МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №2 з дисципліни

«Методи та технології штучного інтелекту»

«Моделювання функції з двох змінних засобами нечіткої математики »

Перевірив:
Шимкович В.М.

Виконав:
студент 3 курсу
групи ІП-11 ФІОТ
Прищепа В.С.

Київ-2023

Лабораторна робота №2

Моделювання функції з двох змінних засобами нечіткої математики

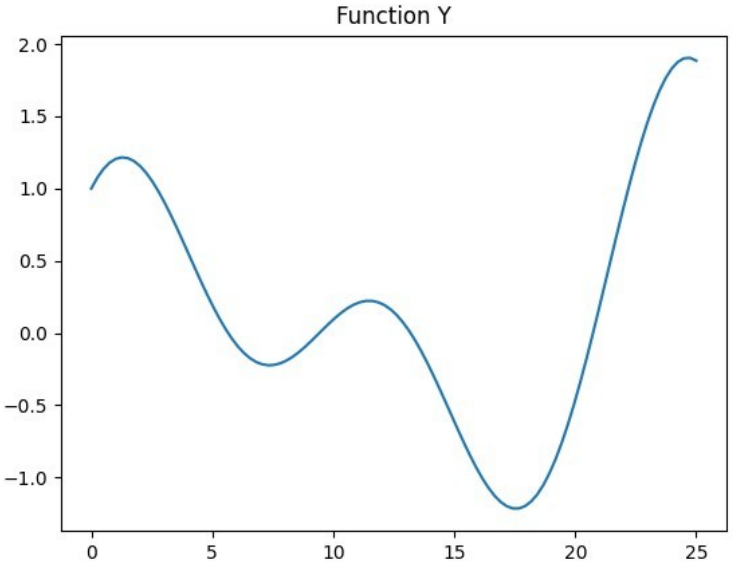
Мета роботи: Промоделювати засобами нечіткої логіки функцію з двох змінних. Провести дослідження форми функції приналежності на якість моделювання.

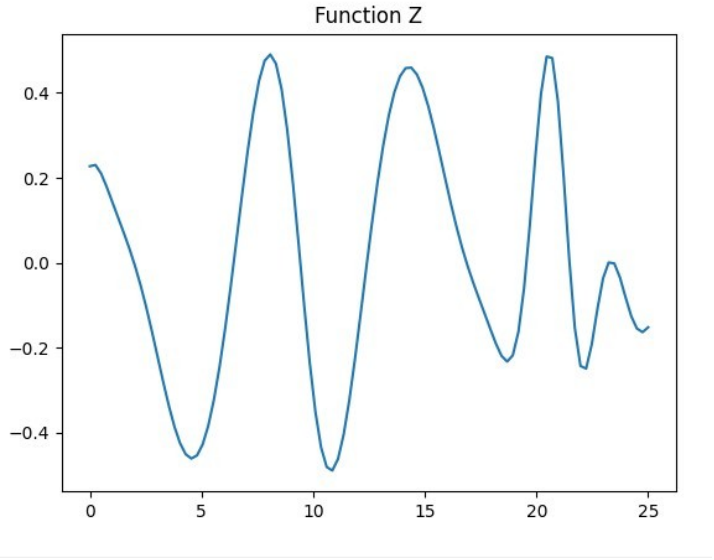
Завдання. За допомогою пакетів моделювання або мови програмування високого рівня:
1. Побудувати нечітку модель функції двох змінних згідно з варіантом з додатку А, що містить 6 функцій приналежності для вхідних змінних і не менше 9 для вихідної. Варіант з додатку А – 74 = 14.

14.	$y = \cos(x/2) + \sin(x/3)$ $z = 0.5\sin(x+y)\cdot\cos(y)$	14.	17.	3.

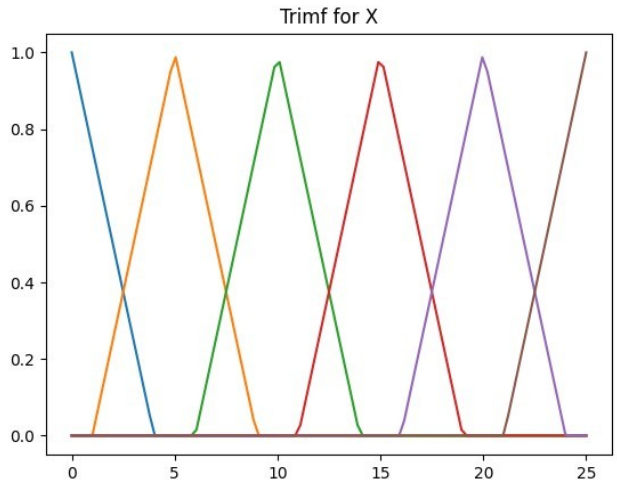
2. Дослідити вплив форми функції приналежності (трикутник, трапеція, Гауса) на якість моделювання (порівняти відносні помилки моделювання).
3. Дослідити можливість зменшення числа правил за рахунок виключення деяких (перевірити достатність використання правил, що представляють тільки діагональ таблиці).
4. Зробити висновки.

