МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ОТЧЁТ ПО ВЫПОЛНЕНИЮ РГЗ**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Акулов

(подпись)

Направление подготовки 02.03.03 — «Математическое обеспечение\_\_\_\_\_\_\_\_

(код, наименование)

и администрирование информационных систем»\_\_\_\_\_\_\_\_\_\_ курс\_\_\_\_\_3\_\_\_\_\_

Направленность (профиль)\_\_\_\_ Технология программирования\_\_\_\_\_\_\_\_\_\_\_\_

Краснодар

2024

**Постановка задачи**

Задача заключается в численном решении уравнения теплопроводности явной и неявной разностной схемой решения:

с начальными и граничными условиями:

где,

T – выбирается пользователем при запуске программы.

Требуется определить устойчивость решений и сравнить полученные результаты аналитически и графически.

**Описание методов решения**

Для численного решения уравнения применяется дискретизация области [0,]×[0,T] с использованием пространственного шага Δx и временного шага Δt.

Пусть xi=iΔx, (i=0,1,…,Nx) и tn = nΔt (n=0,1,…,Nt), где Nx​ и Nt​ — число узлов по пространству и времени. Решение в узлах сетки приближается значениями uin ≈ u(xi,tn).

**Разностные схемы**

Разностная схема представляет собой систему линейных уравнений, решение которой осуществляется по слоям.

Явная схема основывается на формуле:

uin+1 = uin +r\*(ui−1n−2uin+ui+1n), r=

Граничные условия:

u0n = 0,

uNxn=e−tn

Начальные условия:

ui0=sin(xi),i=0,1,…,Nx.

Стоит отметить, что явная схема устойчива при 𝑟≤0.5. При нарушении этого условия численное решение становится неустойчивым.

Неявная схема решает следующую систему уравнений на каждом временном шаге:

−r\*ui−1n+1​+(1+2r)\*uin+1​−r\*ui+1n+1​=uin​, i=1,2,…,Nx​−1.

Неявная схема в матричной форме записывается следующим образом:

Aun+1=b,

где:

A — трёхдиагональная матрица, учитывающая коэффициенты r,

b — правая часть, зависящая от значений uin.

Граничные и начальные условия эквивалентны описанным выше.

Неявная схема абсолютно устойчива и позволяет использовать большие временные шаги Δt.

Обе схемы позволяют получить численное приближение решения уравнения теплопроводности, которое затем сравнивается с точным решением u(x,t) = e−t sin(x).

**Расчёты и числовые результаты**

Расчёты проводились с помощью программы, написанной на языке Python. Её код реализует решения явной схемой, неявной схемой. Реализованы функции для анализа и отображения графиков, полученных в результате работы алгоритмов. Код программы представлен в приложении А.

Числовые результаты программы представлены в таблице 1. В первой строке мелкая сетка, во второй – грубая, в третьем – проверка как в долговременном будет

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nx | Nt | T | Максимальная ошибка явной схемы | Максимальная ошибка неявной схемы |
| 20 | 500 | 1.0 | 6.45522e-05 | 2.00569e-04 |
| 10 | 10 | 0.01 | 1.44213e-05 | 2.34554e-05 |
| 5 | 105 | 4.0 | 1.45839e-03 | 3.49074e-03 |

Таблица 1 – результаты работы алгоритма

Анализ каждой строки таблицы в виде графика, на котором отображено сравнение решений разных схем решения предоставлен на рисунках 1-3.

