Numerical Matrix Analysis
Lecture Notes #21 — Eigenvalues
The QR-Algorithm

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://terminus.sdsu.edu/

Spring 2016
Outline

1. First Section
   - Recap

2. The QR-Algorithm
   - The “Pure” QR-Algorithm
   - Improvements... for Cubic Convergence
   - Detour: Simultaneous Iteration

3. Back from the Detour
   - Simultaneous Iteration ⇔ QR-Algorithm
   - Equivalence Proof
   - Putting it Together: Convergence
We noted that eigenvalue revealing computations are generally divided into 2 phases; in **phase 1** we transform the matrix in into **Hessenberg form** in a finite number of steps, and in **phase 2** we apply a (possibly infinite) number of transformations to transform the Hessenberg matrix into **upper triangular form**.

Where the “+”-entries are zeros if $A$ is Hermitian.
Phase 1  Can be backwardly stably computed using a slightly modified version of the Householder-QR algorithm (making all the reflectors one element shorter, and applying $Q_k$ from the left and the right.)

Phase 2  Instead of directly talking about phase 2, we looked at...

- **Rayleigh quotient** (eigenvalue estimation).
- **Power iteration** (only useful as a basis for...)
- **Inverse iteration** (eigenvector estimation).
- **Rayleigh quotient iteration** (eigenvalue and eigenvector estimation — cubically convergent).

Next, we look at the QR-algorithm, and make some connections with the ideas above.
The QR-Algorithm

The QR-algorithm can be viewed as a stable procedure for computing QR-factorizations of the matrix power $A$, $A^2$, $A^3$, ... .

Algorithm (The “Pure” QR-Algorithm)

```plaintext
The “Pure” QR-Algorithm
$A^{(0)} = A$
for $k = 1$: ...
    $[Q^{(k)}, R^{(k)}] = \text{qr}(A^{(k-1)})$
    $A^{(k)} = R^{(k)}Q^{(k)}$
endfor
```

Under suitable (non-restrictive) assumptions, this simple algorithm converges to a Schur form for the matrix $A$ — Upper triangular if $A$ is arbitrary, and diagonal if $A$ is Hermitian.

Before we go any further, let us verify this numerically...
The QR-Algorithm: Applied to a Non-Hermitian $A$

Figure: The QR-algorithm applied to a non-Hermitian matrix $A$. The panels show the initial matrix, and iterations 1, 4, 16, 32, and 64.
The QR-Algorithm: Applied to $A = A^*$

Figure: The QR-algorithm applied to a Hermitian matrix $A$. The panels show the initial matrix, and iterations 1, 4, 16, 32, and 64.
The QR-Algorithm: What Are We Doing???

Just a quick sanity-check... If $QR = A$, then

$$Q^* AQ = Q^* (QR) Q = (Q^* Q) RQ = RQ$$

Hence the matrices that we form are unitarily similar

$$A_{(k)} = Q_{(k)}^* \cdots Q_{(1)}^* A_{(0)} Q_{(1)} \cdots Q_{(k)}$$

In fact, this is the idea we had to reject in our effort to compute the Hessenberg form of $A$. Even though it has to be rejected as a finite-step method for transforming $A$, it turns out to be quite powerful as the basis of an iterative scheme.

As in the last lecture, in order to keep the discussion simple(r), we assume that $A \in \mathbb{R}^{m \times m}$, and $A = A^*$ so that $\lambda_k(A) \in \mathbb{R}$, and the set of eigenvectors is orthonormal.
The QR-Algorithm: Modifications

Since we will be applying the QR-algorithm to real symmetric matrices, we are looking for the diagonalization $\Lambda(A)$.

Like the Rayleigh quotient algorithm, the QR-algorithm (for real symmetric matrices) can be made to converge cubically. In order to achieve this, we must introduce three modifications

1. Before entering the iteration, $A$ must be reduced to tri-diagonal form (using the “Hessenberg algorithm” (phase-1)).

2. Instead of $A^{(k)}$, the shifted matrix $A^{(k)} - \mu^{(k)}I$ is factored at each step, where $\mu^{(k)}$ is an eigenvalue estimate.

