Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Лабораторна робота №8

Чисельні методи

Розв’язування крайової задачі для ЗДР

Варіант 17

Виконала
студентка групи ІТ-91:

Луцай К. А.

Перевірила:

ас. Тимофєєва Ю. С.

Київ 2021

Мета: навчитися розв'язувати крайову задачу для ЗДР за допомогою мови програмування Python, використати кінцево-різницевий метод.

Теоретичні відомості:

Прикладом крайової задачі є двоточкова крайова задача для звичайного диференціального рівняння другого порядку.

$$y'' = f(x, y, y')$$

з граничними умовами, заданими на кінцях відрізка [a,b].

$$\begin{matrix} y(a) = y_0 & & y'(a) = \tilde{y}_0 & & \alpha y(a) + \beta y'(a) = \hat{y}_0 \\ y(b) = y_1 & \text{або} & y'(b) = \tilde{y}_1 & \text{або} & \delta y(b) + \gamma y'(b) = \hat{y}_1 \end{matrix}$$

Кінцево-різницевий метод розв'язування крайової задачі

Розглянемо двоточкову крайову задачу для лінійного диференціального рівняння другого порядку на відрізку [a,b]

$$y'' + p(x)y' + q(x)y = f(x)$$

$$y(a) = y_0, y(b) = y_1$$

Введемо різницеву сітку на відрізку [a,b] з кроком h. Введемо різницеву апроксимацію похідних у такий спосіб:

$$y_k' = \frac{y_{k+1} - y_{k-1}}{2h} + O(h^2);$$

$$y_k'' = \frac{y_{k+1} - 2y_k + y_{k-1}}{h^2} + O(h^2);$$

Підставляючи апроксимації похідних отримаємо систему рівнянь для знаходження yk

$$\begin{cases} y_0 = y_a \\ \dfrac{y_{k+1} - 2y_k + y_{k-1}}{h^2} + p(x_k)\dfrac{y_{k+1} - y_{k-1}}{2h} + q(x_k)y_k = f(x_k), k = 1, N-1 \\ y_N = y_b \end{cases}$$

Приводячи подібні та з огляду на те, що коли задаються граничні умови першого роду, два невідомих y0, yN вже фактично визначено, отримаємо систему лінійних алгебраїчних рівнянь із трьохдіагональною матрицею коефіцієнтів

$$\begin{cases} (-2+h^2q(x_1))y_1+(1+\frac{p(x_1)h}{2})y_2 = h^2f(x_1)-(1-\frac{p(x_1)h}{2})y_a \\ (1-\frac{p(x_k)h}{2})y_{k-1}+(-2+h^2q(x_k))y_k+(1+\frac{p(x_k)h}{2})y_{k+1} = h^2f(x_k) \qquad ,k=2,\dots,N\text{-}2 \\ (1-\frac{p(x_{N-1})h}{2})y_{N-1}+(-2+h^2q(x_{N-1}))y_{N-1} = h^2f(x_{N-1})-(1+\frac{p(x_{N-1})h}{2})y_b \end{cases}$$

У разі, коли використовуються граничні умови другого та третього роду, апроксимація похідних проводиться за допомогою односторонніх різниць першого та другого порядків.

$$y_0' = \frac{y_1-y_0}{h}+O(h); \qquad y_0' = \frac{-3y_0+4y_1-y_2}{2h}+O(h^2);$$

$$y_N' = \frac{y_N-y_{N-1}}{h}+O(h)$$ або $$y_N' = \frac{y_{N-2}-4y_{N-1}+3y_N}{2h}+O(h^2);$$

Розв’язок задачі в аналітичній формі:

$$y''-2^x y'-x^2y+4=0$$
$$y(0.5)=1$$
$$y(1.5)+y'(1.5)=0$$ з кроком 0.2

Тут p(x) = 2^x , q(x) = x^2, f (x) = -4, N = 5 ,
X = [0.5, 0.7, 0.9, 1.1, 1.3, 1.5]

