ASSIGNMENT-3

21MAT204

MIS-3  
Professor- Dr.Neethu Mohan



1. Prove and verify the following using MATLAB.

a) Nonzero eigenvalues of ATA and AAT are same.

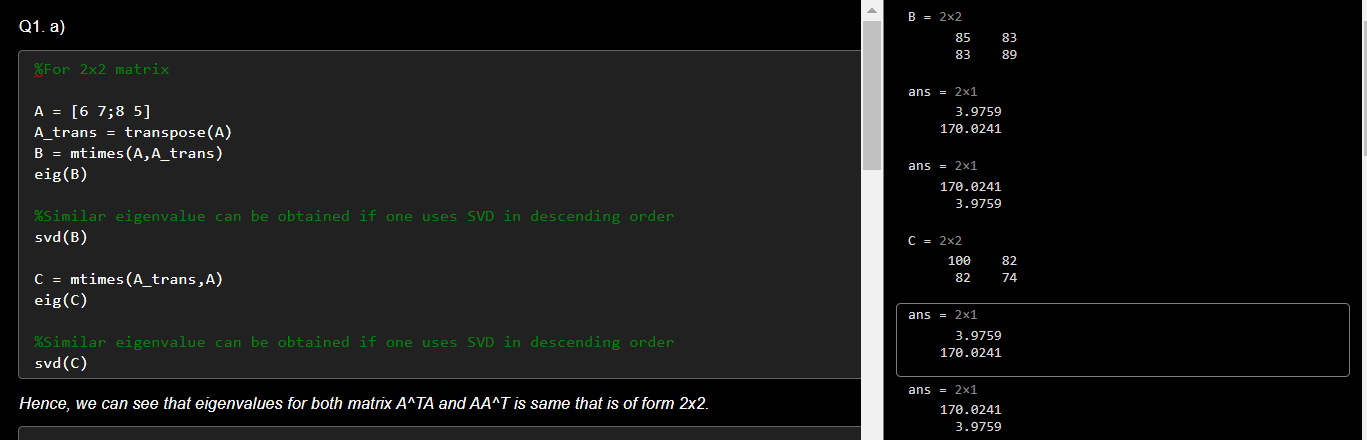
**Code: For 2x2 matrix**

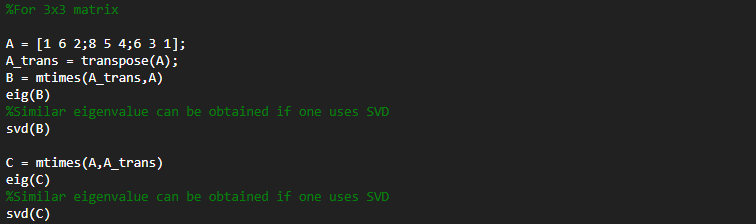
mtimes(A,B) – It is used for matrix multiplication in MATLAB. It will check for correct matrix dimension to follow. It will multiply A with B

Eig(B) – Gives the eigen value of the selected matrix B in form of diagonal matrix or a vector and depending upon parameter one can also get the right and left eigenvector for that matrix using eig funcntion. It can only be applied to square matrix

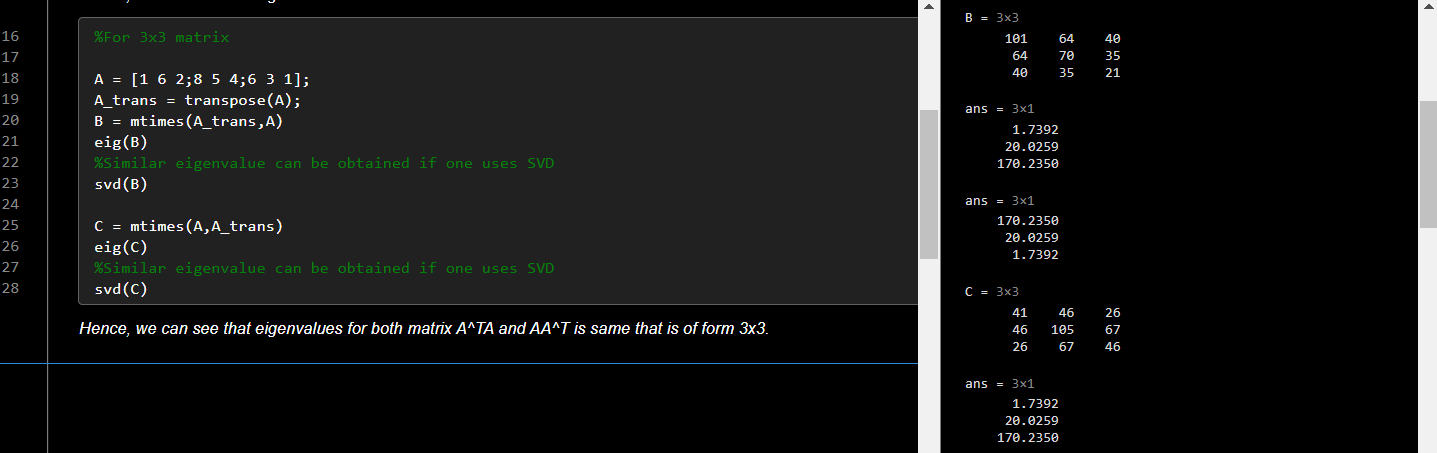
Svd(B) – Gives the singular value decomposition in form of U∑V’ . It is applied to arbitrary sized matrix. It is not a compulsion to use it on square matrix.

