Основы математического моделирования. Отсчет по практическому заданию №2

Рем Данилин группа 307

Вариант48

1 Аналитическое решение

$$\begin{cases} \frac{\partial u}{\partial t} = \Delta u + \sin x \sin t \quad 0 < x < \pi \quad 0 < y < 3 \\ u|_{x=0} = u|_{x=\pi} = 0 \\ u|_{y=0} = u|_{y=3} = 0 \\ u|_{t=0} = 0 \end{cases} \tag{1}$$

Из вида задачи (1) можно понять, что начальные условия и неоднородность ортогональны. Поэтому будем искать решение в виде:

$$u(x, y, t) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} T_{nm} V_{nm}(x, y)$$

Подставляя в уравнение получем следующую систему уравнений для поиска T_{nm} и V_{nm}:

$$\begin{cases} \frac{dT_{nm}}{dt} + \lambda_{nm} T_{nm} = f_{nm}(t) \\ T_{nm}(0) = 0 \end{cases}$$

$$\begin{cases} \Delta V_{nm} + \lambda_{nm} V_{nm} = 0 \\ V|_{x=0} = V|_{x=\pi} = V|_{y=0} = V|_{y=3} = 0 \end{cases}$$

Повторно разделяя переменные в последней системе для $V_{nm} = X_n Y_m$ приходим к следующему результату:

$$\begin{cases} X_n = \sin(nx) \\ Y_m = \sin(\frac{m\pi}{3}) \\ \lambda_{nm} = n + \frac{m\pi}{3} \end{cases}$$

Преступим к решению задачи по поиску T_{nm}. Для этого, сначала необходимо получить явный вид функции f_{nm}. Его можно получить, вычислив интеграл

$$f_{nm} = \frac{1}{|V_{nm}|^2} \int\int \sin x \sin t V_{nm} dx dy = \frac{4 \sin t}{3\pi} \delta_{1n} \frac{6}{\pi m} \quad \text{if m is odd}$$

Получаем задачу Коши:

$$\begin{cases} \frac{dT_{nm}}{dt} + \lambda_{nm}T_{nm} = \frac{4\sin t}{3\pi}\delta_{1n}\frac{6}{\pi m} \quad \text{if m is odd} \\ \lambda_{nm} = n + \frac{m\pi}{3} \\ T_{nm}(0) = 0 \end{cases}$$

Решение задачи Коши для можно записать с использованием импульсной функции Коши:

$$T_{nm} = \int\limits_0^t e^{-\lambda_{nm}(t-\tau)}\frac{4\sin\tau}{3\pi}\delta_{1n}\frac{6}{\pi m}d\tau = \frac{12}{\pi^2 m}\frac{e^{-\lambda_{nm}t} - \cos t + \lambda_{nm}\sin t}{1+\lambda_{nm}^2} \quad \text{if m is odd and n} = 1$$

Остальные члены ряда, когда $n \neq 1$ и m- четное занулятся.

В итоге конечное решение будет представлять ряд:

$$u(x,y,t) = \sum_{\text{m is odd}}^{\infty} \frac{12}{\pi^2 m}\frac{e^{-\lambda_{nm}t} - \cos t + \lambda_{nm}\sin t}{1+\lambda_{nm}^2} \quad \sin x \sin\left(\frac{\pi m y}{3}\right)$$

Численно ряд был посчитан только до 1000 члена, что в дальнейшем сильно повлияет на погрешность численного решения на концах области.

2 Описание разностной схемы

Так как в данной задаче мы имеем граничные условия Дерихле, то вводить в расчетной области равномерную сетку можно следующим способом.

$$x_n = nh_x \quad n = 0,1,2...N \quad h_x = \frac{1}{N}$$

$$y_m = mh_y \quad m = 0,1,2...M \quad h_y = \frac{2}{M}$$

$$t_j = j\tau \quad j = 0,1,2...J \quad \tau = \frac{T}{J}$$

Начальные и граничные условия аппроксимируются точно. Граничные условия не зависят от времени. Пусть ω - сеточная функция, которая является решением.

