Московский авиационный институт
(Национальный исследовательский университет)
Факультет прикладной математики и физики
Кафедра вычислительной математики и программирования

Лабораторная работа № 7
по курсу «Численные методы»

Студент: Аксенов А. Е.
Группа: М80-408Б-20
Преподаватель: Пивоваров Д. Е.
Оценка:

Москва, 2023

Лабораторная №7

Задание

Решить краевую задачу для дифференциального уравнения эллиптического типа. Аппроксимацию уравнения произвести с использованием центрально-разностной схемы. Для решения дискретного аналога применить следующие методы: метод простых итераций (метод Либмана), метод Зейделя, метод простых итераций с верхней релаксацией. Вычислить погрешность численного решения путем сравнения результатов с приведенным в задании аналитическим решением $U(x, y)$. Исследовать зависимость погрешности от сеточных параметров h_x, h_y.

Вариант 1

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0,$$

$u(0, y) = y,$

$u(1, y) = 1 + y,$

$u(x, 0) = x,$

$u(x, 1) = 1 + x.$

Аналитическое решение: $U(x, y) = x + y$.

Теоретический материал

Классическим примером уравнения эллиптического типа является уравнение Пуассона

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y), \qquad (1)$$

Или уравнение Лапласа при $f(x, y) = 0$

Первая краевая задача для уравнения Лапласа или Пуассона называется задачей Дирихле

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} = f(x, y), (x, y)\epsilon\Omega; & (2) \\ u(x, y)|_1 = \varphi(x, y), (x, y)\epsilon\Gamma; & (3) \end{cases}$$

Если на границе Г задается нормальная производная искомой функции, то соответствующая вторая краевая задача называется задачей Неймана для уравнения Лапласа или Пуассона

$$\begin{cases} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), (x,y)\epsilon\Omega; & (4) \\ \frac{\partial u(x,y)}{\partial n}|_\Gamma = \varphi(x,y), (x,y)\epsilon\Gamma; & (5) \end{cases}$$

При этом n – направление внешней к границе Г нормали.

Третья краевая задача для уравнения Пуассона (Лапласа) имеет вид:

$$\begin{cases} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), (x,y)\epsilon\Omega; & (4) \\ \frac{\partial u(x,y)}{\partial n}|_\Gamma + \alpha u|_\Gamma = \varphi(x,y), (x,y)\epsilon\Gamma; & (5) \end{cases}$$

Разностные схемы для аппроксимации:

Для решения – строим сетку по x и y. И на ней аппроксимируем задачу во внутренних узлах с помощью отношения конечных разностей по следующей схеме:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_1^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h_2^2} + O\left(h_1^2 + h_2^2\right) = f(x_i, y_j),$$
$$i = \overline{1, N_1 - 1}, \quad j = \overline{1, N_2 - 1}$$

В результате получаем слау, которую можно решить разными итерационными методами.

В данной лабораторной будут использованы итерационные методы Либмана, Зейделя, и простых итераций с верхней релаксацией.

Ключевые моменты программы

Метод Либмана:

```
x_begin = 0.0
x_end = 1.0
y_begin = 0.0
y_end = 1.0

def liebman_method(N_x = 30, N_y = 30, epsilon = 1e-5, max_iter = 10**4):
    h_x = (x_end - x_begin) / (N_x - 1)
    h_y = (y_end - y_begin) / (N_y - 1)

    x = np.linspace(x_begin, x_end, N_x)
    y = np.linspace(y_begin, y_end, N_y)

    X = [x for _ in range(N_y)]
    Y = [[y[i] for _ in x] for i in range(N_y)]
    ans = [[0 for _ in range(N_x)] for _ in range(N_y)]
    for i in range(N_x):
        coeff = (cond_4(X[-1][i]) - cond_3(X[0][i])) / (y_end - y_begin)
        bias = cond_3(X[0][i])
        for j in range(N_y):
            ans[j][i] = coeff * (Y[j][i] - y_begin) + bias

    coeffs = [h_x*h_x / (h_x*h_x + h_y*h_y), h_y*h_y / (h_x*h_x + h_y*h_y)]

    iter_number = 0
    norm = 2 * epsilon

    while(norm >= epsilon and iter_number < max_iter):
        last_line = [[0 for _ in range(N_x)] for _ in range(N_y)]
        norm = 0
        for i in range(1, N_y - 1):
            last_line[i][0] = cond_1(Y[i][0])
            diff = abs(last_line[i][0] - ans[i][0])
            norm = max(diff, norm)
            for j in range(1, N_x - 1):
                last_line[i][j] = (coeffs[0] * (ans[i][j-1] + ans[i][j+1]) / 2 + coeffs[1] * (ans[i-1][j] + ans[i+1][j]) / 2
                diff = abs(last_line[i][j] - ans[i][j])
                norm = max(diff, norm)
            last_line[i][-1] = cond_2(Y[i][-1])
            diff = abs(last_line[i][-1] - ans[i][-1])
            norm = max(diff, norm)

        for i in range(1, N_y - 1):
            ans[i] = last_line[i]

        iter_number += 1

    return X, Y, ans
```

