Matrix Theory
Eigenvalue Computation

Kotha Pratheek Reddy
ai24btech11019

1 Algorithms

The chosen algorithm is a combination of the **Householder transformation** and the **QR algorithm**, which computes all eigenvalues of a given matrix guaranteed to have real eigenvalues.

1.1 QR Algorithm

- This algorithm iteratively transforms a matrix into another similar matrix by factoring it into a product of an orthogonal matrix (Q) and an upper triangular matrix (R). The original matrix is updated as the product (RQ):

$$\begin{aligned} A_0 &= Q_0R_0, \\ A_1 &= R_0Q_0, \\ A_1 &= Q_1R_1, \\ A_2 &= R_1Q_1, \\ &\vdots \\ A_i &= R_{i-1}Q_{i-1}, \\ A_i &= Q_iR_i. \end{aligned}$$

- Since Q is orthogonal, Q^{-1} is replaced with $Q^\top$. The iterative process becomes:

$$A_{j+1} = Q_j^\top A_j Q_j.$$

  The eigenvalues of A_j remain the same as those of the original matrix due to similarity.

- Q is computed using the Gram-Schmidt process, where the projections of preceding vectors are removed from each column vector. Each column is then normalized so that the norm of the column is 1.

- The iteration continues until A_j converges to a triangular matrix, with diagonal entries representing the eigenvalues. The time complexity of this method is $O(k \cdot n^3)$, where k is the number of iterations and n is the matrix size.

1.2 Householder Transformation

- The QR algorithm is particularly efficient for Hessenberg matrices. A Hessenberg matrix has zeros either below (lower Hessenberg) or above (upper Hessenberg) the first sub-diagonal.
- For larger matrices, efficiency is improved by using Householder reflections (with $O(n^3)$ complexity) to transform the matrix into a Hessenberg matrix. This significantly reduces the number of iterations in the QR algorithm.
- A Householder transformation reflects a vector $\mathbf{v_1}$ to another vector $\mathbf{v_2}$ (both with the same magnitude). If $\mathbf{u}$ is the unit vector along $\mathbf{v_1} - \mathbf{v_2}$, the reflection matrix is:
$$Q = I - 2\mathbf{u}\mathbf{u}^{\mathbf{T}}.$$
- Q is symmetric and orthogonal, so $Q^{-1} = Q$.
- At each step, $\mathbf{v_1}$ is chosen as a column vector of the matrix, and $\mathbf{v_2}$ is a vector with the first element equal to $||\mathbf{v_1}||$ and the rest set to 0. The resulting Q is extended to the size of the original matrix by adding zeros and ones as needed.
- The update is performed iteratively:
$$A_1 = QA_0Q, \quad A_0 := A_1.$$
- This transformation produces a Hessenberg matrix similar to the original matrix.
- Once the matrix is in Hessenberg form, the QR algorithm is applied to compute eigenvalues. For large matrices, the sparsity of the Hessenberg form accelerates convergence.
- The C code for this algorithm is available at: https://github.com/Pratheek39/EE1030/blob/main/QR/main.c

2 Time Complexity Analysis

The Householder transformation has a time complexity of $O(n^3)$, and the QR algorithm has $O(k \cdot n^3)$ complexity. Combining the two reduces the number of operations per QR iteration and accelerates convergence by minimizing k.

3 Convergence Rate and Space Complexity

The convergence rate of the QR algorithm is linear. The convergence rate depends on the ratio of the two largest eigenvalues. For eigenvalues:

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|,$$

the convergence rate is proportional to $\frac{|\lambda_2|}{|\lambda_1|}$. The space complexity is $O(n^2)$ for both parts of the process.

4 Comparison with Other Algorithms

- **Jacobi Method**: Uses Givens rotations to zero out off-diagonal elements. It is efficient for symmetric matrices but slower than the QR algorithm for larger matrices.
- **Power Iteration**: Simple and computationally inexpensive but finds only the largest eigenvalue. The QR algorithm finds all eigenvalues and converges faster.
- **Divide and Conquer Algorithms**: Recursively divide a matrix and merge results. They are efficient for symmetric matrices but complex to implement and have higher space complexity.
- Other algorithms, like the Lanczos algorithm, are simpler and cheaper but only compute a subset of eigenvalues.

5 Conclusion

To compute all eigenvalues of a matrix guaranteed to have real eigenvalues, the QR algorithm combined with the Householder transformation is efficient, stable, and particularly suitable for dense matrices.