Московский авиационный институт
(национальный исследовательский университет)

Институт №8 "Информационные технологии и прикладная математика"
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №6

по дисциплине:

Численные методы

Вариант №3

Выполнил: студент группы М8О-409Б-20
Чибисов М.Р.
Принял: Пивоваров Е.Д.
Оценка:____

Москва, 2023 г.

1. Задание

Используя явную схему крест и неявную схему, решить начально-краевую задачу для дифференциального уравнения гиперболического типа. Аппроксимацию второго начального условия произвести с первым и со вторым порядком. Осуществить реализацию трех вариантов аппроксимации граничных условий, содержащих производные: двухточечная аппроксимация с первым порядком, трехточечная аппроксимация со вторым порядком, двухточечная аппроксимация со вторым порядком. В различные моменты времени вычислить погрешность численного решения путем сравнения результатов с приведенным в задании аналитическим решением $U(x, t)$. Исследовать зависимость погрешности от сеточных параметров τ, h.

Уравнение:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - 3u$$

$$u(0,t) = \sin(2t)),$$

$$u(\pi,t) = -\sin(2t),$$

$$u(x,0) = 0$$

$$u_t(x,0) = 2\cos x$$

Аналитическое решение:

$$U(x,t) = \cos x \sin(2t)$$

Решение

- Явная схема (крест):

$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = 2\frac{u_{j-1}^k - 2u_j^k + u_{j+1}^k}{h^2} + 2\frac{u_{j+1}^k - u_{j-1}^k}{2h}$$

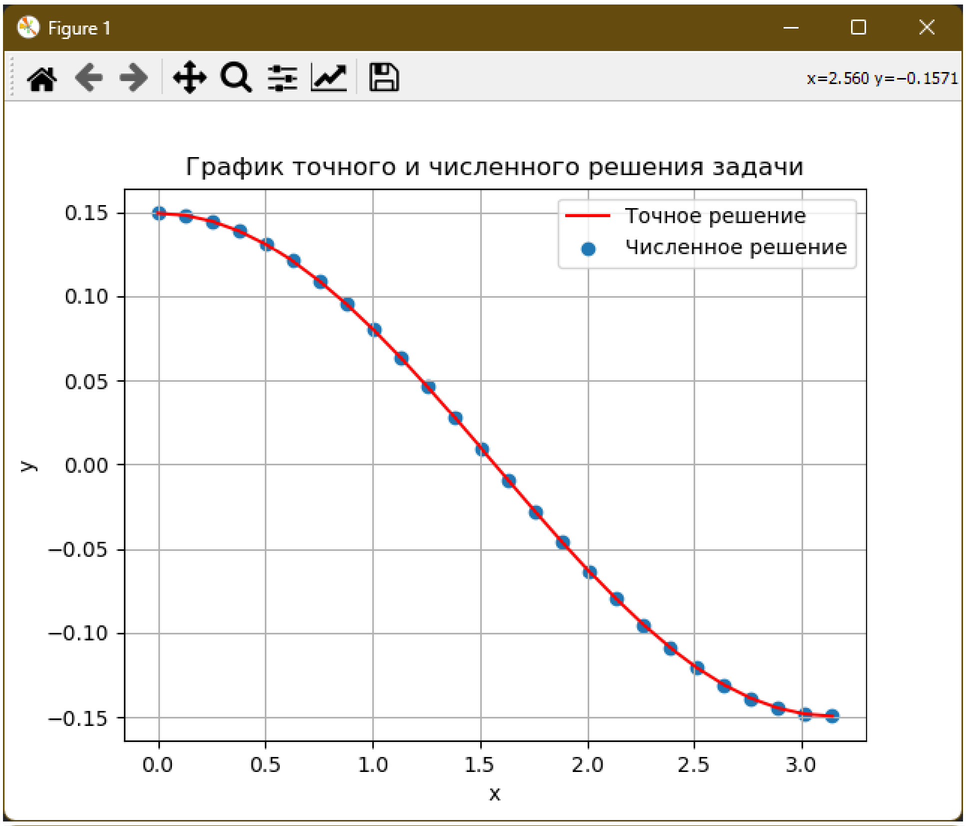
- Неявная схема:

