Московский авиационный институт

(Национальный исследовательский университет)

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

Лабораторная работа №6

«ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ»

Вариант 3

Выполнил: Дондоков В.И.

Группа: М8О-409Б-20

Проверил: Пивоваров Д.Е.

Дата:

Оценка:

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 и *h*.

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - 3u,$$
$$u(0,t) = \sin(2t)),$$
$$u(\pi,t) = -\sin(2t),$$
$$u(x,0) = 0,$$
$$u_t(x,0) = 2\cos x.$$

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

Теоретическая часть:

Явная конечно-разностная схема

Апроксимируем вторую производную по значениям нижнего временного слоя t^k, а именно:

$$\frac{\partial^2 u}{\partial x^2}(x_j, t^k) = \frac{u_{j-1}^k - 2u_j^k + u_{j+1}^k}{h^2}$$

Тогда получим явную схему конечно-разностного метода во внутренних узлах сетки:

$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = \frac{u_{j-1}^k - 2u_j^k + u_{j+1}^k}{h^2} - 5u_j^k,\ \forall j \in \{1, ..., N-1\}, \forall k \in \{0, ..., K-1\}$$

Обозначим $\sigma = \frac{\tau^2}{h^2}$, тогда:

$$u_j^{k+1} = \sigma(u_{j+1}^k - 2u_j^k + u_{j-1}^k) - 5\tau^2 u_j^k + 2u_j^k - u_j^{k-1}$$

Граничные же значения u_0^{k+1} и u_N^{k+1} определяются граничными условиями $u_x(0,t)$ и $u_x(l,t)$ при помощи апроксимации производной.

Конечно-разностная схема

Будем решать задачу на заданном промежутке от 0 до l по координате x и на промежутке от 0 до заданного параметра T по времени t.

Рассмотрим конечно-разностную схему решения краевой задачи на сетке с граничными парамерами l, T и параметрами насыщенности сетки N, K. Тогда размер шага по каждой из координат определяется:

$$h = \frac{l}{N-1},\ \tau = \frac{T}{K-1}$$

Считая, что значения функции $u_j^k = u(x_j, t^k)$ для всех координат $x_j = jh,\ \forall j \in \{0, ..., N\}$ на предыдущих временных известно, попробуем определить значения функции на временном слое t^{k+1} путем разностной апроксимации производной:

$$\frac{\partial^2 u}{\partial t^2}(x_j, t^k) = \frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2}$$

И одним из методов апроксимации второй производной по x:

$$\frac{\partial^2 u}{\partial x^2}(x_j, t^k)$$

Для расчета u_j^0 и u_j^1 можно использовать следующие формулы:

$$u_j^0 = \psi_1(x_j)$$

$$u_j^1 = \psi_1(x_j) + \tau\psi_2(x_j) + \frac{\tau^2}{2}\psi_1''(x_j) + O(\tau^2)$$

$$u_j^1 = \psi_1(x_j) + \tau\psi_2(x_j) + O(\tau^1)$$

Неявная конечно-разностная схема

Апроксимируем вторую производную по значениям верхнего временного слоя t^{k+1}, а именно:

$$\frac{\partial^2 u}{\partial x^2}(x_j, t^k) = \frac{u_{j-1}^{k+1} - 2u_j^{k+1} + u_{j+1}^{k+1}}{h^2}$$

Тогда получим явную схему конечно-разностного метода во внутренних узлах сетки:

$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = \frac{u_{j-1}^{k+1} - 2u_j^{k+1} + u_{j+1}^{k+1}}{h^2} - 5u_j^{k+1},\ \forall j \in \{1, ..., N-1\}, \forall k \in \{0, ..., K-1\}$$

Обозначим $\sigma = \frac{\tau^2}{h^2}$. Тогда значения функции на слое можно найти эффективны образом с помощью методом прогонки, где **СЛАУ**, кроме крайних двух уравнений, определяется коэффицетнами $a_j = 1, b_j = -(2 + 5h^2 + \frac{1}{\sigma}), c_j = 1, d_j = \frac{-2u_j^k + u_j^{k-1}}{\sigma}$ уравнений:

$$a_j u_{j-1}^{k+1} + b_j u_j^{k+1} + c_j u_{j+1}^{k+1} = d_j,\ \forall j \in \{1, ..., N-1\}$$

Первое и последнее уравнение системы содержащие u_0^{k+1} и u_N^{k+1} определяются граничными условиями при помощи апроксимации производной.

Неявная схема является абсолютно устойчивой.