$$\begin{cases} \omega_{0m}^j = \omega_{Nm}^j = 0 \quad m = 0,1,2...M \\ \omega_{n0}^j = \omega_{nM}^j = 0 \quad n = 0,1,2...N \\ \omega_{nm}^0 = \sin 2\pi x_n \sin \pi y_m \quad n = 0,1,2...N \quad m = 0,1,2...M \end{cases} \tag{2}$$

Предположим, что решение на слое j нам известно найдем решение на слое $j+1$ в два этапа:

$$j \to j + \frac{1}{2}$$

$$j+\frac{1}{2} \to j+1$$

Перейдем к рассмотрению **первого этапа**. С учетом того, что граничные условия **не зависят от времени**, систему для данного этапа решения можно написать следующим образом.

$$\begin{cases} \frac{\omega_{nm}^{j+\frac{1}{2}}-\omega_{nm}^{j}}{0.5\tau} = \frac{1}{h_x^2}\left(\omega_{n+1,m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1,m}^{j+\frac{1}{2}}\right) + \frac{1}{h_y^2}\left(\omega_{nm+1}^{j} - 2\omega_{nm}^{j} + \omega_{nm-1}^{j}\right), m=1,2...M-1; n=1..N- \\ \\ \omega_{0m}^{j+\frac{1}{2}} = 0 \quad m=1,2...M-1 \\ \omega_{Nm}^{j+\frac{1}{2}} = 0 \quad m=1,2...M-1 \end{cases} \tag{3}$$

Для каждого $m = 1, 2...M-1$ данную систему можно перезаписать следующим образом (для того, чтобы явно была видна структура трехдиогональной матрицы).

$$\begin{cases} \omega_{0m}^{j+\frac{1}{2}} = 0 \\ \frac{1}{h_x^2}\omega_{n-1,m}^{j+\frac{1}{2}} - (\frac{2}{\tau}+\frac{2}{h_x^2})\omega_{nm}^{j+\frac{1}{2}} + \frac{1}{h_x^2}\omega_{n+1,m}^{j+\frac{1}{2}} = -\frac{1}{h_y^2}\left(\omega_{nm+1}^{j} - 2\omega_{nm}^{j} + \omega_{nm-1}^{j}\right) - \frac{\omega_{nm}^{j}}{0.5\tau} \\ \omega_{Nm}^{j+\frac{1}{2}} = 0 \end{cases} \tag{4}$$

$$\begin{cases} \omega_{0m}^{j+\frac{1}{2}} = 0 \\ A^x\omega_{n-1,m}^{j+\frac{1}{2}} - C^x\omega_{nm}^{j+\frac{1}{2}} + B^x\omega_{n+1,m}^{j+\frac{1}{2}} = -F_n^x \\ \omega_{Nm}^{j+\frac{1}{2}} = 0 \end{cases} \tag{5}$$

Где

$$A^x = \frac{1}{h_x^2} \quad B^x = \frac{1}{h_x^2} \quad C^x = \frac{2}{\tau} + \frac{2}{h_x^2}$$

$$F_n^x = \frac{1}{h_y^2}\left(\omega_{nm+1}^{j} - 2\omega_{nm}^{j} + \omega_{nm-1}^{j}\right) + \frac{\omega_{nm}^{j}}{0.5\tau}$$

Решаем данную систему численно методом прогонки.

n = 1

$$-C^x\omega_{1m}^{j+\frac{1}{2}} + B^x\omega_{2m}^{j+\frac{1}{2}} = -F_1^x$$

Выражаем отсюда $\omega_{1m}^{j+\frac{1}{2}}$.

$$\omega_{1m}^{j+\frac{1}{2}} = \frac{B^x}{C^x}\omega_{2m}^{j+\frac{1}{2}} + \frac{F_1^x}{C^x} = \alpha_1\omega_{2m}^{j+\frac{1}{2}} + \beta_1$$

n = 2

$$A^x\omega_{1m}^{j+\frac{1}{2}} - C^x\omega_{2m}^{j+\frac{1}{2}} + B^x\omega_{3m}^{j+\frac{1}{2}} = -F_2^x$$

Подставляем ранее найденный $\omega_{1m}^{j+\frac{1}{2}}$ и выражаем $\omega_{2m}^{j+\frac{1}{2}}$.

