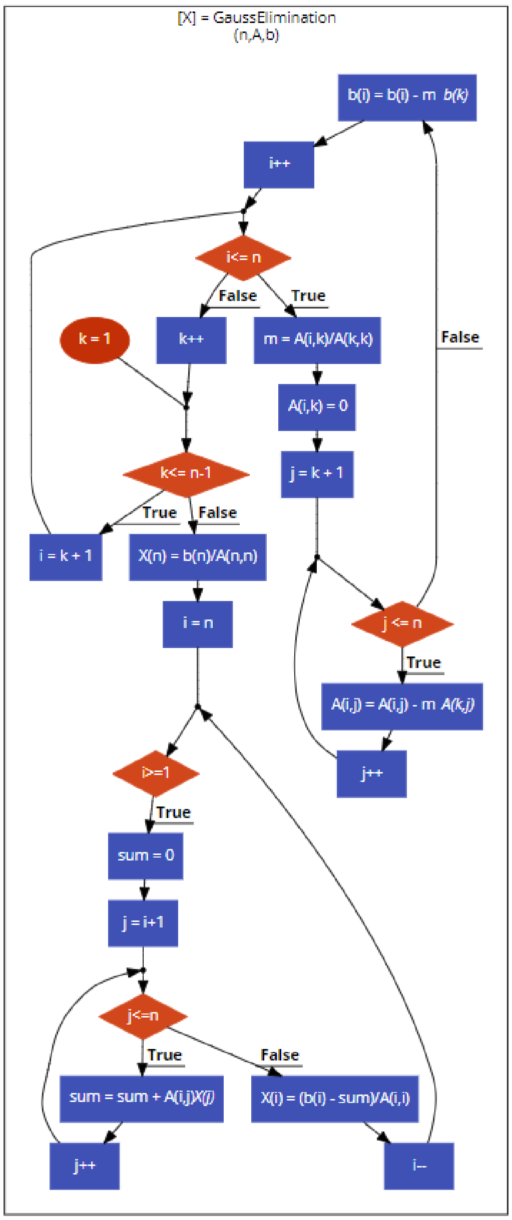
NUMERICAL ANALYSIS ASSIGNMENT 1 - PART 2

14 May 2019

FLOWCHART

1. Gaussian-elimination

Describe how gauss elimination method works:



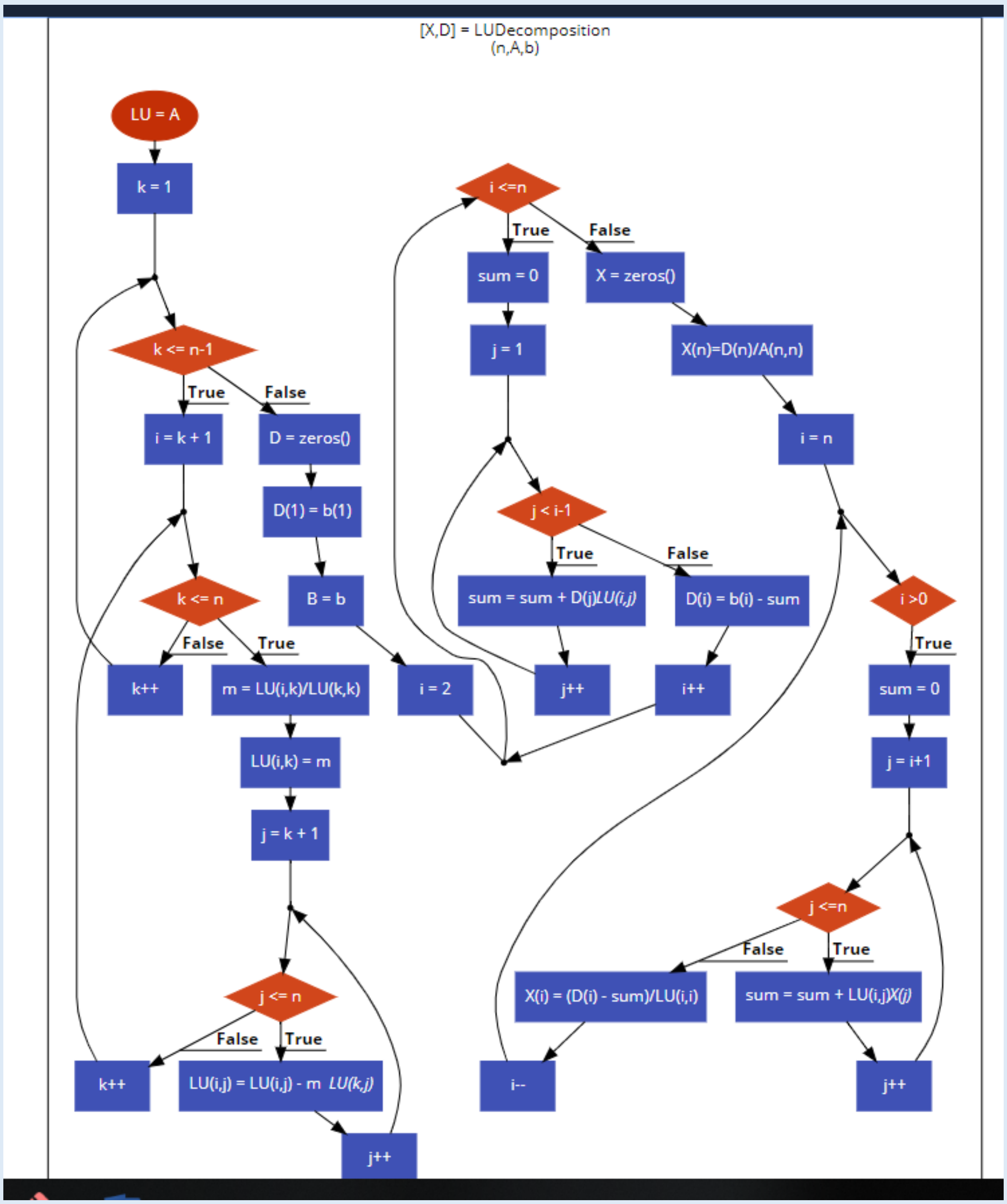
Analysis and conclusion for the behavior of Gauss Elimination:

Gauss Elimination finds the roots of linear equation by eliminating the lower elements in the coefficient matrix then applying backward substitution to get the roots.

Gauss Elimination always converges to the roots of the equation in $O(n^3)$.

2. LU decomposition

Describe how LU decomposition method works:



Analysis and conclusion for the behavior of LU Decomposition:

LU Decomposition finds the roots by splitting the coefficient matrix into lower and upper matrices multiplied together, applying forward substitution on the lower matrix the applying backward substitution on the upper matrix.

LU Decomposition always converges to the roots of the equation in $O(n^3)$.

3. Gaussian-Jordan

Describe how Gaussian Jordan method works:

```
function [timeElapsed,system_matrix,steps,lastMatrix,solution] = gauss_jordan(coeff_matrix,
constants_matrix, num_of_unknowns)
   %create_system_matrix
   for index=1 to length(constants_matrix)
      coeff_matrix(index, length(constants_matrix)+1) <-- constants_matrix(index)
   end
   system_matrix <-- coeff_matrix
   result <-- system_matrix

   %forward_elimination
   steps(1) <-- result
   for pivot_index=1 to num_of_unknowns
      %normalize
      pivot <-- result(pivot_index, pivot_index)
      for col=1 to num_of_unknowns+1
         result(pivot_index, col) <-- result(pivot_index, col)/pivot
      end
       steps(pivot_index) <-- result
      %apply elimination
      for row=1 to num_of_unknowns
         if row equals pivot_index
            continue
         end
         row_pivot <-- result(row,pivot_index)
         for col=pivot_index  to  num_of_unknowns+1
           result(row,col) <-- result(row,col)-row_pivot*result(pivot_index,col)
         end
      end
   end
   lastMatrix <-- result

   %back_substitution
   solution <-- [0,0]
   for i=1  to num_of_unknowns
      solution(i) <-- lastMatrix(i, num_of_unknowns+1)
   end
```

Analysis and conclusion for the behavior of Gauss Jordan:

Its two main purposes are to solve system of linear equations and calculate the inverse of a matrix.

Similar to the Gauss elimination except

1. Elimination is applied to all equations (excluding the pivot equation) instead of just the subsequent equations.

2. All rows are normalized by dividing them by their pivot elements.

3. No back substitution is required.

4. Gauss-Seidel

Describe how Gauss Seidel method works:

```
 function [numOfIterations,final,errorArray,answers] =
GaussSeidel(numberOfFunctions,coeff_matrix,constants_matrix,initialGuess,maxIterations,epsil
on)
answers = double.empty
previous = double.empty
for i = 1 to numberOfFunctions
    answers <-- [answers;initialGuess(i)]
end
iteration = 1 , error = 100
errorArray = double.empty
final = double.empty
while  (iteration less than or equal maxIterations & error greater than epsilon)
    for i = 1 to length(answers)
        j = 1
        previous(i) <-- answers(i)
        answers(i) <-- constants_matrix(i)
        for k = 1 to length(answers) - 1
            if (j equal i)
                j <-- j + 1
            end
            answers(i) <-- answers(i) - coeff_matrix(i,j)*answers(j)
            j = j + 1
        end
        answers(i) <-- answers(i) / coeff_matrix(i,i)
        error <-- abs(((answers(i) - previous(i)) / answers(i)) * 100)
        errorArray(i , iteration) <-- error
        final(i,iteration) <-- answers(i)
    end
    iteration  <-- iteration + 1
end
numOfIterations <-- iteration - 1
```

Analysis and conclusion for the behavior of Gauss Seidal:

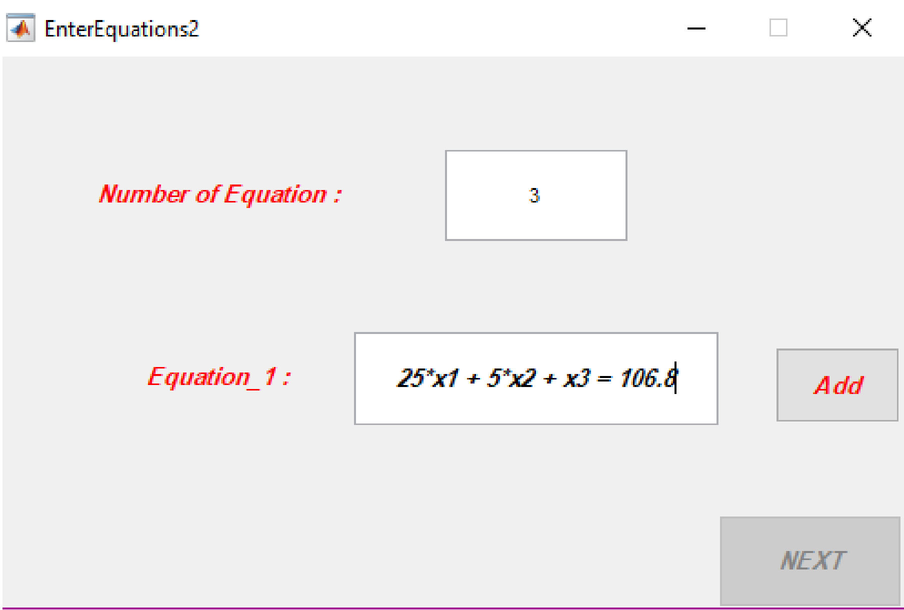
It is an iterative method used to solve a linear system of equations of n linear equations with unknown x: A x = B

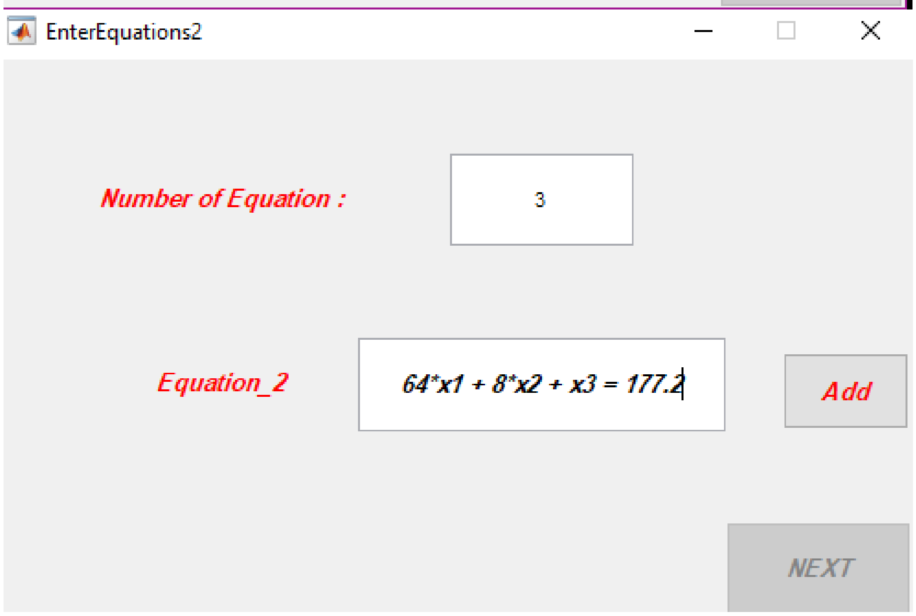
Though it can be applied to any matrix with non-zero elements on the diagonals, convergence is only guaranteed if the matrix is either diagonally dominant, or symmetric and positive definite.

