

Mathematics, Pusan National University

Numerical Linear Algebra

Lecture 30. Other Eigenvalue Algorithms

Taehyeong Kim
th_kim@pusan.ac.kr

November 29, 2020



Jacobi

Bisection

Divide-and-Conquer



Jacobi algorithm

One of the oldest ideas for computing eigenvalues of matrices introduced by Jacobi in 1845.

This method has attracted attention throughout the computer era, especially since the advent of parallel computing, though it has never quite managed to displace the competition.

standard approach

A 2×2 real symmetric matrix can be diagonalized in the form

$$J^T \begin{bmatrix} a & d \\ d & b \end{bmatrix} J = \begin{bmatrix} \neq 0 & 0 \\ 0 & \neq 0 \end{bmatrix}, \quad \text{where } J \text{ is orthogonal.} \quad (1)$$



There are several ways to choose J .

One could take it to be a 2×2 Householder reflection of the form

$$F = \begin{bmatrix} -c & s \\ s & c \end{bmatrix} \quad \text{where } s = \sin \theta \text{ and } c = \cos \theta \text{ for some } \theta. \quad (2)$$

One can use not a reflection but a rotation,

$$J = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \quad \text{with } \det J = 1. \quad (3)$$

This is the standard approach for the Jacobi algorithm. It can be shown that the diagonalization (1) is accomplished if θ satisfies

$$\tan(2\theta) = \frac{2d}{b-a} \quad (4)$$

and the matrix J based on this choice is called a *Jacobi rotation*.

Definition 1.1 (Givens rotation)

A Givens rotation is represented by a matrix of the form

$$G(i, j, \theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & -s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & c & \cdots & 1 \end{bmatrix} \quad \text{where } c = \cos \theta \text{ and } s = \sin \theta$$

The product $G(i, j, \theta)\mathbf{x}$ represents a counterclockwise rotation of the vector in the (i, j) plane of θ radians, hence the name Givens rotation.

Property of Givens rotation

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a & b \end{bmatrix} = \begin{bmatrix} r & 0 \end{bmatrix} \quad \begin{bmatrix} a \\ b \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

where $r = \sqrt{a^2 + b^2}$, $c = \frac{a}{\sqrt{a^2 + b^2}}$, $s = \frac{b}{\sqrt{a^2 + b^2}}$.

Example 1.2

$$A = \begin{bmatrix} 6 & 5 \\ 5 & 1 \end{bmatrix}$$

$$\Rightarrow r = \sqrt{A(1,1)^2 + A(1,2)^2} \approx 7.8102,$$

$$\Rightarrow c = \frac{6}{r} \approx 0.7682, s = \frac{-5}{r} \approx 0.6402$$

$$\Rightarrow G \approx \begin{bmatrix} 0.7682 & 0.6402 \\ -0.6402 & 0.7682 \end{bmatrix}$$

$$\Rightarrow GA \approx \begin{bmatrix} 7.8102 & 4.4813 \\ 0 & -2.4327 \end{bmatrix} = R, Q = G^T$$

Example 1.3

$$A = \begin{bmatrix} 6 & 5 \\ 5 & 1 \end{bmatrix}$$

$$\Rightarrow \theta = \frac{\arctan \frac{2A(1,2)}{A(2,2)-A(1,1)}}{2} \approx -0.5536,$$

$$\Rightarrow c = \cos \theta \approx 0.8507, s = \sin \theta \approx -0.5257$$

$$\Rightarrow G \approx \begin{bmatrix} 0.8507 & -0.5257 \\ 0.5257 & 0.8507 \end{bmatrix}$$

$$\Rightarrow G^T A G \approx \begin{bmatrix} 9.0902 & 0 \\ 0 & -2.0902 \end{bmatrix}$$

The Jacobi method is attractive because it deals only with pairs of rows and columns at a time, making it easily parallelizable (Exercise 30.4). The matrix is not tridiagonalized in advance; the Jacobi rotations would destroy that structure. Convergence for matrices of dimension $m \leq 1000$ is typically achieved in fewer than ten sweeps, and the final componentwise accuracy is generally even better than can be achieved by the QR algorithm. Unfortunately, even on parallel machines, the Jacobi algorithm is not usually as fast as tridiagonalization followed by the QR or divide-and-conquer algorithm (discussed below), though it usually comes within a factor of 10 (Exercise 30.2).



example_of_Jacobi.m

Our next algorithm is *bisection* is important.

We can find

- ▶ the largest 10% of the eigenvalues
- ▶ the smallest thirty eigenvalues
- ▶ all the eigenvalues in the interval $[1, 2]$.

The starting point is elementary. Since the eigenvalues of a real symmetric matrix are real, we can find them by searching the real line for roots of the polynomial $p(x) = \det(A - xI)$.

