Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

Лабораторная работа №5

«Применение базовых методов решения ДУЧП2 эллиптического типа»

ДИСЦИПЛИНА: «Моделирование»

Выполнил: студент гр. ИУК4-62Б ______________ (Калашников А.С.)
(подпись) (Ф.И.О.)

Проверил: ______________ (Никитенко У.В.)
(подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2023

Цель работы: сформировать практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для стандартных вычислений и визуализации результатов численного или приближенно-аналитического решения ДУЧП2 эллиптического типа на основе сравнения результатов.

Задачи: решить уравнение, указанное в варианте численными методами и оценить точность аппроксимации. Оценить устойчивость и сходимость. Выбрать среду для проведения расчетов и вычислительного эксперимента. Написать программу, реализующую решение разностной задачи. Оценить результаты расчетов. Визуализировать результаты, сравнить результаты, выдвинуть и обосновать гипотезу целесообразности использования того или иного метода в зависимости от предложенной задачи и ее вариаций, точности результата, трудоемкости, сложности алгоритма, сложности обоснования применимости метода, вычислительной эффективности алгоритма.

Задание

Рассматривается задача Дирихле для эллиптического уравнения

$$-Lu = f(x,y), (x,y) \in G$$
$$u = \mu(x,y), (x,y) \in \Gamma$$

Пусть $\bar{G} = G \cup \Gamma = \{0 \le x \le l_x, 0 \le y \le l_y\}$ – прямоугольник, а

$$Lu = \frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right)$$

Здесь $p(x,y), q(x,y)$ – достаточное гладкие функции такие, что $0 < c_1 \le p(x,y) \le c_2, 0 < d_1 \le q(x,y) \le d_2$, где c_1, c_2, d_1, d_2 – постоянные

Вариант №14

$$Lu = -f(x,y),$$

$$\text{где } Lu = \frac{\partial}{\partial x}\left((3\,x+2)\frac{\partial u}{\partial x}\right) + \frac{\partial^2 u}{\partial y^2},\ 0 < x < 1,\ 0 < y < 1,$$

$$u(x,y)|_\Gamma = \mu(x,y)$$

Задание №4

Найти решение задачи попеременно-треугольным итерационным методом с Чебышевским набором параметров;

Решение:

Имеем уравнение вида:

$$Lu = \frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial x}(q(x,y)\frac{\partial u}{\partial x})$$

Исходя из уравнения и условия задачи можем получить

$$p(x,y) = (3x+2)$$

$$q(x,y) = 1$$

$$u^*(x,y) = x^2y^2(1+y)$$

$$l_x = 1$$

$$l_y = 1$$

$$0 < c_1 \le p(x,y) \le c_2, 0 < d_1 \le q(x,y) \le d_2, \text{где } c_1, c_2, d_1, d_2 - \text{постоянные}$$

$$c_1 = \frac{1}{3}$$

$$c_2 = 1$$

$$d_1 = 1$$

$$d_2 = 1$$

Расчетная формула для метода итерации с оптимальным параметром в общем случае:

$$u_{ij}^k = u_{ij}^{k-1} + \tau(p_{i+\frac{1}{2}j}\frac{u_{i+1j}^{k-1} - u_{ij}^{k-1}}{h_x^2} - p_{i-\frac{1}{2}j}\frac{u_{ij}^{k-1} - u_{i-1j}^{k-1}}{h_x^2} +$$

$$+q_{ij+\frac{1}{2}}\frac{u_{ij+1}^{k-1} - u_{ij}^{k-1}}{h_y^2} - q_{ij-\frac{1}{2}}\frac{u_{ij}^{k-1} - u_{ij-1}^{k-1}}{h_y^2} + f_{ij})$$

