МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет
ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине
ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Вариант – прямые методы: метод Гаусса

Выполнил:
Студент группы P3232
Чмурова Мария
Владиславовна

г. Санкт-Петербург
2024 год

Оглавление

Задание

Решите систему линейных алгебраических уравнений, реализуя метод Гаусса. Также рассчитайте невязки.

Формат входных данных:

n

a11 a12 ... a1n b1

a21 a22 ... a2n b2

...

an1 an2 ... ann bn

Формат вывода:

x1

x2

...

xn

r1

r2

...

rn

, где x1..xn - значения неизвестных, а r1..rn - невязки.

Для систем, которые не имеют решений или имеют неограниченное количество решений, должно быть напечатано только следующее сообщение: "The system has no roots of equations or has an infinite set of them.". Для этого задайте значение переменной isSolutionExists и сообщение об ошибке.

Описание метода

Необходимо найти решение СЛАУ воспользовавшись методом Гаусса.

Первым шагом необходимо привести систему уравнений к треугольному виду. Это достигается последовательным исключением неизвестных системы.

Например, для СЛАУ:

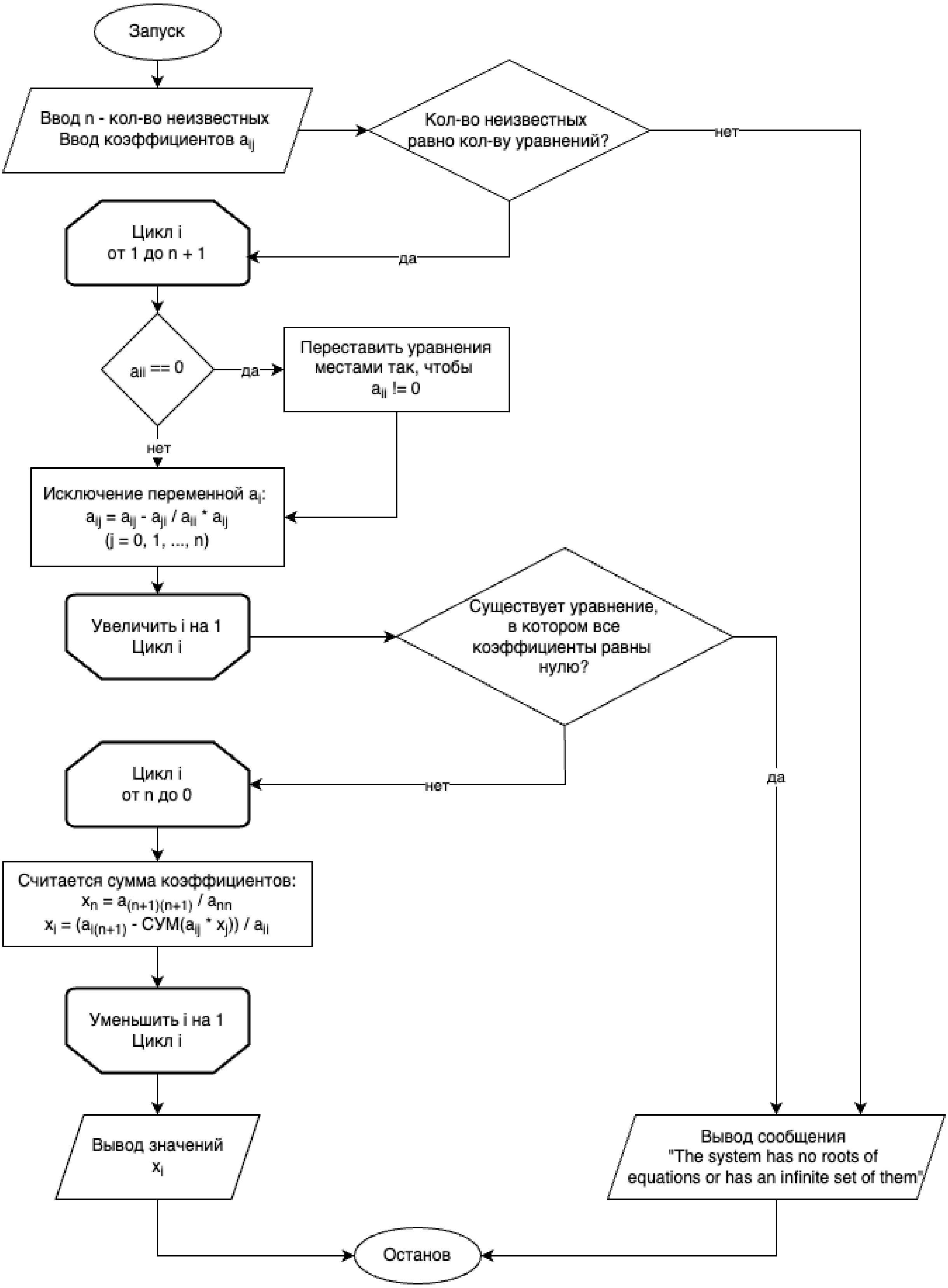
$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = a_{14} \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = a_{24} \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = a_{34} \end{cases}$$

