**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

Отчет по лабораторной работе №2

Численное решение уравнения теплопроводности

Преподаватель: Будник Анатолий Михайлович

Студент: Высоцкий Богдан Георгиевич

3 курс 5 группа

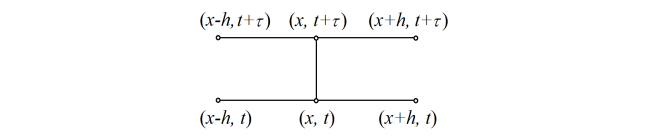
**Постановка задачи**

Для решения краевой задачи вида

на сетке построить разностную схему с весами при c погрешностью аппроксимации не ниже .

**Теория**

Для аппроксимации дифференциальной задачи выберем шеститочечный шаблон



и построим неявную схему с весами при . Неявная схема с весами является линейной комбинацией явной и чисто неявной схем. То есть

Рассмотрим погрешность аппроксимации первого уравнения:

Таким образом мы показали, что погрешность аппроксимации первого уравнения равна .

Второе и третье уравнения краевой задачи аппроксимируются точно, поэтому их погрешности равны нулю:

Рассмотрим аппроксимацию второго граничного условия:

При для аппроксимации производной по будем использовать левую разностную производную, т.е. . Однако при такой замене данное граничное условие аппроксимируется с первым порядком по . Будем пользоваться видом дифференциального уравнения для повышения порядка аппроксимации данного граничного условия:

Таким образом мы аппроксимировали второе граничное условие с порядком . В итоге мы приходим к следующему разностному аналогу второго граничного условия:

Таким образом, разностная схема принимает вид:

**Реализация**

Для реализации схемы нам понадобится двумерный массив , в который на каждой итерации по времени мы будем добавлять вычисленный временной слой.

Первым делом мы добавляем “нулевой” по времени слой (наше начальное условие) в массив. Последующие же временные слои () будут рассчитываться по следующим формулам:

1. При
2. При :
3. При :

Можно заметить, что на каждом временном шаге мы получаем СЛАУ с трехдиагональной матрицей, коэффициенты которой рассчитываются по следующим формулам:

поэтому на каждом временном шаге мы должны решать СЛАУ, сформированную этими коэффициентами. Для решения будем использовать метод разностной прогонки.

**Устойчивость**

Для обеспечения устойчивости необходимо выполнение условия , откуда следует, что . Поэтому при реализации схемы положим .

**Листинг**

def solve\_heat\_equation(h, t):

N1 = int(1 / h) # количество разбиений по h

N2 = int(1 / t) # количество разбиений по t

u = [] # массив для решения

u.append([phi(j \* h) for j in range(N1 + 1)]) # заполняем 1-ый слой

for i in range(N2):

A = [0] \* (N1 + 1)

B = [0] \* (N1 + 1)

C = [0] \* (N1 + 1)

F = [0] \* (N1 + 1)

# формируем матрицу системы для нахождения следующего слоя

B[0] = 1

C[0] = 0

F[0] = mu\_1(i \* t)

for j in range(1, N1):

A[j] = -t / (4 \* h\*\*2)

B[j] = 1 + t / (2 \* h\*\*2)

C[j] = -t / (4 \* h\*\*2)

F[j] = u[i][j] + (3 \* t / 4 \* h\*\*2) \* (u[i][j - 1] - 2 \* u[i][j] + u[i][j + 1]) + t \* f(j \* h, i \* t)

A[N1] = -1

B[N1] = 1 + h\*\*2 / (2 \* t)

F[N1] = h \* mu\_2(i \* t) + (h\*\*2 / (2 \* t)) \* u[i][N1] + (h\*\*2 / 2) \* f(N1 \* h, i \* t)

u.append(tridiagonal\_method(A, B, C, F, N1))

return u

**Результаты**

График “точного” решения (получено при )

