МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра Математического и компьютерного моделирования

ОТЧЕТ ПО ЗАДАЧЕ №1

РЕАЛИЗАЦИЯ ОДНОГО ИЗ МЕТОДОВ ВЗВЕШЕННЫХ НЕВЯЗОК

студента 4 курса 411 группы

направление 01.03.02 — Прикладная математика и информатика

механико-математического факультета

Яфарова Шамиля Валериевича

Саратов 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью работы является приобретение и закрепление практических навыков при выполнение численных процедур построения приближенного решения системы дифференциальных (или интегральных) уравнений вида

$$L(u_0) = p, \qquad x \in V$$

Задача 1. Метод коллокаций.

Реализовать метод коллокаций для решения дифференциального уравнения, получить приближенное и точное решение, сравнить их между собой, построить графики доказательства сходимости метода.

1 ФУНКЦИИ НА PYTHON

В ходе работы были использованы следующие функции на Python.

np.linalg.inv() - это библиотечная функция numpy, которая вычисляет обратную (мультипликативную) матрицу.

numpy.dot() - это математическая функция, которая используется для возврата математической точки двух заданных векторов(списков). Функция np.dot() принимает три аргумента и возвращает скалярное произведение двух заданных векторов в Python. Векторы могут быть как одномерными, так и многомерными. В обоих случаях это следует правилу математического скалярного произведения.

2 ХОД РЕШЕНИЯ

Рассмотрим следующее уравнение второго порядка

$$y'' + \frac{y}{x^2} = 0$$

В качестве граничных условий выберем:

$$\begin{cases} y(1) = 1; \\ y\left(exp^{\frac{\pi}{\sqrt{3}}}\right) = 2. \end{cases}$$

Уравнение имеет решение:

$$y(x) = C_1 \frac{\cos\left(\frac{\sqrt{3}}{2}\ln(x)\right)}{\sqrt{x}} + C_2 \frac{\sin\left(\frac{\sqrt{3}}{2}\ln(x)\right)}{\sqrt{x}}$$

Тогда

$$\begin{cases} C_1 = 1; \\ C_2 = 2exp^{\pi/2\sqrt{3}}. \end{cases}$$

Вычисление граничного условия

```
x_a = 1
print("x_a:")
print(x_a)
print("--------")
x_b = exp(pi/sqrt(3))
print("x_b:")
print(x_b)
```

```
x_a:
1
--------
x_b:
6.133707406236227
```

Операции для нахождения C_1, C_2

Подсчет C_1, C_2 при x=1, 6.1337($exp^{\frac{\pi}{\sqrt{3}}}$)

```python
def C(x,y):
    C1 = cos(sqrt(3)/2 * log(x))/sqrt(x)
    C2 = cos(sqrt(3)/2 * log(y))/sqrt(y)
    C3 = sin(sqrt(3)/2 * log(x))/sqrt(x)
    C4 = sin(sqrt(3)/2 * log(y))/sqrt(y)
    return np.matrix('{} {};{} {}'.format(C1, C3, C2, C4))

matrix = C(x_a, x_b)
```

Граничные условия в данных точках обозначаются как

```python
Y_a = 1
Y_b = 2
```

Из Y образовался вектор

```python
def Y(x,y):
    Y1 = x
    Y2 = y
    return np.matrix('{};{}'.format(Y1, Y2))

Y = Y(Y_a, Y_b)
```

Обратная матрица

```python
inverse_array = np.linalg.inv(matrix)
```

```
inverse_array:
[[ 1.00000000e+00  0.00000000e+00]
 [-6.12323400e-17  2.47663227e+00]]
```

При умножение вектора на матрицу получаем C(C_1, C_2)

```python
C = np.dot(inverse_array, Y)
```

```
C:
[[1.        ]
 [4.95326454]]
```

```
C1 = 1
print("C1:")
print(C1)
print("--------")
#Проверяем наш C2
C2 = 2 * exp(pi/(2*sqrt(3)))
print("C2:")
print(C2)
print("--------")
```

```
C1:
1
--------
C2:
4.953264542192847
```

C_2 из np.dot и вычисления `C2 = 2 * exp(pi/(2*sqrt(3)))` одинаковое

Методом коллокаций будем приблежать функцию y(x) с помощью суммы некоторых известных функций:

$$y(x) = \varphi_0(x) + \sum_{i=1}^{n} \varphi_i(x)$$

$\varphi_0(x)$ на границах принимает требуемые значения

$\varphi_i(x)$ на границах будут принимать 0

$$\begin{cases} \varphi_0(1) = 1; \\ \varphi_0\left(exp^{\frac{\pi}{\sqrt{3}}}\right) = 2. \end{cases}$$

$$\begin{cases} \varphi_i(1) = 0; \\ \varphi_i\left(exp^{\frac{\pi}{\sqrt{3}}}\right) = 0. \end{cases}$$

$\varphi_0(x)$ по линейному закону

$\varphi_0(x) = kx + b$

$$\begin{cases} kx + b = 1; \\ k\left(exp^{\frac{\pi}{\sqrt{3}}}\right) + b = 2. \end{cases}$$

