ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В.ЛОМОНОСОВА»

Физический факультет

КАФЕДРА ФИЗИЧЕСКОЙ ЭЛЕКТРОНИКИ

Основы математического моделирования

«**Отчет по практическому заданию №1, Вариант 1**»

Выполнила студентка

323 группы

Мелкозерова Юлия Алексеевна

Москва

2018

ОГЛАВЛЕНИЕ

[ПОСТАНОВКА ЗАДАЧИ 3](#_Toc10638539)

[ХАРАКТЕРИСТИКИ УРАВНЕНИЯ 3](#_Toc10638540)

[МЕТОД РЕШЕНИЯ 4](#_Toc10638541)

[УСТОЙЧИВОСТЬ СХЕМЫ 5](#_Toc10638542)

[РЕАЛИЗАЦИЯ ПРОГРАММЫ 6](#_Toc10638543)

## ПОСТАНОВКА ЗАДАЧИ

Используя схему бегущего счета и итерационные методы, решить задачу:

# ХАРАКТЕРИСТИКИ УРАВНЕНИЯ

Для того, чтобы определить не претерпевает ли решение разрыв составим уравнения характеристик и посмотрим, будут ли они пересекаться. Для данной задачи уравнение характеристик будет иметь вид:

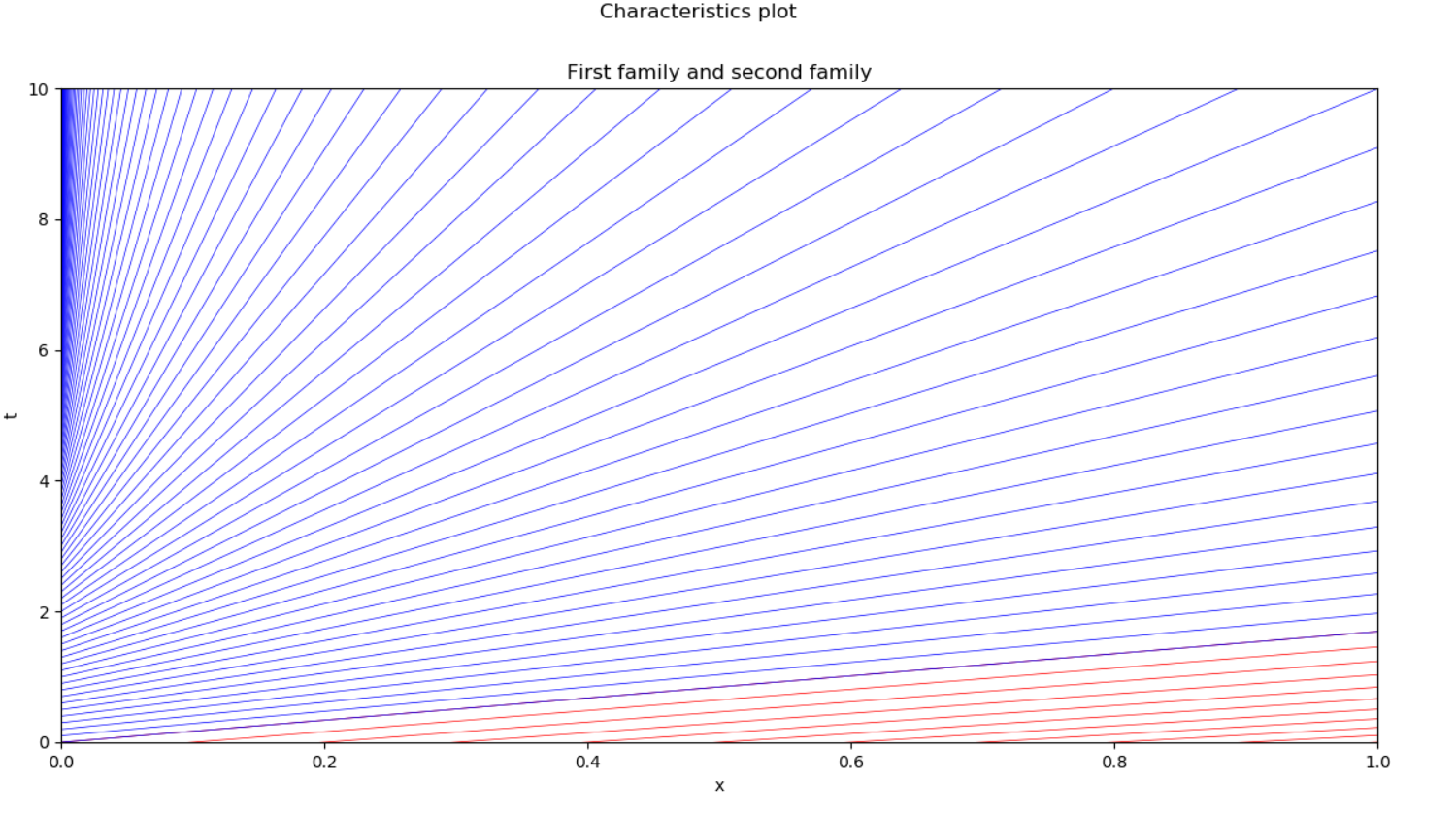
Отсюда получаем систему уравнений:

Подставляя начальные и граничные условия, получаем два семейства характеристик:

1 Семейство ()

2 Семейство ()

Как видно на рисунке, они не пересекаются в заданной области.



