НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи

з дисципліни «АЛГОРИТМИ І СИСТЕМИ КОМП’ЮТЕРНОЇ

МАТЕМАТИКИ 1.МАТЕМАТИЧНІ АЛГОРИТМИ»

на тему

“Розв’язання систем лінійних рівнянь”

|  |  |
| --- | --- |
| Виконала: | Перевірила: |
| студентка групи КМ-01 | Асистент кафедри ПМА |
| Резниченко Є. С. | Ковальчук-Химюк Л. О. |

Київ — 2023

Зміст

[Вступ 3](#_Toc154071414)

[Основна частина 4](#_Toc154071415)

[Варіант 15 4](#_Toc154071416)

[Опис роботи програми 4](#_Toc154071417)

[Висновки 5](#_Toc154071418)

[Відповіді на контрольні питання 6](#_Toc154071419)

[Перелік посилань 7](#_Toc154071420)

[Додаток А 8](#_Toc154071421)

[Додаток Б 9](#_Toc154071422)

[Вміст файлу main.py 9](#_Toc154071423)

[Вміст файлу main.m 10](#_Toc154071424)

# Вступ

Мета роботи – вивчити правила використання програмних засобів для факторизації матриць і розв’язання систем лінійних алгебраїчних рівнянь, розв’язувати задану систему рівнянь, провести порівняльний аналіз вивчених чисельних методів розв’язання СЛР.

# Основна частина

## Варіант 15

|  |  |
| --- | --- |
| матриця | вектор |
|  |  |

## Опис роботи програми

У обидвох мовах було застосовано LU алгоритм [[1]](https://johnfoster.pge.utexas.edu/numerical-methods-book/LinearAlgebra_LU.html)

Python:

Використано бібліотечну функцію numpy.linalg.solve() [[3]](https://numpy.org/doc/stable/reference/generated/numpy.linalg.solve.html)

Octave:

Використано вбудовану функцію розв’язку СЛАР

Основний алгоритм LU-розкладу є модифікованим методом Гауса і вимагає 2n^3 / 3 арифметичних операцій. Ось як це працює:

1. Позначимо як l\_ij, u\_ij, a\_ij елементи матриць L, U та A відповідно.
2. З означення LU-розбиття l\_ij = 0 (j>i), u\_ij = 0 (j<i), u\_ii = 11.
3. Звідси легко отримати вирази для елементів матриць L та U1.

Цей метод має перевагу над методом Гауса, оскільки дозволяє більш просто отримати розв’язок для різних векторів b системи лінійних алгебраїчних рівнянь A \* x = b.

# Висновки

|  |  |  |
| --- | --- | --- |
|  | Python | Octave |
| Час виконання (c) | 0.007 | 0.0086839 |
| Похибка | 1.459e-28 | 2.019e-28 |
| Результат | [ 4.38112979 5.22361945  6.23205926  -67.80998246] | [4.38112978733865;  5.22361944925462;  6.23205925893555;  -67.8099824648614] |

У ході розв’язку СЛАР отримали результати наведені в таблиці. Похибка і час виконання малі для обох мов програмування.

Додаток А містить скріншоти роботи програми.

# Відповіді на контрольні питання

1. Система лінійних алгебраїчних рівнянь має однозначний розв’язок, коли число рівнянь в системі дорівнює числу невідомих. Це також можливо, якщо детермінант матриці системи не є нулем.
2. Факторизація матриць допомагає розв’язати систему лінійних рівнянь, перетворюючи початкову матрицю на трикутну форму за допомогою лінійних перетворень.
3. Методи розв’язання систем лінійних алгебраїчних рівнянь поділяються на прямі (або точні) та ітераційні (або наближені). Прямий метод, наприклад метод Гаусса, дає точний розв’язок за визначену кількість кроків. Ітераційний метод, наприклад метод Зейделя, навпаки, дає можливість отримати розв’язок з заданою точністю.
4. Компактна схема методу Гаусса - це спосіб запису обчислень, який повністю виконує традиційний метод Гаусса, але в більш економному форматі.
5. Метод Зейделя - це варіант методу простих ітерацій. В процесі обчислення i-ої координати нової точки x використовуються вже обчисленi (i-1) координати нової точки та (n-i+1) координати попередньої точки.
6. Збiжнiсть iтерaцiйного процесу означaє, що процес iтерaцiй продовжується до досягнення необхiдної точностi розв’язку.
7. Самовиправляючими називають методи, якi використовуються для зменшення впливу помилок, якi можуть виникати пiд час обчислень, та для покращення точностi результатiв. Основнi методи автоматичного виправлення помилок включають: • Метод найменших квадратiв (МНК): Цей метод дозволяє знаходити найкращий наближений розв’язок СЛАР, навiть якщо вихiднi данi мiстять помилки або шум. • Метод QR-розкладу: Цей метод дозволяє розкласти матрицю системи на добуток ортогональної матрицi i верхньотрикутної матрицi. • Методи регуляризацii: Цi методи використовуються для стабiлiзацii обчислень i запобiгання перенавчанню в разi надмiрного обмiну даними. • Методи iтерaцiiного покращення: Цi методи включають iтерaцiiйнi алгоритми, такi як метод Гаусса-Зейделя, якi дозволяють поетапно покращувати розв’язок СЛАР. Вони можуть бути використанi для автоматичного виправлення результатiв, якщо вони не задовольняють точностi. Цi методи допомагають покращити стiйкiсть i точнiсть обчислень в системах лiнiйних алгебраїчних рiвнянь в разi наявностi помилок або неточностей в даних або обчисленнях.