3. Whenever possible, and in particular whenever an eigenvalue is found, the problem is “deflated” by breaking $A^{(k)}$ into sub-matrices.
Algorithm (Modified QR-Algorithm)

\[ A^{(0)} = \text{hessenberg\_form}(A) \]
\[ \delta = \text{small tolerance} \sim \sqrt{\epsilon_{\text{mach}}} \]
for \( k = 1 : \ldots \)
Select \( \mu^{(k)} \)
\[ [Q^{(k)}, R^{(k)}] = \text{qr}(A^{(k-1)} - \mu^{(k)}I) \]
\[ A^{(k)} = R^{(k)}Q^{(k)} + \mu^{(k)}I \]
If any \( A^{(k)}_{j, j+1} \leq \delta \) then
Set \( A^{(k)}_{j, j+1} = A^{(k)}_{j+1, j} = 0 \), so that
\[ \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} = A^{(k)} \]
recursively apply the QR-algorithm to \( A_1 \) and \( A_2 \)
endfor
The Modified QR-Algorithm: Components

1. We have already discussed reduction to Hessenberg form.

2. We will return to a discussion on selecting the shifts $\mu^{(k)}$.

3. We leave the discussion on deflation as “an exercise for the motivated student.” (There are many details, e.g. pivoting to split the matrix exactly in “half,” to be taken care of to make this step maximally efficient)

For now, we focus the discussion on the “pure” form of the QR-algorithm... We relate the QR-algorithm to another method — simultaneous iteration — whose behavior is more obvious.
The Idea: Apply the power iteration to several vectors at once.

Suppose we have a set of \textbf{linearly independent} vectors \( \{v_1^{(0)}, \ldots, v_n^{(0)}\} \), then the spaced spanned by the vectors \( \{A^k \bar{v}_1^{(0)}, \ldots, A^k \bar{v}_n^{(0)}\} \) generated by simultaneous power iteration, converges to the space spanned by the \( n \) eigenvectors \( \bar{q}_k \) corresponding to the \( n \) abs-largest eigenvalues \( |\lambda_k| \), i.e.

\[
\lim_{k \to \infty} \langle A^k \bar{v}_1^{(0)}, \ldots, A^k \bar{v}_n^{(0)} \rangle = \langle \bar{q}_1, \ldots, \bar{q}_n \rangle.
\]

In matrix form

\[
V^{(0)} = \begin{bmatrix} \bar{v}_1^{(0)} & \cdots & \bar{v}_n^{(0)} \end{bmatrix}, \quad V^{(k)} = A^k V^{(0)} = \begin{bmatrix} \bar{v}_1^{(k)} & \cdots & \bar{v}_n^{(k)} \end{bmatrix}
\]
Since we are interested in the span, $\langle A^k \bar{v}_1^{(0)}, \ldots, A^k \bar{v}_n^{(0)} \rangle$, i.e. the column-space of $V^{(k)}$ we compute the reduced QR-factorization

$$\hat{Q}^{(k)} \hat{R}^{(k)} = V^{(k)}.$$ 

We can justify that the columns of $\hat{Q}^{(k)}$ converge to the eigenvectors $\bar{q}_k$; if we write both $\bar{v}_j^{(0)}$ and $\bar{v}_j^{(k)}$ in term of the eigenvectors of $A$

$$\bar{v}_j^{(0)} = a_{1j} \bar{q}_1 + \cdots + a_{mj} \bar{q}_m$$
$$\bar{v}_j^{(k)} = \lambda_1^k a_{1j} \bar{q}_1 + \cdots + \lambda_m^k a_{mj} \bar{q}_m.$$ 

For simplicity we assume that the first $n$ eigenvalues are distinct, and ordered so that

**Assumption #1**

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > |\lambda_{n+1}| \geq |\lambda_{n+2}| \geq \cdots \geq |\lambda_m|$$
We need one further assumption before we can state a theorem —
Let $\hat{Q}$ be the $m \times n$ matrix whose columns are the eigenvectors $\bar{q}_k$. We need the following to be true

**Assumption #2**

All the leading principal sub-matrices of $\hat{Q}^* V^{(0)}$ are non-singular.

A leading principal sub-matrix is anchored in the upper left corner (the $m_{11}$-element) and is a square matrix of size $1 \times 1$, or $2 \times 2$, $\ldots$, or $n \times n$.

