Lecture 10: Unsymmetric Eigen Value Problem

Prof. Krishna R. Pattipati

Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

ECE 6435
Adv Numerical Methods in Sci Comp

Fall 2008
October 29, 2008

Outline of Lecture 10

- What is the Eigen value ?
- Major Applications
- Properties of Eigen values and Eigen vectors
- Eigen value conditioning problem
- Power method for finding the maximum Eigen value and its Eigen vector
- Inverse power method
  - Smallest Eigen value and its Eigen vector
  - Iterative improvement
- QR method
  - Hessenberg decomposition
  - Shifted QR
  - Implicit Q-theorem
  - Francis double QR-step

Preliminaries

- What is the Eigen value problem?
  - Computing the Eigen values and Eigen vectors of a matrix A
  - Eigen values are also called **modes**, **spectrum**, **characteristic values or latent roots**
- Major applications
  - Converting a system $\dot{\underline{x}} = A\underline{x} + \underline{b}u$ into a canonical form (e.g., SCF, SOF, Diagonal form, Jordan form)
  - Solution of Riccati equation by Potter's method
- Properties of Eigen values and Eigen vectors:
  - Consider: $A\underline{x} = \lambda \underline{x}$

    λ: Eigen value of A; $\underline{x}$: Eigen vector of A

    $\Rightarrow A\underline{x} - \lambda I\,\underline{x} = 0 \Rightarrow (A - \lambda I)\underline{x} = 0$ or $\underline{x} \in N(A - \lambda I)$

  - $|A - \lambda I| = 0$ ch. equation of $A = \lambda^n + a_n\lambda^{n-1} + \ldots + a_2\lambda + a_1 = 0$

Properties of Eigen Values & Eigen Vectors

- $\sum_{i=1}^{n} \lambda_i = tr(A) = -a_n ; \prod_{i=1}^{n} \lambda_i = |A| = a_1$
- $A\underline{x}_i = \lambda_i \underline{x}_i ; \quad \underline{y}_i^T A = \underline{y}_i^T \lambda_i$

  $\underline{x}_i$ is called the right Eigen vector of A

  $\underline{y}_i$ is called the left Eigen vector of A = Eigen vector of A^T
- $\underline{x}_i^T \underline{y}_j = 0 \ \forall \ i \neq j$ why?

$$T^{-1} A T = \Lambda I \qquad T^{-1} = \begin{pmatrix} y_1^T \\ \dots \\ y_n^T \end{pmatrix}; \qquad T = (\underline{x}_1 \ \dots \ \underline{x}_n)$$

$$\Rightarrow \quad \underline{y}_j^T A \underline{x}_i = \lambda_i \underline{y}_j^T \underline{x}_i = 0; \underline{y}_i^T A \underline{x}_i = \lambda_i \underline{y}_i^T \underline{x}_i = \lambda_i$$

- $|\lambda_i| \leq \|A\|$ a pessimistic bound
- For symmetric matrices, we have

  $\lambda_{\min} \leq \underline{x}^T A \underline{x} / (\underline{x}^T \underline{x}) \leq \lambda_{\max}$

  $\underline{x}^T A \underline{x} / (\underline{x}^T \underline{x})$ is called the Rayleigh quotient
- For PD and symmetric $0 \leq \lambda_{\min} \leq \min a_{ii}$

Gershgorin Circle Theorem

Gershgorin Circle Theorem

- Valid for arbitrary *n* x *n* matrices
- Eigen values lie in the union of circles, i.e., $\lambda_i(A) \in \bigcup_e^n D_i,\ i = 1,...,n$ where $D_i = \{\lambda \in C:\ |\lambda - a_{ii}| \le \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| = r_i\}$

Proof:

Let λ be an Eigen value with Eigen vector, $\underline{x} = (x_1\ x_2\ \ldots\ x_n)^T$

$$A\underline{x} = \lambda \underline{x} \Rightarrow (\lambda - a_{ii})x_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}x_j \quad i = 1,...,n$$

pick $i = k$ ∋ $|x_k| = \|\underline{x}\|_\infty$, i.e., $|x_k| = \max_{1 \le j \le n} |x_j|$

Then, $$|\lambda - a_{kk}| \le \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| \frac{|x_j|}{|x_k|} \le \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| = r_k$$

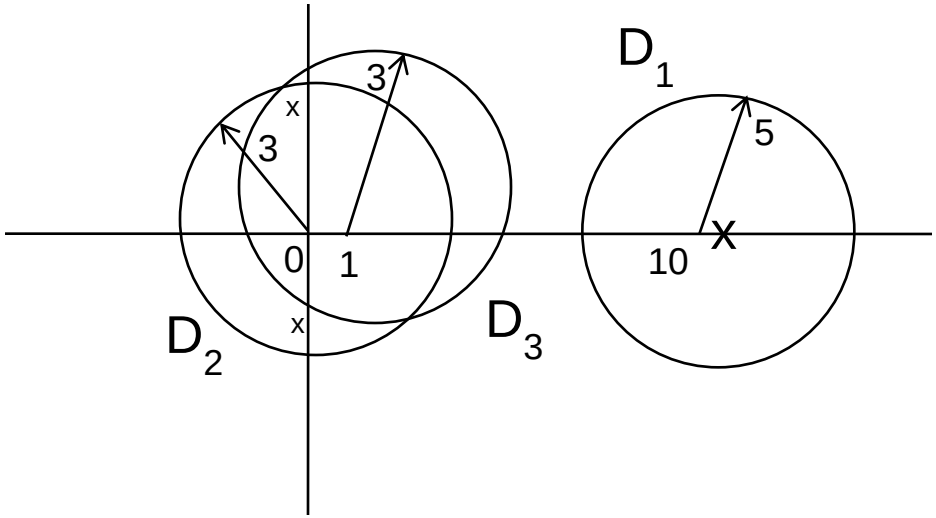
But don't know *k* a priori. So, take all possible circles

Gershgorin Circle Theorem

Example:

$$A = \begin{bmatrix} 10 & 2 & 3 \\ -1 & 0 & 2 \\ 1 & -2 & 1 \end{bmatrix}$$; λ(A)=(10.226, 0.3870+2.2216i, 0.3870 - 2.2216i)

$D_1 = \{\lambda : |\lambda\text{-}10| \le 5\}$; $D_2 = \{\lambda : |\lambda| \le 3\}$; $D_3 = \{\lambda : |\lambda\text{-}1| \le 3\}$;



Eigen Value Conditioning Problem - 1

- Eigen value conditioning problem
  - Suppose A has Eigen values $\lambda_1\ \lambda_2\ \ldots\ \lambda_n$.
  - Suppose A is perturbed $A+\delta A=B$
  - Q: how close $\lambda_i(A)$ and $\lambda_i(B)$ are?