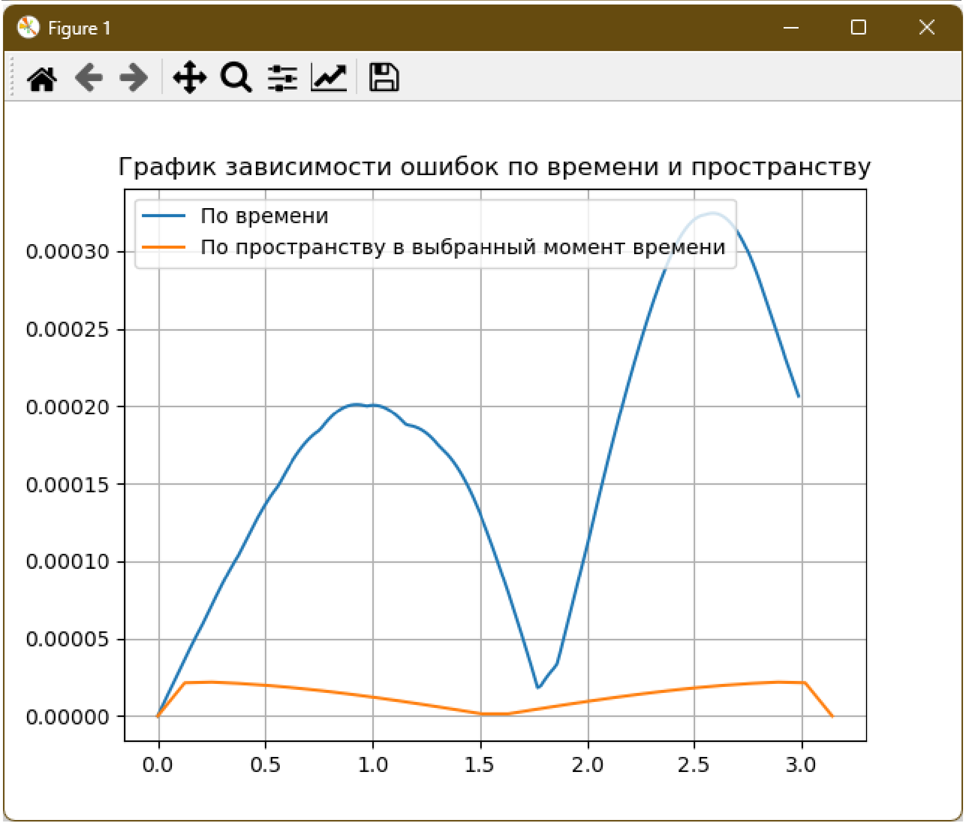
$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = 2\frac{u_{j-1}^{k+1} - 2u_j^{k+1} + u_{j+1}^{k+1}}{h^2} + 2\frac{u_{j+1}^{k+1} - u_{j-1}^{k+1}}{2h}$$

Погрешность между численным и аналитическим решением рассчитывается как модуль разности.

