**Министерство образования Республики Беларусь**

**Белорусский Государственный Университет**

**Физический факультет**

**Лабораторная работа №1**

**«Прямые методы решения СЛАУ*»***

***Выполнил:***

студент 2 курса, 4 группы, физического факультета БГУ

Мельников Владислав Сергеевич

Минск, 2021

**Цель:**

Реализовать алгоритм метода Гаусса решения СЛАУ. Для проверки правильности работы вывести на экран или в файл получившуюся верхнюю треугольную матрицу, решение и невязку. Реализовать метод Гаусса в модификации «Постолбцовый выбор главного элемента».

Придумать подходящие системы с различным числом уравнений. Построить зависимость времени нахождения решения от числа уравнений. Проверить асимптотику O(n^3). Реализовать на выбор одно из следующего:

* Написать функции, вычисляющие определитель матрицы, обратную матрицу и функцию, реализующую алгоритм прогонки.
* Написать класс рациональных чисел, перегрузить арифметические операторы и точно решить СЛАУ с рациональными коэффициентами.

**Содержимое проекта:**

***GaussMethod.h –*** набор необходимых объявлений для класса GaussMethod

***GaussMethod.cpp –*** реализация основной логики программы

***Source.cpp –*** содержит точку входа в программу

**Вывод результата:**

***slae.txt* –** исходное СЛАУ

***upper\_triangle\_matrix*** – СЛАУ в верхнем треугольном виде

***result.txt*** – найденные неизвестные

***discrepancy.txt*** – невязка

***complexity.txt, dimension.txt*** – данные для построения зависимости времени выполнения метода от кол-ва уравнений

**Основные функции:**

Реализация метода Гаусса, гдеa - матрица коэффициентов, y - массив свободных членов, n - размерность матрицы

double\* gauss(double\*\* a, double\* y, int n);

Нахождение невязки, гдеa - матрица коэффициентов, b - массив свободных членов, x – массив найденных неизвестных, n - размерность матрицы

double discrepancy(double\*\* a, double\* b, double\* x, int n);

Нахождение обратной матрицы, где matrix – исходная матрица, n – размерность матрицы

double\*\* reverseMatrix(double\*\* matrix, int n);

Нахождение определителя матрицы, где matrix – исходная матрица, n – размерность матрицы, epsilon - погрешность

double determinant(double\*\* matrix, int n, double epsilon);

Метод прогонки, принимает "тройную" диагональную матрицу. v1 -нижняя диагональ, v2 - диагональ по середине, v3 - верхняя диагональ, d - то, чему равняется кажая строчка этой матрицы

void sweepMethod(double\* v1, double\* v2, double\* v3, double\* d, double\* result, int n);

Нахождение зависимости Времени выполнения метода Гаусса от кол-ва уравнений

std::map<int, double> checkAsymptotics(int count);

**Вспомогательный функции:**

Считывание размерности матрицы коэффициентов из файла

int dimensionOfMatrixFromFile(std::string slaepath);

Считывание СЛАУ из файла

double\*\* slaeFromFile(std::string slaepath);

Извлечение из СЛАУ nxn+1 матрицы коэффициентов nxn

double\*\* aFromSlae(double\*\* slae, int n);

Извлечение из СЛАУ массива свободных членов n

double\* bFromSlae(double\*\* slae, int n);

Генерация квадратной матрицы случайных элементов

double\*\* randMatrixA(int n);

Генерация массива случайных элементов

double\* randMatrixB(int n);

Умножение матрицы размерностью nxn на nx1

double\* multMatrix(double\*\* a, double\* b, int n);

