ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8

Уравнения гиперболического типа

(Вариант 9)

Выполнил студент 3 курса ПМиИ
Кондратьев Виталий

Цель работы: усвоить сущность и методы решения ***линейного дифференциального уравнения 2-го порядка гиперболического типа***.

Численное решение дифференциального уравнения в частных производных предполагает получение двумерной числовой таблицы приближенных значений U_{ij} искомой функции $U(t,x)$ с заданной точностью для некоторых значений аргументов

$x_j \in [a, b]$, $t_i \in [c, d]$

Численное решение таких дифференциальных уравнений возможно методами конечных разностей.

Погрешность решения, найденного этими методами, оценивается величиной $O(\tau^p, h^q)$, где p, q - порядок метода.

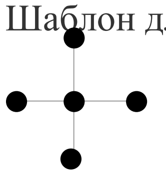
Задание.

Решить волновое уравнение

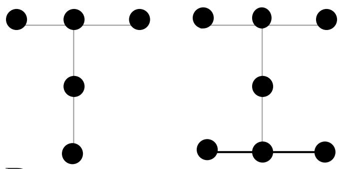
$$\frac{\partial^2 U}{\partial t^2} = D^2 \frac{\partial^2 U}{\partial x^2} + f(t,x)$$

явным методом и неявными методами второго порядка точности

Шаблон для явного метода:



Шаблон для неявного метода:



Вывести результаты в виде графиков U(x) для разных значений t от 1 до 10 с шагом 1

Неявные методы решать с помощью прогонки.

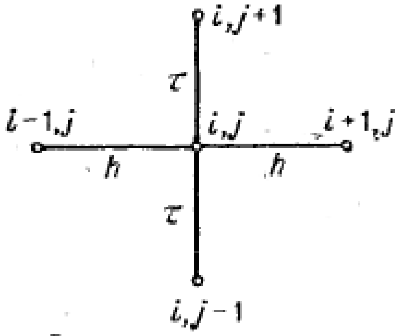
Варианты задания (лабораторная № 8)

Для всех вариантов $[a, b] = [0; 1]$, $[c, d] = [0; 10]$, $f(x,t)=0$

Погрешность решения 0,01.

№ вар.	Начальные условия	Граничные условия	D
9	$U(0,x) = x^2, \quad \frac{\partial U}{\partial t}(0,x) = 1$	$U(t,0) = 0, \quad U(t,1) = 1$	1

1. ЯВНЫЙ МЕТОД



- *схема «крест»*

$$\frac{\partial^2 U}{\partial t^2} = \frac{\partial^2 U}{\partial x^2}$$

Разностное уравнение, построенное по данной схеме:

$$\frac{u_i^{j+1} - 2u_i^j + u_i^{j-1}}{\tau^2} = a^2 \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{h^2}$$
$$i = \overline{1, I-1}, \qquad j = \overline{1, J-1}$$

Разрешая уравнение относительно u_i^{j+1}:

$$u_i^{j+1} = 2(1-\lambda)u_i^j + \lambda\left(u_{i+1}^j + u_{i-1}^j\right) - u_i^{j-1}, \qquad \lambda = \frac{a^2\tau^2}{h^2} \text{или } \frac{D\tau^2}{h^2}$$

На нулевом слое имеем:

$$u_i^0 = \varphi_i, \qquad i = \overline{0, I}$$

На первом слое имеем:

$$u_i^1 = u_i^0 + \tau\psi_i + \frac{a^2\tau^2}{2}\varphi_i'', \qquad i = \overline{0, I}$$

Граничные условия в сеточном виде:

$$u_0^j = \theta_0^j, \qquad j = \overline{1, J}$$
$$u_I^j = \theta_I^j, \qquad j = \overline{1, J}$$

