# The QR Method for Finding Eigenvalues

***CHAPTER 6 - PROJECT A***

*Note: The MATLAB M-files* **`qrmethod.m`***,* **`qrbasic.m`***, and* **`qrshift.m`** *are needed for this project. The M-files are downloadable from the same location as this PDF. Octave can be used in place of MATLAB.*

The purpose of this project is to show how the QR factorization of a matrix may be used to calculate the eigenvalues of the matrix.

There is no simple way to calculate eigenvalues for matrices larger than $2\times 2$ matrices. The method of calculating the characteristic polynomial and then finding its zeros is not good numerically. Finding the characteristic polynomial involves taking a determinant, which uses a large amount of computer time. Even if the characteristic polynomial is known, numerical algorithms like Newton's method cannot be depended upon to produce all of the zeros of the characteristic polynomial with reasonable speed and accuracy.

One of the great triumphs of numerical analysis since 1950 is to develop an entirely different way to calculate the eigenvalues of a square matrix $A$. The idea is to create a sequence of matrices similar to $A$ which converges to an upper triangular matrix; if this can be done then the diagonal entries of the limit matrix are the eigenvalues of $A$. The algorithm which produces this sequence of matrices uses the QR factorization of a matrix; almost every mathematical software package uses a variation of this strategy to compute the eigenvalues of a matrix. To introduce the algorithm, some properties that underlie this **QR method** for finding eigenvalues are first established. Recall that if $A$ is an $m\times n$ matrix with linearly independent columns, then $A$ can be factored as $A=QR$, where $Q$ is an $m\times n$ matrix with orthonormal columns and $R$ is an $n\times n$ invertible upper triangular matrix with positive entries on its main diagonal.

**Question:**

1. Suppose $A$ is an $n\times n$ matrix. Let $A=Q_0R_0$ be a QR factorization of $A$ and create $A_1=R_0Q_0$ . Let $A_1=Q_1R_1$ be a QR factorization of $A_1$ and create $A_2=R_1Q_1$.

   a) Show that $A=Q_0A_1Q_0^T$. (This is Exercise 31, Section 5.2.)

   b) Show that $A=(Q_0Q_1)A_2(Q_0Q_1)^T$.

   c) Show that $Q_0Q_1$ is an orthogonal matrix. (This is Exercise 37, Section 6.2.)

   d) Show that $A$, $A_1$, and $A_2$ all have the same eigenvalues.

The QR method for finding the eigenvalues of an $n \times n$ matrix $A$ extends this process to create a sequence of matrices with the same eigenvalues.

**The QR Method:**

**Step 1:** Let $A = Q_0 R_0$ be a QR factorization of $A$; create $A_1 = R_0 Q_0$.

**Step 2:** Let $A_1 = Q_1 R_1$ be a QR factorization of $A_1$; create $A_2 = R_1 Q_1$.

**Step 3:** Continue this process. Once $A_m$ has been created, let $A_m = Q_m R_m$ be a QR factorization of $A_m$ and create $A_{m+1} = R_m Q_m$.

**Step 4:** Stop the process when the entries below the main diagonal of $A_m$ are sufficiently small, or stop if it appears that convergence will not happen.

To see this process on MATLAB, enter the matrix $A$ and then type the following line:

```
[Q R]=qr(A); A=R*Q
```

Then use the arrow up key to repeat the command.

The special MATLAB command **`qrbasic`** in the downloaded file can do these steps repeatedly. After entering the matrix $A$, **`qrbasic(A,0.001)`** will perform these steps and will stop when all entries below the diagonal are smaller than 0.001 (or after 200 steps if that test is never met).

**Question:**

2. The matrices $B$, $C$, and $D$ are listed below. Using MATLAB, do enough steps of the QR method to make each entry below the main diagonal of the matrix smaller than 0.1. Record the number of steps, the final result, and give estimates for the eigenvalues of each matrix. You can either enter the matrices directly into MATLAB or type **`qrmethod`** for the data.

a) $B = \begin{pmatrix} 1 & -2 & 8 \\ 7 & -7 & 6 \\ 5 & 7 & -8 \end{pmatrix}$

b) $C = \begin{pmatrix} 4 & -2 & 3 & -7 \\ 1 & 2 & 6 & 8 \\ 8 & 5 & 1 & -5 \\ -5 & 8 & -5 & 3 \end{pmatrix}$

c) $D=\begin{pmatrix} 2 & 6 & -3 & 4 & -9 \\ -1 & 7 & -4 & -3 & -7 \\ -6 & -6 & -1 & 6 & 5 \\ 9 & 2 & 6 & 2 & -8 \\ -7 & -8 & 6 & -9 & -1 \end{pmatrix}$

The QR method can fail to converge, and it also can be extremely slow. The algorithm may be modified to make it converge both more quickly and for more matrices. One modification is **shifting**: at each step a scalar $c_i$ is chosen, then the QR factorization is done not on $A_i$ but $A_i - c_i I$. The matrix $c_i I$ is added back when $A_{i+1}$ is defined. If the scalars $c_i$ are chosen so that they get closer and closer to an eigenvalue of the matrix, this will dramatically speed up convergence.