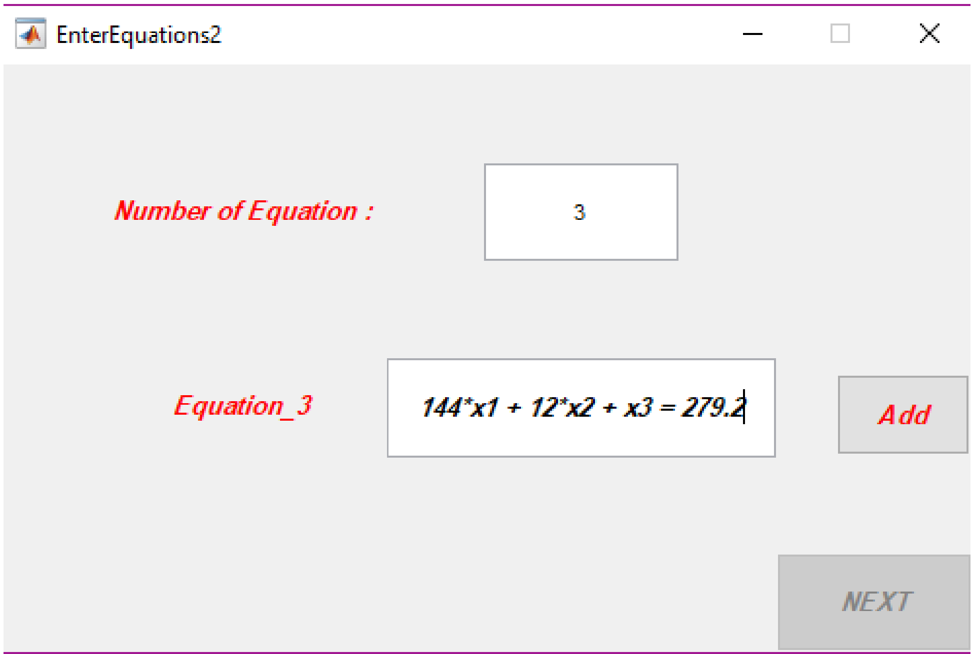
PROBLAMATIC FUNCTIONS

<u>In Gauss Seidel method :</u>
One class of system of equations always converges: One with a diagonally dominant coefficient matrix, with non-zero elements on the diagonals.
If a system of linear equations is not diagonally dominant, check to see if rearranging the equations can form a diagonally dominant matrix.
Not every system of equations can be rearranged to have a diagonally dominant coefficient matrix.