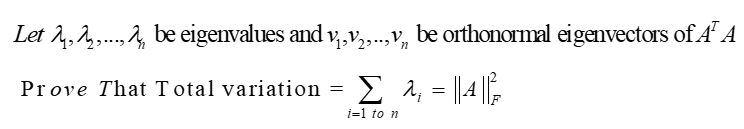
**Output: For 2x2 matrix –**

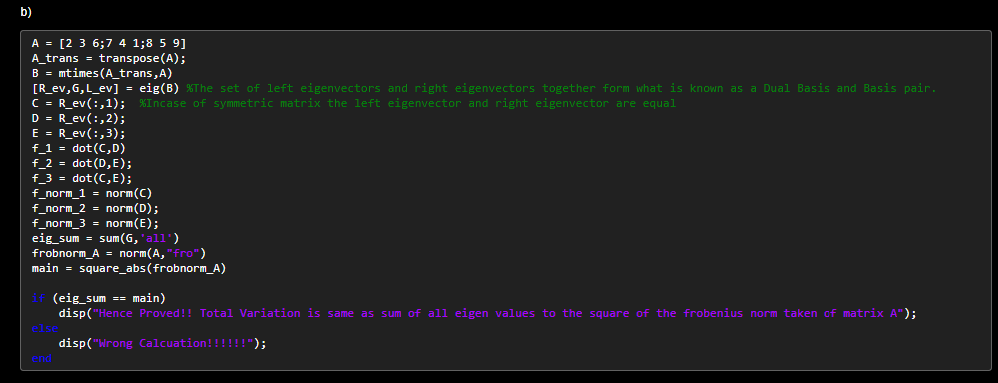


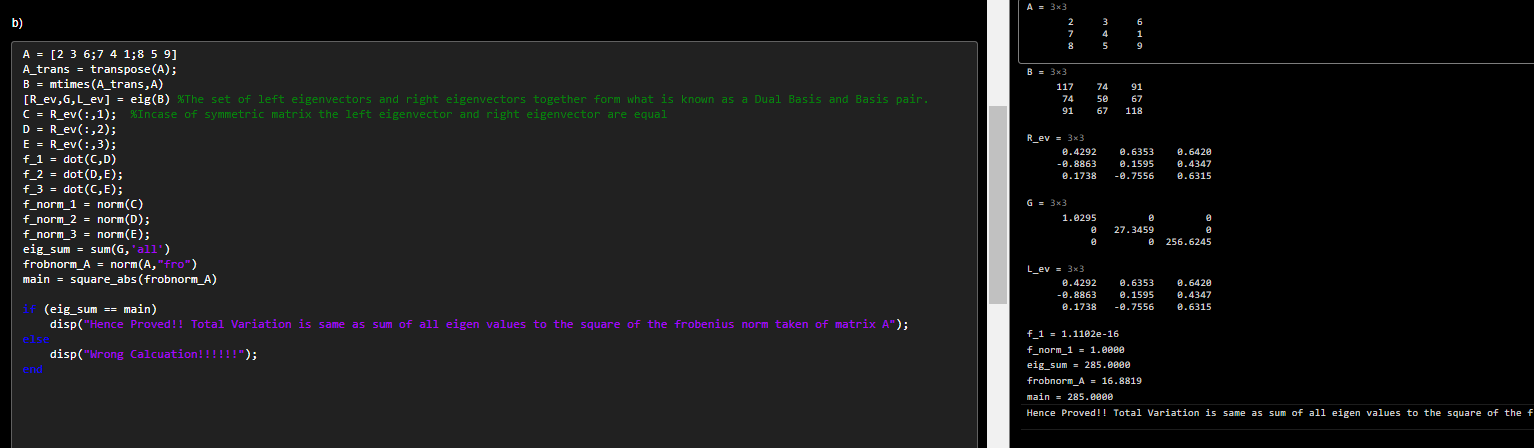
**Code: For 3x3 matrix ––**

**Output: For 3x3 matrix –**



b)

CODE:

OUTPUT: Using eig() – {Please zoom in the picture for the results}

Initially, from eig(B) we are getting right eigenvector (in form of matrix because of three eigen values the matrix has) , diagonal matrix which have eigenvalue and left eigenvector matrix. Then we are going for confirmation regarding vectors that whether they are orthonormal or not.