Example:

$$A=\begin{bmatrix}0&1&\ldots&0\\0&0&1..&0\\0&\ldots&0&1\\0&\ldots&0&0\end{bmatrix},\quad B=\begin{bmatrix}0&1&\ldots&0\\0&0&1..&0\\0&\ldots&0&1\\10^{-10}&\ldots&0\end{bmatrix}$$

$$n=10\ \Rightarrow\ |\lambda I-A|=\lambda^{10}\Rightarrow\lambda=0$$

$$\text{SCF for}\ \Rightarrow\ |\lambda I-B|=\lambda^{10}-10^{-10}\Rightarrow|\lambda|=10^{-1}$$

Eigen Value Conditioning Problem - 2

- Analysis
  - Consider $A\underline{x} = \lambda \underline{x}$, $\underline{y}^T A = \underline{y}^T \lambda$ where $\underline{x}$ is the left Eigen vector and $\underline{y}$ is the right Eigen vector
  - $\Rightarrow d\lambda / d\varepsilon = S(\lambda) = \text{sensitivity} = 1/|\underline{y}^T \underline{x}|$
    Assume $\|\underline{x}\|_2 = \|\underline{y}\|_2 = 1$ (normalized)
  - So, sensitivity ≈ inverse of inner product of left and right Eigen vectors corresponding to the Eigen value
    $\Rightarrow$ if inner product is small $\Rightarrow$ trouble
  - Now, to formalize this notion
    - Consider $(A+\delta A)(\underline{x}+\delta \underline{x}) = (\lambda + d\lambda)(\underline{x}+\delta \underline{x})$
    - $A\underline{x} + \delta A\ \underline{x} + A\delta \underline{x} + \delta A\ \delta \underline{x} = \lambda \underline{x} + \lambda\ \delta \underline{x} + \delta\lambda\ \underline{x} + \delta\lambda\ \delta \underline{x}$
    - Noting that $\underline{y}^T A \delta \underline{x} = \lambda \underline{y}^T \delta \underline{x}$ and neglecting second order terms, we obtain:
      $\Rightarrow \delta\lambda = \underline{y}^T \delta A \underline{x} / (\underline{y}^T \underline{x}) \le |\delta A|\|\underline{x}\|\|\underline{y}\| / |\underline{y}^T \underline{x}| \le \|\delta A\| / |\underline{y}^T \underline{x}|$
    - So, smaller the inner product of left and right Eigen vectors, larger is the sensitivity of Eigen values

Eigen Value Conditioning Problem - 3

- So, condition number of the Eigen value problem is related to the inverse of the inner product of left and right Eigen vectors

$\underline{x}$ nearly $\perp^r$ to $\underline{y}$ $\Rightarrow$ problems

$$\text{Sensitivity, } S(\lambda) = \frac{1}{|\underline{y}^T \underline{x}|}$$

How are Eigen vectors affected

- To first order, we have for Eigen value λ_k and Eigen vector $\underline{x}_k$

$$A\,\delta\underline{x}_k + \delta A.\,\underline{x}_k = \delta\lambda_k\,\underline{x}_k + \lambda_k\delta\underline{x}_k$$

$$\delta\underline{x}_k = \sum_{i=1}^{n}\alpha_i\,\underline{x}_i \Rightarrow \sum_{i\neq k}\alpha_i(\lambda_i - \lambda_k)\underline{x}_i + \delta A\underline{x}_k = \delta\lambda_k\,\underline{x}_k$$

$$\Rightarrow \quad \alpha_i = \frac{y_i^T\,\delta A\underline{x}_k}{(\lambda_k - \lambda_i)y_i^T\,\underline{x}_i} \;\; \forall\; i \neq k \text{ (since } y_i^T\,\underline{x}_k = 0 \;\forall\; i \neq k)$$

$$\Rightarrow \quad \delta\underline{x}_k = \sum_{i=1}^{k}\frac{y_i^T\,\delta A\underline{x}_k}{(\lambda_k - \lambda_i)y_i^T\,\underline{x}_i}\underline{x}_i$$

- Sensitivity of $\underline{x}_k$ depends on $S_i(\lambda)$ and EV separation
- Eigen vectors associated with repeated Eigen values are wobbly!

Power Method for $\lambda_{max}(A)$ or $\lambda_{min}(A)$

- Power Method for finding $\lambda_{max}(A)$
  - Suppose $\{\lambda_i\}$ are simple, real and $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$

    $\Rightarrow$ Eigen vectors $\underline{x}_1\ \underline{x}_2\ \ldots\ \underline{x}_n$ are independent
  - Take any arbitrary $\underline{u}_0$. Then, since $\{\underline{x}_i\}$ span R^n ,

$$\underline{u}_0 = \sum_{i=1}^{n} c_i \underline{x}_i$$

  - Let us consider the following iteration

$$\underline{u}_{k+1} = A\underline{u}_k$$

$$\Rightarrow \quad \underline{u}_{k+1} = \lambda_1^{k+1} c_1 [\underline{x}_1 + \sum_{i=2}^{n} \frac{c_i}{c_1} \left(\frac{\lambda_i}{\lambda_1} \right)^k \underline{x}_i]$$

  - As $k \to \infty$, $\underline{u}_{k+1} \to c_1 \lambda_1^{k+1} \underline{x}_1 \Rightarrow$ proportional to $\underline{x}_1$
  - Note that if $|\lambda_i| < 1 \Rightarrow \|\underline{u}_{k+1}\| \to 0$ as $k \to \infty$

    if $|\lambda_i| > 1 \Rightarrow \|\underline{u}_{k+1}\| \to \infty$ as $k \to \infty$

    } not good
  - So, a good idea is to normalize $\underline{u}_k$ after each iteration

Normalized Power Method

- Normalized scheme

$$\hat{\underline{u}}_k = \frac{\lambda_1^k}{\| \underline{u}_k \|} \left[c_1 \underline{x}_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \underline{x}_2 + \ldots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \underline{x}_n \right]$$

$$\underline{u}_{k+1} = A \hat{\underline{u}}_k \text{ and } \hat{\underline{u}}_{k+1} = \frac{\underline{u}_{k+1}}{\| \underline{u}_{k+1} \|} \text{ continue}$$

as $k \to \infty$, $\underline{u}_{k+1} = \lambda_1 \hat{\underline{u}}_k$

$\Rightarrow$ 2 successive iterates are linearly dependent

can get $\lambda_1 = \frac{(\underline{u}_{k+1})_1}{(\hat{\underline{u}}_k)_1}$

- But, a better way is to treat it as an overdetermined set of equations

$$\underline{u}_{k+1} = \lambda_1 \hat{\underline{u}}_k \Rightarrow \text{can get } \lambda = \hat{\underline{u}}_k^T \underline{u}_{k+1}; \textit{recall } \hat{\underline{u}}_{k1}^T \hat{\underline{u}}_k = 1$$

Power Method & Rayleigh Quotient

- What happens in the symmetric case ?

