The Jacobi Algorithm

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

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

where

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

- Iteratively apply transformation to 2 rows and 2 columns of $A \in \mathbb{R}^{m \times m}$
- Loop over all pairs of rows/columns, quadratic convergence
- $O(m^2)$ steps, $O(m)$ operations per step $\implies O(m^3)$ operation count
The Method of Bisection

- Idea: Search the real line for roots of \( p(x) = \det(A - xI) \)
- Finding roots from coefficients highly unstable, but \( p(x) \) could be computed by elimination
- Important property: Eigenvalues of principal upper-left square submatrices \( A^{(1)}, \ldots, A^{(m)} \) interlace

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

\[
\begin{array}{cccc}
A^{(1)} & & & \\
& A^{(2)} & & \\
& & A^{(3)} & \\
& & & A^{(4)}
\end{array}
\]

The Method of Bisection

- Because of the interlacing property: The number of negative eigenvalues of \( A \) equals the number of sign changes in the Sturm sequence

\[1, \det(A^{(1)}), \det(A^{(2)}), \ldots, \det(A^{(m)})\]

- Shift \( A \) to get number of eigenvalues in \((-\infty, b)\) and twice for \([a, b)\)
- Three-term recurrence for the determinants:

\[
\det(A^{(k)}) = a_k \det(A^{(k-1)}) - b_{k-1}^2 \det(A^{(k-2)})
\]

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

\[
p^{(k)}(x) = (a_k - x)p^{(k-1)}(x) - b_{k-1}^2 p^{(k-2)}(x)
\]

- \( O(m\log(\epsilon_{\text{machine}})) \) flops per eigenvalue, always high relative accuracy
The Divide-and-Conquer Algorithm

- Split symmetric tridiagonal $T$ into submatrices:

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

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

The Divide-and-Conquer Algorithm

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

$$
T = \begin{bmatrix}
Q_1 \\
Q_2
\end{bmatrix} \left( \begin{bmatrix}
D_1 & 0 \\
0 & D_2
\end{bmatrix} + \beta zz^T \right) \begin{bmatrix}
Q_1^T \\
Q_2^T
\end{bmatrix}
$$

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 a similarity transformation $\implies$ Find eigenvalues of diagonal matrix plus rank-one correction
The Divide-and-Conquer Algorithm

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

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

- Solve the secular equation $f(\lambda) = 0$ with nonlinear solver

- $O(m)$ flops per root, $O(m^2)$ flops for all roots

- Total cost for divide-and-conquer algorithm:

$$O \left( m^2 + 2 \frac{m^2}{2^2} + 4 \frac{m^2}{4^2} + 8 \frac{m^2}{8^2} + \cdots + m \frac{m^2}{m^2} \right) = O(m^2)$$

- For computing eigenvalues only, most of the operations are spent in the tridiagonal reduction, and the constant in “Phase 2” is not important

- However, for computing eigenvectors, divide-and-conquer reduces Phase 2 to $4m^3/3$ flops compared to $6m^3$ for QR