This sounds like a bad idea. But, our the idea is to find the roots by evaluating $p(x)$ at various points x , without everlooking at its coefficients, and applying the usual bisection process for nonlinear functions. This could be done, for example, by Gaussian elimination with pivoting (Exercise 21.1), and the resulting algorithm would be highly stable.

Exercises

21.1. Let A be the 4×4 matrix (20.3) considered in this lecture and the previous one.

(a) Determine $\det A$ from (20.5).

(b) Determine $\det A$ from (21.3).

(c) Describe how Gaussian elimination with partial pivoting can be used to find the determinant of a general square matrix.

This much sounds useful enough, but not very exciting. What gives the bisection method its power and its appeal are some additional properties of eigenvalues and determinants that are not immediately obvious.

For given a symmetric matrix $A \in \mathbb{R}^{m \times m}$, let $A^{(1)}, \dots, A^{(m)}$ denote its principal square submatrices of dimension $1, \dots, m$. It can be shown that the eigenvalues of the matrices *interlace*. Assume that A is tridiagonal and *irreducible* in the sense that all of its off-diagonal entries are non zero:

$$A = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & b_2 & a_3 & \ddots & \\ & & \ddots & \ddots & b_{m-1} \\ & & & b_{m-1} & a_m \end{bmatrix}, \quad b_j \neq 0. \quad (5)$$

By Exercise 25.1, the eigenvalues of $A^{(k)}$ are distinct.

Exercise

25.1. (a) Let $A \in \mathbb{C}^{m \times m}$ be tridiagonal and hermitian, with all its sub- and superdiagonal entries nonzero. Prove that the eigenvalues of A are distinct. (Hint: Show that for any $\lambda \in \mathbb{C}$, $A - \lambda I$ has rank at least $m - 1$.)

Sol) Consider $A - \lambda I$, which is also tridiagonal and all its sub- and superdiagonal entries are nonzero. We can write $A - \lambda I$ as a block matrix $\begin{bmatrix} v^T & 0 \\ B & u \end{bmatrix}$ where B is an $(m - 1) \times (m - 1)$ matrix and u and v are vectors of length $m - 1$. Notice that B is uppertriangular and with nonzero diagonal entries. Thus B is non-degenerate. If $|A - \lambda I| = 0$, then $\text{rank}(A - \lambda I) = m - 1$. This means $\dim(\text{null}(A - \lambda I)) = 1$, i.e. the geometric multiplicity of A is 1. As A is hermitian, then geometric multiplicity coincides with the algebraic multiplicity. Thus A has all distinct eigenvalues.

So, let eigenvalues of $A^{(k)}$ be denoted by $\lambda_1^{(k)} < \lambda_2^{(k)} < \dots < \lambda_k^{(k)}$. The crucial property that makes bisection powerful is that there eigenvalues *strictly interlace*, i.e.,

$$\lambda_j^{(k+1)} < \lambda_j^{(k)} < \lambda_{j+1}^{(k+1)} \quad (6)$$

for $k = 1, 2, \dots, m - 1$ and $j = 1, 2, \dots, k - 1$.

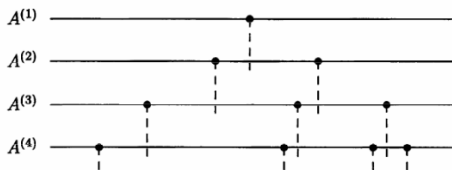


Figure 30.1. Illustration of the strict eigenvalue interlace property (30.6) for the principal submatrices $\{A^{(j)}\}$ of an irreducible tridiagonal real symmetric matrix A . The eigenvalues of $A^{(k)}$ interlace those of $A^{(k+1)}$. The bisection algorithm takes advantage of this property.

It is the interlacing property that makes it possible to count the exact number of eigenvalues of a matrix in a specified interval.

Example 2.1

Consider the 4×4 tridiagonal matrix

$$A = \begin{bmatrix} 1 & 1 & & \\ 1 & 0 & 1 & \\ & 1 & 2 & 1 \\ & & 1 & -1 \end{bmatrix}$$

From the numbers

$$\det(A^{(1)}) = 1, \quad \det(A^{(2)}) = -1, \quad \det(A^{(3)}) = -3, \quad \det(A^{(4)}) = 4$$

Example 2.1

we know that

- ▶ $A^{(1)}$ has no negative eigenvalues,
- ▶ $A^{(2)}$ has one negative eigenvalue,
- ▶ $A^{(3)}$ has one negative eigenvalue,
- ▶ $A^{(4)}$ has two negative eigenvalues.

In general, for any symmetric tridiagonal $A \in \mathbb{R}^{m \times m}$, *the number of negative eigenvalues is equal to the number of sign changes in the sequence*

$$1, \det(A^{(1)}), \det(A^{(2)}), \dots, \det(A^{(m)}), \quad (7)$$

which is known as a *Sturm sequence*.