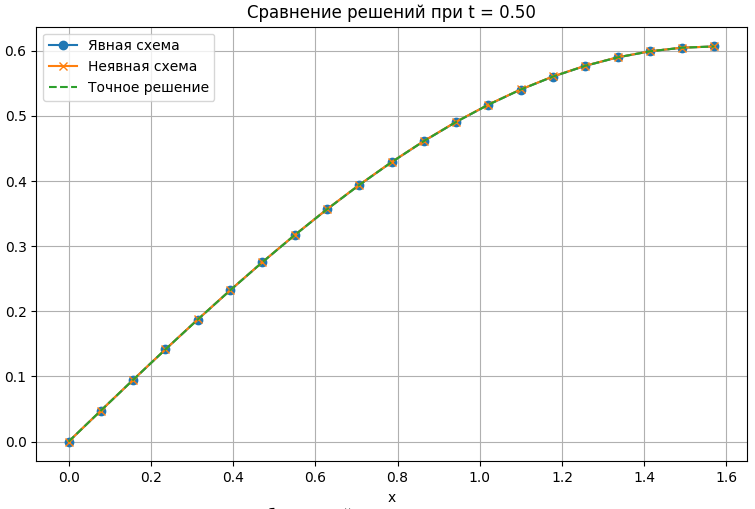


Рисунок 1 – сравнение решений в конкретной точке для мелкой сетки

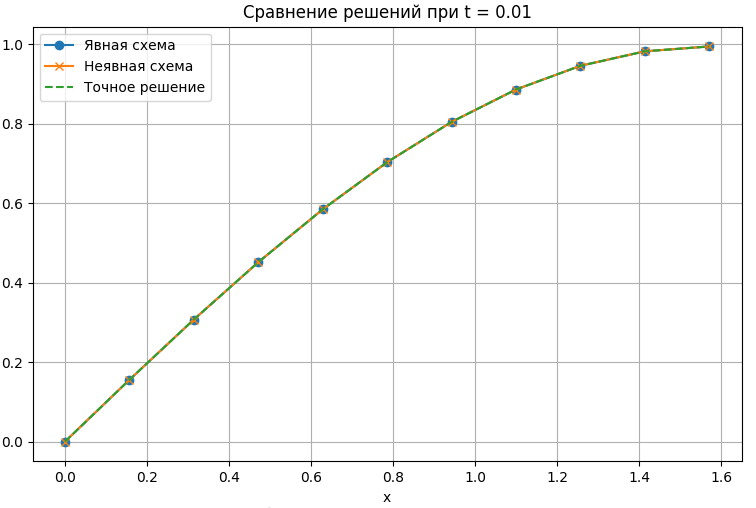


Рисунок 2 – сравнение решений в конкретной точке для грубой сетки

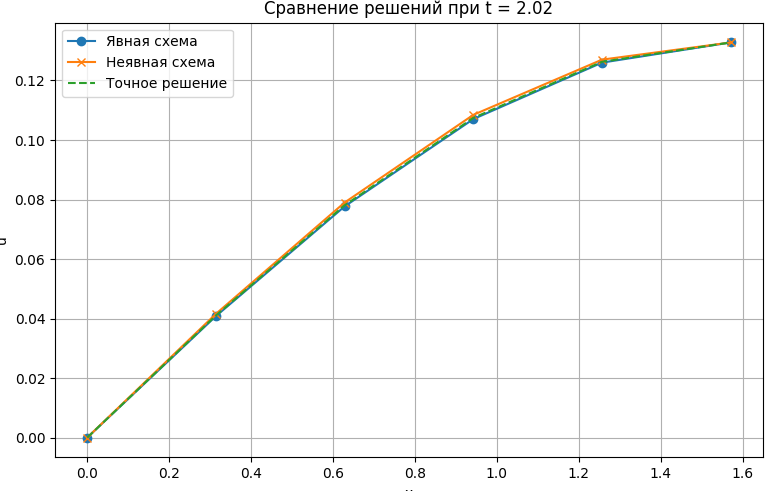


Рисунок 3 – сравнение решений в конкретной точке для долговременного случая

При мелкой сетке (Nx=20) и большом числе временных шагов (Nt=500), обе схемы показывают высокую точность. Ошибки минимальны, так как шаги Δx и Δt малы, обеспечивая соблюдение условий устойчивости для явной схемы.

При очень коротком времени моделирования (T=0.01) обе схемы дают практически одинаково точные результаты. Различия минимальны, так как число временных шагов (Nt=10) достаточно для стабилизации явной схемы, а сама задача решается для небольшого t.

При грубой пространственной сетке (Nx=5) и большом времени моделирования (T=4.0) обе схемы дают заметные ошибки. Это связано с тем, что шаг Δx становится слишком крупным, что ухудшает аппроксимацию второй производной. Неявная схема показывает большую ошибку из-за увеличенной роли накопления погрешностей при решении линейных систем на каждом шаге.

