Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
Московский государственный технический университет имени Н.Э. Баумана

Лабораторная работа №4
по курсу «Численные методы»
«Численное решение краевой задачи для линейного дифференциального уравнения второго порядка методом прогонки и стрельбы»

Выполнил:
студент группы ИУ9-62Б
Головкин Дмитрий

Проверила:
Домрачева А.Б.

Москва, 2023

Цель:

Анализ метода прогонки и стрельбы для решения дифференциальных уравнений.

Решение задачи Коши (частного решения) для дифференциального уравнения 2 порядка с помощью этогих методов.

Постановка задачи:

Дано:

1) Неоднородное дифференциальное уравнение 2 порядка с постоянными коэффициентами:

$$y''(x) + p(x)y'(x) + q(x)y(x) = f(x) \quad (1)$$

где $p(x)$ и $q(x)$ - постоянные функции, а $f(x)$ - функция, непрерывна на интервале интегрирования $[a; b]$.

2) Краевые условия:

$$y(0) = \alpha, \quad y(1) = \beta \quad (2)$$

Найти: Численное решение данного неоднородного линейного дифференциального уравнения 2 порядка отвечающее краевым условиям методом прогонки и стрельбы.

Тестовый пример:

В нашем случае, выберем следующие значения:

$$y(x) = e^x, \quad p(x) = 1, \quad q(x) = -1$$

Дифференциальное уравнение примет вид:

$$y''(x) + y'(x) - y(x) = e^x$$

Определим краевые условия:

$$y(0) = e^0 = 1, \quad y(1) = e^1 = e, \quad x \in [0; 1]$$

Описание алгоритма метода прогонки:

Приближенным численным решением задачи (1), (2) называется сеточная функция $(X_i, y_i), i = 0, ..., n$, заданная в $(n+1)-$ точке $x_i = ih,\ h = 1/n$.

Обозначим через $p_i = p(x_i),\ q_i = q(x_i),\ f_i = f(x_i)$ значения коэффициентов уравнения в точках x_i (узлах сетки), $i = 0, ..., n$. Применяя разностну. аппроксимацию производных по формулам численного дифференцирования, получим приближенную систему уравнений относительно ординат сеточной функции y_i :

$$\frac{y_{i+1} - 2y_i + 2y_{i-1}}{h^2} + p_i \frac{y_{i+1} - yi - 1}{2h} + q_i y_i = f_i$$

или после преобразований

$$y_{i-1}(1 - \frac{h}{2}p_i) + y_i(h^2 q_i - 2) + y_{i+1}(1 + \frac{h}{2}p_i) = h^2 f_i, \quad i = 1, ...n - 1, \ (3)$$

с краевыми условиями

$$y_0 = \alpha,\ y_n = \beta \quad (4)$$

Система (3) является разностной системой с краевыми условиями (4) и представляет собой трехдиагональную систему линейных алгебраических уравнений $(n+1)-$ порядка. Эту систему будем решать методом прогонки.

Описание алгоритма метода стрельбы:

Рассмотрим сеточный аналог метода стрельбы. Разобьем отрезок $[a, b]$ на n частей точками $x_0, x_1, ..., X_n$, где $x_i = a + ih,\ h = \frac{b-a}{n}$, а производные (1) во всех внутренних точках заменим разностными аналогами (аналогично подобным расскаждениям в методе прогонки):

$$y_i' = \frac{y_{i+1} - yi - 1}{2h}, \quad y_i'' = \frac{y_{i+1} - 2y_i + y_{i+1}}{h^2}, \quad i = 1, 2, ..., n-1.$$

Будем искать решения, удовлетворяющие условиям

$$y_0[0] = A, \quad y_0[1] = D_0, \quad y_1[0] = 0, \quad y_1[1] = D_1 \neq 0, \ (5)$$

Используя условия (5) и разностную запись произовдных получем следующие выражения:

$$y_0[i+1] = \frac{f_i h^2 + (2 - q_i h^2) y_0[i] - (1 - p_i \frac{h}{2}) y_0[i-1]}{1 + p_i \frac{h}{2}},$$

$$y_1[i+1] = \frac{(2 - q_i h^2) y_1[i] - (1 - p_i h^2) y_1[i-1]}{1 + p_i \frac{h}{2}},$$

$$i = 1, 2, ..., n-1;$$

$$y_0[0] = A, \quad y_0[1] = D_0, \quad y_1[0] = 0, \quad y_1[1] = D_1$$

После того, как величины $y_0[2], ..., y_0[n],\ y_1[2], ..., y_1[n]$ последовательно определены, назодим C_1 из уравнения $y_0[n] + C_1 y_1[n] = B$, т. е. $C_1 = \frac{B - y_0[n]}{y_1[n]}$. Искомое решение задачи (1) находим теперь по формулам

$$y[i] = y_0[i] + C_1 y_1[i], \quad i = 0, 1, ..., n$$

Практическая реализация:

