นายบสินทร ศิริจารุลงศ์

6004062660091 CSB

Assignment 4 Linear Algebra 1

1. Solve this linear system

$$
\begin{array}{rcrcrcr}
-2x_1 & + & 3x_2 & + & x_3 & & 9 \\
3x_1 & + & 4x_2 & - & 5x_3 & = & 0 \\
* \quad \boxed{x_1} & - & 2x_{\cancel{x}} & + & x_3 & & -4
\end{array}
$$

2

1.1 using Cramer’s Rule

1.2 Using Gauss Elimination Method

1.3 Using Gauss-Jordan Method

1.4 Using LU Decomposition Method

1.5 Write programs to solve this system using 4 methods mentioned above

Assignment 4

1.1

$$-2x_1+3x_2+x_3=9$$
$$3x_1+4x_2-5x_3=0$$
$$x_1-2x_2+x_3=-4$$

เขียนในรูป ความสัมพันธ์ จะได้

$$\begin{bmatrix} -2 & 3 & 1 \\ 3 & 4 & -5 \\ 1 & -2 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=\begin{bmatrix} 9 \\ 0 \\ -4 \end{bmatrix}$$

จะหา x_1 ได้จาก $x_1=\dfrac{\det[A]_1}{\det[A]}$

หา $\det[A]_1=\left|\begin{array}{ccc|cc} 9 & 3 & 1 & 9 & 3 \\ 0 & 4 & -5 & 0 & 4 \\ -4 & -2 & 1 & -4 & -2 \end{array}\right|$ $=(36+60+0)-(-16+90+0)$ $=96-74$ $=22$

หา $\det[A]=\left|\begin{array}{ccc|cc} -2 & 3 & 1 & -2 & 3 \\ 3 & 4 & -5 & 3 & 4 \\ 1 & -2 & 1 & 1 & -2 \end{array}\right|$ $=(-8-15-6)-(4-20+9)$ $=-29-(-7)=-29+7=-22$

$\therefore\ x_1=\dfrac{22}{-22}=-1$ #

ต่อไป หา $x_2=\dfrac{\det[A]_2}{\det[A]}$

หา $\det[A_2]=\left|\begin{array}{ccc|cc} -2 & 9 & 1 & -2 & 9 \\ 3 & 0 & -5 & 3 & 0 \\ 1 & -4 & 1 & 1 & -4 \end{array}\right|$ $=(0-45-12)-(0-40+27)$ $=-57-(-13)=-57+13$ $=-44$

$\therefore\ x_2=\dfrac{-44}{-22}=2$ #

ต่อไปหา $x_3 = \dfrac{\det[A]_3}{\det[A]}$

หา $\det[A]_3 = \begin{vmatrix} -2 & 3 & 9 \\ 3 & 4 & 0 \\ 1 & -2 & -4 \end{vmatrix} \begin{matrix} -2 & 3 \\ 3 & 4 \\ 1 & -2 \end{matrix}$ $= (32+0-54)-(36+0-36)$

$= -22-0 = -22$

$\therefore x_3 = \dfrac{-22}{-22} = 1$ #

ทำการตรวจสอบในสมการที่ 2 จะได้ $3(-1)+4(2)-5(1) = -3+8-5 = -8+8=0$ ซึ่งถูกต้อง

1.2 เขียนความสัมพันธ์จะอยู่ในรูป

$$\begin{bmatrix} -2 & 3 & 1 \\ 3 & 4 & -5 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9 \\ 0 \\ -4 \end{bmatrix}$$

step1: Forward Elimination

$$\left[\begin{array}{ccc|c} -2 & 3 & 1 & 9 \\ 3 & 4 & -5 & 0 \\ 1 & -2 & 1 & -4 \end{array}\right] \xrightarrow{R_1 \leftrightarrow R_3} \left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 3 & 4 & -5 & 0 \\ -2 & 3 & 1 & 9 \end{array}\right]$$

$$\begin{matrix} R_2 - 3R_1 \rightarrow \\ -R_3 \rightarrow \end{matrix} \left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 10 & -8 & 12 \\ 2 & -3 & -1 & -9 \end{array}\right] \xrightarrow{R_3 - 2R_1} \left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 10 & -8 & 12 \\ 0 & 1 & -3 & -1 \end{array}\right]$$

