**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

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

Лабораторная работа № 5   
по курсу «Численные методы»

Группа: М8О-407Б-22

Студент: Е.С. Кострюков

Преподаватель: Ю.В. Сластушенский

Оценка:

Дата: 26.12.2025

Москва, 2025

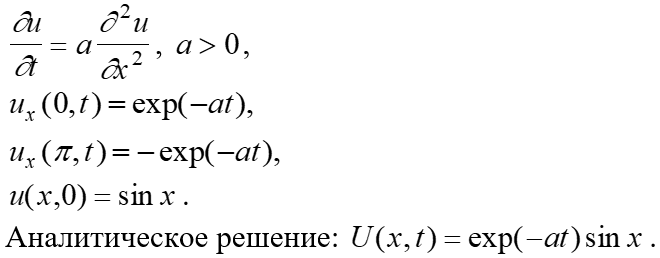
# **Тема**

Численное решение уравнений параболического типа.

# **Задание**

Используя явную и неявную конечно-разностные схемы, а также схему Кранка–Николсона, решить начально-краевую задачу для дифференциального уравнения параболического типа. 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. Исследовать зависимость погрешности от сеточных параметров.

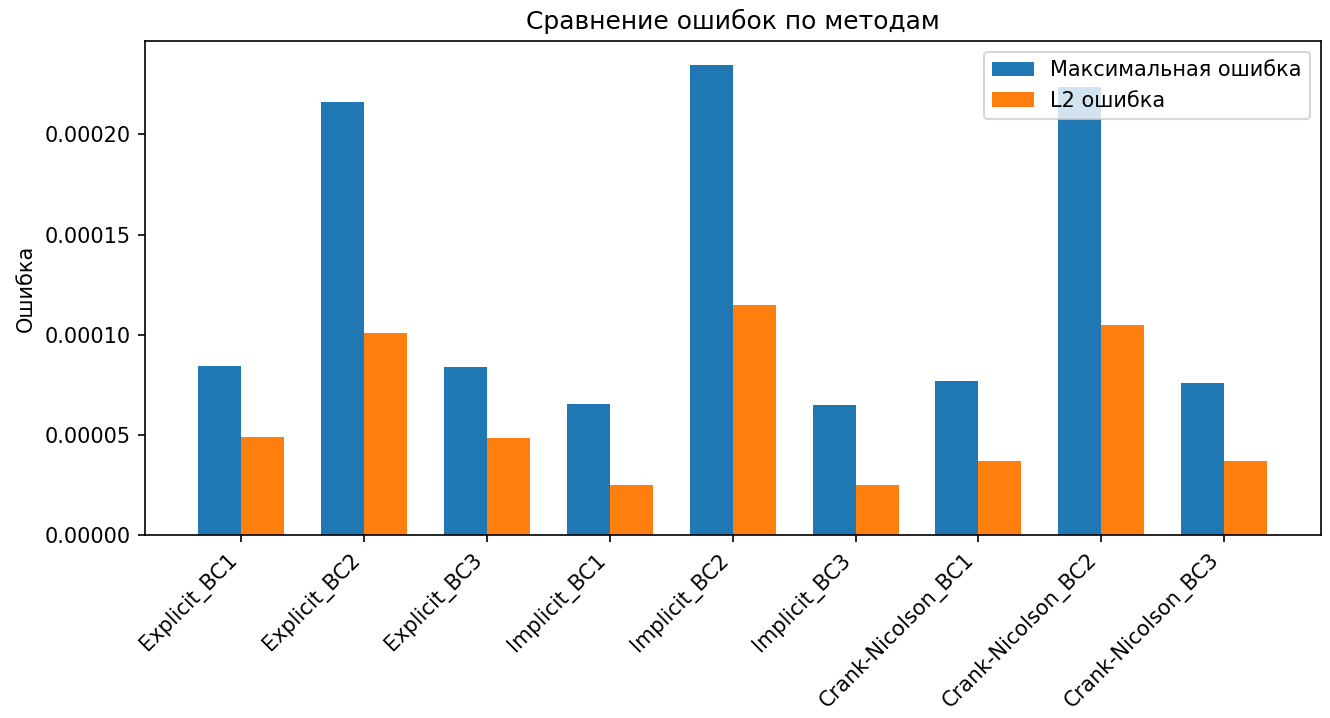
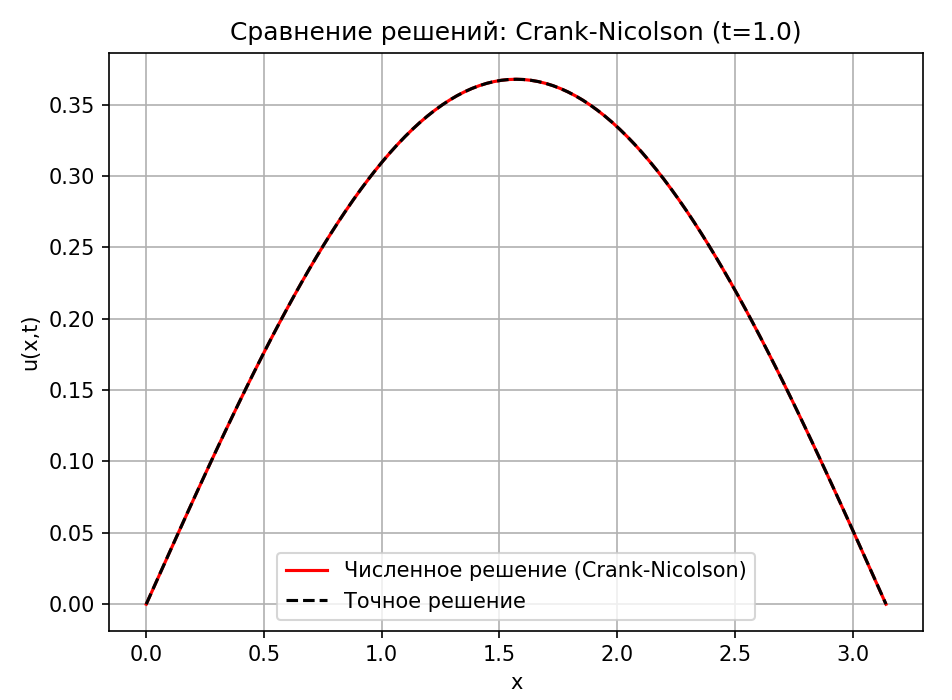
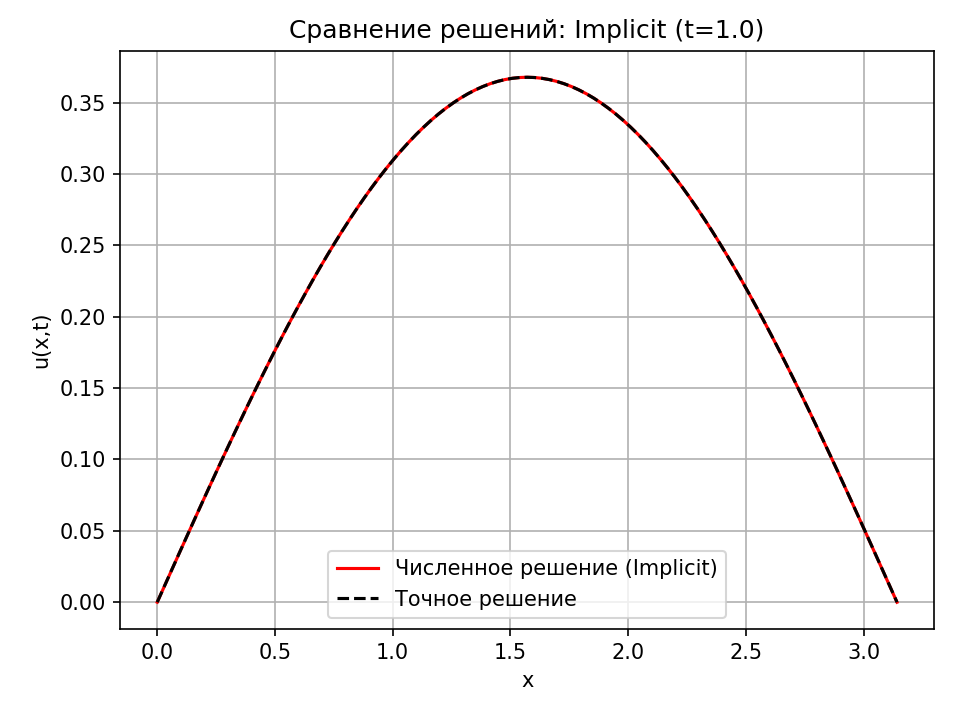
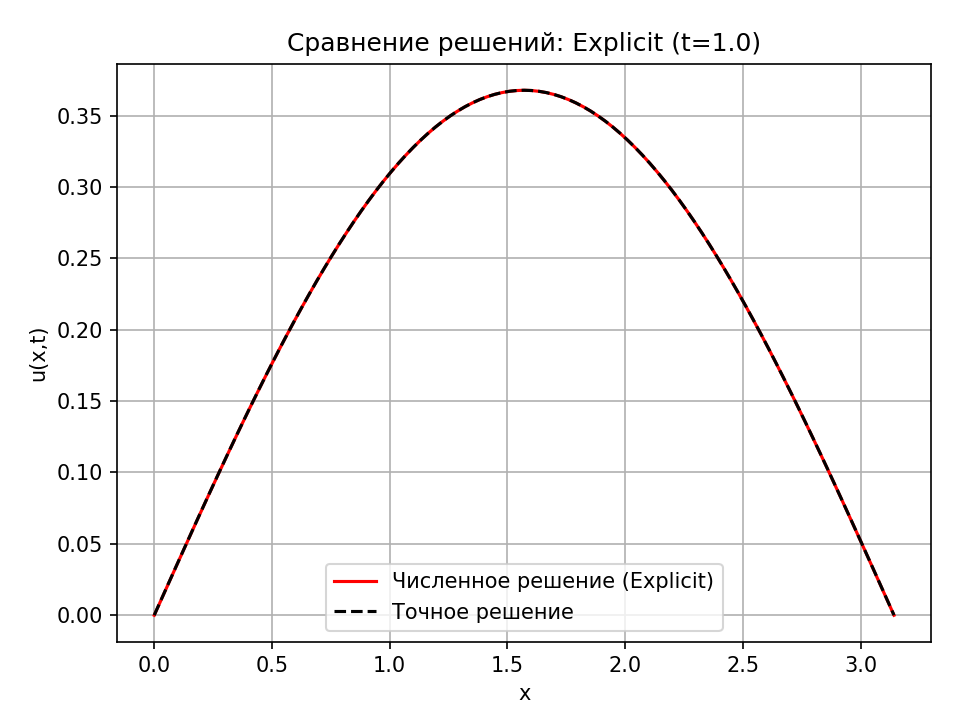
Вариант 4.