Example 2.1

By shifting A by a multiple of the identity, we can determine the number of eigenvalues in any interval $[a, b)$. i.e. it is $(\# \text{ of eigenvalues in } (-\infty, b)) - (\# \text{ of eigenvalues in } (-\infty, a))$. One more observation completes the description of the bisection algorithm: for a tridiagonal matrix, the determinants of the matrices $A^{(k)}$ are related by a three-term recurrence relation. Expanding $\det(A^{(k)})$ by minors with respect to its entries b_{k-1} , and a_k in row k gives, from (5),

$$\det(A^{(k)}) = a_k \det(A^{(k-1)}) - b_{k-1}^2 \det(A^{(k-2)}). \quad (8)$$

Introducing the shift by xI and writing $p^{(k)}(x) = \det(A^{(k)} - xI)$, we get

$$p^{(k)}(x) = (a_k - x)p^{(k-1)}(x) - b_{k-1}^2 p^{(k-2)}(x). \quad (9)$$

If we define $p^{(-1)}(x) = 0$ and $p^{(0)}(x) = 1$, then this recurrence is valid for all $k = 1, 2, \dots, m$.

Example 2.1

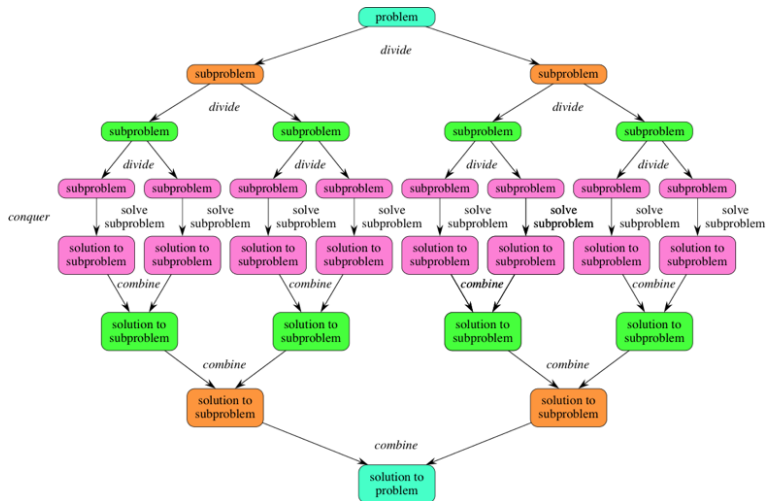
By applying (9) for a succession of values of x and counting sign changes along the way, the bisection algorithm locates eigenvalues in arbitrarily small intervals. The cost is $O(m)$ flops for each evaluation of the sequence, hence $O(m \log(\epsilon_{\text{machine}}))$ flops in total to find an eigenvalue to relative accuracy $\epsilon_{\text{machine}}$. If a small number of eigenvalues are needed, this is a distinct improvement over the $O(m^2)$ operation count for the QR algorithm. On a multiprocessor computer, multiple eigenvalues can be found independently on separate processors.

The divide-and-conquer algorithm, based on a recursive subdivision of a symmetric tridiagonal eigenvalue problem into problems of smaller dimension, represents the most important advance in matrix eigenvalue algorithms since the 1960s. First introduced by Cuppen in 1981, this method is more than twice as fast as the QR algorithm if eigenvectors as well as eigenvalues are required.

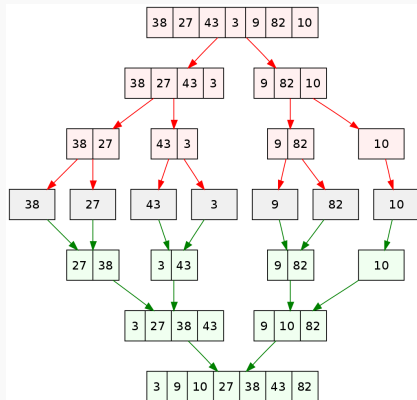
Concept of Divide-and-Conquer

1. Divide : Divide the original problem into smaller sub-problems of similar type.
2. Conquer: Solve each sub-problem recursively. If the sub-problem is small enough, put it as an escape condition and solve it.
3. Combine: Solve the original problem by combining the answers of the sub-problems.

Divide-and-Conquer



Example 3.1



The sorting algorithm is a representative example of using the divide-and-conquer algorithm.

1. Divide the original problem until it is easy enough to compare (until there are 2 remaining).
2. Sort by comparison over small lists.
3. Combine each sorted list.

We shall give just the essential idea, omitting all details.

