Санкт-Петербургский государственный политехнический

университет им. Петра Великого

Кафедра “Прикладная математика”

Отчет по лабораторной работе №8

Методы оптимизации

**Работу выполнили студент группы № 5030102/10401**

Дубовицкий Владислав Александрович

**Преподаватель**

Козлов Константин Николаевич

Санкт-Петербург, 2024

Условие:

Требуется запрограммировать Модифицированный метод Эйлера. Программа должна работать для произвольной размерности системы уравнений.

Функция правой части системы и начальное условие подаются на вход программе. Вычисления должны производиться с пошаговым контролем точности по правилу Рунге. Если на текущем шаге точность не достигается, то шаг уменьшается в 2 раза, если достигнутая погрешность меньше заданной в 64 раза, то шаг увеличивается в 2 раза.

**Входные данные**

На вход подаются несколько строк, в которых:

* начало промежутка t\_0
* конец промежутка T
* начальный шаг h\_0
* максимальное число вызовов функции правой части N\_x
* желаемая точность eps
* число уравнений
* следующие n+3 строк определяют функцию правой части на Python
* последняя строка содержит n чисел - начальное условие

**Выходные данные**

Программа печатает в консоль следующие столбцы, одна строчка соответствует одному шагу интегрирования:

1. значение t
2. значение шага h
3. оценка Рунге R
4. истраченое число вычислений правой части N
5. значения функций решений

Алгоритм:

while t < T and kounter[0] < N\_x:  
 v\_First = euler\_Modf(t,v,h)  
 v\_Second = euler\_Modf(t,v,h/2)  
 v\_Second = euler\_Modf(t + h/2, v\_Second, h/2)  
  
 R = np.linalg.norm(v\_First - v\_Second) / (pow(2,2) - 1)  
  
 if R > eps:  
 h /= 2  
  
 elif R < (eps / 64):  
 h \*= 2  
  
 else:  
 v = v\_First  
 t += h  
  
 if t + h > T:  
 h = T - t

Результат:

Изображение выглядит как текст, снимок экрана, черный, черно-белый

Автоматически созданное описание

Изображение выглядит как текст, меню, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание

Изображение выглядит как текст, меню, снимок экрана, книга

Автоматически созданное описание

Изображение выглядит как текст, меню, снимок экрана

Автоматически созданное описание

…Изображение выглядит как текст, меню, снимок экрана, черно-белый

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание

…Изображение выглядит как текст, снимок экрана, меню, книга

Автоматически созданное описание

Изображение выглядит как текст, меню, снимок экрана

Автоматически созданное описание

…Изображение выглядит как текст, снимок экрана, меню, книга

Автоматически созданное описание

Графики:  
  
Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание

Изображение выглядит как текст, диаграмма, линия, График

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Код:

import numpy as np  
import matplotlib.pyplot as plt  
  
def modified\_euler\_method(rhs\_func, initial\_condition, t0, T, h0, N\_x, eps):  
 t = t0  
 h = h0  
 v = np.array(initial\_condition)  
 kounter = [0]  
 results = []  
  
 print("{:12.6f} {:12.6f} {:12s} {:12d} {:12.6f} {:12.6f} {:12.6f}".format(  
 t, h, "0", kounter[0], \*v))  
  
 def euler\_Modf(t,v,h):  
 v\_hat = rhs\_func(t, v, kounter)  
 v\_tilde = rhs\_func(t + h, v + h \* v\_hat, kounter)  
 return v + (h / 2) \* (v\_hat + v\_tilde)  
  
  
  
 while t < T and kounter[0] < N\_x:  
 v\_First = euler\_Modf(t,v,h)  
 v\_Second = euler\_Modf(t,v,h/2)  
 v\_Second = euler\_Modf(t + h/2, v\_Second, h/2)  
  
 R = np.linalg.norm(v\_First - v\_Second) / (pow(2,2) - 1)  
  
 if R > eps:  
 h /= 2  
  
 elif R < (eps / 64):  
 h \*= 2  
  
 else:  
 v = v\_First  
 t += h  
  
 print("{:12.6f} {:12.6f} {:12.5e} {:12d} {:12.6f} {:12.6f} {:12.6f}".format(  
 t, h, R, kounter[0], \*v))  
  
 if t + h > T:  
 h = T - t  
  
 results.append((t, h, R, kounter[0], \*v))  
  
 return results  
  
t0 = 1.5  
T = 2.5  
h0 = 0.1  
N\_x = 10000  
eps = 0.0001  
  
eps\_count = [0.001, 0.0001, 0.00001, 0.000001, 0.0000001, 0.00000001]  
  
kounter\_mas = []  
def fs(t, v, kounter):  
 A = np.array([[-0.4, 0.02, 0], [0, 0.8, -0.1], [0.003, 0, 1]])  
 kounter[0] += 1  
 return np.dot(A, v)  
  
initial\_condition = [1, 1, 2]  
  
result\_list = []  
for eps in eps\_count:  
 print("Eps = ", eps)  
 results = modified\_euler\_method(fs, initial\_condition, t0, T, h0, N\_x, eps)  
 result\_list.append(results)  
 print(" ")  
  
for i, results in enumerate(result\_list):  
 t\_values = [r[0] for r in results]  
 h\_values = [r[1] for r in results]  
 plt.plot(t\_values, h\_values, label = f'eps={eps\_count[i]}')  
plt.xlabel('t')  
plt.ylabel('h')  
plt.title('Изменение шага по отрезку для разных точностей')  
plt.legend()  
plt.show()  
  
min\_h\_values = [min([r[1] for r in results]) for results in result\_list]  
plt.plot(eps\_count, min\_h\_values)  
plt.xlabel('eps')  
plt.ylabel('мигимальное h')  
plt.title('Зависимость минимального шага от заданной точности')  
plt.show()  
  
num\_steps = [len(results) for results in result\_list]  
plt.loglog(eps\_count, num\_steps)  
plt.xlabel('eps')  
plt.ylabel('число шагов')  
plt.title('Зависимость числа шагов от заданной точности')  
plt.show()  
  
for i, results in enumerate(result\_list):  
 t\_values = [r[0] for r in results]  
 v\_values = [r[3:] for r in results]  
 plt.loglog(t\_values, v\_values, label = f'eps = {eps\_count[i]}')  
plt.xlabel('t')  
plt.ylabel('v')  
plt.title('Решение для разных значений заданной точности')  
plt.legend()  
plt.show()