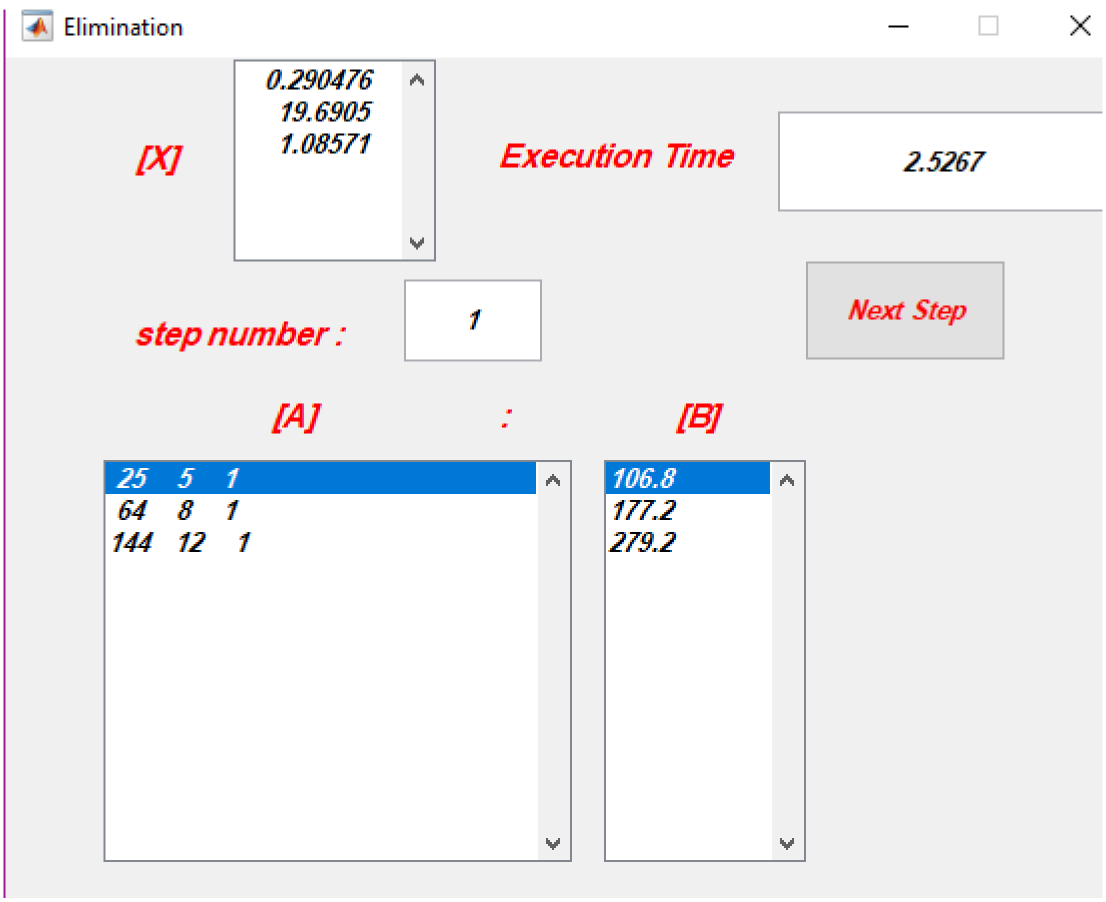
SAMPLE RUNS AND SNAPSHOTS

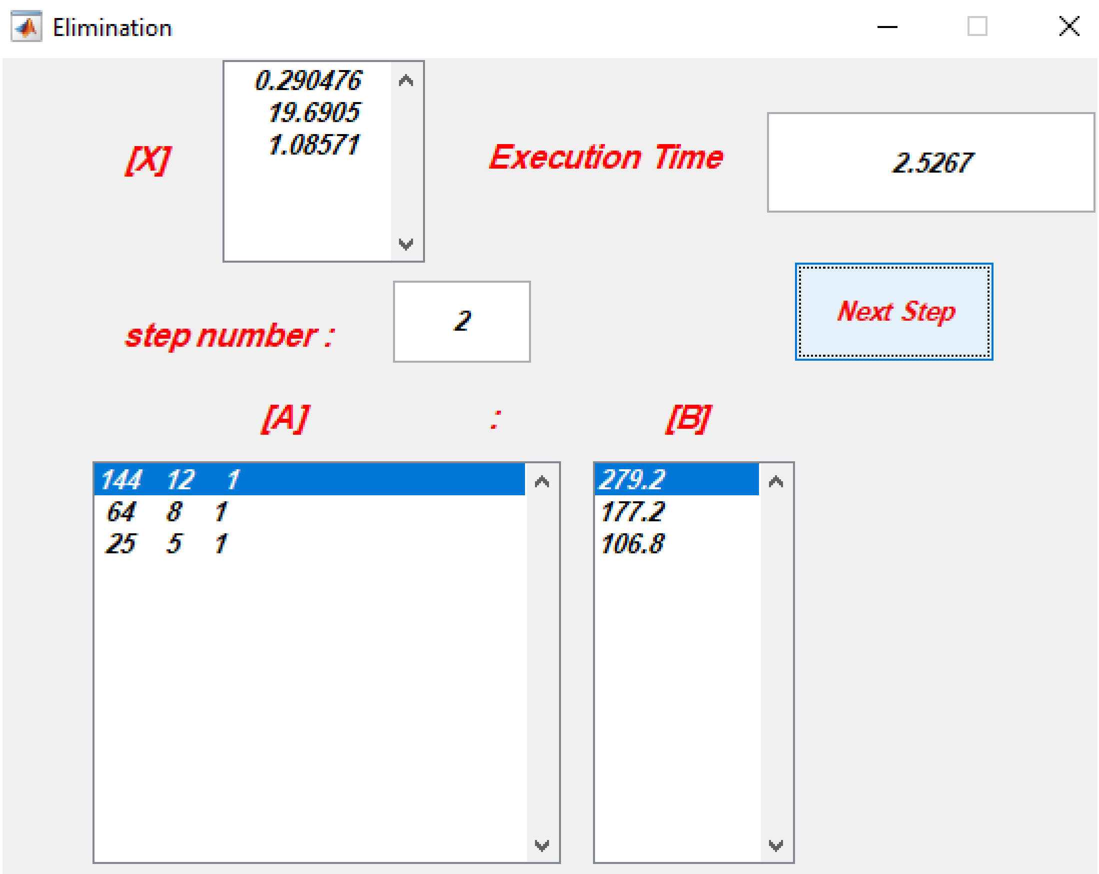


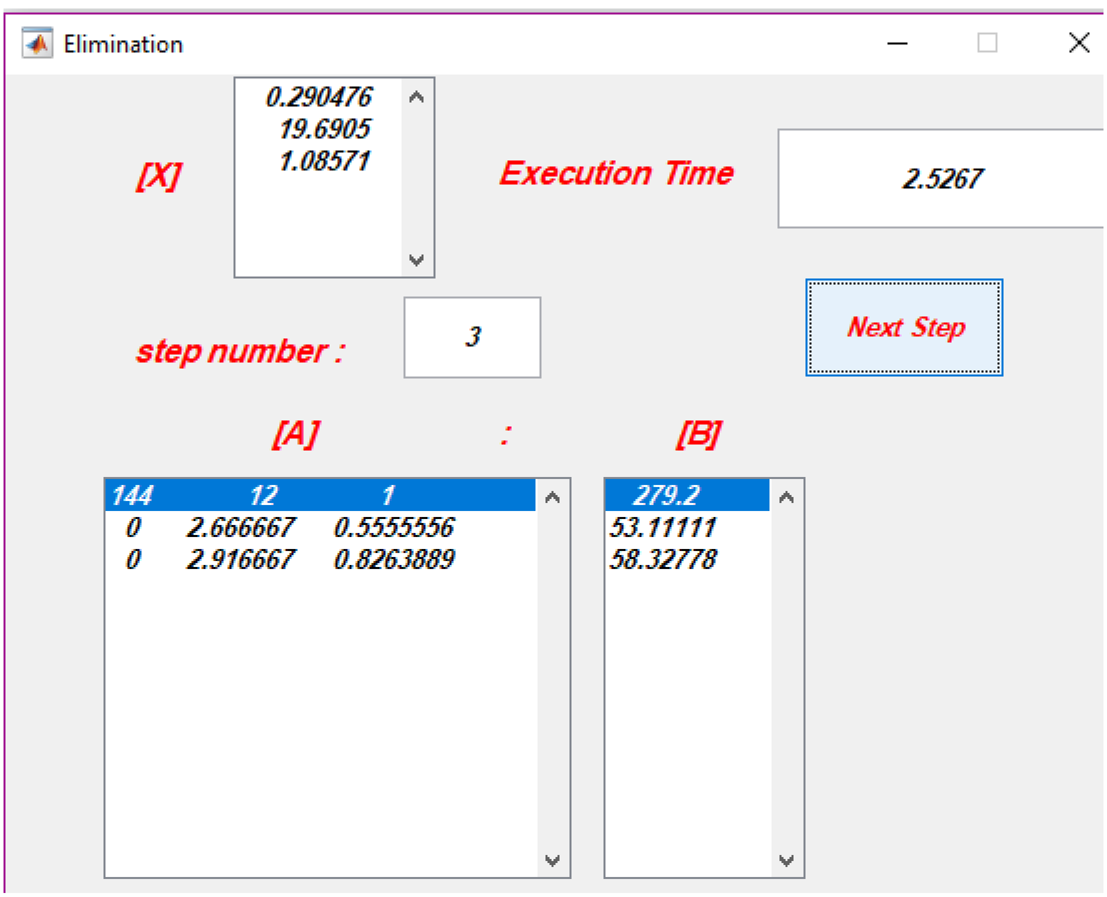


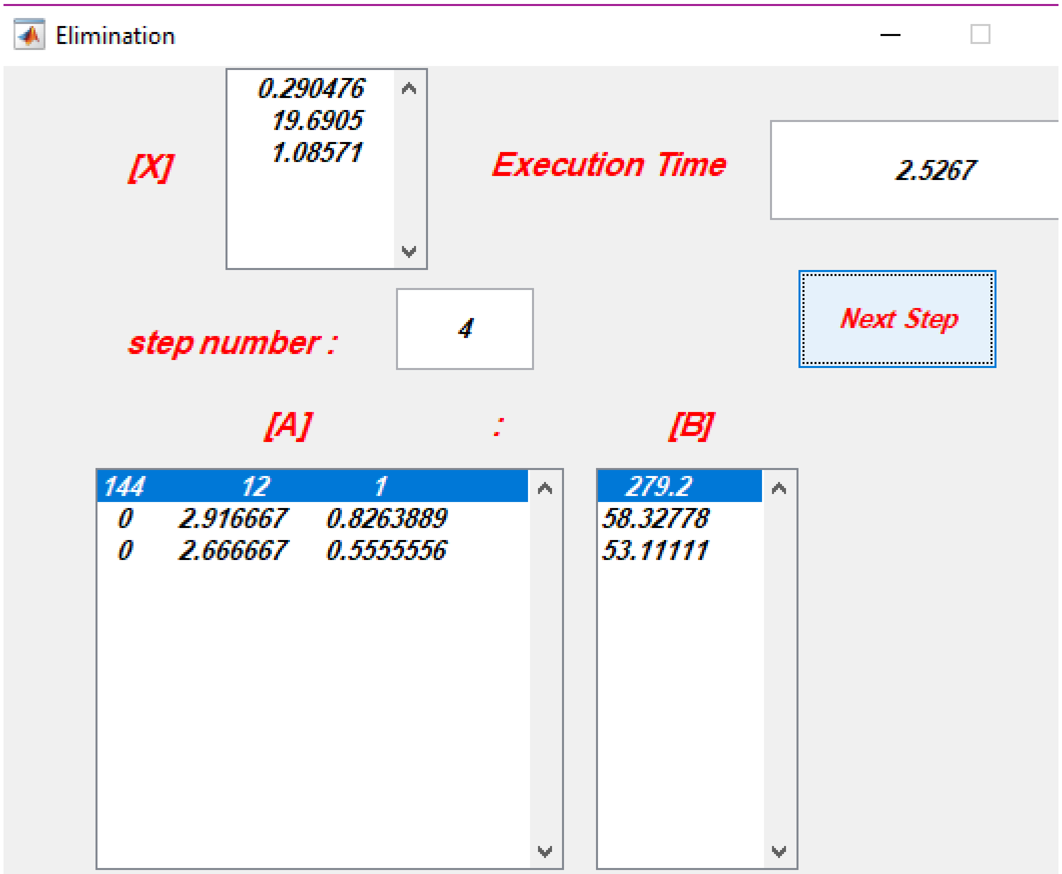


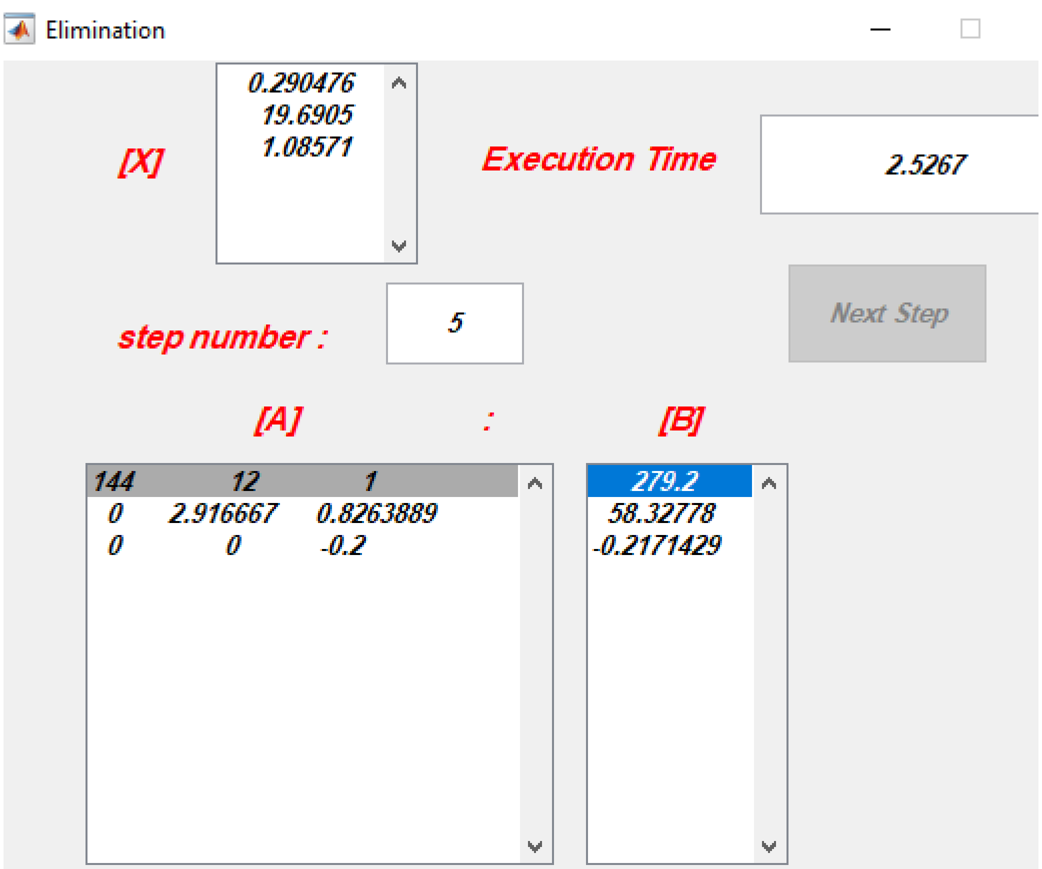
Gaussian-elimination



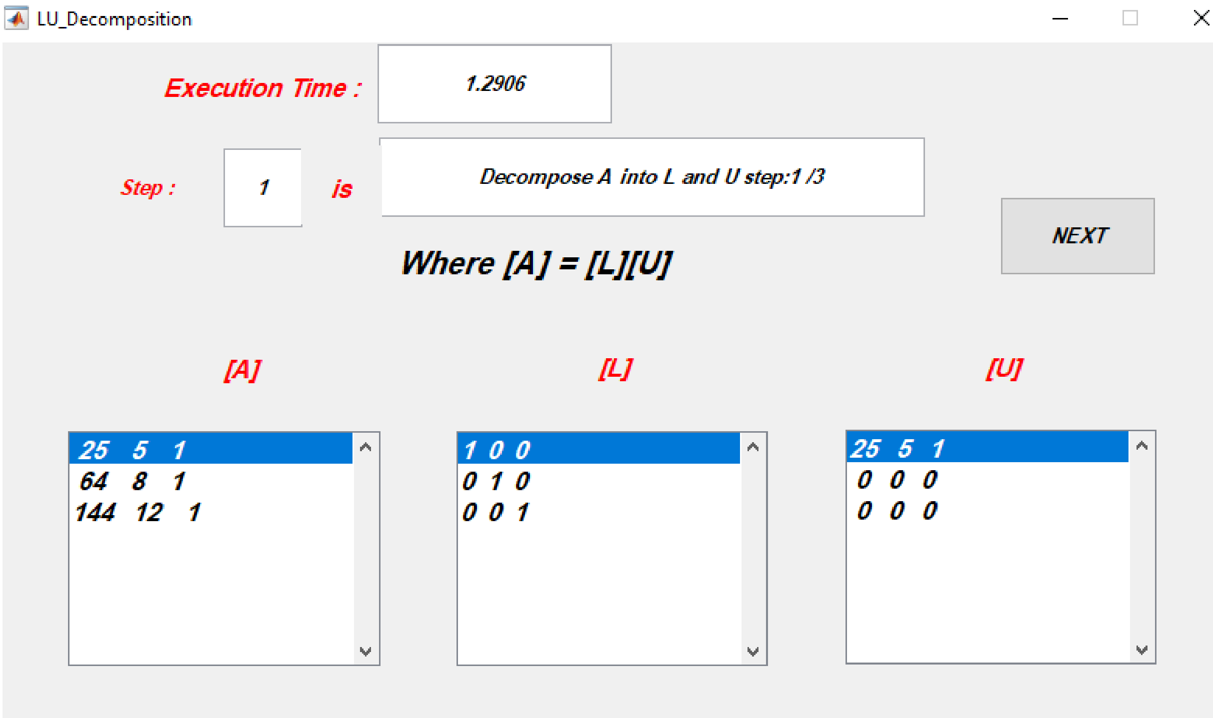

Elimination
[X]
0.290476
19.6905
1.08571
Execution Time
2.5267
step number :
2
Next Step
[A]
:
[B]
144 12 1
64 8 1
25 5 1
279.2
177.2
106.8

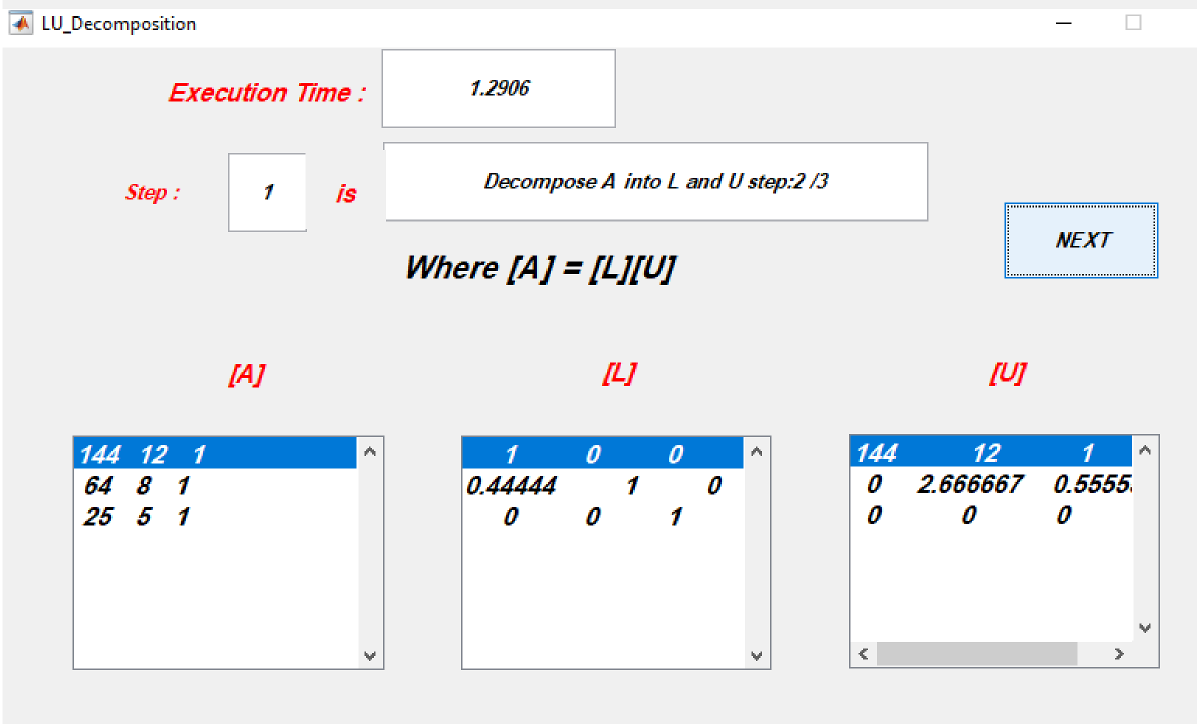

Elimination
[X]
0.290476
19.6905
1.08571
Execution Time
2.5267
step number :
3
Next Step
[A]
:
[B]
144 12 1
0 2.666667 0.5555556
0 2.916667 0.8263889
279.2
53.11111
58.32778

