Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 3

з дисципліни «Спеціальні розділи математики-2.
Чисельні методи»

на тему

«Розв'язання систем лінійних алгебраїчних рівнянь (СЛАР) ітераційними методами. Метод простої ітерації. Метод Зейделя»

Виконав:
студент гр. ІС-34
Колосов Ігор
Викладач:
доц. Рибачук Л.В.

Київ – 2024

Зміст

1. Постановка Задачі

2 Завдання

Якщо матриця не є матрицею із діагональною перевагою, звести систему до еквівалентної, у якій є діагональна перевага (виконати письмово, включити в звіт). Можна, наприклад, провести одну ітерацію метода Гауса, зкомбінувавши рядки з метою отримати нульовий недіагональний елемент у стовпчику.

Розробити програму, що реалізує розв'язання системи методом простої ітерації та методом Зейделя. Обчислення проводити з з кількістю значущих цифр $m = 6$. Для кожної ітерації розраховувати нев'язку $\boldsymbol{r = b - Ax}$, де $\boldsymbol{x}$ – отриманий розв'язок.

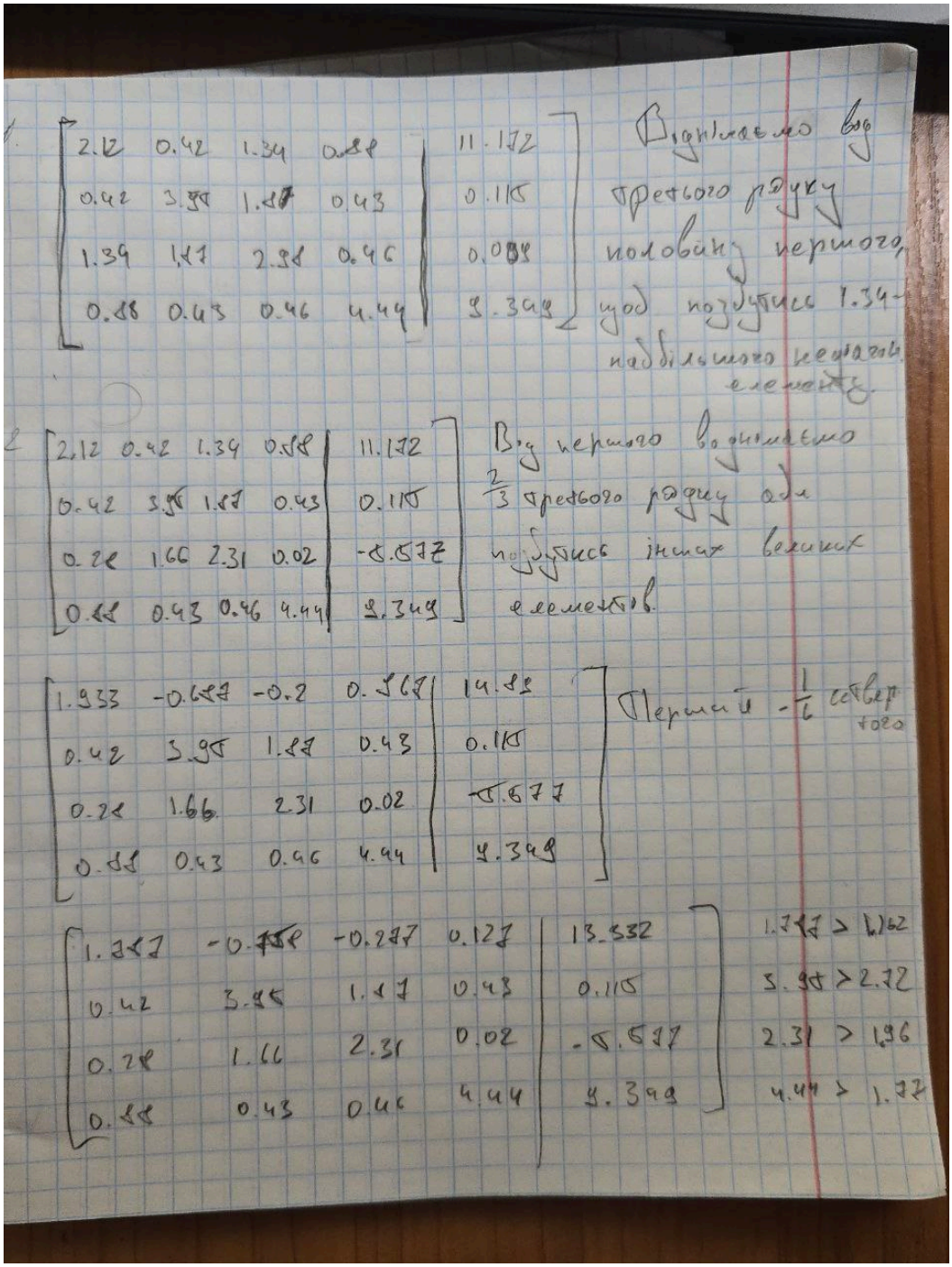
Розв'язати задану систему рівнянь за допомогою програмного забезпечення Mathcad. Навести результат перевірки: вектор нев'язки $\boldsymbol{r = b - Ax_m}$, де $\boldsymbol{x_m}$ – отриманий у Mathcad розв'язок.

