Eigenvalue Calculation

EE1030 : Matrix Theory

Satyanarayana Gajjarapu
AI24BTECH11009
ai24btech11009@iith.ac.in

November 17, 2024

1 Eigenvalues & Eigenvectors

1.1 Definition

Eigenvalues are set of scalars associated with system of linear equations. Consider a square matrix A of dimension $n \times n$, a non-zero vector $\mathbf{v}$ of length n. If

$$A\mathbf{v} = \lambda\mathbf{v}$$

where λ is a scalar, then $\mathbf{v}$ is called an eigenvector of A and λ is the corresponding eigenvalue.

1.2 Finding Eigenvalues

Since $\mathbf{v}$ is a non-zero vector ($\det \mathbf{v} \neq 0$)

$$\begin{aligned} A\mathbf{v} &= \lambda\mathbf{v} \\ (A - \lambda I)\mathbf{v} &= 0 \\ \det((A - \lambda I)\mathbf{v}) &= 0 \\ \det(A - \lambda I) &= 0 \end{aligned}$$

where I is the identity matrix. The above equation is called the characteristic equation or eigen equation. The eigenvalues of a matrix are the roots of λ in this equation.

2 QR Algorithm

The algorithm chosen to solve for eigenvalues of a matrix is **QR Algorithm**.

2.1 QR decomposition process

In the algorithm, a matrix A is decomposed into product of two matrices Q and R using Gram-Schmidt process, where

• Q is an orthogonal matrix ($Q^TQ = I$) in real space or an unitary matrix ($Q^*Q = I$) in

complex space.
• R is an upper triangular matrix.

Let $A = [\mathbf{a}_1 \quad \mathbf{a}_2 \quad \cdots \quad \mathbf{a}_n]$, where $\mathbf{a}_1, \cdots, \mathbf{a}_n$ are column matrices of length n. When Gram-Schmidt process is applied to A

$$Q = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{bmatrix}, \quad R = \begin{bmatrix} \langle \mathbf{q}_1, \mathbf{a}_1 \rangle & \langle \mathbf{q}_1, \mathbf{a}_2 \rangle & \cdots & \langle \mathbf{q}_1, \mathbf{a}_n \rangle \\ 0 & \langle \mathbf{q}_2, \mathbf{a}_2 \rangle & \cdots & \langle \mathbf{q}_2, \mathbf{a}_n \rangle \\ 0 & 0 & \cdots & \langle \mathbf{q}_3, \mathbf{a}_n \rangle \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \langle \mathbf{q}_n, \mathbf{a}_n \rangle \end{bmatrix}$$

$$\mathbf{u}_k = \mathbf{a}_k - \sum_{j=1}^{k-1} \text{proj}_{\mathbf{u}_j} \mathbf{a}_k, \quad \mathbf{q}_k = \frac{\mathbf{u}_k}{||\mathbf{u}_k||}$$

$$\text{proj}_{\mathbf{u}} \mathbf{a} = \frac{\langle \mathbf{u}, \mathbf{a} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u} \quad \text{and} \quad \langle \mathbf{v}, \mathbf{w} \rangle \text{ is the inner product, } \mathbf{v}^T \mathbf{w}$$

2.2 Algorithm Description

After decomposition of the matrix A into Q and R, A is updated as the product RQ. This process repeated iteratively until the updated matrix A becomes a diagonal matrix or a upper triangular matrix, where the elements below the diagonal become zero. The diagonal elements of the resulting matrix are the eigenvalues of matrix A. This iterative process can be expressed by

$$A_k = Q_k R_k$$
$$A_{k+1} = R_k Q_k$$

A_k represents the matrix after k-th iteration.
• The final matrix obtained will be diagonal matrix, if the matrix A is diagonalizable.
• The final matrix obtained will be upper triangular matrix of Schur form, if A is non-diagonalizable.

2.3 Analysis of the algorithm

2.3.1 Time Complexity

In the matrix multiplication function, the resultant matrix C contains n^2 terms and each of them requires $O(n)$ operations. Therefore the total number of operations are $O(n) \times O(n^2) = O(n^3)$, resulting the time complexity of this function to be $O(n^3)$.

In QR decomposition function, calculating inner product has $O(n)$ operations, subtracting the projections also has $O(n)$ operations. Therefore each column construction has $O(n^2)$ operations and normalization for each column involve $O(n)$ operations. So, the process of orthogonalization has time complexity of $O(n^3)$. For computing R, each inner product calculation requires $O(n)$ operations and to compute it for n^2 elements the time complexity becomes $O(n^3)$.

The total time complexity of QR decomposition function is

$$O(n^3) + O(n^3) = O(n^3)$$

In QR algorithm function, each iteration has matrix multiplication and QR decomposition. So, for one iteration time complexity is $O(n^3) + O(n^3) = O(n^3)$. Let the number of iterations before convergence of matrix A is P, then the time complexity of this function is $O(P \cdot n^3)$.

The time complexity of the main function is negligible compared to $O(P \cdot n^3)$. Therefore overall time complexity is approximated to $O(P \cdot n^3)$, where P depends on convergence properties of the matrix.

2.3.2 Memory Usage

The memory used for

1. the matrix A is $n \times n \times 16 = 16n^2$
2. the matrix Q is $n \times n \times 16 = 16n^2$
3. the matrix R is $n \times n \times 16 = 16n^2$
4. eigenvalues array is $n \times 16 = 16n$

Therefore, the total memory used is $48n^2 + 16n$.

2.3.3 Convergence Rate

The algorithm has

1. fast convergence if matrix type is diagonal or symmetric
2. moderate convergence for normal diagonalizable and non-symmetric matrices
3. slow convergence for defective and large matrices.

2.3.4 Suitability

This algorithm is suitable mostly to all types of matrices except defective matrices and certain sparse matrices.

2.4 Merits

1. This algorithm gives all eigenvalues of a matrix.
2. It can handle matrices with complex entries and output complex eigenvalues.
3. It is easy to implement and has a good numerical stability for most matrices.
4. It is flexible because it can handle matrices of different sizes.

2.5 Demerits

1. It is inefficient for large matrices.
2. Convergence condition is inaccurate for some matrices.
3. In case of defective matrices the algorithm may fail to converge.

3 Comparison with other Algorithms

Comparison of QR algorithm with other well known algorithms like Power Iteration algorithm, Jacobi eigenvalue algorithm and Arnoldi Iteration.

3.1 Time Complexity

Algorithm	**Time Complexity**
QR Algorithm	$O(n^3)$ per iteration
Power Iteration	$O(n^2)$ per iteration
Jacobi's Method	$O(n^3)$
Arnoldi Iteration	$O(n^2)$ per iteration

3.2 Output

Algorithm	**Output**
QR Algorithm	All eigenvalues
Power Iteration	Largest eigenvalue
Jacobi's Method	All eigenvalues
Arnoldi Iteration	Subset of eigenvalues

3.3 Accuracy

• QR algorithm is highly accurate for almost all types of matrices except certain sparse matrices.
• Power iteration algorithm is accurate for almost all types of matrices but gives only largest eigenvalue.
• Jacobi's method is highly accurate for only symmetrical matrices.
• Arnoldi iteration is accurate for hermition and sparse matrices.