Министерство образования и науки РФ

Федеральное Государственное Бюджетное

Образовательное Учреждение Высшего Образования

Тульский Государственный Университет

Лабораторная работа №5

по дисциплине: «Численные методы»

Тема: «Решение уравнений эллиптического типа

методом сеток».

Вариант 21

Выполнил: студент гр. 221241 Ильина Е.А.

Принял: профессор каф. ПМиИ Баранов В.П.

Тула 2017

**1.Цель работы**

Приобретение навыков решения уравнений эллиптического типа методом сеток.

**2.Теоретические сведения**

Требуется найти решение уравнения

 (1)

в области,если

, (2)

где -граница области ; -заданная непрерывная функция.

Чтобы найти решение данной задачи методом сеток покроем область  прямоугольной сеткой :

; ,

где -точка, лежащая внутри области;  и  - шаги сетки по  и  соответственно; 

Заменим в узлах  производные  и  конечно- разностными соотношениями

 ;

 ;

Тогда для каждого внутреннего узла сетки уравнение (1) заменится конечно-разностным уравнением вида

 , (3)

где .

Границу данной области  заменим границей  сеточной области. Если узел сетки  лежит на границе области , то значение  в этом узле совпадает со значением  в данной точке. Если же граничный узел не лежит на границе, то можно выполнить одну из следующих процедур:

1. Положить, что в данном узле  функция  равна значению функции  в ближайшей точке  границы, отстоящей от данного узла  на расстояние  по оси  или

.

2. Для определения значения функции в граничном узле  использовать линейную интерполяцию

,

где -соседний внутренний узел, причем , если  лежит внутри области, и ,если  есть внешняя точка для области .

Выбор шагов  производится в зависимости от конкретной задачи, но таким образом, чтобы при этом контур  сеточной области  как можно лучше аппроксимировал контур  данной области .

От выбора  зависит также величина остаточного члена при замене дифференциального уравнения (1) конечно-разностным уравнением (3). Следовательно,  должны быть выбраны таким образом ,чтобы этот остаточный член был меньше погрешности, допустимой при решении.

Особенно простой вид примет система (3) при:

 (4)

Следовательно, чтобы решить задачу ,надо выбрать шаг сетки, построить сеточную область, найти значения в граничных узлах сетки, записать систему алгебраических уравнений для внутренних и граничных узлов сетки, решить полученную полную систему любым методом (метод Гаусса, метод Зейделя и т.д.). При этом погрешность приближенного решения задачи Дирихле будет складываться из трех погрешностей: погрешности замены дифференциального уравнения разностным, погрешности аппроксимации граничных условий, погрешности решения системы уравнений.

При большом числе внутренних узлов решение системы уравнений затруднительно. Чтобы решить задачу Дирихле в данном случае, применяют процесс Либмана.

Для этого выбирают начальные приближения . Теоретически в качестве этих значений можно выбрать любую систему чисел. Практически, чтобы найти значения , решают задачу Дирихле с большим шагом, обычно с шагом , чтобы получить систему меньшего числа уравнений, принимая значения  в граничных узлах равными значениям функции в ближайших точках границы. Значения функции во всех остальных внутренних узлах находят по формуле

.

Затем значения функции в граничных узлах  исправляют по формулам линейной интерполяции, а значения функции во внутренних узлах исправляют по формулам

.

Процесс продолжается до тех пор , пока не совпадут значения функций в двух последовательных приближениях.

**3.Задание**

Найти решение уравнения



в области , если на границе области 



Где:; ;

**4. Листинг программы**

include"stdafx.h"

#include<cstdlib>

#include<math.h>

#include<stdio.h>

#include<iostream>

enum { node\_inner, node\_outter, node\_edge };//перечислениетиповузлов

constint dim = 13;//количествоузлов

constfloatsmin = -3, smax = 3.0; // областьсимметричная

constfloat h = fabs(smax - smin) / float(dim);//шаготiдо i+1

constfloat eps = 0.001;// этопогрешность

boolcontflag = true, new\_contflag;// true -этоправда, false-ложь ,contflag-этофлажокобозначающийвнашемслучае true

int steps = 0;

intgetNodeType(float\_x, float\_y) {//получитьтипузла

if (pow(\_x, 2) / 9 + pow(\_y, 2) / 16 > 1)

returnnode\_outter;//тогда вернуть узел внешний

else// иначе

if (

pow(\_x - h, 2) / 9 + pow(\_y, 2) / 16 <= 1 &&

pow(\_x + h, 2) / 9 + pow(\_y, 2) / 16 <= 1 &&

pow(\_x, 2) / 9 + pow(\_y - h, 2) / 16 <= 1 &&

pow(\_x, 2) / 9 + pow(\_y + h, 2) / 16 <= 1

)

returnnode\_inner;// при выполнении вернуть узел внутренний

else//иначе

returnnode\_edge;// вернуть узел граничный , если предыдущие 2 не выполняются

}

void main() {

int\* umatrix = newint[dim\*dim];//матрица

float\* udata = newfloat[dim\*dim];// матрица преобразованная сюда записываем результативные данные

inttemp;// временная переменная (целочисленная)

floatftemp;

float delta;

for (int x = 0; x < dim; x++)

for (int y = 0; y < dim; y++) {

temp = getNodeType(smin + h \* x, smin + h \* y);

umatrix[y\*dim + x] = temp;//одномер.мат

if (temp == node\_edge)//если временная переменная temp содержит(относится ) к граничный узел

udata[y \* dim + x] = fabs(smin + h\*x)\*fabs(smin + h\*y);//это phi

else

udata[y \* dim + x] = 1.0;

}// Процесс Либмана

while (contflag) { // пока правда тогда выполняем следующее

steps++; //шаги ++ увеличиваем на 1

new\_contflag = false;

for (int I = 0; I < dim; I++) //пооси OX

for (int j = 0; j < dim; j++) // пооси OY

if (umatrix[I \* dim + j] == node\_inner) {

ftemp = 0.25 \* (udata[(I + 1) \* dim + j] +

udata[(I - 1) \* dim + j] +

udata[I \* dim + j - 1] +

udata[I \* dim + j + 1] + h \* h \* (-2) \* ((smin + I \* h)\*(smin + j \* h)));// f(x,y)

if (fabs(udata[I \* dim + j] - ftemp) > eps) {

udata[I \* dim + j] = ftemp;

new\_contflag = true;

}

}

contflag = new\_contflag;

}// Выводрезультатов

for (int I = 0; I < dim; I++) {

for (int j = 0; j < dim; j++)

if (umatrix[I \* dim + j] == node\_outter)

printf("- ");

else

printf("%-+#.2f ", udata[I\*dim + j]);

printf("\n");

}

getchar();

}

**5. Результат**