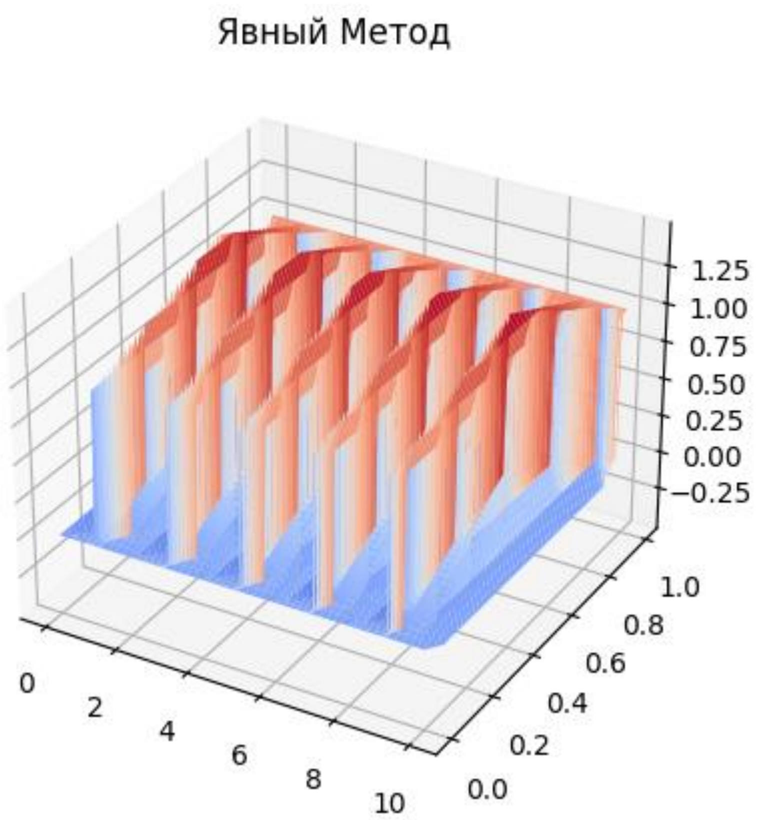
Рассмотренная разностная схема условно устойчива. Необходимое и достаточное условие устойчивости:

$$\frac{a\tau}{h} < 1 \text{ или } \sqrt{D} * \frac{\tau}{h} < 1$$

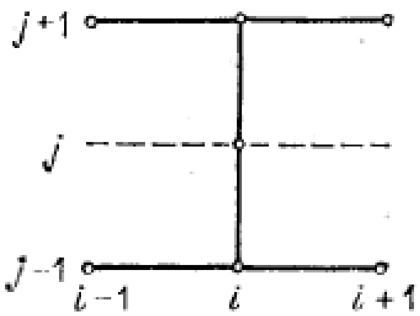
Возьмем в качестве первого шага разбиения: $h = 0.1$ и $\tau = 0.01$. Итерации завершаются при выполнении условия:

$$\max_i \left| U_i^{(k)} - U_i^{(k+1)} \right| < 0{,}01$$

График:



2. НЕЯВНЫЙ МЕТОД



- неявная схема

Разностное уравнение, построенное по данной схеме:

$$\frac{u_i^{j+1}-2u_i^j+u_i^{j-1}}{\tau^2}=\frac{a^2}{2}\left(\frac{u_{i+1}^{j+1}-2u_i^{j+1}+u_{i-1}^{j+1}}{h^2}+\frac{u_{i+1}^{j-1}-2u_i^{j-1}+u_{i-1}^{j-1}}{h^2}\right)$$

Из этого разностного соотношения можно получить систему уравнений относительно неизвестных значений сеточной функции на $j+1$слое:

$$-\lambda u_{i-1}^{j+1}+(1+2\lambda)u_i^{j+1}-\lambda u_{i+1}^{j+1}=-(1+2\lambda)u_i^{j-1}+\lambda\left(u_{i+1}^{j-1}+u_{i-1}^{j-1}\right)+2u_i^j$$

$$\lambda=\frac{a^2\tau^2}{h^2} \text{ или } \frac{D\tau^2}{h^2}, \qquad i=\overline{1,I-1}, \qquad j=\overline{1,J-1}$$

