**Алгоритъм на Гаус – Жордан**

30 Ноември 2008 г. 14:46

1.      **Същност на метода**.

Чрез този алгоритъм се решават линейни системи с n – уравнения и n - неизвестни

Същността на метода ще разкрием, като го приложим върху решаването на следната конкретна система :

x1 – 2x2 + 3x3 + x4 = 10  
 2x2 + 8x3 -4x4 = 12  
 -2x1 + 7x2 + 8x3 -3x4 = 24  
 -3x1 + 9x2 + 4x3 -7x4 = 10

Общата идея на алгоритъма е следната:

Системата се подлага на еквивалентни преобразувания с цел коефициентите по главния диагонал да станат единица, а всички останали коефициенти ( без свободните членове ) да бъдат нули.

Горното правило ще приложим върху нашия пример. С цел по – малко писане, ще използваме представянето на системите чрез таблици, в които няма да записваме неизвестните, плюсовете и символа за равенство. Така горната система ще запишем във вида :

1– 2 3 1 10

0 2 8 -4 12

-2 7 8 -3 24  
 -3 9 4 -7 10

В рамките на първата итерация ще анулираме коефициентите пред x1. За целта ще умножим първото уравнение по 2 и ще го съберем с третото. След това ще умножим първото по 3 и ще го съберем с четвъртото. Така ще получим следната таблица ( еквивалентна система на горната ) :

1– 2 3 1 10

0 2 8 -4 12

0 3 14 -1 44  
 0 3 13 -4 29

Сега предстои да повторим нещата за коефициентите пред x2 . Във второто уравнение коефициентът е 2. За да стане 1 е необходимо да разделим всичките коефициенти на второто уравнение на 2 и резултатът отново да запишем като втори ред. Коефициентите от втория ред ( 0 1 4 -2 6 ) ще умножим по 2 и ще ги прибавим към първия ред – така получаваме новите стойности от първия ред. След това втория ред ще умножим по 3 и съответните стойности ще прибавим към третото и четвъртото уравнение. В резултат се получава следната система, еквивалентна на горната :

1 0 11 -3 22

0 1 4 -2 6

0 0 2 5 26

0 0 1 2 11

На третата итерация следва да имаме 1 в третото уравнение, а останалите коефициенти пред x3 да бъдат нули. Отново се налага да разделим коефициентите в третото уравнение на 2, Сега те са ( 0 0 1 2,5 13). След еквивалентните преобразувания системата добива следния вид :

1 0 0 -30,5 -121

0 1 0 -12 -46

0 0 1 2,5 13

0 0 0 -0,5 -2,0

При четвъртата итерация се осигурява 1 в четвъртото уравнение пред x4 , а в останалите уравнения – нула. Така новия вид на еквивалента система е:

1 0 0 0 1

0 1 0 0 2

0 0 1 0 3

0 0 0 1 4

Последната система всъщност директно посочва решението на началната система ( 1; 2; 3; 4).

2.      **Задача.** Да се състави конзолен проект с име Gaus, който решава линейна система от n уравнения с n неизвестно метода на Гаус – Жордан, като коефициентите са реални числа и се въвеждат от текстов файл. За визуализиране на матрицата да се състави самостоятелен модул ViewMatrix.

Решението ще реализираме на няколко стъпки :

а/ необходими структури от данни

-          глобален двумерен масив от реални числа double ar[20][20];;

-          глобална променлива n, съдържаща броя на неизвестните;б/ създаване на текстов файл с име table.txt.