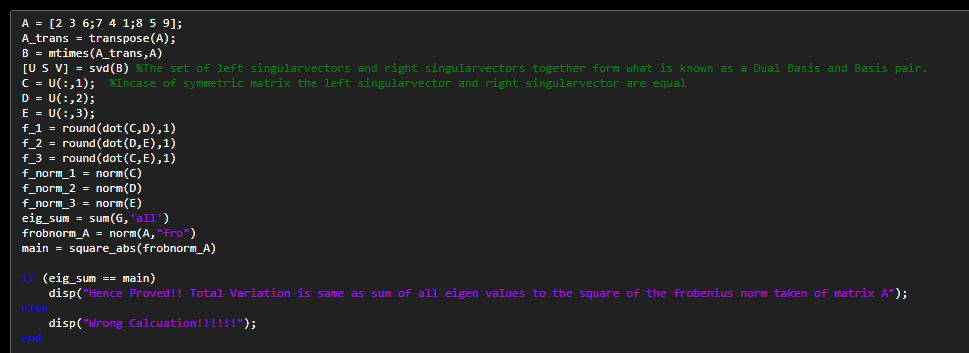
This can be done by checking whether the eigenvector is orthogonal or not and norm of that eigenvector is 1 or not.

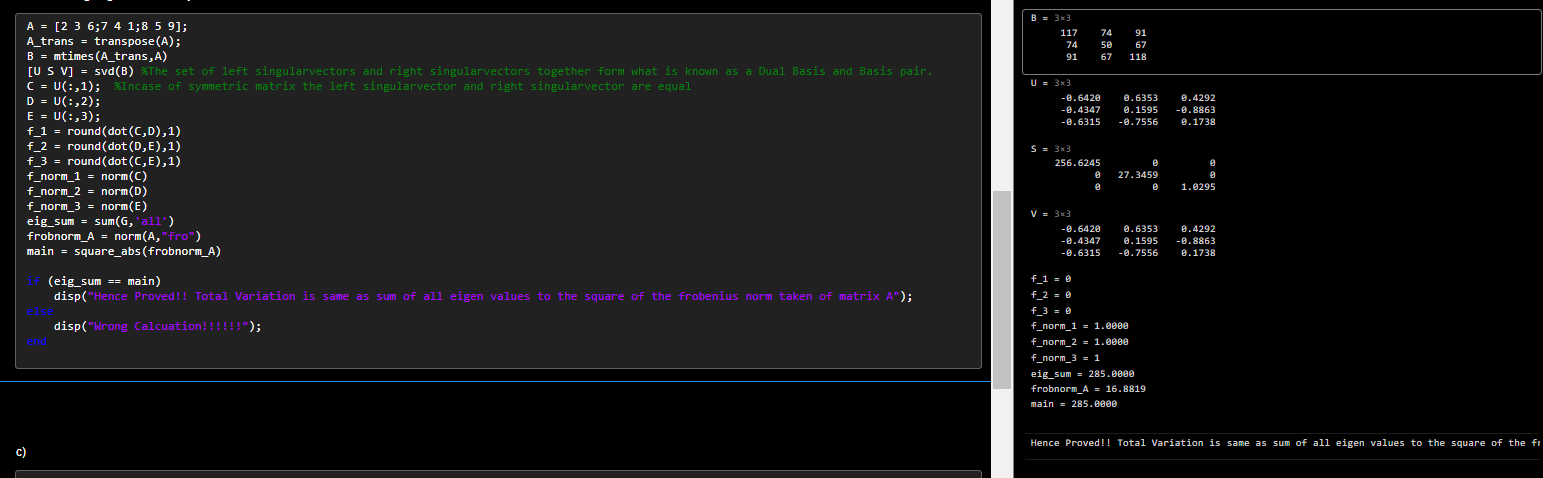
In line 13, we are creating a variable named eig\_sum which will store the sum of all the eigenvalue of ATA.

