Министерство образования и науки Российской Федерации
Федеральное государственное баджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Фундаментальные науки»
Кафедра «Высшая математика»

Отчёт

по технологической практике за 5 семестр 2020-2021 учебного года

Студент ФН1-51 (Группа) _____ (Подпись, дата) **А.Ю. Митрофанова** (И.О.Фамилия)

Руководитель практики ст. преп. кафедры ФН1 _____ (Подпись, дата) **О.В. Кравченко** (И.О.Фамилия)

Москва, 2020г.

Постановка задачи

Решить сингулярную двуточечную краевую задачу методом конечных разностей

$$\varepsilon y''(x) + p(x)y'(x) + q(x)y(x) + f(x) = 0, \quad x \in [0,\, 1],$$

со смешанными граничными условиями

$$\begin{aligned} -\alpha_1 y'(0) + \alpha_2 y(0) &= \gamma_1, \\ \beta_1 y'(1) + \beta_2 y(1) &= \gamma_2, \end{aligned}$$

при различных значениях параметра

$$\varepsilon = 1,\ 0.1,\ 0.01,\ 0.001.$$

$$p(x) = \sqrt{x^2 + x}, \qquad q(x) = \sqrt{x - x^2}, \qquad f(x) = \sqrt{x^2 - x^3} + x.$$

$$\alpha_1 = 0,\ \alpha_2 = 1,\ \beta_1 = 0,\ \beta_2 = 1,\ \gamma_1 = 6,\ \gamma_2 = 9.$$

Метод конечных разностей

Пусть имеем дифференциальное уравнение второго порядка

$$\varepsilon y''(x) + p(x)y'(x) + q(x)y(x) + f(x) = 0, \qquad x \in [a, b] \tag{1}$$

со смешанными граничными условиями

$$\begin{cases} -\alpha_1 y'(a) + \alpha_2 y(a) = \gamma_1, \\ \beta_1 y'(b) + \beta_2 y(b) = \gamma_2. \end{cases} \tag{2}$$

Разобьём интервал $[a, b]$ на узлы

$$a \leqslant x_0 < x_1 < ... < x_n \leqslant b$$

и введём обозначения $y(x_i) = y_i$, $p(x_i) = p_i$, $q(x_i) = q_i$, $f(x_i) = f_i$.

Выберем постоянный шаг разбиения h так, что $h = x_i - x_{i-1}$, $i = \overline{1, n}$ и заменим в дифференциальном уравнении (1) первые производные конечными разностями. На концах отрезка

$$y'(x_0) = \frac{y_1 - y_0}{h}, \qquad y'(x_n) = \frac{y_n - y_{n-1}}{h}$$

и в промежуточных точках

$$y'(x) = \frac{dy}{dx} \approx \frac{\Delta y}{\Delta x} = \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}} = \frac{y_{i+1} - y_{i-1}}{2h}.$$

Конечные разности для второй производной имеют вид

$$y''(x) = \frac{d^2y}{dx^2} \approx \frac{\Delta(\Delta y)}{\Delta(\Delta x)} = \frac{(y_{i+1} - y_i) - (y_i - y_{i-1})}{h^2} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}.$$

Теперь вместо дифференциального уравнения (1) с граничными условиями (2) имеем систему

$$\begin{cases} -\alpha_1 \dfrac{y_1 - y_0}{h} + \alpha_2 y_0 = \gamma_1, \\ \varepsilon \dfrac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + p_i \dfrac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = -f_i, \qquad i = 1, ..., n-1, \\ \beta_1 \dfrac{y_n - y_{n-1}}{h} + \beta_2 y_n = \gamma_2, \end{cases}$$

состоящую из $n + 1$ уравнений с таким же количеством неизвестных $y_0, y_1, ..., y_n$.

