Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

Звіт
Про виконання лабораторної роботи №5

На тему:

«Наближені методи розв’язування систем лінійних алгебраїчних рівнянь»
з дисципліни «Чисельні методи»

Лекторка:
доцент каф. ПЗ
Мельник Н. Б.
Виконав:
ст. гр. ПЗ-11
Солтисюк Д.А.
Прийняла:
доцент каф. ПЗ
Мельник Н. Б.

« __ » ________ 2022 р.

∑ = _____ _________ .

Львів – 2022

Тема: Наближені методи розв'язування систем лінійних алгебраїчних рівнянь.
Мета: Ознайомлення на практиці з методами Якобі та Зейделя розв'язування систем лінійних алгебраїчних рівнянь.

Теоретичні відомості

Метод Якобі – базується на використанні зведеної системи лінійних алгебраїчних рівнянь, яку отримують вираженням невідомих на головній діагоналі через інші невідомі кожного рівняння, при чому діагональні елементи вхідної матриці ненульові. Матриця такого вигляду матиме нулі на головній діагоналі. Тоді підставляємо певне початкове наближення в кожне рівняння (наприклад вектор вільних членів зведеної системи) і отримуємо новий вектор, який підставлятимемо доти, доки не виконається умова

$$\left|X^{(k)}-X^{(k-1)}\right| = \sqrt{\sum_{i=1}^{n}\left(x_i^{(k)} - x_i^{(k-1)}\right)^2} < \varepsilon .$$

Обчислення виконуємо за формулами

$$\begin{cases} x_i^{(0)} = \beta_i \\ x_i^{(k)} = \beta_i + \sum_{j=1}^{i-1} a_{ij}x_j^{(k-1)} + \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}, \end{cases} \quad i = \overline{1,n},\ k = 1,2,\ldots$$

Збіжність методу можна перевірити за нормами матриць, які можна знайти трьома способами – максимальне значення суми абсолютних значень рядків матриці коефіцієнтів, стовпців, та сума квадратів всіх елементів матриці коефіцієнтів. Якщо хоч одна норма менша одиниці – метод збіжний.

Метод Зейделя – є модифікацію методу Якобі, в якому одразу ж використовуються щойно отримані нові значення вектора X. Тоді ітераційна формула відрізнятиметься лише однією змінною, проте зазвичай збіжність цього методу краща: $x_i^{(k)} = \beta_i + \sum\limits_{j=1}^{i-1} a_{ij}x_j^{(k)} + \sum\limits_{j=i+1}^{n} a_{ij}x_j^{(k-1)}, i = \overline{1,n},\ k = 1,2,\ldots$

Умова збіжності та критерій виходу із циклу такі ж як і для методу Якобі.

Індивідуальне завдання

Варіант 9

Розв’язати систему лінійних алгебраїчних рівнянь методом Зейделя та методом Якобі з точністю ε = 0,001. Порівняти кількість ітерацій для обох методів:

$$9)\begin{cases} 0{,}81x_1 - 0{,}07x_2 + 0{,}38x_3 - 0{,}21x_4 = 0{,}81 \\ -0{,}22x_1 + 0{,}78x_2 + 0{,}11x_3 + 0{,}33x_4 = -0{,}64 \\ 0{,}51x_1 - 0{,}07x_2 + 3{,}09x_3 - 0{,}11x_4 = -1{,}71 \\ 0{,}33x_1 - 0{,}41x_2 + 0{,}83x_4 = -1{,}21 \end{cases}$$