Порівняти корені рівнянь, отримані у Mathcad, із власними результатами за допомогою методу середньоквадратичної похибки.

2. Вихідна система рівнянь

10	$\begin{pmatrix} 2{,}12 & 0{,}42 & 1{,}34 & 0{,}88 \\ 0{,}42 & 3{,}95 & 1{,}87 & 0{,}43 \\ 1{,}34 & 1{,}87 & 2{,}98 & 0{,}46 \\ 0{,}88 & 0{,}43 & 0{,}46 & 4{,}44 \end{pmatrix}$	$\begin{pmatrix} 11{,}172 \\ 0{,}115 \\ 0{,}009 \\ 9{,}349 \end{pmatrix}$

3. Письмовий етап приведення матриці до діагональної переваги



4. **Результати трьох та останньої ітерації методу (Із вектором нев'язки)**

- **Метод Якобі**

```
(jacobi) a @ x - b; iteration  1
          +0.91410, -0.47586, +2.17940, +5.46723
      -> x, with change (x_new - x)  7.46055
          +7.46055, +0.02911, -2.41429, +2.10563


(jacobi) a @ x - b; iteration  2
          +0.01364, -2.50860, +0.03213, -0.83234
      -> x, with change (x_new - x)  1.23136
          +6.94902, +0.14959, -3.35775, +0.87427


(jacobi) a @ x - b; iteration  3
          -0.45374, +0.05140, +1.05586, +0.25997
      -> x, with change (x_new - x)  0.63509
          +6.94139, +0.78467, -3.37165, +1.06174


(jacobi) a @ x - b; iteration 23
          -0.00000, -0.00002, +0.00001, +0.00000
      -> x, with change (x_new - x)  0.00001
          +7.21770, +1.08341, -4.07630, +0.99249

(jacobi) a @ x - b, change (x_new - x):  0.00001
 -0.00000, -0.00002, 0.00001, 0.00000
```

- **Метод Зейделя**

```
(seidel) a @ x - b; iteration  1
          +1.47184, -4.75409, +0.01976, +0.00000
      -> x, with change (x_new - x)  7.46055
          +7.46055, -0.76416, -2.76946, +0.98790


(seidel) a @ x - b; iteration  2
          -0.73110, -1.51066, +0.00250, +0.00000
      -> x, with change (x_new - x)  1.29114
          +6.63691, +0.52698, -3.60601, +1.11277


(seidel) a @ x - b; iteration  3
          -0.18601, -0.58611, -0.00167, +0.00000
      -> x, with change (x_new - x)  0.40912
          +7.04604, +0.86593, -3.90025, +1.02934


(seidel) a @ x - b; iteration 14
          -0.00000, -0.00001, -0.00000, +0.00000
      -> x, with change (x_new - x)  0.00001
          +7.21771, +1.08341, -4.07631, +0.99249

(seidel) a @ x - b, change (x_new - x):  0.00001, epsilon: 0.00001
 -0.00000, -0.00001, -0.00000, 0.00000
```

