**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

дисциплина: Вычислительная математика

Лабораторная работа № 9

тема: «Решение систем линейных уравнений методом итераций»

|  |  |
| --- | --- |
|  | Выполнил: ст. группы ПВ-21  Зановская А. И.  Проверил: Бондаренко Т.В. |

Белгород

2017

**Цель работы:** изучить метод итераций для решения систем линейных уравнений и получить практические навыки его применения.

**Вариант 10**

**Задания к работе**

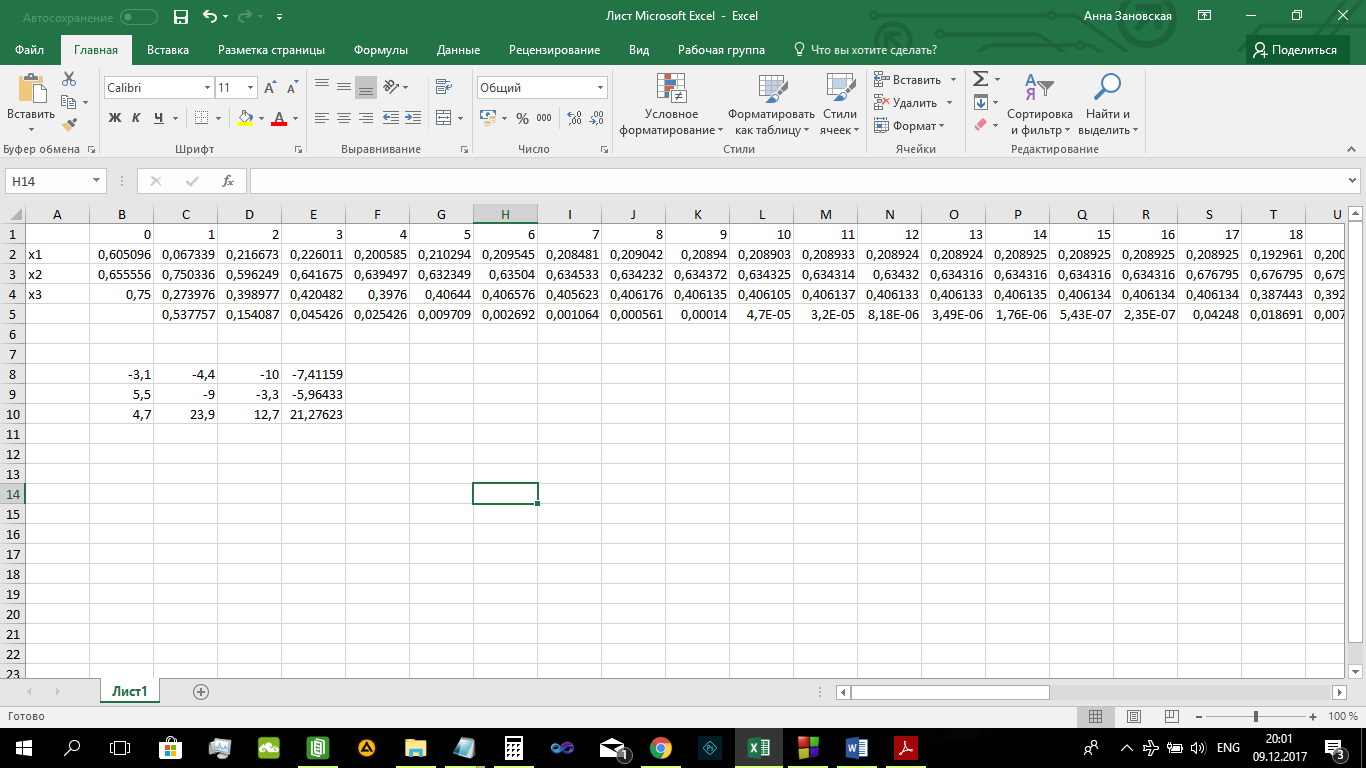
1. Преобразовать систему линейных уравнений заданного варианта к виду, требуемому для метода итераций (получить матрицу В коэффициентов при неизвестных, для которой выполняется условие диагонального преобладания).

2. Вычислить нормы первую ||В||1 и бесконечную ||В||∞ матрицы В и выбрать из них меньшую.

3. Вычислить количество шагов (итераций) метода, используя априорную оценку.

4. Выбрать начальное приближение метода итераций.

5. Методом итераций получить приближенное решение системы уравнений с заданной точностью ε=0,1.



6. Подсчитать число фактически выполненных итераций.

K = 3.

7. Сравнить количество итераций, полученных применением априорной и

апостериорной оценок.

8. Вычислить вектор невязки для полученного приближенного решения.