Умножив первое и последнее уравнение системы на h, а остальные — на h^2, перепишем в матричной форме $\boldsymbol{W} \cdot \boldsymbol{Y} = \boldsymbol{F}$, где

$$\boldsymbol{W} = \begin{pmatrix} \alpha_1 + \alpha_2 h & -\alpha_1 & 0 & 0 & 0 & 0 & 0 \\ \varepsilon - \dfrac{p_1 h}{2} & q_1 h^2 - 2\varepsilon & \varepsilon + \dfrac{p_1 h}{2} & 0 & 0 & 0 & 0 \\ 0 & \varepsilon - \dfrac{p_2 h}{2} & q_2 h^2 - 2\varepsilon & \varepsilon + \dfrac{p_2 h}{2} & 0 & 0 & 0 \\ 0 & 0 & . & . & . & . & 0 \\ 0 & 0 & 0 & 0 & \varepsilon - \dfrac{p_{n-1} h}{2} & q_{n-1} h^2 - 2\varepsilon & \varepsilon + \dfrac{p_{n-1} h}{2} \\ 0 & 0 & 0 & 0 & 0 & -\beta_1 & \beta_1 + \beta_2 h \end{pmatrix},$$

$$\boldsymbol{Y} = \begin{pmatrix} y_0 \\ y_1 \\ ... \\ y_{n-1} \\ y_n \end{pmatrix}, \qquad \boldsymbol{F} = \begin{pmatrix} \gamma_1 h \\ -f_1 h^2 \\ ... \\ -f_{n-1} h^2 \\ \gamma_2 h \end{pmatrix}.$$

Таким образом, решение сингулярной двуточечной краевой задачи (1),(2) сводится к решению СЛАУ. В нашем случае применим для этой цели метод прогонки.

Листинг программы

```
import numpy as np
import matplotlib.pyplot as plt

def define_plot(title='', x_label='$x$', y_label='$y$', size=14, grid=True, y_phi=90, label_rot=0):
    fig, ax = plt.subplots()
    ax.set_title(title, fontsize=size)
    ax.set_xlabel(x_label, fontname="Times New Roman", fontsize=size)
    ax.set_ylabel(y_label, fontname="Times New Roman", fontsize=size, rotation=y_phi)

    ax.tick_params(axis='x', labelrotation = label_rot)    #  Поворот подписей

    for tick in ax.get_xticklabels():
        tick.set_fontname("Times New Roman")
        tick.set_fontsize(size)
    for tick in ax.get_yticklabels():
        tick.set_fontname("Times New Roman")
        tick.set_fontsize(size)

    if grid:
        ax.grid()
    return fig, ax

# сетка узлов x_i, p(x_i)=p_i, q(x_i)=x_i, f(x_i)=f_i
def find_params(a, b, h):
    x = np.arange(a, b + h, h)
    p = np.sqrt(x[1:-1] ** 2 + x[1:-1])
    q = np.sqrt(x[1:-1] - x[1:-1] ** 2)
    f = np.sqrt(x[1:-1] ** 2 - x[1:-1] ** 3) + x[1:-1]
    return x, p, q, f

# метод прогонки
def tridiagonal_algorithm(matrix, f):
    def direct_run(a, b, c, d, n):
        p = np.zeros(n)
        q = np.zeros(n)
        p[0] = -b[0] / a[0]
        q[0] = d[0] / a[0]
        for i in range(1, n - 1):
            p[i] = -b[i] / (a[i] + c[i] * p[i - 1])
            q[i] = (d[i] - c[i] * q[i - 1]) / (a[i] + c[i] * p[i - 1])
        q[-1] = (d[-1] - c[-1] * q[n - 2]) / (a[-1] + c[-1] * p[n - 2])
        return p, q
    def reverse_run(p, q, n):
        x = np.zeros((n, 1))
        x[-1] = q[-1]
        for i in range(n - 2, -1, -1):
            x[i] = p[i] * x[i + 1] + q[i]
        return x
    n = len(matrix)
    vec_a = np.diag(matrix)
    vec_b = np.hstack((np.diag(matrix, 1), 0))  # np.reshape(np.hstack((np.diag(W, 1), 0)), (len(W), 1))
    vec_c = np.hstack((0, np.diag(matrix, -1)))
    alpha, beta = direct_run(vec_a, vec_b, vec_c, f, n)
    vec_x = reverse_run(alpha, beta, n)
    return vec_x
```

```
a = 0
b = 1
n = 10
eps_list = [1, 0.1, 0.01, 0.001]
a1 = 0
a2 = 1
b1 = 0
b2 = 1
g1 = 6
g2 = 9
colors_list = ['blue', 'red', 'green', 'darkorange']

fig, ax = define_plot(y_phi=0)

for i, eps in enumerate(eps_list):
    h = (b - a) / n
    x, p, q, f = find_params(a, b, h)
    diag_dn = np.hstack((eps - p * h / 2, -b1))
    diag_0 = np.hstack((a2 * h +a1, q * (h ** 2) - 2 * eps, b2 * h + b1))
    diag_up = np.hstack((-a1, eps + p * h / 2))
    W = np.diag(diag_dn, k=-1) + np.diag(diag_0, k=0) + np.diag(diag_up, k=1)
    F = np.hstack((g1 * h, -f * (h ** 2), g2 * h))
    Y = tridiagonal_algorithm(W, F)
    n *= 2

    ax.plot(x, Y, '-o', color=colors_list[i], label='$\\epsilon=$' + str(eps))
ax.legend(loc='best')
fig.savefig('practice', dpi=300)
```