5. Копія розв’язку задачі у Mathcad; Вектор нев’язки

$$\text{jacobi} = \begin{pmatrix} 7.218 \\ 1.083 \\ -4.076 \\ 0.992 \end{pmatrix} \qquad b^T - A \cdot \text{jacobi} = \begin{pmatrix} 2.708 \times 10^{-6} \\ 1.531 \times 10^{-5} \\ -1.176 \times 10^{-5} \\ -1.113 \times 10^{-6} \end{pmatrix}$$

$$\text{seidel} = \begin{pmatrix} 7.218 \\ 1.083 \\ -4.076 \\ 0.992 \end{pmatrix} \qquad b^T - A \cdot \text{seidel} = \begin{pmatrix} 3.852 \times 10^{-6} \\ 1.112 \times 10^{-5} \\ 2.466 \times 10^{-8} \\ 0 \end{pmatrix}$$

$$\text{sol} := \text{lsolve}\left(A, b^T\right) = \begin{pmatrix} 7.218 \\ 1.083 \\ -4.076 \\ 0.992 \end{pmatrix} \qquad b^T - A \cdot \text{sol} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -1.776 \times 10^{-15} \end{pmatrix}$$

6. Порівняння власного розв'язку та розв'язку отриманого у Mathcad

```
(jacobi)
 7.21770, 1.08341, -4.07630, 0.99249

(seidel)
 7.21770, 1.08341, -4.07630, 0.99249
```

$$\text{python\_sol} := (7.21770 \quad 1.08341 \quad -4.07630 \quad 0.99249)$$

$$\text{python\_sol}^{T} - \text{sol} = \begin{pmatrix} -8.577 \times 10^{-6} \\ -6.312 \times 10^{-6} \\ 1.137 \times 10^{-5} \\ 1.111 \times 10^{-6} \end{pmatrix}$$

7. Лістинг розв'язку Mathcad

```
sum :=  n ← 3
        "Sum of the matrix rows except diagonal"
        for i ∈ 0..n
          S_i ← Σ_{j=1}^{n} (|A_{i,j}| - |A_{i,i}|)
        S

check := n ← 4
         "Check if matrix is diagonally dominant"
         for i ∈ 1..n
           if |A_{i,i}| < sum_i
             result ← "Матриця А не має діагональної переваги"
             break
           otherwise
             result ← "Матриця А має діагональну перевагу"
             break
         result

check = "Матриця А має діагональну перевагу"
```

```
jacobi := "Jacobi method"
          x ← (0 0 0 0)
          xnew ← x
          D_inv ← diag(1 / diag(A))
          itmat ← [(identity(rows(A))) - D_inv·A]^T
          itvec ← D_inv·b^T
          i ← 0
          while i < 10000
            xnew ← x·itmat + itvec^T
            error ← max(|xnew - x|)
            return xnew^T if error < 0.00001
            x ← xnew
            i ← i + 1
          x^T
```

```
seidel := "Gauss-Seidel method"
          n ← rows(A)
          x ← [(0 0 0 0)^T]
          k ← 0
          while k < 100
            i ← 0
            xnew ← x
            while i < n
              sum1 ← 0
              j ← 0
              while j < i
                sum1 ← sum1 + A_{i,j}·xnew_j
                j ← j + 1
              sum2 ← 0
              j2 ← i + 1
              while j2 < rows(A)
                sum2 ← sum2 + A_{i,j2}·x_{j2}
                j2 ← j2 + 1
              xnew_i ← [(b^T)_i - sum1 - sum2] / A_{i,i}
              i ← i + 1
            error ← max(|xnew - x|)
            return xnew if error < 0.00001
            k ← k + 1
            x ← xnew
          x
```