**Question:**

3. Confirm that this modification will not change the outcome with the following steps. Let $A$ be an $n \times n$ matrix, let $I$ be the $n \times n$ identity matrix, and let $c$ be a scalar.

a) Let $\lambda$ be an eigenvalue of $A$. Show that $A\mathbf{x} = \lambda\mathbf{x}$ if and only if $(A - cI)\mathbf{x} = (\lambda - c)\mathbf{x}$. Use this to explain why the eigenvalues of $A - cI$ are those numbers obtained by subtracting $c$ from each eigenvalue of $A$. (This is why the process is called "shifting.")

b) Let $A - cI = QR$ be a QR factorization of $A - cI$ and define $A_1 = RQ + cI$. Show that $A = QA_1Q^T$.

Another improvement to the algorithm is called **deflating**. This process will allow us to remove the last row and column of the matrix when the last row looks like $\begin{pmatrix} 0 & 0 & \dots & 0 & x \end{pmatrix}$.

**Question:**

4. Suppose that $A = \begin{pmatrix} B & C \\ O & D \end{pmatrix}$, where $B$ and $D$ are square and $O$ is a zero matrix. Explain why the eigenvalues of $A$ are the eigenvalues of $B$ together with the eigenvalues of $D$. (This is Supplemental Exercise 34 in Chapter 5).

Including scaling and deflating produces the following improved algorithm for finding the eigenvalues of a matrix.

**The Modified QR Method:**

**Step 1:** Choose $c_0$ to be the last diagonal entry in $A$. Let $A - c_0 I = Q_0 R_0$ be a QR factorization of $A - c_0 I$; create $A_1 = R_0 Q_0 + c_0 I$.

**Step 2:** Choose $c_1$ to be the last diagonal entry in $A_1$. Let $A_1 - c_1 I = Q_1 R_1$ be a QR factorization of $A_1 - c_1 I$*;* create $A_2 = R_1 Q_1$.

**Step 3:** Continue this process. Once $A_m$ has been created, choose $c_m$ to be the last diagonal entry in $A_m$. Let $A_m - c_m I = Q_m R_m$ be a QR factorization of $A_m - c_m I$ and create $A_{m+1} = R_m Q_m + c_m I$.

**Step 4:** When the entries in the final row of $A_m$ (except for the final entry) are zero (to a tolerance previously selected), deflate $A_m$ by removing the final row and column. The final entry in the row just removed is an eigenvalue.

**Step 5:** Repeat Steps 1 through 4 on the deflated matrix until all the eigenvalues have been found or until it appears that convergence to a triangular limit matrix will not happen.

The special MATLAB command **`qrshift`** performs Steps 1-3 on a matrix. For example, the repeated use of the command **`A=qrshift(A, 0.001, 1)`** will perform each deflation step. To speed up the process, type

```
tic, qrshift(A, 0.0001), toc
```

The function **`qrshift`** does shifting as described in step 4, and it deflates after the entries below the diagonal are less than the tolerance you specify. It will display each deflation, the final matrix and how many iterative steps were done.

**Questions:**

5. Use MATLAB and the Modified QR Method to find the eigenvalues of the matrices $B$, $C$, and $D$ given above. Record the number of steps it takes to get the final result.

6. Let $A$ be the following matrix.

$$A = \begin{pmatrix} 0 & 0 & -1 & 4 & -1 & -6 \\ 0 & -2 & 2 & -5 & -2 & -5 \\ -1 & 2 & 8 & -4 & 3 & 2 \\ 4 & -5 & -4 & -6 & 1 & 0 \\ -1 & -2 & 3 & 1 & -2 & 7 \\ -6 & -5 & 2 & 0 & 7 & 10 \end{pmatrix}$$

Use MATLAB and the Modified QR Method to make all the entries below the main diagonal of $A$ less than 0.01. Record how many steps it takes to get this result, and record your estimates for the eigenvalues of $A$.

7. Is the Modified QR Method more efficient than the QR Method? What evidence do you have to support your conclusion?

**Closing Remarks:** For more information on the QR method, see Reference 1. Note that the following statements are true about these methods.

1. The above discussion applies only to matrices which have all real eigenvalues. If the matrix happens to have complex eigenvalues, then the QR methods may generate a sequence of matrices which converges, but the limit matrix will not be upper triangular.

2. If the eigenvalues of $A$ all have different magnitudes, then the QR Method will converge to an upper triangular matrix.

3. In practice, software performs a modified QR Method in two steps. First, the software finds a matrix which is similar to $A$ and is an **upper Hessenberg matrix**: it has all zeroes below its first subdiagonal. Then the QR Method is applied to this new matrix. The QR Method produces an upper Hessenberg matrix when given an upper Hessenberg matrix to work on, so the method will produce a sequence of upper Hessenberg matrices. In fact, this sequence will converge to a block upper triangular matrix.

**References:**

1. Watkins, D. "Understanding the QR Algorithm." *SIAM Review* **24** (1982), pp. 427-440.

**Acknowledgment:** The contents of this set of exercises were adapted from a computer project authored by Jane M. Day of San Jose State University.