# МЕТОД РЕШЕНИЯ

Введем разностную сетку:

Введем сеточную функцию

Перепишем уравнение в виде и опозначим

Выберем для построения разностной схемы четырёxточечный шаблон, так как при таком выборе разностная схема сходится со вторым порядком аппроксимации и безусловно устойчива. Тогда конечно-разностная аппроксимация данного уравнения в точке выглядит следующим образом:

Разностные аппроксимации начального и конечного условия принимают вид:

Полученная схема аппроксимирует исходную дифференциальную задачу. Уравнение для сеточной функции в узле имеет вид:

Данное уравнение решаем с помощью итерационного метода Ньютона.

Пусть известно некоторое приближение к корню . Тогда уравнение примет вид:

Раскладывая данное уравнение в ряд Тейлора вблизи точки , ограничившись членами первого порядка, получим:

Условие остановки итераций (достижение заданной точности) можно задать как

.

# УСТОЙЧИВОСТЬ СХЕМЫ

Проверим выполнение необходимого условия устойчивости – критерия Неймана. Зафиксируем коэффициент перед . Рассмотрим точку , тогда разностная схема будет иметь вид:

Будем искать решение в виде

Обозначим и

= 1

# РЕАЛИЗАЦИЯ ПРОГРАММЫ

Бегущий счет

import math  
import pylab  
from mpl\_toolkits.mplot3d import Axes3D  
import numpy as np  
from matplotlib.colors import LinearSegmentedColormap  
  
  
def F(U, m, n):  
 return U[m][n]\*\*2 / 2  
  
#def C(U, m, n):  
# return -(2\*U[m][n] + tn[n])  
  
def df(U, m1, n1):  
 return 1/(2\*ht) + (U[m1][n1])/(2\*hx)  
  
def func(U,m2,n2):  
 return (U[m2][n2] + U[m2-1][n2] - U[m2-1][n2-1] - U[m2][n2-1]) / (2\*ht) + (F(U,m2,n2) + F(U,m2,n2-1) - F(U,m2-1,n2-1) - F(U,m2-1,n2))/(2\*hx)  
  
x1, x2 = 0, 1  
t1, t2 = 0, 0.3  
Nx, Nt = 100, 100  
hx = abs(x2-x1)/Nx  
ht = abs(t2-t1)/Nt  
eps = 1e-5  
xn = np.arange(x1, x2, hx)  
tn = np.arange(t1, t2, ht)  
U = [[0 for x in xn] for t in tn]  
  
for t in range(Nt):  
 U[0][t]= (2 - 4/math.pi \* math.atan(2))\*math.exp(-tn[t])  
for x in range(Nx):  
 U[x][0] = 4/math.pi \* math.atan(xn[x] - 2) + 2  
  
  
for t in range(Nt)[1:Nt]:  
 for x in range(Nx)[1:Nx]:  
 x0 = U[x-1][t-1]  
 while abs(func(U,x,t))>eps:  
 U[x][t] += - func(U, x, t)/df(U, x, t)  
  
def makeData ():  
 xgrid, ygrid = np.meshgrid(xn, tn)  
 zgrid = np.array(U)  
 return xgrid, ygrid, zgrid  
  
x, y, z = makeData()  
  
fig = pylab.figure()  
axes = Axes3D(fig)  
  
axes.plot\_surface(x, y, z, rstride=3, cstride=3, cmap = LinearSegmentedColormap.from\_list ("flower", ['#d71868', 'w', '#1cac78'], 256))  
  
pylab.show()

Характеристики

import numpy as np  
import math  
import matplotlib.pyplot as plt  
  
x1 = np.linspace(-100, 100, 10000)  
t1 = np.linspace(-100, 100, 10000)  
  
x2 = np.linspace(-100, 100, 10000)  
t2 = np.linspace(-100, 100, 10000)  
  
c = [0]\*100  
d = [-1]\*100  
  
f, axs = plt.subplots(2, 2, figsize=(14, 7))  
f.suptitle('Characteristics plot')  
plt.subplot(1, 1, 1)  
plt.xlim(0, 1)  
plt.ylim(0, 10)  
plt.title('First family and second family')  
plt.xlabel('x')  
plt.ylabel('t')  
  
for x0 in range(0, 100):  
 t1 = (x1 - 0.1\*x0) / (4/math.pi \* math.atan(0.1\*x0 - 2) + 2)  
 plt.plot(x1, t1, color = 'red', linewidth = 0.5)  
  
#plt.subplot(1, 2, 2)  
#plt.title('Second family')  
#plt.xlim(0, 2)  
#plt.ylim(0, 10)  
  
for t0 in range(0, 100):  
 x2 = (2 - 4/math.pi \* math.atan(2)) \* math.exp(-0.1\*t0) \* (t2 - 0.1\*t0)  
 plt.plot(x2, t2, color = 'blue', linewidth = 0.5)  
  
  
  
#plt.subplot(1, 1, 1)  
plt.show()