$10R_3 - R_2$ $\left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 10 & -8 & 12 \\ 0 & 0 & -22 & -22 \end{array}\right]$

หา x_3 จาก $-22x_3 = -22$

$x_3 = \frac{-22}{-22} = 1$ #

หา x_2 จาก $10x_2 - 8x_3 = 12$

$10x_2 - 8 = 12$

$10x_2 = 20$

$x_2 = 2$ #

หา x_1 จาก $x_1 - 2x_2 + x_3 = -4$

$x_1 - 4 + 1 = -4$

$x_1 - 3 = -4$

$x_1 = -4 + 3$

$x_1 = -1$ #

1.3 Gauss-Jordan Method

จากวิธีที่ 1.2 $10R_3 - R_2$ $\left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 10 & -8 & 12 \\ 0 & 0 & -22 & -22 \end{array}\right]$

$\frac{R_2}{10} \rightarrow$, $\frac{R_3}{-22} \rightarrow$ $\left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 1 & -\frac{8}{10} & \frac{12}{10} \\ 0 & 0 & 1 & 1 \end{array}\right]$ $R_2 + \frac{8}{10}R_3 \rightarrow$ $\left[\begin{array}{ccc|c} 1 & -2 & 1 & -4 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right]$

$R_1 - R_3$ $\left[\begin{array}{ccc|c} 1 & -2 & 0 & -5 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right]$ $R_1 + 2(R_2)$ $\left[\begin{array}{ccc|c} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right]$

$\therefore$ จะได้ $x_1 = -1,\ x_2 = 2,\ x_3 = 1$ #

1.4 LU Decomposition method

วิธีทำ $[L][U] = [A]$

$$\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix}\begin{bmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -2 & 3 & 1 \\ 3 & 4 & -5 \\ 1 & -2 & 1 \end{bmatrix}$$

จากข้อที่ 1.2 $10R_3 - R_2 \begin{bmatrix} 1 & -2 & 1 \\ 0 & 10 & -8 \\ 0 & 0 & -22 \end{bmatrix} \begin{matrix} \frac{R_2}{10} \rightarrow \\ \\ \frac{R_3}{-22} \rightarrow \end{matrix} \begin{bmatrix} 1 & -2 & 1 \\ 0 & 1 & -\frac{8}{10} \\ 0 & 0 & 1 \end{bmatrix}$ จะได้ U ออกมา

หา L จาก $[L] = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix}$ หา l_{21}, l_{31}, l_{32} จาก ค่า U ที่เราทำได้

$$\begin{matrix} \\ R_2 - 3R_1 \rightarrow \\ R_3 - 2R_1 \rightarrow \end{matrix}\begin{bmatrix} 1 & -2 & 1 \\ \boxed{0} & 10 & -8 \\ \boxed{0} & \underset{10R_3 - R_2}{\boxed{0}} & -22 \end{bmatrix}$$

จะได้ $\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix} = [L]$

จากความสัมพันธ์

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix}}_{[L]} \underbrace{\underbrace{\begin{bmatrix} 1 & -2 & 1 \\ 0 & 1 & -\frac{8}{10} \\ 0 & 0 & 1 \end{bmatrix}}_{[U]} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{[X]}}_{[Y]} = \underbrace{\begin{bmatrix} 9 \\ 0 \\ -4 \end{bmatrix}}_{[B]}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 9 \\ 0 \\ -4 \end{bmatrix} \rightarrow$$

$y_1 = 9$

$27 + y_2 = 0 \rightarrow y_2 = -27$

$18 - 27 + y_3 = -4$

$-9 + y_3 = -4$

$y_3 = 5$

หา x_1, x_2, x_3

$$\begin{bmatrix} 1 & -2 & 1 \\ 0 & 1 & -\frac{8}{10} \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9 \\ -27 \\ 5 \end{bmatrix}$$