2. Вывод программы

- Явная схема





- Неявная схема

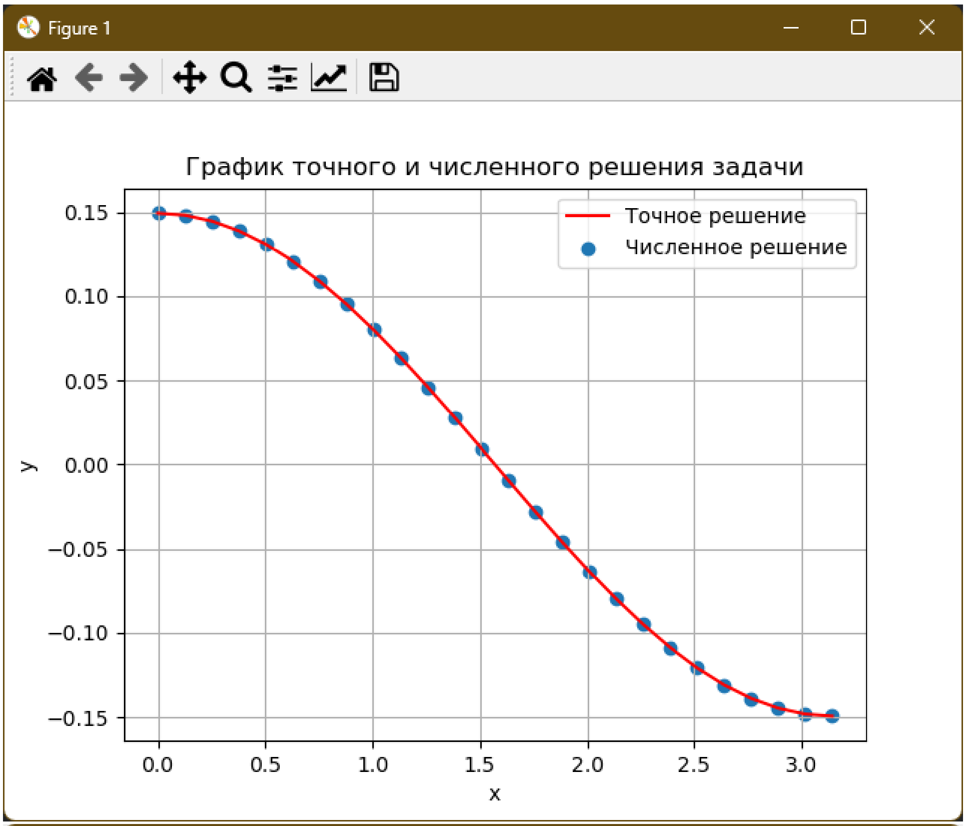


Figure 1
График точного и численного решения задачи
Точное решение
Численное решение
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
y
0.0
0.5
1.0
1.5
2.0
2.5
3.0
x