Метод Зейделя

```
def zeidel_method(N_x = 30, N_y = 30, epsilon = 1e-5):
    return relaxation_method(N_x=N_x, N_y=N_y, epsilon=epsilon, w=1.0)
```

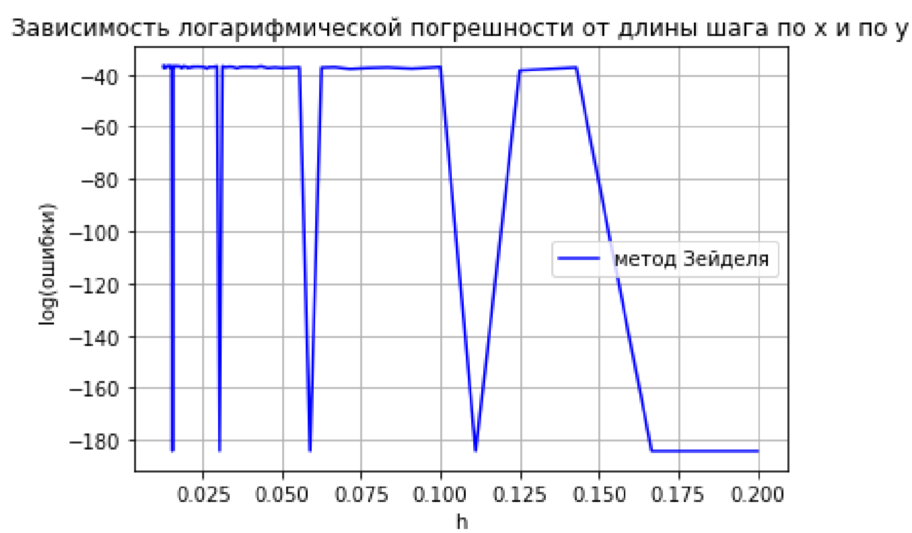
Метод верхней релаксации

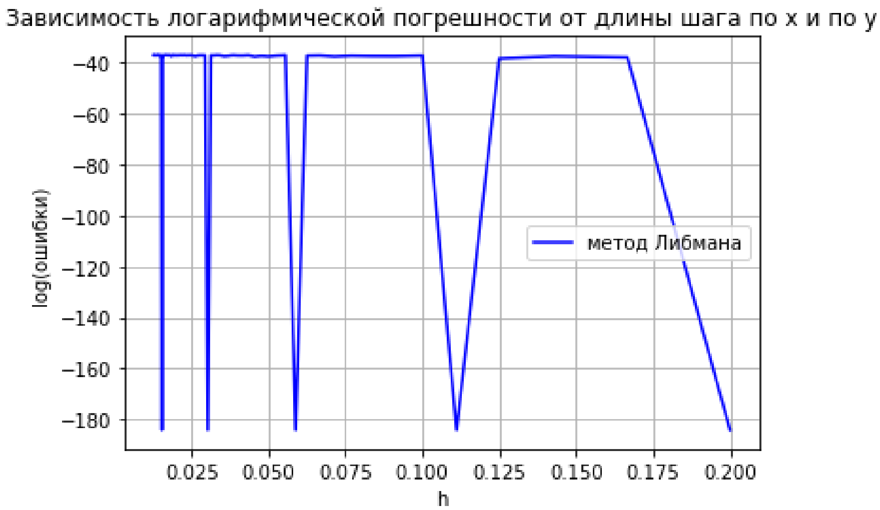
```python
x_begin = 0.0
x_end = 1.0
y_begin = 0.0
y_end = 1.0

def relaxation_method(N_x = 30, N_y = 30, epsilon = 1e-5, w = 0.01, max_iter = 10**4):
    h_x = (x_end - x_begin) / (N_x - 1)
    h_y = (y_end - y_begin) / (N_y - 1)

    x = np.linspace(x_begin, x_end, N_x)
    y = np.linspace(y_begin, y_end, N_y)

    X = [x for _ in range(N_y)]
    Y = [[y[i] for _ in x] for i in range(N_y)]
    ans = [[0 for _ in range(N_x)] for _ in range(N_y)]
    for i in range(N_x):
        coeff = (cond_4(X[-1][i]) - cond_3(X[0][i])) / (y_end - y_begin)
        bias = cond_3(X[0][i])
        for j in range(N_y):
            ans[j][i] = coeff * (Y[j][i] - y_begin) + bias

    coeffs = [h_x*h_x / (h_x*h_x + h_y*h_y), h_y*h_y / (h_x*h_x + h_y*h_y)]

    iter_number = 0
    norm = 2 * epsilon

    while(norm >= epsilon and iter_number < max_iter):
        norm = 0
        for i in range(1, N_y - 1):
            diff = w * (cond_1(Y[i][0]) - ans[i][0])
            ans[i][0] += diff
            diff = abs(diff)
            norm = max(diff, norm)
            for j in range(1, N_x - 1):
                diff = w * (coeffs[0] * (ans[i][j-1] + ans[i][j+1]) / 2 + coeffs[1] * (ans[i-1][j] + ans[i+1][j]) / 2 - ans[
                ans[i][j] += diff
                diff = abs(diff)
                norm = max(diff, norm)
            diff = w * (cond_2(Y[i][-1]) - ans[i][-1])
            ans[i][-1] += diff
            diff = abs(diff)
            norm = max(diff, norm)

        iter_number += 1
    return X, Y, ans
```

