ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9

Параболические уравнения

(Вариант 9)

Выполнил студент 3 курса ПМиИ
Кондратьев Виталий

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

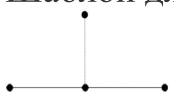
Задание.

Решить параболическое уравнение

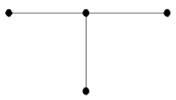
$$\frac{\partial U}{\partial t} = D\frac{\partial^2 U}{\partial x^2}$$

явным методом и неявным методом.

Шаблон для явного метода:



Шаблон для неявного метода:



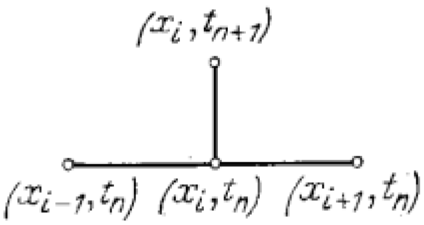
Вывести результаты в виде графиков U(x) для разных значений t от 1 до 10 с шагом 1

Для всех вариантов $[a, b]$ = [0; 1], $[c, d]$ = [0; 10], D=1. Погрешность решения 0,01.

Материал стр. 243 и 278 Турчак ЛИ «Основы численных методов»

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

1. *ЯВНАЯ СХЕМА*



Уравнение теплопроводности:

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

Введём схему, состоящую из четырёх узлов. Разностное уравнение, построенное по данной схеме:

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

$$i = \overline{1, I-1}, \qquad j = \overline{0, J-1}$$

Разрешая уравнение относительно u_i^{j+1}:

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

Начальные условия:

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

Граничные условия в сеточном виде:

$$u_0^j = {\psi_1}_0^j, \qquad j = \overline{1, J}$$

$$u_I^j = {\psi_2}_I^j, \qquad j = \overline{1, J}$$

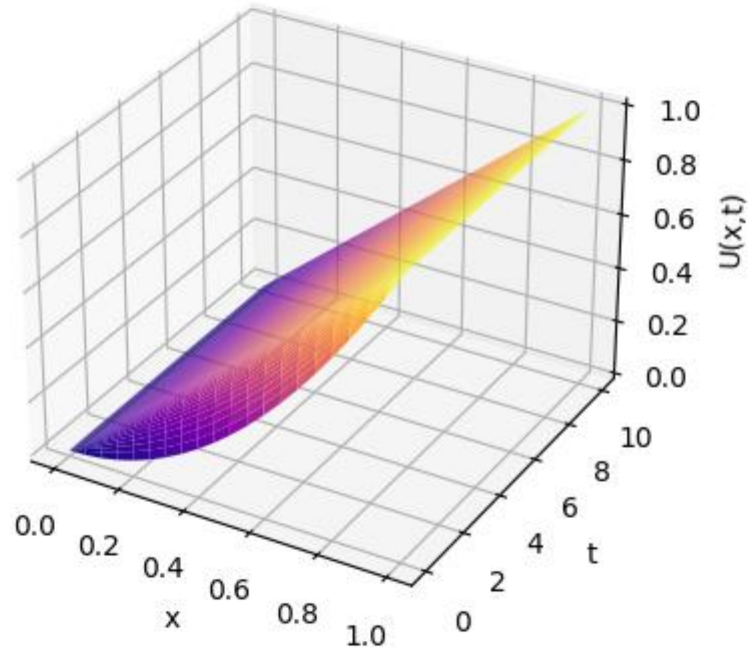
Рассмотренная разностная схема условно устойчива. Необходимое и достаточное условие устойчивости:

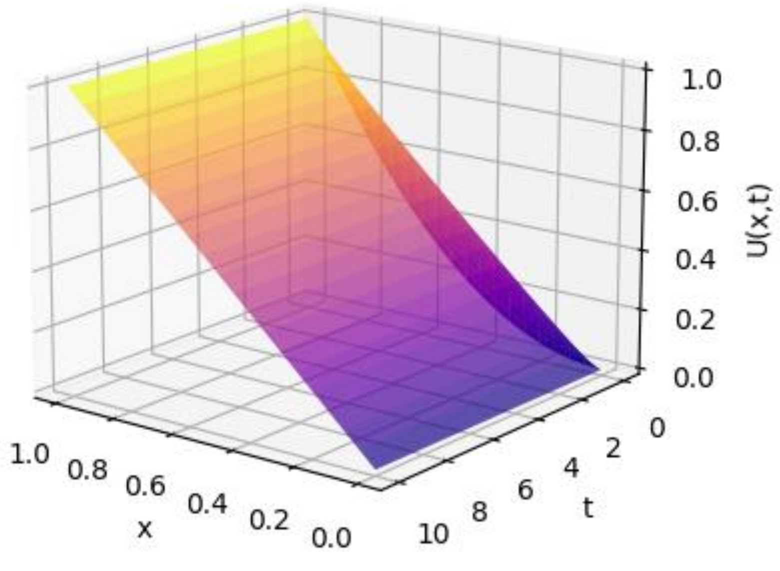
$$\frac{a\tau}{h^2} \leq \frac{1}{2}$$

Возьмем в качестве первого шага разбиения: $h = 0.1$ и $\tau = 0.005$. Итерации завершаются при выполнении условия:

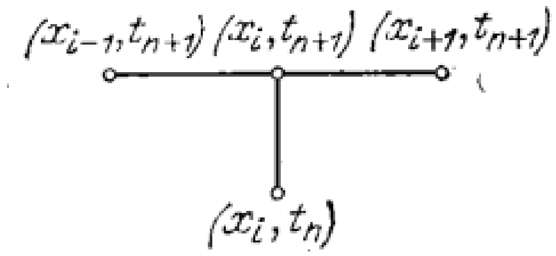
$$\max_i \left| U_i^{(k)} - U_i^{(k+1)} \right| < 0{,}01$$

Графики:





2. НЕЯВНАЯ СХЕМА



Построим простейшую неявную схему. Разностное уравнение, построенное по данной схеме:

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

$$i = \overline{1, I-1}, \qquad j = \overline{0, J-1}$$

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

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

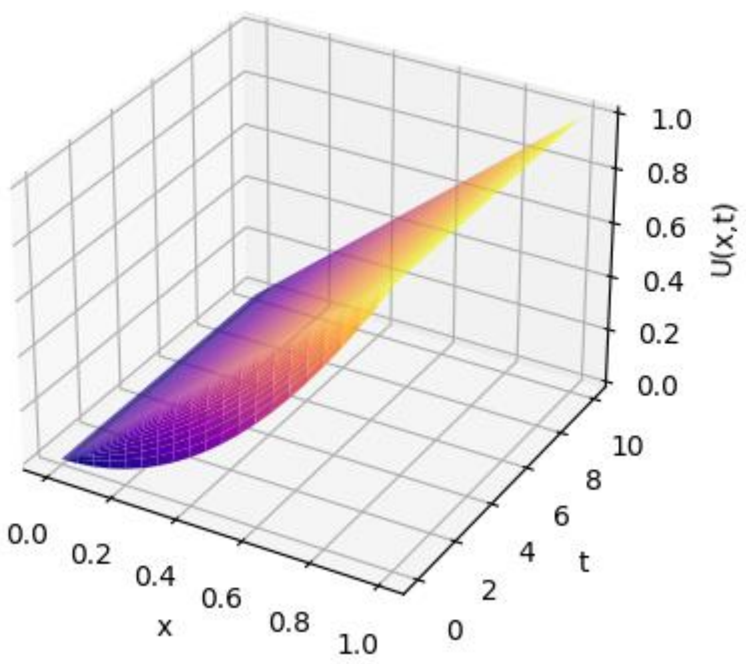
Полученная схема устойчива и сходится со скоростью $O(\tau + h^2)$. Граничные и начальные условия аналогичны явной схеме.

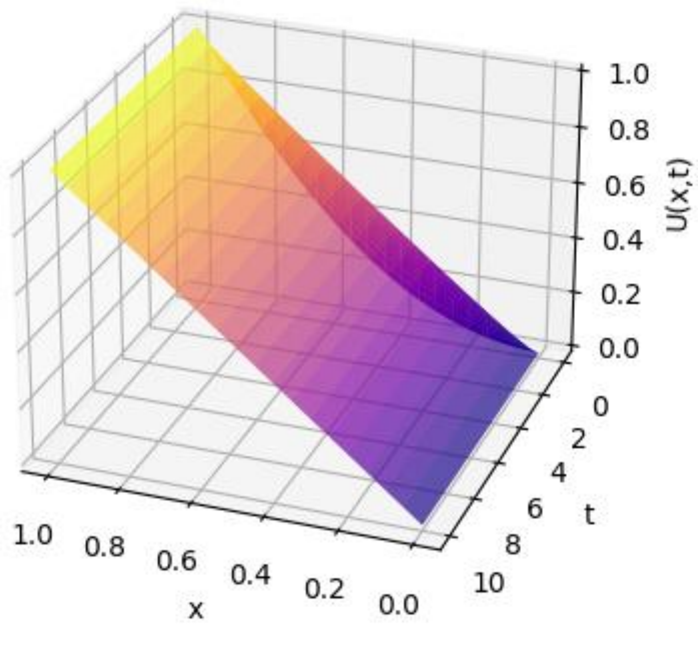
Возьмем в качестве первого шага разбиения: $h = 0.1$ и $\tau = 0.01$. Итерации завершаются при выполнении условия:

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

Систему линейных уравнений решаем методом прогонки.

