

Лабораторна робота № 4

з дисципліни «Чисельні методи»

на тему

«Обчислення власних значень та власних векторів матриць»

Виконав:
студент гр. ІП-93
Домінський Валентин
Викладач:
доц. Рибачук Л.В.

Зміст

Зміст	2
1 Постановка задачі	3
2 Розв'язок	4
3 Розв'язок у Mathcad.....	7
4 Лістинг програми	9
Висновок:	10

1 Постановка задачі

Створити програму, для приведення матриці A до нормальної форми Фробеніуса. Отримане характеристичне рівняння розв'язати довільним способом у Mathcad і отримати всі власні числа λ_i , $i = 1, \dots, m$ з точністю 5 знаків після коми. Знайти по одному власному вектору для кожного власного числа

Перевірити точність знайдених результатів, підставляючи у рівняння (1) знайдені власні числа та власні вектори

Знайти власні числа матриці A виключно за допомогою Mathcad і порівняти з отриманими раніше результатами.

2 Розв'язок

Матриця:

$$: \begin{bmatrix} 7.25 & 0.98 & 1.09 & 1.105 \\ 0.98 & 3.17 & 1.3 & 0.16 \\ 1.09 & 1.3 & 6.43 & 2.1 \\ 1.105 & 0.16 & 2.1 & 5.11 \end{bmatrix}$$

Нижче наведені результати виконання програми.

Проміжні матриці M, M⁻¹ та P:

M Matrix =

$$\begin{bmatrix} 1. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ -0.52619 & -0.07619 & 0.47619 & -2.43333 \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

M Matrix Inverted =

$$\begin{bmatrix} 1. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 1.105 & 0.16 & 2.1 & 5.11 \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

...

M Matrix =

$$\begin{bmatrix} 1. & 0. & 0. & 0. \\ -0.81939 & 0.3141 & -3.83599 & 9.62333 \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

M Matrix Inverted =

$$\begin{bmatrix} 1. & 0. & 0. & 0. \\ 2.60868 & 3.18368 & 12.2126 & -30.63764 \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

...

M Matrix =

$$\begin{bmatrix} 0.11862 & -1.90005 & 8.75301 & -12.21815 \\ 0. & 1. & 0. & 0. \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

M Matrix Inverted =

$$\begin{bmatrix} 8.43056 & 16.0185 & -73.79276 & 103.00583 \\ 0. & 1. & 0. & 0. \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

Final result as Frobenius Matrix =

```
[[ 21.96 -166.59148 516.81412 -552.28401]
 [ 1.    0.    0.    0.   ]
 [ 0.    1.    0.    0.   ]
 [ 0.    0.    1.    0.   ]]
```

Увесь вивід:

Start Matrix =

```
7.25    0.98    1.09    1.105
0.98    3.17    1.3     0.16
1.09    1.3     6.43    2.1
1.105   0.16    2.1     5.11
```

N = 4

Iteration - 1

M Matrix =

```
[[ 1.    0.    0.    0.   ]
 [ 0.    1.    0.    0.   ]
 [-0.52619 -0.07619 0.47619 -2.43333]
 [ 0.    0.    0.    1.   ]]
```

S Matrix =

```
[[ 1.    0.    0.    0.   ]
 [ 0.    1.    0.    0.   ]
 [-0.52619 -0.07619 0.47619 -2.43333]
 [ 0.    0.    0.    1.   ]]
```

M Matrix Inverted =

```
[[1.  0.  0.  0. ]
 [0.  1.  0.  0. ]
 [1.105 0.16 2.1  5.11 ]
 [0.  0.  0.  1.   ]]
```

Temporary result =

```
[[ 6.67645 0.89695 0.51905 -1.54733]
 [ 0.29595 3.07095 0.61905 -3.00333]
 [ 2.60868 3.18368 12.2126 -30.63764]
 [ 0.    0.    1.    0.   ]]
```