$$\omega_{2m}^{j+\frac{1}{2}} = \frac{B^x}{C^x - A^x\alpha_1}\omega_{3m}^{j+\frac{1}{2}} + \frac{F_2^x + A^x\beta_1}{C^x - A^x\alpha_1} = \alpha_2\omega_{3m}^{j+\frac{1}{2}} + \beta_2$$

И так далее продолжаем подсчитывать коэффициенты α_n и β_n по реккурентным формулам:

$$\alpha_n = \frac{B^x}{C^x - A^x\alpha_{n-1}} \quad \beta_n = \frac{F_n^x + A^x\beta_{n-1}}{C^x - A^x\alpha_{n-1}} \quad \forall n = 1, 2, ..., N-1$$

$$\alpha_0 = \beta_0 = \alpha_N = \beta_N = 0$$

По коэффициентам восстанавливаем значения сеточной функции на слое $j + \frac{1}{2}$, используем обратную прогонку.

$$\begin{cases} \omega_{Nm}^{j+\frac{1}{2}} = \beta_N \\ \omega_{nm}^{j+\frac{1}{2}} = \alpha_n \omega_{n+1m}^{j+\frac{1}{2}} + \beta_n \end{cases} \tag{6}$$

Теперь переходим ко **второму этапу**, а именно переходу со слоя $j + \frac{1}{2} \to j + 1$. Запишем разностную систему для этого перехода.

$$\begin{cases} \frac{\omega_{nm}^{j+1} - \omega_{nm}^{j+\frac{1}{2}}}{0.5\tau} = \frac{1}{h_x^2}\left(\omega_{n+1m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1m}^{j+\frac{1}{2}}\right) + \frac{1}{h_y^2}\left(\omega_{nm+1}^{j+1} - 2\omega_{nm}^{j+1} + \omega_{nm-1}^{j+1}\right) \\ \omega_{n0}^{j+1} = 0 \\ \omega_{nM}^{j+1} = 0 \end{cases} \tag{7}$$

При каждом фиксированном $n = 1, 2, ..., N-1$ решаем систему, как и в прошлый раз, методом прогонки. Перепишем уравнение.

$$\begin{cases} \omega_{n0}^{j+1} = 0 \\ \frac{1}{h_y^2}\omega_{nm+1}^{j+1} - (\frac{2}{h_y^2} + \frac{2}{\tau})\omega_{nm}^{j+1} + \frac{1}{h_y^2}\omega_{nm-1}^{j+1} = -\frac{\omega_{nm}^{j+\frac{1}{2}}}{0.5\tau} - \frac{1}{h_x^2}\left(\omega_{n+1m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1m}^{j+\frac{1}{2}}\right) \\ \omega_{nM}^{j+1} = 0 \end{cases} \tag{8}$$

$$\begin{cases} \omega_{n0}^{j+1} = 0 \\ A^y\omega_{n,m-1}^{j+1} - C^y\omega_{nm}^{j+1} + B^y\omega_{n,m+1}^{j+1} = -F_m^y \\ \omega_{nM}^{j+1} = 0 \end{cases} \tag{9}$$

Где

$$A^y = B^y = \frac{1}{h_y^2}$$

$$F_m^y = \frac{\omega_{nm}^{j+\frac{1}{2}}}{0.5\tau} + \frac{1}{h_x^2}\left(\omega_{n+1m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1m}^{j+\frac{1}{2}}\right)$$

Решаем данную систему численно методом прогонки.

m = 1

$$-C^y\omega_{n1}^{j+1} + B^x\omega_{n2}^{j+1} = -F_1^y$$

Выражаем отсюда ω_{n1}^{j+1}.

$$\omega_{n1}^{j+1} = \frac{B^y}{C^y}\omega_{2m}^{j+1} + \frac{F_1^y}{C^y} = \alpha_1\omega_{n2}^{j+1} + \beta_1$$

m = 2

$$A^y\omega_{n1}^{j+1} - C^y\omega_{n2}^{j+1} + B^x\omega_{n3}^{j+1} = -F_3^y$$

Подставляем ранее найденный ω_{n2}^{j+1} и выражаем ω_{n2}^{j+1}.

$$\omega_{n2}^{j+1} = \frac{B^y}{C^y - A^y\alpha_1}\omega_{n3}^{j+1} + \frac{F_2^y + A^y\beta_1}{C^y - A^y\alpha_1} = \alpha_2\omega_{n3}^{j+1} + \beta_2$$

