**Pseudo-code for each method**:

**Gaussian-elimination**:

For k = 1, ..., n − 1

Find ik ≥ k such that |aikk| = max k≤i≤n |aik|.

If ik > k, interchange akj ↔ aikj

for j = k, ..., n. For i = k + 1, ..., n

aik ← −aik/akk

For j = k + 1, ..., n

aij ← aij + aikakj

Row Operations on b:

For k = 1, ..., n − 1

If ik > k, interchange bk ↔ bik.

For i = k + 1, ..., n

bi ← bi + aikbk

Back Substitution:

bn ← bn/ann

For i = n − 1, ..., 1

bi ← (bi −∑n j=i+1 aijbj )/ aii

**LU decomposition**:

LUDecomp(a, b, n, x, tol, er) {

Declare s[n]

Declare o[n]

er = 0

Decompose(a, n, tol, o, s, er)

if (er != -1)

Substitute(a, o, n, b, x)

}

Decompose(a, n, tol, o, s, er) {

for i = 1 to n {

o[i] = i

s[i] = abs(a[i,1])

for j = 2 to n

if (abs(a[i,j]) > s[i])

s[i] = abs(a[i,j])

}

for k = 1 to n-1 {

Pivot(a, o, s, n, k)

if (abs(a[o[k],k]) / s[o[k]]) < tol) {

er = -1

return

}

for i = k+1 to n {

factor = a[o[i],k] / a[o[k],k]

a[o[i],k] = factor

for j = k+1 to n

a[o[i],j] = a[o[i],j] – factor \* a[o[k],j]

}

}

if (abs(a[o[n],n]) / s[o[n]]) < tol)

er = -1

}

Pivot(a, o, s, n, k) {

p = k

big = abs(a[o[k],k]) / s[o[k]])

for i = k+1 to n {

dummy = abs(a[o[i],k] / s[o(i)])

if (dummy > big) {

big = dummy

p = i

}

}

dummy = o[p]

o[p] = o[k]

o[k] = dummy

}

Substitute(a, o, n, b, x) {

Declare y[n]

y[o[1]] = b[o[1]]

for i = 2 to n {

sum = b[o[i]]

for j = 1 to i-1

sum = sum – a[o[i],j] \* b[o[j]]

y[o[i]] = sum

}

x[n] = y[o[n]] / a[o[n],n]

for i = n-1 downto 1 {

sum = 0

for j = i+1 to n

sum = sum + a[o[i],j] \* x[j]

x[i] = (y[o[i]] – sum) / a[o[i],i]

}

}

**Gaussian-Jordan**:

1. Start

2. Input the Augmented Coefficients Matrix (A):

For i = 1 to n

For j = 1 to n+1

Read Ai,j

Next j

Next i

3. Apply Gauss Jordan Elimination on Matrix A:

For i = 1 to n

If Ai,i = 0

Print "Mathematical Error!"

Stop

End If

For j = 1 to n

If i ≠ j

Ratio = Aj,i/Ai,i

For k = 1 to n+1

Aj,k = Aj,k - Ratio \* Ai,k

Next k

End If

Next j

Next i

4. Obtaining Solution:

For i = 1 to n

Xi = Ai,n+1/Ai,i

Next i

5. Display Solution:

For i = 1 to n

Print Xi

Next i

6. Stop

**Gauss-Seidel**:

read x, tolerance, iterations

counter = 0

E = tolerance + 1

while E > tolerance and counter < iterations

aux = x;

for i from 1 to n

acum1=0

for p from 1 to i - 1

acum1 = acum1 + A i p\*xp

end

acum2=0;

for q from i + 1 to n

acum2 = acum2 + A i q \*xq

end

xi = (b I - acum1 - acum2)/A i i

end

E = Norma(x1-x0)

counter = counter + 1

end

if E < tolerance

Show x0 is the solution whit an error of E

else

The program failures in these iterations

End

**Jacobi Iterative:**

read x0, tolerance, iterations

counter = 0

E = tolerance + 1

while E > tolerance y counter < iterations

for i from 1 to n

acum1=0

for p from 1 to i - 1

acum1 = acum1 + A i p\*x0p

end

acum2=0;

for q from i + 1 to n

acum2 = acum2 + A i q \*x0q

end

x1i = (b I - acum1 - acum2)/A i i

end

E = Norma(x1-x0)

x0 = x1

counter = counter + 1

end

if E < tolerance

Show x0 is the solution whit an error of E

else

The program failures in these iterations

End

**Analysis for the behavior of different examples:**

First example:

10\*x1+2\*x2-x3 = 27

-3\*x1-6\*x2+2\*x3 = -61.5

x1+x2+5\*x3 = -21.5

In case of Gaussian-elimination, after 3 steps elimination we reach the roots.

In case of LU decomposition , after 3 steps elimination of U and L we reach the roots.

In case of Gaussian-Jordan , after 6 steps elimination we reach the roots.

In case of Gauss-Seidel , it will take 4 iterations to reach the roots.

In case of Jacobi Iterative , it will take 4 iterations to reach the root.

Second example:

2\*x1+x2+x3 = 2

2\*x1+6\*x2+x3 = 3

x1+4\*x2-2\*x3 = 6

In case of Gaussian-elimination, after 3 steps elimination we reach the roots.

In case of LU decomposition , after 3 steps elimination of U and L we reach the roots.

In case of Gaussian-Jordan , after 6 steps elimination we reach the roots.

In case of Gauss-Seidel , it will take 9 iterations to reach the roots.

In case of Jacobi Iterative , it will take 57 iterations to reach the root.

**Problematic functions:**

First:

x1+ x2 + x3 = 4

2x1 – x2 – x3 = 7

x1– 2x2 – 2x3 = 12

we can not find the roots of this equations because this equations are singular where the determinant of the matrix is zero.

Second:

3\*x1+7\*x2+13\*x3 = 76

x1+5\*x2+3\*x3 = 28

12\*x1+3\*x2- 5\*x3 = 1

In case of Gauss-Seidel this equations diverge because the coefficient matix is not diagonally dominant but we we swap the row 3 with row 1 , it will become diagonally dominant and it will converge.

**Sample runs:**













