Recap: Last Time 1 of 2

We noted that eigenvalue revealing computations are generally divided into 2 phases; in phase 1 we transform the matrix 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

\[
\begin{bmatrix}
* & * & * & * & * \\
* & * & * & * & * \\
* & * & * & * & * \\
* & * & * & * & * \\
* & * & * & * & *
\end{bmatrix}
\rightarrow
\begin{bmatrix}
* & + & + & + & + \\
* & * & + & + & + \\
* & * & + & + & + \\
* & * & + & + & + \\
* & * & + & + & +
\end{bmatrix}
\rightarrow
\begin{bmatrix}
* & + & + & + & + \\
* & * & + & + & + \\
* & * & + & + & + \\
* & * & + & + & + \\
* & * & + & + & +
\end{bmatrix}
\]

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 $A, A^2, A^3, \ldots$.

Algorithm (The "Pure" QR-Algorithm)

The "Pure" QR-Algorithm

$A^{(0)} = A$

for $k = 1; \ldots$

$[Q^{(k)}, R^{(k)}] = qr(A^{(k-1)})$

$A^{(k)} = R^{(k)}Q^{(k)}$

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...

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 $A = A^*$

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 $\lambda_i(A) \in \mathbb{R}$, 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.

The Modified QR-Algorithm

Algorithm (Modified QR-Algorithm)

$A^{(0)} = \text{hessenberg\_form}(A)$

$\delta = \text{small\ tolerance} \sim \sqrt{\epsilon_{\text{mach}}}$

for $k = 1$:...

  Select $\mu^{(k)}$

  $[Q^{(k)}, R^{(k)}] = qr(A^{(k-1)} - \mu^{(k)}I)$

  $A^{(k)} = R^{(k)}Q^{(k)} + \mu^{(k)}I$

  If any $A^{(k)}_{jj+1} \leq \delta$ then

  Set $A^{(k)}_{j,j+1} = A^{(k)}_{j+1,j} = 0$, so that

  $\begin{bmatrix}
  A_{11} & 0 \\
  0 & A_{22}
  \end{bmatrix} = A^{(k)}$

  recursively apply the QR-algorithm to $A_1$ and $A_2$

endfor

Unnormalized Simultaneous Iteration

The Idea: Apply the power iteration to several vectors at once.

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

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

In matrix form

$$V^{(0)} = \begin{bmatrix} v_1^{(0)} & \cdots & v_n^{(0)} \end{bmatrix}, \quad V^{(k)} = A^kV^{(0)} = \begin{bmatrix} v_1^{(k)} & \cdots & v_n^{(k)} \end{bmatrix}$$
Since we are interested in the span, \( \langle A^k \vec{v}_1^{(0)}, \ldots, A^k \vec{v}_n^{(0)} \rangle \), 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 \( \vec{q}_k \); if we write both \( \vec{v}_j^{(0)} \) and \( \vec{v}_j^{(k)} \) in term of the eigenvectors of \( A \)
\[
\begin{align*}
\vec{v}_j^{(0)} &= a_{1j} \vec{q}_1 + \cdots + a_{mj} \vec{q}_m \\
\vec{v}_j^{(k)} &= \lambda_k^j a_{1j} \vec{q}_1 + \cdots + \lambda_m^k a_{mj} \vec{q}_m.
\end{align*}
\]

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 \( \vec{q}_k \).

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 case 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 \( A \)

\[
\| \vec{q}_j^{(k)} + \vec{v}_j \| = O(c^k)
\]

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 \( \vec{v}_1^{(k)}, \ldots, \vec{v}_n^{(k)} \) in the unnormalized simultaneous iteration converge to the same dominant eigenvector \( \vec{q}_1(A) \).

Even though the span \( \langle \vec{v}_1^{(k)}, \ldots, \vec{v}_n^{(k)} \rangle \) converges to something useful, i.e. \( \langle \vec{q}_1, \ldots, \vec{q}_n \rangle \), 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.
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 \(\hat{Q}^{(k)}\) 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-Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{Q^{(0)}}{Z^{(k)}} = I )</td>
<td>( A^{(0)} = A )</td>
</tr>
<tr>
<td>( \frac{Q^{(k)}}{R^{(k)}} = Z^{(k)} )</td>
<td>( Q^{(k)}R^{(k)} = A^{(k-1)} )</td>
</tr>
<tr>
<td>( A^{(k)} = (Q^{(k)})^*AQ^{(k)} )</td>
<td>( Q^{(k)} = Q^{(1)}Q^{(2)} \ldots Q^{(k)} )</td>
</tr>
</tbody>
</table>

\( R^{(k)} = R^{(k)}R^{(k-1)} \ldots R^{(1)} \)

Table: The operations and quantities that define the Simultaneous Iteration algorithm and Pure 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

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

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

\[ A^k = \begin{bmatrix} AQ^{(k-1)}R^{(k-1)} \end{bmatrix} = Q^{(k)}R^{(k)}R^{(k-1)} = Q^{(k)}R^{(k)} \]

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

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

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

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

[3] From \( R^{(k)} = R^{(k)}R^{(k-1)} \ldots R^{(1)} \)
From the inductive hypothesis

\[ A^k = [Q(k-1)R(k-1)]^2 Q(k-1)A(k-1)R(k-1)^2 = Q(k)R(k) \]

\[ [1] \] 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 \( R(k) = R(k)R(k-1) \ldots R(1) \)

Next we consider \( k \geq 1 \) for QR-Alg: We verify the first part of the theorem by the sequence

Finally, we verify the second part by the sequence

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

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

\[ [2] \] From the inductive hypothesis \( A^{(k-1)} = (Q^{(k-1)})^*AQ^{(k-1)} \), 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)
  
  \[ [i] \] \( Q^{(k)}R^{(k)} = A^{k} \), tell us why we expect to find the eigenvectors — the QR-Algorithm constructs orthonormal bases for successive powers of \( A^{k} \)

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

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