Ошибки

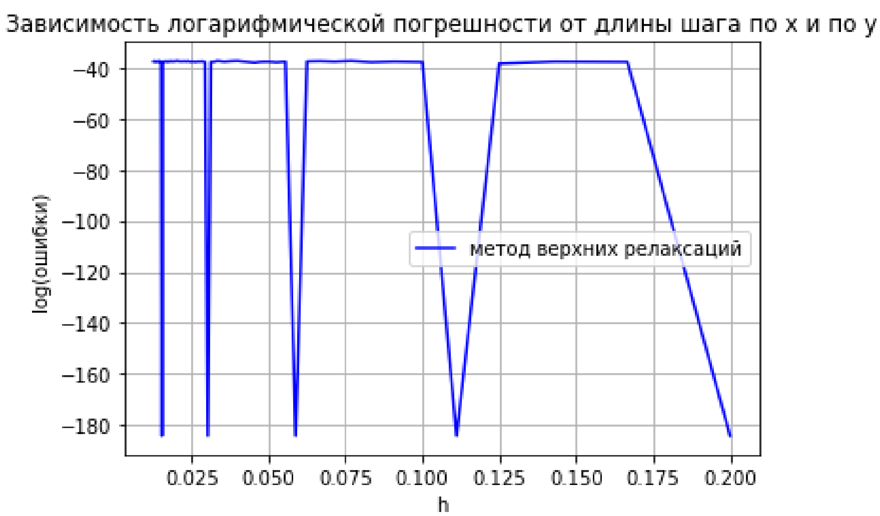
h

```python
logerror_plot(zeidel_method, "метод Зейделя")
```





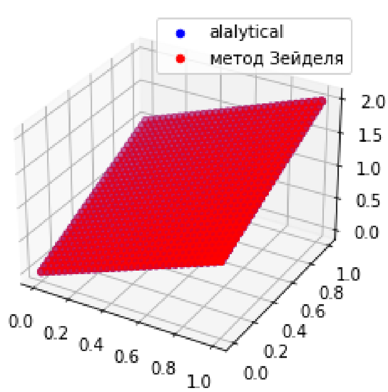

```
logerror_plot(relaxation_method, "метод верхних релаксаций")
```



Аналитические и численные решения

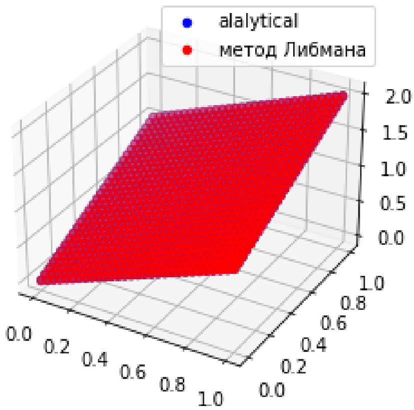
```
R3_plot(zeidel_method, "метод Зейделя")
```

```
RMSE =  8.817462896590622e-17
```



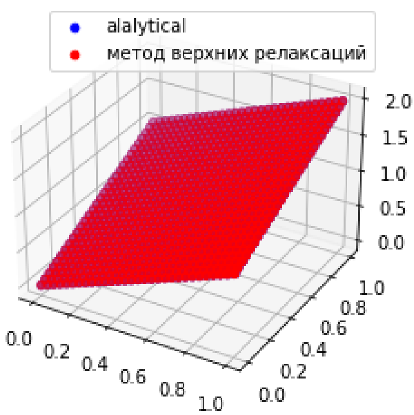
```
R3_plot(liebman_method, "метод Либмана")
```

RMSE = 8.685913635774265e-17




```
R3_plot(relaxation_method, "метод верхних релаксаций")
```

RMSE = 6.326032273250456e-17



Вывод

Были реализованы методы для решения задачи эллиптического типа:

Разностная схема и метод Либмана

Разностная схема и метод Зейделя

Метод верхней релаксации

Также мной были отрисованы графики аналитического и численного решения. Графики почти совпадают. Данный факт говорит о том, что алгоритмы сходятся.