И так далее продолжаем подсчитывать коэффициенты α_m и β_m по реккурентным формулам:

$$\alpha_m = \frac{B^y}{C^y - A^y\alpha_{m-1}} \quad \beta_m = \frac{F_m^y + A^y\beta_{m-1}}{C^y - A^y\alpha_{m-1}} \quad \forall m = 1, 2, ..., M-1$$

$$\alpha_0 = \beta_0 = \alpha_M = \beta_M = 0$$

По коэффициентам восстанавливаем значения сеточной функции на слое $j+1$, используем обратную прогонку.

$$\begin{cases} \omega_{nM}^{j+1} = \beta_M \\ \omega_{nm}^{j+1} = \alpha_m\omega_{nm+1}^{j+1} + \beta_m \end{cases} \tag{10}$$

Обоснование устойчивости

Рассмотрим уравнения, полученные ранее (5) и (9):

$$\frac{\omega_{nm}^{j+\frac{1}{2}} - \omega_{nm}^{j}}{0.5\tau} = \frac{1}{h_x^2}\left(\omega_{n+1,m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1,m}^{j+\frac{1}{2}}\right) + \frac{1}{h_y^2}\left(\omega_{nm+1}^{j} - 2\omega_{nm}^{j} + \omega_{nm-1}^{j}\right)$$

$$\frac{\omega_{nm}^{j+1} - \omega_{nm}^{j+\frac{1}{2}}}{0.5\tau} = \frac{1}{h_x^2}\left(\omega_{n+1m}^{j+\frac{1}{2}} - 2\omega_{nm}^{j+\frac{1}{2}} + \omega_{n-1m}^{j+\frac{1}{2}}\right) + \frac{1}{h_y^2}\left(\omega_{nm+1}^{j+1} - 2\omega_{nm}^{j+1} + \omega_{nm-1}^{j+1}\right)$$

Выражаем $\omega^{1+\frac{1}{2}}$ из первого и подставляем во второе, также введем обозначение для разностных операторов.

$$\Lambda_1 u = u_{\bar{x}x} = \frac{u_{n-1m} - 2u_{nm} + u_{n+1m}}{h_x^2}$$

$$\Lambda_2 u = u_{\bar{y}y} = \frac{u_{nm-1} - 2u_{nm} + u_{nm+1}}{h_y^2}$$

В итоге получим

$$\omega_t - \frac{\tau}{2}\Lambda_2\omega_t - \frac{\tau}{2}\Lambda_1\omega_t + \frac{\tau^2}{4}\Lambda_1\Lambda_2\omega_t = \Lambda_1\omega^j + \Lambda_2\omega^j$$

Перепишем его немного в другом виде

$$(E - \frac{1}{2}\tau\Lambda_1)(E - \frac{1}{2}\tau\Lambda_2)\omega_t = \Lambda\omega^j$$

Исследуем схему на устойчивость по начальным условиям методом гармоник. Рассмотрим задачу Коши

$$\begin{cases} (E - \frac{1}{2}\tau\Lambda_1)(E - \frac{1}{2}\tau\Lambda_2)\omega_t = \Lambda\omega^j \\ \omega^0_{nm} = e^{i(\alpha_q n + \beta_p m)} \end{cases} \tag{11}$$

Тогда на слое $j+1$ решение будет иметь вид

$$\omega^j_{nm} = \lambda^j_{qp} e^{i(\alpha_q n + \beta_p m)}$$

Найдем явный вид множителя роста, подставляя его в основное уравнение

$$(E - \frac{1}{2}\tau\Lambda_1)(E - \frac{1}{2}\tau\Lambda_2)\frac{\lambda^{j+1}_{qp} - \lambda^j_{qp}}{\tau} e^{i(\alpha_q n + \beta_p m)} = \lambda^j_{qp}\Lambda e^{i(\alpha_q n + \beta_p m)}$$

Так как

$$\Lambda_1 e^{i(\alpha_q n + \beta_p m)} = e^{i(\alpha_q n + \beta_p m)} \frac{e^{i\alpha_q} - 2 + e^{-i\alpha_q}}{h_x^2} = -e^{i(\alpha_q n + \beta_p m)} \frac{4}{h_x^2} \sin^2 \frac{\alpha_q}{2},$$

$$
\Lambda_2 e^{i(\alpha_q n+\beta_p m)} = -e^{i(\alpha_q n+\beta_p m)} \frac{4}{h_y^2} \sin^2 \frac{\beta_p}{2},
$$

