

Phase 1 Report

Team Members:

Badr Elsayed – 22010664

Abdel Rahman Nasr – 22010887

Lojine Sameh – 22011054

Nour Khaled – 22011319

Adham Anas – 22010601

1. Pseudo-Code:

For solving system of linear equations with alphabetical coefficients

Function alphabeticalSolution(matrix):

If number of rows of matrix != number of unique rows:

return "No Solution For Matrix"

Define symbolic variables for each of the unknowns based on the number of rows

variables = generate symbolic variables ('x', 'y', ...)

Convert string coefficients to symbolic variables

symbolic_matrix = Empty list

For each row i in matrix:

symbolic_row = Empty list

For each element j in row i:

If element is a string: Convert to symbolic variable and add to symbolic_row

Else: Add the numeric value to symbolic_row

Add symbolic_row to symbolic_matrix

Define the system of equations

equations = Empty list

Iterate through the matrix to create the equations

```

For each row i in symbolic_matrix:
    equation = 0
    For each element j in row i (except the last one):
        equation = equation + (element at i,j * corresponding variable in
        variables)
    Add (equation, last element in row i) to equations

# Solve the system of equations

solutions = Solve equations for variables using "sympy.solve()"

return solutions

```

Helper functions for different methods

Function `forward_Elimination(augmented_matrix,n):`

```

for i=0 to n-1:
    # Find the row with the maximum absolute value in column i
    max_row = row with maximum value in column i
Swap rows i and max_row

    # Check if the pivot element is zero or very close to zero
    if pivot element is too small (close to zero):
        return "Pivot element is zero or very small, cannot proceed."

    # Perform the row reduction
    for j = i + 1 to n-1:
        factor = augmented_matrix[j, i] / augmented_matrix[i, i]
        augmented_matrix[j, i:] -= factor * augmented_matrix[i, i:]

return augmented_matrix

```

Function `forward_substitution(augmented_matrix, n):`

```

x = array of zeroes with length n
for i=0 to n-1:
    if augmented_matrix[i, i] == 0:
        return "System has no unique solution."
    x[i] = augmented_matrix[i, -1] / augmented_matrix[i, i]
    for j=i + 1 to n-1:
        augmented_matrix[j, -1] -= augmented_matrix[j, i] * x[i]

return x, augmented_matrix

```

Function backward_Elimination(augmented_matrix,n):

```
    for i = n-1 to 0:
        Normalize the pivot row (make the pivot element equal to 1
        # Eliminate the elements above the pivot
        for j=i-1 to 0: # Go through rows above the pivot
            factor = augmented_matrix[j, i]
            each element in row j from i to n-1 -= factor * each element in row i from
i to n-1
    return augmented_matrix
```

Function backward_substitution(augmented_matrix,n):

```
    x = array of zeroes with length n
    for i = n-1 to 0:
        if augmented_matrix[i, i] == 0:
            return "System has no unique solution."
        x[i] = augmented_matrix[i, n] / augmented_matrix[i, i]
        Update augmented_matrix for rows above the pivot

    return x , augmented_matrix
```

The methods

Function Gauss_Elimination(A, B):

```
    n = number of rows of B
    augmented_matrix = append B as last column to A
    augmented_matrix = forward_Elimination(augmented_matrix, n)
    x, augmented_matrix = backward_substitution(augmented_matrix, n)
    return x, augmented_matrix
```

Function Gauss_Jordan_Elimination(A, B):

```
    augmented_matrix = append B as last column to A
    augmented_matrix = forward_Elimination(augmented_matrix, n)
    augmented_matrix = backward_Elimination(augmented_matrix, n)
    x = last column of augmented_matrix
    return x, augmented_matrix
```

Function Jacobi(self, A: np.ndarray, b: np.ndarray, epsilon=1e-9, iterations=50,
x=None, mode=2):

```
    # 'mode' is what determines whether to use iterations or epsilon for solving
    # 'mode' = 1 is iterations, 'mode' = 2 is epsilon (absolute relative error)
    # if 'mode' is negative it means Gauss Seidel is applied and not Jacobi
```

```

max_its = 2000
GaussSeidel = False
if (mode < 0):
    GaussSeidel = True
    mode = abs(mode)

if (mode != 1 and mode != 2):
    print("Unknown Mode choosen in Jacobi.")

if (x is None):
    x = zeroes with length rows of A

D = diagonal of A
if an element in D is 0
    print("Matrix contains zero diagonal elements, Jacobi method cannot
proceed.")

curr_it = 0
while (True):
    x_new = x.copy()
    for each row i A:
        x_new[i] = b[i]
        for each column j in A:
            if (i == j):
                continue
            if (GaussSeidel):
                x_new[i] -= A[i][j] * x_new[j]
            else:
                x_new[i] -= A[i][j] * x[j]
        x_new[i] /= A[i][i]

    if (mode == 1):
        if (curr_it > iterations):
            return x_new, curr_it-1

        if (curr_it > max_its or there's an overflow):
            if (GaussSeidel): print("Divergence occurred in Gauss Seidel")
            else: print("Divergence occurred in Jacobi")
            return x_new, curr_it-1

    else if (mode == 2):
        condition = True