จะได้ $x_3 = 5$

$x_2 - \frac{45}{10} = -27$

$x_2 = -27 + \frac{9}{2} = \frac{-54+9}{2} = \frac{-45}{2}$

$x_1 - 2x_2 + x_3 = 9$

$x_1 - 2\left(\frac{-45}{2}\right) + 5 = 9$

$x_1 - \frac{90}{2} = 4$

$x_1 - 45 = 4$

$x_1 = 49$

$\therefore$ จะได้ $x_1 = 49$, $x_2 = \frac{-45}{2}$, $x_3 = 5$ #

1.5 Cramer's Rule

```
#include <iostream>
using namespace std;
double determinantOfMatrix(double mat[3][3]) {
    double ans;
    ans = mat[0][0] * (mat[1][1] * mat[2][2] - mat[2][1] * mat[1][2])
        - mat[0][1] * (mat[1][0] * mat[2][2] - mat[1][2] * mat[2][0])
        + mat[0][2] * (mat[1][0] * mat[2][1] - mat[1][1] * mat[2][0]);
    return ans;
}
void findSolution(double coeff[3][4]) {
    double d[3][3] = {
        { coeff[0][0], coeff[0][1], coeff[0][2] },
        { coeff[1][0], coeff[1][1], coeff[1][2] },
        { coeff[2][0], coeff[2][1], coeff[2][2] },
```

```
        { coeff[1][0], coeff[1][1], coeff[1][2] },
        { coeff[2][0], coeff[2][1], coeff[2][2] },
};
double d1[3][3] = {
        { coeff[0][3], coeff[0][1], coeff[0][2] },
        { coeff[1][3], coeff[1][1], coeff[1][2] },
        { coeff[2][3], coeff[2][1], coeff[2][2] },
};
double d2[3][3] = {
        { coeff[0][0], coeff[0][3], coeff[0][2] },
        { coeff[1][0], coeff[1][3], coeff[1][2] },
        { coeff[2][0], coeff[2][3], coeff[2][2] },
};
double d3[3][3] = {
```

```
};
double d3[3][3] = {
        { coeff[0][0], coeff[0][1], coeff[0][3] },
        { coeff[1][0], coeff[1][1], coeff[1][3] },
        { coeff[2][0], coeff[2][1], coeff[2][3] },
};
double D = determinantOfMatrix(d);
double D1 = determinantOfMatrix(d1);
double D2 = determinantOfMatrix(d2);
double D3 = determinantOfMatrix(d3);
cout<<"Det[A] = "<<D<<endl;
cout<<"Det[A]1= "<<D1<<endl;
cout<<"Det[A]2= "<<D2<<endl;
cout<<"Det[A]3= "<<D3<<endl;
```

```
}
int main() {
    double coeff[3][4] = {
        { -2, 3, 1, 9 },
        { 3, 4, -5, 0 },
        { 1, -2, 1, -4 },
    };
    findSolution(coeff);
    return 0;
}
```

C:\Users\User\Desktop\love.exe

```
Det[A]= -22
Det[A]1= 22
Det[A]2= -44
Det[A]3= -22
x1 = -1
x2 = 2
x3 = 1
```

Gaussian Elimination method

m.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
#define N 3
int forwardElim(double mat[N][N+1]);
void backSub(double mat[N][N+1]);
void gaussianElimination(double mat[N][N+1]) {
    int singular_flag = forwardElim(mat);
    backSub(mat);
}
void swap_row(double mat[N][N+1], int i, int j) {
    for (int k=0; k<=N; k++) {
        double temp = mat[i][k];
        mat[i][k] = mat[j][k];
        mat[j][k] = temp;
```

```cpp
        mat[i][k] = mat[j][k];
        mat[j][k] = temp;
    }
}
int forwardElim(double mat[N][N+1]) {
    for (int k=0; k<N; k++) {
        int i_max = k;
        int v_max = mat[i_max][k];
        for (int i = k+1; i < N; i++)
            if (abs(mat[i][k]) > v_max)
                v_max = mat[i][k], i_max = i;
        if (!mat[k][i_max])
            return k;
        if (i_max != k)
```

```cpp
            return k;
        if (i_max != k)
            swap_row(mat, k, i_max);
        for (int i=k+1; i<N; i++)
        {
            double f = mat[i][k]/mat[k][k];
            for (int j=k+1; j<=N; j++)
                mat[i][j] -= mat[k][j]*f;
            mat[i][k] = 0;
        }
    }
    return -1;
}
void backSub(double mat[N][N+1]) {
    double x[N];
```

```cpp
}
void backSub(double mat[N][N+1]) {
    double x[N];
    for (int i = N-1; i >= 0; i--) {
        x[i] = mat[i][N];
        for (int j =i+1; j<N; j++)     {
            x[i] -= mat[i][j]*x[j];
        }
        x[i] = x[i]/mat[i][i];
    }
    for (int i=0; i<N; i++){
        cout<<x[i]<<endl;
    }
}
```