Iteration - 2

M Matrix =

```
[[ 1.    0.    0.    0.   ]
 [-0.81939 0.3141 -3.83599 9.62333]
 [ 0.    0.    1.    0.   ]
 [ 0.    0.    0.    1.   ]]
```

S Matrix =

```
[[ 1.    0.    0.    0.   ]
 [-0.81939 0.3141 -3.83599 9.62333]
 [-0.46376 -0.02393 0.76846 -3.16654]
 [ 0.    0.    0.    1.   ]]
```

M Matrix Inverted =

```
[[ 1.    0.    0.    0. ]
 [ 2.60868  3.18368 12.2126 -30.63764]
 [ 0.    0.    1.    0. ]
 [ 0.    0.    0.    1. ]]
```

Temporary result =

```
[[ 5.9415  0.28173 -2.92166  7.08433]
 [ 8.43056 16.0185 -73.79276 103.00583]
 [ 0.    1.    0.    0. ]
 [ 0.    0.    1.    0. ]]
```

Iteration - 3

M Matrix =

```
[[ 0.11862 -1.90005  8.75301 -12.21815]
 [ 0.    1.    0.    0. ]
 [ 0.    0.    1.    0. ]
 [ 0.    0.    0.    1. ]]
```

S Matrix =

```
[[ 0.11862 -1.90005  8.75301 -12.21815]
 [-0.09719  1.87099 -11.00813  19.63477]
 [-0.05501  0.85724 -3.29084  2.49976]
 [ 0.    0.    0.    1. ]]
```

M Matrix Inverted =

```
[[ 8.43056 16.0185 -73.79276 103.00583]
 [ 0.    1.    0.    0. ]
 [ 0.    0.    1.    0. ]
 [ 0.    0.    0.    1. ]]
```

Temporary result =

```
[[ 21.96  -166.59148  516.81412 -552.28401]
 [ 1.    0.    0.    0. ]
 [ 0.    1.    0.    0. ]
 [ 0.    0.    1.    0. ]]
```

Final result as Frobenius Matrix =

```
[[ 21.96  -166.59148  516.81412 -552.28401]
 [ 1.    0.    0.    0. ]
 [ 0.    1.    0.    0. ]
 [ 0.    0.    1.    0. ]]
```

3 Розв'язок у Mathcad

Нижче наведено розв'язок системи у Mathcad

$$k := 3 \cdot (3 - 4) + 9 = 6 \quad t := 9 \quad a := 0.11 \cdot t = 0.99 \quad b := 0.02 \cdot k = 0.12 \quad g := b \quad d := 0.015 \cdot t = 0.135$$

$$matrix := \begin{bmatrix} 6.26 + a & 1.10 - b & 0.97 + g & 1.24 - d \\ 1.10 - b & 4.16 - a & 1.30 & 0.16 \\ 0.97 + g & 1.30 & 5.44 + a & 2.10 \\ 1.24 - d & 0.16 & 2.10 & 6.1 - a \end{bmatrix} = \begin{bmatrix} 7.25 & 0.98 & 1.09 & 1.105 \\ 0.98 & 3.17 & 1.3 & 0.16 \\ 1.09 & 1.3 & 6.43 & 2.1 \\ 1.105 & 0.16 & 2.1 & 5.11 \end{bmatrix} \quad \begin{matrix} \text{ORIGIN} := 1 \\ N := 4 \end{matrix}$$

$$fMatrix := \begin{bmatrix} 21.96 & -166.59148 & 516.81412 & -552.28401 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Знаходжу коефіцієнти для
polyroots

$$v := \lambda^4 - fMatrix_{1,1} \lambda^3 - fMatrix_{1,2} \lambda^2 - fMatrix_{1,3} \lambda - fMatrix_{1,4} \xrightarrow{\text{coeffs}, \lambda} \begin{bmatrix} 552.28401 \\ -516.81412 \\ 166.59148 \\ -21.96 \\ 1.0 \end{bmatrix} = \begin{bmatrix} 552.28401 \\ -516.81412 \\ 166.59148 \\ -21.96 \\ 1 \end{bmatrix}$$