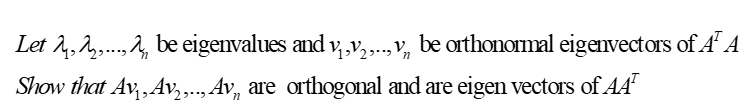
Now we will calculate the square of frobenius norm using norm() function and will keep it inside an if condition.

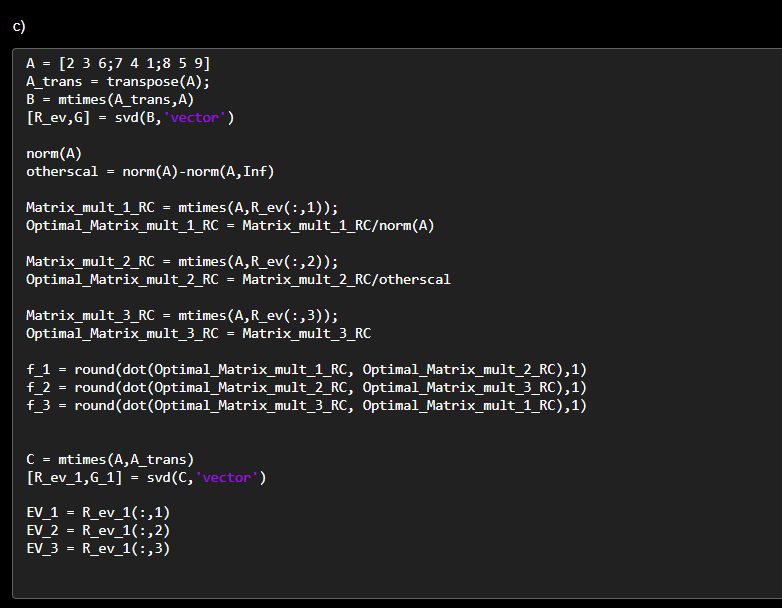
We could have used [U S V] = svd(B) instead of [R\_ev,G,L\_ev] = eig(B) but eig(B) helps in achieving much accurate result in terms of decimal point but the answer will remain unchanged.

We are now going to cross-verify the same.

CODE:

OUTPUT: Using svd() – {Please zoom in the picture for the results}

c)

CODE:

Using svd() we are getting right eigenvector and vector which contains three different eigenvalues.

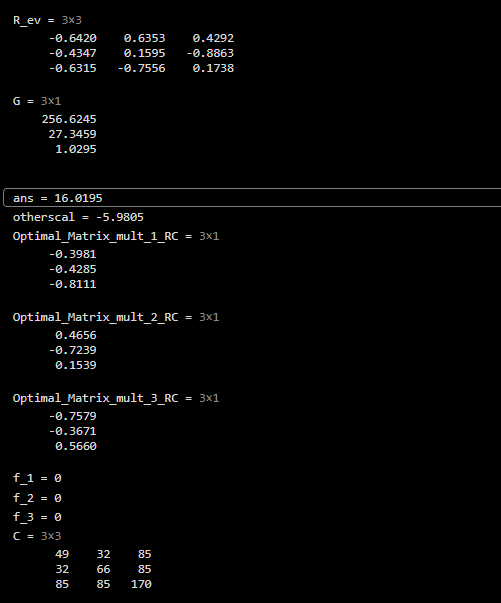
Matrix\_mult\_1\_RC contains the vector which one gets after multiplying it by matrix A

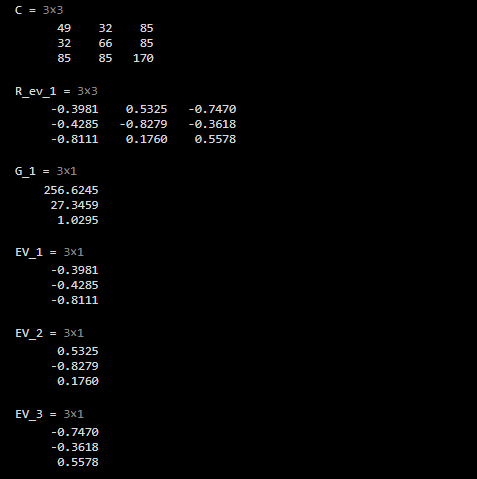
Optimal vector is one which we will get when we divide it by the norm

f\_1 is having dot product which is used to prove that Av1, Av2 and Av3 are orthogonal.

After proving orthogonality we can check Av1, Av2 and Av3 with eigenvector of AAT , we will be getting approximate equality for eigenvectors.

