The QR algorithm

➤ The most common method for solving small (dense) eigenvalue problems. The basic algorithm:

QR without shifts

1. Until Convergence Do:
2. Compute the QR factorization $A = QR$
3. Set $A := RQ$
4. EndDo

➤ "Until Convergence" means "Until A becomes close enough to an upper triangular matrix"

- **Note:** $A_{new} = RQ = Q^H(QR)Q = Q^HAQ$
- A_{new} **is similar to** A **throughout the algorithm .**
- **Above basic algorithm is never used in practice. Two variations:**

(1) use shift of origin and

(2) Transform A **into Hessenberg form..**

Practical QR algorithms: Shifts of origin

<u>Observation:</u> (from theory): Last row converges fastest. Convergence is dictated by $\frac{|\lambda_n|}{|\lambda_{n-1}|}$

➤ We will now consider only the real symmetric case.

➤ Eigenvalues are real.

➤ $A^{(k)}$ remains symmetric throughout process.

➤ As k goes to infinity the last column and row (except $a_{nn}^{(k)}$) converge to zero quickly.,,

➤ and $a_{nn}^{(k)}$ converges to lowest eigenvalue.

$$A^{(k)} = \left(\begin{array}{ccccc|c} . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ \hline a & a & a & a & a & a \end{array}\right)$$

➤ **Idea: Apply QR algorithm to $A^{(k)} - \mu I$ with $\mu = a_{nn}^{(k)}$. Note: eigenvalues of $A^{(k)} - \mu I$ are shifted by μ, and eigenvectors are the same.**

QR with shifts

1. Until row $a_{in}, 1 \leq i < n$ converges to zero DO:
2. Obtain next shift (e.g. $\mu = a_{nn}$)
3. $A - \mu I = QR$
5. Set $A := RQ + \mu I$
6. EndDo

➤ Convergence is cubic at the limit! [for symmetric case]

➤ **Result of algorithm:**

$$A^{(k)} = \left(\begin{array}{ccccc|c} \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & \lambda_n \end{array}\right)$$

➤ **Next step: deflate, i.e., apply above algorithm to $(n-1) \times (n-1)$ upper triangular matrix.**

Practical QR algorithms: Use of the Hessenberg Form

Recall: Upper Hessenberg matrix is such that

$$a_{ij} = 0 \text{ for } j < i - 1$$

<u>Observation:</u> The QR algorithm preserves Hessenberg form (tridiagonal form in symmetric case). Results in substantial savings.

Transformation to Hessenberg form:

- Consider the first step only on a 6×6 matrix.
- We want $H_1 A H_1^T = H_1 A H_1$ to have the form:

$$\begin{pmatrix} \star & \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \end{pmatrix}$$

- Choose a w in $H_1 = I - 2ww^T$ to make the first column have zeros from position 2 to n. So $w_1 = 0$.
- Apply to left: $B = H_1 A$
- Apply to right: $A_1 = BH_1$.

Main observation: the Householder matrix H_1 which transforms the column $A(2:n, 1)$ into e_1 works only on rows 2 to n. When applying the transpose H_1 to the right of $B = H_1 A$, we observe that only columns 2 to n will be altered. So the first column will retain the desired pattern (zeros below row 2).

- Algorithm continues the same way for columns 2, ...,$n - 2$.

QR for Hessenberg matrices

➤ **Need the "implicit Q theorem"**

Suppose that $Q^T AQ$ is an unreduced upper Hessenberg matrix. Then columns 2 to n of Q are determined uniquely (up to signs) by the first column of Q.

Implication: to compute $A_{i+1} = Q_i^T AQ_i$ we can:

➤ Compute 1st column of Q_i [== scalar $\times A(:,1)$]

➤ Choose other columns so Q_i = unitary, and A_{i+1} = Hessenberg.

WIl do this with Givens rotations

Example: With $n = 6$:

$$A = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

1. Choose $G_1 = G(1, 2, \theta_1)$ so that $Q(:, 1) = scal * A(:, 1)$

$$A_1 = G_1^T A G_1 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ + & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

2. Choose $G_2 = G(2, 3, \theta_2)$ so that $(G_2 A_1)_{31} = 0$

➤

$$A_2 = G_2^T A_1 G_2 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & + & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

3. Choose $G_3 = G(3, 4, \theta_3)$ so that $(G_3 A_2)_{42} = 0$

➤

$$A_3 = G_3^T A_2 G_3 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & + & * & * \end{pmatrix}$$

4. Choose $G_4 = G(4, 5, \theta_4)$ so that $(G_4 A_3)_{53} = 0$

- $$A_4 = G_4^T A_3 G_4 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$
- **Process known as "Bulge chasing"**
- **Similar idea for the symmetric (tridiagonal) case**

The symmetric eigenvalue problem: Basic facts

- Consider the Schur form of a real symmetric matrix A:

$$A = QRQ^H$$

Since $A^H = A$ then $R = R^H$ ➤

Eigenvalues of A are real

In addition, Q can be taken to be real when A is real.

$$(A-\lambda I)(u+iv) = 0 \rightarrow (A-\lambda I)u = 0 \;\&\; (A-\lambda I)v = 0$$

- Can select eigenvectors to be real.

There is an orthonormal basis of eigenvectors of A

The min-max theorem (Courant-Fischer)

Label eigenvalues decreasingly:

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$$

The eigenvalues of a Hermitian matrix A are characterized by the relation

$$\lambda_k = \max_{S,\ \dim(S)=k} \quad \min_{x \in S, x \neq 0} \quad \frac{(Ax, x)}{(x, x)}$$

➤ **Consequence:**

$$\lambda_1 = \max_{x \neq 0} \frac{(Ax, x)}{(x, x)} \qquad \lambda_n = \min_{x \neq 0} \frac{(Ax, x)}{(x, x)}$$

The Law of inertia

➤ Inertia of a matrix = [m, z, p] with m = number of < 0 eigenvalues. z = number of zero eigenvalues, and p = number of > 0 eigenvalues.