Код функцій

```python
import numpy as np
from common.main import Matrix, MatrixOrientedMethod, Row
from common.utils import is_convergent, print_header


class SeidelMethod(MatrixOrientedMethod):

    def __init__(self, A: Matrix, B: Row) -> None:
        super().__init__(A, B)
        self.method_name = "Seidel"

        if not is_convergent(self.A):
            raise ValueError("A is not convergent")

    def execute_method(self, eps=1e-3, max_iterations=1000):
        x = np.zeros_like(self.B, dtype=np.double)

        for k in range(max_iterations):
            print_header(f"\nIteration {k}")
            print(f"x={x}")

            x_old = x.copy()

            for i in range(self.A.shape[0]):
                x[i] = (self.B[i] - np.dot(self.A[i, :i], x[:i]) - np.dot(
                    self.A[i, (i + 1):], x_old[(i + 1):])) / self.A[i, i]

            if np.linalg.norm(x - x_old, ord=np.inf) / np.linalg.norm(
                    x, ord=np.inf) < eps:
                break

        return x


class JakobiMethod(MatrixOrientedMethod):

    def __init__(self, A: Matrix, B: Row) -> None:
        super().__init__(A, B)
        self.method_name = "Jakobi"
```

```python
        if not is_convergent(self.A):
            raise ValueError("A is not convergent")

    def execute_method(self, eps=1e-3, max_iterations=10000):
        x = np.zeros_like(self.B, dtype=np.double)
        T = self.A - np.diag(np.diagonal(self.A))  # type: ignore

        for i in range(max_iterations):
            print_header(f"\nIteration {i}")
            print(f"x={x}")

            x_old = x.copy()

            x = (self.B - np.dot(T, x)) / np.diagonal(self.A)

            if np.linalg.norm(x - x_old, ord=np.inf) / np.linalg.norm(
                    x, ord=np.inf) < eps:
                break

        return x
```

Протокол роботи

```
>>> Jakobi method

Initial values

A:
0.55    0.0     -0.08   0.14
0.3     -0.73   -0.12   0.11
0.17    0.06    -0.58   0.12
0.21    -0.16   0.0     0.36

B:
0.48    -1.24   -1.15   -0.88

Iteration 0
x=[0. 0. 0. 0.]

Iteration 1
x=[ 0.87272727  1.69863014  1.98275862 -2.44444444]

Iteration 2
x=[ 1.78335075  1.36301131  1.90853123 -2.19858863]

Iteration 3
x=[ 1.70997256  1.78648867  2.19158564 -2.87894958]

Iteration 4
x=[ 1.92432689  1.60728361  2.07312191 -2.64793347]

Iteration 5
x=[ 1.84829171  1.74965857  2.16520788 -2.85262019]

Iteration 6
x=[ 1.91378811  1.67243061  2.11530117 -2.74498857]

Iteration 7
x=[ 1.87913181  1.72376924  2.14877791 -2.81751835]

Iteration 8
x=[ 1.90246327  1.69309476  2.12892476 -2.774485   ]

Iteration 9
x=[ 1.8886216   1.71243104  2.14149352 -2.80172813]

Iteration 10
x=[ 1.8973844   1.70057146  2.13380028 -2.78505992]

Iteration 11
x=[ 1.89202256  1.7079489   2.13859042 -2.79544247]

Iteration 12
x=[ 1.89536215  1.70339349  2.13563392 -2.78903587]

Iteration 13
x=[ 1.89330134  1.7062173   2.13746702 -2.79300859]


Resulting vectors
[ 1.89457921  1.70447043  2.13633316 -2.79055143]
```

Рис.1. Метод Якобі

```
>>> Seidel method

Initial values

A:
0.55    0.0     -0.08   0.14
0.3     -0.73   -0.12   0.11
0.17    0.06    -0.58   0.12
0.21    -0.16   0.0     0.36

B:
0.48    -1.24   -1.15   -0.88

Iteration 0
x=[0. 0. 0. 0.]

Iteration 1
x=[ 0.87272727  2.05728518  2.4513806  -2.03918638]

Iteration 2
x=[ 1.74835735  1.70689182  2.24988257 -2.70570098]

Iteration 3
x=[ 1.8887068   1.69725894  2.15212341 -2.79185277]

Iteration 4
x=[ 1.89641684  1.70351567  2.13720598 -2.79356953]


Resulting vectors
[ 1.89468402  1.70499704  2.13649614 -2.79190033]


Results verification
AX - B = [ 2.90474823e-04  2.68792422e-04  2.00303775e-04 -1.11022302e-16]
Norms: [1.23, 1.26, 0.9817915402727148]
```

Рис.2. Метод Зейделя

Висновки

Виконуючи лабораторну роботу №5, я ознайомився з наближеними методами розв’язування систем лінійних алгебраїчних рівнянь, навчився розв’язувати СЛАР методами Якобі та Зейделя, а також склав програму, яка їх розв’язує автоматично.