Output:

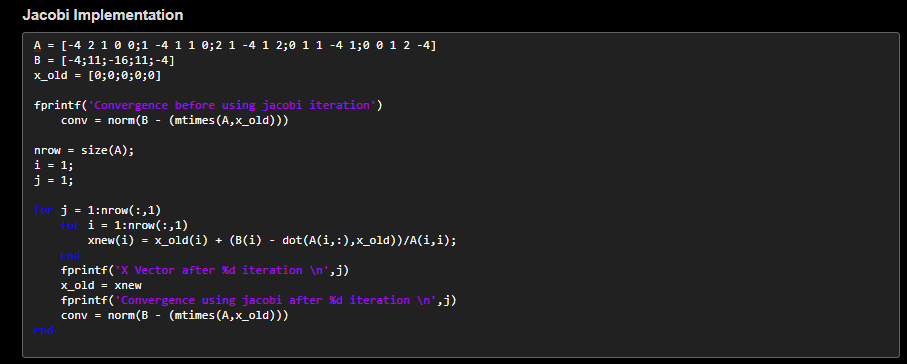




Q2. Demonstrate Jacobi and Gauss-Seidel and SOR iterations for the following data.



a) Using Jacobi Iteration

CODE:

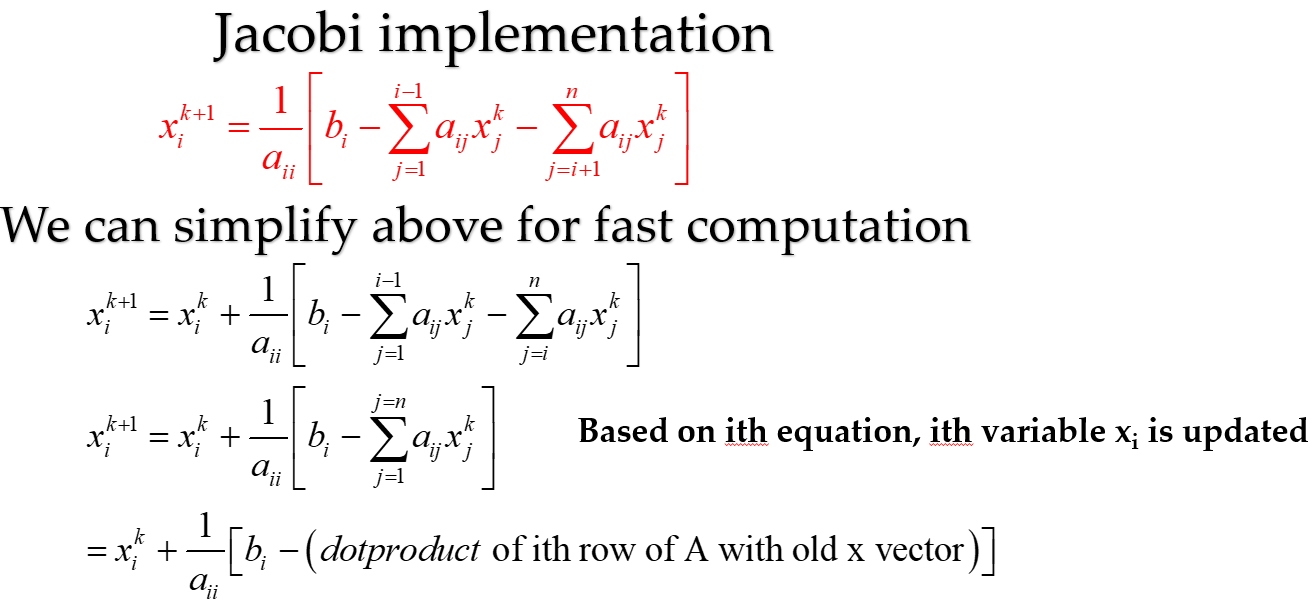
X\_old is the first iteration vector which can be taken randomly and here it is taken as [0,0,0,0,0].

We are calculating norm just before the iteration using norm().

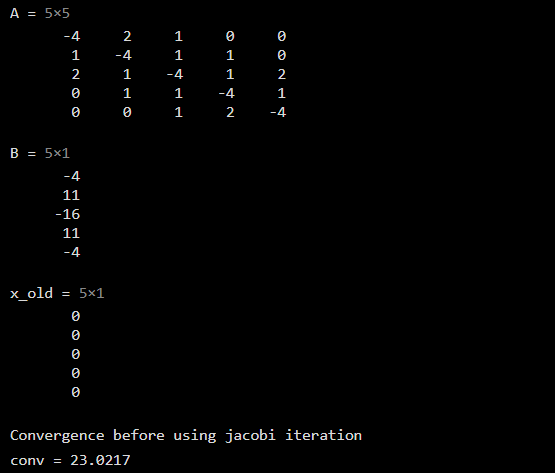
Dimensionality of matrix A is helping us in setting up the number of iterations of the FOR loop