```

```

for each row i in A:
    if (not (absolute(x[i] - x_new[i]) < epsilon)):
        condition = False
if (condition):
    return x_new, curr_it-1

if (curr_it > max_its or there's an overflow):
    if (GaussSeidel): print("Divergence occurred in Gauss Seidel")
    else: print("Divergence occurred in Jacobi")
    return x_new, curr_it-1

x = x_new
curr_it += 1

```

Function GaussSeidel(A, b, epsilon=1e-9, iterations=50, x=None, mode=2):

```

return Jacobi(A, b, epsilon, iterations, x, mode = -mode)

```

Function LUCroutsForm(A, B):

```

L = zeroes matrix with same size of A
U = zeroes matrix with same size of A
L_and_U = zeroes matrix with same size of A
n = number of rows of B

```

```

for i=0 to n-1:
    for j=0 to n-1:
        L[i][j] = A[i][j]
        A[i] -= L[i][j] * U[j]

L[i][i] = A[i][i] #i==j (diagonal)

if L[i][i] != 0:
    U[i] = A[i] / L[i][i]
else: # infinite number of solution or no solution
    return "System has no unique solution or no solution."

```

```

#solve the equation
augmented_L = add column B to L
Y, augmented_L = forward_substitution(augmented_L,n)
augmented_U = add column Y to U
X, augmented_U = backward_substitution(augmented_U,n)
for i=0 to n-1:

```

```

    for j=0 to n-1:
        if i <= j:
            L_and_U[i][j] = U[i][j]
        else:
            L_and_U[i][j] = L[i][j]

    return X, L_and_U

```

Function LUCholeskyForm(A, B):

```

    L = zeroes matrix with same size of A
    If matrix A isn't positive definite:
        Print("Error")
        return

    for i=0 to n-1:
        Update L matrix using Cholesky method
        U= Transpose of L  #U = L transpose
        n=number of rows of B
        #solve the equation
        augmented_L = add column b to L
        Y, augmented_L = forward_substitution(augmented_L,n)
        augmented_U = add column Y to U
        X, augmented_U = backward_substitution(augmented_U,n)

    return X,L

```

Function LUDoolittlesForm(A, B):

```

    L = zeroes matrix with same size of A
    U = zeroes matrix with same size of A
    L_and_U = zeroes matrix with same size of A
    n = number of rows of B

    for i=0 to n-1:
        L[i][i] = 1
        # Compute Upper Triangular Matrix
        for j=i to n-1:
            sum = 0
            for k=0 to i-1:
                sum += L[i][k] * U[k][j]
            U[i][j] = A[i][j] - sum

        if U[i, i] == 0:

```

```

    return "Matrix is singular, no unique solution."

# Compute Lower Triangular Matrix
for j=i+1 to n-1:
    sum = 0
    for k=0 to i-1:
        sum += L[j][k] * U[k][i]
    L[j][i] = (A[j][i] - sum) / U[i][i]

# Storing L and U in one matrix
for i=0 to n-1:
    for j=0 to n-1:
        if i <= j:
            L_and_U[i][j] = U[i][j]
        else:
            L_and_U[i][j] = L[i][j]

#solve the equation
augmented_LB = add colomn B to L
Y , augmented_LB = forward_substitution(augmented_LB, n)
if isinstance(Y, str):
    return Y
else:
    augmented_UY = add coloumn Y to U
    X , augmented_UY = backward_substitution(augmented_UY, n)
return X , L_and_U

```

2. Sample runs for each method:

❖ General Cases Independent of Method:

1- A singular matrix will return an error

Choose Size

4

1	x1+	2	x2+	3	x3+	4	x4 =	-5
2	x1+	4	x2+	6	x3+	8	x4 =	-10
4	x1+	-8	x2+	5	x3+	3	x4 =	4
2	x1+	0	x2+	4	x3+	7	x4 =	0

Choose Operation

Gauss Elimination

Significant Digits

4

Answer

Result

Error was found : Matrix is singular , no unique solution exists

2- Entered a non-number or left some fields

Choose Size

4

a

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

vas

x2+

s

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

a

x4 =

16

1

x1+

1

x2+

1

x3+

a

x4 =

17

Choose Operation

LU Decomposition

Choose LU Sub Operation

Cholesky Decomposition

Significant Digits

4

Answer

Please only enter numerals

❖ Gauss Elimination

1-

Choose Size
4

1

x1+

2

x2+

3

x3+

4

x4 =

-5

2

x1+

4

x2+

6

x3+

6

x4 =

-10

4

x1+

-8

x2+

5

x3+

3

x4 =

4

2

x1+

0

x2+

4

x3+

7

x4 =

0

Choose Operation
Gauss Elimination

Significant Digits
4

Answer

Result

Result X Vector :

