Example 68. Using Gram–Schmidt, find an orthogonal basis for $W = \operatorname{span}\left\{\begin{bmatrix}0\\3\\0\\0\end{bmatrix}, \begin{bmatrix}2\\1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\1\\1\\1\end{bmatrix}\right\}$.

Solution. We begin with the (not orthogonal) basis $\boldsymbol{w}_1 = \begin{bmatrix}0\\3\\0\\0\end{bmatrix}$, $\boldsymbol{w}_2 = \begin{bmatrix}2\\1\\0\\0\end{bmatrix}$, $\boldsymbol{w}_3 = \begin{bmatrix}1\\1\\1\\1\end{bmatrix}$.

We then construct an orthogonal basis $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3$:

- $\boldsymbol{q}_1 = \boldsymbol{w}_1 = \begin{bmatrix}0\\3\\0\\0\end{bmatrix}$
- $\boldsymbol{q}_2 = \boldsymbol{w}_2 - \begin{pmatrix}\text{projection of}\\ \boldsymbol{w}_2 \text{ onto } \boldsymbol{q}_1\end{pmatrix} = \begin{bmatrix}2\\1\\0\\0\end{bmatrix} - \frac{3}{9}\begin{bmatrix}0\\3\\0\\0\end{bmatrix} = \begin{bmatrix}2\\0\\0\\0\end{bmatrix}$
- $\boldsymbol{q}_3 = \boldsymbol{w}_3 - \begin{pmatrix}\text{projection of } \boldsymbol{w}_3\\ \text{onto } \operatorname{span}\{\boldsymbol{q}_1, \boldsymbol{q}_2\}\end{pmatrix} = \boldsymbol{w}_3 - \begin{pmatrix}\text{projection of}\\ \boldsymbol{w}_3 \text{ onto } \boldsymbol{q}_1\end{pmatrix} - \begin{pmatrix}\text{projection of}\\ \boldsymbol{w}_3 \text{ onto } \boldsymbol{q}_2\end{pmatrix}$
  $= \begin{bmatrix}1\\1\\1\\1\end{bmatrix} - \frac{3}{9}\begin{bmatrix}0\\3\\0\\0\end{bmatrix} - \frac{2}{4}\begin{bmatrix}2\\0\\0\\0\end{bmatrix} = \begin{bmatrix}0\\0\\1\\1\end{bmatrix}$
  Make sure you understand how $\boldsymbol{q}_3$ was designed to be orthogonal to both $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$!
  Also note that breaking up the projection onto $\operatorname{span}\{\boldsymbol{q}_1, \boldsymbol{q}_2\}$ into the projections onto $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is only possible because $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are orthogonal.

Hence, $\begin{bmatrix}0\\3\\0\\0\end{bmatrix}, \begin{bmatrix}2\\0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\\1\end{bmatrix}$ is an orthogonal basis of W.

Important. Normalizing, we obtain an orthonormal basis: $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, \frac{1}{\sqrt{2}}\begin{bmatrix}0\\0\\1\\1\end{bmatrix}$.

Example 69. Determine the QR decomposition of $A = \begin{bmatrix}0&2&1\\3&1&1\\0&0&1\\0&0&1\end{bmatrix}$.

Solution. The first step is Gram–Schmidt orthonormalization on the columns of A. We then use the resulting orthonormal vectors as the columns of Q.

We already did Gram–Schmidt in Example 68: from that work, we have $Q = \begin{bmatrix}0&1&0&0\\1&0&0&0\\0&0&1/\sqrt{2}&1/\sqrt{2}\\0&0&1/\sqrt{2}&1/\sqrt{2}\end{bmatrix}$.

Hence, $R = Q^T A = \begin{bmatrix}0&1&0&0\\1&0&0&0\\0&0&1/\sqrt{2}&1/\sqrt{2}\end{bmatrix}\begin{bmatrix}0&2&1\\3&1&1\\0&0&1\\0&0&1\end{bmatrix} = \begin{bmatrix}3&1&1\\0&2&1\\0&0&\sqrt{2}\end{bmatrix}$.

Comment. As commented earlier, the entries of R have actually all been computed during Gram–Schmidt, so that, if we pay attention, we could immediately write down R (no extra work required). Looking back at Example 68, can you see this?

Letting Sage do the work for us.

```
Sage] A = matrix(QQbar, [[0,2,1],[3,1,1],[0,0,1],[0,0,1]])
```

```
Sage] A.QR(full=false)
```

$$\left(\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0.7071067811865475\,? \\ 0 & 0 & 0.7071067811865475\,? \end{bmatrix}, \begin{bmatrix} 3 & 1 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1.414213562373095\,? \end{bmatrix}\right)$$

Comment. Can you figure out what happens if you omit the `full=false`? Check out the comment under **Variations** for the QR decomposition in the previous lecture sketch. On the other hand, the `QQbar` is telling Sage to compute with algebraic numbers (instead of just rational numbers); if omitted, it would complain that square roots are not available

Example 70. (extra) Determine the QR decomposition of $A = \begin{bmatrix} 1 & 2 & 4 \\ 0 & 0 & -5 \\ 0 & 3 & 6 \end{bmatrix}$.

