|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **ИУК «Информатика и управление»** |
| **КАФЕДРА** | **ИУК4 «Программное обеспечение ЭВМ,** |
| **информационные технологии»** | |

**Лабораторная работа №2**

**«Численное решение стационарных задач теплопроводности»**

**ДИСЦИПЛИНА: «Моделирование»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-62Б | |  |  | ( | Калашников А.С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Никитенко У.В. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

Калуга, 2023

**Цель работы:** сформировать практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для стандартных вычислений и визуализации результатов применения метода конечных разностей.

**Постановка задачи**

Найти приближенное решение краевой задачи методом конечных разностей c заданной точностью tol и построить его график для:

**ПОРЯДОК РЕШЕНИЯ**

1. Используя встроенные функции/библиотеки PYTHON/MATLAB etc, получить “точное” решение задачи в узлах основной сетки, обозначим его Yex.

2. Cоставить разностную схему второго порядка точности.

3. Для реализации алгоритма метода прогонки следует создать модуль с процедурой, параметрами которой должны являться порядок системы, массивы коэффициентов системы уравнений и коэффициенты правой части.

4. Для вычисления решения задачи с заданной точностью произвести расчет с начальным шагом h, затем уменьшить шаг вдвое. Вывести на экран в виде таблицы два соседних приближенных решения и сравнить результаты.

5. Если заданная точность не достигнута, то продолжить уменьшение шага. Построить график найденного решения и указать шаг, при котором заданная точность достигается.

**Результаты выполнения работы**

Представим дифференциальное уравнение в виде :

При помощи формул центральной разности. Тогда первая и вторая производные в точках заданной сетки примут вид:

Подставим их в исходное уравнение:

Помножим обе части на :

или

Выпишем граничные условия для точки :

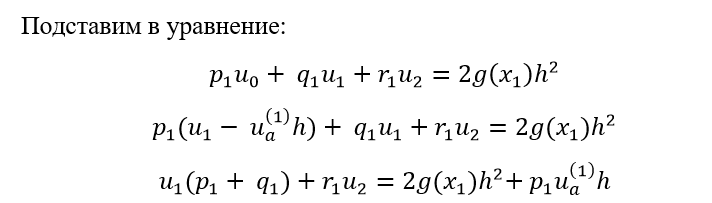
Подставим приближённое значение:

Так как значение не представлено в сетке, выразим его через существующие узлы:

Подставим полученное значение в уравнение

или

По левой границе решим с помощью метода Неймана



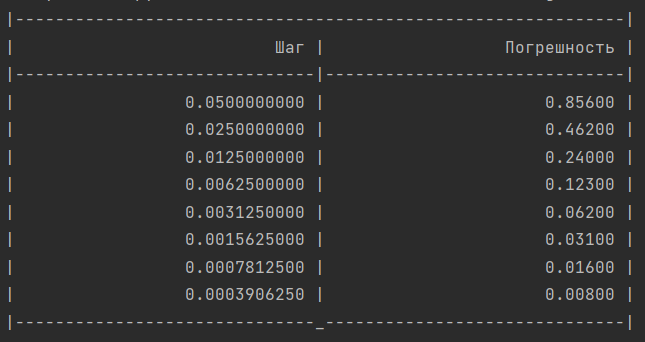
Составим систему из полученных уравнений:

…

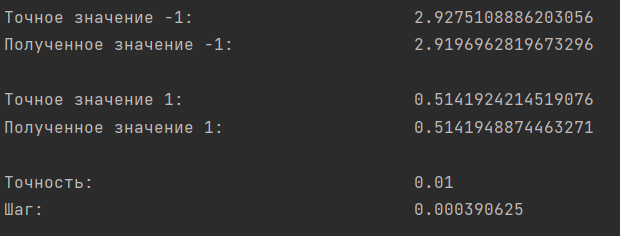
Таким образом, разностная схема примет вид:

Для «точного» решения воспользуемся методом solve\_bvp библиотеки scipy. Уменьшая каждый раз шаг в 2 раза, начиная с шага , получим следующее решение (см. рис. 1).