32.0000
5.5000
-16.0000
0.0000

Resultant Matrix :

4.0000 -8.0000 5.0000 3.0000 128.0000
0.0000 8.0000 3.5000 4.5000 44.0000
0.0000 0.0000 -0.2500 3.2500 4.0000
0.0000 0.0000 0.0000 1.0000 0.0000

Time taken = 0.009709358215332031

2-

Choose size
3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation
Gauss Elimination

Significant Digits
4

Answer

Result

Result X Vector :

0.2558
1.3023
0.8372

Resultant Matrix :

4.0000 1.0000 2.0000 1.0233
0.0000 2.7500 0.5000 3.5814
0.0000 0.0000 3.9091 3.2727

Time taken = 1e-06

3-

Matrix Solver
Alphabetical Coefficients Solver

Choose Size
4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation
Gauss Elimination

Significant Digits
4

Answer

Result

Result X Vector :

0.9449

0.8640

0.8931

0.7943

Resultant Matrix :

10.0000	1.0000	1.0000	1.0000	9.4486
0.0000	11.9000	0.9000	0.9000	10.2813
0.0000	0.0000	14.8319	0.8319	13.2467
0.0000	0.0000	0.0000	17.7853	14.1275

Time taken = 1e-06

❖ Gauss-Jordan

1-

Choose Size
4

1

x1+

2

x2+

3

x3+

4

x4 =

-5

2

x1+

4

x2+

6

x3+

6

x4 =

-10

4

x1+

-8

x2+

5

x3+

3

x4 =

4

2

x1+

0

x2+

4

x3+

7

x4 =

0

Choose Operation
Gauss-Jordan Elimination

Significant Digits
4

Answer

Result

Result X Vector :

32.0000

5.5000

-16.0000

0.0000

Resultant Matrix :

1.0000	0.0000	0.0000	0.0000	32.0000
0.0000	1.0000	0.0000	0.0000	5.5000
0.0000	0.0000	1.0000	0.0000	-16.0000
0.0000	0.0000	0.0000	1.0000	0.0000

Time taken = 0.006777763366699219

2-

Choose Size
3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation
Gauss-Jordan Elimination

Significant Digits
4

Answer

Result

Result X Vector :

0.2558
1.3023
0.8372

Resultant Matrix :

1.0000	0.0000	0.0000	0.2558
0.0000	1.0000	0.0000	1.3023
0.0000	0.0000	1.0000	0.8372

Time taken = 1e-06

3-

Matrix Solver
Alphabetical Coefficients Solver

Choose Size
4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation
Gauss-Jordan Elimination

Significant Digits
4

Answer

Result

Result X Vector :

0.9449
0.8640
0.8931
0.7943

Resultant Matrix :

1.0000	0.0000	0.0000	0.0000	0.9449
0.0000	1.0000	0.0000	0.0000	0.8640
0.0000	0.0000	1.0000	0.0000	0.8931
0.0000	0.0000	0.0000	1.0000	0.7943

Time taken = 1e-06

❖ Jacobi

1- Diverge

Choose size

4

1	x1+	2	x2+	3	x3+	4	x4 =	-5
2	x1+	4	x2+	6	x3+	6	x4 =	-10
4	x1+	-8	x2+	5	x3+	3	x4 =	4
2	x1+	0	x2+	4	x3+	7	x4 =	0

Choose Operation

Jacobi

Initial Guess

0

0

0

0

Iterative

500

Significant Digits

4

Answer

Result

Result X Vector :

2326250731251412995330617518399079994324498435887568263711250022481391485869717520731893655694653573491447370123441098756380965257590341632.0000
1092239547566579656385969862481003655000728933394384665198153228845702850800277963903359996964447334723843052110103546331752040902086885376.0000
190166138380223809918959023105985538004371173617754332247695166084043924467633601810048840805392725115736591218001551579051255681448411136.0000
409497900597073040922091934541855879499970017694585734734532109698696438170333168383123961769923161134351064396470307356790185191683391488.0000

Number of iterations taken = 500

Time taken = 0.01914191246032715

2- Converge

Choose Size
3

4	x1+	1	x2+	2	x3 =	4
1	x1+	3	x2+	1	x3 =	5
2	x1+	1	x2+	5	x3 =	6

Choose Operation
Jacobi

Initial Guess
0 0 0

Iterative

100

Significant Digits
4

Answer

Result

Result X Vector :