$(25-5*2^{x_k+1})y_{k+1}-(50+x^2)y_k+(25+5*2^{x_k+1})y_{k-1}=-4$ k = 1...4

$5(y_5-y_4)+y_5=0$

Система рівнянь:

$-2.1y_1+0.8586y_2=-1.3014$

$1.1625y_1-2.0196y_2+0.8375y_3=-0.16$

$1.1866y_2-2.0324y_3+0.8134y_4=-0.16$

$1.2144y_3+2.0484y_4+0.7856y_5=-0.16$

$-5y_4+6y_5=0$

Y = [1, 1.241, 1.389, 1.437, 1.367, 1.4]

Лістинги програм розв’язування задачі:

Початкові задані за умовою значення:

```
import math
import numpy as np

h = 0.2
a = 0.5
ya = 1
b = 1.5
yb = 0
```

В якості аргументів методи приймають крок, грані, граничні значення. Метод для побудови матриці приймає додаткові параметри функцій від x та повертає матрицю і стовбець СЛАР.

```
def makeMatrix(X, h, p, q, f, yA, yB):
    N = X.shape[0] - 1
    Y = np.zeros([N, N])
    B = np.zeros(N)
    for i in range(N):
        if i == 0:
            Y[i, i] = (-2 + (h**2) * q(X[i]))
            Y[i, i+1] = (1 + h * p(X[i]) / 2)
            B[i] = (h**2) * f - (1 - h * p(X[i]) / 2) * yA
        elif i == N-1:
            Y[i, i] = 1/h + 1
            Y[i, i-1] = -1/h
            B[i] = yB
        else:
            Y[i, i] = (-2 + (h**2) * q(X[i]))
            Y[i, i-1] = (1 - h * p(X[i]) / 2)
            Y[i, i+1] = (1 + h * p(X[i]) / 2)
            B[i] = (h**2) * f
    return Y, B

def kinzriz(h, a, b, yA, yB):
    def fP(x):
        return -(2**x)

    def fQ(x):
        return -(x**2)

    X = np.linspace(a, b, int((b-a)/h)+1)
    A, B = makeMatrix(X, h, fP, fQ, -4, yA, yB)
    Y = np.linalg.solve(A, B)

    print("Кінцево-різницевий метод:")
    for row in range(len(B)):
        for col in range(len(A[row])):
            if A[row][col] > 0:
                print(" + ", end='')
```

```
            if A[row][col] < 0:
                print(" - ", end='')
            if A[row][col] == 0:
                print(" "*11, end='')
                continue
            print(str(round(math.fabs(A[row][col]), 4)).rjust(6), end='')
            print("y", end='')
            print(col+1, end='')
        print(" =", round(B[row], 4))

    print("i X   Y")
    for i in range(X.shape[0]):
        if i == 0:
            print(i, X[i], yA)
        else:
            print(i, X[i], Y[i-1])
```

Виклик методів та графічне представлення:

```
print("y`` - 2^x * y` - x^2 * y + 4 = 0")
print("y(0.5) = 1")
print("y(1.5) + y`(1.5) = 0")
kinzriz(h, a, b, ya, yb)
```

Результати виконання програми:

```
y`` - 2^x * y` - x^2 * y + 4 = 0
y(0.5) = 1
y(1.5) + y`(1.5) = 0
Кінцево-різницевий метод:
 -   2.01y1 + 0.8586y2                                 = -1.3014
 + 1.1625y1 - 2.0196y2 + 0.8375y3                      = -0.16
            + 1.1866y2 - 2.0324y3 + 0.8134y4           = -0.16
                       + 1.2144y3 - 2.0484y4 + 0.7856y5 = -0.16
                                  -    5.0y4 +    6.0y5 = 0.0
i X   Y
0 0.5 1
1 0.7 1.2409814676313005
2 0.9 1.3894491813510956
3 1.1 1.4369921241131147
4 1.3 1.3668825246552094
5 1.5 1.1390687705460079
```

Висновки: було написано програму для розв’язання крайової задачі для ЗДР на мові Python, було реалізовано кінцево-різницевий метод.