```
int main() {
    double mat[N][N+1] = {{-2, 3, 1, 9},
                          {3, 4, -5, 0},
                          {1, -2, 1, -4}
                          };
    gaussianElimination(mat);
    return 0;
}
```

```
-1
2
1
----------------------------------------
```

Gauss-Jordan Elimination method

```
#include <bits/stdc++.h>
using namespace std;
#define M 10
void PrintMatrix(float a[][M], int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j <= n; j++)
        cout << a[i][j] << " ";
        cout << endl;
    }
}
int PerformOperation(float a[][M], int n) {
    int i, j, k = 0, c, flag = 0, m = 0;
    float pro = 0;
    for (i = 0; i < n; i++) {
        if (a[i][i] == 0) {
            c = 1;
            while (a[i + c][i] == 0 && (i + c) < n)
                c++;
            if ((i + c) == n) {
                flag = 1;
                break;
            }
            for (j = i, k = 0; k <= n; k++)
                swap(a[j][k], a[j+c][k]);
        }
        for (j = 0; j < n; j++) {
            if (i != j) {
                float pro = a[j][i] / a[i][i];
                for (k = 0; k <= n; k++)
                    a[j][k] = a[j][k] - (a[i][k]) * pro;
            }
        }
    }
    return flag;
}
void PrintResult(float a[][M], int n, int flag) {
    cout << "Result is : ";
    for (int i = 0; i < n; i++)
        cout << a[i][n] / a[i][i] << " ";
}
int CheckConsistency(float a[][M], int n, int flag) {
    int i, j;
```

```
    float sum;
    flag = 3;
    for (i = 0; i < n; i++)
    {
        sum = 0;
        for (j = 0; j < n; j++)
            sum = sum + a[i][j];
        if (sum == a[i][j])
            flag = 2;
    }
    return flag;
}
int main() {
    float a[M][M] = {{ -2, 3, 1, 9 },
                     { 3, 4, -5, 0 },
                     { 1,-2, 1, -4 }};
    int n = 3, flag = 0;
    flag = PerformOperation(a, n);
    if (flag == 1)
        flag = CheckConsistency(a, n, flag);
    cout << "Final Augumented Matrix is : " << endl;
    PrintMatrix(a, n);
    cout << endl;
    PrintResult(a, n, flag);
    return 0;
}
```

```
Final Augumented Matrix is :
-2 0 0 2
0 8.5 -2.38419e-007 17
0 0 1.29412 1.29412

Result is : -1 2 1
--------------------------------
```

LU-Decomposition method

```
#include <bits/stdc++.h>
using namespace std;
const int MAX = 100;
void luDecomposition(int mat[][MAX], int n) {
    int lower[n][n], upper[n][n];
    memset(lower, 0, sizeof(lower));
    memset(upper, 0, sizeof(upper));
    for (int i = 0; i < n; i++) {
        for (int k = i; k < n; k++) {
            int sum = 0;
            for (int j = 0; j < i; j++)
                sum += (lower[i][j] * upper[j][k]);
            upper[i][k] = mat[i][k] - sum;
        }
        for (int k = i; k < n; k++) {
            if (i == k)
                lower[i][i] = 1;
            else {
                int sum = 0;
                for (int j = 0; j < i; j++)
                    sum += (lower[k][j] * upper[j][i]);
                lower[k][i] = (mat[k][i] - sum) / upper[i][i];
            }
        }
    }
    cout << setw(6) << "      Lower Triangular"
         << setw(32) << "Upper Triangular" << endl;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++)
            cout << setw(6) << lower[i][j] << "\t";
        cout << "\t";
        for (int j = 0; j < n; j++)
            cout << setw(6) << upper[i][j] << "\t";
        cout << endl;
    }
}
```

```
int main()
{
    int mat[][MAX] = { { -2, 3, 1 },
                       { 3, 4, -5 },
                       { 1, -2, 1 } };
    luDecomposition(mat, 3);
    return 0;
}
```

```
     Lower Triangular                Upper Triangular
  1       0       0              -2       3       1
 -1       1       0               0       7      -4
  0       0       1               0       0       1

--------------------------------
Process exited after 0.2062 seconds with return value 0
Press any key to continue
```