0.2558

1.3023

0.8372

Number of iterations taken = 100

Time taken = 0.0010123252868652344

Matrix Solver Alphabetic Coefficients Solver

Choose Size
3

4	x1+	1	x2+	2	x3 =	4
1	x1+	3	x2+	1	x3 =	5
2	x1+	1	x2+	5	x3 =	6

Choose Operation
Jacobi

Initial Guess
0 0 0

Relative Error

0.005

Significant Digits
4

Answer

Result

Result X Vector :

0.2578

1.3042

0.8389

Number of iterations taken = 13

Time taken = 0.0014445781707763672

3- Converge

Matrix Solver Alphabetical Coefficients Solver

Choose Size
4

10

 x1+

1

 x2+

1

 x3+

1

 x4 =

12

1

 x1+

12

 x2+

1

 x3+

1

 x4 =

13

1

 x1+

1

 x2+

15

 x3+

1

 x4 =

16

1

 x1+

1

 x2+

1

 x3+

18

 x4 =

17

Choose Operation
Jacobi

Initial Guess

1

1

1

1

Iterative

150

Significant Digits
4

Answer

Result
Result X Vector :
0.9449
0.8640
0.8931
0.7943
Number of iterations taken = 150
Time taken = 0.004008769989013672

Okay

Matrix Solver Alphabetical Coefficients Solver

Choose Size
4

10

 x1+

1

 x2+

1

 x3+

1

 x4 =

12

1

 x1+

12

 x2+

1

 x3+

1

 x4 =

13

1

 x1+

1

 x2+

15

 x3+

1

 x4 =

16

1

 x1+

1

 x2+

1

 x3+

18

 x4 =

17

Choose Operation
Jacobi

Initial Guess

1

1

1

1

Relative Error

0.005

Significant Digits
4

Answer

Result
Result X Vector :
0.9453
0.8643
0.8934
0.7946
Number of iterations taken = 2
Time taken = 0.0005068778991699219

❖ Gauss-Seidel

1- Diverge

Choose Size

4

1

x1+

2

x2+

3

x3+

4

x4 =

-5

2

x1+

4

x2+

6

x3+

6

x4 =

-10

4

x1+

-8

x2+

5

x3+

3

x4 =

4

2

x1+

0

x2+

4

x3+

7

x4 =

0

Choose Operation

Gauss-Seidel

Initial Guess

0

0

0

0

Iterative

500

Significant Digits

4

Answer

Result

Result X Vector :

```
-172840394378253134622206159514502192125742266770466204642142040226791424.0000  
-19991367801321876453660604394203823168515316082689625881748734788763648.0000  
112866290974287486871277855941224422320031992718687123572026369657274368.0000  
-15112053591520526338668752599921248351718792276436117068396840987131904.0000
```

Number of iterations taken = 500

Time taken = 0.010123729705810547

2- Converge

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation

Gauss-Seidel

Initial Guess

0

0

0

Iterative

100

Significant Digits

4

Answer

Result

Result X Vector :

0.2558

1.3023

0.8372

Number of iterations taken = 100

Time taken = 0.0010104179382324219

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation

Gauss-Seidel

Initial Guess

0

0

0

Relative Error

0.005

Significant Digits

4

Answer

Result

Result X Vector :

0.2557

1.3028

0.8372

Number of iterations taken = 3

Time taken = 1e-06

3- Converge

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation

Gauss-Seidel

Initial Guess

1

1

1

1

Iterative

150

Significant Digits

4

Answer

Result

Result X Vector :

0.9449

0.8640

0.8931

0.7943

Number of iterations taken = 150

Time taken = 0.002505064010620117

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation

Gauss-Seidel

Initial Guess

1

1

1

1

Relative Error

0.005

Significant Digits

4

Answer

Result

Result X Vector :

0.9449

0.8640

0.8931

0.7943

Number of iterations taken = 1

Time taken = 0.0005359649658203125

❖ LU Decomposition

➤ Doolittle

1-

Choose Size
4

1

 x1+

2

 x2+

3

 x3+

4

 x4 =

-5

2

 x1+

4

 x2+

6

 x3+

6

 x4 =

-10

4

 x1+

-8

 x2+

5

 x3+

3

 x4 =

4

2

 x1+

0

 x2+

4

 x3+

7

 x4 =

0

Choose Operation
LU Decomposition

Choose LU Sub Operation
Doolittle Decomposition

Significant Digits
4

Answer

Result

Result X Vector :

32.0000
5.5000
-16.0000
0.0000

Resultant Matrix :

4.0000 -8.0000 5.0000 3.0000
0.5000 8.0000 3.5000 4.5000
0.5000 0.5000 -0.2500 3.2500
0.2500 0.5000 -0.0000 1.0000