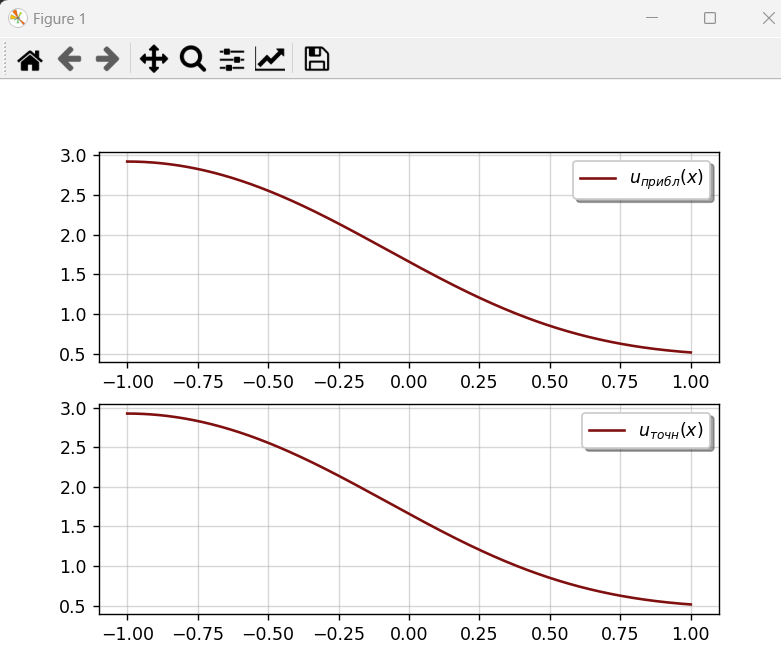
Для полученного решения построим график, а также график решения, полученного встроенным методом (см. рис 3).Значение, удовлетворяющее точности, было получено при шаге .

**

**Рисунок 1** *–* Вывод таблицы

**

**Рисунок 2** *–* Вывод значений

**

**Рисунок 3** *–* Вывод графика

**Вывод:** в ходе выполнения работы были сформированы практические навыки анализа моделей с использованием специализированных программных пакетов и библиотек для стандартных вычислений и визуализации результатов применения метода конечных разностей.

**ПРИЛОЖЕНИЯ**

**Листинг программы**

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.integrate import solve\_bvp, solve\_ivp  
import scipy.linalg as la  
  
def g(x):  
 return -(1)/(x+3)  
  
def task(a2, a1, a0, g, robin\_A, robin\_B, rng, h):  
 n = int((rng[1]-rng[0])/h) + 1  
  
 pf = lambda x: 1 - a1(x)/a2(x) \* h / 2  
 qf = lambda x: -(2 - a0(x)/a2(x) \* h\*\*2)  
 rf = lambda x: 1 + a1(x)/a2(x) \* h / 2  
 df = lambda x: g(x)/a2(x) \* h\*\*2  
  
 xs = np.linspace(rng[0], rng[1], n)  
 A = np.zeros((n, n))  
 B = np.zeros(n)  
  
 A[n-1, n-2] = robin\_A[1, 1] \* (pf(xs[n-1]) + rf(xs[n-1]))  
 A[n-1, n-1] = robin\_A[1, 1] \* qf(xs[n-1]) - 2 \* h \* robin\_A[1, 0]  
 B[n-1] = df(xs[n-1]) \* robin\_A[1, 1] - 2 \* h \* robin\_B[1] \* pf(xs[n-1])  
  
 #xn = xs[n-1]  
 #A[n-1, n-1] = robin\_A[1, 1] \* qf(xn) - 2 \* h \* robin\_A[1, 0]  
 #A[n-1, n-2] = robin\_A[1, 1] \* (pf(xn) + rf(xn))  
 #B[n-1] = df(xn) \* robin\_A[1, 1] - 2 \* h \* robin\_B[1] \* pf(xn)  
  