Розв'язую характеристичне рівняння $r := \text{polyroots}(v) = \begin{bmatrix} 2.42737 \\ 3.94118 \\ 6.05117 \\ 9.54028 \end{bmatrix}$

Знаходжу власні вектори матриці P

$$y1 := \begin{bmatrix} r_1^3 \\ r_1^2 \\ r_1^1 \\ r_1^0 \\ r_1 \end{bmatrix} = \begin{bmatrix} 14.30233 \\ 5.89211 \\ 2.42737 \\ 1 \end{bmatrix} \quad y2 := \begin{bmatrix} r_2^3 \\ r_2^2 \\ r_2^1 \\ r_2^0 \\ r_2 \end{bmatrix} = \begin{bmatrix} 61.21793 \\ 15.5329 \\ 3.94118 \\ 1 \end{bmatrix} \quad y3 := \begin{bmatrix} r_3^3 \\ r_3^2 \\ r_3^1 \\ r_3^0 \\ r_3 \end{bmatrix} = \begin{bmatrix} 221.57402 \\ 36.6167 \\ 6.05117 \\ 1 \end{bmatrix} \quad y4 := \begin{bmatrix} r_4^3 \\ r_4^2 \\ r_4^1 \\ r_4^0 \\ r_4 \end{bmatrix} = \begin{bmatrix} 868.32692 \\ 91.01693 \\ 9.54028 \\ 1 \end{bmatrix}$$

Вписую матрицю подібності S

$$SMatrix := \begin{bmatrix} 0.11862 & -1.90005 & 8.75301 & -12.21815 \\ -0.09719 & 1.87099 & -11.00813 & 19.63477 \\ -0.05501 & 0.85724 & -3.29084 & 2.49976 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Знаходжу власні вектори матриці A (matrix)

$$x1 := SMatrix \cdot y1 = \begin{bmatrix} -0.47015 \\ 2.54803 \\ -1.22413 \\ 1 \end{bmatrix}$$

$$x2 := SMatrix \cdot y2 = \begin{bmatrix} 0.02743 \\ -0.63812 \\ -0.52221 \\ 1 \end{bmatrix}$$

$$x3 := SMatrix \cdot y3 = \begin{bmatrix} -2.54262 \\ -0.00263 \\ 1.78683 \\ 1 \end{bmatrix}$$

$$x4 := SMatrix \cdot y4 = \begin{bmatrix} 1.35223 \\ 0.51321 \\ 1.36092 \\ 1 \end{bmatrix}$$

Порівнюю Результати

Множу власний вектор матриці matrix на матрицю matrix

$$z1 := matrix \cdot x1 = \begin{bmatrix} -1.14079 \\ 6.18515 \\ -2.9712 \\ 2.42749 \end{bmatrix}$$

$$z2 := matrix \cdot x2 = \begin{bmatrix} 0.10927 \\ -2.51485 \\ -2.05748 \\ 3.94156 \end{bmatrix}$$

$$z3 := matrix \cdot x3 = \begin{bmatrix} -15.38394 \\ -0.01723 \\ 10.81444 \\ 6.05233 \end{bmatrix}$$

$$z4 := matrix \cdot x4 = \begin{bmatrix} 12.89503 \\ 4.88124 \\ 12.99179 \\ 9.54425 \end{bmatrix}$$

Множу власний вектор матриці matrix на власний вектор матриці P

$$k1 := y1_3 \cdot x1 = \begin{bmatrix} -1.14122 \\ 6.18502 \\ -2.97142 \\ 2.42737 \end{bmatrix}$$

$$k2 := y2_3 \cdot x2 = \begin{bmatrix} 0.10809 \\ -2.51496 \\ -2.05812 \\ 3.94118 \end{bmatrix}$$

$$k3 := y3_3 \cdot x3 = \begin{bmatrix} -15.38584 \\ -0.01592 \\ 10.81242 \\ 6.05117 \end{bmatrix}$$

$$k4 := y4_3 \cdot x4 = \begin{bmatrix} 12.90069 \\ 4.89613 \\ 12.98351 \\ 9.54028 \end{bmatrix}$$

