Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

**Лабораторная работа № 3**

**«Метод сеток решения задачи Дирихле для уравнения Пуассона»**

по курсу «Методы численного анализа»

**Выполнил:**

Волчецкий А. М. гр. 253504

**Проверила:**

Ероминек К.Р.

Минск 2015

**Цель выполнения задания:**

* изучить метод разностных аппроксимаций для уравнения Пуассона,
* составить алгоритмы решения задачи Дирихле для уравнения Пуассона методом сеток применимыми для организации вычислений на ПЭВМ;
* составить программы решения задачи Дирихле для уравнения Пуассона по разработанным алгоритмам;
* выполнить тестовые примеры и проверить правильность работы программ.
* получить численное решение заданной задачи Дирихле для уравнения Пуассона.

Краткие теоретические сведения.

Пусть D = {(x, у): 0 < х < 1, 0 < y < 1} — открытый квадрат, Г — его граница, =  — замкнутый квадрат, f (х, у) — заданная на  достаточно гладкая функция.

Задача Дирихле состоит в следующем. Требуется найти непрерывную на  функцию u (х, у), удовлетво­ряющую на открытом квадрате D уравнению Пуассона

|  |  |  |
| --- | --- | --- |
|  | = f(x, y) | (1) |

и обращающуюся на границе квадрата в нуль, т. е.

|  |  |  |
| --- | --- | --- |
|  | u = 0 на Г. | (2) |

Задача Дирихле (1), (2) имеет единственное реше­ние u (х, у).

Положим h = l/N, = kh, = mh, . Построим сетки

={: k, m = 0, 1, . . . , N },

={: k, m = 1, 2, . . . , N-1},

=\ ( — множество узлов, лежащих на Г).

Зададим нормы

, .

Разностная схема:

Разностное уравнение соответствующее (1) имеет вид

|  |  |  |
| --- | --- | --- |
|  | . (3)  на , (4) |  |

Его шаблон изображен на рис. 4.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | ◦ |  |
|  | ◦ | ◦ | ◦ |
|  |  | ◦ |  |
|  |  |  |  |
|  |  |  |  |
|  | Рис. 4. | |  |

Решение  разностной задачи Дирихле (3), (4) находится методом последовательных приближений по схеме пе­ременных направлений, где ,  —произвольные. Можно доказать, что

, k, m = 1, 2, . . . , N-1,

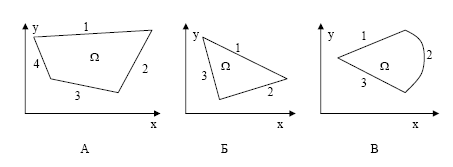
при любых начальных приближениях , причем наи­большая скорость сходимости достигается при . Здесь положена в основу идея о стабилизации при t  решения уравнения теплопроводности к решению уравнения Пуассона, если f не зависит от t.

**ЗАДАНИЕ.**

Решить двухмерное стационарное уравнение Пуассона



Область Ω представляет собой либо четырехугольник (А), либо треугольник (Б), либо сектор (В).



Участки границы Гi обозначим номерами 1,2,3,4. На границах Гi заданы следующие условия:

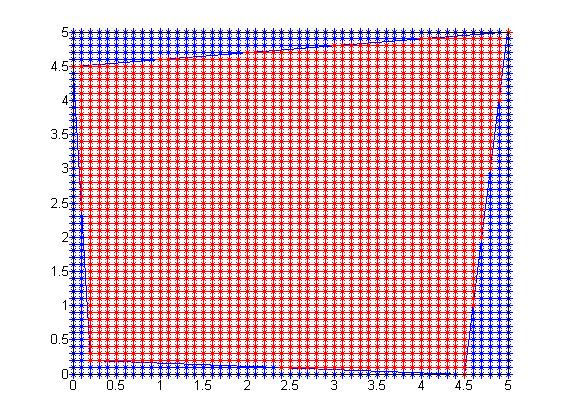


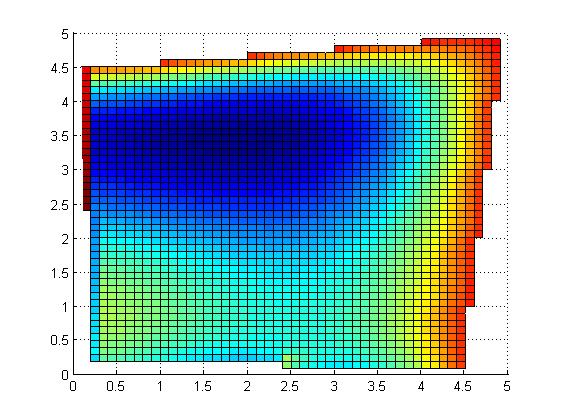
Требуется в соответствии с выбранным вариантом,