Time taken = 1e-06

2-

Matrix Solver

Choose Size
3

4

 x1+

1

 x2+

2

 x3 =

4

1

 x1+

3

 x2+

1

 x3 =

5

2

 x1+

1

 x2+

5

 x3 =

6

Choose Operation
LU Decomposition

Choose LU Sub Operation
Doolittle Decomposition

Significant Digits
4

Answer

Result

Result X Vector :

0.2558
1.3023
0.8372

Resultant Matrix :

4.0000 1.0000 2.0000
0.2500 2.7500 0.5000
0.5000 0.1818 3.9091

Time taken = 0.001027822494506836

3-

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation

LU Decomposition

Choose LU Sub Operation

Doolittle Decomposition

Significant Digits

4

Answer

Result

Result X Vector :

0.9449

0.8640

0.8931

0.7943

Resultant Matrix :

10.0000 1.0000 1.0000 1.0000

0.1000 11.9000 0.9000 0.9000

0.1000 0.0756 14.8319 0.8319

0.1000 0.0756 0.0561 17.7853

Time taken = 1e-06

➤ Crout

1-

Choose Size

4

1

x1+

2

x2+

3

x3+

4

x4 =

-5

2

x1+

4

x2+

6

x3+

6

x4 =

-10

4

x1+

-8

x2+

5

x3+

3

x4 =

4

2

x1+

0

x2+

4

x3+

7

x4 =

0

Choose Operation

LU Decomposition

Choose LU Sub Operation

Crout Decomposition

Significant Digits

4

Answer

Result

Error was found : System has no unique solution or no solution.

2-

Matrix Solver
Alphabetical Coefficients Solver

Choose Size
3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation
LU Decomposition

Choose LU Sub Operation
Crout Decomposition

Significant Digits
4

Answer

Result

Result X Vector :
0.2558
1.3023
0.8372

Resultant Matrix :
1.0000 0.2500 0.5000
1.0000 1.0000 0.1818
2.0000 0.5000 1.0000

Time taken = 1e-06

3-

Matrix Solver
Alphabetical Coefficients Solver

Choose Size
4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation
LU Decomposition

Choose LU Sub Operation
Crout Decomposition

Significant Digits
4

Answer

Result

Result X Vector :
0.9449
0.8640
0.8931
0.7943

Resultant Matrix :
1.0000 0.1000 0.1000 0.1000
1.0000 1.0000 0.0756 0.0756
1.0000 0.9000 1.0000 0.0561
1.0000 0.9000 0.8319 1.0000

Time taken = 1e-06

➤ Cholesky

1-

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

4

1

x1+

2

x2+

3

x3+

4

x4 =

-5

2

x1+

4

x2+

6

x3+

6

x4 =

-10

4

x1+

-8

x2+

5

x3+

3

x4 =

4

2

x1+

0

x2+

4

x3+

7

x4 =

0

Choose Operation

LU Decomposition

Choose LU Sub Operation

Cholesky Decomposition

Significant Digits

4

Answer

Result

Error was found : matrix isnt positive definite

2-

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

4

x1+

1

x2+

2

x3 =

4

1

x1+

3

x2+

1

x3 =

5

2

x1+

1

x2+

5

x3 =

6

Choose Operation

LU Decomposition

Choose LU Sub Operation

Cholesky Decomposition

Significant Digits

4

Answer

Result

Result X Vector :

0.2558

1.3023

0.8372

L Matrix :

2.0 0.0 0.0

0.5 1.6583123951777 0.0

1.0 0.30151134457776363 1.9771421064483223

Time taken = 0.0015096664428710938

3-

Matrix Solver

Alphabetical Coefficients Solver

Choose size

4

10

x1+

1

x2+

1

x3+

1

x4 =

12

1

x1+

12

x2+

1

x3+

1

x4 =

13

1

x1+

1

x2+

15

x3+

1

x4 =

16

1

x1+

1

x2+

1

x3+

18

x4 =

17

Choose Operation

LU Decomposition

Choose LU Sub Operation

Cholesky Decomposition

Significant Digits

4

Answer

Result

Result X Vector :

0.9449

0.8640

0.8931

0.7943

L Matrix :

3.1622776601683795 0.0 0.0 0.0

0.31622776601683794 3.449637662132068 0.0 0.0

0.31622776601683794 0.2608969660436018 3.8512248406330736 0.0

0.31622776601683794 0.2608969660436018 0.21601771060774747 4.217258484111809

Time taken = 1e-06

❖ Bonus: Coefficients can be letters

1-

2

a

x1+

b

x2 =

1

c

x1+

d

x2 =

2

Answer

Result

$$X = (1*d - 2*b)/(a*d - b*c)$$

$$y = (-1*c + 2*a)/(a*d - b*c)$$

2-

3

a	x1 +	b	x2 +	c	x3 =	d
e	x1 +	f	x2 +	g	x3 =	h
i	x1 +	j	x2 +	k	x3 =	l

Answer

Result

$$X = \frac{(b \cdot g \cdot l - b \cdot h \cdot k - c \cdot f \cdot l + c \cdot h \cdot j + d \cdot f \cdot k - d \cdot g \cdot j)}{(a \cdot f \cdot k - a \cdot g \cdot j - b \cdot e \cdot k + b \cdot g \cdot i + c \cdot e \cdot j - c \cdot f \cdot i)}$$