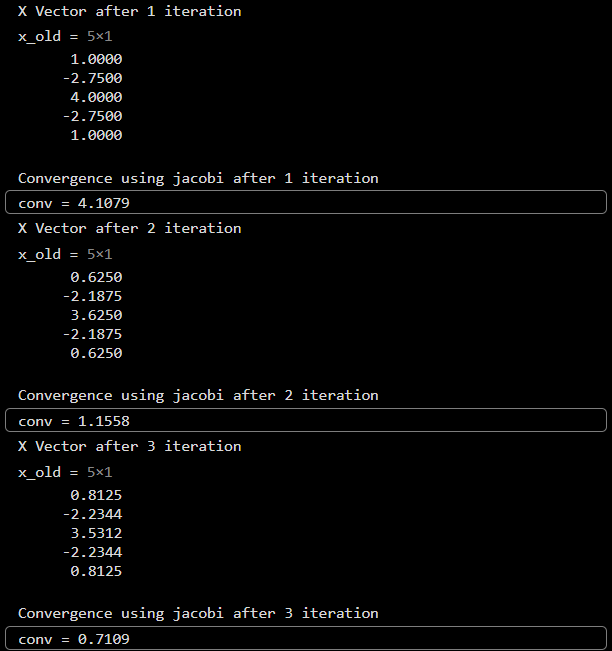
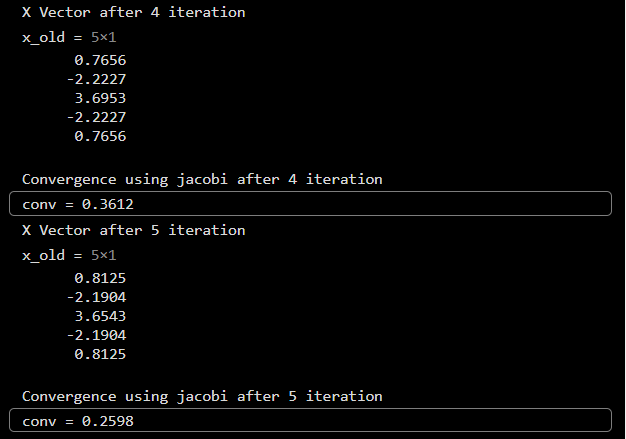
We are going to use two FOR loops where the first loop will change the row and the second loop will change the column and then store that x\_new vector in x\_old vector for repetitive iterations over the new vector.

We are updating the x\_old vector after every iteration and we are printing norm after every iteration.

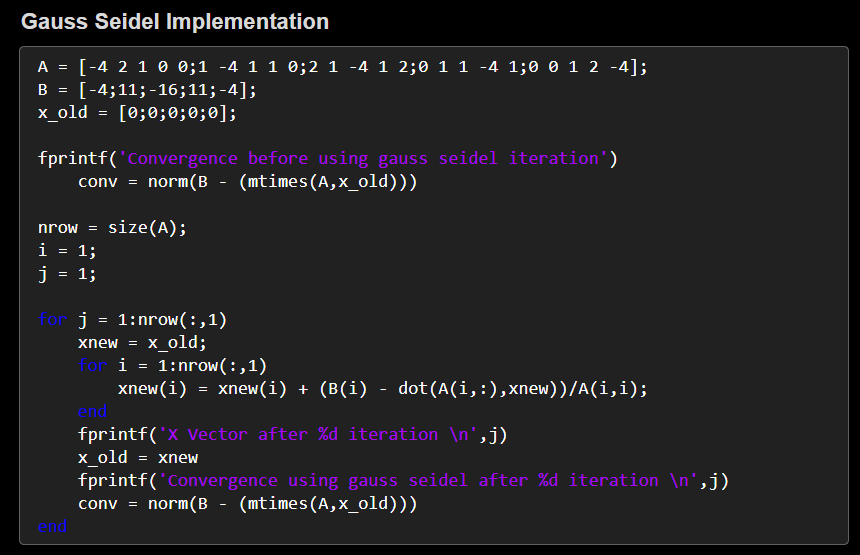
Formula:

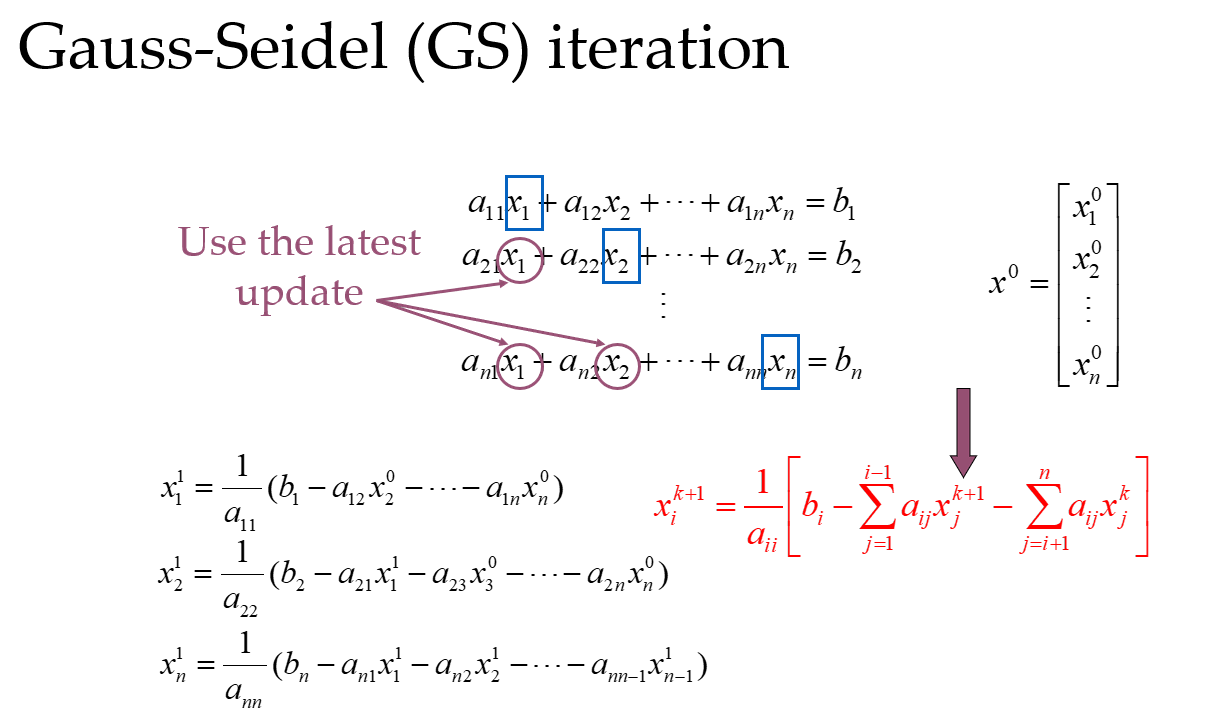
Output:

