|  |  |
| --- | --- |
|  | 中国矿业大学 |
| China University of Mining and Technology |

《空间数据误差处理》实验报告

**姓 名： 武成龙**

**班 级： 地信2019-1班**

**学 号： 07192329**

**中国矿业大学环境与测绘学院**

**2022-01-21**

# 实验一：矩阵求逆（高斯—若尔当法）

## 新建项目

本实验使用C#作为编程语言。首先在Visual Studio中新建一个项目，选择“Windows窗体应用（.NET Framework）”，将其命名为“07192329武成龙01”。程序所需数据文件一并放在“C#”文件夹中。

## 添加控件，构建窗体

（1）添加相关控件（label，pictureBox，Button，richTextBox），修改属性。

（2）要求输入的矩阵保存在txt文件中，而且第一行为阶数，第二行开始为矩阵。输入和输出的矩阵格式相同。

（3）两个富文本框可以显示输入的矩阵和求解的逆矩阵。如图1：

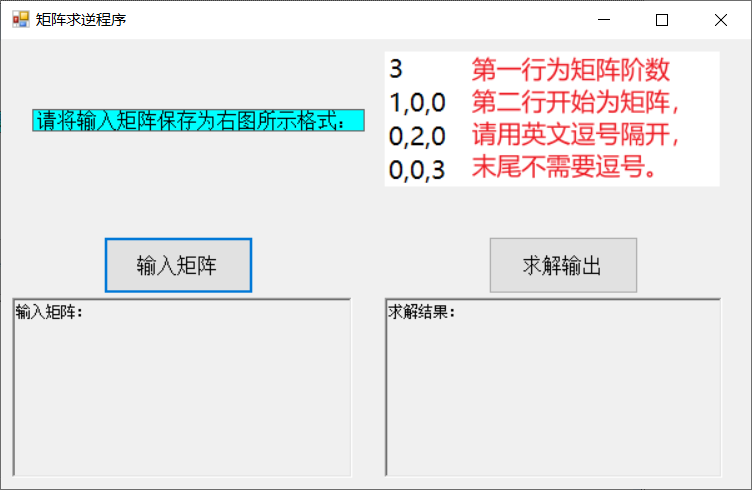


图 1 窗体构建结果

## 添加代码，实现功能

（1）在Form1.cs开始处加入引用“using System.IO”。

（2）双击“输入矩阵”，为该控件添加代码如下：

/\*局部变量声明：

\*FileInput：打开的文件

\*FileInputName：读取文件流

\*StrRow：文件的某一行

\*m：临时存放矩阵的二维数组

\*MatrixRow：矩阵的某一行

\*/

//打开新TXT文件

OpenFileDialog FileInput = new OpenFileDialog();

FileInput.Filter = "文本文件|\*.txt";

if (FileInput.ShowDialog() == DialogResult.OK)

{

//放入读取文件流

string FileInputName = FileInput.FileName;

StreamReader FileInputReader = File.OpenText(FileInputName);

//第一行读取为阶数

StrRow = FileInputReader.ReadLine();

Rank\_static = int.Parse(StrRow);

//循环读取为矩阵

double[,] m = new double[Rank\_static, Rank\_static];

for (int i = 0; i < Rank\_static; i++)

{

StrRow = FileInputReader.ReadLine();

string[] MatrixRow = StrRow.Split(',');

for (int j = 0; j < Rank\_static; j++)

{

m[i, j] = double.Parse(MatrixRow[j]);

}

}

//进行一次深复制

M = (double[,])m.Clone();

//显示成功

MessageBox.Show("读取成功!");

//显示到文本框

richTextBox\_input.LoadFile(FileInputName, RichTextBoxStreamType.PlainText);

}

（3）同理为“求逆运算”添加代码，关键代码如下：

矩阵求逆：

for (rank = 0; rank < Rank; rank++)

{

pivot = 0; I0 = 0; J0 = 0;

for (i = rank; i < Rank; i++)

{

for (j = rank; j < Rank; j++)

{

if (Math.Abs(M\_out[i, j]) > Math.Abs(pivot))

{

pivot = M\_out[i, j]; I0 = i; J0 = j;

}

}

}

if (Math.Abs(pivot) < ep)

{

MessageBox.Show("主元的绝对值过小，矩阵奇异！");

}

if (I0 != rank)

{

for (j = 0; j < Rank; j++)

{

temp = M\_out[I0, j]; M\_out[I0, j] = M\_out[rank, j]; M\_out[rank, j] = temp;

}

}

if (J0 != rank)

{

for (i = 0; i < Rank; i++)

{

temp = M\_out[i, J0]; M\_out[i, J0] = M\_out[i, rank]; M\_out[i, rank] = temp;

}

}

p[rank] = I0; q[rank] = J0;//记录最大主元位置

for (j = 0; j < Rank; j++)