Код программы:

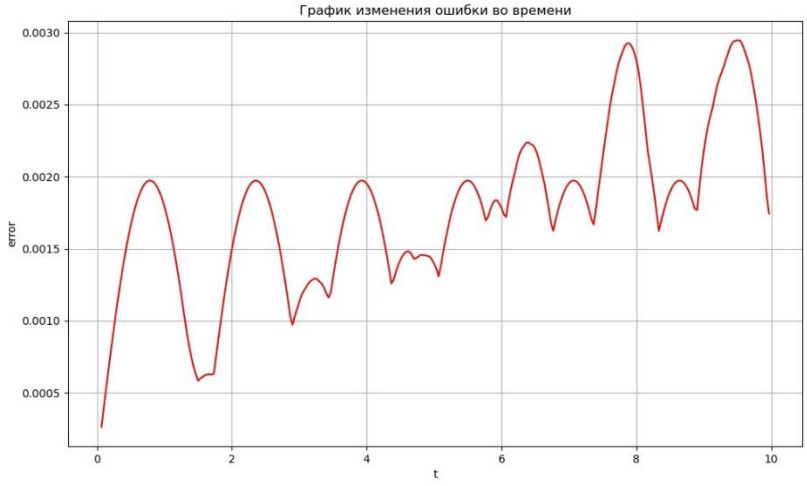
```
def phi0(t):
    return np.sin(2*t)


def phi1(t):
    return -np.sin(2*t)


def psi1(x, t=0):
    return 0


def psi2(x, t=0):
    return 2*np.cos(x)
```

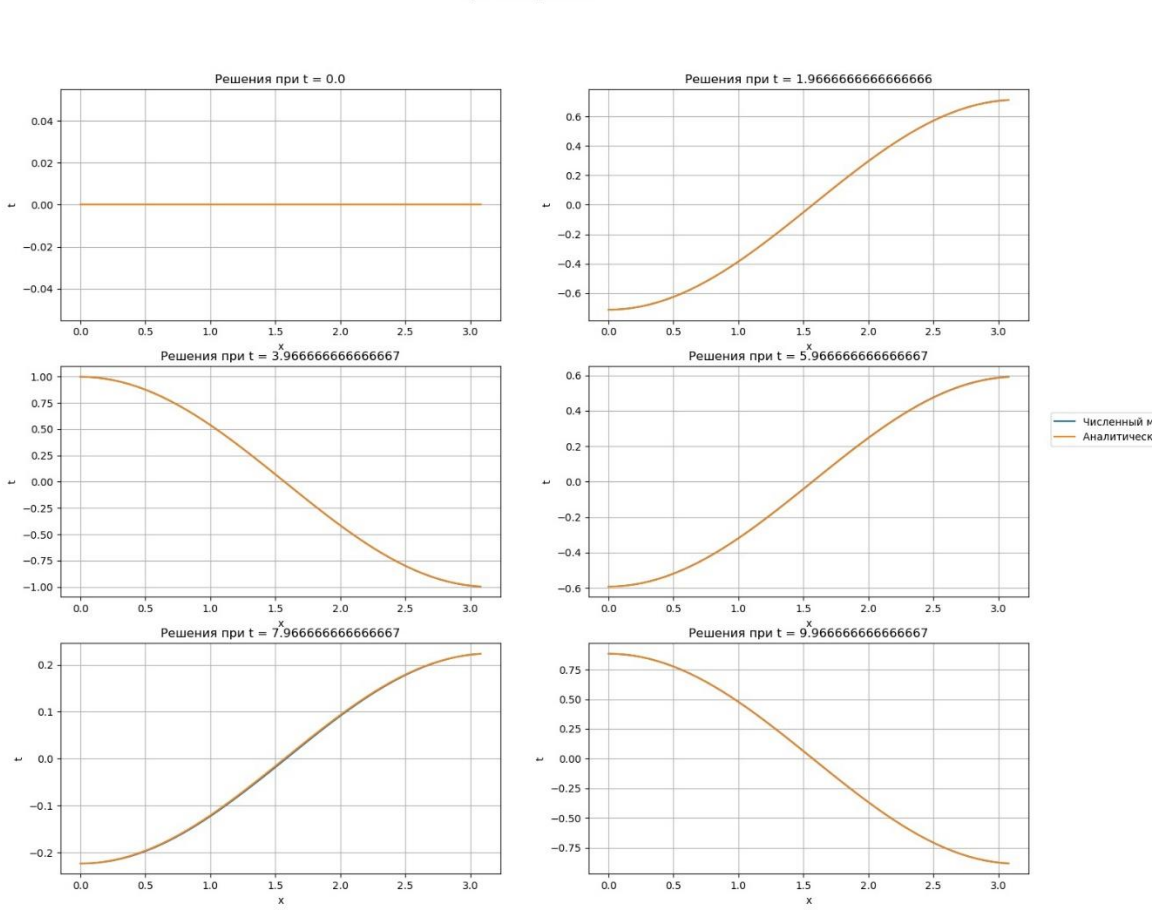
```
def U(x, t):
    return np.cos(x)*np.sin(2*t)

# Метод прогонки
def equation_solve(a, b, c, d):
    size = len(a)
    p = np.zeros(size)
    q = np.zeros(size)
    p[0] = -c[0] / b[0]
    q[0] = d[0] / b[0]
    for i in range(1, size):
        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])
    x = np.zeros(size)
    x[-1] = q[-1]
    for i in range(size - 2, -1, -1):
        x[i] = p[i] * x[i + 1] + q[i]
    return x

def explicit(n: int, tc: int, t: float, h: float, a, c):
    sigma = a**2 * t**2 / h**2
    print(f'Sigma = {sigma}')
    if sigma > 1:
        raise Exception(f'Явная схема не устойчива sigma = {sigma}')
    u = np.zeros((tc, n))
    u[0] = psi1(h * np.arange(n))
    u[1] = u[0] + t*psi2(h * np.arange(n))
    for k in range(1, tc - 1):
        for j in range(1, n - 1):
            u[k+1][j] = u[k][j+1] * sigma + u[k][j] * (-2*sigma + 2 + c*t**2)
+ u[k][j-1] * sigma - u[k-1][j]
        u[k+1][0] = phi0((k+1) * t)
        u[k+1][-1] = phi1((k+1) * t)
    return u

def implicit(n: int, tc: int, t: float, h: float, ap, cp):
    sigma = ap**2 * t**2 / h**2
    print(f'Sigma = {sigma}')
    u = np.zeros((tc, n))
    u[0] = psi1(h * np.arange(n))
    u[1] = u[0] + t*psi2(h * np.arange(n))
    for k in range(1, tc - 1):
        a = np.full(n, sigma)
        b = np.full(n, -(1 + 2*sigma))
        c = np.full(n, sigma)
        d = np.zeros(n)
        for j in range(1, n - 1):
            d[j] = u[k-1][j] - (cp*t**2 + 2)*u[k][j]
        a[0] = 0
        b[0] = 1
        c[0] = 0
        d[0] = phi0((k+1)*t)
        a[-1] = 0
        b[-1] = 1
        c[-1] = 0
        d[-1] = phi1((k+1)*t)
        u[k+1] = equation_solve(a, b, c, d)
    return u

def draw_results(tc, x_max, u, a, n, t_max):
    times = np.zeros(tc)
    t = t_max/tc
    for i in range(tc):
```