Графики:


1.0
0.8
0.6
0.4
0.2
0.0
U(x,t)
10
8
6
4
2
0
t
0.0
0.2
0.4
0.6
0.8
1.0
x


1.0
0.8
0.6
0.4
0.2
0.0
U(x,t)
0
2
4
6
8
10
t
1.0
0.8
0.6
0.4
0.2
0.0
x

ПРИЛОЖЕНИЕ

```
"""
Лабораторная работа №9
Студент ОНК «ИВТ» ВШ КНиИИ направления ПМиИ 3 курса
Кондратьев Виталий
Вариант 9
"""

import numpy
import matplotlib.pyplot as plt


def Ux0(x):
    return x**2


def U0t(t):
    return 0


def Ult(t):
    return 1


h = 0.05
r = 0.00125
a = 1
p = int(1 / h) + 1
q = int(10 / r) + 1
l = a * r / (h * h)

U = [0] * p
for i in range(p):
    U[i] = [0] * q

for i in range(0, p):
    x = h * i
    U[i][0] = Ux0(x)

for j in range(1, q):
    t = r * j
    U[0][j] = U0t(t)
    U[p - 1][j] = Ult(t)

for j in range(0, q - 1):
    for i in range(1, p - 1):
        U[i][j + 1] = l * (U[i + 1][j] + U[i - 1][j]) + (1 - 2 * l) * U[i][j]

u, v = numpy.mgrid[0:p, 0:q]
x = h * u
y = r * v
z = x - x
for i in range(0, p):
    for j in range(0, q):
        z[i][j] = U[i][j]
a
fig = plt.figure(figsize=plt.figaspect(0.5))
axes = fig.add_subplot(1, 2, 1, projection='3d')
axes.set_xlabel("x")
axes.set_ylabel("t")
axes.set_zlabel("U(x,t)")
suf = axes.plot_surface(x, y, z, rstride=1, cstride=15, cmap='plasma',
edgecolor='none')
```

```python
plt.show()

U = [0] * p
for i in range(p):
    U[i] = [0] * q

for i in range(0, p):
    x = h * i
    U[i][0] = Ux0(x)

for j in range(1, q):
    t = r * j
    U[0][j] = U0t(t)
    U[p - 1][j] = Ult(t)

for j in range(0, q - 1):
    mb = [0] * p
    for i in range(0, p - 2):
        mb[i] = -l

    mc = [0] * p
    for i in range(1, p - 1):
        mc[i] = 1 + 2 * l

    ma = [0] * p
    for i in range(2, p - 1):
        ma[i] = -l

    mf = [0] * p
    for i in range(1, p - 1):
        mf[i] = U[i][j]
    mf[1] = mf[1] + l * U[0][j]
    mf[p - 2] = mf[p - 2] + l * U[p - 1][j]

    for i in range(2, p - 1):
        m = ma[i] / mc[i - 1]
        mc[i] = mc[i] - m * mb[i - 1]
        mf[i] = mf[i] - m * mf[i - 1]
    U[p - 2][j + 1] = mf[p - 2] / mc[p - 2]

    for i in range(p - 3, 0, -1):
        U[i][j + 1] = (mf[i] - mb[i] * U[i + 1][j + 1]) / mc[i]

u, v = numpy.mgrid[0:p, 0:q]
x = h * u
y = r * v
z = x - x
for i in range(0, p):
    for j in range(0, q):
        z[i][j] = U[i][j]

fig = plt.figure(figsize=plt.figaspect(0.5))
axes = fig.add_subplot(1, 2, 1, projection='3d')
axes.set_xlabel("x")
axes.set_ylabel("t")
axes.set_zlabel("U(x,t)")
suf = axes.plot_surface(x, y, z, rstride=1, cstride=15, cmap='plasma',
edgecolor='none')
plt.show()
```