МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

Отчет по практике

Программная реализация численного метода *Решение системы линейных алгебраических уравнений методом Гаусса*

1 курс, группа 1ИВТ АСОИУ

Выполнил:
________________ В. А. Косицкий
«___» _________ 2024 г.

Руководитель:
________________ С. В. Теплоухов
«___» _________ 2024 г.

Майкоп, 2024 г.

1. Введение

1.1. Текстовая формулировка задачи (Вариант 3)

Написать программу для решения системы линейных алгебраических уравнений методом Гаусса.

1.2. Теория метода

Метод Гаусса — классический метод решения системы линейных алгебраических уравнений (Рис. 1). Это метод последовательного исключения переменных, когда с помощью элементарных преобразований система уравнений приводится к равносильной системе треугольного вида, из которой последовательно, начиная с последних (по номеру), находятся все переменные системы.

Алгоритм решения СЛАУ методом Гаусса подразделяется на два этапа.

На первом этапе осуществляется так называемый прямой ход, когда путём элементарных преобразований над строками систему приводят к ступенчатой или треугольной форме, либо устанавливают, что система несовместна. А именно, среди элементов первого столбца матрицы выбирают ненулевой, перемещают его на крайнее верхнее положение перестановкой строк и вычитают получившуюся после перестановки первую строку из остальных строк, домножив её на величину, равную отношению первого элемента каждой из этих строк к первому элементу первой строки, обнуляя тем самым столбец под ним. После того, как указанные преобразования были совершены, первую строку и первый столбец мысленно вычёркивают и продолжают пока не останется матрица нулевого размера. Если на какой-то из итераций среди элементов первого столбца не нашёлся ненулевой, то переходят к следующему столбцу и проделывают аналогичную операцию. На втором этапе осуществляется так называемый обратный ход, суть которого заключается в том, чтобы выразить все получившиеся базисные переменные через небазисные и построить фундаментальную систему решений, либо, если все переменные являются базисными, то выразить в численном виде единственное решение системы линейных уравнений. Эта процедура начинается с последнего уравнения, из которого выражают соответствующую базисную переменную (а она там всего одна) и подставляют в предыдущие уравнения, и так далее, поднимаясь по «ступенькам» наверх. Каждой строчке соответствует ровно одна базисная переменная, поэтому на каждом шаге, кроме последнего (самого верхнего), ситуация в точности повторяет случай последней строки.

В простейшем случае алгоритм сформулирован так:

Из последнего ненулевого уравнения выражаем базисную переменную через небазисные и подставляем в предыдущие уравнения. Повторяя эту процедуру для всех базисных переменных, получаем фундаментальное решение (Рис. 2).

$$
\begin{cases}
a_{11} \cdot x_1 + a_{12} \cdot x_1 + \dots + a_{1n} \cdot x_n & (1) \\
a_{21} \cdot x_1 + a_{22} \cdot x_1 + \dots + a_{2n} \cdot x_n & (2) \\
\dots & \\
a_{m1} \cdot x_1 + a_{m2} \cdot x_1 + \dots + a_{mn} \cdot x_n & (m)
\end{cases}
$$

Рис. 1. Общий вид системы линейных алгебраических уравнений

$$
\begin{array}{rcc}
(2) \to (2) - (1) \cdot (\frac{a_{21}}{a_{11}}) & : & a'_{22} \cdot x_2 + a'_{23} \cdot x_3 + \dots + a'_{2n} \cdot x_n = b'_2 \\
(3) \to (3) - (1) \cdot (\frac{a_{31}}{a_{11}}) & : & a'_{32} \cdot x_2 + a'_{33} \cdot x_3 + \dots + a'_{3n} \cdot x_n = b'_3 \\
\dots & & \\
(m) \to (m) - (1) \cdot (\frac{a_{m1}}{a_{11}}) & : & a'_{m2} \cdot x_2 + a'_{m3} \cdot x_3 + \dots + a'_{mn} \cdot x_n = b'_n \\
(3) \to (3) - (2) \cdot (\frac{a'_{32}}{a'_{22}}) & : & a''_{33} \cdot x_3 + \dots + a''_{3n} \cdot x_n = b''_3 \\
\dots & & \\
(m) \to (m) - (m-1) \cdot (\frac{a^{(m-2)}_{m,n-1}}{a^{m-2}_{m-1,n-1}}) & : & a^{(m-1)}_{mm} \cdot x_m + \dots + a^{(m-1)}_{mn} \cdot x_n = b^{(m-1)}_m
\end{array}
$$

Рис. 2. Алгоритм в общем виде

2. Ход работы

2.1. Выбор средств для разработки

Для разработки программы для решения системы линейных алгебраических уравнений методом Гаусса мной был выбран язык программирования TypeScript и фреймворк для разработки Web-приложений Angular 17. Данный стек был выбран с целью обеспечения совместимости со всеми основными операционными системами как на ПК (macOS, Windows и ОС на ядре Linux), так и с мобильными платформами (iOS и Android). Использование TypeScript позволяет организовать типобезопасность приложения, а Angular позволяет использовать привязки данных и работу по паттерну MVVM.

