A Study about QR Algorithm Using Givens Rotations

EE24BTECH11053 - S A Aravind Eswar

November 18, 2024

Contents

1 Abstract

Matrices are an important tool for for performings calculations in a variety of applications. And the eigenvalues of the matrices have a wide variety of applications from AI and ML to Molecular Physics. There are multiple algorithms to compute the eigenvalues of a given matix. We shall look into the QR Algorithm using Givens Rotations and discuss about it's pros and cons and implement the algorithms.

2 Introduction

Matrices are an important tool for for performings calculations in a variety of applications. And the eigenvalues of the matrices have a wide variety of applications from AI and ML to Molecular Physics. There are multiple algorithms to compute the eigenvalues of a given matix. The most widely used algorithm for computing the eigenvalues is the QR algorithm or QR iteration algorithm. Other widely used algorithms include the Power interation algorithm, Inverse interation algorithm and so on. To perform the said QR decomposition for the QR algorithm, we are going to look into the method of Given's rotation. There are other algorithms that can perform QR decomposition such as the Gram-Schmidt algorithm and Householder transformations and so on. The reasons of choosing the said algorithms will be discussed in the later sections.

3 Given's Rotations

Givens rotaiton is a rotation in a plane spanned by 2 coordinate axes in a $\mathbb{R}^n$ space. Let a vector $\mathbf{x}$ in $\mathbb{R}^n$ to be rotated in the plane (i, j) by an angle of θ. Then,

$$\mathbf{x_r} = G(i, j, \theta)\mathbf{x} \tag{3.1}$$

is the vector produced after rotation and the matrix $G(i, j, \theta)$ is given by,

$$G(i,j,\theta) = \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix} \tag{3.2}$$

$$G_{kk} = 1 \text{ for } k \neq i, j \tag{3.3}$$

$$G_{kk} = c \text{ for } k = i, j \tag{3.4}$$

$$G_{ij} = -G_{ji} = s \tag{3.5}$$

$$\text{or } 0 \text{ otherwise} \tag{3.6}$$

where, The rotation can be scalted to $\mathbb{C}^n$ by replacing G_{jj} with $\bar{c}$ and G_{ji} with $-\bar{s}$

3.1 Performing QR decomposition with Givens Rotation

Let A be a $n \times n$ matrix in $\mathbb{C}$. We can use Givens rotation to make exactly one of the coordinates of $\mathbf{A}$ as 0.
Then,

$$A = \left(A_{ij}\right), A_{ij} \in \mathbb{C} \tag{3.7}$$

$$\tag{3.8}$$

Let,

$$\mathbf{G}(p,q) = \begin{pmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & c & s & \cdots & 0 \\ 0 & \cdots & -s & c & \cdots & 0 \\ \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{3.9}$$

$$G_{kk} = 1 \text{ for } k \neq i, j \tag{3.10}$$

$$G_{p,p} = \overline{G}_{p-1,p-1} = c \tag{3.11}$$

$$\overline{G}_{p-1,p} = -G_{p,p-1} = s \tag{3.12}$$

$$\text{or } 0 \text{ otherwise} \tag{3.13}$$

where,

$$r = \sqrt{\mathbf{A}_{p,q}^2 + \mathbf{A}_{p-1,q}^2} \tag{3.14}$$

$$c = \mathbf{A}_{p,q}/r \tag{3.15}$$

$$s = \mathbf{A}_{p-1,q}/r \tag{3.16}$$

Then,

$$\mathbf{A_1} = \mathbf{G}(p,q)\mathbf{A} \tag{3.17}$$

will make $A_{p,q}$ zero.
Performing another rotation on $\mathbf{A_{1,1}}$ and so on,

$$\mathbf{G}(1,2)\mathbf{G}(1,3)\cdots\mathbf{G}(n,n-1)\mathbf{A} = \mathbf{R} \tag{3.18}$$

where R is an Upper Triangular Matrix. Then let,

$$\mathbf{Q}^\top = \mathbf{G}(1,2)\mathbf{G}(1,3)\cdots\mathbf{G}(n,n-1) \tag{3.19}$$

$$\mathbf{Q} = (\mathbf{G}(1,2)\mathbf{G}(1,3)\cdots\mathbf{G}(n,n-1))^\top \tag{3.20}$$

$$\mathbf{Q} = \mathbf{G}^\top(n,n-1)\mathbf{G}^\top(n,n-1)\cdots\mathbf{G}^\top(1,2) \tag{3.21}$$