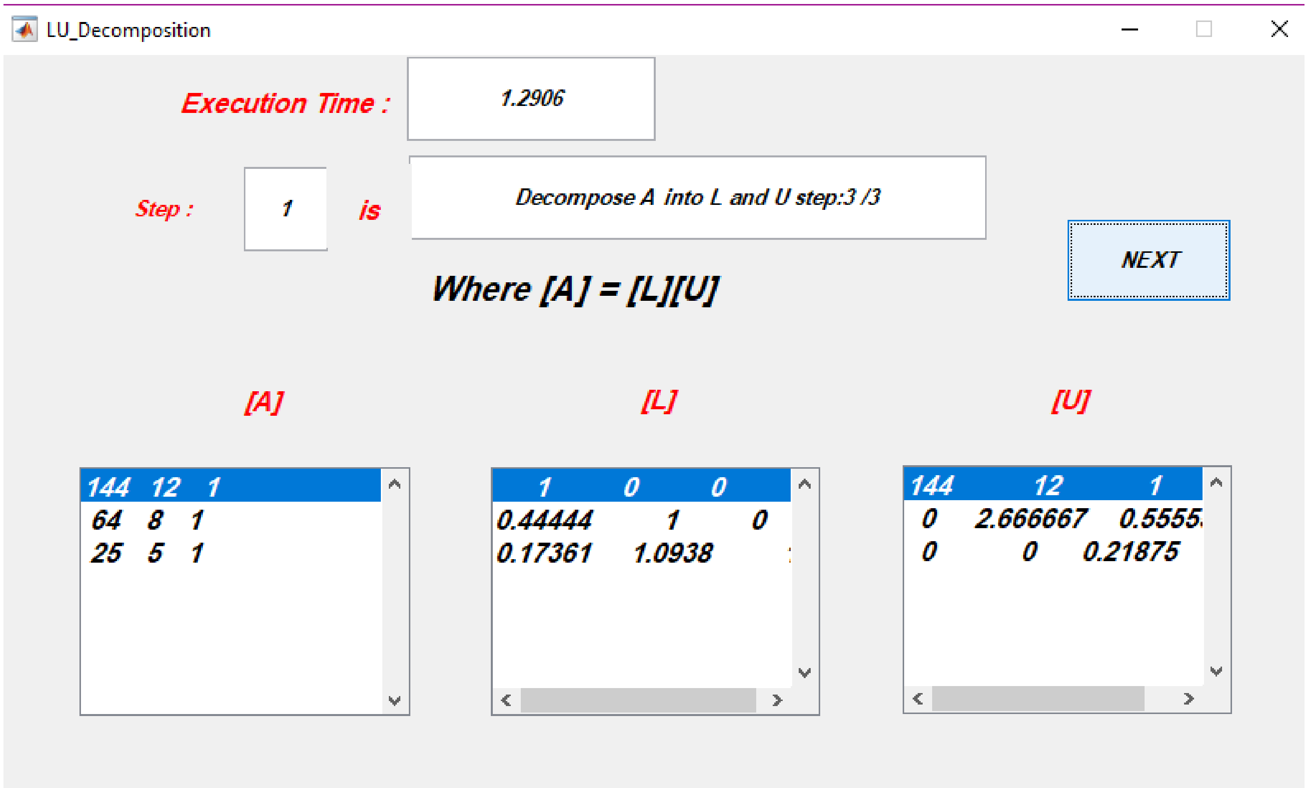

Elimination
[X]
0.290476
19.6905
1.08571
Execution Time
2.5267
step number :
4
Next Step
[A]
:
[B]
144 12 1
0 2.916667 0.8263889
0 2.666667 0.5555556
279.2
58.32778
53.11111


Elimination
[X]
0.290476
19.6905
1.08571
Execution Time
2.5267
step number :
5
Next Step
[A]
:
[B]
144 12 1
0 2.916667 0.8263889
0 0 -0.2
279.2
58.32778
-0.2171429