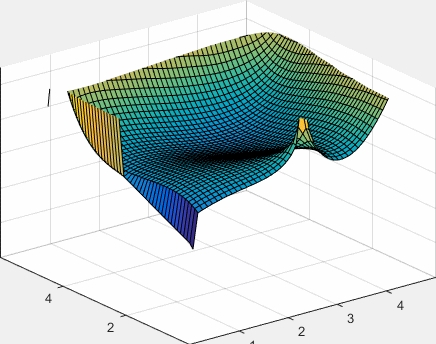
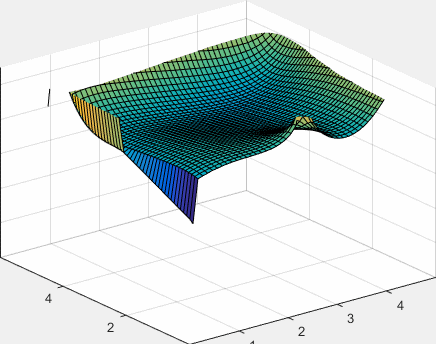
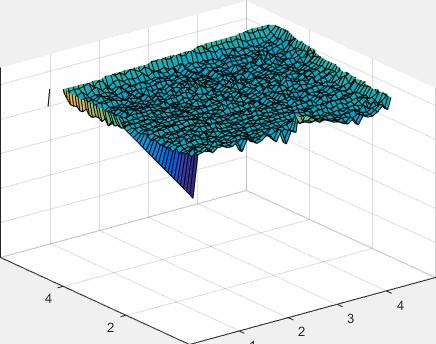
1) Построить область, задав ее координатами вершин (выбрать самостоятельно).

1. 2) Решить уравнение, используя последовательность сгущающихся сеток (три раза), проанализировать сходимость метода и погрешность решения.
2. Вариант 8:
3. Область Б.
4. *g(x,y)=1+0.5\*cos(x+y);*
5. *f(x,y)=x+y;*
6. *q1 = 0, q2 = 0, q3 = 1;*
7. *a1 = 1, a2 = 1, a3 = 0.5;*
8. *b1 = 3, b2 = 3, b3 = 0.2;*

**Демонстрация работы.**

Заданная область в виде четырехугольника:

Решение:



**Реализация алгоритма.**

syms x y

q = [0 0 1];

a = [1 1 0.5];

b = [3 3 0.2];

f\_xy = inline('x+y', 'x', 'y');

g\_xy = inline('1+0.5\*cos(x+y)', 'x', 'y');

h = 0.1;

x0 = 0;

xn = 5;

y0 = 0;

yn = 5;

x = x0 : h : xn;

y = y0 : h : yn;

n = length(x);

m = length(y);

u = rand(m, n);

tau = h / pi;

sigma = tau / h^2;

[X Y] = meshgrid(x, y);

xv = [0 5 4.5 0.2];

yv = [4.5 5 0 0.2];

xv = [xv xv(1)];

yv = [yv yv(1)];

in = inpolygon(X, Y, xv, yv);

u(~in) = NaN;

plot(xv, yv, X(in), Y(in), 'r\*', X(~in), Y(~in), 'b\*');

eps = 0.05;

iteration = 0;

figure;

surf(x, y, u);

zlim([-10 1]);

f = getframe;

[im, map] = rgb2ind(f.cdata, 256);

im(1, 1, 1, 1) = 0;

for k = 1 : fix(max(n, m) / (2 \* pi) \* log(1 / eps))

u\_temp = zeros(m, n);

u\_temp(~in) = NaN;

for j = 1 : m

[A B C F alpha beta] = InitMas(n);

ind = find(~isnan(u(j, :)));

if isempty(ind)

continue;

elseif length(ind) < 3

u\_temp(j, ind) = 0;

else

for i = ind(2) : 1 : ind(end - 1)

if j == 1 && ~isnan(u(j, i))

[A(i) C(i) B(i) F(i)] = InitBCn(a, b, q, h, 3);

elseif j == m && ~isnan(u(j,i))

[A(i) C(i) B(i) F(i)] = InitBC0(a, b, q, h, 1);

elseif isnan(u(j - 1, i))

[A(i) C(i) B(i) F(i)] = InitBCn(a, b, q, h, 3);

elseif isnan(u(j + 1, i))