Then,

$$\mathbf{A} = \mathbf{QR} \tag{3.22}$$

3.2 Analysis of the Givens algorithm

Givens rotation of a matrix has a property where it only modifies only 2 of the columns of a matrix. This makes it more numerically stable, unlike Gram-Schmidt which is not very stable numerically. This also comes with a perk of being more parallaizable than other algorithms like Householder transformations.

Givens Algorithm is not too efficient for a dense matrix as it doesn't have many zero entries to skip over. But it is more efficient when it comes to sparse matrix when compared to Householder, which is more efficient when it comes to dense matrix.

The time complexity of the Givens algorithm is $O(n^3)$, which is similar to other algorithms for QR decomposition.

3.3 Algorithm

Let $\mathbf{A}_{m\times m}$
$\mathbf{Q} \leftarrow \mathbf{I}_{n\times n}$
$\mathbf{R} \leftarrow \mathbf{A}$
for $i = 0, 1, \ldots n-1$ **do**
    for $j = n-1, n-2, \ldots, i+1$ **do**
        if $|\mathbf{R}(j,i)| \neq 0$ **then**
            Generate $\mathbf{G}(j,i)$
            $\mathbf{Q} \leftarrow \mathbf{QG}^\top(j,i)$
            $\mathbf{R} \leftarrow \mathbf{G}(j,i)\mathbf{R}$
return $\mathbf{Q}, \mathbf{R}$

4 QR iteration algorithm

QR Iteration algorithm as discussed is a widely used algorithm. It works on the principle of converging a matrix to it's Schur form and extracting eigenvalues from it.

$$\mathbf{A}_{k-1} = \mathbf{QR} \tag{4.1}$$

$$\mathbf{A}_k = \mathbf{RQ} \tag{4.2}$$

$$\mathbf{A}_k = \mathbf{Q}^\top\mathbf{RQ} \tag{4.3}$$

Eventually $\mathbf{A_k}$ will converge to the Schur form of $\mathbf{A}$, where is a $n \times n$ matrix in $\mathbb{C}$

4.1 Working of QR algorithm

QR rotation theoretically is an infinitely iterating algorithm. But for it to converge to machine level precision, it takes it about $O(n^3)$ time to converge to the Schur form. The convergence rate can be accelerated but using techniques like Hessenberg reduction and Shifts.

We shall now discuss about how the QR iteration work mathematically.

Theorem 4.1. *Suppose that A has eigenvalues* $\lambda_1, \ldots, \lambda_n$ *satisfying* $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$. *Let X denote the matrix whose ith column is an eigenvector of A corresponding to* λ_i *and suppose that* X^{-1} *has an LU decomposition. Then the subdiagonal elements of the matrices* A_s *of the basic QR algorithm tend to zero and for* $i = 1, 2, \ldots, n$, $(A_s)_{ii} \to \lambda_i$.

Note that since we are assuming that the eigenvalues are distinct, we know there are a complete set of linearly independent eigenvectors and hence X^{-1} exists.

Without giving a formal proof, let us try to understand what makes this algorithm work. Recall that the QR algorithm sets $A_1 = A$, factors $A_i = Q_i R_i$ and then sets $A_{i+1} = R_i Q_i$. Since $Q_i^{-1} = Q_i^T$, we have $R_i = Q_i^T A_i$ and so

$$A_{i+1} = Q_i^T A_i Q_i.$$

Thus, we have performed a similarity transformation, which preserves the eigenvalues. If we iterate the above equation, we get

$$A_{i+1} = Q_i^T A_i Q_i = Q_i^T Q_{i-1}^T A_{i-1} Q_{i-1} Q_i = \ldots Q_i^T \ldots Q_1^T A_1 Q_1 \ldots Q_i = P_i^T A_1 P_i,$$

where $P_i = Q_1 \ldots Q_i$. Note that P_i is the product of orthogonal matrices and hence is orthogonal.

