|  |
| --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  Учреждение образования «Полоцкий государственный университет»    Факультет информационных технологий  Кафедра технологий программирования  ОТЧЕТ по  ЛАБОРАТОРНОЙ РАБОТЕ №7  По дисциплине  **«Методы численного анализа»**  На тему: «Решение задачи Дирихле в области простой геометрии»  ВЫПОЛНИЛ студент группы 15-КБ  Сонич А.Д.  ПРОВЕРИЛА ассистент кафедры ТП  Рудькова Т.С.  Полоцк, 2017 г. |

**Название**: «Решение задачи Дирихле в области простой геометрии».

**Цель работы**: Изучить методы сеток для решения задачи Дирихле в области простой геометрии.

**Теоретическая часть**:

**Разностная задача Дирихле в прямоугольной области**

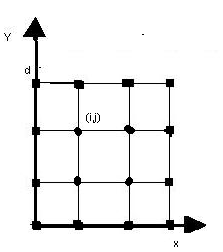
Будем рассматривать задачу Дирихле для уравнения Пуассона:

 (1)

 (2) краевое условие.

В предположение, что область *D* является прямоугольником, т.е. *D={0≤x≤a,0≤z≤d}* а функции *f(x,z), ϕ(x,z)* –заданные. Будем искать решение этой задачи (1)-(2) т.е. функцию *u(x,z)*, непрерывную в указанном прямоугольнике, конечно-разностным методом.

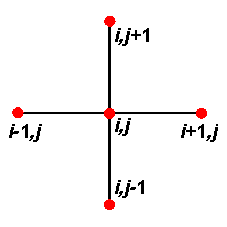
Для этого в области *D* введем сетку с равномерными шагами *h*1 и *h*2 соответственно по направлениям координатных осей *X,Z* таким образом, что *h*1=*a*/*N*1 и *h*2=*d*/*N*2.

Обозначим координаты по осям следующим образом: .

Т.о. мы получили прямоугольную сетку  (3) , которая состоит из совокупности узлов вида (*xi,zj*). Причем  - это совокупность всех внутренних узлов области D, а узлы, лежащие на границе - .

Обозначим функцию, заданную в узлах сетки (3) как.

Будем предполагать, что эта сеточная функция аппроксимирует искомое решение задачи(1) - (2) в узлах сетки, т.е.. Для аппроксимации вторых производных конечноразностными отношениями с минимальным количеством точек нужно использовать 5-точечный шаблон «крест»:



Получим конечно-разностную схему следующего вида:

 (7)

*i*=1,2,…,*N*1-1 *j*=1,2,…,*N*2-1

  *i*=0,..,*N*1 (8)

  *j*=0,..,*N*2 (9)

В теории разностных схем доказывается, что построенная схема имеет единственное решение, сама схема устойчива и имеет второй порядок аппроксимации по параметрам *h*1 и *h*2. Разностная схема (7) - (9) представляет собой систему линейных алгебраических уравнений, количество которых (N1-1)·(N2-1). Эта система может быть решена *методом Гаусса* (для небольшого числа неизвестных), но предпочтительнее использовать *итерационные методы*.

*Тогда каждое из уравнений системы необходимо записать в виде, разрешенном относительно сеточной функции*:

Если строится квадратная сетка, т.е. *h*1 = *h*2, то конечно-разностное уравнение (7) можно преобразовать к виду:

 (10)

Если в основе задачи Дирихле лежит уравнение Лапласа, то для квадратной сетки его конечно-разностная аппроксимация запишется следующим образом:

 (11)

**Практическая часть:**

Дана задача Дирихле для уравнения Лапласа  в квадрате *ABCD* с вершинами *A(0; 0), B(0; 1), C(1; 1), D(1; 0).* В таблице вариантов приведены формулы, задающие искомую функцию на сторонах квадрата *ABCD*. Используя метод сеток, составьте приближенное решение для этой задачи с шагом h = 0,2.

**Вариант №11**

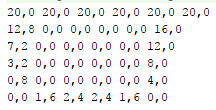
**Ход работы:**

Напишем программу для метода и оценим её точность.

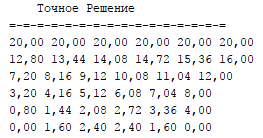
Листинг 1 – Программа метода написанная на Java.