В формуле есть следующие элементы:

$$h_x = \frac{l_x}{N}$$

$$h_y = \frac{l_y}{M}$$

$$x_i = ih_x$$

$$y_i = jh_y$$

$$\omega = \frac{2}{\sqrt{\delta\Delta}}, \gamma_1 = \frac{\delta}{(2 + 2\sqrt{n})}, \gamma_2 = \frac{2}{h^2} sin\pi \frac{h}{2}$$

$$\delta = \frac{8}{h^2} sin^2\pi \frac{h}{2}, \Delta = \frac{8}{h^2}, n = sin^2\pi \frac{h}{2}, \omega = \frac{h^2}{4sin\pi \frac{h}{2}}$$

$$\gamma_1 = \frac{4sin^2\pi \frac{h}{2}}{h^2(1 + sin\pi \frac{h}{2})}$$

$$\gamma_2 = \frac{2}{h^2} sin\pi \frac{h}{2}$$

$$\tau_k = \frac{2}{\gamma_2 + \gamma_1 + (\gamma_2 - \gamma_1)cos\frac{2k-1}{2n}\pi}, \qquad k = 1,2, \dots, n;$$

Граничные условия:

$$u_{i0}^k = \mu(x_i, 0) = 0$$

$$u_{iM}^k = \mu(x_i, l_y) = 2x_i$$

$$u_{0J}^k = \mu\left(0, y_j\right) = 0$$

$$u_{N0}^k = \mu\left(l_x, y_j\right) = y_j(1 + y_j)$$

Решение:

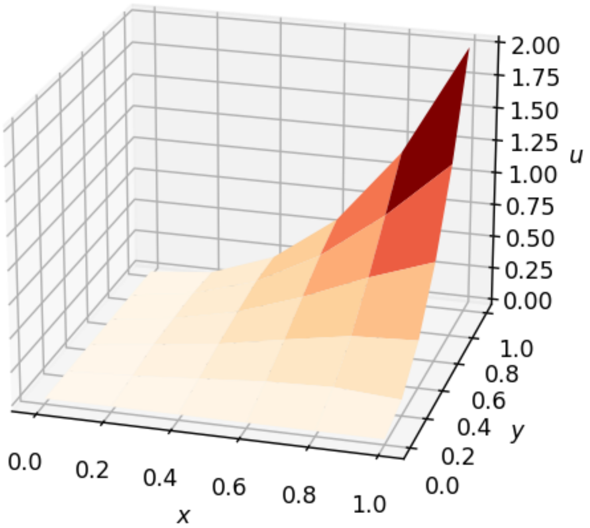

Рис. **1**. График функции

|| F-Au^* || = 9.149172613007357

|| F-AU^0 || = 1750.6355565067051

$$k_{max} = 23$$

k	|| F - AU^k ||	rel.d	|| U^k - u* ||	rel.error	|| U^k - U^(k-1) ||
0	1750.6356	1.0	1.5703	1.0	2.0
1	1179.5041	0.6738	1.2308	0.7838	0.3654
2	829.9393	0.4741	1.1844	0.7543	0.2487
3	603.1116	0.3445	1.1307	0.72	0.1786
4	448.9429	0.2564	1.0655	0.6785	0.1339
5	375.3049	0.2144	0.9945	0.6333	0.1039
6	313.2817	0.179	0.9213	0.5867	0.0917
7	259.7275	0.1484	0.8691	0.5534	0.0817
8	231.7233	0.1324	0.8403	0.5351	0.0732
9	204.0063	0.1165	0.8061	0.5133	0.0702
10	176.4754	0.1008	0.7664	0.4881	0.0651
11	149.8826	0.0856	0.7212	0.4592	0.0587
12	127.0285	0.0726	0.6702	0.4268	0.0421
13	108.602	0.062	0.6147	0.3915	0.0332
14	89.7403	0.0513	0.577	0.3674	0.0201
15	73.6088	0.042	0.5324	0.339	0.0163
16	58.6886	0.0335	0.4758	0.303	0.0101
17	45.3815	0.0259	0.4108	0.2616	0.0094
18	31.8216	0.0182	0.5123	0.2202	0.0087
19	20.2442	0.0116	0.4786	0.3001	0.0077
20	9.7232	0.0056	0.3401	0.2328	0.0071
21	5.7483	0.0033	0.1828	0.1253	0.0067
22	6.0221	0.0032	0.0889	0.0832	0.0063
23	6.0221	0.0032	0.0455	0.051	0.006

