

AMS526: Numerical Analysis I  
(Numerical Linear Algebra for  
Computational and Data Sciences)  
Lecture 17: Other Eigenvalue Algorithms;  
Generalized Eigenvalue Problems

Xiangmin Jiao

SUNY Stony Brook

# Outline

1 Other Eigenvalue Algorithms (NLA§30)

2 Generalized Eigenvalue Problems

# Three Alternative Algorithms

- Jacobi algorithm: earliest known method
- Bisection method: standard way for finding few eigenvalues
- Divide-and-conquer: faster than QR and amenable to parallelization

# The Jacobi Algorithm

- Diagonalize  $2 \times 2$  real symmetric matrix by *Jacobi rotation*

$$J^T \begin{bmatrix} a & d \\ d & b \end{bmatrix} J = \begin{bmatrix} \neq 0 & 0 \\ 0 & \neq 0 \end{bmatrix}$$

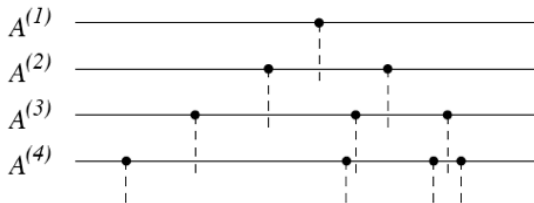
where  $J = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix}$ , and  $\tan(2\theta) = 2d/(b - a)$

- What are its similarity and differences with Givens rotation?
- Iteratively apply transformation to two rows and two corresponding columns of  $A \in \mathbb{R}^{n \times n}$
- Need not tridiagonalize first, but loop over all pairs of rows and columns by choosing greedily or cyclically
- Magnitude of nonzeros shrink steadily, converging quadratic
- In each iteration,  $O(n^2)$  Jacobi rotation,  $O(n)$  operations per rotation, leading to  $O(n^3 \log(|\log \epsilon_{\text{machine}}|))$  flops total
- Jacobi method is easy to parallelize (QR algorithm does not scale well), delivers better accuracy than QR algorithm, but far slower than QR algorithm

# Method of Bisection

- Idea: Search the real line for roots of  $p(x) = \det(A - xI)$
- Finding roots from coefficients is highly unstable, but computing  $p(x)$  from given  $x$  is stable (e.g., can be computed using Gaussian elimination with partial pivoting)
- Let  $A^{(i)}$  denote principal square submatrix of dimension  $i$  for irreducible matrix  $A$  (note: different from notation in QR algorithm)
- Key property: eigenvalues of  $A^{(1)}, \dots, A^{(n)}$  strictly interlace

$$\lambda_j^{(k+1)} < \lambda_j^{(k)} < \lambda_{j+1}^{(k+1)}$$



# Method of Bisection

- Interlacing property allows us to determine number of negative eigenvalues of  $A$ , which is equal to number of sign changes in *Sturm* sequence

$$1, \det(A^{(1)}), \det(A^{(2)}), \dots, \det(A^{(n)})$$

- Shift  $A$  to get number of eigenvalues in  $(-\infty, b)$  and  $(-\infty, a)$ , and in turn  $[a, b)$
- Three-term recurrence for determinants for tridiagonal matrices

$$\det(A^{(k)}) = a_{k,k} \det(A^{(k-1)}) - a_{k,k-1}^2 \det(A^{(k-2)})$$

- With shift  $xI$  and  $p^{(k)}(x) = \det(A^{(k)} - xI)$ :

$$p^{(k)}(x) = (a_{k,k} - x)p^{(k-1)}(x) - a_{k,k-1}^2 p^{(k-2)}(x)$$

- Bisection algorithm can locate eigenvalues in arbitrarily small intervals
- $O(n |\log(\epsilon_{\text{machine}})|)$  flops per eigenvalue, always high relative accuracy

# Notes on Bisection

- It is standard algorithm if one needs a few eigenvalues
- Key step of bisection is to determine the inertia (i.e., the numbers of positive, negative, and zero eigenvalues) of  $A - \mu I$
- *Sylvester's Law of Inertia*: inertia is invariant under *congruence transformation*  $SAS^T$ , where  $S$  is nonsingular (proved in 1852)
- Therefore,  $LDL^T$  may be used to determine inertia