Різниця результатів

$$\delta1 := \sqrt{\frac{1}{N} \sum_{k=1}^N (k1_k - z1_k)^2}$$

$$\delta2 := \sqrt{\frac{1}{N} \sum_{k=1}^N (k2_k - z2_k)^2}$$

$$\delta1 = 0.000259801028667$$

$$\delta2 = 0.000700415730698$$

$$\delta3 := \sqrt{\frac{1}{N} \sum_{k=1}^N (k3_k - z3_k)^2}$$

$$\delta4 := \sqrt{\frac{1}{N} \sum_{k=1}^N (k4_k - z4_k)^2}$$

$$\delta3 = 0.001639353004499$$

$$\delta4 = 0.009189242554905$$

У технічних розрахунках точність вимірювань характеризують відносною похибкою. Результат вважають гарним, якщо відносна похибка не перевищує 0,1 %. Отже Наш результат є гарним

4 Лістинг програми

Lab4.py

```
# region Starting Values
import numpy as np
np.set_printoptions(suppress=True)

matrix = [[7.25, 0.98, 1.09, 1.105],
          [0.98, 3.17, 1.3, 0.16],
          [1.09, 1.3, 6.43, 2.1],
          [1.105, 0.16, 2.1, 5.11]]

N = len(matrix)
rounding = 5

# endregion Starting Values

# region Identity

def Identity(N):
    matrixForIdentity = [[0 for x in range(N)] for y in range(N)]
    for i in range(0, N):
        matrixForIdentity[i][i] = 1
    return matrixForIdentity

# endregion Identity

# region Dot

def Dot(matrix1: list, matrix2: list, N: int) -> list:
    res = [[0 for x in range(N)] for y in range(N)]

    for i in range(len(matrix1)):
        for j in range(len(matrix2[0])):
            for k in range(len(matrix2)):

                # resulted matrix
                res[i][j] += matrix1[i][k] * matrix2[k][j]
    return res

# endregion Dot

# region Prints

# print matrix
def PrintMatrix(matrixName, matrix):
    print("\n", matrixName, "=")
    for i in matrix:
        for j in i:
            print(round(j, rounding), end=" \t")
            if len(str(j)) <= 6:
                print(end=" \t")
        print()

# print matrix
def PrintMatrixAsNp(matrixName, matrix):
```

```

print("\n", matrixName, "=")
npMatrix = np.array(matrix)
print(npMatrix.round(rounding))

# print additional parametrs
def PrintParameters():
    print("\nN =", N)

# just printing
def PrintAll():
    PrintMatrix("Start Matrix", matrix)

    PrintParameters()

# endregion Prints

PrintAll()

S_matrix = Identity(N)

for x in range(N - 1, 0, -1):
    M_matrix = Identity(N)

    M_matrixInverted = Identity(N)

    # Fill matrix b and minus one b
    for y in range(N):
        if y == x - 1:
            M_matrix[x - 1][y] = 1 / matrix[x][x - 1]
        else:
            M_matrix[x - 1][y] = matrix[x][y] / matrix[x][x - 1] * (-1)
            M_matrixInverted[x - 1][y] = matrix[x][y]

    print("\nIteration -", N - x)

    PrintMatrixAsNp("M Matrix", M_matrix)

    S_matrix = Dot(S_matrix, M_matrix, N)

    PrintMatrixAsNp("S Matrix", S_matrix)

    PrintMatrixAsNp("M Matrix Inverted", M_matrixInverted)

    matrix = Dot(M_matrixInverted, Dot(matrix, M_matrix, N), N)

    PrintMatrixAsNp("Temporary result", matrix)

PrintMatrixAsNp("Final result as Frobenius Matrix", matrix)

```

Висновок:

Я навчився обчислювати власні значення та власні вектори матриць, отримав більше знань для роботи з MathCad та на практиці з'ясував, як порівнювати результати методом середньоквадратичної похибки.