$$y = \frac{(-a \cdot g \cdot l + a \cdot h \cdot k + c \cdot e \cdot l - c \cdot h \cdot i - d \cdot e \cdot k + d \cdot g \cdot i)}{(a \cdot f \cdot k - a \cdot g \cdot j - b \cdot e \cdot k + b \cdot g \cdot i + c \cdot e \cdot j - c \cdot f \cdot i)}$$

$$z = \frac{(a \cdot f \cdot l - a \cdot h \cdot j - b \cdot e \cdot l + b \cdot h \cdot i + d \cdot e \cdot j - d \cdot f \cdot i)}{(a \cdot f \cdot k - a \cdot g \cdot j - b \cdot e \cdot k + b \cdot g \cdot i + c \cdot e \cdot j - c \cdot f \cdot i)}$$

3. Comparison between different methods (time complexity, convergence, best and approximate errors):

Method	Time Complexity	Convergence	Error
Gauss Elimination	$O(n^3)$	Direct method, no convergence issue	Numerical errors for ill-conditioned matrices
Gauss-Jordan	$O(n^3)$	Direct method, no convergence issue	Numerical errors for ill-conditioned matrices
Jacobi	$O(n^2)$ / Iteration	Converges for diagonally dominant or	Slower convergence than Gauss-Seidel

		positive-definite matrices	
Gauss-Seidel	$O(n^2)$ / Iteration	Converges for diagonally dominant or positive-definite matrices	Error decreases with each iteration
LU Decomposition	$O(n^3)$	Converges for non-singular matrices	Sensitive to ill-conditioned matrices

4. Data structures used:

- ❖ 1D Lists (Arrays)
- ❖ 2D Lists (Matrices)
- ❖ Temporary (Scalar) variables
- ❖ Flags (Booleans)
- ❖ Indices

5. Test cases:

- ❖ 1st case
 - Gauss Elimination, precision = 4

Matrix Solver

Alphabetical Coefficients Solver

5

2

x1

1

x2

1

x3

1

x4

1

x5

=

4

1

x1

2

x2

1

x3

1

x4

1

x5

=

5

1

x1

1

x2

2

x3

1

x4

1

x5

=

6

1

x1

1

x2

1

x3

2

x4

1

x5

=

7

1

x1

1

x2

1

x3

1

x4

2

x5

=

8

Gauss Elimination

4

Answer

Result

Result X Vector :

-1.0000

0.0000

1.0000

2.0000

3.0000

Resultant Matrix :

2.0000 1.0000 1.0000 1.0000 1.0000 -2.0000

0.0000 1.5000 0.5000 0.5000 0.5000 0.0000

0.0000 0.0000 1.3333 0.3333 0.3333 1.3333

0.0000 0.0000 0.0000 1.2500 0.2500 2.5000

0.0000 0.0000 0.0000 0.0000 1.2000 3.6000

Time taken = 1e-06

Okay

➤ Gauss Jordan, precision = 4

Matrix Solver
Alphabetical Coefficients Solver

5

2 x1 1 x2 1 x3 1 x4 1 x5 = 4
1 x1 2 x2 1 x3 1 x4 1 x5 = 5
1 x1 1 x2 2 x3 1 x4 1 x5 = 6
1 x1 1 x2 1 x3 2 x4 1 x5 = 7
1 x1 1 x2 1 x3 1 x4 2 x5 = 8

Gauss-Jordan Elimination

4

Answer

Result

Result X Vector :
-1.0000
0.0000
1.0000
2.0000
3.0000

Resultant Matrix :
1.0000 0.0000 0.0000 0.0000 0.0000 -1.0000
0.0000 1.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 1.0000 0.0000 0.0000 1.0000
0.0000 0.0000 0.0000 1.0000 0.0000 2.0000
0.0000 0.0000 0.0000 0.0000 1.0000 3.0000

Time taken = 1e-06

Okay

➤ LU decomposition

- Doolittle, precision = 4

Matrix Solver
Alphabetical Coefficients Solver

5

2 x1 1 x2 1 x3 1 x4 1 x5 = 4
1 x1 2 x2 1 x3 1 x4 1 x5 = 5
1 x1 1 x2 2 x3 1 x4 1 x5 = 6
1 x1 1 x2 1 x3 2 x4 1 x5 = 7
1 x1 1 x2 1 x3 1 x4 2 x5 = 8