[A(i) C(i) B(i) F(i)] = InitBC0(a, b, q, h, 1);

else

A(i) = 0.5 \* sigma \* g\_xy(x(i), y(j));

C(i) = 1 + 0.5 \* sigma \* (g\_xy(x(i) + h/2, y(j)) + ...

g\_xy(x(i) - h/2, y(j)));

B(i) = 0.5 \* sigma \* g\_xy(x(i), y(j));

F(i) = -0.5 \* sigma \* u(j - 1, i) - (1 - sigma) \* u(j, i) - 0.5 \* sigma \* u(j + 1, i) + ...

f\_xy(x(i), y(j)) \* 0.5 \* tau;

end;

end;

[A(ind(1)) C(ind(1)) B(ind(1)) F(ind(1))] = InitBC0(a, b, q, h, 4);

[A(ind(end)) C(ind(end)) B(ind(end)) F(ind(end))] = InitBCn(a, b, q, h, 2);

alpha(ind(1)) = B(ind(1)) / C(ind(1));

beta(ind(1)) = -F(ind(1)) / C(ind(1));

for i = ind(2) : 1 : ind(end)

alpha(i) = B(i) / (C(i) - A(i) \* alpha(i - 1));

beta(i) = - (F(i) - A(i) \* beta(i - 1)) / (C(i) - A(i) \* alpha(i - 1));

end;

u\_temp(j, ind(end)) = beta(ind(end));

for i = ind(end - 1) : -1 : ind(1)

u\_temp(j, i) = alpha(i) \* u\_temp(j, i + 1) + beta(i);

end;

end;

end;

for j = 1 : n

[A B C F alpha beta] = InitMas(m);

ind = find(~isnan(u\_temp(:, j)));

if isempty(ind)

continue;

elseif length(ind) < 3

u(ind, j) = 0;

else

for i = ind(2) : 1 : ind(end - 1)

if j == 1 && ~isnan(u\_temp(i, j))

[A(i) C(i) B(i) F(i)] = InitBC0(a, b, q, h, 4);

elseif j == n && ~isnan(u\_temp(i, j))

[A(i) C(i) B(i) F(i)] = InitBCn(a, b, q, h, 2);

elseif isnan(u\_temp(i, j - 1))

[A(i) C(i) B(i) F(i)] = InitBC0(a, b, q, h, 4);

elseif isnan(u\_temp(i, j + 1))

[A(i) C(i) B(i) F(i)] = InitBC0(a, b, q, h, 2);

else

A(i) = 0.5 \* sigma;

C(i) = 1 + sigma;

B(i) = 0.5 \* sigma;

F(i) = -0.5 \* sigma \* g\_xy(x(j), y(i)) \* u\_temp(i, j - 1) - ...

(1 - 0.5 \* sigma \* (g\_xy(x(j) + h/2, y(i)) + g\_xy(x(j) - h/2, y(i)))) \* u\_temp(i, j) - ...

0.5 \* sigma \* u\_temp(i, j + 1) \* g\_xy(x(j), y(i)) + ...

f\_xy(x(j), y(i)) \* 0.5 \* tau;

end;

end;

[A(ind(1)) C(ind(1)) B(ind(1)) F(ind(1))] = InitBC0(a, b, q, h, 3);

[A(ind(end)) C(ind(end)) B(ind(end)) F(ind(end))] = InitBCn(a, b, q, h, 1);

alpha(ind(1)) = B(ind(1)) / C(ind(1));

beta(ind(1)) = -F(ind(1)) / C(ind(1));

for i = ind(2) : 1 : ind(end)

alpha(i) = B(i) / (C(i) - A(i) \* alpha(i - 1));

beta(i) = - (F(i) - A(i) \* beta(i - 1)) / (C(i) - A(i) \* alpha(i - 1));

end;

u(ind(end), j) = beta(ind(end));

for i = ind(end - 1) : -1 : ind(1)

u(i, j) = alpha(i) \* u(i + 1, j) + beta(i);

end;

end;

end;

iteration = iteration + 1;

surf(x, y, u);

zlim([-10 1]);

f = getframe;

im(:,:,:,iteration) = rgb2ind(f.cdata, map);

end;

fprintf('Количество итераций: %g.\n', iteration);

imwrite(im, map, 'test.gif', 'DelayTime', 0, 'LoopCount', inf);

**Выводы.**

В процессе выполнения данной лабораторной работы мною был изучен метод разностных аппроксимаций для уравнения Пуассона, составлен алгоритм решения задачи Робена и, в частности, Дирихле для уравнения Пуассона.