With these assumptions we can say something about how the vectors generated by the simultaneous iteration converge to the eigenvectors.
Theorem

Suppose the iteration defined by

\[
V^{(0)} = \begin{bmatrix} \vec{v}_1^{(0)} & \cdots & \vec{v}_n^{(0)} \end{bmatrix}, \quad V^{(k)} = A^k V^{(0)}, \quad \hat{Q}^{(k)} \hat{R}^{(k)} = V^{(k)}
\]

is carried out, and that the assumptions (see slide 13 and 14) are satisfied. Then, as \( k \to \infty \), the columns of the matrices \( \hat{Q}^{(k)} \) converge linearly to the eigenvectors of \( A \)

\[
\| \bar{q}_j^{(k)} \mp \bar{q}_j \| = O(c^k)
\]

for each \( j \in [1, n] \), where \( c < 1 \) is the constant

\[
c = \max_{1 \leq k < n} \left| \frac{\lambda_{k+1}}{\lambda_k} \right|.
\]
We have a problem: As $k \to \infty$, all the vectors $\bar{v}_1^{(k)}, \ldots, \bar{v}_n^{(k)}$ in the unnormalized simultaneous iteration converge to the same dominant eigenvector $\bar{q}_1(A)$.

Even though the span $\langle \bar{v}_1^{(k)}, \ldots, \bar{v}_n^{(k)} \rangle$ converges to something useful, i.e. $\langle \bar{q}_1, \ldots, \bar{q}_n \rangle$, these vectors constitute a highly ill-conditioned basis for that space. For practical purposes this approach is useless.

The fix is straight-forward:

Necessary Improvement

We must orthonormalize the basis in every iteration. Instead of forming the sequence $V^{(k)}$, we form a sequence $Z^{(k)}$ with the same column spaces, but where $Z^{(k)}$ is orthonormal.
Algorithm (Simultaneous Iteration)

Let $\mathbf{Q}^{(0)} \in \mathbb{R}^{m \times n}$ with orthonormal columns
for $k = 1$:...

\[
\mathbf{Z}^{(k)} = A\mathbf{Q}^{(k-1)}
\]

\[
[\mathbf{Q}^{(k)}, \mathbf{R}^{(k)}] = \text{qr}(\mathbf{Z}^{(k)})
\]

endfor

Clearly, the column spaces of $\mathbf{Q}^{(k)}$ and $\mathbf{Z}^{(k)}$ are the same.
Also, as long as the initial matrices ($\mathbf{Q}^{(0)}$) are the same, this
algorithm generates the same sequence $\mathbf{Q}^{(k)}$ as the unnormalized
simultaneous iteration.

For the price of a QR-factorization per iteration we get a much
better conditioned sequence of basis for the space; $\langle \bar{\mathbf{z}}_1^{(k)} , \ldots , \bar{\mathbf{z}}_n^{(k)} \rangle$
$\rightarrow \langle \bar{\mathbf{q}}_1 , \ldots , \bar{\mathbf{q}}_n \rangle$. 

---

Peter Blomgren, (blomgren.peter@gmail.com)
The QR-algorithm is **equivalent** to simultaneous iteration applied to the full set \((n = m)\) of initial vectors, *i.e.* \(Q^{(0)} = I_{m \times m}\).

We are now dealing with the full QR-factorizations, so we drop the hats on \(Q^{(k)}\), and \(R^{(k)}\). Further, let \(Q^{(k)}\) denote the matrices generated by the simultaneous iteration, and \(Q^{(k)}\) be the matrices generated by the QR-algorithm...

<table>
<thead>
<tr>
<th>Simultaneous Iteration</th>
<th>Pure QR-Algorithim</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Q^{(0)}) = (I)</td>
<td>(A^{(0)} = A)</td>
</tr>
<tr>
<td>(Z^{(k)}) = (AQ^{(k-1)})</td>
<td>(Q^{(k)}R^{(k)} = A^{(k-1)})</td>
</tr>
<tr>
<td>(Q^{(k)}R^{(k)} = Z^{(k)})</td>
<td>(A^{(k)} = R^{(k)}Q^{(k)})</td>
</tr>
<tr>
<td>(A^{(k)} = (Q^{(k)})^*AQ^{(k)})</td>
<td>(Q^{(k)} = Q^{(1)}Q^{(2)} \ldots Q^{(k)})</td>
</tr>
<tr>
<td>(R^{(k)} = R^{(k)}R^{(k-1)} \ldots R^{(1)})</td>
<td></td>
</tr>
</tbody>
</table>

**Table:** The operations and quantities that define the Simultaneous Iteration algorithm and Pure QR-Algorithm.

- Peter Blomgren, ⟨blomgren.peter@gmail.com⟩

---

The QR Algorithm
The Simultaneous Iteration algorithm and the Pure QR-algorithm generate identical sequences of matrices \( R^{(k)} \), \( Q^{(k)} \), and \( A^{(k)} \), namely those defined by the QR-factorization of the \( k \)th power of \( A \),

\[
Q^{(k)} R^{(k)} = A^k,
\]

together with the projection

\[
A^{(k)} = (Q^{(k)})^* AQ^{(k)}.
\]