{

if (j == rank)

{

b[j] = 1 / pivot; c[j] = 1;

}

else

{

b[j] = -M\_out[rank, j] / pivot; c[j] = M\_out[j, rank];

}

M\_out[rank, j] = 0; M\_out[j, rank] = 0;

}

for (i = 0; i < Rank; i++)

{

for (j = 0; j < Rank; j++)

{

M\_out[i, j] = M\_out[i, j] + c[i] \* b[j];

}

}

}

//消元时完全选主元进行了换行与换列，现在反过来，返回原始行列状态

for (rank = Rank - 1; rank >= 0; rank--)

{

I0 = p[rank]; J0 = q[rank];

if (I0 != rank)

{

for (i = 0; i < Rank; i++)

{

temp = M\_out[i, I0]; M\_out[i, I0] = M\_out[i, rank]; M\_out[i, rank] = temp;

}

}

if (J0 != rank)

{

for (j = 0; j < Rank; j++)

{

temp = M\_out[J0, j]; M\_out[J0, j] = M\_out[rank, j]; M\_out[rank, j] = temp;

}

}

}

保存文件：

private void Save(double[,]M\_save)

{

//打开对话框

SaveFileDialog FileOutput = new SaveFileDialog();

FileOutput.Filter = "txt文件(\*.txt)|\*.txt";

if (FileOutput.ShowDialog() == DialogResult.OK)

{

//写入文件流

StreamWriter TextWriter = File.CreateText(FileOutput.FileName);

//第一行写入阶数

TextWriter.Write(Rank\_static);

TextWriter.WriteLine();

//后面写入矩阵

for (int i = 0; i < Rank\_static; i++)

{

for (int j = 0; j < Rank\_static; j++)

{

TextWriter.Write(Convert.ToString(M\_save[i, j]));

if (j == Rank\_static-1)

continue;

TextWriter.Write(",");

}

TextWriter.WriteLine();

}

TextWriter.Flush();

TextWriter.Close();

//显示成功

MessageBox.Show("文件 "+ FileOutput.FileName+ " 保存成功!");

//显示到文本框

richTextBox\_output.LoadFile(FileOutput.FileName, RichTextBoxStreamType.PlainText);

}

}

## 调试并运行程序

运行界面如图2、3、4所示。

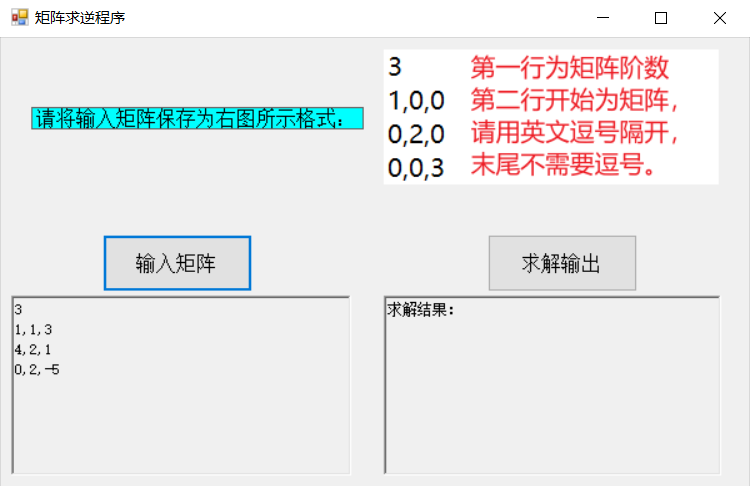


图 2 输入矩阵

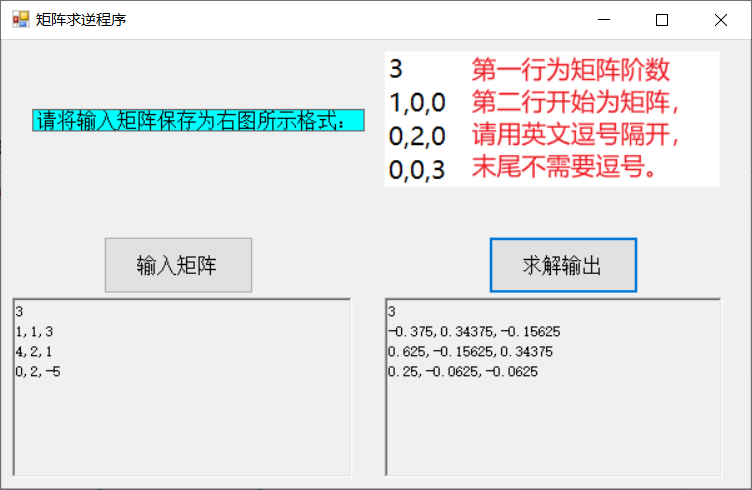


图 3 求解输出

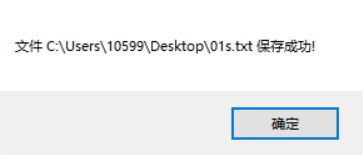


图 4 保存成功

## 小结

为了方便多次计算，要求输入的矩阵保存在txt文件中，而且第一行为阶数，第二行开始为矩阵，并且保证输出和输入的矩阵格式相同。

使用C#编写窗体应用的好处是可以直接调用一些类和函数，比如保存文件使用的类SaveFileDialog和StreamWriter，以及WriteLine()函数可以方便换行。