# Divide-and-Conquer Algorithm

- Split symmetric algorithm  $T$  into submatrices

$$T = \begin{bmatrix} T_1 & \beta \\ \beta & T_2 \end{bmatrix} = \begin{bmatrix} \hat{T}_1 & \\ & \hat{T}_2 \end{bmatrix} + \begin{bmatrix} \beta & \beta \\ \beta & \beta \end{bmatrix}$$

- Sum of  $2 \times 2$  block-diagonal matrix and rank-one correction
- Split  $T$  in equal sizes and compute eigenvalues of  $\hat{T}_1$  and  $\hat{T}_2$  recursively
- Solve nonlinear problem to get eigenvalues of  $T$  from those of  $\hat{T}_1$  and  $\hat{T}_2$



# Divide-and-Conquer Algorithm

- Suppose diagonalization  $\hat{T}_1 = Q_1 D_1 Q_1^T$  and  $\hat{T}_2 = Q_2 D_2 Q_2^T$  have been computed. We then have

$$T = \begin{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix} \left( \begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} + \beta z z^T \right) \begin{bmatrix} Q_1^T & \\ & Q_2^T \end{bmatrix}$$

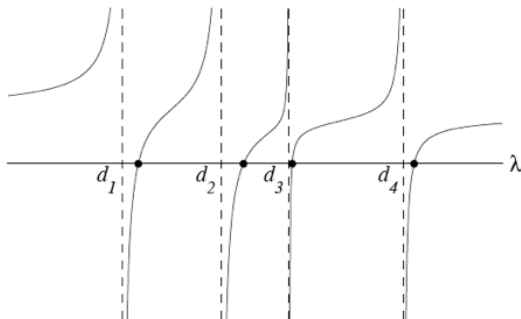
with  $z^T = (q_1^T, q_2^T)$ , where  $q_1^T$  is last row of  $Q_1$  and  $q_2^T$  is first row of  $Q_2$

- This is similarity transformation: Find eigenvalues of diagonal matrix plus rank-one correction

# Divide-and-Conquer Algorithm

- Eigenvalues of  $D + ww^T$  are the roots of rational function

$$f(\lambda) = 1 + \sum_{j=1}^m \frac{w_j^2}{d_j - \lambda}$$



# Divide-and-Conquer Algorithm

- Solve *secular equation*  $f(\lambda) = 0$  with nonlinear solver
- $O(n)$  flops per root,  $O(n^2)$  flops for all roots
- Total cost for divide-and-conquer algorithm is

$$O\left(\sum_{k=1}^{\log n} 2^{k-1} \left(\frac{n}{2^{k-1}}\right)^2\right) = O(n^2)$$

- For computing eigenvalues only, most of operations are spent in tridiagonal reduction, and constant in “Phase 2” is not important
- However, for computing eigenvectors, divide-and conquer reduces phase 2 to  $4n^3/3$  flops compared to  $6n^3$  for QR

# Outline

1 Other Eigenvalue Algorithms (NLA§30)

2 Generalized Eigenvalue Problems

# Generalized Eigenvalue Problem

- Generalized eigenvalue problem has the form

$$Ax = \lambda Bx,$$

where  $A$  and  $B$  are  $n \times n$  matrices

- For example, in structural vibration problems,  $A$  represents the *stiffness matrix*,  $B$  the *mass matrix*, and eigenvalues and eigenvectors determine natural frequencies and modes of vibration of structures
- If  $A$  or  $B$  is nonsingular, then it can be converted into standard eigenvalue problem  $(B^{-1}A)x = \lambda x$  or  $(A^{-1}B)x = (1/\lambda)x$
- If  $A$  and  $B$  are both symmetric, preceding transformation loses symmetry and in turn may lose orthogonality of generalized eigenvectors. If  $B$  is positive definite, alternative transformation is

$$(L^{-1}AL^{-T})y = \lambda y, \text{ where } B = LL^T \text{ and } y = L^Tx$$

- If  $A$  and  $B$  are both singular or indefinite, then use *QZ algorithm* to reduce  $A$  and  $B$  into triangular matrices simultaneously by orthogonal transformation (see Golub and van Loan for detail)