⑧ The QR algorithm

We are now about to use all of the ideas from the previous sections to construct the QR algorithm, which returns, in an iterative way, both all the eigenvalues & all the eigenvectors of a real symmetric matrix.

The algorithm starts with a matrix A & eventually converges to a Schur form

ⓐ Simultaneous iterations

- The Rayleigh Quotient Algorithm of the previous section had the disadvantage of working on only 1 eigenvector at a time.
- We now construct an algorithm to work on all eigenvectors alltogether.

(!) Note that if we start with an orthonormal set $\{v_i^{(0)}\}$ of m vectors, where $m \leq n$

then the space spanned by $\{A^k v_i^{(0)}\}_i$ converges to the space spanned by the m evectors corresponding to the m-largest evalues.

Example: If $m = 2$ then

(suppose $\lambda_1 > \lambda_2 > \lambda_3 \dots > \lambda_n$)

$$v_1^{(0)} = a_1 q_1 + \cdots a_n q_n$$
$$v_2^{(0)} = b_1 q_1 + \cdots b_n q_n$$

so

$$A^k v_1^{(0)} = a_1 \lambda_1^k q_1 + a_2 \lambda_2^k q_2 + \cdots + a_n \lambda_n^k q_n$$
$$A^k v_2^{(0)} = b_1 \lambda_1^k q_1 + b_2 \lambda_2^k q_2 + \cdots b_n \lambda_n^k q_n$$

So as $k \to \infty$,

$$\left.\begin{aligned} A^k v_1^{(0)} &\simeq a_1 \lambda_1^k q_1 + a_2 \lambda_2^k q_2 \\ A^k v_2^{(0)} &\simeq b_1 \lambda_1^k q_1 + b_2 \lambda_2^k q_2 \end{aligned}\right\}$$ these two vectors span the same subspace as q_1 and q_2

However, if we leave $\begin{cases} v_1^{(k)} = A^k v_1^{(0)} \\ v_2^{(k)} = A^k v_2^{(0)} \end{cases}$

the new basis will be near singular because the vectors $v_1^{(k)}$ and $v_2^{(k)}$ are nearly parallel (since their $a_1\lambda_1^k q_1$ and $b_1\lambda_1^k q_1$ terms are respectively much bigger than $a_2\lambda_2^k q_2$ and $b_2\lambda_2^k q_2$).

$\Rightarrow$ the idea is to orthonormalize the vectors $\{v_i^{(k)}\}$ obtained at each step so that eventually the set of vectors $\{\underline{v}_i^{(k)}\}$ converges to the set $\{q_i\}$

Detour / Recap

Recall the QR decomposition.

Writing $A = QR$ is actually equivalent to writing the column vectors of A in the basis formed by the column vectors of Q (which is orthogonal)

Indeed:

$$\begin{pmatrix} | & | & & | \\ \underline{a}_1 & \underline{a}_2 & \cdots & \underline{a}_n \\ | & | & & | \end{pmatrix} = Q \begin{pmatrix} | & | & | & & | \\ \underline{r}_1 & \underline{r}_2 & \underline{r}_3 & \cdots & \underline{r}_n \\ | & | & | & & | \end{pmatrix}$$

$$\Leftrightarrow \quad \begin{cases} \underline{a}_1 = Q\underline{r}_1 \\ \underline{a}_2 = Q\underline{r}_2 \\ \underline{a}_3 = Q\underline{r}_3 \\ \vdots \\ \underline{a}_n = Q\underline{r}_n \end{cases} \Leftrightarrow \begin{cases} \underline{r}_1 = Q^T\underline{a}_1 \\ \underline{r}_2 = Q^T\underline{a}_2 \\ \vdots \\ \underline{r}_n = Q^T\underline{a}_n \end{cases}$$

- The set $\{\underline{r}_i\}$ are the vectors $\{\underline{a}_i\}$ expressed in this new basis.

- But in addition, since $\underline{r}_1 = \begin{pmatrix} x \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ then

$$\underline{a}_1 = \underline{q}_1 x$$

↳ the vector q_1 was selected to be parallel to the first vector of A $(\underline{a}_1)$

- So in fact we have constructed an orthogonal basis formed by the column vectors of Q, the first vector of which is parallel to the first column vector of A

↳ we have orthonormalized the basis formed by the column vectors of A (in a procedure similar to Graham Schmitt orthogonilization).