不过还存在一些问题，如输出结果的小数点位数不确定。

# 实验二：高程控制网间接平差程序设计

## 新建项目

本实验使用C#作为编程语言。首先在Visual Studio中新建一个项目，选择“Windows窗体应用”，将其命名为“07192329武成龙02”。程序所需数据文件一并放在“C#”文件夹中。

## 添加控件，构建窗体

（1）添加相关控件（label，groupBox，Button，richTextBox），按照功能和用途分别为其修改属性值。最终设计效果如图5所示。



图 5 主窗体

（2）要求输入的数据要求单独有一个窗体显示，点击“数据要求”即可查看。如图6。

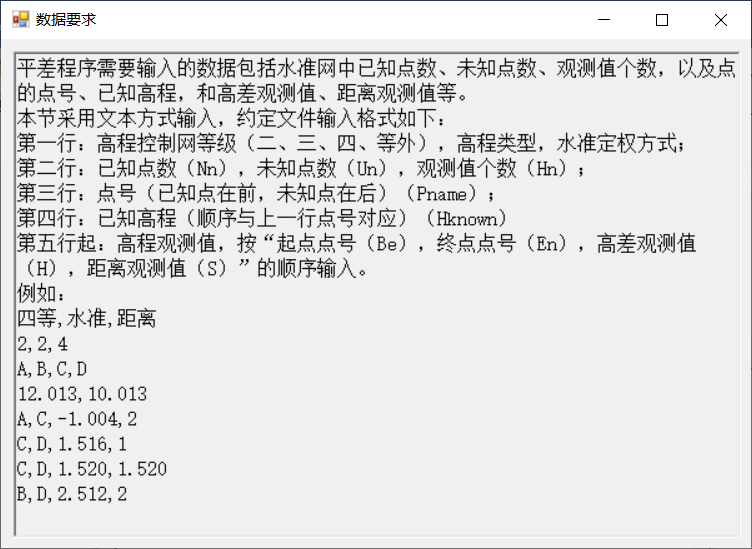


图 6 数据要求

## 添加代码，实现功能

（1）高程近似值计算

主要代码如下：

private void H\_results()

{

int i, j;

for (i = 0; i < Hn; i++) //对每个观测值列误差方程

{

for (j = 0; j < Hn; j++)

{

if ((Be[j] == Nn + i) && (En[j] < Nn + i))

{

Hknown[Nn + i] = Hknown[En[j]] - H[j];

break; //找到一个起点相同且终点已知的观测值

}

if ((En[j] == Nn + i) && (Be[j] < Nn + i))

{

Hknown[Nn + i] = Hknown[Be[j]] + H[j];

break; //找到一个终点相同且起点已知的观测值

}

}

}

}

（2）误差方程列立

主要代码如下：

private void ErrorFormula()

{

A = new double[Hn, Un];

L = new double[Hn, 1];

P = new double[Hn, Hn]; //重命名

int i;

for (i = 0; i < Hn; i++) //对每个观测值列误差方程

{

if (En[i] >= Nn) A[i, En[i] - Nn] = 1;

//若终点未知，则给终点对应的系数矩阵元素赋值

if (Be[i] >= Nn) A[i, Be[i] - Nn] = -1;

//若起点未知，则给起点对应的系数矩阵元素赋值

L[i, 0] = -(Hknown[En[i]] - Hknown[Be[i]] - H[i]);

//根据起终点计算常数项

P[i, i] = 1 / S[i];

//以距离的倒数为权

}

}

（3）通用间接平差函数

主要代码如下：

public void InAdjust(double[,] a, double[,] p, double[,] l, double[] x, double[,] q)

{

int a1, a2, p1, p2, L1, x1; //定义输入矩阵或向量的大小变量

int i, j;

double[,] At, AtP, Naa;

double[,] w; //几个计算过程中的中间过渡矩阵

a1 = a.GetUpperBound(0) - a.GetLowerBound(0) + 1;//矩阵的行数

a2 = a.GetUpperBound(1) - a.GetLowerBound(1) + 1;//矩阵的列数

L1 = l.GetUpperBound(0) - l.GetLowerBound(0) + 1;

x1 = x.GetUpperBound(0) - x.GetLowerBound(0) + 1;

p1 = p.GetUpperBound(0) - p.GetLowerBound(0) + 1;

p2 = p.GetUpperBound(1) - p.GetLowerBound(1) + 1;