LU Decomposition

Doolittle Decomposition

4

Answer

Result

Result X Vector :
-1.0000
0.0000
1.0000
2.0000
3.0000

Resultant Matrix :
2.0000 1.0000 1.0000 1.0000 1.0000
0.5000 1.5000 0.5000 0.5000 0.5000
0.5000 0.3333 1.3333 0.3333 0.3333
0.5000 0.3333 0.2500 1.2500 0.2500
0.5000 0.3333 0.2500 0.2000 1.2000

Time taken = 1e-06

Okay

- Crout, precision = 4

Matrix Solver

Alphabetical Coefficients Solver

5

2

x1

1

x2

1

x3

1

x4

1

x5

=

4

1

x1

2

x2

1

x3

1

x4

1

x5

=

5

1

x1

1

x2

2

x3

1

x4

1

x5

=

6

1

x1

1

x2

1

x3

2

x4

1

x5

=

7

1

x1

1

x2

1

x3

1

x4

2

x5

=

8

LU Decomposition

Crout Decomposition

4

Answer

Result

Result X Vector :

-1.0000

0.0000

1.0000

2.0000

3.0000

Resultant Matrix :

1.0000 0.5000 0.5000 0.5000 0.5000

1.0000 1.0000 0.3333 0.3333 0.3333

1.0000 0.5000 1.0000 0.2500 0.2500

1.0000 0.5000 0.3333 1.0000 0.2000

1.0000 0.5000 0.3333 0.2500 1.0000

Time taken = 0.0010001659393310547

Okay

- Cholesky, precision = 4

Matrix Solver

Alphabetical Coefficients Solver

5

2

x1

1

x2

1

x3

1

x4

1

1

x1

2

x2

1

x3

1

x4

1

1

x1

1

x2

2

x3

1

x4

1

1

x1

1

x2

1

x3

2

x4

1

1

x1

1

x2

1

x3

1

x4

2

LU Decomposition

Cholesky Decomposition

4

Answer

Result

Result X Vector :

-1.0000

0.0000

1.0000

2.0000

3.0000

L Matrix :

1.4142135623730951 0.0 0.0 0.0 0.0

0.7071067811865475 1.224744871391589 0.0 0.0 0.0

0.7071067811865475 0.40824829046386313 1.1547005383792517 0.0 0.0

0.7071067811865475 0.40824829046386313 0.28867513459481287 1.118033988749895 0.0

0.7071067811865475 0.40824829046386313 0.28867513459481287 0.22360679774997896 1.0954451150103321

Time taken = 0.0009915828704833984

Okay

❖ 2nd case

- Jacobi, absolute relative error = 0.00001

Matrix Solver

Alphabetical Coefficients Solver

3

8

x1

3

x2

2

x3 =

13

1

x1

5

x2

1

x3 =

7

2

x1

1

x2

6

x3 =

9

Jacobi

Initial Guess

0

0

0

Relative Error

0.00001

4

Answer

Result

Result X Vector :

1.0000

1.0000

1.0000

Number of iterations taken = 17

Time taken = 0.0020017623901367188

Okay

- Gauss Seidel, absolute relative error = 0.00001

Matrix Solver

Alphabetical Coefficients Solver

3

8

x1

3

x2

2

x3 =

13

1

x1

5

x2

1

x3 =

7

2

x1

1

x2

6

x3 =

9

Gauss-Seidel

Initial Guess

0

0

0

Relative Error

0.00001

4

Answer

Result

Result X Vector :

1.0000

1.0000

1.0000

Number of iterations taken = 6

Time taken = 1e-06

Okay

❖ 3rd case

➤ Gauss Seidel, number of iterations = 100

1

x1+

2

x2+

3

3

x1+

1

x2+

-2

4

x1+

3

x2+

1

1

x1+

-2

x2+

3

2

x1+

-1

x2+

4

3

x1+

2

x2+

2

Choose Operation

Gauss-Seidel

Initial Guess

0

0

0

Iterative

100

Significant Digits

Result

Result X Vector :

1184840482562523788135229919020517275733706466906582655399873052094105625785375103915083143953003773290079427306352279552.0000
-707239837074948307113368070752164559817497234139848670462032555487582820901703726395229688472434084056127607608572903424.0000
1512036471568046708188501362626922116396301173522824474395689042985279252195442760046066350289643577685831847588835360768.0000
-1561968015260569771124185517698156051749051755680277615288038372443514347314017359603481343787437262166726613671461519360.0000
-2781518696304317180160239779479584760193035612433531313026701091879631754549815223603087222793014701050519258332315779072.0000
-3028661677957908216186792823536979556188589129998835660721997444054216136380961835870108581818912408397039250522769457152.0000
-2311208727837958473937860398221396694251804573574937011655235477080633013464386829095719848915058143674343801309756391424.0000