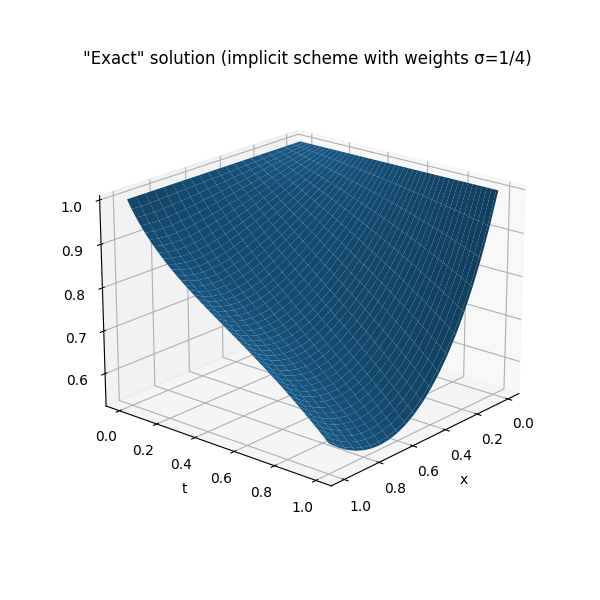
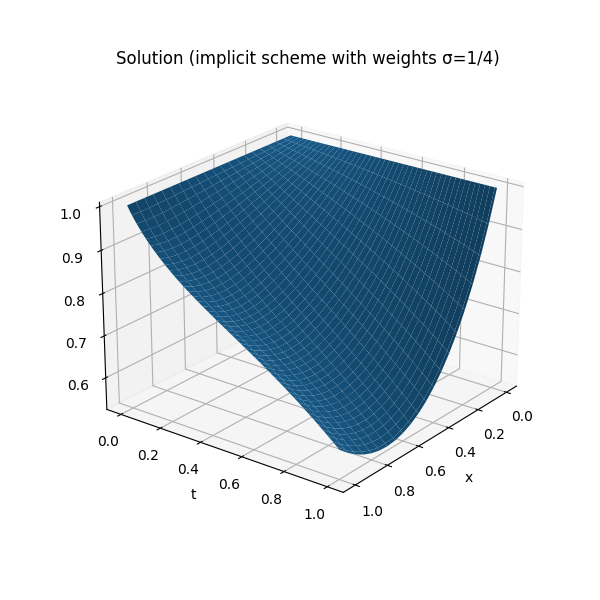


График решения (при )

**

Теперь проверим, влияет ли выбор шага на устойчивость схемы. Для этого немного возмутим каждое уравнение (прибавим 0.1) и увеличим временной интервал до 5 секунд. Получаем следующие результаты:

График “точного” решения (получено при ):

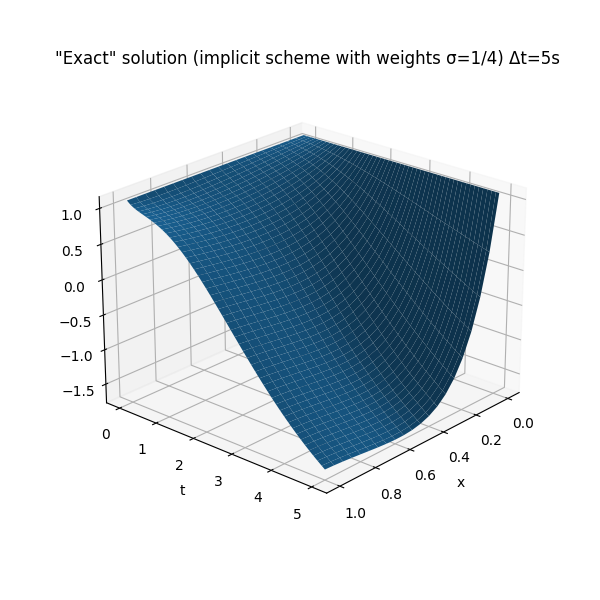


График решения (получено при ):

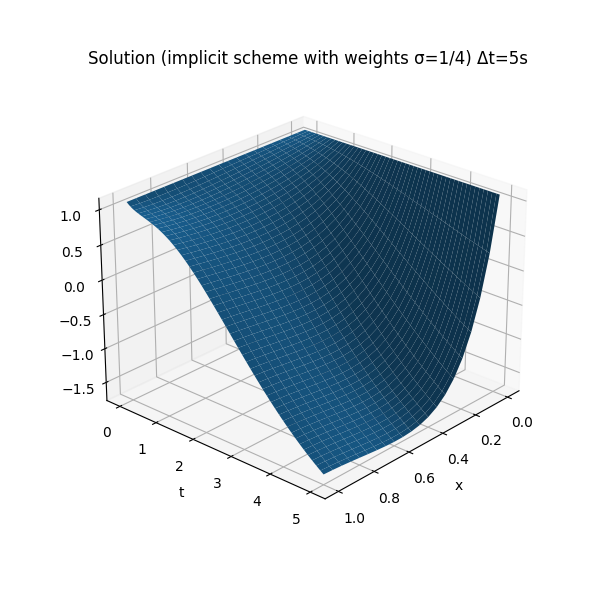


График решения (получено при ):