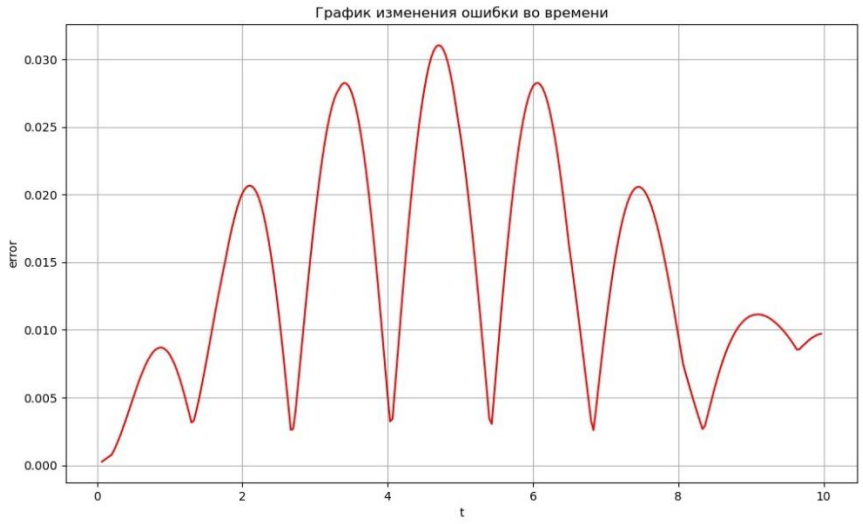
```
        times[i] = t * i
    space = np.zeros(n)
    step = x_max / n
    for i in range(n):
        space[i] = i * step

    times_idx = np.linspace(0, times.shape[0] - 1, 6, dtype=np.int32)
    fig, ax = plt.subplots(3, 2)
    fig.suptitle('Сравнение решений')
    fig.set_figheight(15)
    fig.set_figwidth(16)
    k = 0
    for i in range(3):
        for j in range(2):
            time_idx = times_idx[k]
            ax[i][j].plot(space, u[time_idx], label='Численный метод')
            ax[i][j].plot(space, [U(x, times[time_idx]) for x in space],
label='Аналитическое решение')
            ax[i][j].grid(True)
            ax[i][j].set_xlabel('x')
            ax[i][j].set_ylabel('t')
            ax[i][j].set_title(f'Решения при t = {times[time_idx]}')
            k += 1
    plt.legend(bbox_to_anchor=(1.05, 2), loc='upper left', borderaxespad=0.)
    error = np.zeros(tc)
    for i in range(tc):
        error[i] = np.max(np.abs(u[i] - np.array([U(x, times[i]) for x in
space])))
    plt.figure(figsize=(12, 7))
    plt.plot(times[2:], error[2:], 'red', label='Ошибка')
    plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0.)
    plt.title('График изменения ошибки во времени')
    plt.xlabel('t')
    plt.ylabel('error')
    plt.grid(True)
    plt.show()

a = 1
c = -3

n = 50 # 65
tc = 300

x_max = np.pi
t_max = 10

h = x_max / n
t = t_max / tc

u = explicit(n=n, tc=tc, t=t, h=h, a=a, c=c)
u

draw_results(tc, x_max, u, a, n, t_max)

a = 1
c = -3

n = 50 # 65
tc = 300

x_max = np.pi
t_max = 10

h = x_max / n
```

```
t = t_max / tc

u = implicit(n=n, tc=tc, t=t, h=h, ap=a, cp=c)
u

draw_results(tc, x_max, u, a, n, t_max)
```

