}\|_2$, write an algorithm for finding an orthogonal matrix Q such that $Q\vec{x} = \vec{y}$.

5.13 ("TSQR," [28]) The QR factorization algorithms we considered can be challenging to extend to parallel architectures like MapReduce. Here, we consider QR factorization of $A \in \mathbb{R}^{m \times n}$ where $m \gg n$.

(a) Suppose $A \in \mathbb{R}^{8n \times n}$. Show how to factor $A = Q\bar{R}$, where $Q \in \mathbb{R}^{8n \times 4n}$ has orthogonal columns and $\bar{R} \in \mathbb{R}^{4n \times n}$ contains four $n \times n$ upper-triangular blocks. *Hint:* Write

$$A = \begin{pmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{pmatrix}.$$

(b) Recursively apply your answer from 5.13a to generate a QR factorization of A.

(c) Suppose we make the following factorizations:

$$\begin{aligned} A_1 &= Q_1 R_1 \\ \begin{pmatrix} R_1 \\ A_2 \end{pmatrix} &= Q_2 R_2 \\ \begin{pmatrix} R_2 \\ A_3 \end{pmatrix} &= Q_3 R_3 \\ \begin{pmatrix} R_3 \\ A_4 \end{pmatrix} &= Q_4 R_4, \end{aligned}$$

where each of the R_i's are square. Use these matrices to factor $A = QR$.

(d) Suppose we read A row-by-row. Why might the simplification in 5.13c be useful for QR factorization of A in this case? You can assume we only need R from the QR factorization.