Solution. We first apply Gram–Schmidt orthonormalization to the columns of A. For a variation, like a computer, we normalize after each step (rather than normalize at the end):

- $\boldsymbol{b}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, so that $\boldsymbol{q}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$.
- $\boldsymbol{b}_2 = \begin{bmatrix} 2 \\ 0 \\ 3 \end{bmatrix} - \left(\begin{bmatrix} 2 \\ 0 \\ 3 \end{bmatrix} \cdot \boldsymbol{q}_1\right)\boldsymbol{q}_1 = \begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}$, so that $\boldsymbol{q}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$.
- $\boldsymbol{b}_3 = \begin{bmatrix} 4 \\ -5 \\ 6 \end{bmatrix} - \left(\begin{bmatrix} 4 \\ -5 \\ 6 \end{bmatrix} \cdot \boldsymbol{q}_1\right)\boldsymbol{q}_1 - \left(\begin{bmatrix} 4 \\ -5 \\ 6 \end{bmatrix} \cdot \boldsymbol{q}_2\right)\boldsymbol{q}_2 = \begin{bmatrix} 0 \\ -5 \\ 0 \end{bmatrix}$, so that $\boldsymbol{q}_3 = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$.

Therefore, $Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$. Finally, $R = Q^T A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}\begin{bmatrix} 1 & 2 & 4 \\ 0 & 0 & -5 \\ 0 & 3 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 4 \\ 0 & 3 & 6 \\ 0 & 0 & 5 \end{bmatrix}$.

In conclusion, we have found the QR decomposition: $\underbrace{\begin{bmatrix} 1 & 2 & 4 \\ 0 & 0 & -5 \\ 0 & 3 & 6 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}}_{Q}\underbrace{\begin{bmatrix} 1 & 2 & 4 \\ 0 & 3 & 6 \\ 0 & 0 & 5 \end{bmatrix}}_{R}$

Comment. As noted before, we actually could write down R without any additional computation. Indeed, realize that the second column of R, that is $[2,3,0]^T$ means that

$$\text{2nd col of } A = 2\boldsymbol{q}_1 + 3\boldsymbol{q}_2.$$

Which we already knew from our computation of $\boldsymbol{q}_2$! Also, by construction, we know that the second column of A is a linear combination of $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ only, and that $\boldsymbol{q}_3$ enters the story later on. This corresponds to the fact that R is always upper triangular.

Letting Sage do the work for us.

```
Sage] A = matrix(QQbar, [[1,2,4], [0,0,-5], [0,3,6]])
```

```
Sage] A.QR()
```

$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 4 \\ 0 & 3 & 6 \\ 0 & 0 & 5 \end{bmatrix}\right)$$

Comment. The `QQbar` is telling Sage to compute with algebraic numbers (instead of just rational numbers); in general, if omitted, it would complain that square roots are not available (because the matrices Q and R typically involve square roots). Here, we are lucky that square roots didn't creep in.

Example 71. (extra) Find the QR decomposition of $A = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$.

Solution. (final answer only) $A = QR$ with $Q = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \end{bmatrix}$ and $R = \begin{bmatrix} \sqrt{2} & \frac{1}{\sqrt{2}} & \sqrt{2} \\ 0 & \frac{1}{\sqrt{2}} & \sqrt{2} \\ 0 & 0 & 1 \end{bmatrix}$.

Example 72. One practical application of the QR decomposition is solving systems of linear equations.

$$\begin{aligned} A\boldsymbol{x} = \boldsymbol{b} \quad &\iff \quad QR\boldsymbol{x} = \boldsymbol{b} \qquad \text{(now, multiply with } Q^T \text{ from the left)} \\ &\implies \quad R\boldsymbol{x} = Q^T\boldsymbol{b} \end{aligned}$$

The last system is triangular and can be solved by back substitution.

A couple of comments are in order:

- If A is $n \times n$ and invertible, then the "$\implies$" is actually a "$\iff$".
- The equation $R\boldsymbol{x} = Q^T\boldsymbol{b}$ is always consistent! (Recall that R is invertible.)
  Indeed, if A is not $n \times n$ or not invertible, then $R\boldsymbol{x} = Q^T\boldsymbol{b}$ gives the least squares solutions!
  Why? $A^TA\hat{\boldsymbol{x}} = A^T\boldsymbol{b} \iff \underbrace{(QR)^TQR}_{=R^TQ^TQR}\hat{\boldsymbol{x}} = (QR)^T\boldsymbol{b} \iff R^TR\hat{\boldsymbol{x}} = R^TQ^T\boldsymbol{b} \iff R\hat{\boldsymbol{x}} = Q^T\boldsymbol{b}$
  [For the last step we need that R is invertible, which is always the case when A is $m \times n$ of rank n.]
- So, how does the QR way of solving linear systems compare to our beloved Gaussian elimination (LU)?
  It turns out that QR is a little slower than LU but makes up for it in "numerical stability".
  What does that mean? When computing numerically, we use floating point arithmetic and approximate each number by an expression of the form $0.1234 \cdot 10^{-16}$. A certain (fixed) number of bits is used to store the part 0.1234 (here, 4 decimal places of accuracy) as well as the exponent -16.
  Now, here is something terrible that can happen in numerical computations: mathematically, the quantities x and $(x+1)-1$ are exactly the same. However, numerically, they might not. Take, for instance, $x = 0.1234 \cdot 10^{-6}$. Then, to an accuracy of 4 decimal places, $x+1 = 0.1000 \cdot 10^{1}$, so that $(x+1)-1 = 0.0000$. But $x \neq 0$. We completely lost all the information about x.
  To be numerically stable, an algorithm must avoid issues like that.

$\hat{\boldsymbol{x}}$ is a least squares solution of $A\boldsymbol{x} = \boldsymbol{b}$
$\iff$ $R\hat{\boldsymbol{x}} = Q^T\boldsymbol{b}$ (where $A = QR$)