**Выводы**

При соблюдении условий устойчивости (r≤0.5) явная схема даёт высокую точность, как видно в тестах 1 и 2. Однако при грубой сетке и длительном времени моделирования (тест 3) ошибка возрастает, что обусловлено потерей точности аппроксимации.

Неявная схема демонстрирует стабильность при любых условиях. Тем не менее, её ошибка больше, чем у явной схемы, при одинаковых параметрах сетки. Это связано с особенностями решения трёхдиагональных систем, где метод даёт некоторую численную погрешность.

Явная схема рекомендуется для задач, где возможен мелкий шаг по времени (Δt) и требуется высокая точность. Неявная схема предпочтительна для моделирования на больших интервалах времени (T), так как она устойчива даже при крупных шагах по времени.

Увеличение числа узлов (Nx​) и временных шагов (Nt​) значительно снижает ошибки обеих схем. При коротких временных интервалах и мелкой сетке обе схемы дают сопоставимые результаты. При грубых сетках неявная схема остаётся стабильной, но демонстрирует бо́льшую погрешность из-за специфики алгоритма.

**Приложение А**

**Основной код программы**

import numpy as np  
import matplotlib.pyplot as plt  
  
def solve\_explicit(Nx, Nt, T):  
 *"""Явная схема."""* L = np.pi / 2  
 dx = L / Nx  
 dt = T / Nt  
  
 x = np.linspace(0, L, Nx + 1)  
 t = np.linspace(0, T, Nt + 1)  
  
 u = np.zeros((Nt + 1, Nx + 1))  
  
 u[0, :] = np.sin(x)  
  
 u[:, 0] = 0  
 u[:, -1] = np.exp(-t)  
  
 r = dt / dx \*\* 2  
 if r > 0.5:  
 raise ValueError("Явная схема неустойчива, уменьшите шаги dx или dt.")  
  
 for n in range(0, Nt):  
 for i in range(1, Nx):  
 u[n + 1, i] = u[n, i] + r \* (u[n, i - 1] - 2 \* u[n, i] + u[n, i + 1])  
  
 return x, t, u  
def solve\_implicit(Nx, Nt, T):  
 *"""Неявная схема."""* L = np.pi / 2  
 dx = L / Nx  
 dt = T / Nt  
  
 x = np.linspace(0, L, Nx + 1)  
 t = np.linspace(0, T, Nt + 1)  
  
 u = np.zeros((Nt + 1, Nx + 1))  
 u[0, :] = np.sin(x)  
 u[:, 0] = 0  
 u[:, -1] = np.exp(-t)  
  
 r = dt / dx \*\* 2  
 A = np.zeros((Nx - 1, Nx - 1))  
 for i in range(Nx - 1):  
 A[i, i] = 1 + 2 \* r  
 if i > 0:  
 A[i, i - 1] = -r  
 if i < Nx - 2:  
 A[i, i + 1] = -r  
  
 for n in range(0, Nt):  
 if Nx > 1: # Проверяем, чтобы уравнение имело внутренние узлы  
 b = u[n, 1:-1].copy()  
 b[0] += r \* u[n + 1, 0] # левая граница (всегда 0)  
 b[-1] += r \* np.exp(-t[n + 1]) # правая граница exp(-t[n+1])  
 u[n + 1, 1:-1] = np.linalg.solve(A, b)  
 else:  
 u[n + 1, :] = u[n, :] # Если Nx == 1, внутренние узлы отсутствуют  
  
 return x, t, u  
  
  
def exact\_solution(x, t):  
 *"""Точное решение."""* return np.exp(-t)[:, None] \* np.sin(x)  
  