This is not obvious at first glance, so let’s look at the proof...
Equivalence of Simultaneous Iteration and QR-Algorithm

**Proof:** [By Induction] The base case \( k = 0 \) is trivial, for both SI and QR-Alg we immediately see that

\[
A^0 = Q^{(0)} = R^{(0)} = I, \quad A^{(0)} = A,
\]

from which

\[
A^0 = Q^{(0)} R^{(0)}, \quad A^{(0)} = (Q^{(0)})^* A Q^{(0)}. \quad \checkmark
\]

Now, consider \( k \geq 1 \) for SI: The second part of the theorem is valid by definition — \( A^{(k)} = (Q^{(k)})^* A Q^{(k)} \). The first part follows from

\[
A^k = A Q^{(k-1)} R^{(k-1)} = Q^{(k)} R^{(k)} R^{(k-1)} = Q^{(k)} R^{(k)}
\]

[1] Follows from the inductive hypothesis \( A^{k-1} = Q^{(k-1)} R^{(k-1)} \)

[2] From \( Q^{(k)} R^{(k)} = Z^{(k)} = A Q^{(k-1)} \)

[3] From \( R^{(k)} = R^{(k)} R^{(k-1)} \ldots R^{(1)} \)
Next we consider $k \geq 1$ for QR-Alg: We verify the first part of the theorem by the sequence

\[ A^k \overset{[1]}{=} AQ^{(k-1)}R^{(k-1)} \overset{[2]}{=} Q^{(k-1)}A^{(k-1)}R^{(k-1)} \overset{[3]}{=} Q^{(k)}R^{(k)} \]

- [1] Follows from the inductive hypothesis $A^{k-1} = Q^{(k-1)}R^{(k-1)}$
- [2] From the inductive hypothesis $A^{(k-1)} = (Q^{(k-1)})^*AQ^{(k-1)}$
- [3] From $Q^{(k)}R^{(k)} = A^{(k-1)}$, $Q^{(k)} = Q^{(1)}Q^{(2)}\ldots Q^{(k)}$, and $R^{(k)} = R^{(k)}R^{(k-1)}\ldots R^{(1)}$
Finally, we verify the second part by the sequence

\[ A^{(k)} \overset{[1]}{=} (Q^{(k)})^* A^{(k-1)} Q^{(k)} \overset{[2]}{=} (Q^{(k)})^* A Q^{(k)} \]

[1] Follows from \( Q^{(k)} R^{(k)} = A^{(k-1)} \), and \( A^{(k)} = R^{(k)} Q^{(k)} \)

[2] From the inductive hypothesis \( A^{(k-1)} = (Q^{(k-1)})^* A Q^{(k-1)} \), and \( Q^{(k)} = Q^{(1)} Q^{(2)} \ldots Q^{(k)} \)
Let’s put together the pieces of the “QR-Algorithm Jigsaw Puzzle”

- The relations (from the theorem)

  \[ Q^{(k)} R^{(k)} = A^k \]

  [i] \[ Q^{(k)} R^{(k)} = A^k \] tells us why we expect to find the eigenvectors — the QR-Algorithm constructs orthonormal bases for successive powers of \( A^k \)

  \[ A^{(k)} = (Q^{(k)})^* A Q^{(k)} \]

  [ii] \[ A^{(k)} = (Q^{(k)})^* A Q^{(k)} \] explains why we find the eigenvalues — the diagonal elements of \( A^{(k)} \) are the Rayleigh coefficients of \( A \) corresponding to the columns of \( Q^{(k)} \). As the columns converge to eigenvectors, the Rayleigh coefficients converge (“quadratically faster”) to the corresponding eigenvalues. Since \( Q^{(k)} \) converge to an orthonormal matrix, the off-diagonal elements in \( A^{(k)} \) must converge to zero.
Theorem

Let the pure QR-Algorithm be applied to a real symmetric matrix $A$ whose eigenvalues satisfy $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_m|$ and whose corresponding eigenvector matrix $Q$ has all non-singular leading principal sub-matrices. Then as $k \to \infty$, $A^{(k)}$ converges linearly with constant $\max_{1 \leq j < n} \left| \frac{\lambda_{j+1}}{\lambda_j} \right|$ to $\text{diag}(\lambda_1, \ldots, \lambda_m)$ and $Q^{(k)}$ converges at the same rate to $Q \pmod{\pm 1 \cdot \bar{q}_j^{(k)}}$.

Next, we look into adding shifts to the QR-Algorithm in order to speed up the convergence.