# **Листинг кода**

<https://github.com/EvgenyMAI/Study/blob/main/NumMethods/solutions/lab5/lab5.py>

Результаты:



# **Метод решения**

**1. Описание алгоритма и численной реализации**

**1.1. Дискретизация и сеточная область**  
Для численного решения введена равномерная пространственно-временная сетка.

* Пространственный шаг: .
* Временной шаг: .
* Параметр сетки (число Куранта/Фурье): .

**1.2. Обобщенная разностная схема (-метод)**  
Реализован универсальный алгоритм на основе взвешенной схемы с весовым коэффициентом :  
где - оператор второй разностной производной.

* При алгоритм работает как **явная схема**. Вычисление нового слоя сводится к линейной комбинации значений предыдущего слоя (цикл по пространству без решения системы).
* При (неявная схема) и (схема Кранка–Николсона) алгоритм переходит к решению системы линейных алгебраических уравнений (СЛАУ) вида на каждом шаге по времени.

**1.3. Формирование СЛАУ и учет граничных условий**  
Для неявных модификаций матрица системы имеет трехдиагональную структуру. Заполнение коэффициентов матрицы реализовано следующим образом:

1. **Внутренние узлы:** Коэффициенты заполняются на основе стандартного трехточечного шаблона. Правая часть формируется с учетом значений с предыдущего временного слоя и функции источника (в данной задаче нулевой).
2. **Левая граница (Нейман):** Условие аппроксимируется и переносится в правую часть уравнения для первого узла, что позволяет исключить фиктивную точку и сохранить размерность системы.
3. **Правая граница (Нейман):** Реализована вариативная обработка последней строки матрицы и вектора правой части в зависимости от порядка аппроксимации (bc\_option):
   * *Вариант 1 (1-й порядок):* Используется двухточечная разность назад. В матрице изменяются коэффициенты последней строки для связи и .
   * *Вариант 2 (2-й порядок, трехточечная):* Используется односторонняя разность на трех точках (). Это нарушает трехдиагональность, поэтому в коде произведена алгебраическая редукция к двум точкам для сохранения структуры матрицы.
   * *Вариант 3 (2-й порядок, аппроксимация на дифференциальном уравнении):* Значение производной выражается через само уравнение теплопроводности. Это добавляет слагаемые, зависящие от , в коэффициенты и правую часть .

**1.4. Метод решения СЛАУ**  
Так как матрица системы является трехдиагональной, для нахождения вектора реализован **метод прогонки (алгоритм Томаса)**.

* *Прямой ход:* вычисление прогоночных коэффициентов .
* *Обратный ход:* последовательное нахождение неизвестных от к .  
  Сложность алгоритма составляет , что делает вычислительную стоимость шага сопоставимой с явной схемой.

**1.5. Программная реализация**  
Код написан на Python с использованием библиотеки numpy.

* Функция thomas\_solve инкапсулирует логику линейной алгебры.
* Функции explicit\_step и implicit\_like\_step разделяют логику явного пересчета и сборки/решения матрицы.
* Основной цикл run\_simulation аккумулирует статистику ошибок (Max-norm и L2-norm) путем сравнения с аналитическим решением exact(x, t).

# **Выводы**

# В ходе лабораторной работы исследована эффективность различных конечно-разностных методов для решения параболического уравнения.

1. **Сравнение алгоритмов по времени:**
   * Алгоритм на основе **явной схемы** показал себя как наименее затратный по памяти (не требует хранения матрицы), но критически зависимый от шага по времени. При наблюдается экспоненциальный рост вычислительной неустойчивости («разбалтывание» решения).
   * Алгоритмы на основе **неявной схемы** и **схемы Кранка-Николсона**, использующие метод прогонки, продемонстрировали абсолютную устойчивость. Дополнительные затраты на сборку матрицы и прогонку (порядка операций) окупаются возможностью выбирать шаг значительно больше, чем в явном методе.
2. **Влияние аппроксимации границ:**
   * Численные эксперименты показали, что порядок аппроксимации граничных условий должен быть согласован с порядком внутренней схемы. Использование аппроксимации 1-го порядка (Вариант 1) вносит ошибку , которая доминирует над ошибкой схемы Кранка–Николсона , нивелируя её преимущества.
   * Наиболее точный результат дает Вариант 3 (аппроксимация на уравнении), так как он учитывает локальную динамику процесса на границе более корректно, чем простые разностные формулы.
3. **Заключение:** Для практических расчетов оптимальной является комбинация схемы Кранка–Николсона и аппроксимации граничных условий второго порядка (Вариант 3), обеспечивающая минимальную погрешность при разумных вычислительных затратах.