То подставляя выражения и, сокращая на $\lambda_{pq}^j e^{i(\alpha_q n+\beta_p m)}$, получаем

$$
\frac{\lambda_{q,p}-1}{\tau} \left(1+\frac{2\tau}{h_x^2} \sin^2 \frac{\alpha_q}{2}\right) \left(1+\frac{2\tau}{h_y^2} \sin^2 \frac{\beta_p}{2}\right) = -4 \left(\frac{1}{h_x^2} \sin^2 \frac{\alpha_q}{2} + \frac{1}{h_y^2} \sin^2 \frac{\beta_p}{2}\right),
$$

$$
\lambda_{q,p} = \frac{\left(1-\frac{2\tau}{h_x^2} \sin^2 \frac{\alpha_q}{2}\right) \left(1-\frac{2\tau}{h_y^2} \sin^2 \frac{\beta_p}{2}\right)}{\left(1+\frac{2\tau}{h_x^2} \sin^2 \frac{\alpha_q}{2}\right) \left(1+\frac{2\tau}{h_y^2} \sin^2 \frac{\beta_p}{2}\right)}.
$$

Отсюда следует, что множитель роста по модулю не превосходит единицу, поэтому данная схема является **безусловно устойчивой**.

3 Код программы

```
from mpl_toolkits.mplot3d import Axes3D  # noqa: F401 unused import
import math
from scipy import optimize
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import numpy as np


max_x = math.pi
max_y = 3
TT = 0.07
N = 100
M = 100
J = 100
hx = max_x / N
hy = max_y / M
tau = TT/J

w = np.zeros((N+1,M+1,J+1))  # трехмерный массив для
# записи решения заполняется пока что нулями
# содержит только целые слои j и j+1

w_for_plot = np.zeros((N+1,M+1))

w05 = np.zeros((N+1,M+1))  # двумерный массив для
# записи решения заполняется пока что нулями
# содержит только значения на заданном полуцелом слое j+1/2

err = np.zeros((N+1,M+1)) # погрешность
err_for_plot = np.zeros((N+1,M+1))

analitic_sol = np.zeros((N+1,M+1))  # трехмерный массив для
# записи аналитического решения заполняется пока что нулями
# нужен для построения графика погрешности

analitic_sol_for_plot = np.zeros((N+1,M+1))# двумерный массив для
# построения аналитического решения в конечный момент времени

def Fy(w_plus1:float,w:float,w_minus1:float, x: float, t: float):
    return 1/hy**2 * (w_plus1 - 2*w + w_minus1) + 2*w/tau + math.sin(x) * math.sin(t)
def Fx(w_plus1:float,w:float,w_minus1:float, x: float, t: float):
    return 1/hx**2 * (w_plus1 - 2*w + w_minus1) + 2*w/tau + math.sin(x) * math.sin(t)
def b(i: int):
```

```
    return 12/((math.pi**2)*i)
def lmbd(i: int):
    return 1 + (math.pi*i)/3
def analitic(x: float,y: float,t: float):
    a = 0
    for i in range(1, 1000, 2):
        a += (math.sin(x)*math.sin(math.pi*y*i/3))*( b(i) * math.exp(-lmbd(i) * t) - b(i)*ma
    return a
def plot(X: list,Y: list,sol: np.ndarray):
    fig = plt.figure()
    ax = fig.gca(projection='3d')
    # Plot the surface.
    X, Y = np.meshgrid(X, Y)
    surf = ax.plot_surface(X, Y, sol, cmap='inferno', linewidth=0, antialiased=False)

    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_zlabel('u')
    ax.set_zlim3d(0.0, 0.004)

    # Add a color bar which maps values to colors.
    fig.colorbar(surf, shrink=0.5, aspect=5)
    plt.show()
def plot_error(X: list,Y: list,sol: np.ndarray):
    fig = plt.figure()