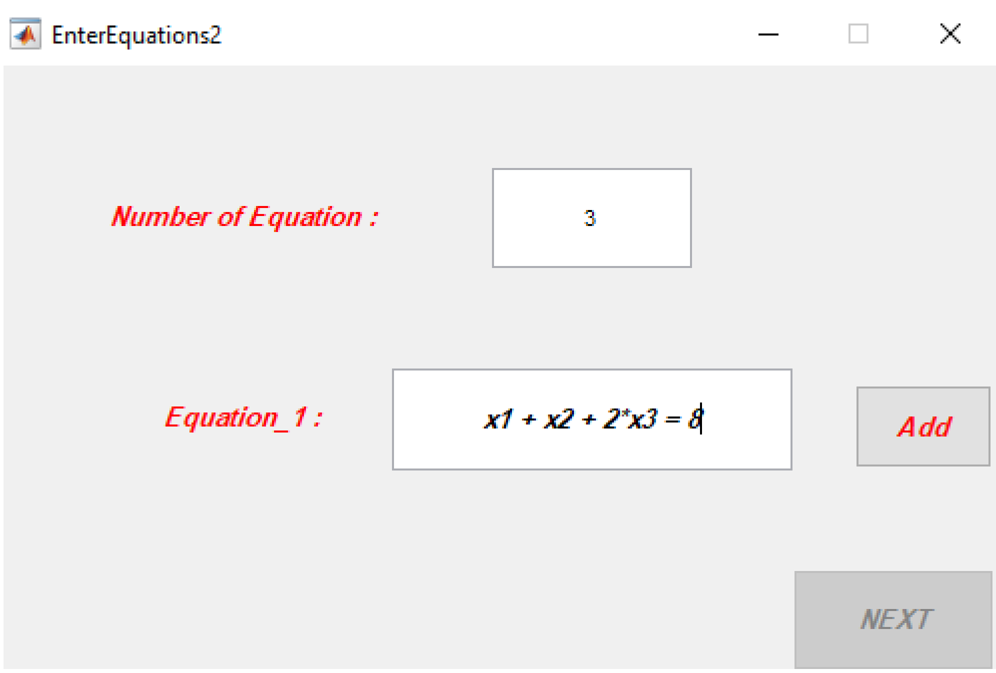
LU Decomposition

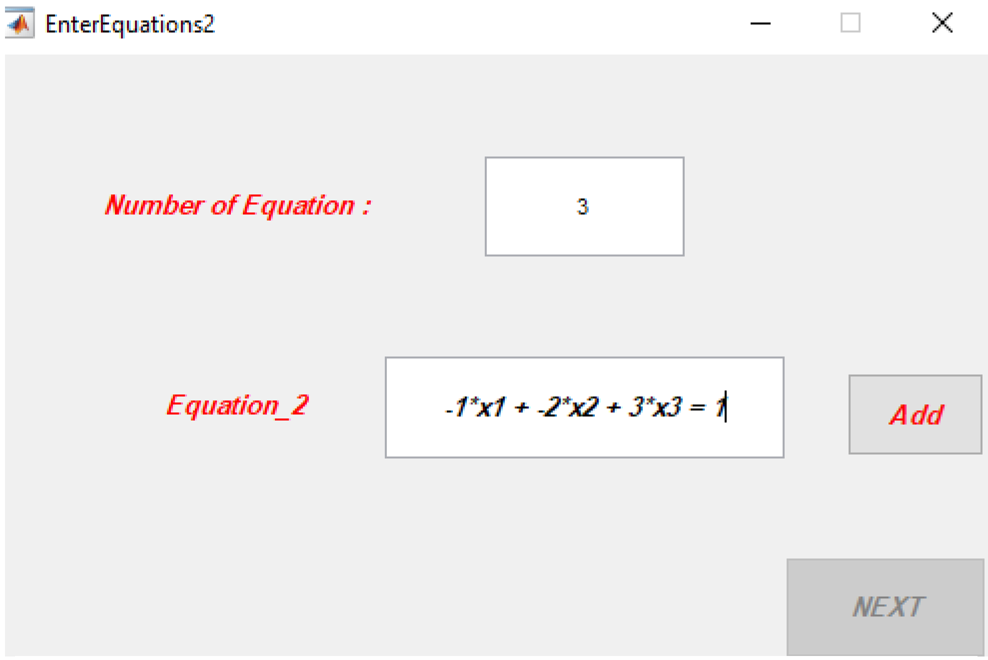


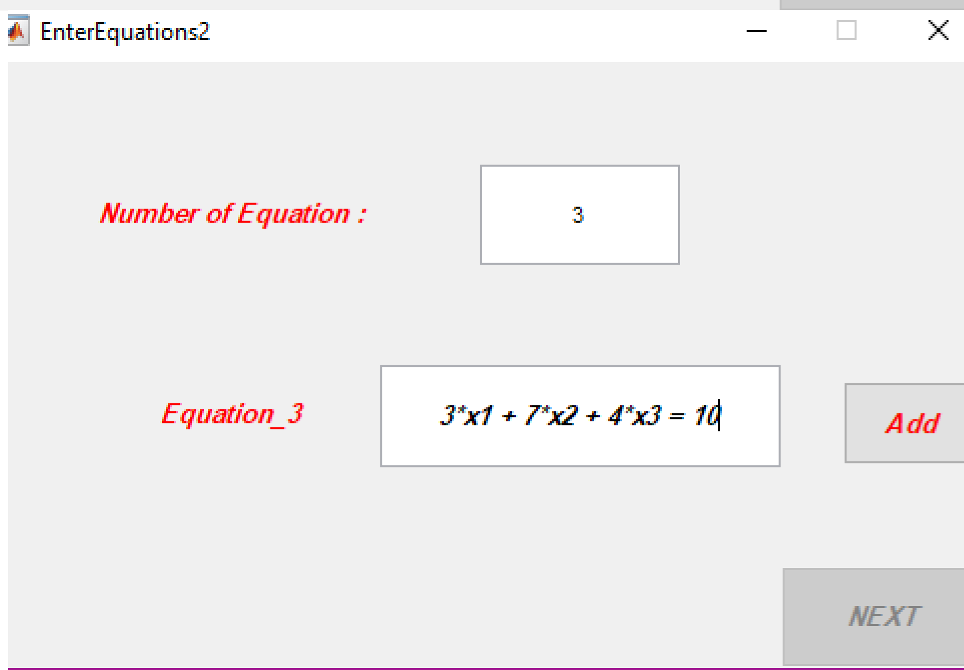


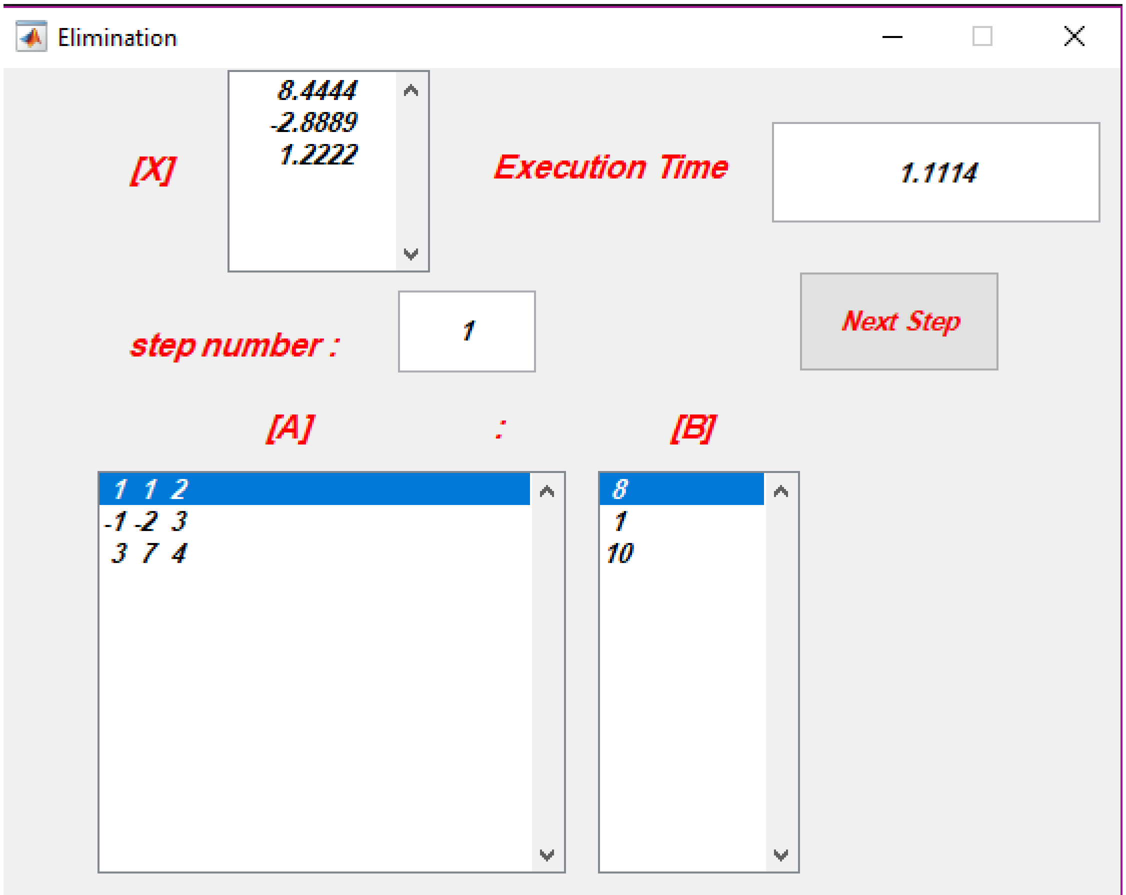


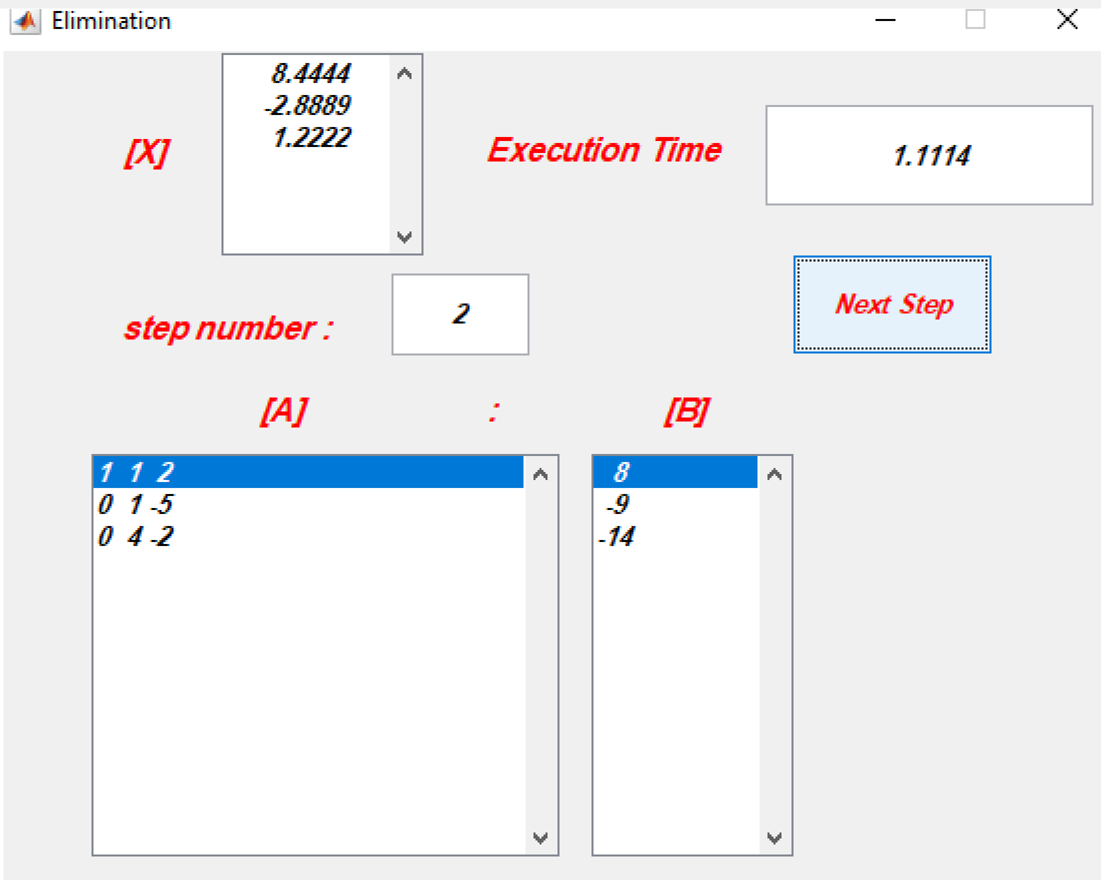
Gauss Jordan

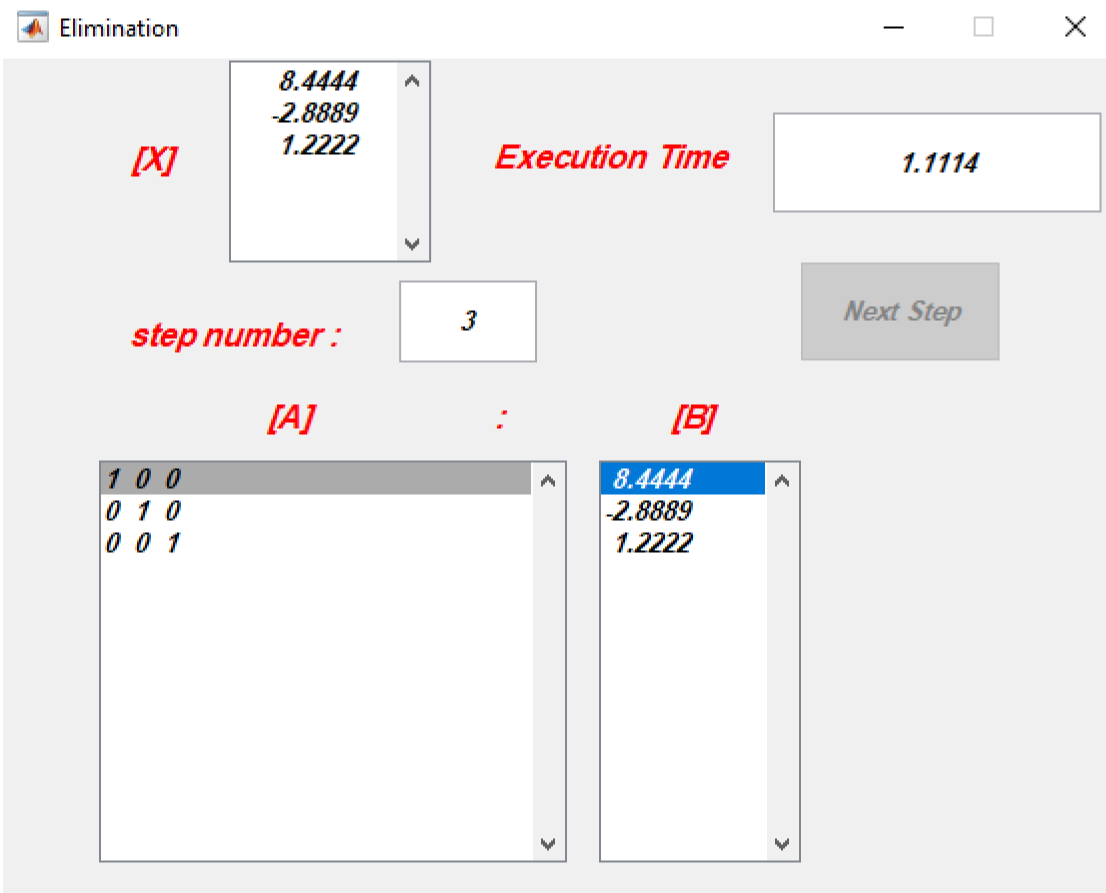



EnterEquations2
Number of Equation :
3
Equation_2
-1*x1 + -2*x2 + 3*x3 = 1
Add
NEXT


EnterEquations2
Number of Equation :
3
Equation_3
3*x1 + 7*x2 + 4*x3 = 10
Add
NEXT

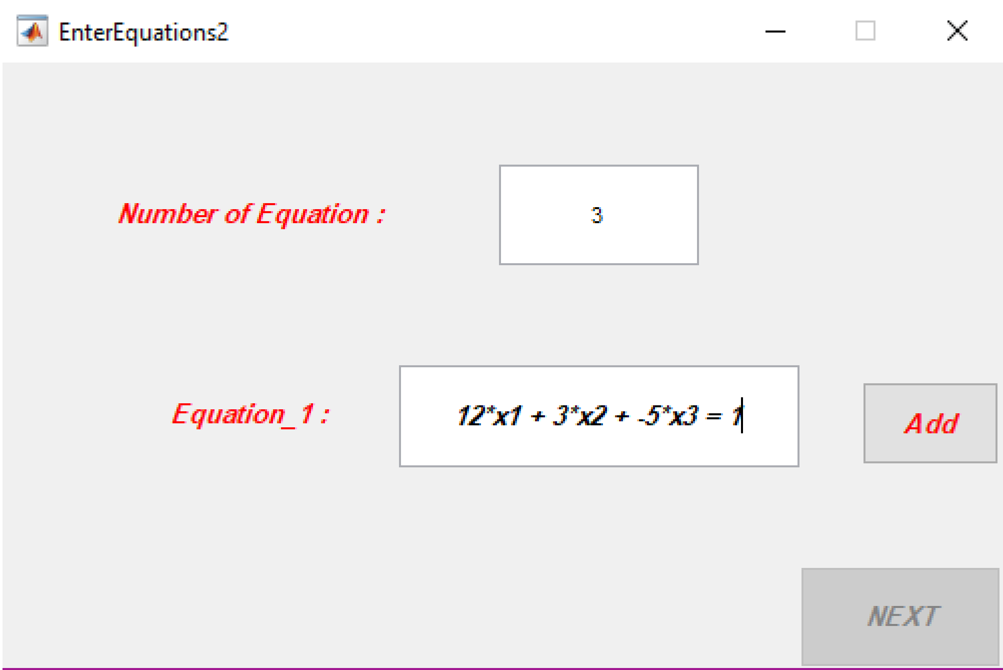

Elimination
[X]
8.4444
-2.8889
1.2222
Execution Time
1.1114
step number :
1
Next Step
[A]
:
[B]
1 1 2
-1 -2 3
3 7 4
8
1
10

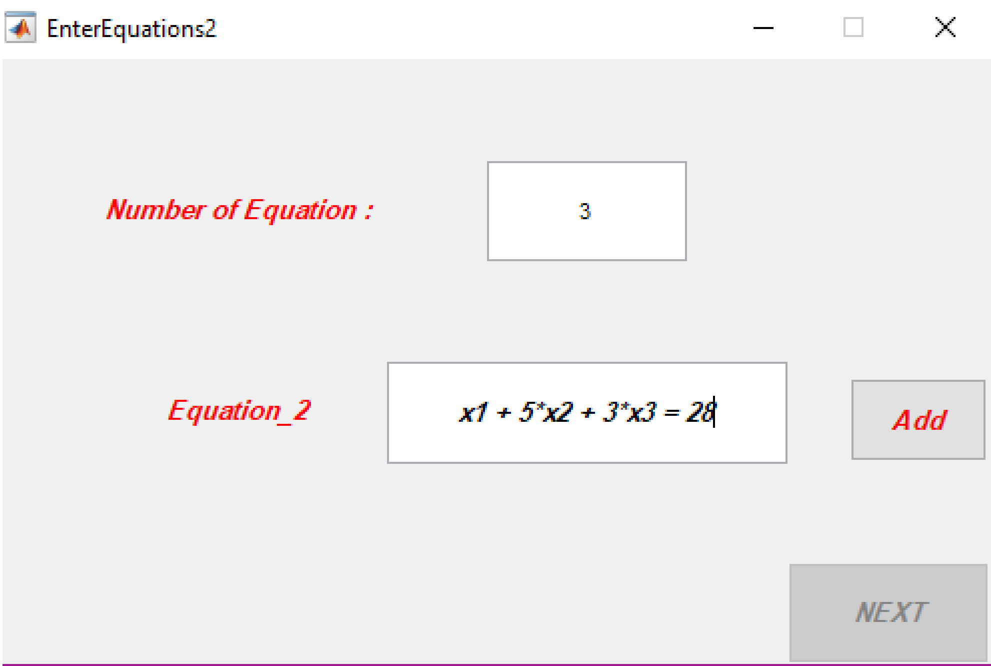

Elimination
[X]
8.4444
-2.8889
1.2222
Execution Time
1.1114
step number :
2
Next Step
[A]
:
[B]
1 1 2
0 1 -5
0 4 -2
8
-9
-14

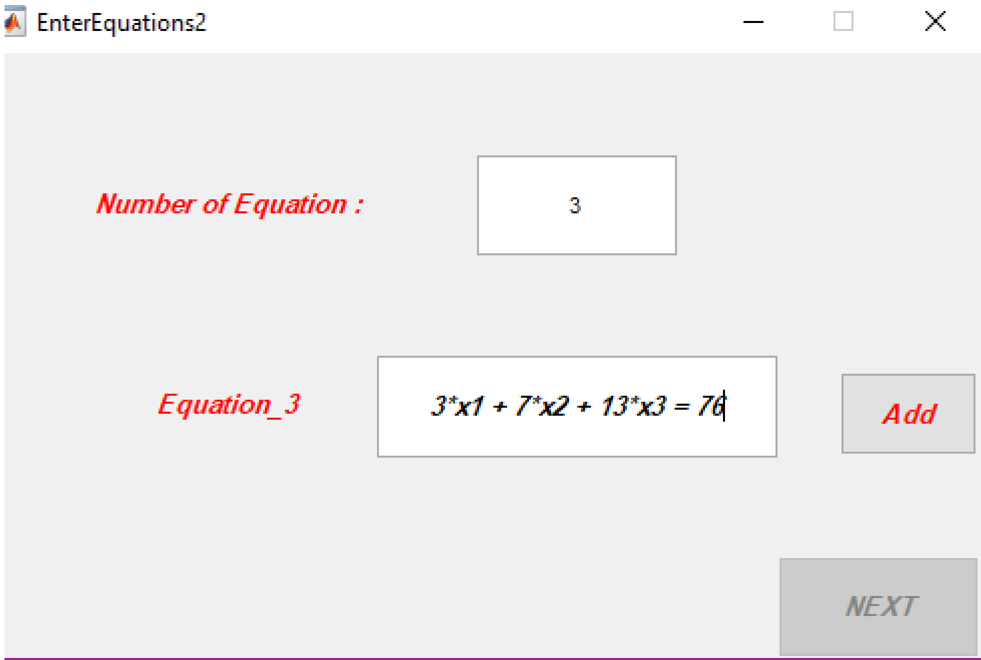

Elimination
[X]
8.4444
-2.8889
1.2222
Execution Time
1.1114
step number :
3
Next Step
[A]
:
[B]
1 0 0
0 1 0
0 0 1
8.4444
-2.8889
1.2222