Let $T \in \mathbb{R}^{m \times m}$ with $m \geq 2$ be symmetric, tridiagonal, and irreducible in the sense of having only nonzeros on the off-diagonal. Then for any n in the range $1 \leq n < m$, T can be split into submatrices as follows:

$$T = \begin{array}{|c|c|} \hline T_1 & \\ \hline \beta & T_2 \\ \hline \end{array} = \begin{array}{|c|c|} \hline \hat{T}_1 & \\ \hline & \hat{T}_2 \\ \hline \end{array} + \begin{array}{|c|c|} \hline & \beta \\ \hline \beta & \beta \\ \hline \end{array} \quad (10)$$

Here T_1 is the upper-left $n \times n$ principal submatrix of T , T_2 is lower-right $(m - n) \times (m - n)$ principal submatrix, and $\beta = t_{n+1,n} = t_{n,n+1} \neq 0$. The modification from T_1, T_2 to \hat{T}_1, \hat{T}_2 is introduced to make the rightmost matrix of (10) have rank 1.

Here is how (10) might be expressed in words.

A tridiagonal matrix can be written as the sum of a 2×2 block-diagonal matrix with tridiagonal blocks and a rank-one correction.

The divide-and-conquer algorithm proceeds as follows.

1. Split the matrix T as in (10) with $n \approx m/2$.
2. Since the correction matrix is of rank one, a nonlinear but rapid calculation can be used to get from the eigenvalues of \hat{T}_1 and \hat{T}_2 to those of T itself.
3. Now recurse on this idea, finding the eigenvalues of \hat{T}_1 and \hat{T}_2 by further subdivisions with rank-one corrections, and so on.

In this process there is one key mathematical point.

How to find eigenvalues of T from eigenvalues of \hat{T}_1 and \hat{T}_2 ?

Suppose that diagonalizations have been computed:

$$\hat{T}_1 = Q_1 D_1 Q_1^T, \quad \hat{T}_2 = Q_2 D_2 Q_2^T.$$

Then from (10),

$$T = \begin{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix} \left(\begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} + \beta z z^T \right) \begin{bmatrix} Q_1^T & \\ & Q_2^T \end{bmatrix} \quad \text{with } z^T = [q_1^T, q_2^T]. \quad (11)$$

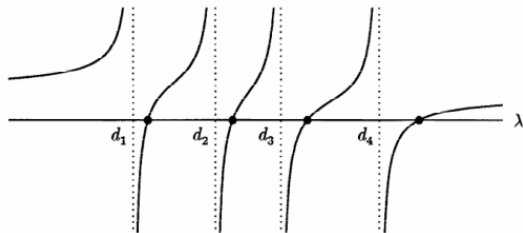
where q_1^T is the last row of Q_1 and q_2^T is the first row of Q_2 .

Since this equation is a similarity transformation, we have reduced the mathematical problem to the problem of finding the eigenvalues of a diagonal matrix plus a rank-one correction.

Suppose we wish to find the eigenvalues of $D + ww^T$, where $D \in \mathbb{R}^{m \times m}$ is a diagonal matrix with distinct diagonal entries $\{d_j\}$ and $w \in \mathbb{R}^m$ is a vector.

We can assume $w_j \neq 0$ for all j , for otherwise, the problem is reducible. Then the eigenvalues of $D + ww^T$ are the roots of the rational function $f(\lambda)$ as illustrated in Figure.

$$f(\lambda) = 1 + \sum_{j=1}^m \frac{w_j^2}{d_j - \lambda}. \quad (12)$$




The equation $f(\lambda) = 0$ is known as the *secular equation*.

$$\begin{aligned}(D + ww^T)q &= \lambda q \Rightarrow (D - \lambda I)q + w(w^T q) = 0 \\&\Rightarrow q + (D - \lambda I)^{-1}w(w^T q) = 0 \\&\Rightarrow w^T q + w^T (D - \lambda I)^{-1}w(w^T q) = 0 \\&\Rightarrow (f(\lambda)(w^T q) = 0 \text{ iff } w^T q \neq 0) \\&\therefore q \text{ is eigenvector of } D + ww^T \text{ with eigenvalue } \lambda, \text{ then } f(\lambda) \text{ must be } 0.\end{aligned}$$



- [1] Nicholas J Higham. *Functions of matrices: theory and computation*. SIAM, 2008.
- [2] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.
- [3] Gilbert Strang. *Linear algebra and learning from data*. Wellesley-Cambridge Press, 2019.
- [4] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*. Vol. 50. Siam, 1997.

The background features a large, faint watermark of the Pusan National University logo. The logo is circular, with the text "PUSAN NATIONAL UNIVERSITY" at the top and "TRUTH LIBERTY DEVOTION" at the bottom. In the center is a shield-shaped emblem with a crown on top and stylized Korean characters inside.

Thank you!