  
def plot\_results(x, t, u\_exp, u\_imp, u\_exact):  
 *"""Построение графиков в разрезе по времени и пространству."""* plt.figure(figsize=(16, 12))  
 time\_idx = len(t) // 2  
 plt.subplot(2, 2, 1)  
 plt.plot(x, u\_exp[time\_idx, :], label='Явная схема', marker='o')  
 plt.plot(x, u\_imp[time\_idx, :], label='Неявная схема', marker='x')  
 plt.plot(x, u\_exact[time\_idx, :], label='Точное решение', linestyle='--')  
 plt.title(f"Сравнение решений при t = {t[time\_idx]:.2f}")  
 plt.xlabel("x")  
 plt.ylabel("u")  
 plt.legend()  
 plt.grid()  
 space\_idx = len(x) // 2  
 plt.subplot(2, 2, 2)  
 plt.plot(t, u\_exp[:, space\_idx], label='Явная схема', marker='o')  
 plt.plot(t, u\_imp[:, space\_idx], label='Неявная схема', marker='x')  
 plt.plot(t, u\_exact[:, space\_idx], label='Точное решение', linestyle='--')  
 plt.title(f"Эволюция решения в точке x = {x[space\_idx]:.2f}")  
 plt.xlabel("t")  
 plt.ylabel("u")  
 plt.legend()  
 plt.grid()  
  
 plt.subplot(2, 2, 3)  
 error\_exp = np.abs(u\_exp - u\_exact)  
 plt.imshow(error\_exp, extent=[0, np.pi / 2, 0, t[-1]], origin='lower', aspect='auto', cmap='viridis')  
 plt.colorbar(label='Ошибка')  
 plt.title("Ошибка явной схемы")  
 plt.xlabel("x")  
 plt.ylabel("t")  
  
 plt.subplot(2, 2, 4)  
 error\_imp = np.abs(u\_imp - u\_exact)  
 plt.imshow(error\_imp, extent=[0, np.pi / 2, 0, t[-1]], origin='lower', aspect='auto', cmap='viridis')  
 plt.colorbar(label='Ошибка')  
 plt.title("Ошибка неявной схемы")  
 plt.xlabel("x")  
 plt.ylabel("t")  
  
 plt.tight\_layout()  
 plt.show()  
  
  
def analyze\_results(u\_exp, u\_imp, u\_exact):  
 *"""Аналитический анализ ошибок."""* error\_exp = np.abs(u\_exp - u\_exact)  
 error\_imp = np.abs(u\_imp - u\_exact)  
  
 max\_error\_exp = np.max(error\_exp)  
 max\_error\_imp = np.max(error\_imp)  
  
 print("Анализ ошибок:")  
 print(f"Максимальная ошибка явной схемы: {max\_error\_exp:.5e}")  
 print(f"Максимальная ошибка неявной схемы: {max\_error\_imp:.5e}")  
  
 print("\nСредняя ошибка по времени и пространству:")  
 print(f"Явная схема: {np.mean(error\_exp):.5e}")  
 print(f"Неявная схема: {np.mean(error\_imp):.5e}")  
  
  
def main():  
 # Ввод параметров  
 while True:  
 try:  
 Nx = int(input("Введите число шагов по пространству Nx (целое число > 0): "))  
 Nt = int(input("Введите число шагов по времени Nt (целое число > 0): "))  
 T = float(input("Введите конечное время T (положительное число): "))  
 if Nx <= 0 or Nt <= 0 or T <= 0:  
 raise ValueError("Все значения должны быть положительными.")  
 break  
 except ValueError as e:  
 print(f"Ошибка ввода: {e}. Попробуйте снова.")  
  
 # Решение явной схемой  
 try:  
 x\_exp, t\_exp, u\_exp = solve\_explicit(Nx, Nt, T)  
 except ValueError as e:  
 print(e)  
 return  
  
 # Решение неявной схемой  
 x\_imp, t\_imp, u\_imp = solve\_implicit(Nx, Nt, T)  
  
 # Точное решение  
 x\_exact = np.linspace(0, np.pi / 2, Nx + 1)  
 t\_exact = np.linspace(0, T, Nt + 1)  
 u\_exact = exact\_solution(x\_exact, t\_exact)  
  
 # Построение графиков  
 plot\_results(x\_exact, t\_exact, u\_exp, u\_imp, u\_exact)  
  
 # Аналитический анализ  
 analyze\_results(u\_exp, u\_imp, u\_exact)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()