Since $\underline{x}_i^T \underline{x}_j = 0 \quad \forall \; j \neq i,$ we have

$$\hat{\underline{u}}_k^T \underline{u}_{k+1} = \lambda_1 \left[1 + \mathrm{O}\left(\frac{\lambda_2}{\lambda_1} \right)^{2k+1} \right]$$

$$\hat{\underline{u}}_k^T \underline{u}_{k+1} = \hat{\underline{u}}_k^T A \hat{\underline{u}}_k \quad = \quad \text{Reyleigh quotient for symmetric matrices}$$

- Example

$$A = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \text{ has } \lambda_1 = 3 \text{ with } \underline{x}_1 = \begin{bmatrix} 0.7071 \\ -0.7071 \end{bmatrix}$$

$$\text{and } \lambda_2 = 1 \text{ with } \underline{x}_2 = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}$$

The power method gives a sequence:

$$\underline{u}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; \underline{u}_1 = \begin{bmatrix} 0.8994 \\ -0.4472 \end{bmatrix}, \; \underline{u}_2 = \begin{bmatrix} 0.7809 \\ -0.6247 \end{bmatrix}, \; \underline{u}_4 = \begin{bmatrix} 0.7328 \\ -0.6805 \end{bmatrix}, \; \ldots, \underline{u}_{10} = \underline{u}_\infty = \begin{bmatrix} 0.7071 \\ -0.7071 \end{bmatrix}$$

Power Algorithm for Real Eigen values - 1

Power algorithm for Real Eigen values:

Given an n x n matrix A and a tolerance parameter ε, the following algorithm computes $\lambda_{\max}(A)$.

Initialize $\underline{u} =$ arbitrary

$k = 0$

$s_{old} = 0$

$c = \| A \|_{\infty}$

For $k = 0,\ 1,\ 2,\ \ldots$ DO

$\rho = \| \underline{u} \|_2$

$\underline{w} = \underline{u} / \rho$

$\underline{u} = A\underline{w}$

$s_{new} = \underline{u}^T \underline{w}$

If $|s_{new} - s_{old}| < \varepsilon\ c$ stop: $\lambda_{\max} = s_{new},\ \underline{x}_1 = \underline{w}$

else

$k = k + 1$

$s_{old} = s_{new}$

end DO

Power Algorithm for Real Eigen values - 2

- Don't expect convergence for at least $\sqrt{n}$ iterations
- Note: $\hat{\underline{u}}_k \| \underline{u}_k \| = \underline{u}_k$ can be thought of as QR decomposition of $\underline{u}_k$
- Can I get p biggest λ_i ? Yes !!

Start with an n x p orthogonal matrix Q_0 ∋ $\underline{q}_i^T \underline{q}_j$

$$= 0 \quad \forall \quad i \neq j \quad \text{and} \quad \|\underline{q}_i\|_2 = 1$$

For $k = 1, 2, \ldots$ DO

$$Z_k = A\, Q_{k-1}$$

$$Q_k\, R_k = Z_k \quad \text{(QR factorization)}$$

SIMULTANEOUS ITERATION

end DO

- What if λ_i is complex:
  - Fact: complex Eigen values always occur in pairs
  - So, start with $\underline{u}_0 = c_1 \underline{x}_1 + c_1^* \underline{x}_2 + \ldots$
  - Assume λ_1 and λ_2 are rooots of $\lambda^2 - p\lambda - q = 0$

$$\Rightarrow \lambda_1 + \lambda_2 = p \quad \text{and} \quad \lambda_1 \lambda_2 = -q$$

Power Method for Complex Eigen values - 1

- Then

$$\underline{u}_k = \lambda_1^k c_1 \underline{x}_1 + \lambda_2^k c_1^* \underline{x}_2 + \ldots$$

$$\underline{u}_{k+1} = \lambda_1^{k+1} c_1 \underline{x}_1 + \lambda_2^{k+1} c_1^* \underline{x}_2 + \ldots = A\, \underline{u}_k$$

$$\underline{u}_{k+2} = \lambda_2^{k+1} c_1 \underline{x}_1 + \lambda_2^{k+2} c_1^* \underline{x}_2 + \ldots = A^2\, \underline{u}_k$$

$\Rightarrow$ as $k \to \infty$, it is easy to prove that

$$\underline{u}_{k+2} - (\lambda_1 + \lambda_2)\underline{u}_{k+1} + \lambda_1 \lambda_2 \underline{u}_k = \underline{0} \qquad \ldots \quad (1)$$

$\Rightarrow$ 3 successive iterates are linearly dependent

$\Rightarrow$ solve for $p,\ q$ as follows:

$$\hat{\underline{u}}_k = \text{unit norm vector} = \frac{\underline{u}_k}{\| \underline{u}_k \|_2}$$

$$\underline{u}_{k+1} = A\, \hat{\underline{u}}_k \ ; \ \hat{\underline{u}}_{k+1} = \frac{\underline{u}_{k+1}}{\| \underline{u}_{k+1} \|_2}$$

$$\underline{u}_{k+2} = A\, \underline{u}_{k+1} = \| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2\, \hat{\underline{u}}_{k+2}$$

Power Method for Complex Eigen values - 2

$\Rightarrow$ then, from (1),

$$\| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2 \hat{\underline{u}}_{k+2} - p \| \underline{u}_{k+1} \|_2 \hat{\underline{u}}_{k+1} - q \hat{\underline{u}}_k = 0$$

$$\begin{bmatrix} \hat{\underline{u}}_{k+1} & \hat{\underline{u}}_k \end{bmatrix} \begin{bmatrix} p \| \underline{u}_{k+1} \|_2 \\ q \end{bmatrix} = \| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2 \hat{\underline{u}}_{k+2}$$

- Least-squares normal equations

$$\begin{bmatrix} 1 & \hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \\ \hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k & 1 \end{bmatrix} \begin{bmatrix} p \| \underline{u}_{k+1} \|_2 \\ q \end{bmatrix} = \| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2 \begin{bmatrix} \hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_{k+2} \\ \hat{\underline{u}}_k^T \hat{\underline{u}}_{k+2} \end{bmatrix}$$

Power Method for Complex Eigen values - 3

$$\Rightarrow \quad p = \| \underline{u}_{k+2} \|_2 \frac{\left[\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_{k+2} - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right) \left(\hat{\underline{u}}_k^T \hat{\underline{u}}_{k+2} \right) \right]}{\left[1 - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right)^2 \right]}$$

$$= \| \underline{u}_{k+2} \|_2 \frac{\left[\hat{\underline{u}}_{k+2} - \left(\hat{\underline{u}}_k^T \hat{\underline{u}}_{k+2} \right) \hat{\underline{u}}_k \right]^T \hat{\underline{u}}_{k+1}}{1 - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right)^2}$$

$$q = \| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2 \frac{\left[\hat{\underline{u}}_k^T \hat{\underline{u}}_{k+2} - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right) \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_{k+2} \right) \right]}{\left[1 - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right)^2 \right]}$$

$$= \| \underline{u}_{k+1} \|_2 \| \underline{u}_{k+2} \|_2 \frac{\hat{\underline{u}}_k^T \left[\hat{\underline{u}}_{k+2} - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_{k+2} \right) \hat{\underline{u}}_{k+1} \right]}{\left[1 - \left(\hat{\underline{u}}_{k+1}^T \hat{\underline{u}}_k \right)^2 \right]}$$