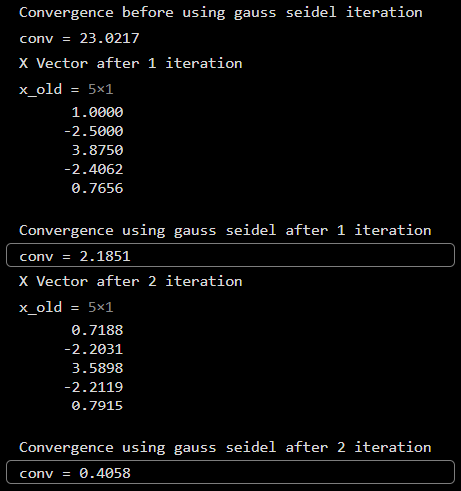


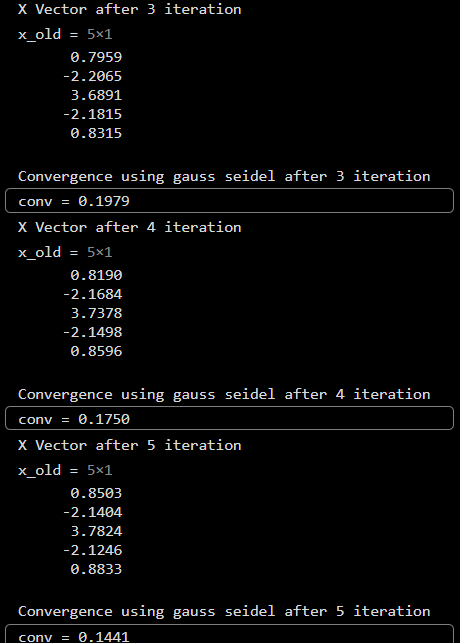


b) Using Gauss-Seidel

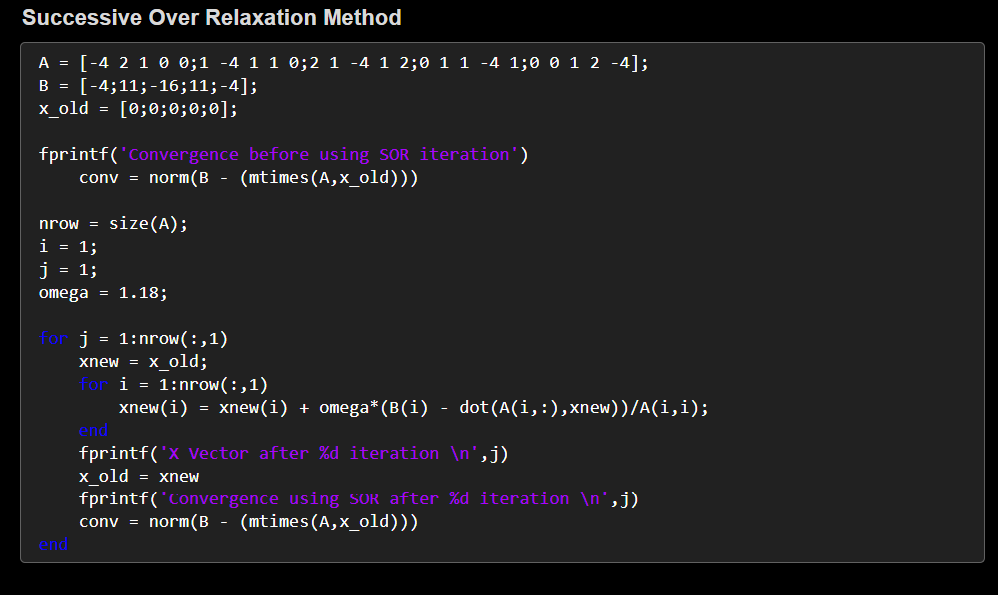
Code:

When compared with code jacobi implementation, one can see that the iteration is taking place considering the new value of x vector rather than taking the old vector itself. X\_Vector is getting updated on a regular basis.  
This helps in accelerating the convergence in comparison to jacobi.

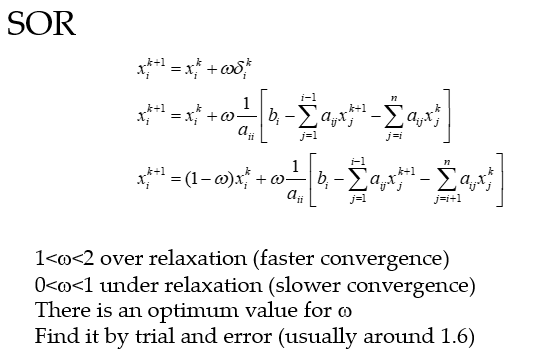
Output:

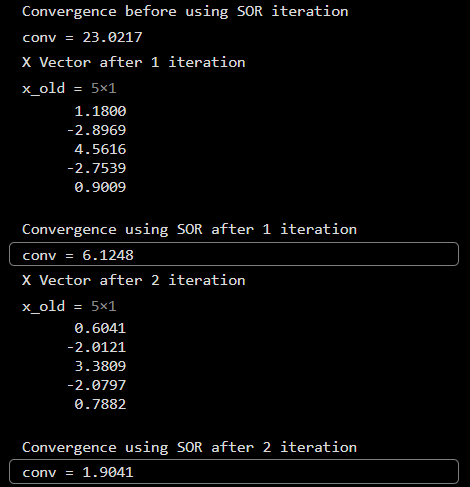


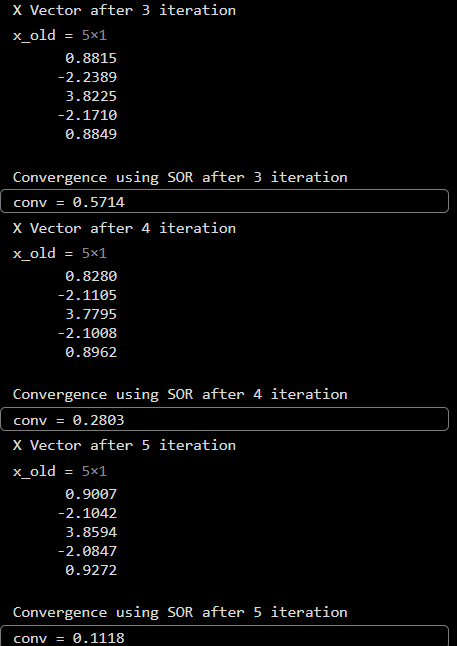
c) Using Successive Over Relaxation method

Code:

When compared to jacobi and gauss-seidel, one can notice that while updating x\_vector for each row there is an omega which is being multiplied to again accelerate the convergence. Here omega is around 1.2.There is a range of omega or which convergence takes place faster and slower accordingly.



Output:



*THANK YOU!!*