**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

**Кафедра вычислительной математики**

**ЛАБОРАТОРНАЯ РАБОТА № 2**

**” Численное решение уравнения Пуассона ”**

Возовикова Никиты Александровича

студента 3 курса группы 9

специальности «Компьютерная Безопасность»

дневной формы получения

высшего образования

Научный руководитель:

Доцент

Никифоров Иван Васильевич

Минск, 2021

**ОГЛАВЛЕНИЕ**

**1. Постановка задачи**

**2. Краткие теоретические сведения**

**3. Листинг программы**

**4. Результаты**

**Постановка задачи**

Используя метод Якоби, метод Зейделя, метод переменных направлений, найти решение задачи Дирихле в области прямоугольной формы с указанными шагами при следующих входных данных:

**Краткие теоретические сведения**

Типичным уравнением в частных производных эллиптического типа является уравнение вида:

При μ ≠ 0, *f* ≠ 0 уравнение представляет собой неоднородное уравнение Гельмгольца, при μ ≠ 0, f = 0 – однородное уравнение Гельмгольца, при μ = 0 и *f* ≠ 0 — уравнение Пуассона и при μ = 0 и *f* = 0 — уравнение Лапласа.

Вводя в прямоугольнике *G* равномерную сетку, имеющую *N* шагов, но *х* и *М* шагов по *у,* аппроксимируем вторые производные в операторе Δ = ∂2/∂*x*2 + ∂2/∂*y*2 конечно-разностными формулами на пятиточечном шаблоне «крест» и построим разностную схему:

Приведём её к виду:

Здесь *hx* - шаг сетки по *х*, *hy* - шаг по *у*, (*xj yi*) — узлы сетки.

Схема имеет погрешность аппроксимации *O* (*h*2*x*, *h*2*y*), т. е. это схема второго порядка; значения *uji* при *j* = 0, *N* и *i* = 0, *M* задаются краевыми условиями. Уравнения представляют собой систему (*N —*1)(*М* —1) линейных алгебраических уравнений, которую можно записать в виде: *Au* = *b*.

Здесь *u* — вектор значений *uji* во всех внутренних узлах сетки, имеющий (*N* – 1) (*M* – 1) элементов; *b* — вектор правой части, включающий как значения *fji* во внутренних узлах сетки, так и краевые значения; *А* — симметричная редкая матрица размерности (*N* – l) (*M* – 1) × (*N* – 1) (*М* – 1).

Решим уравнение итерационным методом Зейделя:

где *k*=l, 2...— номер итерации.

Вкачестве начального приближения можно принять любые значения, например:

Вычисление предварительного значения методом Зейделя:

где *k* =1,2... — номер итерации.

Вычисление окончательного значения в узле (*j*, *i*) по релаксационной формуле:

где 0<ω <2 — релаксационный параметр.

Объединяя эти два этапа, получаем расчетную формулу:

Максимальная скорость сходимости итераций достигается при оптимальном значении релаксационного параметра:

где λmax — модуль максимального собственного значения матрицы метода простой итерации в правой части исходного уравнения, т. е. λ — это решение проблемы собственных значений:

Отыскивая решение уравнения в прямоугольнике *G* в виде:

находим собственные значения:

Максимальное значение λ достигается при *п = т =*1*:*

В частности, в случае квадратной области *G* при *N = M*:

В качестве начального приближения можно по-прежнему принять приведенные ранее значения.

Итерации прекращаются при выполнении условия:

где ε — заданная малая величина

**Листинг программы**

import matplotlib.pyplot as plt

def puasson(func, x, y, g, hx, hy):

a = 1 / (hx \*\* 2)

b = 1 / (hy \*\* 2)

c = 2 \* (a+b)

e = 1e-8

err = e+1

nx = int((x[1]-x[0])/hx)

ny = int((y[1]-y[0])/hy)

u = [[g(x[0] + hx\*i,y[0]+hy\*j) for i in range(ny-1)] for j in range(nx-1)]

while(err > e):

old = [list(i) for i in u]

for i in range(1,nx-2):

for j in range(1,ny-2):

u[i][j] = (a \* (u[i-1][j] + u[i+1][j]) + b \*

(u[i][j-1]+ u[i][j+1])- func(x[0] + hx\*i,y[0] + hy \* j))/c

err = max([max([abs(old[i][j] - u[i][j])

for i in range(len(u[i]))]) for i in range(len(u))])

return u

if \_\_name\_\_ == "\_\_main\_\_":

solution = puasson(

func= lambda x,y : 2.4 \* (x\*\*2) + 6 \* (y\*\*2),

x = (0, 1),

y = (0,1.5),

g= lambda x,y : 0.2 \* (x\*\*4) + 0.5\* (y \*\* 2) + 2,

hx=0.1,

hy=0.1

)

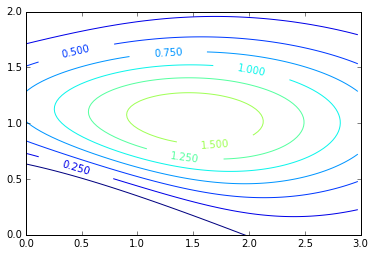
fig = plt.figure()

axes = fig.add\_subplot(projection='2d')

axes.plot\_surface(solution, edgecolor='black', linewidth=0.3)

plt.show()

**Результаты**



*Рис 1.1 График линий уровня*