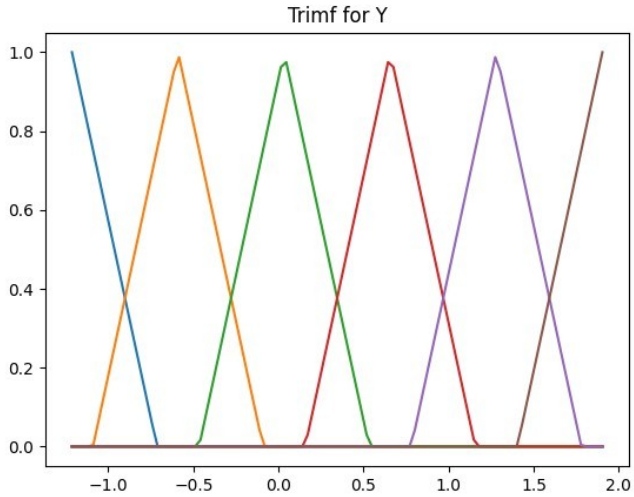
Хід роботи.
Оберемо значення x від 0 до 25. Виведемо графіки функцій y та z:

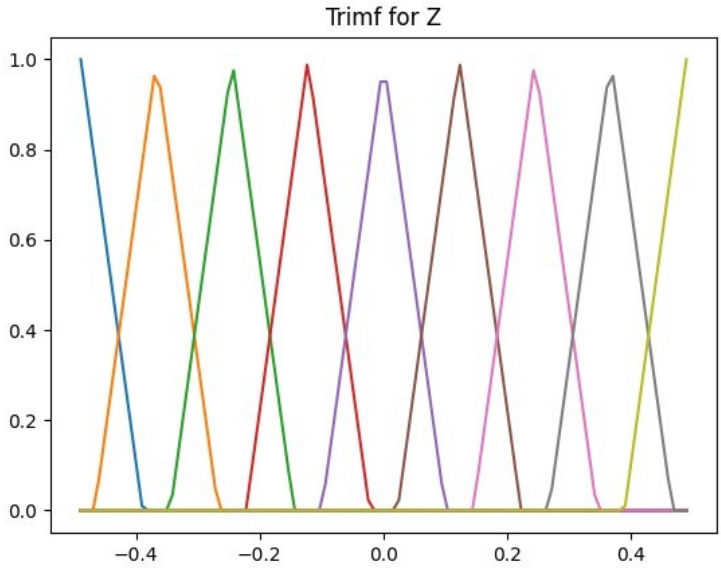




Тепер виведемо трикутні функції приналежності для змінних. По 6 для вхідних та 9 для вихідної:







Таблиця значень функції по максимумам вхідних функцій приналежності та таблиця імен функцій:

Values' table:

y\x	0	5	10	15	20	25
-1.21	-0.16	-0.1	0.1	0.16	-0.01	-0.17
-0.59	-0.23	-0.4	0.01	0.4	0.22	-0.27
0.03	0.02	-0.47	-0.29	0.31	0.46	-0.05
0.66	0.24	-0.23	-0.37	0.02	0.38	0.2
1.28	0.14	-0	-0.14	-0.08	0.09	0.13
1.9	-0.15	-0.1	0.1	0.15	-0.01	-0.16

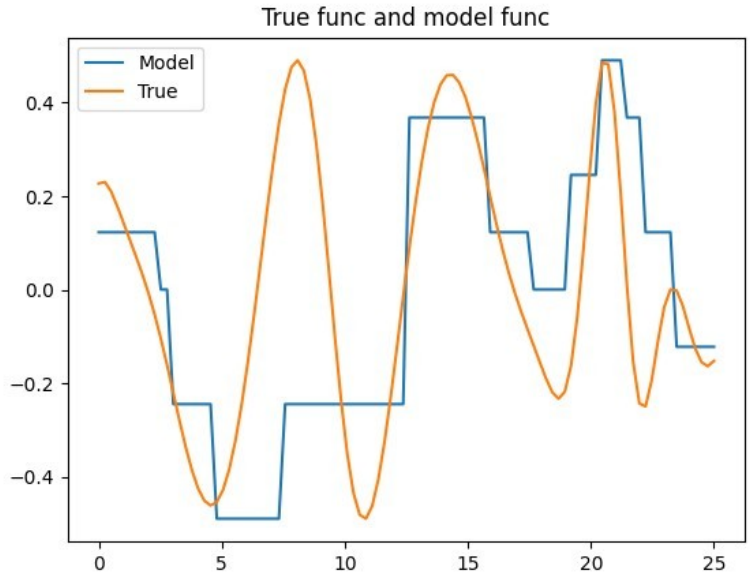
Function's name's table:

y\x	mx1	mx2	mx3	mx4	mx5	mx6
my1	mf4	mf4	mf6	mf6	mf5	mf4
my2	mf3	mf2	mf5	mf8	mf7	mf3
my3	mf5	mf1	mf3	mf8	mf9	mf5
my4	mf7	mf3	mf2	mf5	mf8	mf7
my5	mf6	mf5	mf4	mf4	mf6	mf6
my6	mf4	mf4	mf6	mf6	mf5	mf4