Рис. **2**. Решения

y \ x	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.2	0.0	0.00192	0.00768	0.01728	0.03072	0.048
0.4	0.0	0.00896	0.03584	0.08064	0.14336	0.224
0.6	0.0	0.02304	0.09216	0.20736	0.36864	0.576
0.8	0.0	0.04608	0.18432	0.41472	0.73728	1.152
1.0	0.0	0.08	0.32	0.72	1.28	2.0

Рис. **3**. Точное решение на крупной сетке

y \ x	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.2	0.0	0.0115	0.0222	0.0303	0.0412	0.053
0.4	0.0	0.0493	0.0903	0.1424	0.1905	0.231
0.6	0.0	0.1231	0.2411	0.3586	0.4732	0.589
0.8	0.0	0.2417	0.4722	0.6999	0.9338	1.162
1.0	0.0	0.4	0.8	1.2	1.6	2.0

Рис. **4**. Решение на крупной сетке

Вывод: в ходе выполнения работы были сформированы практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для стандартных вычислений и визуализации результатов численного или приближенно-аналитического решения ДУЧП2 параболического типа на основе сравнения результатов.

Приложение

Листинг

```
import numpy as np
import matplotlib.pyplot as plt
from prettytable import PrettyTable

f_p = lambda x = None, y = None: 1 + x/2
f_q = lambda x = None, y = None: 1
f_f = lambda x, y: -6*x*y - 2*x - y**2*(y + 1)/2
f_mu = lambda x, y: x**2*y**2*(1 + y)

l_x = 1
l_y = 1

c_1 = 1/3
c_2 = 1
d_1 = 1
d_2 = 1

f_u_exact = lambda x, y: x**2*y**2*(1 + y)

l_u = lambda x, y, u, i, j, h_x, h_y: \
    f_p(x + h_x/2, y) * (u[i + 1, j] - u[i, j]) / pow(h_x, 2) \
    - f_p(x - h_x/2, y) * (u[i, j] - u[i - 1, j]) / pow(h_x, 2) \
    + f_q(x, y + h_y/2) * (u[i, j + 1] - u[i, j]) / pow(h_y, 2) \
    - f_q(x, y - h_y/2) * (u[i, j] - u[i, j - 1]) / pow(h_y, 2)


def FU(x, y, h_x, h_y, last_u, i, j, k):
    tau_k = 2/(Delta + delta + (Delta - delta)*np.cos((2*k-1)*np.pi/(2*n)))

    p_plus = f_p(x + h_x/2, y)
    p_minus = f_p(x - h_x/2, y)
    q_plus = f_q(x, y + h_y/2)
    q_minus = f_q(x, y - h_y/2)

    return last_u[i, j] + tau_k*(p_plus*(last_u[i + 1, j] - last_u[i,
j])/h_x**2 - \
                             p_minus*(last_u[i, j] - last_u[i - 1,
j])/h_x**2 + \
                             q_plus*(last_u[i, j + 1] - last_u[i,
j])/h_y**2 - \
                             q_minus*(last_u[i, j] - last_u[i, j -
1])/h_y**2 + \
                                f_f(x, y))

lambda_max = 1 - pow(h_x, 2) / 4 * delta
lambda_min = 1 - pow(h_x, 2) / 4 * Delta

k_list = []
exact_diff = []
last_diff = []
rel_d = []
rel_error = []
discrepancies = []
# diff = []
# rhos = []
```

```
u_exact = np.array([[f_u_exact(x, y) for x in xs] for y in ys])

k = 0
eps = 5e-2
k_max = np.log(1/eps) / 4 / eps

LU = np.zeros((N, M))
F = np.zeros((N, M))

for i in range(1, N-1):
    for j in range(1, M-1):
        LU[i, j] = l_u(xs[i], ys[j], u_exact, i, j, h_x, h_y)
        F[i, j] = f_f(xs[i], ys[j])