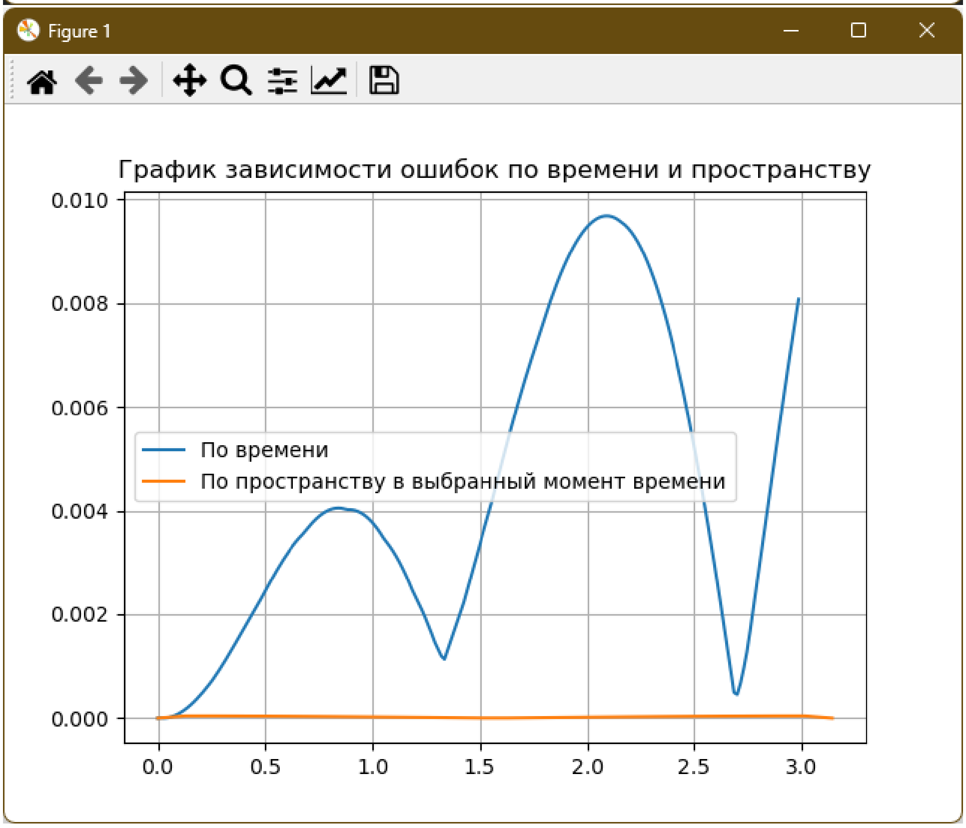


Figure 1
График зависимости ошибок по времени и пространству
По времени
По пространству в выбранный момент времени
0.010
0.008
0.006
0.004
0.002
0.000
0.0
0.5
1.0
1.5
2.0
2.5
3.0

3. Листинг

```
import matplotlib.pyplot as plt
import numpy as np

def analyt_func(x, t):
    return np.cos(x) * np.sin(2 * t)
def func_border1(t):
    return np.sin(2 * t)
def func_border2(t):
    return -np.sin(2 * t)
def run_through(a, b, c, d, s):
    P = np.zeros(s + 1)
    Q = np.zeros(s + 1)

    P[0] = -c[0] / b[0]
    Q[0] = d[0] / b[0]

    k = s - 1
    for i in range(1, s):
        P[i] = -c[i] / (b[i] + a[i] * P[i - 1])
        Q[i] = (d[i] - a[i] * Q[i-1])/(b[i] + a[i]*P[i-1])
    P[k] = 0
    Q[k] = (d[k] - a[k] * Q[k-1])/(b[k] + a[k] * P[k-1])

    x = np.zeros(s)
    x[k] = Q[k]

    for i in range(s - 2, -1, -1):
        x[i] = P[i] * x[i + 1] + Q[i]

    return x
def explicit(K, t, tau, h, x):
    N = len(x)
    U = np.zeros((K, N))
    t += tau
    for j in range(N):
        U[0, j] = 0

        U[1][j] = 2 * np.cos(x[j]) * tau

    for k in range(1, K - 1):
```

```
        t += tau
        for j in range(1, N - 1):
            U[k + 1, j] = (U[k, j + 1] * (tau ** 2 / h ** 2) + U[k, j] * (-2 * tau ** 2 / h ** 2 + 2 - 3 * tau ** 2) + U[k, j - 1] * tau ** 2 / h ** 2 - U[k - 1, j])

        U[k + 1, 0] = func_border1(t)
        U[k + 1, N - 1] = func_border2(t)

    return U

def implicit(K, t, tau, h, x):
    N = len(x)
    U = np.zeros((K, N))
    t += tau
    for j in range(N):
        U[0, j] = 0

        U[1, j] = 2 * np.cos(x[j]) * tau

    for k in range(1, K - 1):
        a = np.zeros(N)
        b = np.zeros(N)
        c = np.zeros(N)
        d = np.zeros(N)
        t += tau

        for j in range(1, N - 1):
            a[j] = 1 / h ** 2
            b[j] = -2 / h ** 2 - 1 / tau ** 2
            c[j] = 1 / h ** 2
            d[j] = U[k, j] * (3 - 2 / tau ** 2) + U[k - 1, j] / tau ** 2

        b[0] = 1
        c[0] = 0
        d[0] = func_border1(t)

        a[N - 1] = 0
        b[N - 1] = 1
        d[N - 1] = func_border2(t)
```

```python
        u_new = run_through(a, b, c, d, N)
        for i in range(N):
            U[k + 1, i] = u_new[i]

    return U


N = 25
K = 200
time = 3

h = np.pi / N
tau = time / K
print(h, tau)
x = np.arange(0, np.pi + h / 2 - 1e-4, h)
T = np.arange(0, time, tau)
t = 0
```

4. Вывод

В ходе выполнения лабораторной работы были изучены явная и неявная схемы решений начально-краевой задачи для дифференциального уравнения гиперболического типа. Также были изучены три варианта аппроксимации граничных условий и два варианта аппроксимации начальных условий. Были получены результаты в графическом представлении и подсчитаны погрешности для каждого варианта решения.