Университет ИТМО

Вычислительная математика

Лабораторная работа №1

ФИО студента: Готовко Алексей Владимирович
Направление подготовки: 09.03.04 (СППО)
Учебная группа: P32101
ФИО преподавателей:
Малышева Т.А.
Рыбаков С.Д.

Санкт-Петербург
2023г.

1 Цель работы

Разработка программы для решения СЛАУ.

2 Описание метода

Метод Гаусса основан на приведении матрицы системы к треугольному виду так, чтобы ниже ее главной диагонали находились только нулевые элементы.

Прямой ход метода состоит в последовательном исключении неизвестных из уравнений системы. Сначала с помощью первого уравнения исключается x_1 из всех последующих уравнений системы. Затем с помощью второго уравнения исключается x_2 из третьего и всех последующих уравнений. Такие операции продолжаются до исключения x_{n-1} из предпоследнего уравнения.

Обратный ход метода Гаусса состоит в последовательном вычислении искомых неизвестных: решая последнее уравнение, находим единственное в этом уравнении неизвестное x_n . Далее, используя это значение, из предыдущего уравнения вычисляем x_{n-1} и т. д. Последним найдем x_1 из первого уравнения.

3 Исходный код

```
def matrix_sort(matrix):
    n = len(matrix)
    for i, row in enumerate(matrix):
        if row[i] == 0:
            swapped = False
            for j in range(i + 1, n):
                if matrix[j][i] != 0:
                    matrix[i], matrix[j] = matrix[j], matrix[i]
                    swapped = True
                    break
            if not swapped:
                raise Exception(
                "The determinant equals to zero. Matrix has an infinite number of solutions or has not 
    return matrix


def matrix_to_triangular(matrix):
    n = len(matrix)

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

        for k in range(i + 1, n):
            for m in range(n, i - 1, -1):
                matrix[k][m] -= matrix[k][i] * matrix[i][m]

    return matrix


def matrix_triangular_determinant(matrix):
    d = 1
    for i in range(len(matrix)):
        d *= matrix[i][i]
    return d


def matrix_gauss_reverse(matrix):
    n = len(matrix)
    x = [0] * n
    for i in range(n - 1, -1, -1):
```

```
        s = 0
        for j in range(i + 1, n):
            s -= matrix[i][j] * x[j]
        s += matrix[i][n]
        x[i] = s / matrix[i][i]
    return x


def matrix_gauss_inaccuracy(matrix, solutions):
    n = len(matrix)
    errors = []
    for row in matrix:
        s = 0
        for i in range(n):
            s += row[i] * solutions[i]

        errors.append(s - row[n])

    return errors
```

4 Результат работы программы

Input matrix:
3
1 2 3 4
1 1 1 1
1 2 4 4

```
 __                Provided matrix:
|
|  1.00  2.00  3.00 |  4.00
|  1.00  1.00  1.00 |  1.00
|  1.00  2.00  4.00 |  4.00
|__

 __                Triangular matrix:
|
|  1.00  2.00  3.00 |  4.00
|  0.00  1.00  2.00 |  3.00
|  0.00  0.00  1.00 |  0.00
|__

_________________
|   Solution    |
|_______________|
| Index | Value |
|_______|_______|
|     1 | -2.00 |
|     2 |  3.00 |
|     3 |  0.00 |
|_______|_______|

_________________
|  Inaccuracy   |
|_______________|
| Index | Value |
|_______|_______|
|     1 |  0.00 |
|     2 |  0.00 |
|     3 |  0.00 |
|_______|_______|
```

5 Вывод

Для наиболее оптимального распределения человеческих ресурсов и поддержания количества нервных клеток в головном мозгу, настоятельно рекомендуется использовать готовые библиотеки, содержащие наиболее эффективные реализации алгоритмов, вместо самостоятельной реализации оных.

А вообще лучше зачиллиться и не изобретать велосипед.