Федеральное государственное автономное
образовательное учреждение высшего образования

Университет ИТМО

Лабораторная работа №1

Курса “Вычислительная математика”

Вариант 8

Выполнил:
Попов Дмитрий Юрьевич
Группа: P3213
Преподаватель:
Машина Екатерина Алексеевна

2024 г.

Цель работы:

Реализовать и протестировать программу для решения системы линейных уравнений методом Гаусса, размерность до 20 включительно неизвестных.

Описание метода, расчетные формулы:

Суть метода заключается в преобразования расширенной СЛАУ к треугольному виду и последующему нахождению всех неизвестных. Если матрица квадратная и она имеет определитель, не равный нулю, то мы имеем единственное решение. Далее мы находим все неизвестные начиная с последней строки. Каждая неизвестная выражается через предыдущие, а последняя известна сразу.

$$\left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} & b_n \end{array}\right) \sim \left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ 0 & a_{2,2} - \frac{a_{2,1} * a_{1,2}}{a_{1,1}} & \cdots & a_{2,n} - \frac{a_{2,1} * a_{1,n}}{a_{1,1}} & b_2 - \frac{b_1 * a_{2,1}}{a_{1,1}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n,2} - \frac{a_{n,1} * a_{1,2}}{a_{1,1}} & \cdots & a_{n,n} - \frac{a_{n,1} * a_{1,n}}{a_{1,1}} & b_n - \frac{b_1 * a_{n,1}}{a_{1,1}} \end{array}\right) \sim$$

$$\sim \left(\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \cdots & a_{1,n} & b_1 \\ 0 & a_{2,2} & \cdots & a_{2,n} & b_2' \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{n,n} & b_n \end{array}\right)$$

(к треугольному виду)

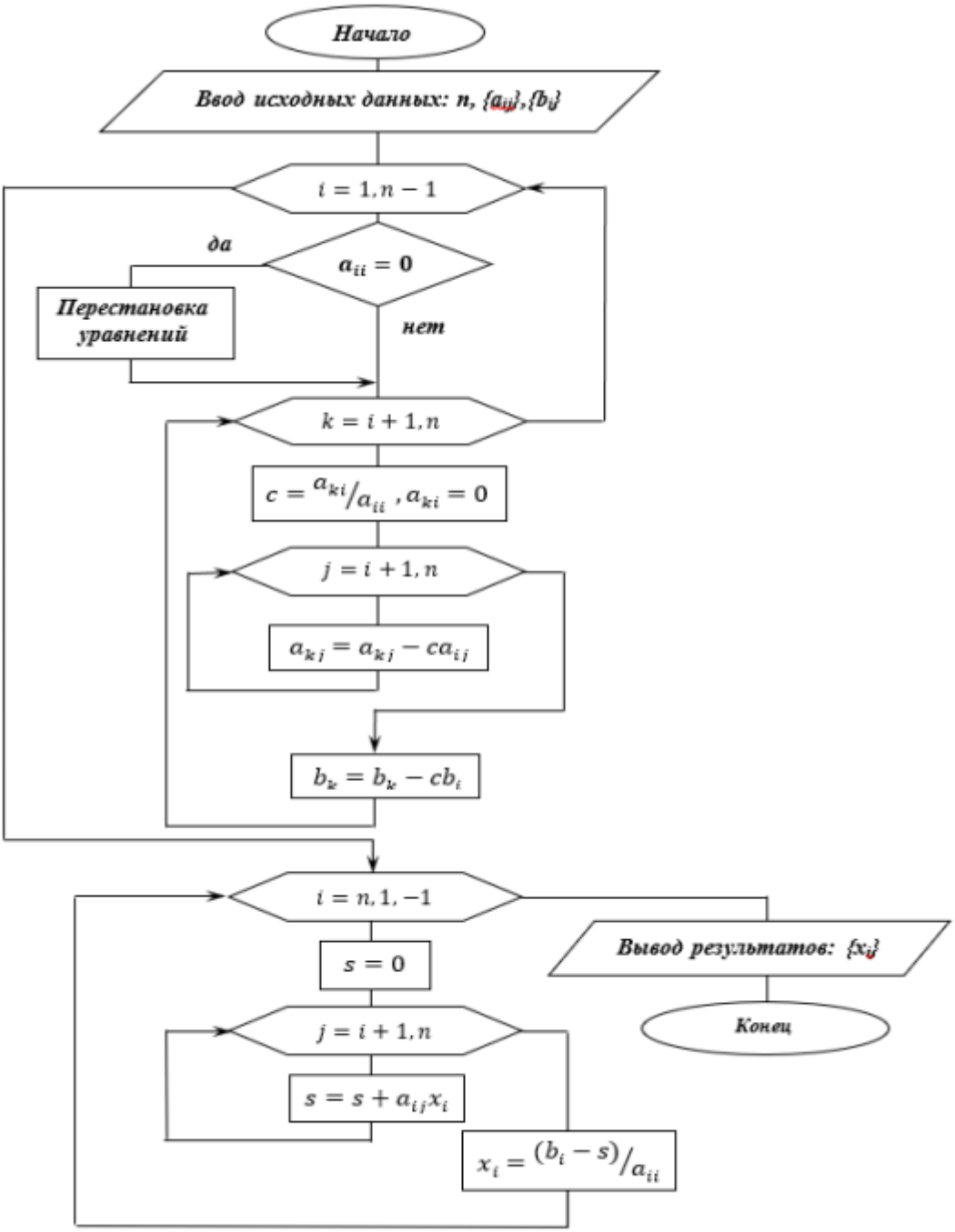
$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{i,j} * x_j}{a_{i,i}}$$

