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

Факультет «Фундаментальные науки»
Кафедра ФН1 «Высшая математика»

Отчет

по учебной практике за 5 семестр

Преподаватель ____________________ Кравченко О. В.
подпись, инициалы

Студент группы ФН1–51Б ____________________ Байбикова А. В.
подпись, инициалы

Москва
2020

Содержание

1 Задание **3**

2 Описание метода **3**

3 Листинг кода **4**

3.1 Метод конечных разностей 4

3.2 Метод прогонки 5

4 Результаты работы программы **6**

1 Задание

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

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

и функциях

$$p(x) = \sqrt{x^2 + x}, \quad q(x) = \sin^2(x) + \cos(x), \quad f(x) = x^3 - x^2 + x$$

2 Описание метода

Поставленная краевая задача решается с помощью перехода от исходной задачи к новой, записанной в конечно-разностной форме. Тогда решение новой задачи будет являться приближенным решением исходной задачи. В силу того, что первая и вторая производные, входящие в уравнение и в краевые условия, будут заменены приближенными конечно-разностными формулами, решения с применением метода конечных разностей получается не в виде непрерывной функции $y(x)$, а виде таблицы ее значений в отдельных точках. Для этого разобьем отрезок $[0, 1]$ на n частей так, чтобы $x_0 = 0, x_n = 1, x_{i+1} = x_i + h, h = (b - a)/n$. Наша задача – найти значения функции $y(x)$ в точках $x_k, k = 0...n$. Для того, чтобы перейти от исходной задачи к конечно-разностной, надо получить формулы для представления первой и второй производных в конечно-разностном виде. Они получаются, если применить разложение функции $y(x)$ в окрестности некоторой точки x_k в ряд Тейлора, ограничиваясь вторыми производными:

$$y(x_i + h) \approx y(x_i) + hy'(x_i) + \frac{h^2}{2} y''(x_i)$$

$$y(x_i - h) \approx y(x_i) - hy'(x_i) + \frac{h^2}{2} y''(x_i)$$

Складывая (вычитая) эти выражения, получим приближенные выражения для второй (первой) производных:

$$y''(x_i) \approx \frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2}$$

$$y'(x_i) \approx \frac{y(x_i + h) - y(x_i - h)}{2h}$$

Обозначим $y(x_i) = y_i,\ y(x_i + h) = y_{i+1},\ y(x_i - h) = y_{i-1}$, а также $p(x_i) = p_i,\ q(x_i) = q_i,\ f(x_i) = f_i$.

В итоге, получим для узловых точек сетки:

$$\varepsilon \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = -f_i, \quad i = 1...n-1 \tag{1}$$

И, для краевых условий:

$$-\alpha_1 \frac{y_1 - y_0}{h} + \alpha_2 y_0 = \gamma_1 \tag{2}$$

$$\beta_1 \frac{y_n - y_{n-1}}{h} + \beta_2 y_n = \gamma_2 \tag{3}$$

Мы получили СЛАУ с неизвестными $y_0 ... y_n$, матрица которой имеет трёхдиагональный вид. Такую систему можено решить, например, методом прогонки.

3 Листинг кода

3.1 Метод конечных разностей

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

def p(x):
    return np.sqrt(x * x + x)

def q(x):
    return np.sin(x)*np.sin(x) + np.cos(x)

def f(x):
    return x * x * x - x * x + x

eps = 1;
a1 = 0;
a2 = 1;
b1 = 0;
b2 = 1;
g1 = 3;
g2 = 6;
h = 0.05;
x0 = 0;
x1 = 1;

n = round(abs(x1 - x0) / h);
X = np.zeros([n + 1, 1]);
F = np.zeros([n + 1, 1]);
A = np.zeros([n + 1, n + 1]);

for i in range(n+1):
    X[i] = x0 + i * h;


for i in range(1,n):
        A[i][i]    = ((-2*eps) / (h * h)) + q(X[i]);
        A[i][ i-1] = (eps / (h * h)) - p(X[i]) / (2 * h);
```

```
        A[i][i+1] = (eps / (h * h)) + p(X[i]) / (2 * h);
        F[i]      = -f(X[i]);

F[0]    = g1;
F[n] = g2;
A[0][0] = (a1 / h) + a2;
A[0][1] = -a1 / h;
A[n][ n] = (b1 / h) + b2;
A[n][n-1]      = -b1 / h;
Y = progonka(A, F);
plt.plot(X, Y)
```

3.2 Метод прогонки