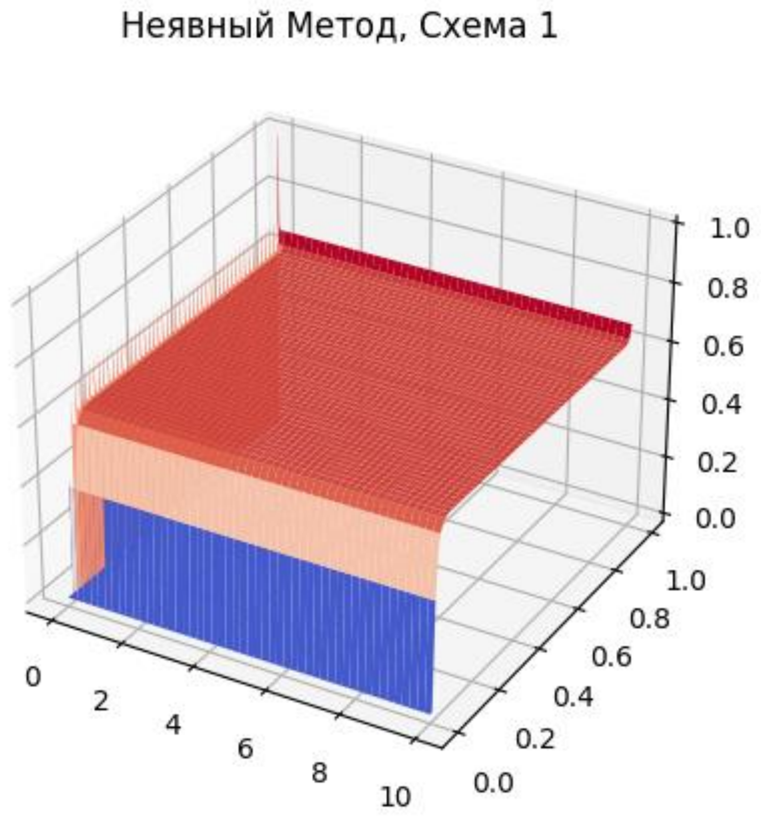
Полученная схема устойчива и сходится со скоростью $O(\tau^2 + h^2)$. Граничные и начальные условия аналогичны явной схеме.

Возьмем в качестве первого шага разбиения: $h = 0.1$ и $\tau = 0.01$. Итерации завершаются при выполнении условия:

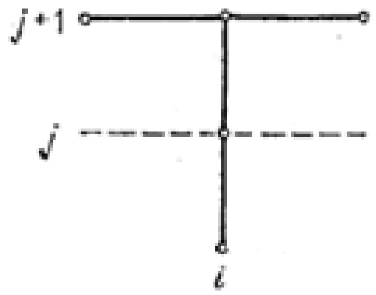
$$\max_i \left| U_i^{(k)} - U_i^{(k+1)} \right| < 0{,}01$$

Систему линейных уравнений решаем методом прогонки.

График:



3. НЕЯВНЫЙ МЕТОД



\- неявная разностная схема

Разностное уравнение, построенное по данной схеме:

$$\frac{u_i^{j+1} - 2u_i^j + u_i^{j-1}}{\tau^2} = a^2 \frac{u_{i-1}^{j+1} - 2u_i^{j+1} + u_{i+1}^{j+1}}{h^2}$$

$$i = \overline{1, I-1}, \qquad j = \overline{1, J-1}$$

Из этого разностного соотношения можно получить систему уравнений относительно неизвестных значений сеточной функции на $j+1$слое:

$$-\lambda u_{i-1}^{j+1} + (1+2\lambda)u_i^{j+1} - \lambda u_{i+1}^{j+1} = 2u_i^j - u_i^{j-1}$$

$$\lambda = \frac{a^2\tau^2}{h^2} \text{или} \frac{D\tau^2}{h^2}, \qquad i = \overline{1, I-1}, \qquad j = \overline{1, J-1}$$

Полученная схема устойчива и сходится со скоростью $O(\tau^2 + h^2)$. Граничные и начальные условия аналогичны явной схеме.