1) -7,41

2) -5,96

3) 21,28

|  |
| --- |
|  |
|  |
|  |

9. Описать в модуле функцию, выполняющую решение заданной системы линейных уравнений с заданной точностью методом итераций за конечное число шагов.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <math.h>

float\*\* InitMemMatr(int N);

float\* InitMemArr(int N);

void Add (float \*\*A, float \*\*B, float \*C, int n);

float Sum1 (float \*\*A, int i, int n);

float SumB (float \*\*A, int j, int n);

int KIter (float \*\*B, float \*C, int n, float e, float \*e1);

int Fl (float \*x0, float \*x1, float e1, int n);

float\* Desicion (float \*\*B, float \*C, int n, float e, float e1, int k);

void main ()

{

float \*\*A, \*\*B, \*X0, \*C, \*D,e=0.000001,e1;

int n=3,k,i,j;

A = InitMemMatr(n);

B = InitMemMatr(n);

C = InitMemArr (n);

D = InitMemArr (n);

X0= InitMemArr (n);

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

D[i]=C[i];

Read (A, C);

Output(A,C,n);

Add(A,B,C,n);

Output(B,C,n);

k=KIter(B,C,n,e,&e1); printf ("\nk = %d\n",k);

X0=Desicion (B, C, n, e, e1, k);

}

void Read (float \*\*A, float \*B)

{

A[0][0] = 15.7; A[0][1] = 5.9; A[0][2] = 6.1;

A[1][0] = 5.5; A[1][1] = -9; A[1][2] = -3.3;

A[2][0] = -3.1; A[2][1] = -4.4; A[2][2] = -10;

B[0] = 9.5; B[1] = -5.9; B[2] = -7.5;

}

void Output (float \*\*A, float \*B, int N)

{

int i,j;

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

{

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

{

if (A[i][j]>=0&&j)

printf ("+");

printf ("%.2f",A[i][j]);

if (j==N-1)

printf ("=%.2f", B[i]);

}

printf ("\n");

}

printf ("\n");

}

void Add (float \*\*A, float \*\*B, float \*C, int n)

{

int i,j;

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

{

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

{

if (i!=j)

B[i][j]=-(float)(A[i][j]/A[i][i]);

else

B[i][j]=0;

}

C[i]=(float)(C[i]/A[i][i]);

}

}

float NormMatr1 (float \*\*A, int N)

{

int i,j;

float max,s;

max=Sum1(A,0,N);

for (i=1;i<N;i++)

{

s=Sum1(A,i,N);

if (s>max)

max=s;

}

return max;

}

float Sum1 (float \*\*A, int i, int n)

{

int j;

float sum=0;

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

sum+=A[i][j];

return sum;

}

float NormMatrB (float \*\*A, int N)

{

int i,j;

float max,s;

max=SumB(A,0,N);

for (i=1;i<N;i++)

{

s=Sum1(A,i,N);

if (s>max)

max=s;

}

return max;

}

float SumB (float \*\*A, int j, int n)

{

int i;

float sum=0;

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

sum+=A[i][j];

return sum;

}

int KIter (float \*\*B, float \*C, int n, float e, float \*e1)

{

float N1, Nb, Nx=0, Norm;

int i;

N1=NormMatr1(B,n);

Nb=NormMatrB(B,n);

if (N1>Nb)

{

Norm=Nb;

Nx=fabs(C[0]);

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

if (fabs(C[i])>Nx)

Nx=fabs(C[i]);

}

else

{

Norm=N1;

Nx=0;

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

Nx+=fabs(C[i]);

Nx=pow(Nx,(float)(1./n));

}

i=(int)((log(e)+log(1-Norm)-log(Nx))/log(Norm));

i+=2;

\*e1=(float)(e\*(1-Norm)/Norm);

return i;

}

float\* Desicion (float \*\*B, float \*C, int n, float e, float e1, int k)

{

//B - матрица коэффициентов

int i, j, t=0, flag=0;

float \*X0, \*X1;

X0=InitMemArr(n);

X1=InitMemArr(n);

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

X0[i]=C[i];

while (t<k&&!flag)

{

t++;

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

{

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

X1[i]+=B[i][j]\*X0[j];

X1[i]+=C[i];

}

flag=Fl(X0,X1,e1,n);

CopyArr(X0,X1,n);

}

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

printf ("%.6f ", X0[i]);

return X0;

}

//возвращает 1, если выполнено условие остановки

int Fl (float \*x0, float \*x1, float e1, int n)

{

int i;

float max=0,t;

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

{

t=fabs(x0[i]-x1[i]);

if (t>max)

max=t;

}

return (max<e1);

}

//x0:=x1

void CopyArr (float \*X0, float \*X1, int n)

{

int i;

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

{

X0[i]=X1[i];

X1[i]=0;

}

}

float\*\* InitMemMatr(int N)

{

float \*\*A;

int i;

A = (float\*\*)calloc(N,sizeof(float\*));

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

A[i] = (float\*)calloc(N,sizeof(float));

return A;

}

float\* InitMemArr(int N)

{

float \*B, i;

B = (float\*)calloc(N,sizeof(float));

return B;

}

void FreeMem(float\*\* A, int M)

{

int i;

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

free(A[i]);

free(A);

}