# Перелік посилань

1. LU alhorithm - <https://johnfoster.pge.utexas.edu/numerical-methods-book/LinearAlgebra_LU.html>
2. Octave guide - <https://oct2py.readthedocs.io/en/latest/examples.html>
3. Numpy.linalg.solve - <https://numpy.org/doc/stable/reference/generated/numpy.linalg.solve.html>

# Додаток А

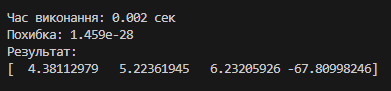


Рис. 1 – Виконання коду на Python

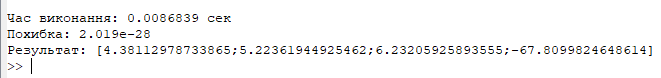


Рис. 2 – Виконання коду на Octave

# Додаток Б

## Вміст файлу main.py :

import time

import os

import numpy as np

def read\_matrix(filename):

    """Функція для зчитування матриці з файлу

    """

    try:

        with open(filename, 'r') as file:

            lines = file.readlines()

            matrix = []

            for line in lines:

                row = [float(x) for x in line.strip().split()]

                matrix.append(row)

            return np.array(matrix)

    except FileNotFoundError:

        print('ERROR', f'Помилка: Файл `{filename}` не знайдено.')

    except ValueError:

        print('ERROR', f'Помилка: Матриця в файлі `{filename}` містить неправильні значення.')

    except Exception as e:

        print('ERROR', f'Помилка: Виникла невідома помилка при зчитуванні матриці: {str(e)}')

def read\_vector(filename):

    """Функція для зчитування вектора з файлу

    """

    try:

        with open(filename, 'r') as file:

            lines = file.readlines()

            vector = [float(line.strip()) for line in lines]

            return np.array(vector)

    except FileNotFoundError:

        print('ERROR', f'Помилка: Файл `{filename}` не знайдено.')

    except ValueError:

        print('ERROR', f'Помилка: Вектор в файлі `{filename}` містить неправильні значення.')

    except Exception as e:

        print('ERROR', f'Помилка: Виникла невідома помилка при зчитуванні вектора: {str(e)}')

if \_\_name\_\_ == '\_\_main\_\_':

    os.system('cls')  #  Очищення консолі

    start\_python\_time = time.time()

    # Зчитування матриці A та вектора b з файлів

    matrix\_A = read\_matrix('D:\KPI\ASKM\Labs\Reznichenko\Lab\_1\matrix.txt')

    vector\_b = read\_vector('D:\KPI\ASKM\Labs\Reznichenko\Lab\_1\\vector.txt')

    if len(vector\_b) != len(matrix\_A):

        print('ERROR', f'Помилка: Вектор b не відповідає розмірності матриці А.')

    if len(vector\_b) >= 10 or len(matrix\_A) >= 10:

        print(f'Введіть матрицю меншої розмірності')

    # Розв'язання СЛАР Ax = b

    try:

        solution = np.linalg.solve(matrix\_A, vector\_b)

        # Обчислення похибки

        accuracy = np.mean((vector\_b - np.dot(matrix\_A, solution)) \*\* 2)

        python\_run\_time = time.time() - start\_python\_time

        print()

        print(f'Час виконання: {round(python\_run\_time, 4)} сек')

        print(f'Похибка: {"{:.3e}".format(accuracy)}')

        print(f'Результат: \n{solution}')

    except:

        pass

## Вміст файлу main.m :

% Зчитування матриці A та вектора b з файлів

try

tic; % Початок вимірювання часу

matrix\_A = dlmread('D:\\KPI\\ASKM\\Labs\\Reznichenko\\Lab\_1\\matrix.txt');

vector\_b = dlmread('D:\\KPI\\ASKM\\Labs\\Reznichenko\\Lab\_1\\vector.txt');

catch

disp('ERROR: Помилка при зчитуванні матриці або вектора');

end\_try\_catch

if length(vector\_b) != size(matrix\_A, 1)

disp('ERROR: Помилка: Вектор b не відповідає розмірності матриці А.');

end

if length(vector\_b) >= 10 || size(matrix\_A, 1) >= 10

disp('Введіть матрицю меншої розмірності');

end

% Розв'язання СЛАР Ax = b

try

solution = matrix\_A \ vector\_b;

% Обчислення похибки

accuracy = mean((vector\_b - matrix\_A \* solution) .^ 2);

execution\_time = toc; % Кінець вимірювання часу

disp(['Час виконання: ', num2str(execution\_time), ' сек']);

disp(['Похибка: ', num2str(accuracy, '%.3e')]);

disp(['Результат: ', mat2str(solution)]);

catch

disp('ERROR: Помилка при розв`язанні СЛАР');

end\_try\_catch