Now under our assumptions, we can write $A = XDX^{-1}$ where $D = \text{diag}(\lambda_1, \ldots, \lambda_n)$ and X is a real matrix of eigenvectors of A. We know there is a factorization of $X = QR$, where Q is orthogonal and R is upper triangular. Then

$$A = QRDR^{-1}Q^{-1} \text{ and so } Q^{-1}AQ = RDR^{-1}.$$

Since RDR^{-1} is the product of upper triangular matrices, it is also upper triangular, and hence we know that $Q^{-1}AQ = Q^TAQ$ is upper triangular. Note that the eigenvalues of A will then lie on the diagonal of Q^TAQ.

The theorem is proved by showing that $\lim_{i\to\infty} P_i = Q$, since this implies that

$$\lim_{i\to\infty} A_{i+1} = \lim_{i\to\infty} P_i^T A_1 P_i = Q^T A Q.$$

In other words, the matrices A_i are converging to an upper triangular matrix whose diagonal elements are the eigenvalues of A.

To see why $\lim_{i\to\infty} P_i = Q$, we look at the quantity $P_i U_i$, where $U_i = R_i R_{i-1} \ldots R_1$. Then

$$P_i U_i = Q_1 \ldots Q_i R_i \ldots R_1 = Q_1 \ldots Q_{i-1} A_i R_{i-1} \ldots R_1 = P_{i-1} A_i U_{i-1}.$$

But since $A_{i+1} = P_i^T A_1 P_i$, we have $P_i A_{i+1} = A_1 P_i$, or after reducing the indices by one, $P_{i-1} A_i = A_1 P_{i-1}$. Hence,

$$P_i U_i = A_1 P_{i-1} U_{i-1}.$$

If we iterate this identity, we get

$$P_i U_i = (A_1)^{i-1} P_1 U_1 = (A_1)^{i-1} Q_1 R_1 = A_i.$$

Using the fact that $A = XDX^{-1}$, we also have that $A_i = XD^i X^{-1}$. We know that $X = QR$ and by hypothesis $X^{-1} = LU$. Hence,

$$A_i = QRD^i LU = QR(D^i L D^{-i}) D^i U.$$

Equating these two expressions for A_i, we get that

$$P_i U_i = QR(D^i L D^{-i}) D^i U.$$

The key step in the proof is to show that

$$\lim_{i \to \infty} D^i L D^{-i} = I.$$

Assuming for the moment that this is true, the right hand side becomes $QRD^i U$. But $RD^i U$ is the product of upper triangular matrices and is therefore upper triangular. Hence, we are essentially able to identify $\lim_{i\to\infty} P_i$ with Q, since both are orthogonal matrices, and $\lim_{i\to\infty} U_i$ with $\lim_{i\to\infty} RD^i U$ since both quantities are upper triangular matrices. We have ignored a subtle point; namely that the QR factorization is only unique if R is chosen to have positive diagonal entries. So we must choose all the decompositions to insure this property so that we can identify the individual pieces of the decomposition. Returning to the key step, we observe that the matrix $D^i L D^{-i}$ is a lower triangular matrix whose j,kth element is given by $l_{jk}(\lambda_j/\lambda_k)^i$, when $j > k$. Since $|\lambda_j/\lambda_k| < 1$ for $j > k$,

$$\lim_{i \to \infty} D^i L D^{-i} = I.$$

Since we expect that $(A_i)_{nn}$ is converging to an eigenvalue of A, we can choose $s_i = (A_i)_{nn}$. We will then get an iteration similar to the Rayleigh quotient iteration.

The above discussion holds only when A have purely real eigenvalue. If it were to have to complex eigenvalues, it would form 2×2 blocks along the diagonal, whose eigenvalues are complex conjugates of each other and also a subset of the eigenvalues of A. The theorem and proof for this statement is discussed below.