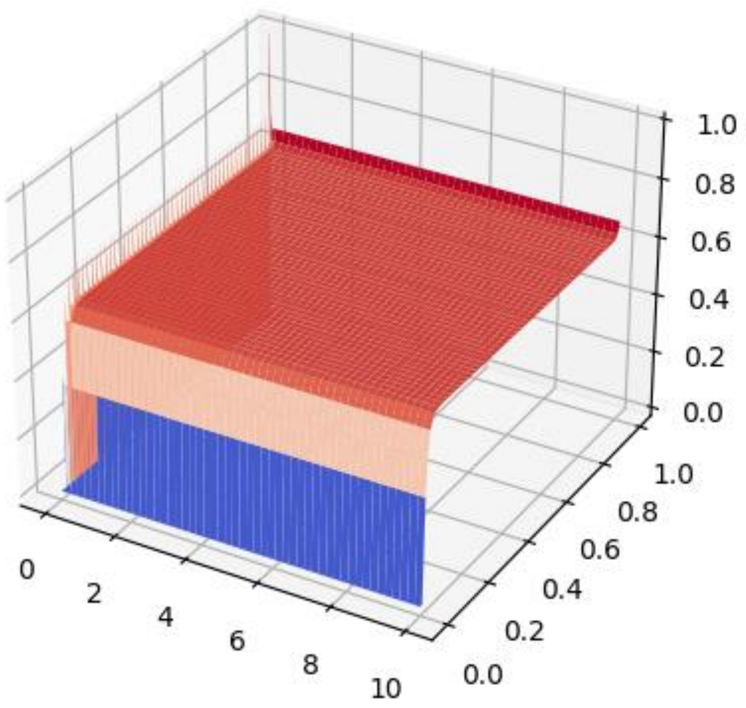
Возьмем в качестве первого шага разбиения: $h = 0.1$ и $\tau = 0.01$. Итерации завершаются при выполнении условия:

$$\max_i \left| U_i^{(k)} - U_i^{(k+1)} \right| < 0{,}01$$

Систему линейных уравнений решаем методом прогонки.

График:

Неявный Метод, Схема 2



ПРИЛОЖЕНИЕ

```
"""
Лабораторная работа №8
Студент ОНК «ИВТ» ВШ КНиИИ направления ПМиИ 3 курса
Кондратьев Виталий
Вариант 9
"""
import numpy as np
from matplotlib import cm
import matplotlib.pyplot as plt
a,b = [0,1]
c,d = [0,10]
eps, h, tau = 0.01, 0.01, 0.01
D = 1
l_ambda = (D * tau / h) ** 2
f = lambda x: 0
U_0 = lambda x,t: x ** 2
der_U_t = lambda x: 1
U_t_0 = lambda x,t: 0
U_t_1 = lambda x,t: 1
def Scheme1(a, b, c, d, tau, l_ambda):
    x_count = int((b - a) / tau)
    t_count = int((d - c) / tau)
    u = np.zeros((t_count, x_count))
    for i in range(0, x_count):
        u[0][i] = U_0(0, i * h)
        u[1][i] = u[0][i] + (-1) * tau
    for i in range(0, t_count):
        u[i][0] = U_t_0(i * tau, 0)
        u[i][x_count - 1] = U_t_1(i * tau, 1)
    for j in range(1, t_count - 1):
      for i in range(1, x_count - 1):
         u[j + 1][i] = 2 * (1 - l_ambda) * u[j][i] + l_ambda * (u[j][i + 1] +
u[j][i - 1]) - u[j - 1][i]
    return u

def Scheme2(a, b, c, d, tau, l_ambda):
    x_count = int((b - a) / tau)
    t_count = int((d - c) / tau)
    A = l_ambda
    B = l_ambda * 2 + 1
    u = np.zeros((t_count, x_count))
    a = b = c = np.zeros(x_count)
    for i in range(0, x_count):
        u[0][i] = U_0(0, i * h)
        u[1][i] = u[0][i] + (-1) * tau
    for i in range(0, t_count):
        u[i][0] = U_t_0(i * tau, 0)
        u[i][x_count - 1] = U_t_1(i * tau, 1)
    for j in range(1, t_count - 1):
        a[x_count - 1] = 0
        b[x_count - 1] = u[j + 1][x_count - 1]
        c[x_count - 1] = 1 / (B - A * a[x_count - 1])
        for i in range(x_count - 1, 0, -1):
            a[i - 1] = c[i] * A
            b[i - 1] = c[i] * (A * b[i] - (u[j - 1][i] - 2 * u[j][i]))
            c[i - 1] = 1 / (B - A * a[i - 1])
        for i in range(1, x_count - 1):
            u[j + 1][i + 1] = a[i] * u[j + 1][i] + b[i]
    return u

def Scheme3(a, b, c, d, tau, l_ambda):
    x_count = int((b - a) / tau)
```

```
    t_count = int((d - c) / tau)
    A = l_ambda
    B = l_ambda * 2 + 1
    u = np.zeros((t_count, x_count))
    a = b = c = np.zeros(x_count)
    for i in range(0, x_count):
        u[0][i] = U_0(0, i * h)
        u[1][i] = u[0][i] + (-1) * tau
    for i in range(0, t_count):
        u[i][0] = U_t_0(i * tau, 0)
        u[i][x_count - 1] = U_t_1(i * tau, 1)
    for j in range(1, t_count - 1):
        a[x_count - 1] = 0
        b[x_count - 1] = u[j + 1][x_count - 1]
        c[x_count - 1] = 1 / (B - A * a[x_count - 1])
        a[x_count - 2] = c[x_count - 1] * A
        b[x_count - 2] = (c[x_count - 1] * (A * b[x_count - 1]
        - (B * u[j - 1][x_count - 1] - A * (u[j - 1]
        [x_count - 1]
        + u[j - 1][x_count - 2]) - 2 * u[j][x_count - 1])))
        c[x_count - 2] = 1 / (B - A * a[x_count - 2])
        for i in range(x_count - 2, 0, -1):
            a[i - 1] = c[i] * A
            b[i - 1] = (c[i] * (A * b[i] - (B * u[j - 1][i]
            - A * (u[j - 1][i + 1] + u[j - 1][i - 1]) - 2 * u[j][i])))
            c[i - 1] = 1 / (B - A * a[i - 1])
        for i in range(1, x_count - 1):
            u[j + 1][i + 1] = a[i] * u[j + 1][i] + b[i]
    return u

def draw(a, b, c, d, h, tau, u, method):
    x = np.arange(a, b, h)
    t = np.arange(c, d, tau)
    x, t = np.meshgrid(x, t)
    fig, ax = plt.subplots(subplot_kw={"projection": "3d"})
    ax.plot_surface(t, x, u, cmap=cm.coolwarm)
    ax.set_title(method.title())
    plt.show()

u1 = Scheme1(a, b, c, d, tau, l_ambda)
u2 = Scheme2(a, b, c, d, tau, l_ambda)
u3 = Scheme3(a, b, c, d, tau, l_ambda)
draw(a, b, c, d, h, tau, u1, "Явный метод")
draw(a, b, c, d, h, tau, u2, "Неявный метод, схема 1")
draw(a, b, c, d, h, tau, u3, "Неявный метод, схема 2")
```