Результаты

На рис. 1 представлены решения сингулярной двуточечной краевой задачи, найденные с помощью метода конечных разностей. Заметим, что при уменьшении параметра ε локальные максимумы кривых $y(x)$ возрастают и сдвигаются по x ближе к начальной точке отрезка a, также потребовалось увеличивать количество узлов для отображения гладких кривых.

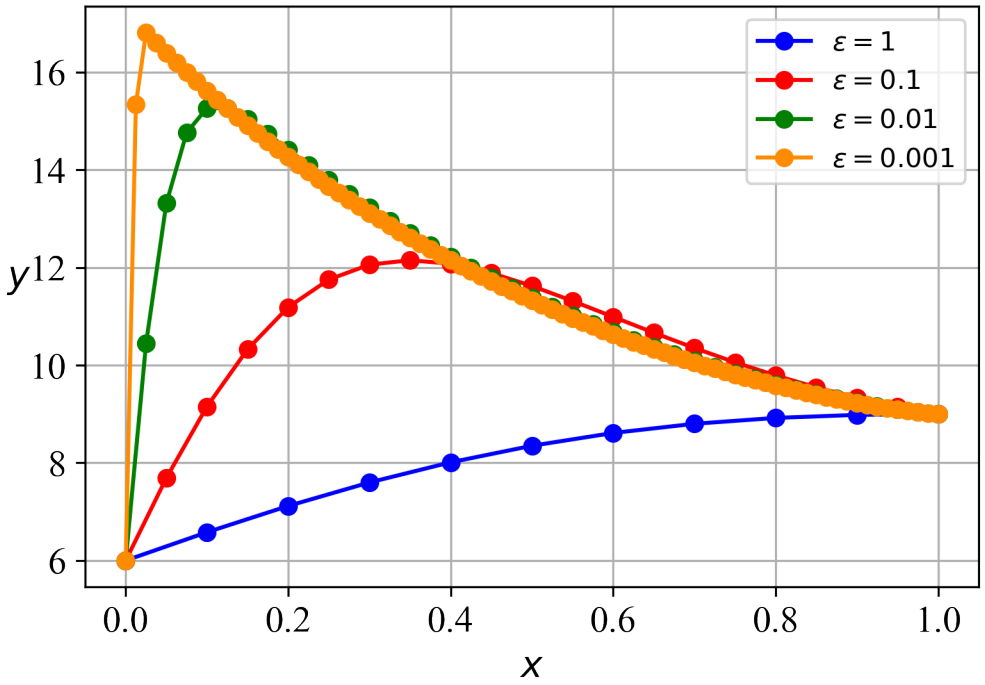

Рис. 1. Решения задачи (1), (2) для разных ε

Список литературы

[1] Блюмин А.Г., Федотов А.А., Храпов П.В. Численные методы вычисления интегралов и решения задач для обыкновенных дифференциальных уравнений: Методические указания к выполнению лабораторных работ по курсу «Численные методы». — М.: МГТУ им. Н.Э.Баумана, 2008. — 74 с.

[2] Самарский А.А. Введение в численные методы. СПб.: Издательство «Лань», 2005. — 288 с.

[3] Щетинина Е.В. Методы возмущений и решение обыкновенных дифференциальных уравнений: методические указания / Е.В. Щетинина. — Самара: Изд-во "Универс групп 2010. — 31с.