Клонирование квадратной матрицы

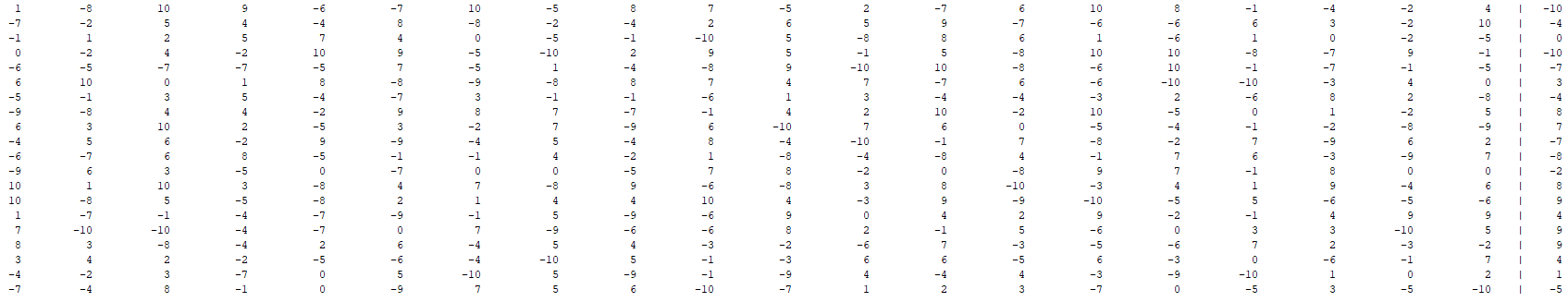
double\*\* matrixClone(double\*\* matrix, int n);

Клонирование массива

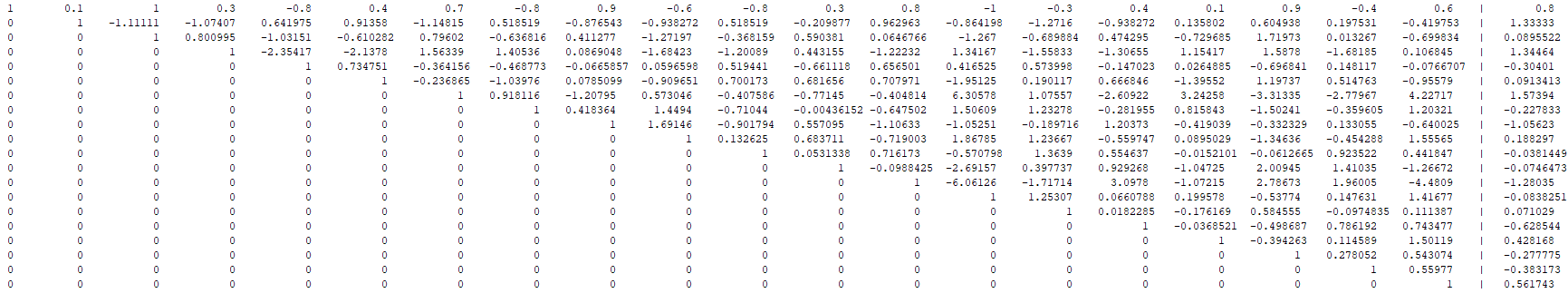
double\* arrayClone(double\* array, int n);

**Ход работы:**

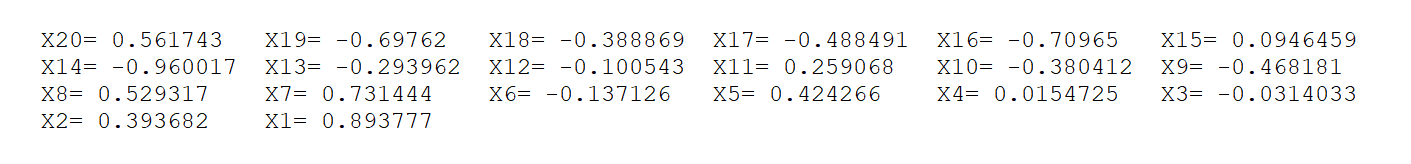
Матрица до преобразования:

****

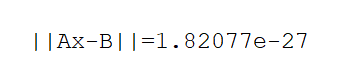
Верхняя треугольная матрица:

****

Неизвестные:



Невязка:



***Замечание:*** *в теле метода гаусса во время построения графика вывод результата и пр. не нужные вещи были закомментированы.*