Sylvester's Law of inertia: If X is an $n \times n$ nonsingular matrix, then A and $X^T AX$ have the same inertia.

✍ Suppose that $A = LDL^T$ where L is unit lower triangular, and D diagonal. How many negative eigenvalues does A have?

✍ Assume that A is tridiagonal. How many operations are required to determine the number of negative eigenvalues of A?

✍ Devise an algorithm based on the inertia theorem to compute the i-th eigenvalue of a tridiagonal matrix.

✍ What is the inertia of the matrix

$$\begin{pmatrix} I & F \\ F^T & 0 \end{pmatrix}$$

where F is $m \times n$, with $n < m$, and of full rank?

[Hint: use a block LU factorization]

The QR algorithm for symmetric matrices

➤ Most important method used : reduce to tridiagonal form and apply the QR algorithm with shifts.

➤ Householder transformation to Hessenberg form yields a tridiagonal matrix because

$$HAH^T = A_1$$

is symmetric and also of Hessenberg form ➤ it is tridiagonal symmetric.

Tridiagonal form preserved by QR similarity transformation

Practical method

➤ How to implement the QR algorithm with shifts?

➤ It is best to use Givens rotations – can do a shifted QR step without explicitly shifting the matrix..

➤ Two most popular shifts:

$s = a_{nn}$ and $s =$ smallest e.v. of $A(n-1:n, n-1:n)$

Jacobi iteration - Symmetric matrices

➤ **Main idea: Rotation matrices of the form**

$$J(p,q,\theta) = \begin{array}{cc} \begin{pmatrix} 1 & \dots & 0 & & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & \cdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ 0 & \dots & 0 & & \dots & & 1 \end{pmatrix} & \begin{array}{c} \\ \\ p \\ \\ q \\ \\ \\ \end{array} \end{array}$$

$c = \cos\theta$ and $s = \sin\theta$ are so that $J(p,q,\theta)^T A J(p,q,\theta)$ has a zero in position (p,q) (and also (q,p))

➤ Frobenius norm of matrix is preserved – but diagonal elements become larger ➤ convergence to a diagonal.

- Let $B = J^T A J$ (indices p, q omitted).
- Look at 2×2 matrix $B([p,q],[p,q])$ (matlab notation)
- Keep in mind that $a_{pq} = a_{qp}$ and $b_{pq} = b_{qp}$

$$\begin{pmatrix} b_{pp} & b_{pq} \\ b_{qp} & b_{qq} \end{pmatrix} =$$

$$\begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{pmatrix} \begin{pmatrix} c & s \\ -s & c \end{pmatrix} =$$

$$\begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} ca_{pp} - sa_{pq} & sa_{pp} + ca_{pq} \\ ca_{qp} - sa_{qq} & sa_{pq} + ca_{qq} \end{pmatrix} =$$

$$\begin{pmatrix} c^2 a_{pp} + s^2 a_{qq} - 2sc\, a_{pq} & (c^2 - s^2) a_{pq} - sc(a_{qq} - a_{pp}) \\ * & c^2 a_{qq} + s^2 a_{pp} + 2sc\, a_{pq} \end{pmatrix}$$

- Want:

$$\boxed{(c^2 - s^2) a_{pq} - sc(a_{qq} - a_{pp}) = 0}$$

$$\frac{c^2 - s^2}{2sc} = \frac{a_{qq} - a_{pp}}{2a_{pq}} \equiv \tau$$

- Letting $t = s/c (= \tan\theta) = \quad \rightarrow$ quad. equation

$$\boxed{t^2 + 2\tau t - 1 = 0}$$

- $t = -\tau \pm \sqrt{1 + \tau^2}$
- Select sign to get a smaller t so $\theta \le \pi/4$.
- Then

$$c = \frac{1}{\sqrt{1+t^2}}; \qquad s = c * t$$

- Implemented in matlab script `jacrot(A,p,q)` – see matlab webpage of class.

➤ Define $$Off(A) = \|A - \mathrm{Diag}(A)\|_F$$

➤ Observations: (1) Unitary transformations preserve $\|.\|_F$. (2) Only changes are in rows and columns p and q.

➤ Let $B = J^T AJ$ (indices p, q omitted). Then,

$$a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2 = b_{pp}^2 + b_{qq}^2 + 2b_{pq}^2 = b_{pp}^2 + b_{qq}^2$$

because $b_{pq} = 0$. Then, a little calculation leads to

$$\begin{aligned} Off(B)^2 &= \|B\|_F^2 - \sum b_{ii}^2 = \|A\|_F^2 - \sum b_{ii}^2 \\ &= \|A\|_F^2 - \sum a_{ii}^2 + \sum a_{ii}^2 - \sum b_{ii}^2 \\ &= Off(A)^2 + (a_{pp}^2 + a_{qq}^2 - b_{pp}^2 - b_{qq}^2) \\ &= Off(A)^2 - 2a_{pq}^2 \end{aligned}$$

➤ $Off(A)$ will decrease from one step to the next.

✍ **Let $A_O = A -$ Diag(A). Then $Off(A) = \|A_O\|_F$. Let $\|A_O\|_I = \max_{i\neq j} |a_{ij}|$. Show that**

$$\|A_O\|_F \leq \sqrt{n(n-1)}\|A_O\|_I$$

✍ **Use this to show convergence in the case when largest entry is zeroed at each step.**