Theorem 4.2. *(Real Schur Decomposition). If $A \in \mathbb{R}^{n\times n}$, then there exists an*

orthogonal $Q \in \mathbb{R}^{n\times n}$ *such that*

$$Q^T A Q = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ 0 & R_{22} & \cdots & R_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_{mm} \end{bmatrix} \tag{7.4.2}$$

where each R_{ii} *is either a 1-by-1 matrix or a 2-by-2 matrix having complex conjugate eigenvalues.*

Proof. The complex eigenvalues of A occur in conjugate pairs since the characteristic polynomial $\det(zI - A)$ has real coefficients. Let k be the number of complex conjugate pairs in $\lambda(A)$. We prove the theorem by induction on k.

Obviously, theorem holds for $k = 0$, as it coverges to upper triangular matrix.

Now suppose that $k \geq 1$. If $\lambda = \gamma + i\mu \in \lambda(A)$ and $\mu = 0$, then there exist vectors y and z in $\mathbb{R}^n$ ($z = 0$) such that $A(y + iz) = (\gamma + i\mu)(y + iz)$, i.e.,

$$A \begin{bmatrix} y & z \end{bmatrix} = \begin{bmatrix} y & z \end{bmatrix} \begin{bmatrix} \gamma & \mu \\ -\mu & \gamma \end{bmatrix}.$$

The assumption that $\mu = 0$ implies that y and z span a 2-dimensional, real invariant subspace for A.

Then we use the result, which we will not prove,

Lemma 4.1. *Let* $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{p\times p}$, *and* $X \in \mathbb{R}^{n\times p}$ *satisfy*

$$AX = XB, \quad rank(X) = p,$$

then there exists a unitary $Q \in \mathbb{R}^{n\times n}$ *such that*

$$Q^H A Q = T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{matrix} p \\ n-p \end{matrix}$$

and $\lambda(T_{11}) = \lambda(A) \cap \lambda(B)$.

to say that an orthogonal $U \in \mathbb{R}^{n\times n}$ exists such that

$$U^T A U = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{matrix} 2 \\ n-2 \end{matrix}$$

where $\lambda(T_{11}) = \{\lambda, \bar{\lambda}\}$. By induction, there exists an orthogonal $\tilde{U}$ so $\tilde{U}^T T_{22} \tilde{U}$ has the required structure. The theorem follows by setting $Q = U \cdot \mathrm{diag}(I_2, \tilde{U})$. □

4.2 Hessenberg reduction

Hessenberg matrix is where all the elements below the lower sub-diagonal is all zero.

We can write,

$$\mathbf{H}_0 = \mathbf{U}_0^\top \mathbf{A}\mathbf{U} \tag{4.4}$$

where $\mathbf{H}_0$ is Hessenberg matrix. We can show that both $\mathbf{H}_0$ and $\mathbf{A}$ and identical and thus have the same eighenvalues.

Performing QR decomposition for the Hessenberg matrix is much faster ($O(n^2)$) as the number of calculations is much less. But Hessenberg transformations can only used when $A_{i,j} \in \mathbb{R}$

4.3 Shifts

Let,

$$\mu \to \mathbb{R} \tag{4.5}$$

$$\mathbf{QR} = \mathbf{A}_{k-1} - \mu\mathbf{I} \tag{4.6}$$

$$\mathbf{A}_k = \mathbf{RQ} + \mu\mathbf{I} \tag{4.7}$$

This is called shifting. There are 2 ways of selecting the shift value μ

4.3.1 Raleigh Shift: If $\mathbf{A}_k$ is a Hessenberg matrix of size $n \times n$, then $\mu = \mathbf{A}_k(n, n)$

4.3.2 Wilkinson Shift: If $\mathbf{A}_k$ is a Hessenberg matrix of size $n \times n$ then μ is determined based on the eigenvalues of bottom right most 2×2 matrix of $\mathbf{A}_k$ (let a_k and b_k). If both a_k and b_k are real, then the smallest of $|a_k - A_k(n, n)|$ and $|b_k - A_k(n, n)|$ is taken. Else if both are conjugate pairs of each other, then $\mu = \mathbb{R}(a_k)$.