Power Algorithm for Complex Eigen Values - 1

- Algorithm for finding complex Eigen values
  - Given an n x n matrix A and tolerance parameter ε, the following algorithm finds the complex Eigen values

$k = 0$

$\underline{u}_0 =$ any unit norm

$\underline{u}_1 = A\underline{u}_0$

$r_1 = \| \underline{u}_1 \|_2$

$c = \| A \|$

$p_{old} = 0$

$q_{old} = 0$

For $k = 0, 1, 2, \ldots$ DO

$\underline{u}_2 = A\underline{u}_1$

$r_2 = \| \underline{u}_2 \|$

$r = r_1 \, r_2$

$a = \underline{u}_0^T \underline{u}_1$

$b = \underline{u}_0^T \underline{u}_2$

$c = \underline{u}_1^T \underline{u}_2$

Power Algorithm for Complex Eigen values - 2

$$p_{new} = r_2(\underline{u}_2 - b\underline{u}_0)^T \underline{u}_1 / (1-a^2)$$

$$q_{new} = r\underline{u}_0^T(\underline{u}_2 - c\underline{u}_1) / (1-a^2)$$

If $|p_{new} - p_{old}| < c\varepsilon$ and $|q_{new} - q_{old}| < c\varepsilon$:

$$\mathrm{Re}(\lambda_1) = p_{new} / 2$$

$$\mathrm{Im}(\lambda_1) = \sqrt{p_{new}^2 + 4q_{new}} / 2$$

else

$$p_{old} = p_{new}$$

$$q_{old} = q_{new}$$

$$\underline{u}_0 = \underline{u}_1$$

$$\underline{u}_1 = \underline{u}_2$$

$$r_1 = r_2$$

$$k = k + 1$$

end if

end DO

Aitken's Acceleration

- To improve convergence, use Aitken δ^2 - iteration

  - if $\underline{u}_k = \underline{x} + a\varepsilon^k \quad k >> 1; \varepsilon < 1$

  - do component wise:

$$(\underline{u}_{k+1})_i = (\underline{u}_k)_i - \frac{(\Delta \underline{u}_k)_i^2}{(\Delta^2 \underline{u}_k)_i} = (\underline{u}_k)_i - \frac{((\underline{u}_k)_i - (\underline{u}_{k-1})_i)^2}{(\underline{u}_k)_i - 2(\underline{u}_{k-1})_i + (\underline{u}_{k-2})_i}$$

where $\underline{u}_k, \underline{u}_{k-1}, \underline{u}_{k-2}$ were generated from the usual algorithm.

$$(\underline{u}_{k+1})_i = (x_i + a\varepsilon^k) - \frac{(a\varepsilon^k - a\varepsilon^{k-1})^2}{a\varepsilon^k - 2a\varepsilon^{k-1} + a\varepsilon^{k-2}}$$

$$= x_i + a\varepsilon^k - \frac{a^2\varepsilon^{2(k-1)}(1-\varepsilon)^2}{a\varepsilon^{k-2}(1-\varepsilon)^2} \cong x_i \text{ to first order}$$

  - so, can do Aitken δ^2-iteration once every three iterations

Inverse Power iteration

- Useful for finding the smallest Eigen value as well as for iterative improvement

$$\underline{v}_{k+1} = A^{-1}\underline{v}_k \Rightarrow A\underline{v}_{k+1} = \underline{v}_k$$

$\Rightarrow$ do LU on A

$\Rightarrow$ store L and U factors

- can be used for iterative improvement as follows:

Suppose $\bar{\lambda}$ and $\underline{\bar{x}}$ are approximations to λ_1 and x_1.

$\underline{\bar{x}}$ is normalized, so that $\| \underline{\bar{x}} \|_2 = 1$

$$| \lambda_1 | > | \lambda_2 | > \ldots > | \lambda_n |$$

- then $(A - \bar{\lambda} I)$ has Eigen value $\lambda_i - \bar{\lambda} \approx 0$ with Eigen vector $\underline{x}_1$.

- $(A - \bar{\lambda} I)^{-1}$ has Eigen value $(\lambda_i - \bar{\lambda})^{-1} \approx$ big with Eigen vector $\underline{x}_1$.

- use power method on $B = (A - \bar{\lambda} I)^{-1}$

$$\Rightarrow \text{Do LU on } (A - \bar{\lambda} I) = \bar{L}\bar{U}$$

Inverse Power Iteration Algorithm - 1

- Example again:
  - the inverse power method $\underline{v}_{k+1} = A^{-1}\underline{v}_k$ gives the sequence

$$\underline{v}_0 = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}, \underline{v}_1 = \begin{bmatrix} 0.6727 \\ 0.7399 \end{bmatrix}, \underline{v}_2 = \begin{bmatrix} 0.6958 \\ 0.7182 \end{bmatrix}, \underline{v}_3 = \begin{bmatrix} 0.7034 \\ 0.7108 \end{bmatrix}, \ldots, \underline{v}_9 = \underline{v}_\infty = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}$$

❑ Inverse Power Iteration Algorithm:

Given A and approximate $\bar{\lambda}$ and $\underline{\bar{x}}$ for λ_1 and $\underline{x}_1$ respectively,

the algorithm computes better approximation to λ_1 and $\underline{x}_1$

$$\underline{u} = \underline{\bar{x}}$$

$$s_{old} = \bar{\lambda}$$

For k=0,1,2,...DO

$$\underline{w} = \frac{\underline{u}}{\| \underline{u} \|_2}$$

solve $\bar{L}\underline{v} = \underline{w}$

solve $\bar{U}\underline{u} = \underline{y}$

Inverse Power Iteration Algorithm - 2

$s_{new} = \underline{u}^T \underline{w}$

end DO

if $s_{new} \approx s_{old}$:Stop $\lambda_1 = \underline{u}^T A \underline{u} / \underline{u}^T \underline{u},\ \underline{x}_1 = \underline{u}$

else

$s_{old} = s_{new}$

$k = k + 1$

end if

end DO

- when $\underline{u}_k^s$ have converged $\lambda = \underline{u}_k^T A \underline{u}_k / (\underline{u}_k^T \underline{u}_k)$
- method works extremely well:

since $\underline{u}_k = \sum_{i=1}^{n} \frac{c_i}{(\lambda_i - \bar{\lambda})^k} \underline{x}_i$

if $\lambda_1 \approx \bar{\lambda}$, *then* $\underline{u}_k \to \underline{x}_1$