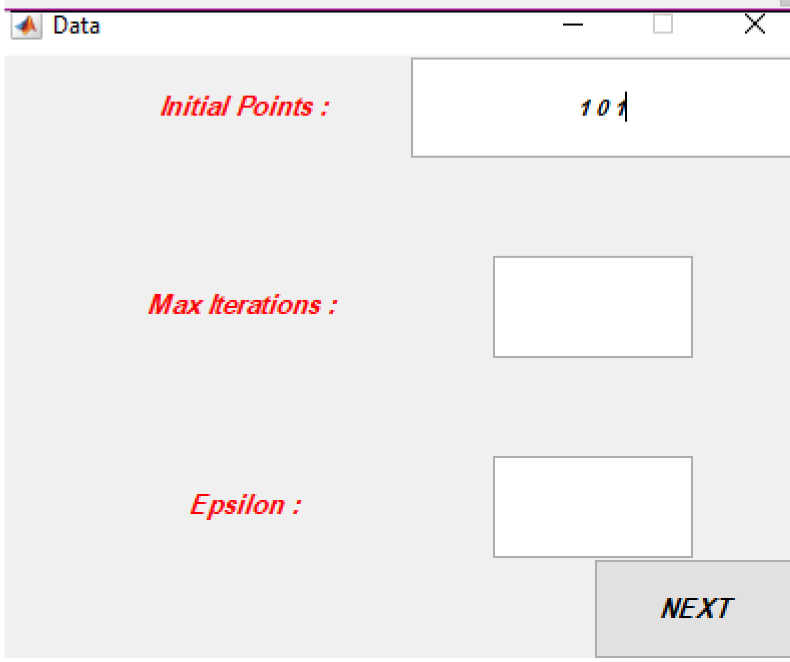
Gauss Seidel Method

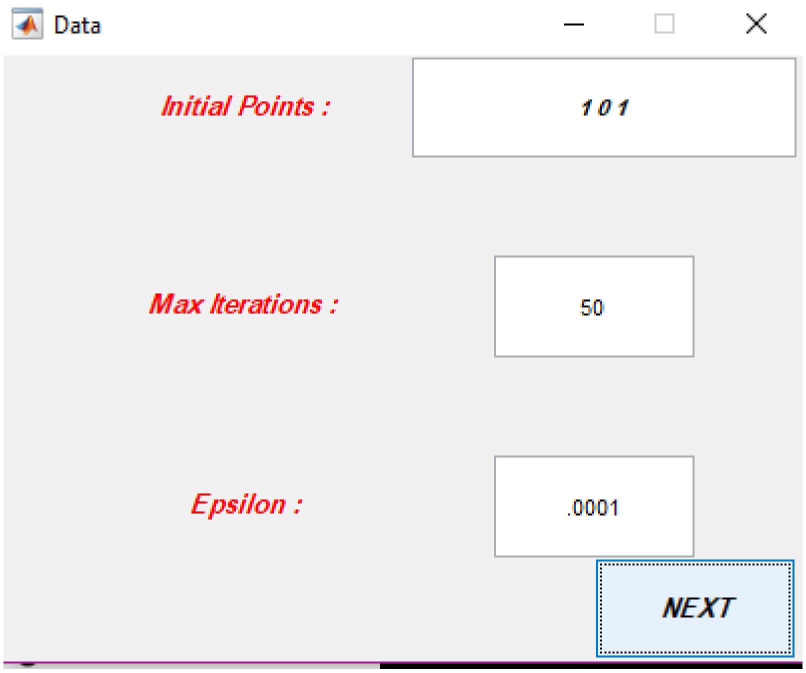
Which converges :

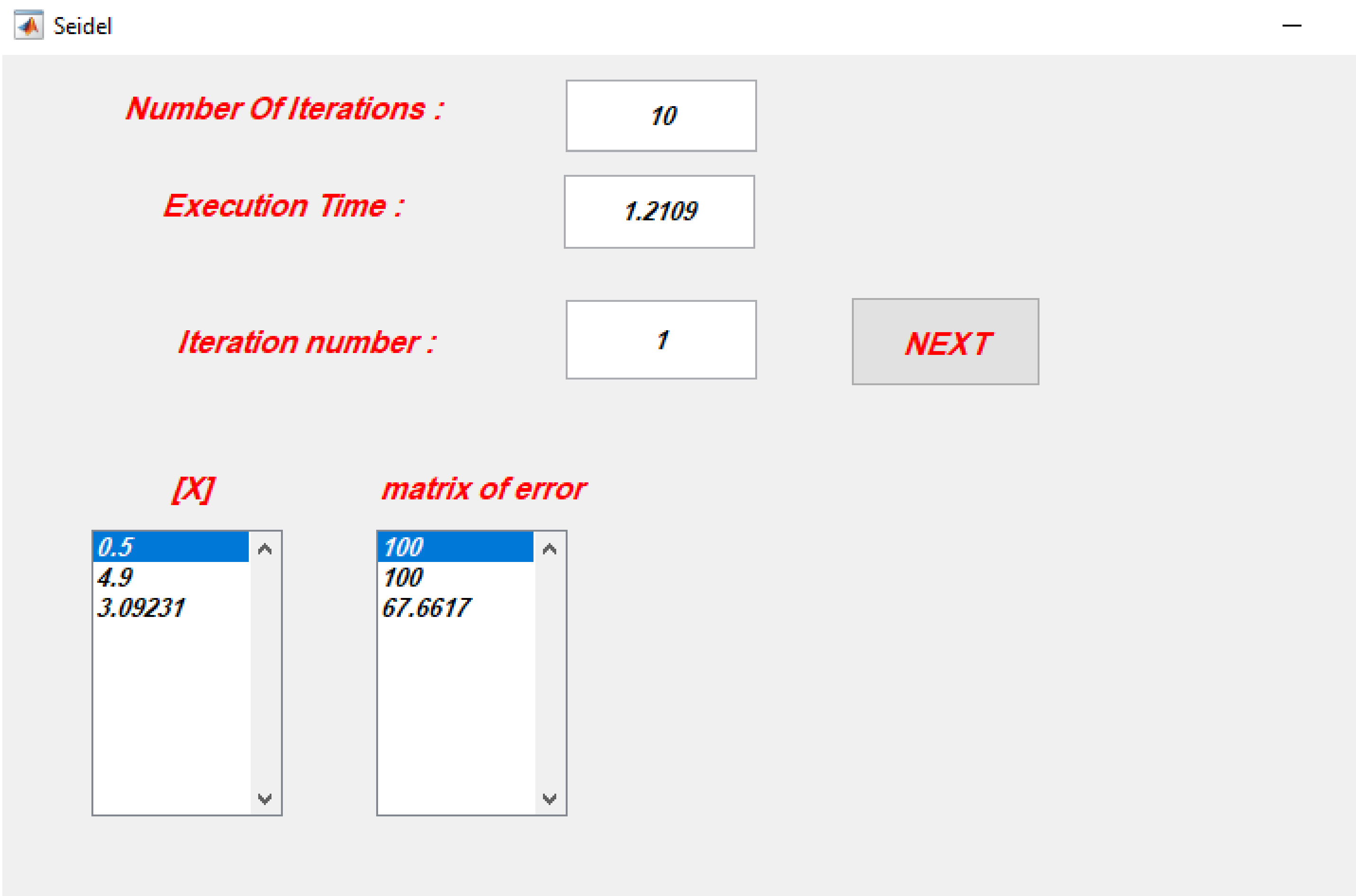


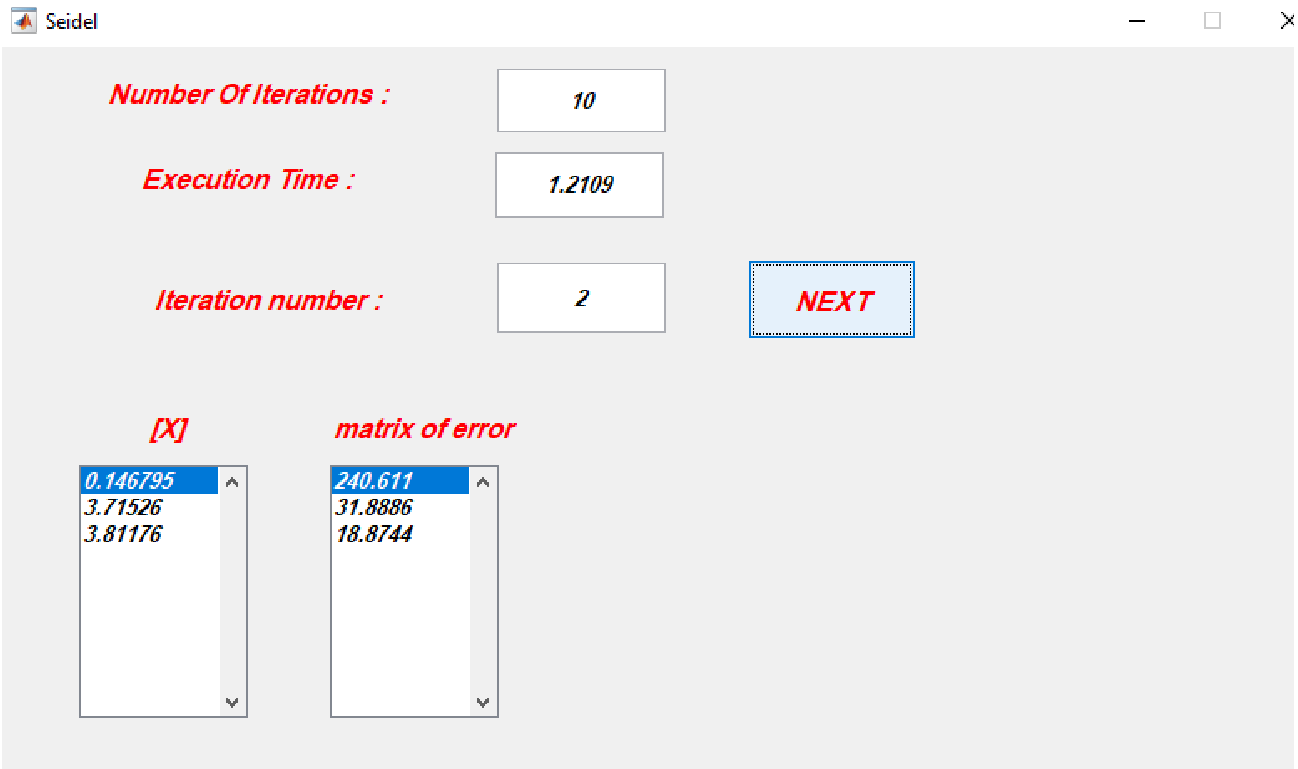


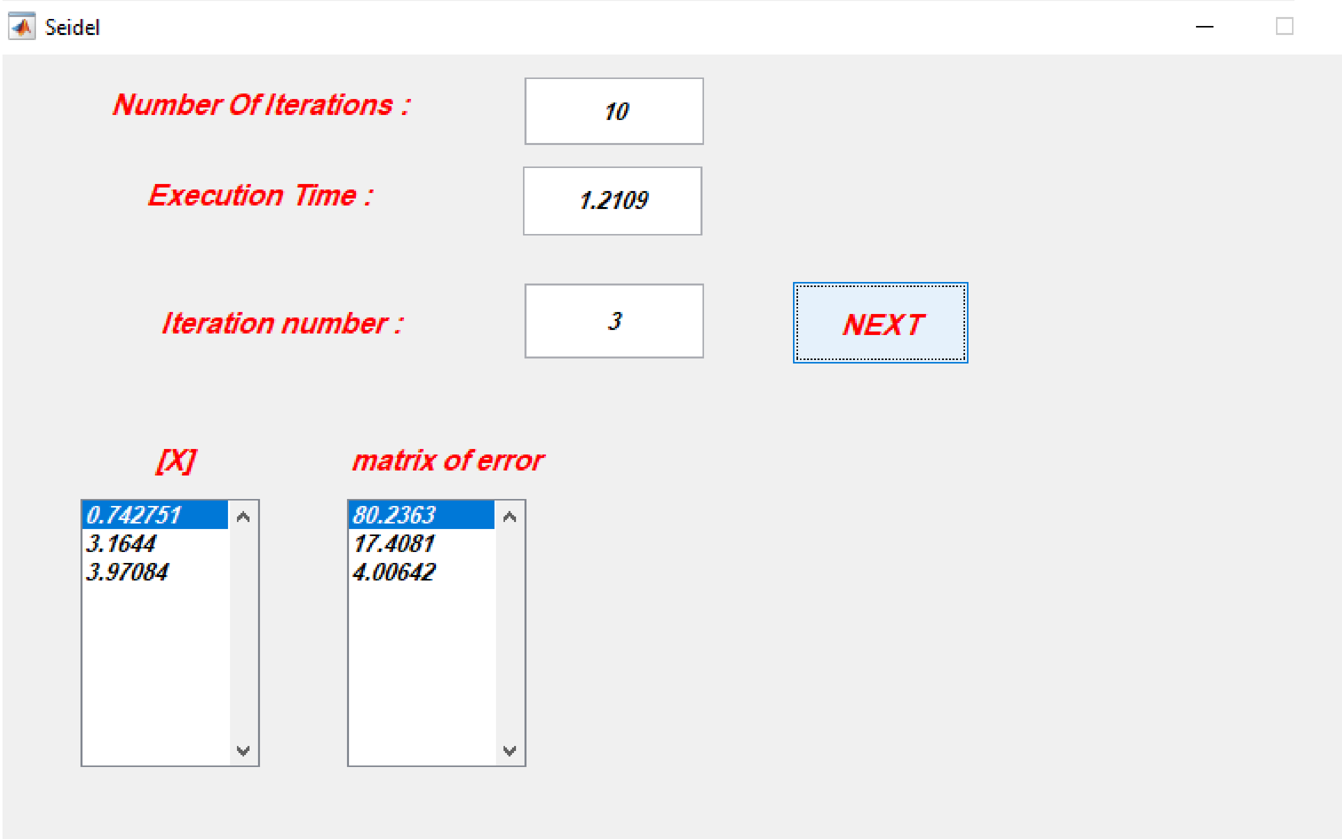

EnterEquations2
Number of Equation :
3
Equation_3
3*x1 + 7*x2 + 13*x3 = 76
Add
NEXT