⇒ Idea

① Select m orthonormal vectors $\{v_i^{(0)}\}$ and construct the matrix

$$V^{(0)} = \begin{pmatrix} | & | & & | \\ v_1^{(0)} & v_2^{(0)} & \cdots & v_m^{(0)} \\ | & | & & | \end{pmatrix}$$

② Iterate the algorithm

$$Z^{(k)} = AV^{(k-1)}$$ ← multiply V by A to apply power iteration

$$Q^{(k)}R^{(k)} = Z^{(k)}$$ ← perform a QR decomposition on $Z^{(k)}$

$$V^{(k)} = Q^{(k)}$$ ← Say $Q^{(k)}$ is the next $V^{(k)}$ = orthonormalize the $\{V_i^{(k)}\}$ basis!

The idea here is that the QR reduction is also an orthogonalization of the columns of Z since Q is orthogonal by construction

(in other words, the columns of Z and the columns of Q span the same space, but the columns of Q have the advantage of being orthogonal & normalized).

After ∞ steps, $V^{(k)}$ converges to the set of m - orthogonal eigenvectors corresponding to the m largest evalues. This algorithm is called "the simultaneous iteration algorithm".

(b) Basic QR algorithm

The Basic QR Algorithm is the following.

$A^{(0)} = A$

for $k = 1 \ldots$

$Q^{(k)} R^{(k)} = A^{(k-1)}$ ← compute the QR factorization of $A^{(k-1)}$

$A^{(k)} = R^{(k)} Q^{(k)}$ ← define the new $A^{(k)}$ matrix as $R^{(k)} Q^{(k)}$

Claim ① This algorithm converges to a Schur form

② It is equivalent to simultaneous iterations

We will prove that it is indeed equivalent to the simultaneous iteration, and then give the exact Schur form in terms of the $Q^{(k)}$ and $R^{(k)}$

let's compare the two algorithms

QR algorithm

$A^{(0)} = A$

$Q^{(k)} R^{(k)} = A^{(k-1)}$

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

SI algorithm

$\hat{Q}^{(0)} = I$

$Z = A \hat{Q}^{(k-1)}$

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

(note the hats to denote QR factorization in the SI algor.)

Claims:

① $\hat{R}^{(k)} = R^{(k)}$

② $\hat{Q}^{(k)} = Q^{(1)} \cdots Q^{(k)}$

③ $A^{(k)} = \hat{Q}^{(k)T} A \hat{Q}^{(k)}$

④ $A^{k} = \hat{Q}^{(k)} R^{(k)} \cdots R^{(1)}$
$= Q^{(1)} \cdots Q^{(k)} R^{(k)} \cdots R^{(1)}$

Proof by induction

at $k=0$

$\hat{Q}^{(0)} = \hat{R}^{(0)} = I$ define $R^{(0)} = I = Q^{(0)}$

so ①,② ✓

$A^{0} = I$ which proves ④

$A^{(0)} = IAI$ so ③ ✓

assume it is true @ $k = k_0$

So $\hat{R}^{(k_0)} = R^{(k_0)}$

$\hat{Q}^{(k_0)} = Q^{(1)} \cdots Q^{(k_0)}$

$A^{(k_0)} = \hat{Q}^{(k_0)T} A \hat{Q}^{(k_0)}$

$A^{k_0} = \hat{Q}^{(k_0)} R^{(k_0)} \cdots R^{(1)}$

Then $A^{k_0+1} = AA^{k_0}$

$$= A\hat{Q}^{(k_0)} R^{(k_0)} \cdots R^{(1)}$$ ← From SI algorithm

$$= \hat{Q}^{(k_0+1)} \hat{R}^{(k_0+1)} R^{(k_0)} \cdots R^{(1)}$$

but also $= \hat{Q}^{(k_0)} A^{(k_0)} R^{(k_0)} \cdots R^{(1)}$ ← From QR

$$= \hat{Q}^{(k_0)} Q^{(k_0+1)} R^{(k_0+1)} R^{(k_0)} \cdots R^{(1)}$$

So now we can identify

$\hat{Q}^{(k_0+1)} = \hat{Q}^{(k_0)} Q^{(k_0+1)}$ which proves (2)

$\hat{R}^{(k_0+1)} = R^{(k_0+1)}$ which proves (1).