- shifting of spectrum is the key idea of QR algorithm as well
- shifting of spectrum improves convergence rate

QR Method

- A form of simultaneous iteration and much more
- Key ideas of QR
  - Reduce A to upper Hessenberg

$$A_1 = H_0 = Q_0^T A Q_0$$

$$\Rightarrow \text{a similarity transformation}$$

$$A = \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} \xrightarrow{Q_0^T A Q_0} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix} = H_0 = A_1$$

  - this can be done via Householder
  - Perform QR decomposition

    For k=1,2,…

$$A_k = Q_k R_k \quad \text{via Givens transformation}$$

$$A_{k+1} = R_k Q_k = Q_k^T A_k Q_k \sim \text{upper Hessenberg if } A_k \text{ is!!}$$

  - Note:

$$\lambda_i(A) = \lambda_i(H_0) \text{ and } \lambda_i(A_k) = \lambda_i(A_{k+1})$$

Example

- Illustration of $A_k = Q_k R_k \text{ and } A_{k+1} = R_k Q_k = Q_k^T A_k Q_k$

$$A = \begin{bmatrix} c & s \\ s & 0 \end{bmatrix} = \underbrace{\begin{bmatrix} c & s \\ s & -c \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} 1 & cs \\ 0 & s^2 \end{bmatrix}}_{R}$$

$$RQ = \begin{bmatrix} c(1+s^2) & s^3 \\ s^3 & -cs^2 \end{bmatrix}$$

- So, the off-diagonal elements go from s to s^3
- Subsequent iterations go to s^9, s^{27}, s^{81} …. because of a shifting operation we will employ in a practical scheme

  $\Rightarrow$ $A_{k+1} \rightarrow$ almost diagonal and we can obtain the Eigen values very fast!!
- Theorem: If $A_{k+1} = A$ is cyclic (i.e., ∃ n independent Eigen vectors), then the sequence

$$A_k = Q_k R_k \rightarrow A_{k+1} = R_k Q_k = Q_k^T A_k Q_k$$
$$= Q_k^T Q_{k-1}^T \ldots Q_0^T A Q_0 \ldots . Q_k$$

Converges to upper Schur form or upper quasi-triangular matrix

Upper Schur Form

$$= \begin{bmatrix} R_{11} & R_{12}.. & R_{1n} \\ 0 & R_{22}.. & R_{2n} \\ & .. & R_{nn} \end{bmatrix}$$ where each R_{ii} is a 1x1 or a 2x2 matrix

- Eigen values from R_{ii} subblocks
- If R_{ii} is 1x1, we get a simple Eigen value
- If R_{ii} is 2x2, we get a complex Eigen value

- Why do we transform to upper Hessenberg matrix first?

  - QR decomposition on a full n x n matrix requires $O(2n^3/3)$ operations
  - If A is upper Hessenberg, QR decomposition takes only $O(4n^2)$ operations
  - If Q is accumulated, a further $4n^2$ operations
  - More importantly, QR iteration maintains upper Hessenberg structure

Hessenberg Decomposition - 1

- Want an upper Hessenberg matrix H_0 from A such that

$$H_0 = Q_0^T A Q_0 \text{ and } Q_0^T Q_0 = I$$

$$Q_0 \sim \text{product of Householder matrices} = [I - 2\underline{u}\underline{u}^T / (\underline{u}^T \underline{u})]$$

$$\begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} \xrightarrow{W_1} \begin{bmatrix} x & y & y & y \\ y & y & y & y \\ 0 & y & y & y \\ 0 & y & y & y \end{bmatrix} \xrightarrow{W_2} \begin{bmatrix} x & y & y & y \\ y & y & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix}$$

Done! n-2 steps in general.

- Note that if first r components of $\underline{u}$ are zero, then

$$W = \begin{bmatrix} I_r & 0 \\ 0 & \Gamma \end{bmatrix} \text{ where } \Gamma = \Gamma^T$$

$$\text{and } WAW = \begin{bmatrix} A_{11} & A_{12}\Gamma \\ \Gamma A_{21} & \Gamma A_{22} \Gamma \end{bmatrix}$$

where $A_{11} = r \text{ x } r$ block of A

- Suppose $r = 3$, n=5 and

$$A = \left[\begin{array}{ccc|cc} x & x & x & x & x \\ x & x & x & x & x \\ - & - & - & + & - \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \end{array}\right]$$

Hessenberg Decomposition - 2

- 3x3 subblock does not change with Housholder

$$\text{pick } u = \begin{bmatrix} 0 & 0 & 0 & u_4 & .. & u_n \end{bmatrix}^T$$

$$\text{pick } u_4, u_5, u_6 \text{ to zero out } a_{53}, a_{63}, etc.$$

$$u_i = a_{i3} \;\; i \geq 5 \;\; u_4 = a_{43} + sign(\sum_{i=4}^{n} a_{i3}^2)^{1/2}$$

- In general, at step k, pivot on element $a_{k+1,k}$

- pivot column

$$\underline{u} = \begin{bmatrix} 0 \\ . \\ a_{k+1,k} + sign(a_{k+1,k})(\sum_{i=k+1}^{n} a_{ik}^2)^{1/2} \\ a_{k+2,k} \\ . \\ a_{n,k} \end{bmatrix}$$

Hessenberg Decomposition - 3

- Given a matrix *A*, the following algorithm computes an upper Hessenberg decomposition.
  - For k=1, 2,…,n-2 Do
    Determine Householder Γ_{n-k} э

$$\Gamma_{n-k}\begin{bmatrix} a_{k+1,k} \\ . \\ a_{n,k} \end{bmatrix} = \begin{bmatrix} sign(a_{k+1,k})(\sum_{i=k+1}^{n} a_{ik}^2)^{1/2} \\ 0 \\ . \\ 0 \end{bmatrix}$$

$$\Gamma_{n-k} = [I_{n-k} - \frac{u_{n-k}u_{n-k}^T}{\beta}]$$

$$\text{where } \underline{u}_{n-k} = \begin{bmatrix} a_{k+1,k} + sign(a_{k+1,k})s \\ a_{k+2,k} \\ . \\ a_{nk} \end{bmatrix}$$

QR Decomposition of *H* - 1

$$s = (\sum_{i=k+1}^{n} a_{ik}^2)^{1/2} \quad and \quad \beta = |u_{k+1} s|$$

$$W_k = Diag(I_k, \Gamma_{n-k})$$

$$A \leftarrow W_k A W_k = \begin{bmatrix} A_{11} & A_{12}\Gamma_{n-k} \\ \Gamma_{n-k}A_{21} & \Gamma_{n-k}A_{22}\Gamma_{n-k} \end{bmatrix}$$

end DO

- Finding *QR* decomposition of *H*:
  - need $H=QR$
  - but, actually need $H_{new} = RQ$ in the *QR*-algorithm
  - use Givens since *H* is almost upper Δ.

