Software Assignment

EE224BTECH11044 - Muthyala koushik

1 Introduction

Given a vector A, an eigenvalue λ such that there exists an non-zero vector x(eigenvector),

$$Ax = \lambda x$$

To find λ(eigenvalues) of matrix by solving the characteristic equation

$$\det(A - \lambda I)$$

2 Chosen Algorithm: QR Algorithm

The QR algorithm is an iterative method for computing the eigenvalues of a matrix. A matrix A is converted into a product of Orthogonal Matrix Q and an upper triangular matrix R and then transforming the matrix iteratively to converge into triangular matrix. And the diagonal elements are eigenvalues.

$$A = QR, A_1 = RQ$$

$A_1 = RQ, R = Q^{-1}$ gives $A_1 = Q^{-1}AQ$ we say that A and A_1 are similar matrices and have same eigenvalues

$$Ax = \lambda x$$

lets put A_1 insted of A

$$Q^{-1}AQx = \lambda x$$

$$AQx = \lambda Qx, Qx = y$$

$$Ay = \lambda y$$

so,we now know that A and A_1 has same eigen values, repeatedly applying QR decomposition to RQ eventually leads to an upper triangular matrix

Each transformation $A_k = R_kQ_k$ shifts the non-diagonal elements of A_k closer to zero, resulting in a matrix that looks more and more like an upper triangular matrix after several iterations.
The diagonal elements of this upper triangular matrix are the eigenvalues of the original matrix A.

2.1 Algorithm Description

The QR algorithm follows these steps:

1. Perform QR decomposition on A to obtain Q and R.
2. Update A using $A_1 = RQ$.
3. Check for convergence by monitoring the off-diagonal elements.
4. Repeat until the off-diagonal elements are below a specified tolerance or a maximum number of iterations is reached.

3 Time Complexity Analysis

The QR algorithm for eigenvalue computation has a time complexity of $O(n^3)$ per iteration, where n is the dimension of the matrix. This complexity arises to perform QR decomposition, involves matrix multiplication and Gram-Schmidt orthogonalization.

4 Other Insights

4.1 Memory Usage

The QR algorithm requires $O(n^2)$ space to store matrices Q, R, A, and intermediate results. This memory usage is reasonable for moderately sized matrices but may become a limitation for very large matrices.

4.2 Convergence Rate

The convergence rate of the QR algorithm depends on the matrix. It converges faster for symmetric matrix. For matrices with closely spaced or degenerate eigenvalues, the convergence can be slow.

4.3 Suitability for Different Matrices

The QR algorithm is particularly well-suited for dense, symmetric matrices. For Asymmetric matrices the algorithm converge more slowly can improve using Hessenberg form

5 Comparison of Algorithms

5.1 Power Iteration

The Power Iteration method is a simple algorithm for finding the largest eigenvalue of a matrix. It has a time complexity of $O(n^2)$ per iteration, making it efficient for large matrices but unsuitable for computing all eigenvalues. It also converges slowly,

especially when the largest eigenvalue is only slightly greater in magnitude than the others.

5.2 Lanczos Algorithm

The Lanczos algorithm is an efficient method for computing eigenvalues of large, sparse, symmetric matrices. It reduces the problem to a smaller tridiagonal matrix, significantly improving efficiency. The time complexity is $O(kn^2)$, where k is the number of iterations. Primarily designed for symmetric matrices.

5.3 Jacobi Method

High accuracy for symmetric matrices. The method converges slowly for non-symmetric matrices.It has a time complexity of $O(n^3)$ per iteration. Generally slower than other methods for non-symmetric or large matrices.

5.4 Comparison Summary

- **Time Complexity**: The QR algorithm has $O(n^3)$ complexity, while Power Iteration is $O(n^2)$, Jacobi is $O(n^3)$ and Lanczos is $O(kn^2)$.
- **Accuracy**: The QR algorithm is accurate for all eigenvalues but may require many iterations. Power Iteration is accurate only for the largest eigenvalue, Lanczos is efficient and accurate for sparse matrices , while Jacobi provides very accurate results for symmetric matrices.
- **Suitability**: The QR algorithm is best for dense, symmetric matrices. Power Iteration is suitable for finding a single dominant eigenvalue, Jacobi is preferred for symmetric matrices and Lanczos is preferred for large, sparse systems.

6 Conclusion

The QR algorithm is widely used method for eigenvalue computation, especially for dense matrices. However, for large and sparse matrices, more specialized algorithms like Lanczos provide better efficiency and performance. Understanding the properties of the matrix is crucial in selecting the appropriate method.