Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики
Факультет программной инженерии и компьютерной техники

Вычислительная математика
Лабораторная №1

Выполнил:
Беляков Дмитрий
Группа:
P3210
Преподаватель:
Перл О.В.

Санкт-Петербург
2020

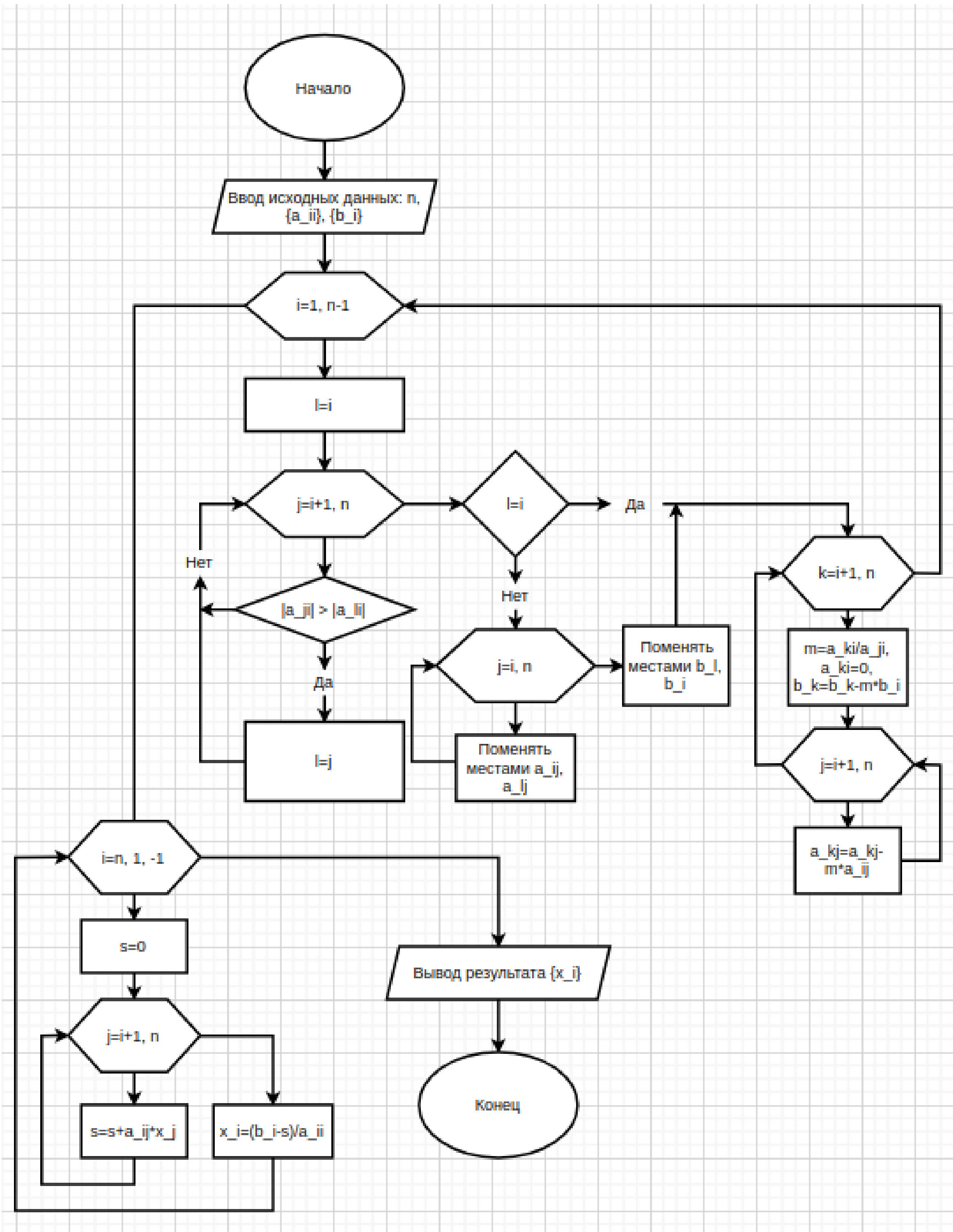
Описание метода

Схема с выбором главного элемента является одной из модификаций метода Гаусса.

Главная идея метода – перестановка уровнений таким образом, чтобы на очередном этапе прямого хода ведущим элементом оказался наибольший по модулю в столбце элемент. Данные перестановки производятся для повышения точности метода.

Рассмотрим подробнее: на k-ом шаге алгоритма в k-ом столбце выбирается максимальный по модулю элемент от k-ого до последнего (i = k, k+1, …, n). Строки i и k меняются местами. Таким образом реализуется выбор главного элемента «по столбцу». Для выбора главного элемента «по строке» алгоритм аналогичен, только элемент выбирается в k-ой строке от k-ого элемента и до последнего (j = k, k+1, …, n). Стобцы j и k меняются местами.