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & & x & x & x \\ & & x & x & x \end{bmatrix}$$

For $k = 1, 2, \ldots, n\text{-}1$ DO

QR Decomposition of *H* - 2

Determine $c_k = \cos\theta_k$ and $s_k = \sin\theta_k$ э

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix}\begin{bmatrix} h_{kk} \\ h_{k+1,k} \end{bmatrix} = \begin{bmatrix} x \\ 0 \end{bmatrix}; c = \frac{h_{kk}}{r}, s = \frac{h_{k+1,k}}{r}, r = -\text{sgn}(h_{kk})\sqrt{h_{kk}^2 + k_{k+1,k}^2}$$

$h_{kk} \leftarrow -\text{sgn}(h_{kk})r$ affects only two rows; $h_{k+1,k} \leftarrow 0$

For $j = k+1, \ldots, n$ *DO*

$$\begin{bmatrix} h_{kj} \\ h_{k,j+1} \end{bmatrix} = \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix}\begin{bmatrix} h_{kj} \\ h_{k+1,j} \end{bmatrix}$$

end DO

end DO

For k=1, 2,..., n-1

For i=1, 2,..., k+1

$$\begin{bmatrix} h_{ik} & h_{i,k+1} \end{bmatrix} = \begin{bmatrix} h_{ik} & h_{i,k+1} \end{bmatrix}\begin{bmatrix} c_k & -s_k \\ s_k & c_k \end{bmatrix}$$

end DO

QR Decomposition of *H* - Example

- This algorithm requires $O(4n^2)$ operations
- Example:

$$H = \begin{bmatrix} 3 & 1 & 2 \\ 4 & 2 & 3 \\ 0 & .01 & 1 \end{bmatrix}; J_1 = \begin{bmatrix} .6 & .8 & 0 \\ -.8 & .6 & 0 \\ 0 & 0 & 1 \end{bmatrix}; J_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & .9996 & .0249 \\ 0 & -.0249 & .9996 \end{bmatrix}$$

$$J_2^T J_1^T H J_1 J_2 = \begin{bmatrix} 4.76 & -2.5442 & 5.4653 \\ .32 & .1856 & -2.1786 \\ 0 & .0263 & 1.054 \end{bmatrix}$$

$\Rightarrow$ form of upper Hessenberg is maintained throughout subsequent iterations

Crude Version of QR

- Obtain $A_1 = H = Q_0 A Q_0$ via Householder Transformation
  - For k=1,2,…
    Obtain $A_k = Q_k R_k$ via Givens Transformation
    $$A_{k+1} = R_k Q_k = Q_k^T A_k Q_k$$
    If at some point A_{k+1} is ∋ a subdiagonal element $a_{p+1,p}$ is "small", i.e., $|a_{p+1,p}| \leq [|a_{pp}|+|a_{p+1,p+1}|]$ then
    set $|a_{p+1,p}| \sim 0$ and break A_{k+1} into subblocks as
    $$A_{k+1} = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}$$
    $A_{11} \sim$ upper Hessenberg
    $A_{22} \sim$ upper Hessenberg
    - If p=n-1, then $a_{nn} = \lambda_n$
    - If p=n-2, solve 2x2 subblock A_{22} for λ_n *and* λ_{n-1}
    - For other p, work on A_{22} and then on A_{11}
- Convergence rate: $|\lambda_n / \lambda_{n-1}|^k$
  where $|\lambda_1| > |\lambda_2| > ..... > |\lambda_n|$
- Typically the above algorithm converges very slowly

Acceleration Methods

- To speed up, use a shifted version of *QR*
- Consider a shifted version of A_k: A_k-$s_k I$
- recall that

$$\lambda_i(A_k - s_k I) = \lambda_i(A) - s_k$$

$$A_k - s_k I = Q_k R_k$$

$$A_{k+1} = R_k Q_k + s_k I = Q_k^T A_k Q_k \sim \text{similar to } A_k$$

$$\text{convergence of } a_{p+1,p} \to 0 \text{ with rate} |\lambda_{p+1} - s_k|^k / |\lambda_p - s_k|^k$$

$$\text{so, if } \lambda_n = s_k \text{ then } a_{n,n-1} \to 0 \text{ with rate } |\lambda_n - s_k|^k / |\lambda_{n-1} - s_k|^k$$

$$\text{if } \lambda_n = \lambda_{n-1}, \text{ No good! should be expected from the conditioning arguments earlier}$$

- Choice of s_k
  - Compute eigen values of lower 2x2 subblock

$$\begin{bmatrix} a_{n-1,n-1} & a_{n-1,n} \\ a_{n,n-1} & a_{n,n} \end{bmatrix} \to s_k, s_k^* \text{ (complex)}$$

$$\Rightarrow s_k + s_k^* = a_{n-1,n-1} + a_{nn};\ s_k s_k^* = a_{n-1,n-1}a_{nn} - a_{n,n-1}a_{n-1,n}$$

Double Shifts for Complex Eigen Values - 1

- Now, consider the following double shift QR

$$A_k - s_k I = Q_k R_k$$

$$A_{k+1} = R_k Q_k + s_k I$$

$$A_{k+1} - s_{k+1} I = Q_{k+1} R_{k+1}; s_{k+1} = s_k^*$$

$$A_{k+2} = R_{k+1} Q_{k+1} + s_{k+1} I$$

- What is happening here?

$$A_{k+2} = (Q_k Q_{k+1})^H A_k Q_k Q_{k+1} \qquad (Q_k)^H - \text{unitary}$$

$$= Q^T A_k Q \,(\text{real})$$

- What is $M = Q_k Q_{k+1} R_{k+1} R_k = QR$?

$$= Q_k (A_{k+1} - s_{k+1} I) R_k$$

$$= Q_k A_{k+1} R_k - s_{k+1}(A_k - s_k I) = (Q_k R_k)^2 + s_k Q_k R_k - -s_{k+1}(A_k - s_k I)$$

$$= (A_k - s_k I) Q_k R_k + s_k Q_k R_k - s_{k+1}(A_k - s_k I)$$

$$= A_k (A_k - s_k I) - s_{k+1}(A_k - s_k I)$$

Double Shifts for Complex Eigen Values - 2

$$= (A_k - s_{k+1} I)(A_k - s_k I)$$

$$= A_k^2 - (s_k + s_{k+1}) A_k + s_k s_k^* I$$

$\Rightarrow M$ is a real matrix

$\Rightarrow$ Since M is real, $Q = Q_k Q_{k+1}$ is real

$\Rightarrow$ so, have : $A_k^2 - (s_k + s_{k+1}) A_k + s_k s_k^* I = M$

$$and\ A_{k+2} = Q^T A_k Q$$

$\Rightarrow A_{k+2}$ can be obtained as a QR decomposition of real matrix:

- form $M = A_k^2 - (s_k + s_{k+1}) A_k + s_k s_k^* I$
- form $M = QR$
- form $A_{k+2} = Q^T A_k Q$

- But, computation of M requires $O(n^3)$ operations
- Q: can we obtain Q directly? Yes
- A: via FRANCIS QR-DOUBLE STEP

Francis Double Step

- Let us get greedy and ask: *Q*: can we go from $A_k \rightarrow A_{k+2}$ in $O(n^2)$ operations?
- A: Yes, via Francis *QR*-double step

❑ Ideas of Francis double step

- Compute the first column of $M = M\underline{e}_1$
- Find a Householder matrix W_0 ∋ $W_0 M\underline{e}_1 = \alpha\underline{e}_1$
- Apply Householder matrices $W_1 W_2 \ldots . W_{n-2}$ to $W_0 A_k W_0$ to reduce it to an upper Hessenberg form.

- Essentially, we have used $Q_1^T A_k Q_1$, where $Q_1 = W_{n-2} W_{n-3} \ldots . W_2 W_1 W_0$
- A theorem called implicit *Q*-theorem (see below) tells us that $Q = Q_1$
- Fact: $M\underline{e}_1$ has only 3 non-zero components

$$\Rightarrow M\underline{e}_1 = [x\ y\ z\ 0 \ldots . 0]^T$$

where $x = a_{11}^2 + a_{12}a_{21} - (s_k + s_{k+1})a_{11} + s_k s_k^*;$

$$(note: s_{k+1} = s_k^*)$$

$$y = a_{21}(a_{11} + a_{22} - (s_k + s_{k+1}))$$

$$z = a_{21}a_{32};\ \text{where}\ a_{ij}\ \text{are elements of}\ A_k$$

Implict *Q*- Theorem - 1

- When n=6, $W_0 A_k W_0$ will change rows and columns 1, 2 &3 only

$$W_0 A_k W_0 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- Implicit *Q*-Theorem:

  - suppose $Q = (\underline{q}_1\, \underline{q}_2 \ldots \underline{q}_n)$ and $V=(\underline{v}_1\, \underline{v}_2 \ldots \underline{v}_n)$ are orthogonal matrices ∋ $Q^T A Q = H$ and $V^T A V = G$ are upper Hessenberg (e.g., H and G are obtained via Householder or Givens transformation)
  - if $\underline{v}_1 = \underline{q}_1$, then $\underline{v}_i = \pm \underline{q}_i$ and $|h_{i,i-1}| = |g_{i,i-1}|$ for $i=1,2,\ldots,n$
  - formally, $AQ=QH$, $AV=VG$ and $\underline{v}_1 = \underline{q}_1$ then

    $Q=VD$, $V=D^{-1}GD=DGD$; $|d_i|=1 \ \forall i$

⇒ If $AQ=QH$ and the first column of Q is given, then Q and H are essentially determined.

Implict *Q*- Theorem - 2

- *Proof:* Consider

$$[A\underline{q}_1\ A\underline{q}_2 \ldots A\underline{q}_n] = [\underline{q}_1\ \underline{q}_2 \ldots \underline{q}_n] \begin{bmatrix} h_{11} & h_{12} & \ldots & h_{1n} \\ h_{21} & h_{22} & \ldots & h_{2n} \\ 0 & h_{32} & \ldots & h_{3n} \\ . & . & . & . \\ 0 & .. & h_{n,n-1} & h_{nn} \end{bmatrix}$$

$$A\underline{q}_1 = h_{11}\underline{q}_1 + h_{21}\underline{q}_2 \Rightarrow h_{11} = \underline{q}_1^T A\underline{q}_1; h_{21}\underline{q}_2 = A\underline{q}_1 - h_{11}\underline{q}_1$$

$$|\ h_{21}\ | = \|\ A\underline{q}_1 - h_{11}\underline{q}_1\ \|_2 \text{ so that } h_{21} \text{ is determined to within + or - sign}$$

$$A\underline{q}_2 = h_{12}\underline{q}_1 + h_{22}\underline{q}_2 + h_{32}\underline{q}_3$$

$$h_{12} = \underline{q}_1^T A\underline{q}_2; h_{22} = \underline{q}_2^T A\underline{q}_2;$$

$$\Rightarrow |\ h_{32}\ | = \|\ A\underline{q}_2 - h_{12}\underline{q}_1 - h_{22}\underline{q}_2\ \|_2$$

- If we insist on positive $h_{k+1,k}$, then q_i are uniquely determined for $i \geq 2$.
- Both Givens and Householder provide the same Q, since $\underline{q}_1 = \underline{e}_1$ for both.

Use of Implicit Q-Theorem - 1

- Coming back to *QR* Algorithm
  - the matrix we have is $M = A_k^2 - (s_k + s_k^*)A_k + s_k s_k^* I$
  - Francis *QR* step avoids forming $M = A_k^2 - (s_k + s_k^*)A_k + s_k s_k^* I$ explicitly
  - Direct formation of M requires $O(n^3)$ operations and not practical
  - We lose advantage of the upper Hessenberg structure
- We use implicit *Q*-theorem as follows:
  - Step 1: $M\underline{e}_1 = (A_k - s_k I)(A_k - s_k^* I)\underline{e}_1$ where A_k is an upper Hessenberg matrix,

$$\begin{bmatrix} a_{11} - s_k & a_{12} & \dots & a_{1n} \\ a_{21} & a_{12} - s_k & \dots & a_{2n} \\ 0 & a_{32} & \dots & a_{3n} \\ 0 & & a_{n,n-1} & a_{nn} \end{bmatrix}$$

$$M\underline{e}_1 = \begin{bmatrix} a_{11} - s_k^* \\ a_{21} \\ 0 \\ . \\ 0 \end{bmatrix} = \begin{bmatrix} (a_{11} - s_k)(a_{11} - s_k^*) + a_{12}a_{21} \\ a_{21}(a_{11} - s_k^*) + (a_{22} - s_k)a_{21} \\ a_{32}a_{21} \\ . \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 0 \\ . \\ 0 \end{bmatrix}$$

Use of Implicit Q-Theorem - 2

- Step 2: Find

$$W_0 \ni W_0 \begin{bmatrix} x \\ y \\ z \\ 0 \\ . \\ 0 \end{bmatrix} = \begin{bmatrix} * \\ 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix}, W_0 = I - \frac{\underline{u}\underline{u}^T}{\beta}, \underline{u} = \begin{bmatrix} x + \mathrm{sgn}(x)(x^2 + y^2 + z^2)^{1/2} \\ y \\ z \\ 0 \\ . \\ 0 \end{bmatrix}$$