Листинг 1. Метод прогонки для решения диф.уравнения

```
#!python
# -*- coding: utf-8 -*-

import numpy as np

def calculate_x_n(a, b, c, d, n):
    #alpha = [0.0] * n
    #beta = [0.0] * n
```

```
    alpha = np.zeros(n)
    beta = np.zeros(n)
    alpha[0] = -c[0] / b[0]
    beta[0] = d[0] / b[0]
    for i in range(1,n-1):
        #print(i)
        alpha[i] = -c[i] / (a[i-1]*alpha[i-1] + b[i])
        beta[i] = (d[i] - a[i-1]*beta[i-1]) / (a[i-1]*alpha[i-1] + b[i])
    x_n = (d[n-1] - a[n-2]*beta[n-2]) / (a[n-2]*alpha[n-2] + b[n-1])
    #print(alpha, beta)
    return x_n, alpha, beta

def calculate_x_vector(x_n, alpha, beta, n):
    x = [0.0] * (n-1)
    x.append(x_n)
    for i in range(n-2, -1, -1):
        x[i] = alpha[i]*x[i+1] + beta[i]
    return x

def func(x):
    return np.exp(x)

n = 10
left_b = 0
right_b = 1
h = (right_b - left_b) / n

x = np.arange(left_b, right_b + h/2, h)
y = func(x)

a_coefs = np.array([0.0] +  np.ones(n-2) - (h/2)*np.ones(n-2))
b_coefs = -1 * h**2 * np.ones(n-1) - 2*np.ones(n-1)
c_coefs = np.ones(n-2) + (h/2)*np.ones(n-2)
d_coefs = y[1:-1] * h**2

y0 = func(left_b)
yn = func(right_b)

d_coefs[0] -= a_coefs[1] * y0
```

```
d_coefs[-1] -= c_coefs[0] * yn

print('a_coefs: ',a_coefs)
print('b_coefs: ',b_coefs)
print('c_coefs: ',c_coefs)
print('d_coefs: ',d_coefs)

x_n, alpha, beta = calculate_x_n(a_coefs, b_coefs, c_coefs, d_coefs, n-1)
x = calculate_x_vector(x_n, alpha, beta, n-1)

final_solution = np.concatenate(([y0], x, [yn]))

delta = np.abs(y - final_solution)

for i in range(len(delta)):
    print("y=", y[i], " | apr_sol=", final_solution[i], " | delta=", delta[i])
```

Листинг 2. Метод стрельбы для решения диф.уравнения

```
#!python
# -*- coding: utf-8 -*-

import numpy as np
```

Результаты:

Для тестирования полученной программы было выбрано уравнение:

$$y''(x) + y'(x) - y(x) = e^x$$

с начальными условиями задачи Коши:

$$y(0) = e^0 = 1, \quad y(1) = e^1 = e, \quad x \in [0; 1]$$

Чтобы проверить точность метода, вычислим погрешность:

$$\varepsilon = |y_n^* - y_n|$$

где y_n^* - численное решение данного уравнения, y_n - полученное решение посредством метода Рунге-Кутты 4 порядка.

Ниже приведена таблица результата полученной программы для вычисления погрешности (Листинг 1) на указанном методе:

Значение x_n	Численное решение y_n^*	Полученное решение y_n	Погрешность ε
0	1	1.0	0.0
0.1	1.1051707977298888	1.1053447081380232	0.00017379006237550065
0.2	1.2214025180676706	1.2217091167030314	0.0003063585428615401
0.3	1.349858446117391	1.3502589327843548	0.00040012520835164267
0.4	1.4918242110670952	1.4922813162899364	0.0004566186486660850 4
0.5	1.6487206531791918	1.6491978157848073	0.00047654508467909196
0.6	1.8221180440212896	1.8225786399609751	0.0004598395704660252
0.7	2.0137518022579552	2.014158408885617	0.0004057014151404026
0.8	2.2255398622919795	2.225853543782731	0.00031261529026327395
0.9	2.459601869586057	2.4597814703303595	0.00017835917340969232
1.0	2.7182803947778686	2.718281828459045	0.0

Ниже приведена таблица результата полученной программы для вычисления погрешности (Листинг 2) на указанном методе:

Значение x_n	Численное решение y_n^*	Полученное решение y_n	Погрешность ε
0	1	1.0	0.0
0.1	1.1051707977298888	1.1053447081380232	0.00017379006237550065
0.2	1.2214025180676706	1.2217091167030314	0.0003063585428615401
0.3	1.349858446117391	1.3502589327843548	0.00040012520835164267
0.4	1.4918242110670952	1.4922813162899364	0.0004566186486660850 4
0.5	1.6487206531791918	1.6491978157848073	0.00047654508467909196
0.6	1.8221180440212896	1.8225786399609751	0.0004598395704660252
0.7	2.0137518022579552	2.014158408885617	0.0004057014151404026
0.8	2.2255398622919795	2.225853543782731	0.00031261529026327395
0.9	2.459601869586057	2.4597814703303595	0.00017835917340969232
1.0	2.7182803947778686	2.718281828459045	0.0

Выводы:

В ходе выполнения лабораторной работы были рассмотрены 2 метода решений дифференциальных уравнений, а именно методы прогонки и стрельбы. Была написана реализация данных методов на языке программирования Python.