8. Лістинг програми

```
import numpy as np
from lib_print import *
np.set_printoptions(precision=6, suppress=True, floatmode="fixed")
```

```
def is_strictly_diagonally_dominant(matrix):
    for i in range(matrix.shape[0]):
        if abs(matrix[i, i]) <= np.sum(np.abs(matrix[i, :])) -
abs(matrix[i, i]):
            return False
    return True


# за завданням
vector_b = np.array([11.172, 0.115, 0.009, 9.349]).T
matrix_a = np.array([
  [2.12, 0.42, 1.34, 0.88],
  [0.42, 3.95, 1.87, 0.43],
  [1.34, 1.87, 2.98, 0.46],
  [0.88, 0.43, 0.46, 4.44],
])

# результат допрограмового етапу
matrix_a = np.array([
    [1.787, -0.758, -0.277, 0.127],
    [0.42, 3.95, 1.87, 0.43],
    [0.28, 1.66, 2.31, 0.02],
    [0.88, 0.43, 0.46, 4.44]])
vector_b = np.array([13.332, 0.115, -5.577, 9.349])

print(is_strictly_diagonally_dominant(matrix_a))

a = matrix_a.copy()
b = vector_b.copy()
n = a.shape[0]



print(printer(a, f"{ANSI.Styles.BOLD} Matrix A{ANSI.Styles.RESET}",
default_style=ANSI.Styles.ITALIC), end="\n\n")
print(printer(b.reshape(-1, 1), f"{ANSI.Styles.BOLD} Vector
b{ANSI.Styles.RESET}", default_style=ANSI.Styles.ITALIC), end="\n\n")

def jacobi(a, b, epsilon=1e-5, max_iterations=1000):
```

```
    __print_strouput = f" {ANSI.Styles.BOLD}{ANSI.FG.BLUE}JACOBI
ALGORYTHM{ANSI.Styles.RESET}\n\n"
    __default_print_style = ANSI.FG.BRIGHT_BLACK

    n = a.shape[0]
    D_inv = np.diag(1 / np.diag(a))
    iteration_matrix = np.eye(n) - D_inv @ a
    iteration_vector = D_inv @ b
    x = np.zeros_like(b)



    __print_highlights_diag = [highlight([(i, i) for i in
range(a.shape[0])], ANSI.FG.BLUE, 1, f"{ANSI.Styles.BOLD}Diagonal ")]
    __print_highlights_not_diag = [highlight([(i, i) for i in
range(a.shape[0])], ANSI.FG.BRIGHT_BLACK, 1,
f"{ANSI.Styles.BOLD}Diagonal ")]
    __print_strouput += printer(D_inv, ANSI.Styles.BOLD +" (jacobi) D
inverted", __print_highlights_diag,
default_style=ANSI.Styles.ITALIC+__default_print_style,
formatting="+0.5f") + "\n\n"
    __print_strouput += printer(iteration_matrix, ANSI.Styles.BOLD +"
(jacobi) iteration matrix ", __print_highlights_not_diag,
formatting="+0.5f", default_style=ANSI.Styles.ITALIC) + "\n\n"
    __print_strouput += printer(iteration_vector.reshape(-1, 1),
ANSI.Styles.BOLD +" (jacobi) iteration vector ",
default_style=ANSI.Styles.ITALIC) + '\n\n'

    for i in range(max_iterations):
        x_new = iteration_matrix @ x + iteration_vector

        error = np.max(np.abs(x_new - x))

        __print_strouput += "\n" + printer((a @ x_new - b).reshape(-1,
1), f"\n\n{ANSI.Styles.BOLD} (jacobi) a @ x - b; iteration {1+i:2d}