Number of iterations taken = 100

Time taken = 0.004002809524536133

❖ 4th case

➤ Gauss elimination

Matrix Solver

Alphabetical Coefficients Solver

5

2

x1

3

x2

-1

x3

4

x4

-1

3

1

x1

2

x2

3

x3

-1

x4

4

3

3

x1

1

x2

-2

x3

3

x4

-4

3

4

x1

3

x2

1

x3

2

x4

0

3

1

x1

-2

x2

3

x3

1

x4

2

3

Gauss Elimination

4

Answer

Result

Error was found : Matrix is singular , no unique solution exists

Okay

❖ 5th case

➤ Gauss Elimination, precision = 6

Matrix Solver

Alphabetical Coefficients Solver

3

3

x1

-0.1

x2

-0.2

x3

=

7.8

0.1

x1

7

x2

-0.3

x3

=

-19

0.3

x1

-0.2

x2

10

x3

=

71.4

Gauss Elimination

6

Answer

Result

Result X Vector :

3.000000
-2.500000
7.000000

Resultant Matrix :

3.000000 -0.100000 -0.200000 9.000000
0.000000 7.003333 -0.293333 -17.508333
0.000000 0.000000 10.012042 70.084293

Time taken = 0.0010008811950683594

Okay

➤ Gauss Elimination, precision = 3

Matrix Solver

Alphabetical Coefficients Solver

3

3

x1

-0.1

x2

-0.2

x3

=

7.8

0.1

x1

7

x2

-0.3

x3

=

-19

0.3

x1

-0.2

x2

10

x3

=

71.4

Gauss Elimination

3

Answer

Result

Result X Vector :

3.000
-2.500
7.000

Resultant Matrix :

3.000 -0.100 -0.200 9.000
0.000 7.003 -0.293 -17.508
0.000 0.000 10.012 70.084

Time taken = 1e-06

Okay

❖ 6th case

➤ Gauss Jordan

Matrix Solver

Alphabetical Coefficients Solver

3

0

x1

2

x2

5

x3 =

1

2

x1

1

x2

1

x3 =

1

3

x1

1

x2

0

x3 =

2

Gauss-Jordan Elimination

4

Answer

Result

Result X Vector :

-2.0000

8.0000

-3.0000

Resultant Matrix :

1.0000 0.0000 0.0000 -2.0000

0.0000 1.0000 0.0000 8.0000

0.0000 0.0000 1.0000 -3.0000

Time taken = 1e-06

Okay

➤ LU Decomposition

▪ Doolittle

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

0

x1+

2

x2+

5

x3 =

1

2

x1+

1

x2+

1

x3 =

1

3

x1+

1

x2+

0

x3 =

2

Choose Operation

LU Decomposition

Choose LU Sub Operation

Doolittle Decomposition

Significant Digits

4

Answer

Result

Result X Vector :

-2.0000

8.0000

-3.0000

Resultant Matrix :

3.0000 1.0000 0.0000

0.0000 2.0000 5.0000

0.6667 0.1667 0.1667

Time taken = 0.0009999275207519531

❖ 7th case

- Jacobi, number of iterations = 50

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

2

x1+

1

x2+

6

x3 =

9

8

x1+

3

x2+

2

x3 =

13

1

x1+

5

x2+

1

x3 =

7

Choose Operation

Jacobi

Initial Guess

0

0

0

Iterative

50

Significant Digits

4

Answer

Result

Result X Vector :

-164891458012232960648511934693376.0000

-138512564348925216008148873117696.0000

-205855982268448925897566330028032.0000

Number of iterations taken = 50

Time taken = 0.0015132427215576172

Okay

- Gauss Seidel, number of iterations = 50

Matrix Solver

Alphabetical Coefficients Solver

Choose Size

3

2

x1+

1

x2+

6

x3 =

9

8

x1+

3

x2+

2

x3 =

13

1

x1+

5

x2+

1

x3 =

7

Choose Operation

Gauss-Seidel

Initial Guess

0

0

0

Iterative

50

Significant Digits

4

Answer

Result

Result X Vector :

-344291506491670363345078024933868933234389157668296657410731982608403437453312.0000

838718347124591243510148591560316676675906848872707567392084943612294429933568.0000

-3849300229131286162737769430593847354201866816198460863812667541749293090078720.0000

Number of iterations taken = 50

Time taken = 0.0010097026824951172

Okay

6. Bonus:

Coefficients can be letters and the output is expressed in term of the letters:

Suitable for system of equations of 2 or 3 variables. Any number of variables bigger than that is too much load for the computer.

Notes:

- 1- To run the app, open “RUN ME FOR PROJECT.exe”
- 2- You may need to install some python libraries

Run the following in a terminal:

```
pip install flask flask-cors numpy sympy
```