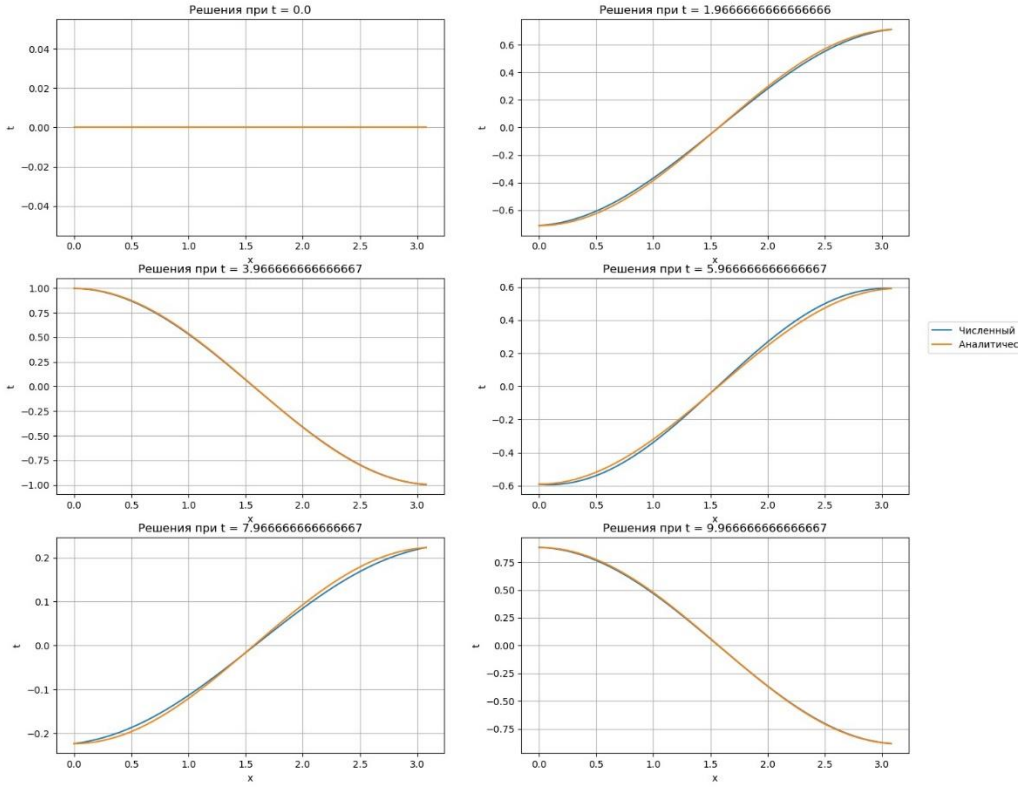
Результат:







Сравнение решений



Вывод:

В ходе лабораторной работы были изучены *явная схема крест и неявная схема*, решена с их помощью начально-краевую задача для дифференциального уравнения гиперболического типа. Проведена аппроксимация второго начального условия с первым и со вторым порядком. Осуществлена реализация 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 вычислена погрешность численного решения путем сравнения результатов с приведенным в задании аналитическим решением $u\ (x,\ t)$. Исследована зависимость погрешности от сеточных параметров τ и h.