    ax = fig.gca(projection='3d')
    # Plot the surface.
    X, Y = np.meshgrid(X, Y)
    surf = ax.plot_surface(X, Y, sol, cmap='inferno', linewidth=0, antialiased=False)

    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_zlabel('u')
    ax.set_zlim3d(-0.1, 100.)
    # Add a color bar which maps values to colors.
    fig.colorbar(surf, shrink=0.5, aspect=5)
    plt.show()
def start_condition(x: float, y: float):
    return  0


# Создание сетки
X = [i*hx for i in range(N+1)]
Y = [i*hy for i in range(M+1)]
T = [i*tau for i in range(J+1)]
# -------------------------------------------------------
```

```
# Аналитическое решение построение
max_u = 0
for i in range(N+1):
    for j in range(M+1):
        a = analitic(X[i], Y[j], T[J])
        analitic_sol[i][j] = a
        if max_u < a:
            max_u = a
#max_u = 0
#for i in range(N+1):
#    for j in range(M+1):
#        analitic_sol_for_plot[i][j] = analitic(X[i], Y[j], T[J])
#        if max_u < analitic(X[i], Y[j], T[J]):
#            max_u = analitic(X[i], Y[j], T[J])

# -------------------------------------------------------
# Численное решение
# зададим начальные условия
for i in range(N+1):
    for j in range(M+1):
        w[i][j][0] = start_condition(X[i], Y[j])
# коэффициенты в ситсемах
Ax = 1/hx**2
Ay = 1/hy**2
Bx = 1/hx**2
By = 1/hy**2
Cx = 2/tau + 2/hx**2
Cy = 2/tau + 2/hy**2
# прогоначные коэффциценты
alpha_x = [0 for i in range(N)]
beta_x = [0 for i in range(N)]
alpha_y = [0 for i in range(M)]
beta_y = [0 for i in range(M)]

for j in range(0,J):
    # переход на слой j+1/2
    for m in range(1,M):
        alpha_x[0] = 0
        beta_x[0] = 0
        # прямой ход прогонки
        for n in range(1,N):
            F = Fy(w[n][m+1][j], w[n][m][j], w[n][m-1][j], X[n], T[j] + tau/2 )
            beta_x[n] = (F + Ax * beta_x[n-1]) / (Cx - Ax * alpha_x[n-1])
            alpha_x[n] = Bx / (Cx - Ax*alpha_x[n-1])
        # обратный ход прогонки
        w05[N][m] = 0
```

```
        for n in range(N-1,-1,-1):
            w05[n][m] = alpha_x[n]*w05[n+1][m]+beta_x[n]

    # переход на слой j+1
    for n in range(1,N):
        alpha_y[0] = 0
        beta_y[0] = 0
        # прямой ход прогонки
        for m in range(1, M):
            F = Fx(w05[n+1][m], w05[n][m],w05[n-1][m], X[n], T[j] + tau/2 )
            beta_y[m] = (F + Ay * beta_y[m- 1]) / (Cy - Ay * alpha_y[m - 1])
            alpha_y[m] = By / (Cy - Ay * alpha_y[m - 1])
        # обратный ход прогонки
        w[n][M][j+1] = 0
        for m in range(M-1,-1,-1):
            w[n][m][j+1] = alpha_y[m] * w[n][m+1][j+1] + beta_y[m]
            if j == J-1:
                err[n][m] = abs( analitic_sol[n][m] - w[n][m][j+1] )*100/max_u


for i in range(N+1):
    for j in range(M+1):
        w_for_plot[i][j] = w[i][j][J]


plot(X, Y, analitic_sol)
plot(X, Y, w_for_plot)
plot_error(X, Y, err)
```