-          за по- лесна работа, същият се поставя в папката на проекта;

-          структурата му е следната :

        n  
        a11 a12 a13 a14 a15  
a21 a22 a23 a24 a25  
a31 a32 a33 a34 a35  
a41 a42 a43 a44 a45

в/ визуализиране на матрица от реални числа

void ViewMatrix()

{cout<<setprecision(2)<<endl;  
 for(int i=1;i<=n;i++)  
 {for(int j=1;j<=n+1;j++) cout<<setw(8)<<arIdea[j];

cout<<endl;

}}

г/ алгоритъм за зареждане на таблица от текстов файл

ifstream fin;  
fin.open(“table.txt”);  
fin>>n;  
for(int i=1;i<=n;i++)  
 for(int j=1;j<=n+1;j++) fin>>arIdea[j];  
fin.close();

д/ еквивалентни преобразувания

for(int k=1;k<=n;k++){// осигуряване единица в диагоналния елемент – ar[k][k]  
 double b=ar[k][k];  
 for(int j=1;j<=n+1;j++) ar[k][j]=ar[k][j]/b;  
// анулиране на останалите елементи в k-та колона  
 for(int i=1;i<=n;i++)  
 if(i!=k){b=-arIdea[k];  
 for(j=k;j<=n+;j++)  
 arIdea[j]=arIdea[j]+b\*ar[k][j];  
 }

ViewMatrix(); //извеждане на таблицата

}

Разработеното до момента преобразува началната матрица четири пъти. Това обаче не решава целия проблем.

**3.Усъвършенстване на програмата.** Съществен недостатък на разработената програма е, че не е направено нищо по отношение предпазване от деление на нула. Желателно е делението на нула да се отдалечи колкото се може повече.Ако се разместят местата на две уравнения, то новополучената система е еквивалентна на дадената. Ще използваме този факт за подобряване на алгоритъма, а именно - в k –та таблица като k-то уравнение ще поставяме онова, което има най – голям по абсолютна стойност коефициент в k-та колона, измежду уравненията с номер по – голям от k.

**4. Задача.** Да се състави модул Update, който реализира посоченото по – горе разместване на редове / уравнения.

void Update(int k)  
{int l=k;  
 for(int i=(k+1);i<=n;i++)  
 if(fabs(ar[l][k])<fabs(arIdea[k])) l=I;  
// разместване if(l>k){ for(int j=1;j<=n+1;j++)  
 {double tmp = ar[k][j];  
 ar[k][j]=ar[l][j];  
 ar[l][j]=tmp;  
 } ViewMatrix();  
 }}

В главната програма в началото на цикъла по k, трябва да се извика подпрограмата Update. Ето я и цялата програма.

#include "stdafx.h"

#include <conio.h>

#include <fstream>

#include <iomanip>

#include <math.h>

using namespace std;

double ar[20][20],b; int n;

void ViewMatrix()

{cout<<setprecision(2)<<endl;

for(int i=1;i<=n;i++)

{ for(int j=1;j<=n+1;j++)

cout<<setw(8)<<arIdea[j];

cout<<endl;

}

}

void Update(int k)

{int l=k;

for(int i=(k+1);i<=n;i++)

if(fabs(ar[l][k])<fabs(arIdea[k])) l=i;

if(l>k){ for(int j=1;j<=n+1;j++)

{double tmp=ar[k][j];

ar[k][j]=ar[l][j];

ar[l][j]=tmp;

}

ViewMatrix();

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{ cout<<fixed;

ifstream fin;

fin.open("table.txt");

fin>>n;

for(int i=1;i<=n;i++)

for(int j=1;j<=n+1;j++) fin>>arIdea[j];

ViewMatrix();

for(int k=1;k<=n;k++)

{Update(k);

//осигуряване 1 в главния диагонал

b=ar[k][k];

for(int j=1;j<=n+1;j++)ar[k][j]=ar[k][j]/b;

//анулиране на останалите коефициенти

for(int i=1;i<=n;i++)

if(i!=k){b=-arIdea[k];

for(int j=k;j<=n+1;j++)

arIdea[j]=arIdea[j]+b\*ar[k][j];

}

ViewMatrix();

}

getch();

return 0;

}

Следващите подобрения могат да се насочат към въвеждане от конзолата името на файла, който ще съдържа началната матрица.

Друго усъвършенстване е да се правят необходимите анализи и извеждат подходящи съобщения, когато системата няма решение или има безбройно много. Последното е наложително, когато матрицата не е квадратна ( с размерност m x n ) и е възможно да има линейно – зависими уравнения или пък решението да зависи от няколко параметъра.