- **Key**: Similarity transformation with W_0 affects rows 1, 2 and 3 and columns 1, 2 and 3 only.

$$W_0 A_k W_0 = \begin{bmatrix} \mathrm{x\,x\,x} & \\ \mathrm{x\,x\,x} & 0 \\ \mathrm{x\,x\,x} & \\ 0 & \mathrm{I} \end{bmatrix} \begin{bmatrix} \mathrm{x} & \mathrm{x} & \mathrm{x} & \ldots & \mathrm{x} \\ \mathrm{x} & \mathrm{x} & \mathrm{x} & \ldots & \mathrm{x} \\ & \mathrm{x} & \mathrm{x} & \ldots & \mathrm{x} \\ . & . & . & . & . \\ 0 & .. & 0 & \mathrm{x} & \mathrm{x} \end{bmatrix} \begin{bmatrix} \mathrm{x\,x\,x} & \\ \mathrm{x\,x\,x} & 0 \\ \mathrm{x\,x\,x} & \\ 0 & \mathrm{I} \end{bmatrix} = \begin{bmatrix} \mathrm{x\,x\,x} & \mathrm{x} \\ \mathrm{x\,x\,x} & \mathrm{x} \\ \mathrm{x\,x\,x} & \mathrm{x} \\ \mathrm{00x} & \mathrm{x} \\ \mathrm{.....0} & . \\ \mathrm{000} & \mathrm{x} \end{bmatrix} \begin{bmatrix} \mathrm{x\,x\,x} & \\ \mathrm{x\,x\,x} & 0 \\ \mathrm{x\,x\,x} & \\ 0 & \mathrm{I} \end{bmatrix}$$

Use of Implicit Q-Theorem - 3

$$= \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ & . & . & . & . & \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}$$

two more nonzero elements than needed
(3,1), (4,1), (4,2)

- Step 3: Compute Householder matrices $W_1, W_2, \ldots, W_{n-1}$ to convert $W_0 A_k W_0$ such that $W_{n-2} W_{n-3} \ldots W_1 W_0 W_1 .. W_{n-2}$ is upper Hessenberg

$$= \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ & . & . & . & . & . \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ . & x & x & x & x \\ . & x & x & x & x \\ & 0 & 0 & .. & \\ & . & . & . & \\ 0 & 0 & 0 & x & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ . & 0 & x & .. & \\ . & 0 & x & .. & \\ & . & x & .. & \\ & . & . & . & \\ 0 & 0 & 0 & x & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ .. & .. & x & x & x & \\ . & .. & x & x & x & \\ & . & . & . & . & . \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ . & 0 & x & x & x & x \\ . & . & . & x & x & x \\ & . & . & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \text{ etc.}$$

$\Rightarrow$ adds two more elements in each column

Formalizing Francis QR-double Step - 1

$$W_k = Diag\begin{bmatrix} I_k & \tilde{W}_k & I_{n-k-3} \end{bmatrix}$$

$$W_{n-2} = Diag\begin{bmatrix} I_{n-2} & \tilde{W}_{n-2} \end{bmatrix}$$

$$W_2 = \begin{bmatrix} 10 & & \\ 01 & 0 & \\ & \text{x x x} & \\ & \text{x x x} & \\ & \text{x x x} & \\ & & \text{I} \end{bmatrix}$$

Note: $W_k \underline{e}_1 = \underline{e}_1 \Rightarrow W_0$ and Q have the same first column $\Rightarrow Q_1 \underline{e}_1 = Q \underline{e}_1$ where $Q_1 = W_0 W_1 \ldots W_{n-2}$
$\Rightarrow Q_1 = Q$ by the implicit Q-theorem

- The mechanization of Francis QR-double step is as follows:

❑ Francis QR-double step:

$$m = n-1; s = a_{mm} + a_{nn}$$

$$t = a_{mm}a_{nn} - a_{mn}a_{nm}$$

$$x = (a_{11})^2 + a_{12}a_{21} - a_{11}s + t$$

$$y = a_{21}(a_{11} + a_{22} - s)$$

$$z = a_{21}a_{32}$$

For $k=1, 2, \ldots, n-2$

Formalizing Francis QR-double Step - 2

if $k < n-2$ then Determine $\tilde{W}_k$ $\tilde{W}_k \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} * \\ 0 \\ 0 \end{pmatrix}$

$$A \leftarrow W_k A W_k; \quad W_k = Diag[I_k \tilde{W}_k I_{n-k-3}]$$

else Determine $\tilde{W}_{n-2}$ э $\tilde{W}_{n-2} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} * \\ 0 \end{pmatrix}$

$$A \leftarrow W_{n-2} A W_{n-2}; \quad W_{n-2} = Diag[I_{n-2} \tilde{W}_{n-2}]$$

end if

$x = a_{k+2,k+1}; \quad y = a_{k+3,k+1}$

if $k < n-3$

$$z = a_{k+4,k+1}$$

end if

end DO

Overall QR- Algorithm - 1

- The overall QR-algorithm employing Francis double-step is as follows:

Overall QR-algorithm

Form $A_0 = Q_0^T A Q_0$ upper Hessenberg via Householder transformation

Form the orthogonal matrix $Q_0 \leftarrow W_{n-2} \ldots W_1$

Repeat

set to zero all subdiagonal elements satisfying

$$| a_{i+1,i} | \leq \varepsilon [| a_{ii} + | a_{i+1,i+1} | |]$$

Find the largest nonnegative q and smallest p э

$$A = \begin{matrix} p \\ n-q-p \\ q \end{matrix} \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$

note: can have $q = p = 0$

where A_{33} isupper quasi-triangular and A_{22} is unreduced upper Hessenberg

If $q = n$, then

upper triangularize all 2 by 2 diagonal blocks in A that have real eigen values. Accumulate orthogonal transformations if necessary and quit .

Overall QR- Algorithm - 2

Apply a Francis QR double – step to A_{22}:
else

$A_{22} \leftarrow Q_1^T A_{22} Q_1$
if Q and upper schur form of A are desired then
$Q \leftarrow \text{Diag}(I_p\ Q_1\ I_q)$
$A_{12} \leftarrow A_{12} Q_1$
$A_{23} \leftarrow Q_1^T A_{23}$
go to Repeat

- Total Computational effort $\approx O(15n^3)$ operations.
- If only Eigen values are desired, the computational load is $O(8n^3)$
- Balance A if A has widely differing Eigen values

Summary

- Properties of Eigen values and Eigen vectors
- Eigen value conditioning problem
- Power method for finding the maximum Eigen value and its Eigen vector
- Inverse power method
  - Smallest Eigen value and its Eigen vector
  - Iterative improvement
- QR method
  - Hessenberg decomposition
  - Shifted QR
  - Implicit Q-theorem
  - Francis double QR-step