Both the above mentioned shifts are used to perform single-shifts. There are otehr types of shifts that can be performed like double-shift, implicit shifts or double-implicit shifts. We shall focus only on single-shifts as of now.

Shifting accelerates the process of convergence of the eigenvalues.

If the $(n, n-1)$ entry converges to zero, it is likely to do so at a quadratic rate. To see this, we borrow an example from Stewart (IMC, p. 366). Suppose H is an unreduced upper Hessenberg matrix of the form

$$H = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & h_{nn} & 0 \end{bmatrix}$$

and that we perform one step of the single-shift QR algorithm, i.e., $UR = H - h_{nn}I$, $\tilde{H} = RU + h_{nn}I$. After $n-2$ steps in the orthogonal reduction of $H - h_{nn}I$ to upper

triangular form we obtain a matrix with the following structure:

$$H = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & a & b \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

It is not hard to show that

$$\tilde{h}_{n,n-1} = -\frac{\epsilon^2 b}{a^2 + \epsilon^2}.$$

If we assume that $\epsilon \lll a$, then it is clear that the new $(n, n-1)$ entry has order 2, precisely what we would expect of a quadratically converging algorithm.

4.4 Algorithm

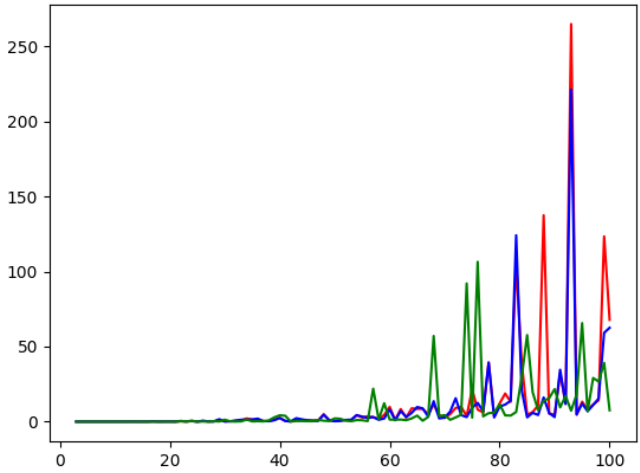
$\mathbf{H}_k \leftarrow \mathbf{U}^\top \mathbf{A}\mathbf{U}$
for $i = 1, 2, \ldots$ **do**
    Compute μ
    $\mathbf{QR} \leftarrow \mathbf{H}_k - \mu\mathbf{I}$
    $\mathbf{H}_k \leftarrow \mathbf{RQ} + \mu\mathbf{I}$
return H_k

4.5 Extracting eigenvalues after performings the QR algorithm

As discussed in the previous sections, if **A** has Complex eigenvalues, QR algorithm will not converge to a completely upper triangular matrix. But rather "blocks" of size 2×2 present along the diagonal. The eigenvalues of the blocks are also eigenvalues of **A**.

We can compute it using the following algorithm

Let A_k
Let $eigs$ be array of eigenvalues
while $i < n$ **do**
    if $i = n$ or $|A_k(i+1, i) = 0|$ **then**
        $eigs[i] \leftarrow A_k(i, i)$
    else
        Find eigenvalues of the 2×2 spanning from $A_k(i, i)$ to $A_k(i+1, i+1)$ and let them be $e1$ and $e2$
        $eigs[i] \leftarrow e1$
        $eigs[i+1] \leftarrow e2$
        $i \leftarrow i + 1$
    $i \leftarrow i + 1$
return $eigs$



4.6 Comparing Different Methods of Optimizations of QR iteration Algorithm

It can be seen that Hessenberg reduction with shifts is overall a much faster algorithm than the other two. But we also can see random spikes in the algorithm. That is because the QR algorithm being performed here needs more tuning as convergence in certain cases is not possible. The reasons for this to happen is discussed in Sandia Technical Report 96-0913J: How the QR algorithm fails to converge and how to fix it by David Day.

5 Conclusion

We discussed about Givens rotation and how to use it to perform QR decomposition of a $n \times n$ matrix. Then we discussed in detail about how the algorithm of QR iteration algorithm works and about the methods of making it more practical.

You can find my codes for the project here: