Some notes on QR factorization

Alen Alexanderian*

Abstract

We discuss some basics from theory and methods for computing QR factorizations.

Let $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \geq n$. The QR factorization of $\mathbf{A}$ is $\mathbf{A} = \mathbf{Q}\mathbf{R}$, where $\mathbf{Q} \in \mathbb{R}^{m \times n}$ has orthonormal columns and $\mathbf{R}$ is upper triangular. We also know, if $\mathbf{A}$ has full column rank, $\mathbf{A}$ has a unique QR factorization:

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

where $\mathbf{Q} \in \mathbb{R}^{m \times n}$ has orthonormal columns, and $\mathbf{R} \in \mathbb{R}^{n \times n}$ is upper triangular with positive diagonal entries.

One of the applications of QR factorization is solution of linear least squares. Consider a linear least squares problem,

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2 ,$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \geq n$, has full column rank, and $\boldsymbol{b} \in \mathbb{R}^m$. To solve this using QR factorization, we note that inserting the QR factorization $\mathbf{A} = \mathbf{Q}\mathbf{R}$ in the normal equations, $\mathbf{A}^T\mathbf{A}\boldsymbol{x} = \mathbf{A}^T\boldsymbol{b}$, and simplifying gives

$$\mathbf{R}\boldsymbol{x} = \mathbf{Q}^T\boldsymbol{b}.$$

Thus, the solution of the least squares problem is given by $\boldsymbol{x}^* = \mathbf{R}^{-1}\mathbf{Q}^T\boldsymbol{b}$. This leads to the following algorithm for solving a linear least squares problem:

1. Factorize $\mathbf{A} = \mathbf{Q}\mathbf{R}$.
2. Compute $\boldsymbol{c} = \mathbf{Q}^T\boldsymbol{b}$.
3. Solve $\mathbf{R}\boldsymbol{x} = \boldsymbol{c}$.

1 Some key results

The following is Theorem 3.2.20 in [2]. It can be proven by construction either using Givens rotators, or via Householder reflectors; see [2, Chapter 3] for details. In particular, in [2, p. 201] an induction proof using reflectors is presented.

Theorem 1.1. *Let $\mathbf{A} \in \mathbb{R}^{n \times n}$. Then there exists an orthogonal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$ such that $\mathbf{A} = \mathbf{Q}\mathbf{R}$.*

The following is Theorem 3.3.3 in [2]. It can be proven easily using the above Theorem; see [2, p. 213]

Theorem 1.2. *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m > n$. Then there exists $\mathbf{Q} \in \mathbb{R}^{m \times m}$ and $\tilde{\mathbf{R}} \in \mathbb{R}^{m \times n}$ such that $\mathbf{Q}$ is orthogonal, $\tilde{\mathbf{R}} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}$, where $\mathbf{R} \in \mathbb{R}^{n \times n}$ is upper triangular, and $\mathbf{A} = \mathbf{Q}\tilde{\mathbf{R}}$.*

The following result provides the existence result for the so called condensed (also known as reduced, or thin, or economy) QR factorization for a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m \geq n$. Note that the form of QR factorization given in the Theorem below is how we defined the QR factorization for such matrices in the beginning of this note.

Theorem 1.3. *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \geq n$. Then, there exist matrices $\mathbf{Q} \in \mathbb{R}^{m \times n}$ and $\mathbf{R} \in \mathbb{R}^{n \times n}$ such that $\mathbf{Q}$ has orthonormal columns, $\mathbf{R}$ is upper triangular, and $\mathbf{A} = \mathbf{Q}\mathbf{R}$.*

*North Carolina State University, Raleigh, NC, USA. E-mail: `alexanderian@ncsu.edu`
Last revised: September 27, 2022.

Proof. If $m = n$, this is just Theorem 1.1. If $m > n$, we know by Theorem 1.2 that there exist $\tilde{\mathbf{Q}} \in \mathbb{R}^{m\times m}$ orthogonal, and $\tilde{\mathbf{R}} = \begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix}$, with $\mathbf{R} \in \mathbb{R}^{n\times n}$ upper triangular such that $\mathbf{A} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}}$. Now, if we partition $\tilde{\mathbf{Q}} = \begin{bmatrix}\mathbf{Q} & \mathbf{Q}'\end{bmatrix}$, where $\mathbf{Q} \in \mathbb{R}^{m\times n}$ consists of the first n columns of $\tilde{\mathbf{Q}}$ and $\mathbf{Q}'$ contains the remaining columns, then

$$\mathbf{A} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}} = \begin{bmatrix}\mathbf{Q} & \mathbf{Q}'\end{bmatrix}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix} = \mathbf{Q}\mathbf{R} + \mathbf{Q}'\mathbf{0} = \mathbf{Q}\mathbf{R}. \qquad \square$$

To get uniqueness of the QR factorization, we need to assume $\mathbf{A}$ has full column rank. The following theorem makes this precise; see [3, p. 248]

Theorem 1.4. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$, have full column rank. Then, there exists unique $\mathbf{Q} \in \mathbb{R}^{m\times n}$ and $\mathbf{R} \in \mathbb{R}^{n\times n}$ such that $\mathbf{Q}$ has orthonormal columns, $\mathbf{R}$ is upper triangular with positive diagonal entries, and $\mathbf{A} = \mathbf{Q}\mathbf{R}$.*

In addition to the above, I also recommend studying [1, Section 3.7–3.8].

2 Computing QR factorizations

2.1 Using Givens rotators

A 2×2 givens rotator is of the form,

$$\mathbf{Q} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix},$$

where $c = \cos\theta$ and $s = \sin\theta$. For $\boldsymbol{x} \in \mathbb{R}^2$, if we define

$$c = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}, \quad s = \frac{x_2}{\sqrt{x_1^2 + x_2^2}},$$

then,

$$\mathbf{Q}^T\boldsymbol{x} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} cx_1 + sx_2 \\ -sx_1 + cx_2 \end{bmatrix} = \begin{bmatrix} \sqrt{x_1^2 + x_2^2} \\ 0 \end{bmatrix}$$

An $n \times n$ Givens rotator (also known as plane rotator) is of the form,

$$Q = \begin{bmatrix}
1 &&&&&&&&&& \\
& \ddots &&&&&&&&& \\
&& 1 &&&&&&&& \\
&&& c &&&& -s &&& \\
&&&& 1 &&&&&& \\
&&&&& \ddots &&&&& \\
&&&&&& 1 &&&& \\
&&& s &&&& c &&& \\
&&&&&&&& 1 && \\
&&&&&&&&& \ddots & \\
&&&&&&&&&& 1
\end{bmatrix},$$

where $c = \cos\theta$, $s = \sin\theta$ appear at the intersections ith and jth rows and columns, and the elements that are left unspecified are zeros. For $\boldsymbol{x} \in \mathbb{R}^n$, if we let $c = x_i/\sqrt{x_i^2 + x_j^2}$, $s = x_j/\sqrt{x_i^2 + x_j^2}$, then

$$\mathbf{Q}^T\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{i-1} & \sqrt{x_i^2 + x_j^2} & x_{i+1} & \cdots & x_{j-1} & 0 & x_{j+1} & \cdots & x_n \end{bmatrix}^T.$$

In the case $x_i = x_j = 0$, take $c = 1$ and $s = 0$.

We can use a finite sequence of Givens rotators to compute a QR factorization. See e.g., [2] for more details.

3 Householder reflectors

In this approach the idea is to use reflectors,

$$\mathbf{P}_{\boldsymbol{u}} = \mathbf{I} - 2\boldsymbol{u}\boldsymbol{u}^T, \quad \boldsymbol{u} \in \mathbb{R}^n \text{with } \|u\|_2 = 1, \tag{3.1}$$

to zero out elements in vectors. In particular, for a given $\boldsymbol{x} \in \mathbb{R}^n$, if we let $\boldsymbol{u}$ in (3.1) be given by

$$\boldsymbol{u} = \frac{\boldsymbol{x} \pm \|\boldsymbol{x}\|_2 \boldsymbol{e}_1}{\|\boldsymbol{x} \pm \|\boldsymbol{x}\|_2 \boldsymbol{e}_1\|_2},$$

then $\mathbf{P}_{\boldsymbol{u}}$ is a Householder reflector and

$$\mathbf{P}_{\boldsymbol{u}}\boldsymbol{x} = \begin{bmatrix} \mp \|\boldsymbol{x}\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

To avoid cancellation issues, we define Householder reflectors as follows

$$\boldsymbol{u} = \frac{\boldsymbol{x} + \operatorname{sign}(x_1) \|\boldsymbol{x}\|_2 \boldsymbol{e}_1}{\|\boldsymbol{x} + \operatorname{sign}(x_1) \|\boldsymbol{x}\|_2 \boldsymbol{e}_1\|_2},$$

with $\operatorname{sign}(a) = 1$ if $a \geq 0$ and $\operatorname{sign}(a) = -1$ otherwise.

Note that in practical computations there is no need to build the matrix for a reflector $\mathbf{P}_{\boldsymbol{u}}$. To apply $\mathbf{P}_{\boldsymbol{u}}$ to a vector $\boldsymbol{x}$, we can use

$$\mathbf{P}_{\boldsymbol{u}}\boldsymbol{x} = (\mathbf{I} - 2\boldsymbol{u}\boldsymbol{u}^T)\boldsymbol{x} = \boldsymbol{x} - 2\left(\boldsymbol{u}^T\boldsymbol{x}\right)\boldsymbol{u}.$$

As before, we can use a finite sequence of Householder reflectors to compute a QR factorization. This process is illustrated for a 5×3 matrix below.

Table 1: Illustration of QR using a succession of Householder reflectors. Matrix entries that are not necessarily zero are indicated by $\times$, and $\otimes$ indicates entries that change in each step.

$$\underset{\mathbf{A}}{\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}} \quad \underset{\mathbf{Q}_1\mathbf{A}}{\begin{bmatrix} \otimes & \otimes & \otimes \\ 0 & \otimes & \otimes \\ 0 & \otimes & \otimes \\ 0 & \otimes & \otimes \\ 0 & \otimes & \otimes \end{bmatrix}} \quad \underset{\mathbf{Q}_2\mathbf{Q}_1\mathbf{A}}{\begin{bmatrix} \times & \times & \times \\ 0 & \otimes & \otimes \\ 0 & 0 & \otimes \\ 0 & 0 & \otimes \\ 0 & 0 & \otimes \end{bmatrix}} \quad \underset{\mathbf{Q}_3\mathbf{Q}_2\mathbf{Q}_1\mathbf{A}}{\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \otimes \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}$$

4 Gram–Schmidt

Given a linearly independent set of vectors $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n\}$, with $\boldsymbol{a}_j \in \mathbb{R}^m$ ($m \geq n$), the Gram–Schmidt process can be used to obtain an orthonormal set $\{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n\}$ such that $\operatorname{span}\{\boldsymbol{a}_j\}_{i=1}^n = \operatorname{span}\{\boldsymbol{q}_j\}_{i=1}^n$. The process is as follows: compute $\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n$ according to

$$\boldsymbol{q}_1 = \frac{\boldsymbol{a}_1}{\|\boldsymbol{a}_1\|_2}, \quad \boldsymbol{q}_2 = \frac{\boldsymbol{a}_2 - \langle \boldsymbol{a}_2, \boldsymbol{q}_1\rangle \boldsymbol{q}_1}{\|\boldsymbol{a}_2 - \langle \boldsymbol{a}_2, \boldsymbol{q}_1\rangle \boldsymbol{q}_1\|_2}, \quad \boldsymbol{q}_3 = \frac{\boldsymbol{a}_3 - \langle \boldsymbol{a}_3, \boldsymbol{q}_1\rangle \boldsymbol{q}_1 - \langle \boldsymbol{a}_3, \boldsymbol{q}_2\rangle \boldsymbol{q}_2}{\|\boldsymbol{a}_3 - \langle \boldsymbol{a}_3, \boldsymbol{q}_1\rangle \boldsymbol{q}_1 - \langle \boldsymbol{a}_3, \boldsymbol{q}_2\rangle \boldsymbol{q}_2\|_2}, \ldots.$$

Generally, we compute $\boldsymbol{q}_1 = \frac{\boldsymbol{a}_1}{\|\boldsymbol{a}_1\|_2}$ and use

$$\boldsymbol{q}_j = \frac{\boldsymbol{a}_j - \sum_{i=1}^{j-1} \langle \boldsymbol{a}_j, \boldsymbol{q}_i\rangle \boldsymbol{q}_i}{\left\|\boldsymbol{a}_j - \sum_{i=1}^{j-1} \langle \boldsymbol{a}_j, \boldsymbol{q}_i\rangle \boldsymbol{q}_i\right\|_2}, \quad j = 2, \ldots, n.$$

The above procedure is known as the classical Gram–Schmidt process. This process can be used to compute the QR factorization of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ as outlined in Algorithm 1.

Algorithm 1 QR using Classical Gram–Schmidt process.

Input: $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$, with full column rank.
Output: $\mathbf{Q} \in \mathbb{R}^{m\times n}$ with orthonormal columns $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ and $\mathbf{R} = [r_{ij}] \in \mathbb{R}^{n\times n}$ upper triangular with $r_{ii} > 0$, such that $\mathbf{A} = \mathbf{QR}$.

```
for j = 1 to n do
  w = a_j                                   {a_j is the jth column of A}
  for i = 1 to j − 1 do
    r_ij = ⟨w, q_i⟩
  end for
  for i = 1 to j − 1 do
    w = w − r_ij q_i
  end for
  r_jj = ‖w‖_2
  q_j = w / r_jj
end for
```

Small illustrative example. Use Gram–Schmidt to compute the QR factorization of the matrix

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

Let $\boldsymbol{a}_i$ denote the ith column of $\mathbf{A}$, and $\boldsymbol{q}_i$ the ith column of $\mathbf{Q}$. In the first step, $\boldsymbol{q}_1$ is obtained by normalizing $\boldsymbol{a}_1$:

$$r_{11} = \sqrt{2}, \quad \boldsymbol{q}_1 = \boldsymbol{a}_1/r_{11} = \begin{bmatrix} 0 \\ -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}.$$

To get $\boldsymbol{q}_2$, first we compute

$$r_{12} = \langle \boldsymbol{a}_2, \boldsymbol{q}_1 \rangle = -1/\sqrt{2},$$

and then get

$$\boldsymbol{w} = \boldsymbol{a}_2 - r_{12}\boldsymbol{q}_1 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} + 1/\sqrt{2} \begin{bmatrix} 0 \\ -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 1 \\ -1/2 \\ -1/2 \end{bmatrix}.$$

Finally, we have $r_{22} = \|\boldsymbol{w}\| = \|(1, -1/2, -1/2)^T\| = \sqrt{3/2}$ and thus, $\boldsymbol{q}_2 = (1, -1/2, -1/2)^T/\sqrt{3/2} = (\sqrt{2/3}, -\sqrt{1/6}, -\sqrt{1/6})^T$. Thus, $\mathbf{Q}$ and $\mathbf{R}$ are given by

$$\mathbf{Q} = \begin{bmatrix} 0 & \sqrt{2/3} \\ -1/\sqrt{2} & -\sqrt{1/6} \\ 1/\sqrt{2} & -\sqrt{1/6} \end{bmatrix}, \quad \text{and } \mathbf{R} = \begin{bmatrix} \sqrt{2} & -1/\sqrt{2} \\ 0 & \sqrt{3/2} \end{bmatrix}$$

5 Computational considerations

While using a succession of orthogonal matrices to compute QR is generally a stable process, classical Gram–Schmidt is numerically unstable, due to loss of orthogonality. Fixes for Gram–Schmidt are available, such as modified Gram–Schmidt or reorthogonalization, and important to know, but of course they entail increased computational cost also.

QR with Householder reflectors is generally the preferred method for computing QR factorization of dense matrices. Using Givens rotators for computing QR for a dense matrix would cost roughly 50% more than that of Householder approach; see [4, p. 366]. However, for sparse matrices or for upper Hessenberg matrices (i.e., matrices with $a_{ij} = 0$ if $i > j + 1$) QR with Givens rotators is the method of choice.

Gram–Schmidt also has its place in computing QR. For one thing, when doing hand calculations for small matrices, Gram–Schmidt is convenient to use. More importantly, Gram–Schmidt is useful when columns of $\mathbf{A}$ are given one at a time. Also, in general, one can use Gram–Schmidt (in practice, its numerically stable variants), when computing an orthonormal basis for a space spanned by a given linearly independent set of vectors.

Finally, writing a computer code for computing QR is tricky. For real research work, it is best to rely on professionally developed software.

References

[1] I.C. Ipsen, Numerical Matrix Analysis. 2009

[2] David S. Watkins, Fundamentals of Matrix Computations, 3rd Edition. 2002.

[3] Golub, Gene H., and Charles F. Van Loan. Matrix computations. Vol. 3. JHU Press, 2012.

[4] Higham, Nicholas J. Accuracy and stability of numerical algorithms. Society for industrial and applied mathematics, 2002.