Разность между 2 уравнением и 1

$$k\left(exp^{\frac{\pi}{\sqrt{3}}} - 1\right) = 1$$

$$k = \frac{1}{exp^{\frac{\pi}{\sqrt{3}}} - 1}$$

b = 1 – k

или

$$b = 1 - k = 1 - \frac{1}{exp^{\frac{\pi}{\sqrt{3}}} - 1} = \frac{exp^{\frac{\pi}{\sqrt{3}}} - 2}{exp^{\frac{\pi}{\sqrt{3}}} - 1}$$

```
k = 1/(exp(pi/sqrt(3))-1)
print("k:")
print(k)
print("--------")
b = 1 - k
print("b:")
print(b)
print("--------")
b_pr = (exp(pi/sqrt(3))-2)/(exp(pi/sqrt(3))-1)
print("b_проверка:")
print(b_pr)
print("--------")
```

```
k:
0.19479100012307657
--------
b:
0.8052089998769234
--------
b_проверка:
0.8052089998769234
```

$$\varphi_{\mathbf{0}}(x) = \frac{x}{exp^{\frac{\pi}{\sqrt{3}}} - 1} + \frac{exp^{\frac{\pi}{\sqrt{3}}} - 2}{exp^{\frac{\pi}{\sqrt{3}}} - 1} = \frac{exp^{\frac{\pi}{\sqrt{3}}} - 2 + x}{exp^{\frac{\pi}{\sqrt{3}}} - 1}$$

$$\varphi_{\boldsymbol{i}}(\mathrm{x}) = \sin\big(i(\omega x + \beta)\big)$$

$$\begin{cases} \omega + \beta = \pi; \\ \omega\left(exp^{\frac{\pi}{\sqrt{3}}}\right) + \beta = 2\pi. \end{cases}$$

$$\omega\left(exp^{\frac{\pi}{\sqrt{3}}} - 1\right) = \pi$$

$$\omega = \frac{\pi}{exp^{\frac{\pi}{\sqrt{3}}} - 1}$$

$\beta = 2\pi - \omega$

или

$$\beta = \pi - \omega = \pi - \frac{\pi}{exp^{\frac{\pi}{\sqrt{3}}} - 1} = \frac{exp^{\frac{\pi}{\sqrt{3}}} - 2}{exp^{\frac{\pi}{\sqrt{3}}} - 1}\pi$$

```
W = pi/(exp(pi/sqrt(3))-1)
print("W:")
print(W)
print("--------")
B = pi - W
print("B:")
print(B)
print("--------")
B_pr = (exp(pi/sqrt(3))-2)/(exp(pi/sqrt(3))-1)*pi
print("B_проверка:")
print(B_pr)
```

```
W:
0.6119539749720658
--------
B:
2.529638678617727
--------
B_проверка:
2.529638678617727
```

$$\varphi_{\boldsymbol{i}}(\mathrm{x}) = \sin\left(i\pi\left(\frac{x}{exp^{\frac{\pi}{\sqrt{3}}}-1}+\frac{exp^{\frac{\pi}{\sqrt{3}}}-2}{exp^{\frac{\pi}{\sqrt{3}}}-1}\right)\right) = \sin\left(i\pi\left(\frac{exp^{\frac{\pi}{\sqrt{3}}}-2+x}{exp^{\frac{\pi}{\sqrt{3}}}-1}\right)\right)$$

$$y(x) = \varphi_{\mathbf{0}}(\mathrm{x}) + \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \alpha_i \varphi_{\boldsymbol{i}}(\mathrm{x}) =$$

$$= \mathrm{kx} + \mathrm{b} + \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \alpha_i \sin\big(i(\omega x + \beta)\big) =$$

$$y'(x) = \mathrm{k} + \sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{i}\omega\alpha_i \cos\big(i(\omega x + \beta)\big)$$

$$y''(x) = \sum_{\mathrm{i=1}}^{\mathrm{n}} -\mathrm{i}^2\omega^2\alpha_i \sin\big(i(\omega x + \beta)\big)$$

$$y'' + \frac{y}{x^2} = 0 = \psi(\alpha_1, \alpha_2, \dots \alpha_n; x)$$

Функция невязки

$$\psi(\alpha_1, \alpha_2, \dots \alpha_n; x)$$
$$= \sum_{\mathrm{i=1}}^{\mathrm{n}} -\mathrm{i}^2\omega^2\alpha_i \sin\big(i(\omega x + \beta)\big) + \frac{k}{x} + \frac{b}{x^2} + \sum_{\mathrm{i=1}}^{\mathrm{n}} \alpha_i \frac{\sin\big(i(\omega x + \beta)\big)}{x^2}$$

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} \alpha_i \left[\frac{\sin\big(i(\omega x + \beta)\big)}{x^2} - -\mathrm{i}^2\omega^2 \sin\big(i(\omega x + \beta)\big)\right] = -\left(\frac{k}{x} + \frac{b}{x^2}\right)$$