**package** com.company;  
  
**public class** Main {  
 **public static double** func1(**double** x, **double** y){  
 **return** 20\*Math.*pow*(y,2);  
 }  
 **public static double** func2(**double** x, **double** y){  
 **return** 20;  
 }  
 **public static double** func3(**double** x, **double** y){  
 **return** 20\*y;  
 }  
 **public static double** func4(**double** x, **double** y){  
 **return** 10\*x\*(1-x);  
 }  
  
 **public static void** main(String[] args) {  
 **double** h = 0.2;  
 **double** u[][]=**new double**[6][6];  
 **double** U[][]=**new double**[6][6];  
 **double** x0 = 0, y0 = 1;  
 **for**(**int** i = 0;i<6; i++){  
 u[i][0]= *func1*(x0,y0);  
 y0 -= h;  
 }  
 x0=0.2;  
 y0=1;  
 **for**(**int** i = 1;i<6; i++){  
 u[0][i]= *func2*(x0,y0);  
 x0 += h;  
 }  
 x0=1;  
 y0=0.8;  
 **for**(**int** i = 1;i<6; i++){  
 u[i][5]= *func3*(x0,y0);  
 y0 -= h;  
 }  
 x0=0.2;  
 y0=0;  
 **for**(**int** i = 1;i<6; i++){  
 u[5][i]= *func4*(x0,y0);  
 x0 += h;  
 }  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j <6; j++) {  
 System.***out***.print(String.*format*(**"%.1f"**,(**float**)u[i][j]));  
 System.***out***.print(**" "**);  
 U[i][j]=u[i][j];  
 }  
 System.***out***.println();  
 }  
 **for** (**int** i = 1; i < 5; i++) {  
 **for** (**int** j = 1; j < 5; j++) {  
 u[j][i] = (u[j][i - 1] + u[j][i + 1] + u[j - 1][i] + u[j + 1][i]) / 4;  
 }  
 }  
 System.***out***.println(**"=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="**);  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j <6; j++) {  
 System.***out***.print(String.*format*(**"%.2f"**,(**float**)u[i][j]));  
 System.***out***.print(**" "**);  
 }  
 System.***out***.println();  
 }  
 **for**(**int** n = 0; n<100; n++) {  
 **for** (**int** i = 1; i < 5; i++) {  
 **for** (**int** j = 1; j < 5; j++) {  
 u[j][i] = (u[j][i - 1] + u[j][i + 1] + u[j - 1][i] + u[j + 1][i]) / 4;  
 }  
 }  
 }  
 System.***out***.println(**"=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="**);  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j <6; j++) {  
 System.***out***.print(String.*format*(**"%.2f"**,(**float**)u[i][j]));  
 System.***out***.print(**" "**);  
 }  
 System.***out***.println();  
 }  
 **double** ht=0;  
 **for** (**int** i = 1; i < 5; i++) {  
 ht = Math.*abs*(U[i][0]-U[i][5])/5;  
 **for** (**int** j = 1; j < 5; j++) {  
 U[i][j]=U[i][j-1] + ht;  
 }  
 }  
 System.***out***.println(**" Точное Решение\n=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="**);  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j <6; j++) {  
 System.***out***.print(String.*format*(**"%.2f"**,(**float**)U[i][j]));  
 System.***out***.print(**" "**);  
 }  
 System.***out***.println();  
 }  
 **double** pogresh[][]=**new double**[6][6];  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j < 6; j++) {  
 pogresh[i][j]=(Math.*abs*(U[i][j]-u[i][j]))/(u[i][j])\*100;  
 }  
 }  
 System.***out***.println(**" Погрешность\n=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="**);  
 **for** (**int** i = 0; i < 6; i++) {  
 **for** (**int** j = 0; j <6; j++) {  
 System.***out***.print(String.*format*(**"%.2f"**,(**float**)pogresh[i][j]));  
 System.***out***.print(**" "**);  
 }  
 System.***out***.println();  
 }  
 }  
}

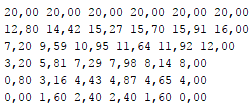
Построим в области решения квадратную сетку с шагом *h*=0,2 и вычислим значение функции в граничных узлах, а так же нанесем эти значения на сетку.

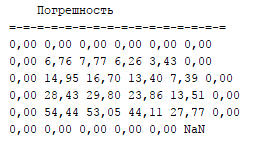


Прежде чем начать считать значения функций во внутренних узлах сетки с помощью формулы:  , найдем точное решение сетки:



Далее сосчитаем значения функций во внутренних узлах сетки :



Оценим погрешность в каждом узле сетки:  


Как мы видим максимальная погрешность составила 54.4%

**Выводы о проделанной работе.**

В данной лабораторной работе была изучена задача Дирихле для уравнения Лапласа в квадрате АВСD, используя 5-точечный шаблон «крест». В ходе работы была оценена погрешность метода, которая сильно различалась на разных участках сетки, и не превышала 54.4%.