- Исключается x_1 для всех последующих уравнений системы, прибавлением к уравнению слагаемых первого уравнения, умноженного на коэффициент $(-\frac{a_{i1}}{a_{11}})$: $a_{ij} = a_{ij} - \frac{a_{i1}}{a_{11}} \cdot a_{1j}$
- Аналогично исключается x_2 из третьего и последующего уравнений

Процесс необходимо продолжать до тех пор, пока не будет получено одно слагаемое в левой части последнего уравнения системы. Таким образом, будет получена треугольная матрица.

Вторым шагом необходимо воспользоваться обратным ходом (проходом по матрице от последнего уравнения к первому). Для этого решается последнее уравнение, находится значение неизвестного x_n, где n – количество неизвестных в уравнении. Используя это значения, вычисляется значение x_{n-1}. Процесс продолжается до тех пор, пока не будут получены значения всех неизвестных.

Блок схема


Запуск
Ввод n - кол-во неизвестных
Ввод коэффициентов a_{ij}
Кол-во неизвестных равно кол-ву уравнений?
нет
да
Цикл i
от 1 до n + 1
a_{ii} == 0
да
Переставить уравнения местами так, чтобы
a_{ii} != 0
нет
Исключение переменной a_i:
$a_{ij} = a_{ij} - a_{ji} / a_{ii} * a_{ij}$
(j = 0, 1, ..., n)
Увеличить i на 1
Цикл i
Существует уравнение, в котором все коэффициенты равны нулю?
да
нет
Цикл i
от n до 0
Считается сумма коэффициентов:
$x_n = a_{(n+1)(n+1)} / a_{nn}$
$x_i = (a_{i(n+1)}$ - СУМ$(a_{ij} * x_j)) / a_{ii}$
Уменьшить i на 1
Цикл i
Вывод значений
x_i
Вывод сообщения
"The system has no roots of equations or has an infinite set of them"
Останов

