The QR-Algorithm
Connections with Other Iterative Schemes
Stability and Accuracy

Outline

1 The QR-Algorithm
   • Recap

2 Connections with Other Iterative Schemes
   • Inverse Iteration
   • Shifted Inverse Iteration
   • Rayleigh Quotient Iteration

3 Stability and Accuracy

Last Time: The QR-Algorithm

We introduced and discussed

Algorithm (The "Pure" QR-Algorithm)

\[
\begin{align*}
A(0) &= A \\
\text{while}(\ldots) & \left[ Q(k), R(k) \right] = q r(A(k-1)) \\
A(k) &= R(k) Q(k)
\end{align*}
\]

Which iteratively transforms a general matrix to upper triangular form, and a Hermitian matrix to diagonal form.

This time we look at introducing shifts into the QR-algorithm in order to speed up the convergence rate.

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

\[
\max_{1 \leq j < n} \left| \frac{\lambda_{j+1}}{\lambda_j} \right| \text{ to } \text{diag}(\lambda_1, \ldots, \lambda_m) \text{ and } Q(k) \text{ converges at the same rate to } Q \text{ (mod } \pm 1 \cdot \vec{x}^{(k)}). \]

Where \( Q(k) = Q(1) Q(2) \cdots Q(k) \).

Through a rather long-winded argument using the simultaneous iteration we were able to argue that the following theorem is true.
Connections with Other Iterative Schemes

We maintain the assumption that \( A \in \mathbb{R}^{m \times m} \) is real and symmetric; with real eigenvalues \( \lambda(A) \) and orthonormal eigenvectors \( \{\tilde{q}_j\}_{j=1,...,m} \).

We now make connections between the QR-algorithm and the three other iterative schemes we explored in our previous “detour” —

1. Inverse Iteration
2. Shifted Inverse Iteration
3. Rayleigh Quotient Iteration

With all these pieces in place, combined with the right shifting strategy, we can define a QR-algorithm with shifts, which generally converges cubically, and at least quadratically in the worst case.

Connection with Inverse Iteration

Let \( Q(k) \) be the orthogonal factor at the \( k^{th} \) step of the QR-algorithm, and let

\[
Q(k) = \prod_{j=1}^{k} Q(j) = [\tilde{q}_1^{(k)} | \tilde{q}_2^{(k)} | \cdots | \tilde{q}_m^{(k)}].
\]

This is the same orthogonal matrix that appears in the \( k^{th} \) step of simultaneous iteration, i.e. \( Q(k) \) is the orthogonal factor in the QR-factorization

\[
Q(k)R(k) = A^k.
\]

Now, using the symmetry of \( A \) (and therefore of \( A^{-1} \)), we have

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

Define the \((m \times m)\) permutation matrix \( P \), which reverses the row \((PA)\) or column \((AP)\) order

\[
P = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ & & 1 & 1 \
\end{bmatrix}, \quad p^2 = I, \quad P^* = P.
\]

We can now rewrite

\[
A^{-k}P = Q(k)P^2(R(k))^{-*}P = [Q(k)]P[(P(R(k))^{-*}P].
\]

Where the first factor \( Q(k)P \) is orthogonal, and the second \( P(R(k))^{-*}P \) is upper triangular. Hence we can interpret [1] as a QR-factorization of \( A^{-k}P \).
The QR-Algorithm
Connections with Other Iterative Schemes
Stability and Accuracy
Inverse Iteration
Rayleigh Quotient Iteration

Connection with Inverse Iteration
∃ Movie

4 of 5

We are, in effect, carrying out simultaneous iteration on $A^{-1}$ applied to the initial matrix $P$, i.e. simultaneous inverse iteration of $A$.

The first column of $Q(k)P$, i.e. the last column of $Q(k)$ is the result of applying $k$ steps of inverse iteration to $e_m$.

Hence, the QR-algorithm is in some sense performing both simultaneous iteration and simultaneous inverse iteration — a sense of perfect symmetry!

From our previous discussion, we know that inverse iteration can be accelerated significantly by introducing appropriate shifts...

Peter Blomgren ⟨blomgren@sdsu.edu⟩
21. The QR-Algorithm with Shifts — (9/21)

Peter Blomgren ⟨blomgren@sdsu.edu⟩
21. The QR-Algorithm with Shifts — (10/21)

Peter Blomgren ⟨blomgren@sdsu.edu⟩
21. The QR-Algorithm with Shifts — (11/21)

Peter Blomgren ⟨blomgren@sdsu.edu⟩
21. The QR-Algorithm with Shifts — (12/21)
The Rayleigh-Shifted QR-algorithm gives, 
\[ \mu_k = \frac{(q^{(k)}_m)^* A q^{(k)}_m}{\| q^{(k)}_m \|^2} = (\bar{q}^{(k)}_m)^* A \bar{q}^{(k)}_m. \]

If we use this shift, then the eigenvalue-eigenvector estimates \((\mu_k, \bar{q}^{(k)}_m)\) are identical to the ones computed by the Rayleigh quotient iteration, starting with \(\bar{e}_m\).