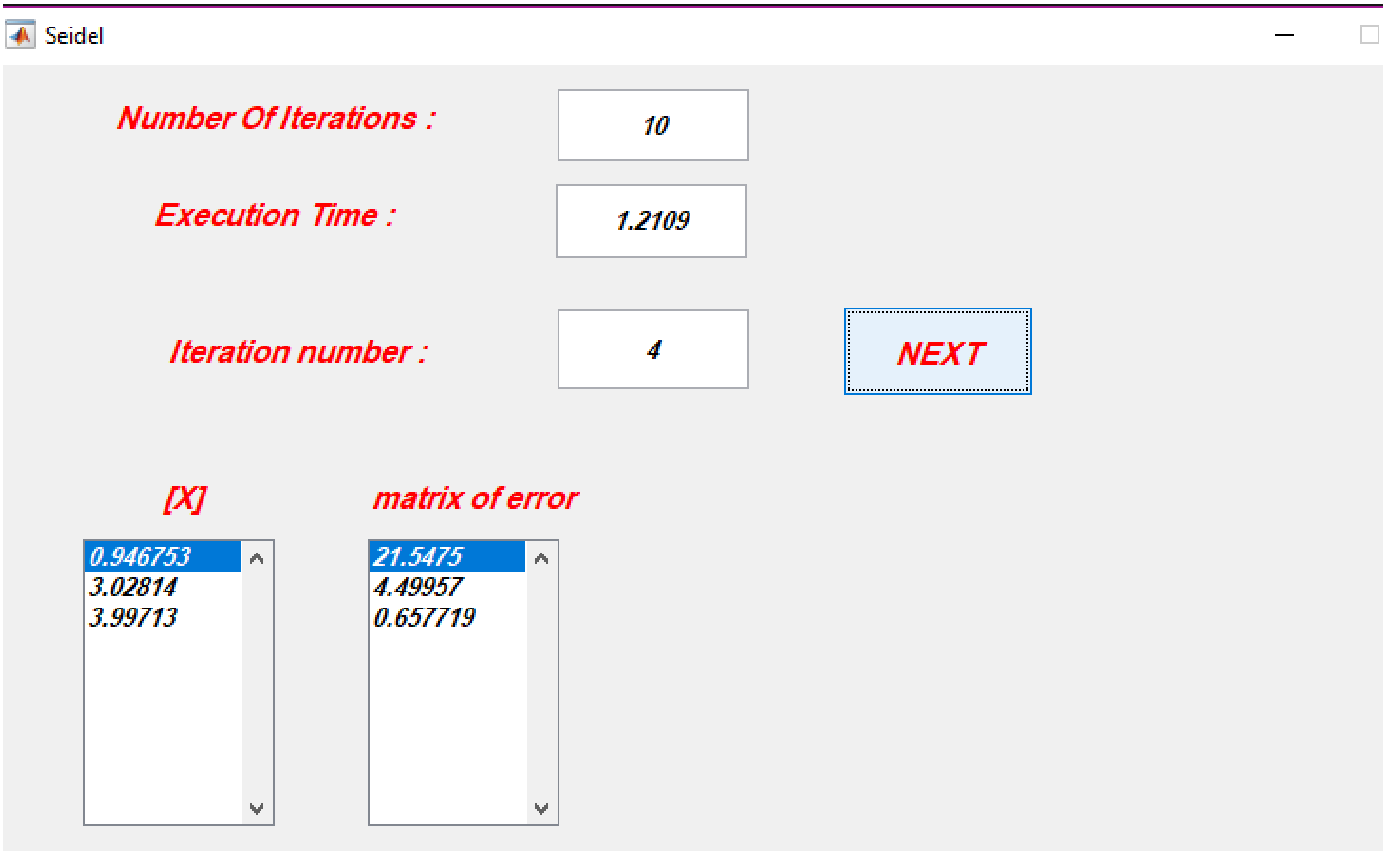

Data
Initial Points :
1 0 1
Max Iterations :
Epsilon :
NEXT

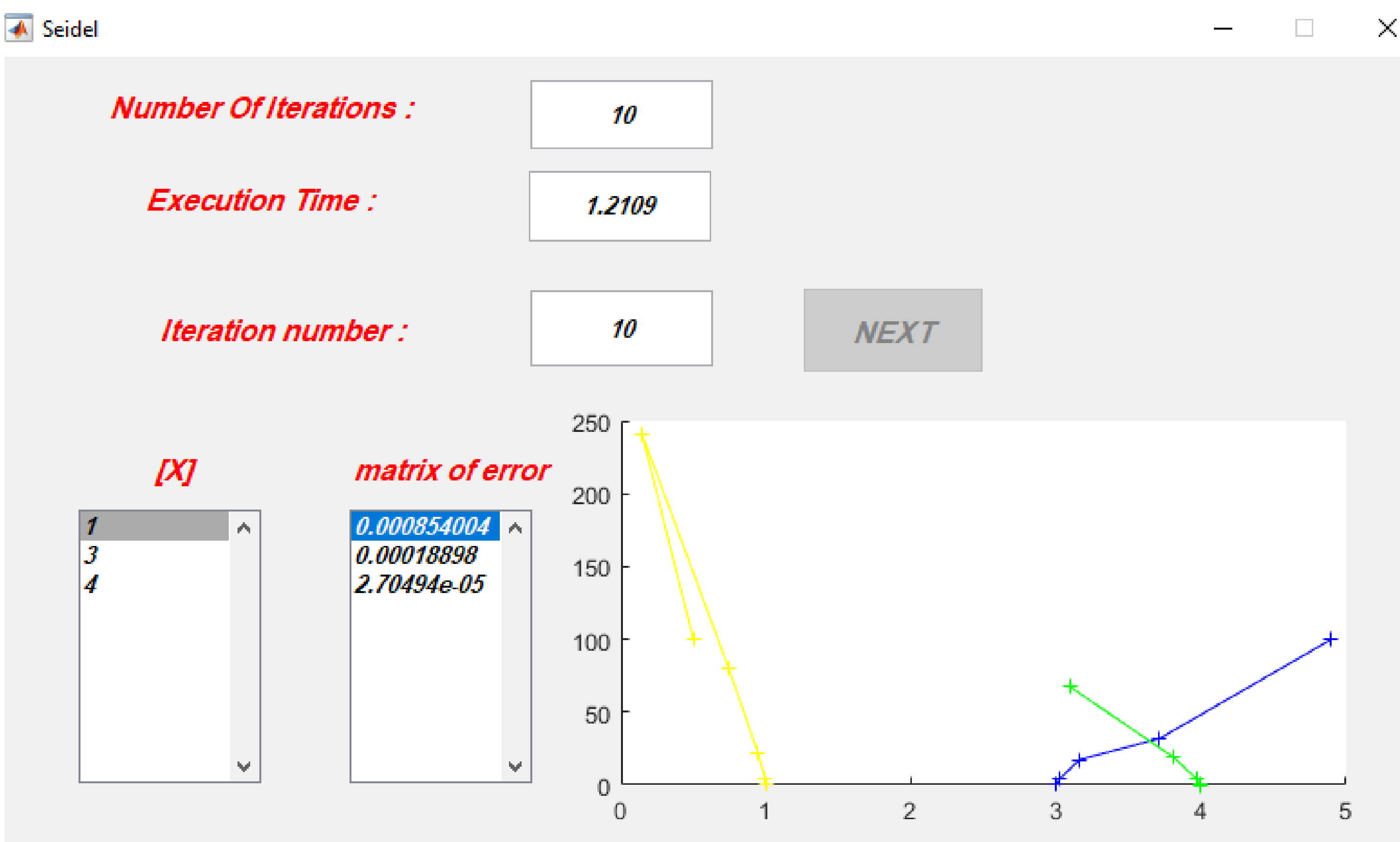

Data
Initial Points :
1 0 1
Max Iterations :
50
Epsilon :
.0001
NEXT


Seidel
Number Of Iterations :
10
Execution Time :
1.2109
Iteration number :
1
NEXT
[X]
matrix of error
0.5
4.9
3.09231
100
100
67.6617

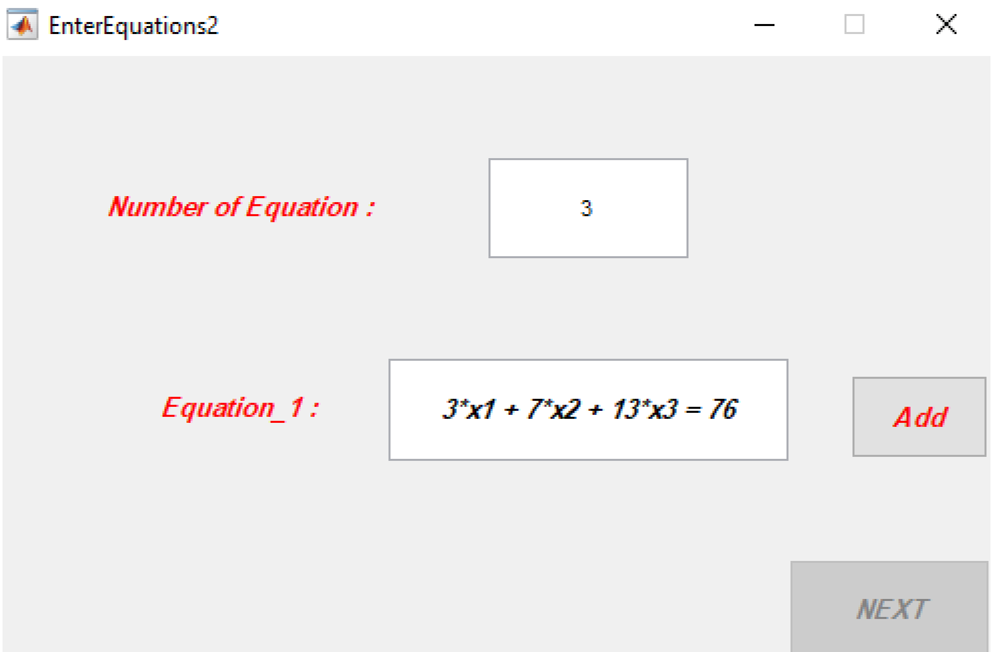

Seidel
Number Of Iterations :
10
Execution Time :
1.2109
Iteration number :
2
NEXT
[X]
0.146795
3.71526
3.81176
matrix of error
240.611
31.8886
18.8744

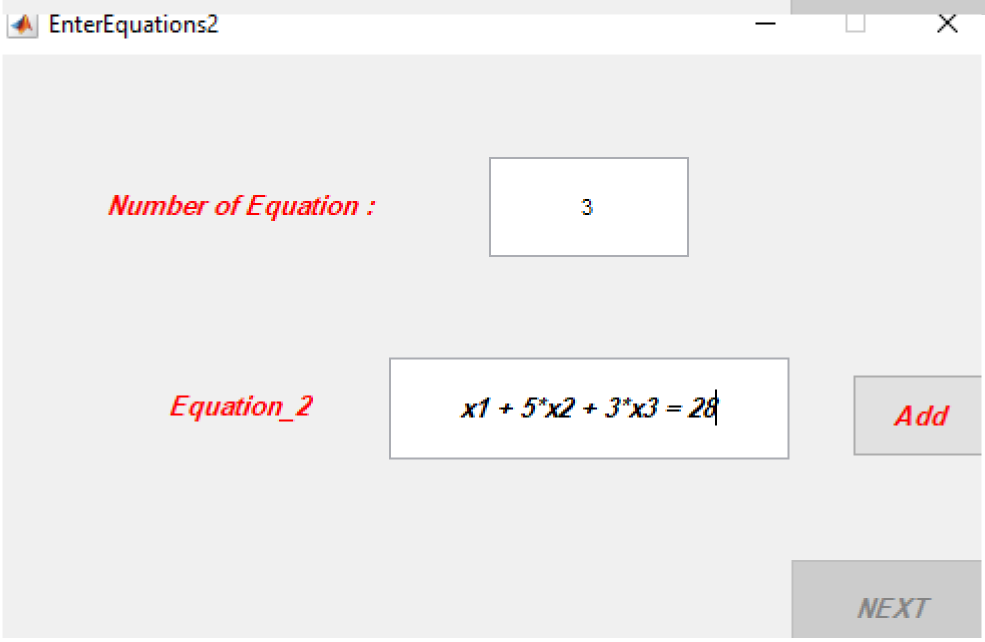

Seidel
Number Of Iterations :
10
Execution Time :
1.2109
Iteration number :
3
NEXT
[X]
0.742751
3.1644
3.97084
matrix of error
80.2363
17.4081
4.00642