Получим левую часть и преобразуем в обратную матрицу

```
print("--------")
inverse_matrix_A = np.linalg.inv(A_matrix)
print("inverse_matrix_A:")
print(inverse_matrix_A)
```

```
inverse_matrix_A:
[[ 0.67527694  1.10485819  1.21684712  1.02000002  0.57752972]
 [-0.2471716  -0.25681512 -0.04710439  0.16634125  0.1818039 ]
 [ 0.11592199  0.0141742  -0.09097488  0.0059953   0.10301224]
 [-0.0541319   0.04490786 -0.00272073 -0.05071905  0.04731414]
 [ 0.01958136 -0.03018811  0.03667562 -0.0303836   0.0181192 ]]
--------
```

Получим правую часть и преобразуем в вектор

```
def K(x, y, z):
    k = -(x/y + z/(y*y))
    return k
print("--------")
K_matrix = np.matrix('{};{};{};{};{}'.format(K(k, X_1, b), K(k, X_2, b), K(k, X_3, b), K(k, X_4, b), K(k, X_5, b)))
print("K_matrix:")
print(K_matrix)
print("--------")
```

```
K_matrix:
[[-0.3388205 ]
 [-0.18138616]
 [-0.11790186]
 [-0.08521551]
 [-0.0658094 ]]
```

φ

```
vals = np.arange(x_a, x_b, 0.1)
```

```
for i in vals:
    fi_0 = k*i+b
    fi_1 = (A[0]*sin(1*(W*i+B)))[0, 0]
    fi_2 = (A[1]*sin(2*(W*i+B)))[0, 0]
    fi_3 = (A[2]*sin(3*(W*i+B)))[0, 0]
    fi_4 = (A[3]*sin(4*(W*i+B)))[0, 0]
    kol = fi_0 + fi_1 + fi_2 + fi_3 + fi_4
```

Точное решение

```
toch = C1 * cos(sqrt(3) / 2 * log(i))/sqrt(i) + C2 * sin(sqrt(3) / 2 * log(i))/sqrt(i)
```

Погрешность вычисления

```
eps = tochnoe[-1] - kollokacii[-1]
```

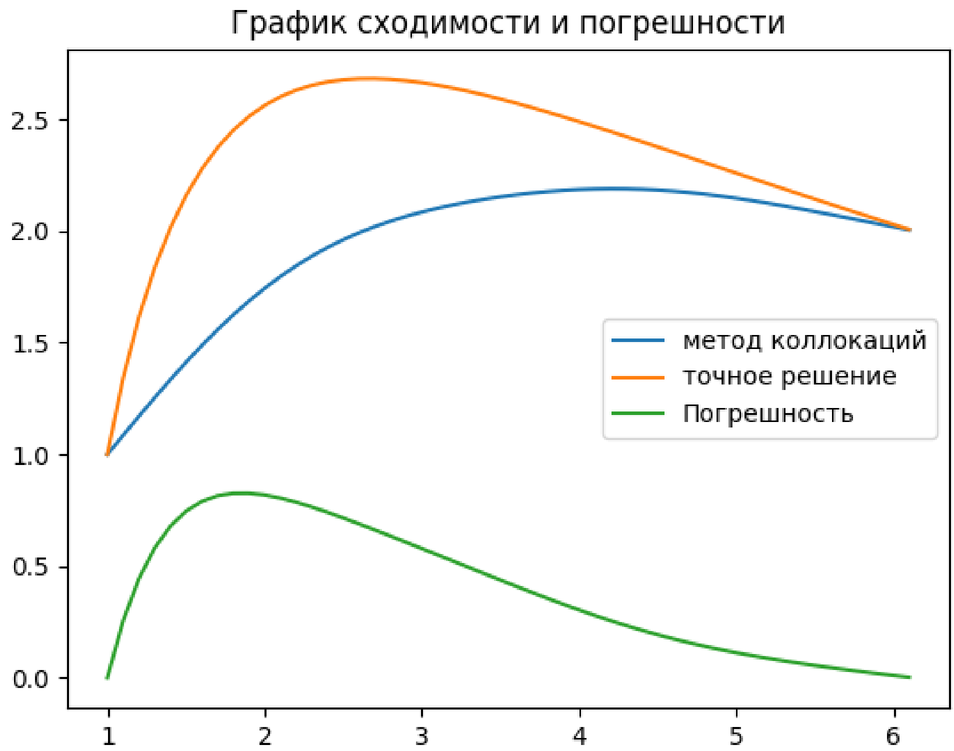
График сходимости и погрешности
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
метод коллокаций
точное решение
Погрешность

ЗАКЛЮЧЕНИЕ

В данной работе продемонстрирована реализация метода коллокаций для решения дифференциального уравнения. На основе полученных данных были вычислены решения

Была проведена оценка погрешности и построен график

A Исходный код

https://github.com/Yafarovsham/Solution-of-DE-by-the-collocation-method

.