2.2. Код приложения

Для упрощения разработки было создано несколько вспомогательных классов. Листинг кода для этих классов приведён ниже:

Листинг 1. Код класса Matrix для инкапсуляции работы с матрицами

```
export class Matrix {
  private _matrix: number[][];

  constructor(rows: number, cols: number, fillValue: number = 0) {
```

```
        this._matrix = Array.from({ length: rows }, () =>
            Array(cols).fill(fillValue));
    }

    get value(): number[][] {
        return this._matrix;
    }

    set value(value: number[][]) {
        this._matrix = value;
    }

    resize(rows: number, cols: number, fillValue: number = 0): void {
        const newData = Array.from({ length: rows }, (_, rowIndex) =>
            Array.from({ length: cols }, (_, colIndex) =>
                this._matrix[rowIndex]?.[colIndex] ?? fillValue)
        );
        this._matrix = newData;
    }

    get rows(): number {
        return this._matrix.length;
    }

    get cols(): number {
        return this._matrix[0]?.length ?? 0;
    }

    setValue(row: number, col: number, value: number): void {
        if (row >= 0 && row < this.rows && col >= 0 && col < this.cols) {
            this._matrix[row][col] = value;
        }
    }

    getValue(row: number, col: number): number {
        if (row >= 0 && row < this.rows && col >= 0 && col < this.cols) {
            return this._matrix[row][col];
        }
        return Number.NaN;
    }
}
```

Листинг 2. Код класса GaussianCalculator для инкапсуляции вычисления корней СЛАУ методом Гаусса

```
import { Matrix } from "../_types/matrix";
```

```
export class GaussianCalculator {

    validateMatrix(matrix: Matrix): boolean {
        return matrix.value.length + 1 == matrix.value[0].length
    }

    recalculateRows(row1: number[], row2: number[], id: number): number[] {
        let result: number[] = []

        for (let i = 0; i < row1.length; i++) {
            let row1Res = row1[id] * row2[i]
            let row2Res = row2[id] * row1[i]
            result.push(row1Res - row2Res)
        }

        return result
    }

    resolveMatrixResults(matrix: Matrix): number[] {
        let resultsMatrix: number[] =
            new Array<number>(matrix.value.length)

        for (let i = 1; i < matrix.value.length + 1; i++) {

            let row = matrix.value[matrix.value.length - i]
            let rowRes = row[row.length - 1]
            let secondValue = row[row.length - i - 1]
            let addValue = 0

            for (let j = resultsMatrix.length - 1; j >= 0; j--) {
                if (resultsMatrix[j]) {
                    addValue += resultsMatrix[j] * row[j]
                }
            }

            resultsMatrix[resultsMatrix.length - i] = (rowRes - addValue) /
                secondValue
        }

        return resultsMatrix
    }

    calculateMatrix(matrix: Matrix): Matrix {
        let iteration = 0
```

```
        while (iteration + 1 < matrix.value.length) {
            for (let i = iteration + 1; i < matrix.value.length; i++) {
                matrix.value[i] =
                    this.recalculateRows(
                        matrix.value[iteration],
                        matrix.value[i],
                        iteration)

            }
            iteration++
        }

        return matrix
    }
}
```

3. Скриншоты программы

Пример внешнего вида программы представлен на рис. 3 и рис. 4.

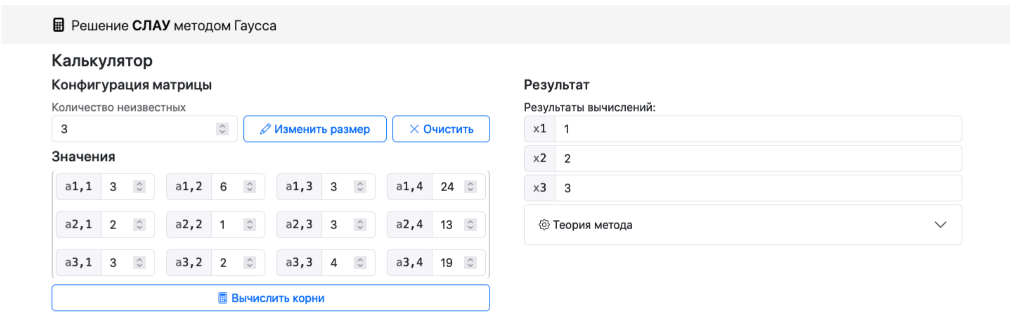

Рис. 3. Внешний вид приложения на ПК

Список литературы

[1] Кнут Д.Э. Всё про TEX. — Москва: Изд. Вильямс, 2003 г. 550 с.

[2] Львовский С.М. Набор и верстка в системе LaTeX. — 3-е издание, исправленное и дополненное, 2003 г.

[3] Воронцов К.В. LaTeX в примерах. 2005 г.

[4] Документация Angular 17. — https://v17.angular.io/docs, 2024 г.

[5] Документация TypeScript. — https://www.typescriptlang.org/docs, 2024 г.

[6] TypeScript CheatSheets. — https://www.typescriptlang.org/cheatsheets, 2024 г.