//根据输入矩阵的行列数定义中间矩阵的大小

At = new double[a2, a1];

AtP = new double[a2, a1];

Naa = new double[a2, a2];

w = new double[a2, 1];

x2 = new double[Un, 1];

//l = new double[Hn,1];

q = new double[a2, a2];

//组成法方程并计算

MatrixTranspose(a, At); //求A的转置矩阵

MatrixMultiply(At, p, AtP); //求AtP

MatrixMultiply(AtP, a, Naa); //法方程系数矩阵，该通用程序的法方程是由误差方程组成而得

for (i = 0; i < a2; i++)

{

for (j = 0; j < a2; j++)

q[i, j] = Naa[i, j];

}

Invert(Naa, Q); //对法方程系数阵应用高斯-约当法求逆，得协因数阵(精度评定)

MatrixMultiply(AtP, l, w); //法方程常数向量

}

（4）精度评定

主要代码如下：

int i, j;//循环变量

double VtPV = 0;

double[] AX = new double[L.GetLength(0)];

double[] V = new double[L.GetLength(0)];

double[] VtP = new double[Hn];

for (i = 0; i < A.GetLength(0); i++)

for (j = 0; j < A.GetLength(1); j++)

AX[i] += A[i, j] \* X[j];

for (i = 0; i < Hn; i++)

V[i] = AX[i] - L[i];

for (i = 0; i < Hn; i++)

for (j = 0; j < Hn; j++)

{

VtP[i] += V[j] \* P[j, i];

VtP[i] = double.Parse(VtP[i].ToString("#0.00000"));

}

for (i = 0; i < Hn; i++)

VtPV += VtP[i] \* V[i];

MiddleError = Math.Sqrt(VtPV / (Hn - Un));

PointErr = new double[Un];

for (i = 0; i < Un; i++)

{

PointErr[i] = MiddleError \* Math.Sqrt(Q[i, i]);

PointErr[i] = double.Parse(PointErr[i].ToString("#0.00000"));

}

（5）精度评定

主要代码如下：

public void Accuracy()

{

double VtPV = 0;

double[,] AX, V, VtP, Vt; //VtPV乘积，及几个过渡一维数组

AX = new double[Hn, 1];

V = new double[Hn, 1];

VtP = new double[1, Hn];

Vt = new double[1, Hn]; //转置

PointErr = new double[Un];

////计算单位权中误差////

int i;

MatrixMultiply(A, x2, AX);//调用矩阵相乘函数

for (i = 0; i < Hn; i++)

V[i, 0] = AX[i, 0] - L[i, 0];

MatrixTranspose(V, Vt);

MatrixMultiply(Vt, P, VtP);

for (i = 0; i < Hn; i++)

VtPV = VtPV + VtP[0, i] \* V[i, 0];

MiddleError = Math.Sqrt(VtPV / (Hn - Un));//(Hn - Un)=r,计算中误差

for (i = 0; i < Un; i++) //计算点位误差

PointErr[i] = MiddleError \* Math.Sqrt(Q[i, i]);

}

## 调试并运行程序

（1）单击“导入数据”，弹出对话框，选择准备好的数据文件，数据即可显示在第一个文本框中，同时也显示路径。效果如图7：

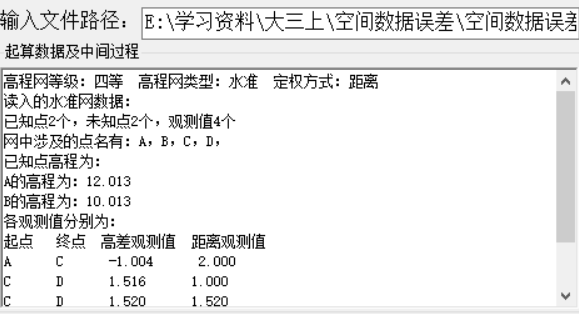


图 7 导入数据

（2）单击“平差结果”，第二个文本框就显示计算结果，如图8：



图 8 平差结果

（3）单击“保存结果”，弹出对话框，选择保存位置，显示“保存成功”即保存完成。

## 小结

要注意读取文件的时候应添加参数Encoding.UTF8，这样读取中文不会出现错误。还专门添加了第二个窗口，显示数据文件格式。

比较合理的流程应该是激活全部按钮的函数，然后编辑函数体，结果会比较规整。

另外，使用的数组较多，应该给合适的命名和注释，方便阅读和编写。使用C#写简介平差程序也有不少不方便之处，比如矩阵相乘、矩阵转置都需要自己先写好函数再调用。

总的来说更加熟悉了Windows窗体应用，以及更好地把握了测量平差的原理。