4 Результат работы программы

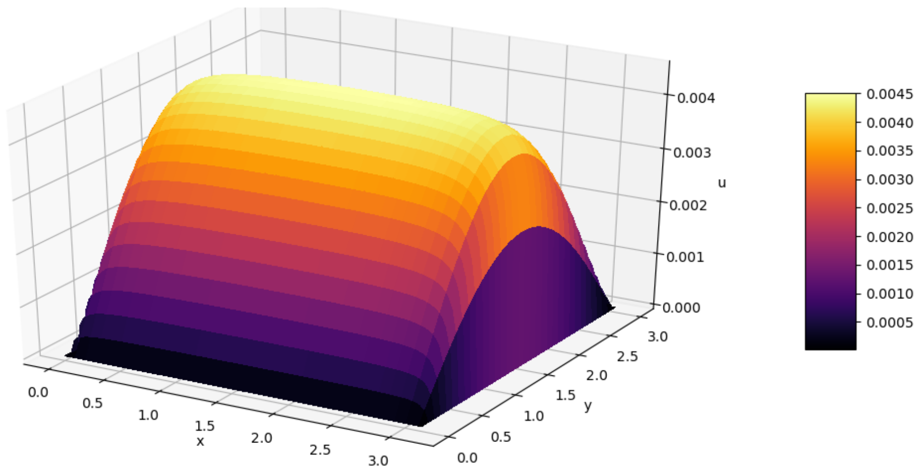

Рис. 1: аналитическое решение t = 0.1

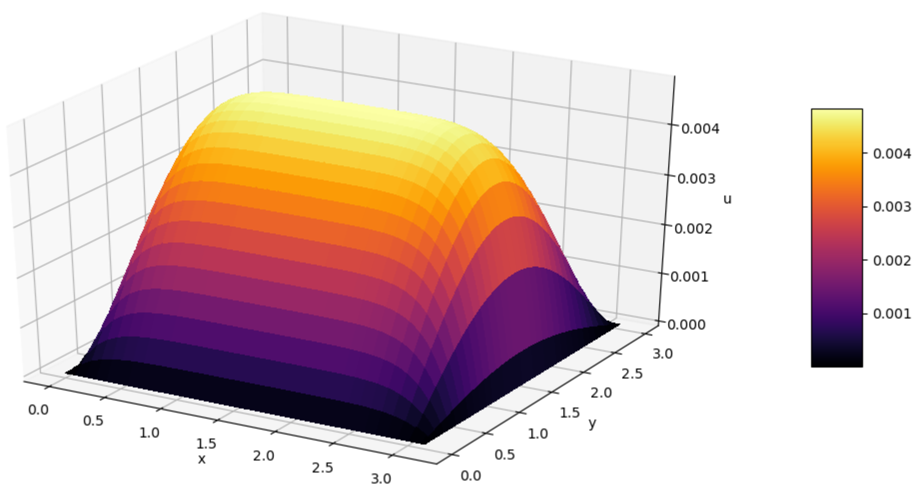

Рис. 2: численное решение t = 0.1

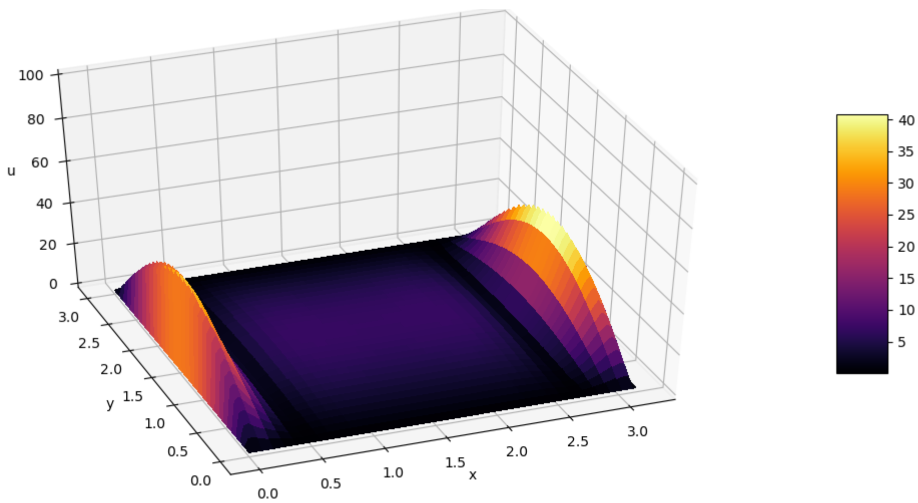