```
def progonka(A, D):
        n = np.size(A,1);
        a=np.diagonal(A);
        b=np.diagonal(A,1);
        c=np.diagonal(A,-1);
        P=np.zeros([n,1]);
        Q=np.zeros([n+2,1]);
        X=np.zeros([n,1]);

        P[1]= b[0]/(-a[0]);
        Q[1]= D[0]/(a[0]);

        for i in range(1,n-1):
                P[i+1]=b[i]/(-a[i]-c[i-1]*P[i]);
                Q[i+1]=(-D[i]+c[i-1]*Q[i])/(-a[i]-c[i-1]*P[i]);

        Q[n]=(-D[n-1]+c[n-2]*Q[n-1])/(-a[n-1]-c[n-2]*P[n-1]);
        X[n-1]=Q[n];

        for i in reversed(range(0,n-1)):
            X[i] = P[i+1]*X[i+1]+ Q[i+1];
        return X
```

4 Результаты работы программы

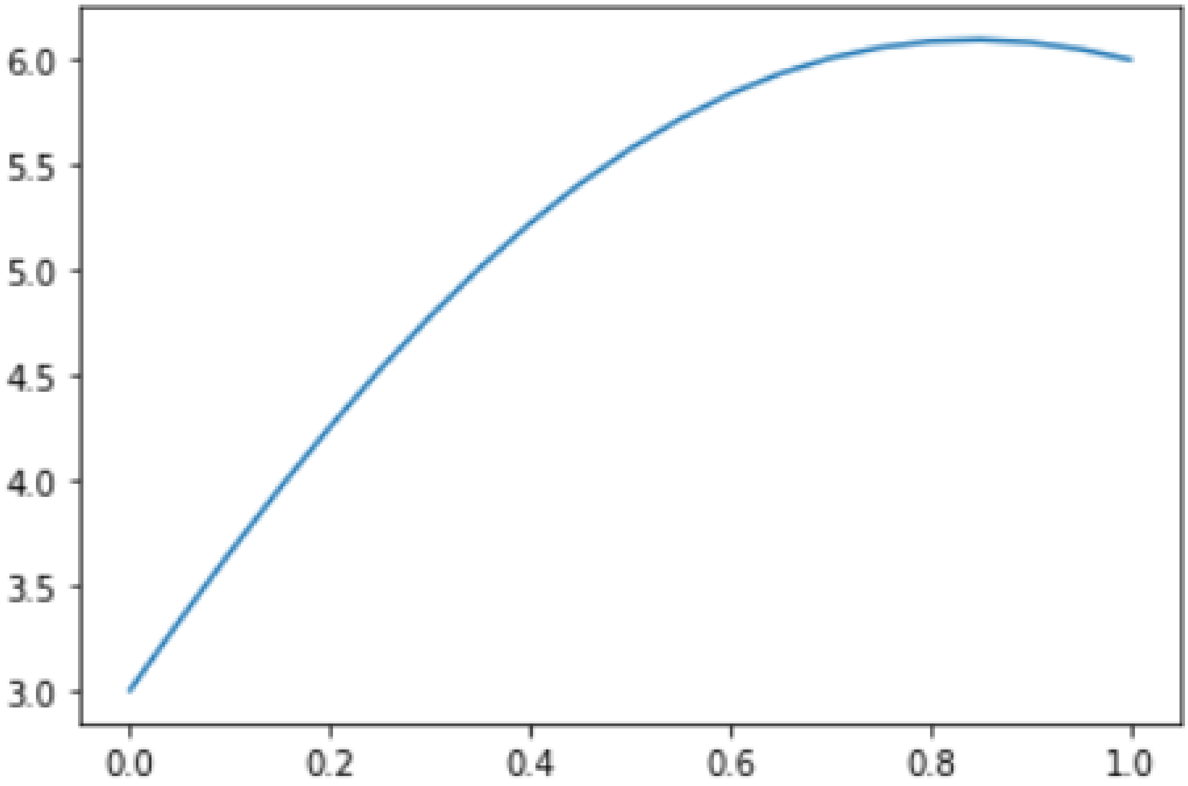

Рис. 1. $\varepsilon = 1, \quad h = 0.05$

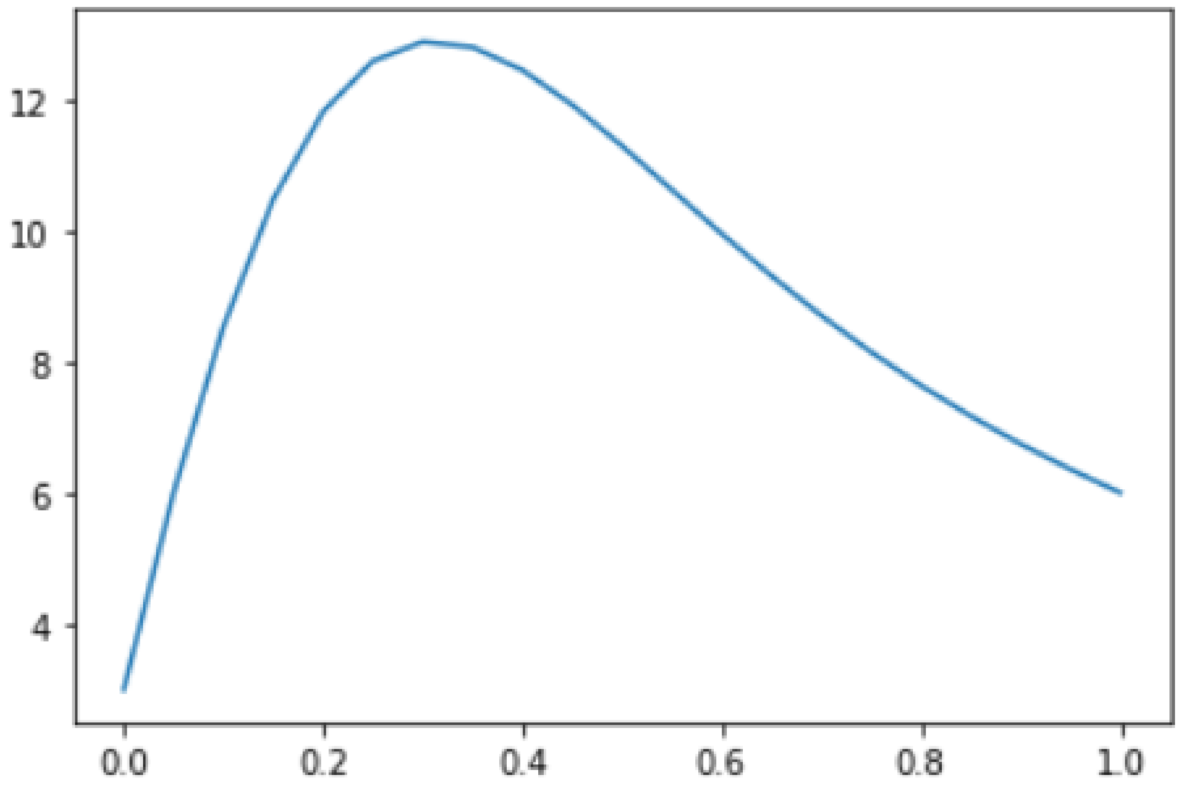

Рис. 2. $\varepsilon = 0.1, \quad h = 0.05$

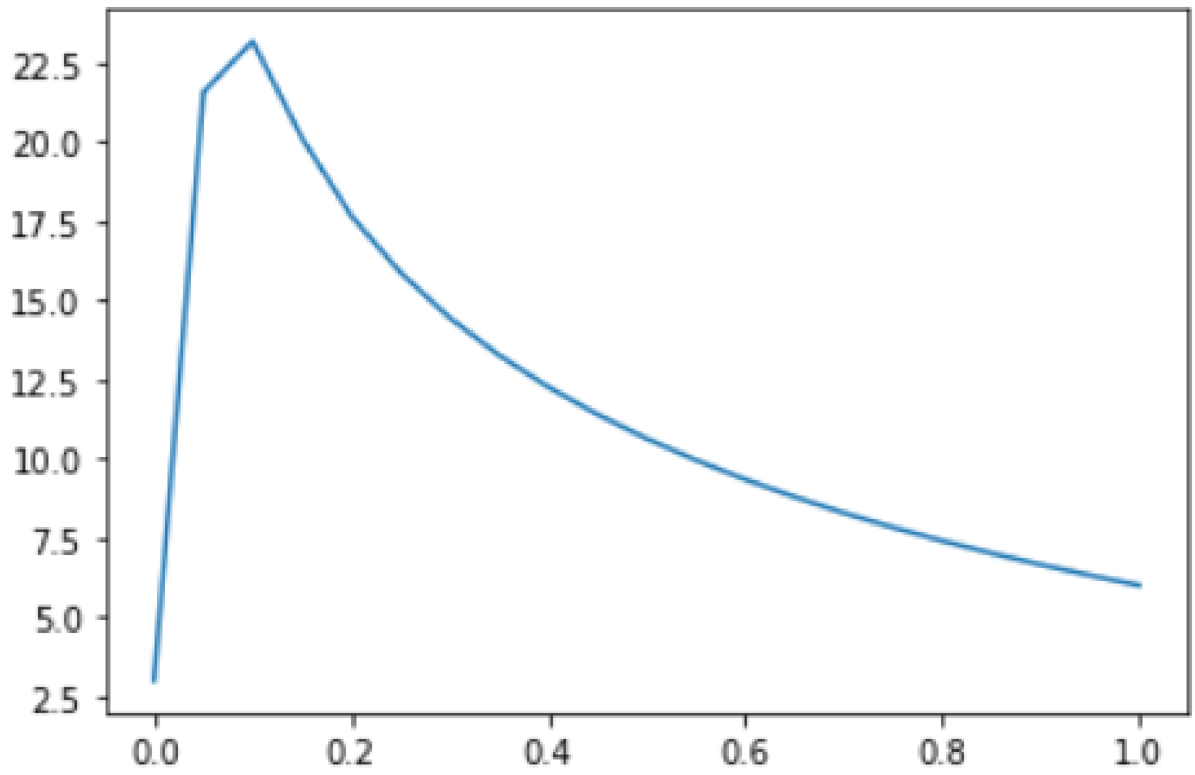

Рис. 2. $\varepsilon = 0.01, \quad h = 0.05$

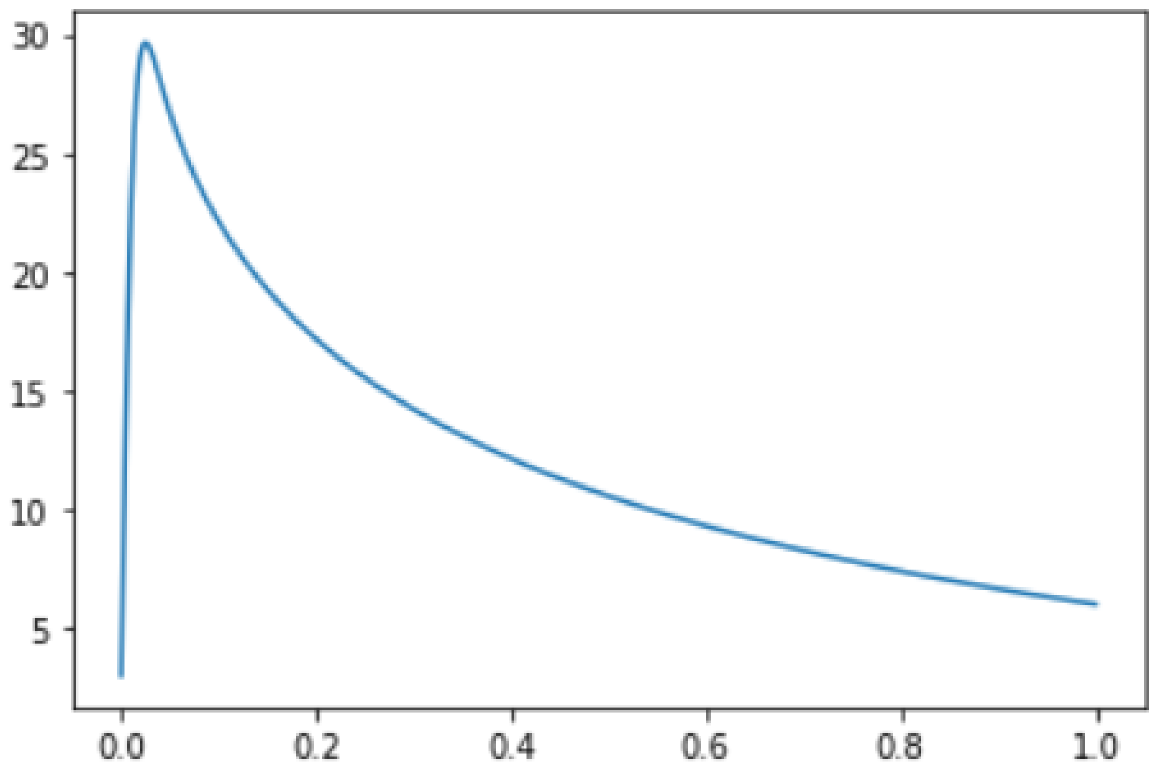

Рис. 3. $\varepsilon = 0.001, \quad h = 0.05$

Метод конечных разностей достаточно часто используют для решения краевых задач. Это связано с его относительной простотой подхода к дискретизации дифференциальных уравнений. В результате мы заменили краевую задачу на разностную схему, представляющей собой систему конечного числа линейных или нелинейных алгебраических уравнений. С помощью метода прогонки было получено решение разностной схемы. Его и принимают за приближенное решение краевой задачи.