print(f'|| F-Au^* || = {np.amax(np.abs(LU + F))}')

u = np.zeros((N, M))
LU = np.zeros((N, M))
F = np.zeros((N, M))

last_u = np.copy(u)
last_last_u = np.copy(u)

u[:, 0] = f_mu(xs, 0)
u[:, -1] = f_mu(xs, l_x)
u[0, :] = f_mu(0, ys)
u[-1, :] = f_mu(l_y, ys)
for i in range(1, N-1):
    for j in range(1, M-1):
        u[i, j] = FU(xs[i], ys[j], h_x, h_y, last_u, i, j, 0)

for i in range(1, N-1):
    for j in range(1, M-1):
        LU[i, j] = l_u(xs[i], ys[j], u, i, j, h_x, h_y)
        F[i, j] = f_f(xs[i], ys[j])

discrepancy_0 = np.amax(np.abs(LU + F))

print(f'|| F-AU^0 || = {discrepancy_0}')


u = np.zeros((N, M))

while len(exact_diff) == 0 or exact_diff[-1] > eps:
    last_last_u = np.copy(last_u)
    last_u = np.copy(u)

    u[:, 0] = f_mu(xs, 0)
    u[:, -1] = f_mu(xs, l_x)
    u[0, :] = f_mu(0, ys)
    u[-1, :] = f_mu(l_y, ys)


    for i in range(1, N-1):
        for j in range(1, M-1):
            u[i, j] = FU(xs[i], ys[j], h_x, h_y, last_u, i, j, k)

    LU = np.zeros((N, M))
```

```
    F = np.zeros((N, M))

    for i in range(1, N-1):
        for j in range(1, M-1):
            LU[i, j] = l_u(xs[i], ys[j], u, i, j, h_x, h_y)
            F[i, j] = f_f(xs[i], ys[j])

    if k == 0:
        u_0 = np.copy(u)
        u_0_diff = np.amax(np.abs(u_0 - u_exact))

    # rhos.append(np.amax(np.abs(u - last_u))/np.amax(np.abs(last_u -
last_last_u)))

    k_list.append(k)
    exact_diff.append(np.amax(np.abs(u - u_exact)))
    last_diff.append(np.amax(np.abs(u - last_u)))
    rel_error.append(np.amax(np.abs(u - u_exact))/u_0_diff)
    discrepancies.append(np.amax(np.abs(LU + F)))
    rel_d.append(discrepancies[-1] / discrepancy_0)
    #diff.append(rho * np.amax(np.abs(u - last_u)) / (1 - rho))

    k += 1

    if k > n: break

for k in range(130):
    last_u = np.copy(U)

    U[:, 0] = f_mu(xs, 0)
    U[:, -1] = f_mu(xs, l_x)
    U[0, :] = f_mu(0, ys)
    U[-1, :] = f_mu(l_y, ys)


    for i in range(1, 5):
        for j in range(1, 5):
            U[i, j] = FU(xs[i], ys[j], h_x, h_y, last_u, i, j, k)


fig = plt.figure()
ax = plt.axes(projection='3d')
X, Y = np.meshgrid(xs, ys)
ax.plot_surface(X, Y, U, rstride=1, cstride=1,
    cmap='OrRd', edgecolor='none')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$u$')
plt.show()

table.add_column("y \ x", ys)
for k in range(len(xs)):
    table.add_column(f"{xs[k]}", U[:, k])

print("Решение на крупной сетке")
print(table)

u_exact = np.array([[f_u_exact(x, y) for x in xs] for y in ys])
table = PrettyTable()
xs = xs.round(5)
```

```
ys = ys.round(5)
u_exact = u_exact.round(5)
table.add_column("y \ x", ys)
for k in range(len(xs)):
    table.add_column(f"{xs[k]}", u_exact[:, k])

print("Точное решение на крупной сетке")
print(table)
```