(решения)

$$detA = (-1)^k \prod_{i=1}^{n} a_{ii}$$

(определитель)

Блок-схема для метода Гаусса:



Код реализации решения на JavaScript:

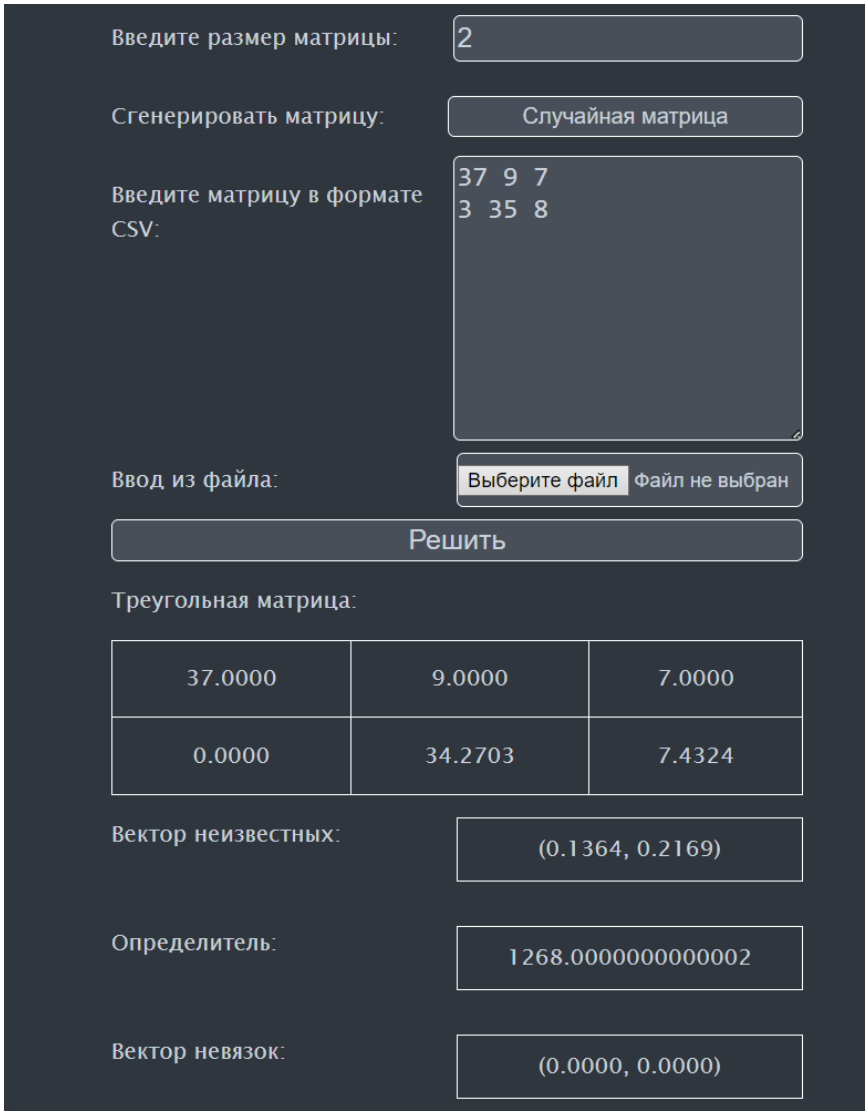
```
/**
 * Получить решение СЛАУ с помощью прямого метода Гаусса.
 */
export function solveLinearEquationsSystem(data) {
    forward(data);

    const solution = back(data);
    const residualsVectors= residuals(data)
    const det = determinant(data);
    const tri = JSON.parse(JSON.stringify(data.matrix));
    tri.map((row,index) => {
```

```
        row.push(data.rightValues[index])
    })
    return new Result(tri,det,solution,residualsVectors);
}

/** Если в процессе вычисления a11, a22, a33, ... = 0 - нужно переставить
соответственно коэф. */
const swapRows = (i, j,data) =>{
    let tempRow = data.matrix[i];
    data.matrix[i] = data.matrix[j];
    data.matrix[j] = tempRow;

    let tempVal = data.rightValues[i];
    data.rightValues[i] = data.rightValues[j];
    data.rightValues[j] = tempVal;
}
/** Прямой ход метода Гаусса */
const forward = (data) =>{
    for (let i = 0; i < data.size; i++) {
        // Проверяем, является ли диагональный элемент очень близким к нулю
        if (data.matrix[i][i] === 0) {
            // Переставляем текущее уравнение с другим уравнением, где
диагональный элемент не равен нулю
            let found = false;
            for (let k = i + 1; k < data.size; k++) {
                if (Math.abs(data.matrix[k][i]) > 1e-10) {
                    swapRows(i, k);
                    found = true;
                    break;
                }
            }
            if (!found) {
                throw new Error('Метод Гаусса не применим');
            }
        }

        for (let j = i + 1; j < data.size; j++) {
            let ratio = data.matrix[j][i] / data.matrix[i][i];