{ANSI.Styles.RESET}", formatting= "+0.5f",
default_style=ANSI.FG.BRIGHT_BLACK+ANSI.Styles.ITALIC, pre_row_str="
") + "\n"
        __print_strouput += printer(x_new.reshape(-1, 1),
```

```
f"{ANSI.Styles.BOLD}       -> x, with change (x_new - x) {ANSI.FG.RED}
{error:0.5f} ", [], "+0.5f",
default_style=ANSI.Styles.ITALIC+ANSI.FG.BRIGHT_BLACK, pre_row_str="
")

        if error < epsilon:

            __print_strouput += printer((a @ x_new - b).reshape(-1, 1),
f"\n\n{ANSI.Styles.BOLD} (jacobi) a @ x - b, change (x_new - x):
{ANSI.FG.RED} {error:0.5f}{ANSI.Styles.RESET}",
default_style=ANSI.FG.GREEN+ANSI.Styles.ITALIC, formatting="0.5f")

            print(__print_strouput)

            return x_new
        x = x_new

    return x


def seidel(a, b, epsilon=1e-5, max_iterations=1000):
    n = a.shape[0]
    x = np.zeros_like(b)

    __print_strouput = f" {ANSI.Styles.BOLD}{ANSI.FG.BLUE}Seidel
ALGORYTHM{ANSI.Styles.RESET}"


    for k in range(max_iterations):
        x_new = np.zeros_like(x)

        for i in range(n):
            sum1 = sum(a[i,j] * x_new[j] for j in range(i))
            sum2 = sum(a[i,j] * x[j] for j in range(i+1, n))
            x_new[i] = (b[i] - sum1 - sum2) / a[i,i]


        error = np.max(np.abs(x_new - x))
        __print_strouput += "\n" + printer((a @ x_new - b).reshape(-1,
```

```
1), f"\n\n{ANSI.Styles.BOLD} (seidel) a @ x - b; iteration {1+k:2d}
{ANSI.Styles.RESET}", formatting= "+0.5f",
default_style=ANSI.FG.BRIGHT_BLACK+ANSI.Styles.ITALIC, pre_row_str="
") + "\n"
        __print_strouput += printer(x_new.reshape(-1, 1),
f"{ANSI.Styles.BOLD}       -> x, with change (x_new - x) {ANSI.FG.RED}
{error:0.5f} ", [], "+0.5f",
default_style=ANSI.FG.BRIGHT_BLACK+ANSI.Styles.ITALIC, pre_row_str="
")


        if error < epsilon:
            break
        x = x_new

    __print_strouput +=  printer((a @ x_new - b).reshape(-1, 1),
f"{ANSI.Styles.BOLD}\n\n (seidel) a @ x - b, change (x_new - x):
{ANSI.FG.RED} {error:0.5f}{ANSI.Styles.RESET}, {ANSI.FG.BLUE}epsilon:
{epsilon:0.5f}", default_style=ANSI.Styles.ITALIC+ANSI.FG.GREEN,
formatting="0.5f")
    print(__print_strouput)
    return x


sei = seidel(a.copy(), b.copy())
print()
jac = jacobi(a.copy(), b.copy())

print(printer((jac).reshape(-1, 1), f"{ANSI.Styles.BOLD}\n\n (jacobi)
{ANSI.FG.RED}{ANSI.Styles.RESET}",
default_style=ANSI.Styles.ITALIC+ANSI.FG.GREEN, formatting="0.5f"),
end="")
print(printer((sei).reshape(-1, 1), f"{ANSI.Styles.BOLD}\n\n (seidel)
{ANSI.FG.RED}{ANSI.Styles.RESET}",
default_style=ANSI.Styles.ITALIC+ANSI.FG.GREEN, formatting="0.5f"))
```