 #pf = lambda x: 2\*a2(x) - a1(x)\*h  
 #qf = lambda x: -4\*a2(x) + 2\*a0(x)\*h\*\*2  
 #rf = lambda x: 2\*a2(x) + a1(x)\*h  
 #df = lambda x: 2\*g(x) \* h\*\*2  
  
 A[0, 1] = qf(xs[0]) + rf(xs[0])  
 A[0, 0] = pf(xs[0])  
 B[0] = df(xs[0]) - rf(xs[0])\*0\*h  
  
 for i in range(1, n - 1):  
 A[i, i - 1] = pf(xs[i])  
 A[i, i] = qf(xs[i])  
 A[i, i + 1] = rf(xs[i])  
 B[i] = df(xs[i])  
  
 u = np.linalg.solve(A, B)  
   
 return xs, u  
  
def exact\_task(rng):  
 a = -1  
 b = 1  
  
 def fun(x, y):  
 return np.vstack((y[1], (x \* y[1] - np.log(2+x) \* y[0] - (x/2) + 1)/g(x)))  
  
 def bc(ya, yb):  
 return np.array([ya[1], 1\*yb[1] + 0.5\*yb[0]])  
  
  
 res = solve\_bvp(fun, bc, [a, b], [[-1, -1], [1, 1]], tol=1e-9)  
  
 return res.x, res.y[0], res  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 h = 0.1  
 rng = [-1, 1]  
  
 tol = 1e-2  
 cnt = 3  
 err = tol + 1  
  
 x\_ex, y\_ex, res = exact\_task(rng)  
  
 x, y = task(a2 = lambda x: -1/(x+3),  
 a1 = lambda x: -1 \* x,  
 a0 = lambda x: np.log(2+x),  
 g = lambda x: 1-(x/2),  
 robin\_A = np.array([[-1, 1], [0.5, 1]]),  
 robin\_B = np.array([0, 0]),   
 rng=rng,   
 h=h)  
  
  
 print(f"|{'-'\*30}-{'- '\*30}|")  
 print(f"|{'Шаг':>29} |{'Погрешность':>29} |")  
 print(f"|{'-'\*30}|{'-'\*30}|")  
  
 while err >= tol:  
 x, y = task(a2 = lambda x: -1/(x+3),  
 a1 = lambda x: -1 \* x,  
 a0 = lambda x: np.log(2+x),  
 g = lambda x: 1-(x/2),  
 robin\_A = np.array([[-1, 1], [0.5, 1]]),  
 robin\_B = np.array([0, 0]),  
 rng=rng,  
 h=h)  
 err = round(max(abs(np.array([res.sol(i)[0] for i in x]) - y)), cnt)  
 h /= 2  
 print(f'|{round(h, 10):29.10f} |{err:29.5f} |')  
  
 print(f"|{'-'\*30}\_{'-'\*30}|")  
  
 print(f'\n{f"Точное значение {rng[0]}:":40} {res.sol(rng[0])[0]}')  
 print(f'{f"Полученное значение {rng[0]}:":40} {y[0]}')  
  
 print(f'\n{f"Точное значение {rng[1]}:":40} {res.sol(rng[1])[0]}')  
 print(f'{f"Полученное значение {rng[1]}:":40} {y[-1]}')  
  
 print(f'\n{f"Точность:":40} {tol}')  
 print(f'{f"Шаг:":40} {h}')  
  
 plt.subplot(2, 1, 1)  
 plt.plot(x, y, color='#FF1010', label='$u\_{прибл}(x)$')  
 plt.grid(alpha=0.5)  
 plt.legend(framealpha=1, shadow=True)  
  
 plt.subplot(2, 1, 2)  
 plt.plot(res.x, res.y[0], color='#FF1010', label='$u\_{точн}(x)$')  
 plt.grid(alpha=0.5)  
 plt.legend(framealpha=1, shadow=True)  
 plt.show()