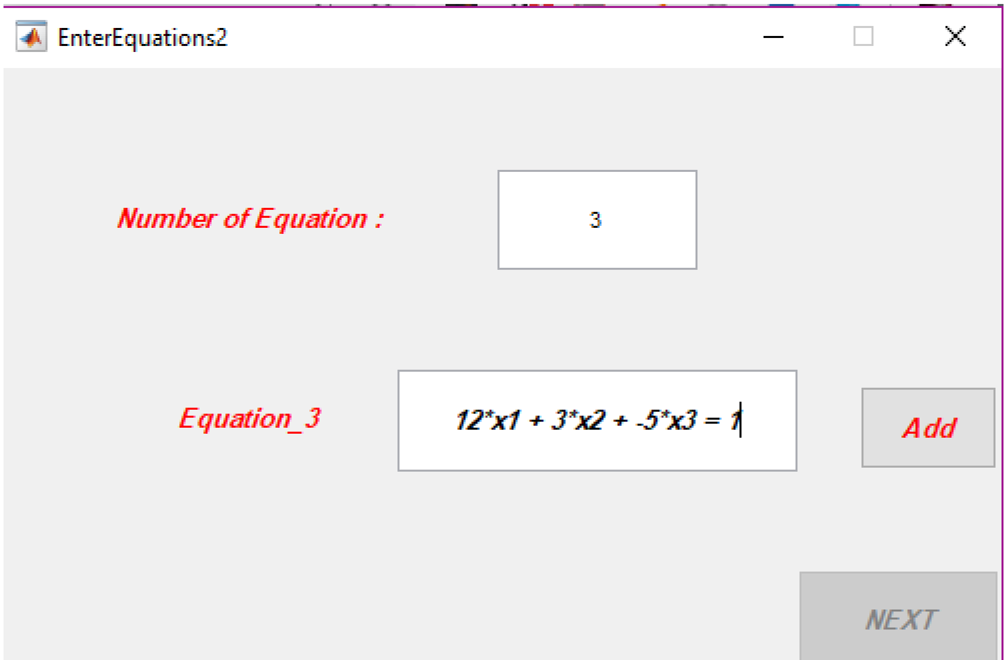

Seidel
Number Of Iterations :
10
Execution Time :
1.2109
Iteration number :
4
NEXT
[X]
0.946753
3.02814
3.99713
matrix of error
21.5475
4.49957
0.657719

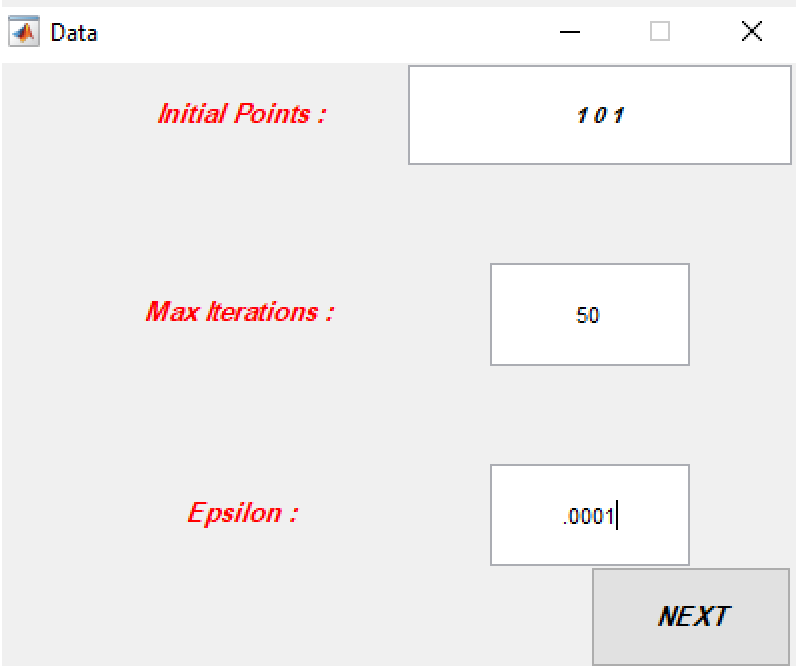

Seidel
Number Of Iterations :
10
Execution Time :
1.2109
Iteration number :
10
NEXT
[X]
1
3
4
matrix of error
0.000854004
0.00018898
2.70494e-05
250
200
150
100
50
0
0
1
2
3
4
5

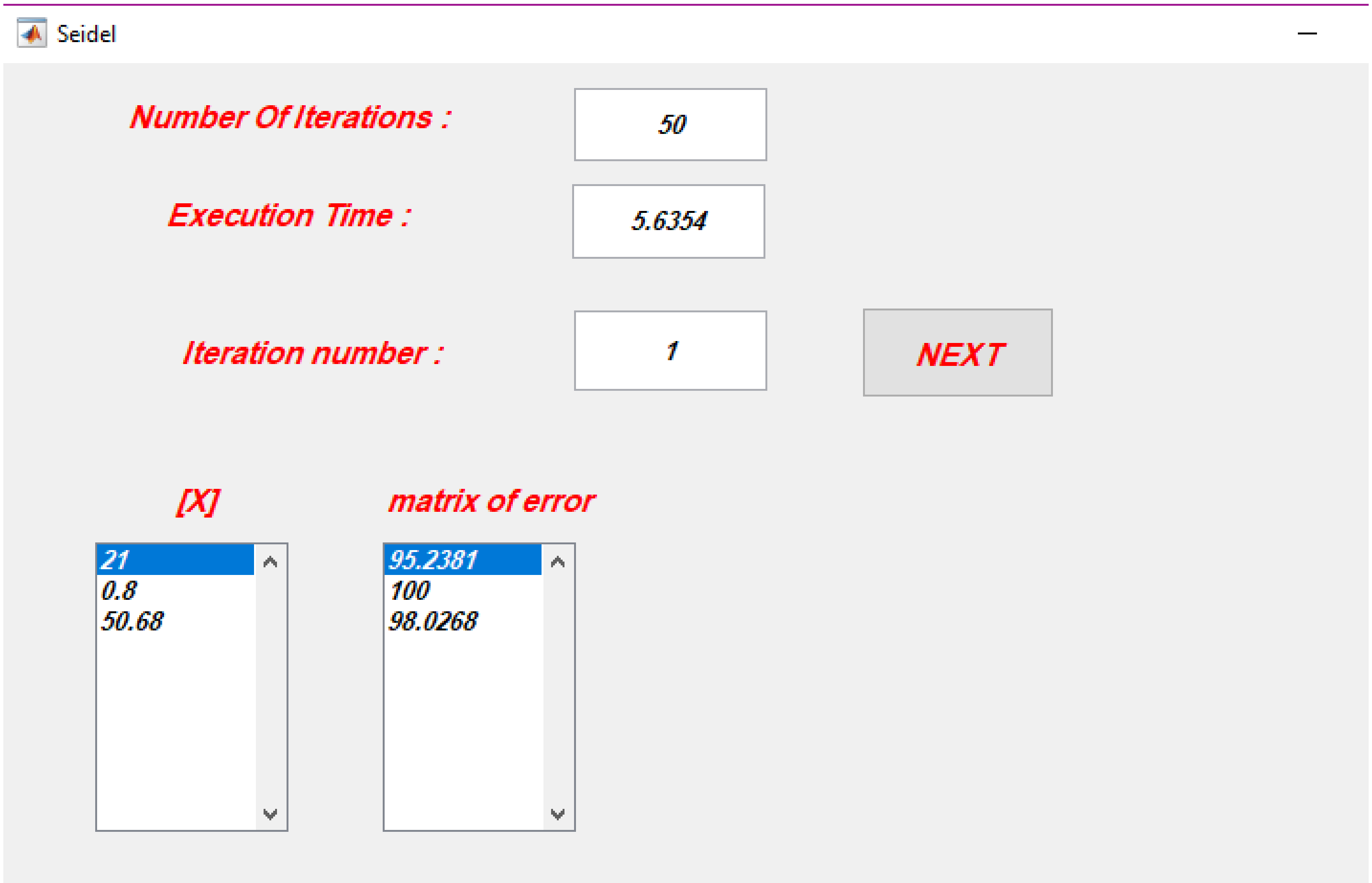
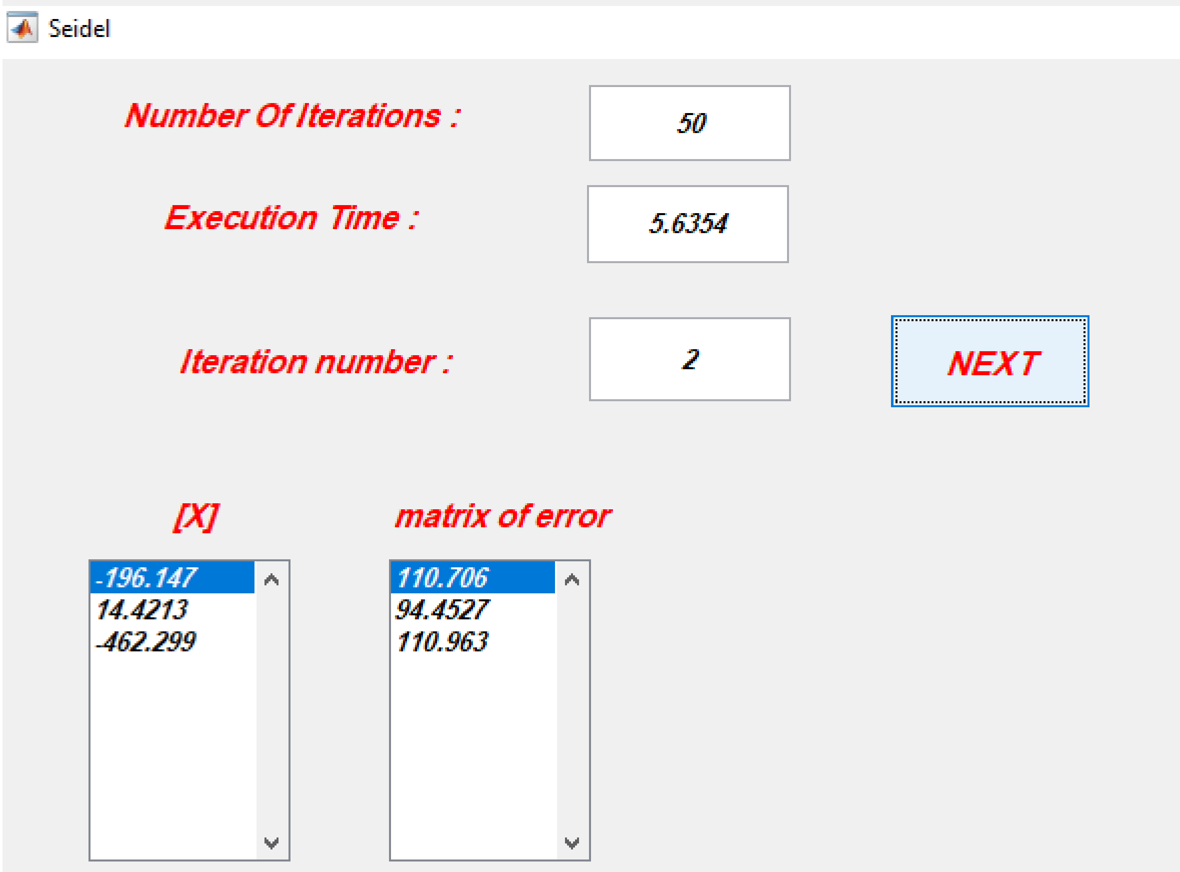
Which diverges :






EnterEquations2
Number of Equation :
3
Equation_3
12*x1 + 3*x2 + -5*x3 = 1
Add
NEXT


Data
Initial Points :
1 0 1
Max Iterations :
50
Epsilon :
.0001
NEXT


Seidel
Number Of Iterations :
50
Execution Time :
5.6354
Iteration number :
1
NEXT
[X]
21
0.8
50.68
matrix of error
95.2381
100
98.0268

Seidel
Number Of Iterations :
50
Execution Time :
5.6354
Iteration number :
2
NEXT
[X]
-196.147
14.4213
-462.299
matrix of error
110.706
94.4527
110.963

```
Error of each Iteration :
   ErrX1      ErrX2      ErrX3
   95.2381      100.0000      98.0268
   110.7063      94.4527      110.9626
   109.8320      112.4304      109.7983
   109.9010      109.6314      109.9046
   109.8947      109.9221      109.8944
   109.8953      109.8926      109.8954
   109.8953      109.8956      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
   109.8953      109.8953      109.8953
```

TEAM

Name	ID
ايمان رفيق عبد القادر محمد على	11
تقى علاء احمد الجندى	14
ميرنا محمد مصطفى اسماعيل مصطفى	٥٣
ندى سلامه محمد على	٥٥
يمنى جمال الدين محمود السيد	60