(3) and (4) follow easily: (4) has already been ok,

From $\hat{R}^{(k_0+1)} = \hat{Q}^{(k_0+1)T} A \hat{Q}^{(k_0)}$ ← from SI

$= R^{(k_0+1)} = Q^{(k_0+1)T} A^{(k_0)}$

but $A^{(k_0+1)} = R^{(k_0+1)} Q^{(k_0+1)} \Rightarrow$ $A^{(k_0+1)} = Q^{(k_0+1)T} A^{(k_0)} Q^{(k_0+1)}$

$$= \hat{Q}^{(k_0+1)T} A \hat{Q}^{(k_0+1)} \quad (3)$$

<u>This proves</u> that the QR algorithm provides matrices $A^{(k)}$

- which are orthogonally similar to A (because $A^{(k)} = \hat{Q}^{(k)T} A \hat{Q}^{(k)}$)

- because $\hat{Q}^{(k)}$ is such that $A^K = \hat{Q}^{(k)} \tilde{R}^{(k)}$ the columns of $\hat{Q}^{(k)}$ form an orthonormal basis for A^K.

- $\hat{Q}^{(k)}$ has columns which converge to the evectors of A (because of the successive iterations).

- $A^{(k)} = \hat{Q}^{(k)T} A \hat{Q}^{(k)}$ implies that the diagonal elements of $A^{(k)}$ contain the Rayleigh quotient of A w.r.t. the column vectors of $\hat{Q}^{(k)}$ so as $\hat{Q}^{(k)}$ converge to the evectors, the diagonal elements of A converge to the Evalues!

- The off-diagonal components of $A^{(k)}$ on the other hand converge to 0 because

$$q_j A q_i = 0 \text{ if } i \neq j.$$

(in a real, symmetric matrix, the evectors are orthogonal).

$\Rightarrow$ $A^{(k)}$ converges to the Schur form of A (in the case of a real, symmetric matrix, the Schur form is actually diagonal).

(c) Refinements to the basic QR algorithm

1. Shifts

- the basic QR algorithm is equivalent to a simultaneous power-iteration on all the vectors of the original basis $\{V_i^{(0)}\}$.
- We saw that the convergence rate of a power iteration can be greatly improved by using instead the inverse iteration, with shifts μ calculated using the Rayleigh Quotient. The slowest convergence is normally achieved for last Evector → select a shift μ close to λ_n (last evalue).

⇒ Shifted QR algorithm (on nxn real symmetric matrix)

- $A^{(0)} = A$
- iterate for $k = 1 \ldots$

$$\mu^{(k)} = A^{(k-1)}_{nn}$$ ← select shif as last evalue

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

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

← do a QR on the shifted matrix

Note that with this algorithm we still have

$$A^{(k)} = Q^{T(k)} A^{(k-1)} Q^{(k)}$$

$$= \hat{Q}^{(k)T} A \hat{Q}^{(k)}$$

so the diagonal elements of $A^{(k)}$ are the Rayleigh quotients.

Moreover, $A^{(k)}$ does converge to the matrix of the eigenvectors of A.

Note that there are other possible choices of shifts (see Trefethen & Bau).

2. Deflating

Because of the iterative nature of the algorithm, some eigenvalues are found faster than others. In that case, for speed of execution it is often interesting to "deflate" the matrix, once these are found, and only work on the remaining submatrix.

Example: If λ_1 is "found" first then

$$A^{(k)} \simeq \begin{pmatrix} \lambda_1 & 0 & \cdots & \cdots & 0 \\ 0 & \times & \times & & \\ \vdots & \times & \times & \times & \\ \vdots & & \times & \times & \ddots \\ 0 & & & \ddots & \ddots \end{pmatrix}$$

in next iteration only work with the remaining submatrix.

If λ_3 is found first then

$$A^{(k)} \simeq \begin{pmatrix} \times & \times & 0 & & & & \\ \times & \times & 0 & & & & \\ 0 & 0 & \lambda_3 & 0 & \cdots & \cdots & 0 \\ & & 0 & \times & \times & & \\ & & \vdots & \times & \times & \times & \\ & & \vdots & & \times & \times & \ddots \\ & & 0 & & & \ddots & \ddots \end{pmatrix}$$

then only work with remaining two other submatrices

3. Use of Hessenberg form

Recall that the use of Hessenberg form prior to using QR puts A into a tridiagonal form (since A is real, symmetric)

This can be used to your advantage by noting that if $A^{(0)}$ is tridiagonal then $A^{(k)}$ is too
$\rightarrow$ greatly reduces op-count, but is only really needed for matrices with large sizes.