

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования  
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ имени Н.Э.БАУМАНА  
(национальный исследовательский университет)»**

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

**Лабораторная работа № 3**

«Реализация метода Гаусса с перестановками»

по дисциплине «Численные методы линейной алгебры»

Работу выполнил

студент группы ИУ9-72Б

Жук Дмитрий

**Цель работы**

Целью данной работы является реализовать три варианта метода Гаусса с перестановками и научиться оценивать погрешности решения системы уравнения для матриц произвольной размерности.

**Реализация**

Был создан репозиторий на GitHub <https://github.com/ZhukDmitryOlegovich/num-methods>

Расширена логика библиотек, реализованных ранее, а именно у eliminationGaussian, которая является абстракцией для решения методом Гаусса, а именно добавлен дополнительный параметр отвечающий за то какой метод использовать: перестановки по столбцам, по строкам, по столбцам и строкам.

За основу бралась реализация предыдущей лабораторной работы дабы упростить логику и вычисления.

Для само тестирования – вызывались все 4 реализации и сравнивались они непосредственно, их погрешность и правильность конечных значений.

**Вывод**

В ходе выполнения лабораторной работы была изучена проблема использования метода Гаусса, а именно – способы устранения погрешностей.

Данную работу так же можно открыть и посмотреть по ссылке <https://zhukdmitryolegovich.github.io/num-methods/lab3/>.

**Приложение**

    eliminationGaussian(

        other: Vector<A>,

        option: { smartColon?: boolean; smartRow?: boolean; } = {},

    ) {

        const b = other.clone();

        const a = this.clone();

        const n = b.countRows();

        const left = fromLength(n, (i) => i);

        const right = fromLength(n, (i) => i);

        // Прямой ход

        for (let k = 0; k < n - 1; k++) {

            if (option.smartRow || option.smartColon) {

                let max1 = k;

                for (let j = k + 1; j < n && option.smartRow; j++) {

                    if (

                        Math.abs(a.matrix[left[max1]][right[k]])

                        < Math.abs(a.matrix[left[j]][right[k]])

                    ) {

                        max1 = j;

                    }

                }

                let max2 = k;

                for (let j = k + 1; j < n && option.smartColon; j++) {

                    if (

                        Math.abs(a.matrix[left[k]][right[max2]])

                        < Math.abs(a.matrix[left[k]][right[j]])

                    ) {

                        max2 = j;

                    }

                }

                if (

                    Math.abs(a.matrix[left[max1]][right[k]])

                    < Math.abs(a.matrix[left[k]][right[max2]])

                ) {

                    [right[max2], right[k]] = [right[k], right[max2]];

                } else {

                    [left[max1], left[k]] = [left[k], left[max1]];

                }

            }

            for (let j = k + 1; j < n; j++) {

                const m = a.matrix[left[k]][right[j]] / a.matrix[left[k]][right[k]];

                for (let i = 0; i < n; i++) {

                    a.matrix[left[i]][right[j]] -= m \* a.matrix[left[i]][right[k]];

                }

                b.matrix[0][right[j]] -= m \* b.matrix[0][right[k]];

            }

        }

        const x = new Vector(fromLength(n, () => 0));

        // Обратный ход

        for (let i = n - 1; i >= 0; i--) {

            x.matrix[0][left[i]] = b.matrix[0][right[i]] / a.matrix[left[i]][right[i]];

            for (let c = n - 1; c > i; c--) {

                x.matrix[0][left[i]] -= a.matrix[left[c]][right[i]]

                    \* x.matrix[0][left[c]] / a.matrix[left[i]][right[i]];

            }

        }

        return x;

    }