            for (let k = i; k < data.size; k++) {
                data.matrix[j][k] -= ratio * data.matrix[i][k];
            }
            data.rightValues[j] -= ratio * data.rightValues[i];
        }
    }
}
/** Вычисление определителя матрицы */
const determinant = (data) => {
    let det = 1;
    try {
        forward(data.matrix,data.rightValues);
        for (let i = 0; i < data.size; i++) {
            det *= data.matrix[i][i];
        }
        return det;
    } catch (error) {
```

```
        console.log(error.message);
        return null;
    }
}
/** Обратных ход метода Гаусса */
const back = (data) => {
    let solution = [];
    for (let i = data.size - 1; i >= 0; i--) {
        let sum = 0;
        for (let j = i + 1; j < data.size; j++) {
            sum += data.matrix[i][j] * solution[j];
        }
        solution[i] = (data.rightValues[i] - sum) / data.matrix[i][i];
    }
    return solution;
}
/** Вычисление веторов невязок */
const residuals = (data) => {
    const residuals = [];
    for (let i = 0; i < data.size; i++) {
        let res = -data.rightValues[i];
        for (let j = 0; j < data.size; j++) {
            res += data.matrix[i][j] * back(data)[j];
        }
        residuals.push(res);
    }
    return residuals;
}
```

Примеры и результаты работы программы:

Решатель Системы Уравнений методом Гаусса!



Вывод:

Метод Гаусса подходит для простого вычисления СЛАУ, так как он более универсален и прост в реализации, а также работает за конечное число арифметических операций. Но и у него есть недостатки так как весь массив приходится хранить в оперативной памяти компьютера. Происходит накопление погрешности в процессе решения, поскольку на любом этапе используют результаты предыдущих операций (решить проблему можно с помощью округлений, но и у этого есть свои недостатки).