Hence, we inherit the cubic convergence for \((\mu_k, \bar{q}^{(k)}_m)\).

**Example of Rayleigh Quotient Shift Breakdown**

\[ A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \]

The Rayleigh-Shifted QR-algorithm gives, \(\mu_{(1)} = 0\), and

\[ Q_{(1)} R_{(1)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \]

\[ A_{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = A. \]

Rayleigh-shifting does not help since \(\mu(k) \equiv 0\).

**Good News:** The Rayleigh quotient is “free.” The \((m, m)\)-entry of \(A_{(k)}\) already contains the value

\[ A_{(k), m, m} = e_m^* A_{(k)} e_m = e_m^*(Q_{(k)})^* A Q_{(k)} e_m = (\bar{q}^{(k)}_m)^* A \bar{q}^{(k)}_m, \]

and \(\| \bar{q}^{(k)}_m \| = 1\).

Therefore, all we have to do is setting \(\mu_k = A_{(k), m, m}\).

This strategy is known as the **Rayleigh Quotient Shift**.

**Bad News:** Although this strategy, in general, gives cubic convergence, there are matrices for which the strategy does not converge at all.
The Wilkinson Shift is the eigenvalue of $B$ that is closest to $a_m$. When there is a tie, the choice is arbitrary [But must be made!]. The shift can be implemented as

$$
\mu_W(k) = a_m - \frac{\text{sign}(\delta)b_{m-1}^2}{|\delta| + \sqrt{\delta^2 + b_{m-1}^2}}, \quad \delta = \frac{a_{m-1} - a_m}{2}.
$$

If $\delta = 0$, then $\text{sign}(\delta)$ can arbitrarily be set to either 1 or $-1$. The Wilkinson shift achieves cubic convergence in general, and quadratic convergence in the worst case. In exact arithmetic the QR-algorithm with the Wilkinson shift always converges.

For the example that “broke” the Rayleigh shift is $\mu_W = \pm 1$, and we converge in one step.

### A Comment on $\text{sign}(x)$

Whereas the mathematical sign/signum function is

$$
\forall x \in \mathbb{R} : \quad \text{sign}(x) = \begin{cases} 
-1 & x < 0 \\
0 & x = 0 \\
1 & x > 0
\end{cases}
$$

The “computational science” sign/signum function is usually (always?)

$$
\forall x_F \in \mathbb{F}_{64,128,...} : \quad \text{sign}(x_F) = (-1)^s
$$

where $s \in \{0, 1\}$ is the value of the sign-bit of the floating-point value $x_F$.

This FORCES a choice $\pm 1$ for all values of $x_F$.

### Cleaning Up...

We now have all but one of the main components of the QR-algorithm: Once we have found $\lambda_m$ to desired accuracy, we should deflate the problem in an appropriate way in order to identify the remaining eigenvalues.

A full implementation, including a discussion of deflation strategies, may be a good project idea... for a dark and stormy night.

We conclude the discussion on the QR-algorithm with some comments regarding stability and accuracy.
Stability and Accuracy

Since the QR-algorithm is built using orthogonal transformations, we expect the algorithm to be backward stable; with \( \tilde{\Lambda} \) being the computed diagonalization, and \( \tilde{Q} \) being the exactly orthogonal matrix assembled from all the numerically computed Householder reflections used along the way, the following result holds:

\[ \tilde{Q} \tilde{\Lambda} \tilde{Q}^* = A + \delta A, \quad \frac{\|\delta A\|}{\|A\|} = O(\varepsilon_{\text{mach}}), \]

for some \( \delta A \in \mathbb{C}^{m \times m} \).

It follows that
\[ \frac{|\tilde{\lambda}_j - \lambda_j|}{\|A\|} = O(\varepsilon_{\text{mach}}). \]

Theorem

Let a real, symmetric, tridiagonal matrix \( A \in \mathbb{R}^{m \times m} \) be diagonalized by the QR-algorithm with shifts and deflation in a floating point environment satisfying the usual axioms, then we have

\[ \tilde{Q} \tilde{\Lambda} \tilde{Q}^* = A + \delta A, \quad \frac{\|\delta A\|}{\|A\|} = O(\varepsilon_{\text{mach}}), \]

for some \( \delta A \in \mathbb{C}^{m \times m} \). Next... Computing the SVD