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 ________________ (Карельский М.К.)
(Подпись)

Проверил: ________________ (Никитенко У.В.)
(Подпись)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2023

Цель: 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 12

Найти решение задачи

$$Lu = -f(x,y),$$
$$\text{где } Lu = \frac{\partial}{\partial x}\left(\left(1+\frac{x}{2}\right)\frac{\partial u}{\partial x}\right) + \frac{\partial^2 u}{\partial y^2}, 0 < x < 1, 0 < y < 1$$
$$u(x,y)|_\Gamma = \mu(x,y)$$

Итерационным методом с чебышевским набором параметров.

В качестве критерия конца вычислений использовать условие: $\|U^k - u^*\| < \varepsilon$

Отладить решение задачи на функции $u^*(x,y) = xy^2(1+y)$

Решение:

Исходя из условия имеем:

$$\mathrm{p}(x,y) = 1 + \frac{x}{2}$$
$$q(x,y) = 1$$
$$f(x,y) = -6xy - 2x - \frac{y^2(y+1)}{2}$$
$$\mu(x,y) = xy^2(1+y)$$

$$l_x = 1$$
$$l_y = 1$$

$$c_1 = 1$$
$$c_2 = 1.5$$
$$d_1 = 1$$
$$d_2 = 1$$

Согласно методу с чебышевским набором параметров:

$$u_{ij}^k = u_{ij}^{k-1} + \tau_k \left(p_{i+\frac{1}{2}j} \frac{u_{i+1j}^{k-1} - u_{ij}^{k-1}}{h_x^2} - p_{i-\frac{1}{2}j} \frac{u_{ij}^{k-1} - u_{i-1j}^{k-1}}{h_x^2} + q_{ij+\frac{1}{2}} \frac{u_{ij+1}^{k-1} - u_{ij}^{k-1}}{h_y^2} - q_{ij-\frac{1}{2}} \frac{u_{ij}^{k-1} - u_{ij-1}^{k-1}}{h_y^2} + f_{ij} \right)$$

В формуле присутствуют следующие элементы:

$$h_x = \frac{l_x}{N}$$
$$h_y = \frac{l_y}{M}$$

$$x_i = ih_x$$
$$y_j = jh_y$$

$$p_{i+\frac{1}{2}j} = p\left(x_i + \frac{h_x}{2}, y_j\right)$$
$$p_{i-\frac{1}{2}j} = p\left(x_i - \frac{h_x}{2}, y_j\right)$$
$$q_{ij+\frac{1}{2}} = q\left(x_i, y_j + \frac{h_y}{2}\right)$$
$$q_{ij-\frac{1}{2}} = q\left(x_i, y_j - \frac{h_y}{2}\right)$$

$$f_{ij} = f(x_i, y_j)$$

$$\tau_k = \frac{2}{\Delta + \delta + (\Delta - \delta)\cos\frac{2k-1}{2n}\pi}, k = 1,2,\dots,n$$

$$\delta = c_1 \frac{4}{h_x^2}\sin^2\frac{\pi h_x}{2l_x} + d_1 \frac{4}{h_y^2}\sin^2\frac{\pi h_y}{2l_y}$$
$$\Delta = c_2 \frac{4}{h_x^2}\cos^2\frac{\pi h_x}{2l_x} + d_2 \frac{4}{h_y^2}\cos^2\frac{\pi h_y}{2l_y}$$

Граничные условия:

$$u_{i0} = \mu(x_i, 0) = 0$$
$$u_{iM} = \mu(x_i, l_y) = 2x_i$$
$$u_{0j} = \mu(0, y_j) = 0$$
$$u_{Nj} = \mu(l_x, y_j) = y_j(1 + y_j)$$

Выбор точности при $\varepsilon = 0.05$:

$$\xi = \frac{\delta}{\Delta}$$
$$m = \frac{\ln\frac{2}{\varepsilon}}{2\sqrt{\xi}} = 23$$

Полученное решение:

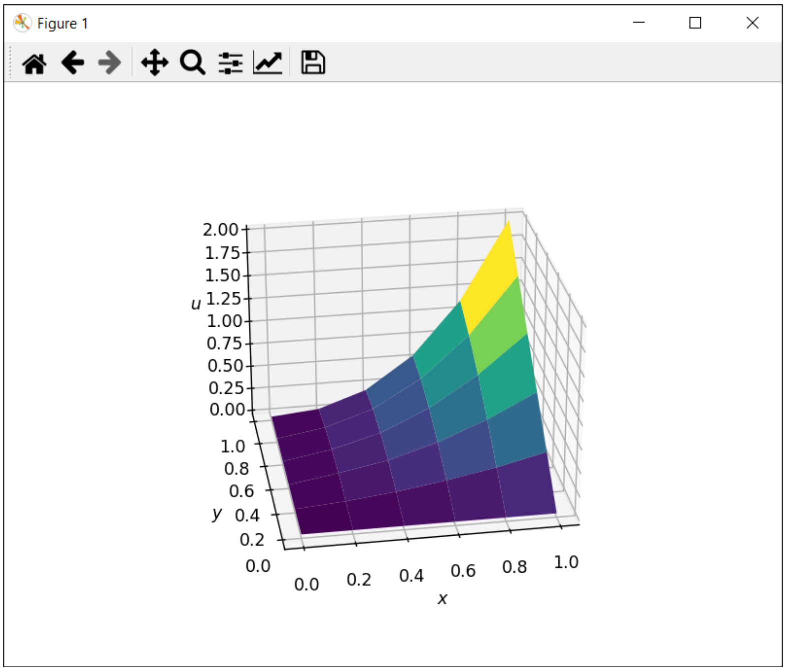

Рис. 1. График функции

$$\|F - Au^*\| = 6.02207504627261$$
$$\|F - AU^0\| = 1790.5247531826053$$
$$k_{max} = 23$$

k	|| F - AU^k ||	rel.d	|| U^k - u* ||	rel.error	|| U^k - U^(k-1) ||
0	1790.5248	1.0	1.6575	1.0	2.0
1	1212.1085	0.677	1.3786	0.8317	0.3737
2	856.8798	0.4786	1.326	0.8	0.2556
3	625.5507	0.3494	1.27	0.7662	0.1844
4	482.2046	0.2693	1.2019	0.7251	0.1389
5	405.988	0.2267	1.1491	0.6933	0.1116
6	339.2036	0.1894	1.1019	0.6648	0.0991
7	281.6304	0.1573	1.052	0.6347	0.0884
8	246.4078	0.1376	1.0081	0.6082	0.0794
9	217.881	0.1217	0.9715	0.5861	0.0702
10	189.3578	0.1058	0.933	0.5629	0.0651
11	161.6366	0.0903	0.902	0.5442	0.0587
12	142.3101	0.0795	0.8655	0.5222	0.0421
13	122.0314	0.0682	0.8227	0.4964	0.0332
14	101.253	0.0565	0.7781	0.4694	0.0201
15	82.0543	0.0458	0.7362	0.4442	0.0163
17	53.296	0.0298	0.6383	0.3851	0.0094
18	38.1268	0.0213	0.5123	0.38	0.0087
19	25.3069	0.0141	0.4786	0.3001	0.0077
20	13.1827	0.0074	0.3401	0.2328	0.0071
21	7.6747	0.0043	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. Характеристики

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. 3. 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	0.0096	0.0192	0.0288	0.0384	0.048
0.4	0.0	0.0448	0.0896	0.1344	0.1792	0.224
0.6	0.0	0.1152	0.2304	0.3456	0.4608	0.576
0.8	0.0	0.2304	0.4608	0.6912	0.9216	1.152
1.0	0.0	0.4	0.8	1.2	1.6	2.0

Рис. 4. Точное решение на крупной сетке

Вывод: в ходе выполнения лабораторной работы были получены 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.

ПРИЛОЖЕНИЯ

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*y**2*(1 + y)
l_x = 1
l_y = 1
c_1 = 1
c_2 = 1.5
d_1 = 1
d_2 = 1
f_u_exact = lambda x, y: x*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)

M = 20
N = M

u_0 = np.zeros((N, M))
u_0_diff = 0
u = np.zeros((N, M))
xs = np.linspace(0, l_x, N)
ys = np.linspace(0, l_y, M)

print(ys[-1])
print(xs[-1])

h_x = l_x/N
h_y = l_y/M
delta = 4/pow(h_x, 2) * pow(np.sin(np.pi*h_x/2/np.pi), 2) \
    +  8/pow(h_y, 2) * pow(np.sin(np.pi*h_y/2/np.pi), 2)

Delta = 4/pow(h_x, 2) * pow(np.cos(np.pi*h_x/2/np.pi), 2) \
    +  8/pow(h_y, 2) * pow(np.cos(np.pi*h_y/2/np.pi), 2)
n = 23

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 = []

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):
```

```python
            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))

        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)
        k += 1

        if k >= n: break

    table = PrettyTable()
    table.add_column("k", np.array(k_list).round(4))
    table.add_column("|| F - AU^k ||", np.array(discrepancies).round(4))
    table.add_column("rel.d", np.array(rel_d).round(4))
```

```python
table.add_column("|| U^k - u* ||", np.array(exact_diff).round(4))
table.add_column("rel.error", np.array(rel_error).round(4))
table.add_column("|| U^k - U^(k-1) ||", np.array(last_diff).round(4))
print(table)

xs = np.linspace(0, l_x, 6)
ys = np.linspace(0, l_y, 6)
h_x = (xs[1] - xs[0]) / 2
h_y = (ys[1] - ys[0]) / 2
U = np.zeros((6, 6))

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='viridis', edgecolor='none')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$u$')
plt.show()

table = PrettyTable()
xs = xs.round(5)
ys = ys.round(5)
U = U.round(5)

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