Matrix Theory
Eigenvalue Computation

Kotha Pratheek Reddy
ai24btech11019

---

1 Algorithms

The chosen algorithm is a combination of **Householder transformation** and **QR algorithm**, which can compute all eigenvalues of a given matrix with only real eigenvalues.

1.1 QR Algorithm

- This algorithm iteratively transforms a matrix into another matrix similar to it by factoring it into a product of an orthogonal matrix (Q) and an upper triangular matrix (R), then updating the original matrix to the product (RQ).

$$\begin{aligned} A_0 &= Q_0 R_0 \\ A_1 &= R_0 Q_0 \\ A_1 &= Q_2 R_1 \\ A_2 &= R_1 Q_1 \\ &\vdots \\ A_i &= R_{i-1} Q_{i-1} \\ A_i &= Q_i R_i \end{aligned}$$

$$A_{j+i} = Q_j^{-1} A_j Q_j$$

  Since Q is an orthogonal matrix, Q^{-1} can be replaced with $Q^\top$

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

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

- The values of Q are found through the Gram-Schmidt process, where projections of preceding vectors are removed from each vector in the matrix. Each vector is then normalized such that the norm of each column is set to 1.

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

1.2 Householder Transformation.

- The QR algorithm works particularly well with Hessenberg matrices. Matrices with all entries below the first sub-diagonal as zero are called lower Hessenberg matrices, while those with zeros above the first upper sub-diagonal are called upper Hessenberg matrices. The QR algorithm also works with other matrices, but the computation becomes more expensive.

- For larger matrices, efficiency can be improved by using Householder reflections, which have a complexity near $O(n^3)$, to transform the matrix into a lower Hessenberg matrix. The QR algorithm can then be applied to this transformed matrix. This significantly reduces the number of iterations required when working with densely filled matrices.

- Householder transformation involves finding a matrix (symmetric and orthogonal) that, when multiplied by a vector, reflects it to another desired vector. For example, if a vector $\mathbf{v_1}$ is to be rotated to $\mathbf{v_2}$ ($\mathbf{v1}$ and $\mathbf{v2}$ must have the same magnitude), and $\mathbf{u}$ is the unit vector along $\mathbf{v_1} - \mathbf{v_2}$, the reflection is carried out by pre-multiplying $\mathbf{v1}$ with $Q = I - \mathbf{2uu^T}$. The Q obtained here is both orthogonal and symmetric. Thus, its inverse is the same as itself.

- At each step, $\mathbf{v_1}$ corresponds to a column of the given matrix, and $\mathbf{v_2}$ is a vector with the first element equal to $||\mathbf{v_1}||$ and the others set to zero. The resulting Q is then extended to match the size of the original matrix by adding zeros in off-diagonal entries and ones in the diagonal entries.

- Let Q be the extended version of the Householder matrix for a certain row. The update is performed iteratively n times as follows:

$$A_1 = QA_0Q$$
$$A_0 := A_1$$

  This is done because $Q = Q^{-1}$.

- In this way, the matrix remains similar to the original matrix, with the elements below the first sub-diagonal set to zero.

- Once the matrix is in Hessenberg form, the QR algorithm can be applied to compute eigenvalues. For larger matrices, the Hessenberg form is much sparser, which accelerates convergence.

2 Time Complexity Analysis

The time complexity of the Householder transform is $O(n^3)$, while the QR algorithm has a time complexity of $O(kn^3)$, where k is the number of iterations. Combining these two methods reduces the operations per QR iteration and accelerates convergence (reduces k).

3 Convergence Rate and Space Complexity

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

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

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

4 Comparison with Other Algorithms

- **Jacobi Method**: This method uses Givens rotations to zero out the off-diagonal elements. It is well-suited for symmetric matrices. However, its convergence is slower than that of the QR algorithm, and it becomes computationally expensive for larger matrices. The QR algorithm also handles non-symmetric matrices better.
- **Power Iterations**: This algorithm is simple and computationally inexpensive but finds only the largest eigenvalue. In contrast, the QR algorithm computes all eigenvalues and converges faster.
- **Divide and Conquer Algorithms**: These algorithms use recursion to divide a matrix into smaller matrices and then merge results. They are efficient for symmetric matrices but are generally more complex to implement and have higher space complexity.
- Several other algorithms, like the Lanczos algorithm, exist and are generally simpler and less expensive than the QR algorithm. However, they are suitable only for finding one or a few eigenvalues, not all.

5 Conclusion

To compute all eigenvalues of a matrix guaranteed to have real eigenvalues, the QR algorithm, combined with the Householder transformation to reduce the matrix to Hessenberg form, is generally more efficient and stable. It is particularly effective for dense matrices, where it outperforms other algorithms.