Код численного метода

```
class Solution:
    isSolutionExists = True
    errorMessage = ""

    # Функция для переставления строк при обнаружении нуля на 
главное диагонали
    def swap_lines(i, matrix):
        for k in range(i + 1, len(matrix)):
            if matrix[k][i] != 0:
                matrix[i], matrix[k] = matrix[k], matrix[i]
                return matrix
        return matrix

    # Функция для проверки существования решения СЛАУ
    def isSolvable(matrix):
        for row in matrix:
            # Проверка всех элементов кроме последнего (после 
знака равно)
            for coefficient in row[:-1]:
                if coefficient != 0:
                    return True
        return False

    # Функция для рассчета невязок
    def calculationOfResiduals(matrix, solution, n):

        for i in range(n):
            sum_of_left_part = 0
            for j in range(n):
                # Подсчет значений левой части уравнения
                sum_of_left_part += matrix[i][j] * solution[j]
            # Вычисление невязки
            solution[n + i] = matrix[i][-1] - 
round(sum_of_left_part, 10)
        return solution

    #Функция для решения СЛАУ методом Гаусса
    def solveByGauss(n, matrix):

        # Создание копии матрицы
        matrix_copy = [row[:] for row in matrix]

        # Проверка на то, что количество неизвестных равно 
количеству уравнений
        for row in matrix:
            if len(row) != n + 1:
                Solution.isSolutionExists = False
                Solution.errorMessage = "The system has no roots 
of equations or has an infinite set of them."
                return
```

```
        # 1. Приведение матрицы к диагональному (треугольному)
виду
        for i in range(n):

            # 1.1 Если элемент на главной диагонали равен нулю,
то необходимо поменять строки местами
            if matrix[i][i] == 0:
                matrix = Solution.swap_lines(i, matrix)
                if matrix[i][i] == 0:
                    Solution.isSolutionExists = False
                    Solution.errorMessage = "The system has no
roots of equations or has an infinite set of them."
                    return

            # 1.2 После переставления высчитываем треугольную
матрицу
            for j in range(i + 1, n):
                # Значение на которое умножаем все элементы
следующих строк
                koef = matrix[j][i] / matrix[i][i]
                for k in range(i, n + 1):
                    matrix[j][k] = matrix[j][k] - koef *
matrix[i][k]

        # 2. Проверка на возможность решения (если все
коэффициенты в строке до знака равно обнулились, то решения не
существует)
        if (Solution.isSolvable(matrix) == False):
            Solution.isSolutionExists = False
            Solution.errorMessage = "The system has no roots of
equations or has an infinite set of them."
            return

        # 3. Нахождение корней уравнения обратным ходом
        # Массив для хранения решений
        solution = [0] * n * 2
        for i in range(n - 1, -1, -1):
            sum_koef = 0
            for j in range(i + 1, n):
                # Сумма коэффициентов в левой части уравнения
                sum_koef = sum_koef + matrix[i][j] * solution[j]
            solution[i] = (matrix[i][-1] - sum_koef) /
matrix[i][i]

        # 4. Рассчет невязок
        Solution.calculationOfResiduals(matrix_copy, solution,
n)

        # 5. Возвращение полученных значений неизвестных
        return solution
```

Примеры работы программы

№ Теста	Входные данные	Выходные данные
1	3 2 3 -1 7 1 -1 2 -1 3 2 4 10	-0.5333333333333332 3.1333333333333333 1.3333333333333333 0.0 0.0 0.0
2	3 0 3 -1 7 0 -1 2 -1 0 2 4 10	The system has no roots of equations or has an infinite set of them.
3	4 1 1 1 1 5 2 2 2 2 10 3 3 3 3 15 4 4 4 4 20	The system has no roots of equations or has an infinite set of them.
4	5 2 1 3 1 -1 10 1 1 -2 -1 2 -3 3 1 2 4 -2 20 1 1 1 -2 3 5 4 1 1 1 1 15	0.8333333333333304 -1.5666666666666629 2.983333333333334 5.600000000000001 4.65 0.0 0.0 0.0 0.0 0.0
5	3 2 3 -1 7 4 1 -1 2 -1 3 3 2 4 10 2	The system has no roots of equations or has an infinite set of them.

Вывод

В ходе данной лабораторной работы была разработана программа, которая успешно находит решение СЛАУ, если решение существует и единственно, с помощью приведения матрицы к треугольному виду и обратному обходу для получения значения неизвестных. Кроме того, программа находит значения невязок.

Из-за выполнения большого количества арифметических операций во время вычисления значений СЛАУ метод Гаусса может быть менее эффективен на больших значениях по сравнению с такими методами, как метод Якоби или метод Зейделя.

Метод Гаусса применяется, когда необходимо решить систему линейных уравнений и является особенно эффективным если матрица треугольная или близка к треугольному виду.

Алгоритмическая сложность данного метода оценивается с помощью Big O Notation как $O(n^3)$, где n – количество неизвестных. Данную сложность вызывают внутренние циклы при приведении матрицы к треугольному виду во время вычисления значений коэффициентов.

При решении может возникнуть ошибки округления при выполнении арифметических операций, в особенности, операций деления во время вычисления значений неизвестных x_n. Это может привести к накоплению ошибки и получении некорректного результата.

Таким образом, метод Гаусса представляет собой простой и эффективный метод для решения СЛАУ, который обладает простой реализацией и может применяться во время необходимости решений систем линейных уравнений.