МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ (ШКОЛА)

Департамент математического и компьютерного моделирования

ОТЧЁТ

к лабораторной работе №2 по дисциплине

«Вычислительная математика»

Направление подготовки
01.03.02 «Прикладная математика и информатика»

Выполнил студент гр.

Б9121-01.03.02сп(1)

Держапольский Ю.В. _____
(Ф.И.О.) *(подпись)*

« 24 » октября 2023 г.

г. Владивосток

2023

Содержание

1. Введение

В этой лабораторной работе будет проведена работа по программированию и тестированию алгоритма выбора главного элемента для решения системы линейных алгебраических уравнений.

2. Описание алгоритма

Дана матрица A и столбец b и система $Ax = b$. Суть метода выбора главного элемента в том, что на каждом шаге находится максимальный элемент матрицы, после чего из всех строк вычитается главная строка, умноженная на коэффициент для того чтобы получить нули в главном столбце.

Далее, на k-том шаге в необработанных до этого строках системы находится максимальный по модулю элемент. Пусть это будет элемент a_{ij}. Из каждой другой необработанной l-той строки вычитается данная строка, умноженная на $\dfrac{a_{lj}}{a_{ij}}$, тем самым получая нули в j-том столбце данных строк.

В итоге, после $n - 1$ шага получается матрица, которую можно решить путём исключения переменных.

3. Тестирование

Для тестирования будут сгенерированы случайные матрицы и столбцы размерностью 10 в количестве 10000. (код алгоритма см. в приложении)

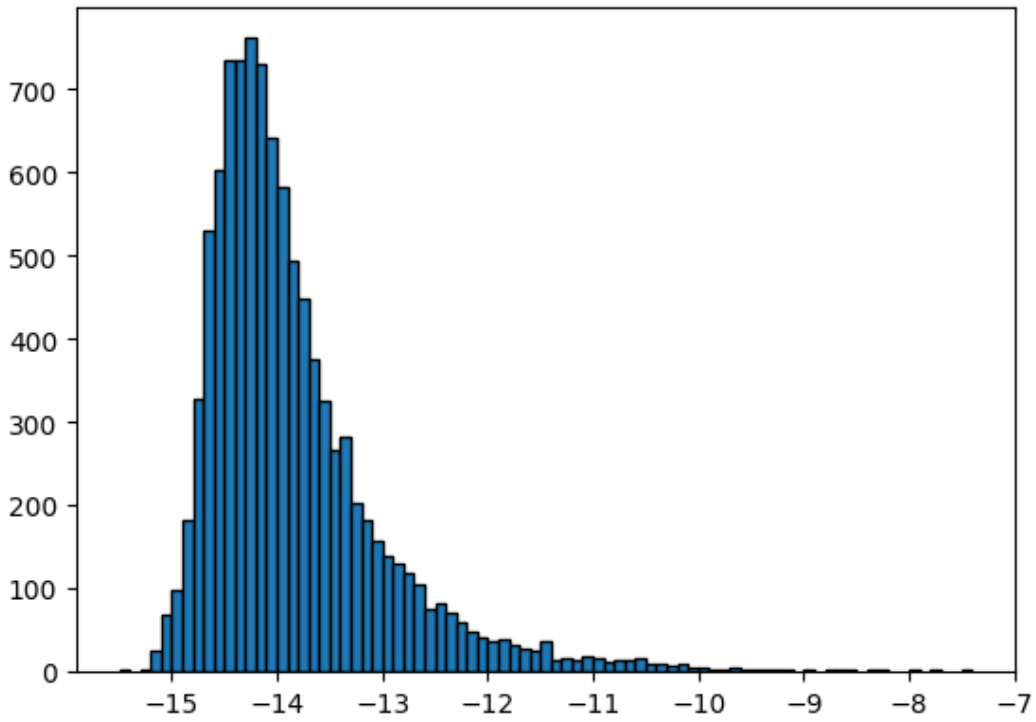

Рис. 1: Log10 максимальной разницы решения алгоритма и встроенной функции numpy, округлённый до десятых при генерации 10000 случайных матриц.

На рисунке 1 можно увидеть распределения точностей решения. Большинство решений по точности не превышает 10^{-10}. Решения с меньшей точностью определяются большим числом обусловленности сгенерированной матрицы – от 2000 до 31000. Однако, количество таких решений – 29, что составляет всего 0.29% от всех решений.

```
-9.5 3534.9413553374156
-9.2 27303.80682425332
-9.6 4831.541312804075
-9.6 8152.530089450168
-9.7 6343.536681509656
-9.4 5478.057441536083
-8.7 5825.808822026051
-9.5 6896.806373102329
-8.7 19100.07164206536
-9.7 7019.068926592585
-9.7 3703.485370576964
-9.7 6288.156392907357
-9.3 5724.979382134973
-8.0 24578.85820338723
-8.3 23680.815022595347
-9.9 13771.008251275467
-9.9 3048.1671775012364
-9.2 4127.141556212409
-9.5 5376.633244655705
-8.8 7023.143689348826
-9.0 31643.532806429073
-7.8 24502.16421115876
-9.3 3153.6765079033908
-9.8 1812.0256026814645
-7.5 27605.11681942663
-8.4 12140.716899280469
-8.6 13772.456853689588
-9.4 6964.29943810372
-9.9 2482.710546876706
```

Рис. 2: Log10 разности и число обусловленности решений, которые не превысили точность 10^{-10}.

4. Заключение

В этой лабораторной работе была проведена работа по программированию и тестированию алгоритма выбора главного элемента для решения системы линейных алгебраических уравнений.

Приложение

```
# main element method
import numpy as np
import utility as ut


def get_max_index(mat: np.matrix, exclude: list):
    args = abs(mat).argmax(axis=1)
    maxes = abs(mat).max(axis=1)
    alist = [((k, args[k]), maxes[k]) for k in range(
      len(args)) if k not in exclude]
    argmax = max(alist, key=lambda x: x[1])
    return argmax[0]


def solve(matrix: np.matrix, values: np.array):
    matrix = matrix.copy().astype(float)
    values = values.copy().astype(float)

    if np.linalg.det(matrix) == 0:
        exitСистема(" несовместна!")

    rows_exclude = []
    for _ in range(len(matrix)):
        ind = get_max_index(matrix, rows_exclude)
        rows_exclude.append(ind[0])
        values[ind[0]] = values[ind[0]] / matrix[ind]
        matrix[ind[0]] = matrix[ind[0]] / matrix[ind]
```

```
        for i in range(len(matrix)):
            if i not in rows_exclude:
                values[i] -= matrix[(i, ind[1])] * values[ind[0]]
                matrix[i] -= matrix[(i, ind[1])] * matrix[ind[0]]

    rows_exclude.reverse()
    for i in rows_exclude:
        ind = matrix[i].argmax()
        for j in range(len(matrix)):
            if j != i:
                values[j] -= matrix[(j, ind)] * values[i]
                matrix[j] -= matrix[(j, ind)] * matrix[i]

    solution = np.array([0.] * len(matrix))
    for i in range(len(matrix)):
        ind = matrix[i].argmax()
        solution[ind] = values[i]

    return solution
```