Eigenvalue Calculation - Report

Manognya Kundarapu - EE24BTECH11037

1) **Introduction:** The Eigenvalues are the special set of scalars associated with the system of linear equations. For a square matrix A, an eigenvalue λ is a scalar that satisfies the equation:
   $A\mathbf{v} = \lambda\mathbf{v}$, $\mathbf{v}$ is a non-zero vector called the eigenvector associated with λ.
   Some of the key applications of eigenvalues:
   a) In dynamical systems, eigenvalues determine stability. Used in control systems and vibration analysis .
   b) These are critical for diagonalizing matrices.
   c) In data science and machine learning, eigenvalues help identify the most significant features of data.
2) **Chosen Algorithm:** The **QR algorithm** is a powerful numerical method for finding the eigenvalues of a square matrix. It relies on QR decomposition and QR iteration.
   QR decomposition:
   It is a matrix factorization method that expresses a given square matrix A as:
   $A = QR$
   Q is an orthogonal matrix, R is an upper triangular matrix. The decomposition can be composed using
   →Gram-Schmidt orthogonalization.
   →Householder reflections
   →Givens rotations
   QR Iteration Algorithm:
   It basically applies the QR decomposition iteratively. Algorithm steps:
   i) Start with $A_0 = A$, A is input matrix.
   ii) Compute $A_k = Q_k R_k$, the QR decomposition, from the k_{th} step.
   iii) Compute the next iterate as $A_{K+1} = R_k Q_k$. Continue the iteration until A_k converges to a form where its diagonal elements approximate the eigenvalues of A.

   Convergence:
   →The QR algorithm typically converges to a form where the matrix A_k becomes upper triangular, with its diagonal elements representing the eigenvalues of A.
   →The convergence rate depends on the separation of the eigenvalues of A.
   Shifts and Improved Convergence:
   The basic QR algorithm can be slow, especially for matrices with closely spaced eigenvalues. Shifts are used to accelerate convergence.
   A **shift** modifies the QR iteration to focus on a specific eigenvalue. Instead of working with A_k, a shifted matrix $A_k - \mu I$ is used, where μ is the shift.
   Shifted QR Algorithm:
   →Choose a shift: Select μ, typically an eigenvalue estimate (e.g., the bottom-right entry of A_k).

→Shift the matrix: Compute the QR decomposition of $A_k - \mu I$ as $(A_k - \mu I) = Q_k R_k$.
→Update the matrix: Form the next iterate:
$A_{k+1} = R_k Q_k + \mu I$
→Repeat: Iterate until convergence.
How shifts improve convergence:
i) Better eigenvalue separation: By focusing the iteration on specific eigenvalues, shifts help the algorithm converge faster for poorly separated eigenvalues.
ii) Reduction of subdominant terms: Shifts minimize the contribution of less significant eigenvalue terms in the iterations, accelerating convergence.
iii) Wilkinson shift: A common choice for the shift μ is based on a more refined eigenvalue estimate, such as the eigenvalues of the $2x2$ trailing principal submatrix of A_k.
The QR algorithm iteratively decomposes a matrix and reassembles it to converge to its eigenvalues. Shifts significantly improve the convergence by targeting specific eigenvalues and reducing computational overhead, making the method both robust and efficient for a wide range of applications in numerical linear algebra.
PROS: Suitable for finding all eigenvalues of a matrix, often converges faster, also handles the complex values.
CONS: Requires some other transformations to improve efficiency and implementation is often complex.

3) **Time complexity:** The QR algorithm is $O\left(n^3\right)$ for each iteration without optimization. Despite its theorotical time complexity of $O\left(n^3\right)$, it becomes faster when enhanced with shifts. Steps:

   → QR decomposition: The QR decomposition itself requires $O(n^3)$ for a general dense matrix using algorithms like Householder reflections.
   → Matrix multiplication: The subsequent $R_k Q_k$ multiplication also requires $O(n^3)$

   →Thus, each iteration is $O(n^3)$, and the total complexity depends on the number of iterations required for convergence, which is typically manageable for well-behaved matrices. With shifts, the number of iterations can often be drastically reduced.
   →The QR algorithm, particularly with shifts, provides highly accurate eigenvalues for symmetric and general matrices. This is essential when precision is critical.
4) **Comparison with other algorithms:**
   → **Power and Inverse Iteration Methods,** $O\left(n^2\right)$**:** These are faster but provide only one eigenvalue at a time and require a good initial guess.
   → **Jacobi Method** $O(n^4)$**:** Simpler to implement but slower for larger matrices.
   →**Divide and Conquer** $O(n^3)$**:** Similar complexity but less straightforward and not as widely implemented.
   → **Lanczos and Arnoldi** $O(k \cdot n^2)$**:** Effective for sparse matrices or approximating a subset of eigenvalues, but not as general-purpose.
   Why QR Algorithm?
   (a) Efficiency for Moderate Matrix Sizes:
   For matrices with moderate dimensions (e.g., $n \leq 1000$), the$O(n^3)$ complexity is

feasible on modern computational systems. Many real-world applications fall within this range.

(b) Accuracy:
The QR algorithm, particularly with shifts, provides highly accurate eigenvalues for symmetric and general matrices. This is essential when precision is critical.

(c) General Applicability:
The QR algorithm works for a wide range of matrices, making it a robust choice for assignments where the matrix structure (e.g., symmetry, sparsity) might not be guaranteed.

Therefore, QR Algorithm achieves a favorable balance between computational effort and accuracy, justifying its use in this context.