Блок-схема



Листинг кода

```
import numpy as np

def transform_to_triangular_matrix(matrix: np.ndarray, matrix_size: int):

    swaps_number = 0

    for i in range(matrix_size-1):

        max_column_value_index = np.argmax(np.abs(matrix[i:, i])) + i

        assert np.abs(matrix[max_column_value_index, i]) != 0, "Однозначаного решения нет"

        if i != max_column_value_index:
            matrix[[i, max_column_value_index]] = matrix[[max_column_value_index, i]]
            swaps_number +=1

        for j in range(i, matrix_size-1):
            matrix[j+1] = matrix[j+1] - matrix[i] * matrix[j+1, i] / matrix[i, i]

        matrix[i+1:,i] = 0

    assert np.abs(matrix[matrix_size-1, matrix_size-1]) != 0, "Однозначного решения нет"

    return swaps_number

def comp_determinant(matrix: np.ndarray, swaps_number: int, matrix_size: int):

    diagonal = [i for i in range(matrix_size)]
    determinant = np.prod(matrix[diagonal, diagonal])

    return determinant if swaps_number % 2 == 0 else -determinant

def comp_back_substitution(matrix: np.ndarray, matrix_size: int):

    decision_vector = np.zeros(matrix_size)

    for i in range(matrix_size-1, -1, -1):
        decision_vector[i] = (matrix[i, matrix_size] - np.sum(decision_vector * matrix[i, :matrix_size]))/matrix[i,i]

    return decision_vector

def comp_residual(source_matrix: np.ndarray, decision_vector: np.ndarray, matrix_size: int):

    return np.sum(source_matrix[:, :matrix_size] * decision_vector, axis = 1) - source_matrix[:, matrix_size]

def solve_by_ghaussian_method(source_matrix: np.array):

    matrix_size = source_matrix.shape[0]

    transformed_matrix = source_matrix.copy()
```

```
    swaps_number = transform_to_triangular_matrix(transformed_matrix, matrix_size)
    determinant = comp_determinant(transformed_matrix, swaps_number, matrix_size)
    decision_vector = comp_back_substitution(transformed_matrix, matrix_size)
    residual = comp_residual(source_matrix, decision_vector, matrix_size)

    return transformed_matrix, determinant, decision_vector, residual
```

Примеры и результаты работы

Тестовые данные

6
1 2 3 4 5 6 21
-10 14 23 6 -20 11 24
0 2 3 0.5 1.5 -5 2
12 5 7 8 9 10 51
0 0 0 5 0 0 5
1 2 1 1 1 1 7

Результат работы программы

Определитель: -106882.49999999999

Треугольная матрица:
12.0 5.0 7.0 8.0 9.0 10.0 51.0
0.0 18.166666666666668 28.833333333333332 12.666666666666668 -12.5 19.333333333333336 66.5
0.0 0.0 -2.0963302752293576 -0.7706422018348623 1.3394495412844034 -1.518348623853211 -3.045871559633027
0.0 0.0 0.0 5.0 0.0 0.0 5.0
0.0 0.0 0.0 0.0 5.277899343544857 3.5514223194748364 8.829321663019693
0.0 0.0 0.0 0.0 0.0 -8.862562189054728 -8.862562189054728

Неизвестные: 1.0 1.0000000000000002 0.9999999999999998 1.0 0.9999999999999998 1.0

Невязки: 0.0 3.552713678800501e-15 0.0 0.0 0.0 0.0

Ожидаемое решение

1 1 1 1 1 1

Тестовые данные

6
1 2 3 4 5 6 21
-10 14 23 6 -20 11 24
0 2 3 0.5 1.5 -5 2
12 5 7 8 9 10 51
0 0 0 0 0 0 0
1 2 1 1 1 1 7

Результат работы программы

Однозначного решения нет

Вывод

Плюсы:
1. При значениях определителя близких к нулю, даёт малые невязки
2. Даёт решение за конечное число арифметических операций
3. Наличие однозначного решения можно проверить на этапе преобразования к треугольной матрице
Минусы:
1. Требует хранение всей матрицы в памяти

Сравнение с другими методами:

1. По сравнению с методом Гаусса даёт меньшую погрешность: так как на главной диагонали могут оказаться числа близкие к нулю, увеличивается влияение ошибок округления. Постолбцовый метод выбора главного элемента позволяет уменьшить вычислительные погрешности.
2. Происходит накапливание погрешностей в процессе решения, так как на каждом этапе используются результаты предыдущих вычислений, что характерно для прямых методов, в отличие от итерационных, точность которых практически не зависит от предыдущих операций.
3. Также большим преимуществом итерационных методов является возможность регулирования точности вычисления.
4. Алгоритмы итерационных методов относительно более сложные, чем алгоритмы прямых методов