Рис. 3: ошибка t = 0.1

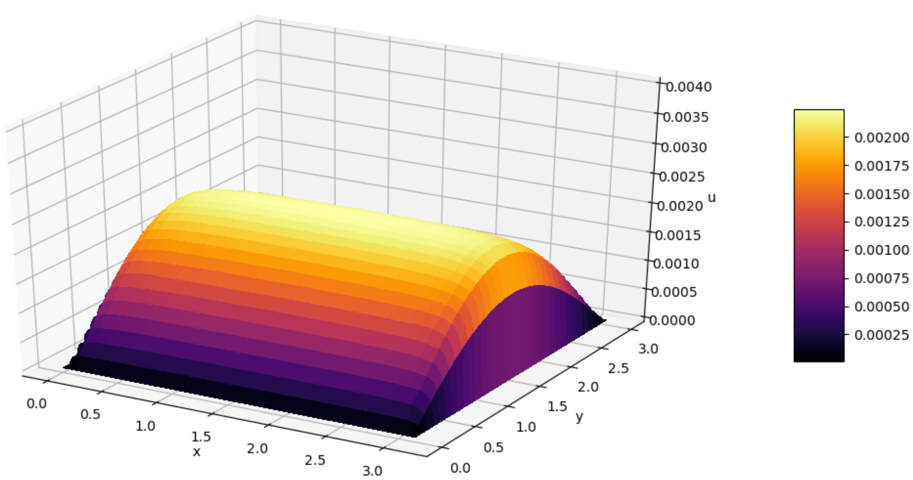

Рис. 4: аналитическое решение t = 0.007

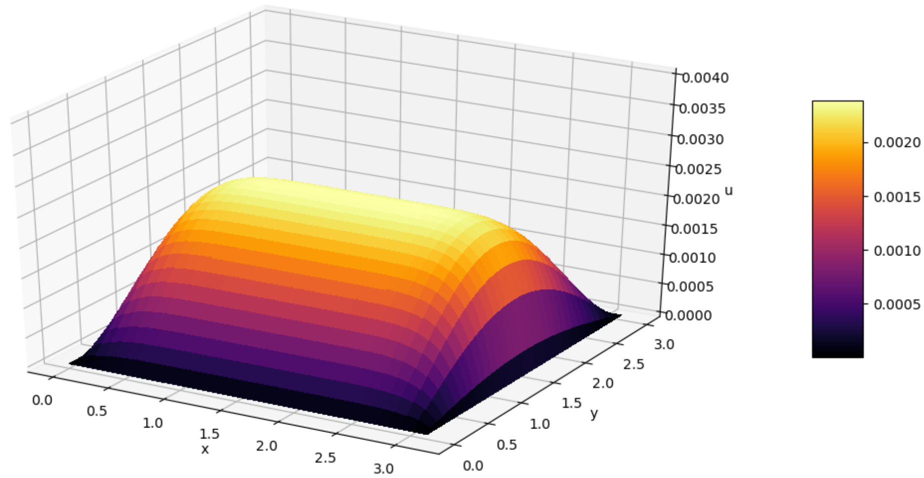

Рис. 5: численное решение t = 0.007

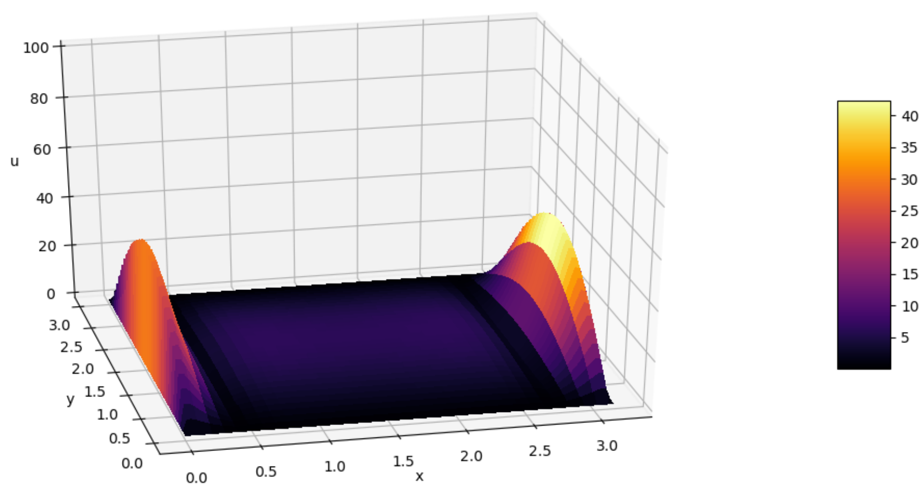

Рис. 6: ошибка t = 0.007