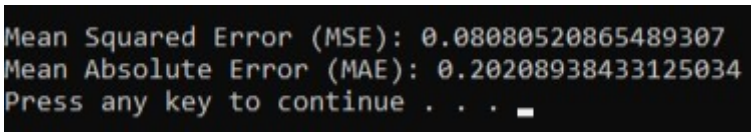
Тепер складемо правила:

```
Rules:
if (x is mx1) and (y is my1) then (z is mf4)
if (x is mx2) and (y is my1) then (z is mf4)
if (x is mx3) and (y is my1) then (z is mf6)
if (x is mx4) and (y is my1) then (z is mf6)
if (x is mx5) and (y is my1) then (z is mf5)
if (x is mx6) and (y is my1) then (z is mf4)
if (x is mx1) and (y is my2) then (z is mf3)
if (x is mx2) and (y is my2) then (z is mf2)
if (x is mx3) and (y is my2) then (z is mf5)
if (x is mx4) and (y is my2) then (z is mf8)
if (x is mx5) and (y is my2) then (z is mf7)
if (x is mx6) and (y is my2) then (z is mf3)
if (x is mx1) and (y is my3) then (z is mf5)
if (x is mx2) and (y is my3) then (z is mf1)
if (x is mx3) and (y is my3) then (z is mf3)
if (x is mx4) and (y is my3) then (z is mf8)
if (x is mx5) and (y is my3) then (z is mf9)
if (x is mx6) and (y is my3) then (z is mf5)
if (x is mx1) and (y is my4) then (z is mf7)
if (x is mx2) and (y is my4) then (z is mf3)
if (x is mx3) and (y is my4) then (z is mf2)
if (x is mx4) and (y is my4) then (z is mf5)
if (x is mx5) and (y is my4) then (z is mf8)
if (x is mx6) and (y is my4) then (z is mf7)
if (x is mx1) and (y is my5) then (z is mf6)
if (x is mx2) and (y is my5) then (z is mf5)
if (x is mx3) and (y is my5) then (z is mf4)
if (x is mx4) and (y is my5) then (z is mf4)
if (x is mx5) and (y is my5) then (z is mf6)
if (x is mx6) and (y is my5) then (z is mf6)
if (x is mx1) and (y is my6) then (z is mf4)
if (x is mx2) and (y is my6) then (z is mf4)
if (x is mx3) and (y is my6) then (z is mf6)
if (x is mx4) and (y is my6) then (z is mf6)
if (x is mx5) and (y is my6) then (z is mf5)
if (x is mx6) and (y is my6) then (z is mf4)
```

Побудуємо модель:



Виведемо похибки:



Бачимо, що похибки невеликі, тому модель працює добре.

Лістинг коду програми:

lab2_triangle.py

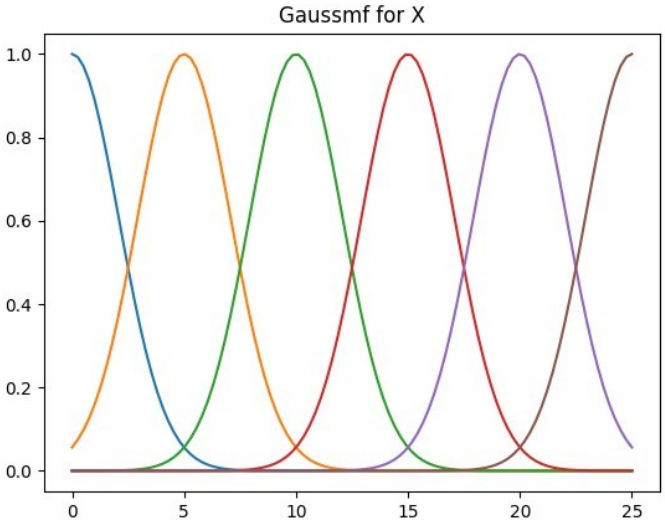
```
import numpy as np
import matplotlib.pyplot as plt
import skfuzzy as fuzz
from tabulate import tabulate
from sklearn.metrics import mean_squared_error, mean_absolute_error

def get_y(x):
    return np.cos(x/2) + np.sin(x/3)

def get_z(x, y):
    return 0.5 * np.sin(x + y) * np.cos(y)

# Get values
x_values = np.linspace(0, 25, 100)
y_values = get_y(x_values)
z_values = get_z(x_values, y_values)

# Plot function Y
plt.plot(x_values, y_values)
plt.title("Function Y")
plt.show()

# Plot function Z
plt.plot(x_values, z_values)
plt.title("Function Z")
plt.show()

# Numbers of membership functions for input and output
mf_num_input = 6
mf_num_output = 9

# Means
x_means = np.linspace(min(x_values), max(x_values), mf_num_input)
y_means = np.linspace(min(y_values), max(y_values), mf_num_input)
z_means = np.linspace(min(z_values), max(z_values), mf_num_output)

# Triangle fuzzy membership functions
```

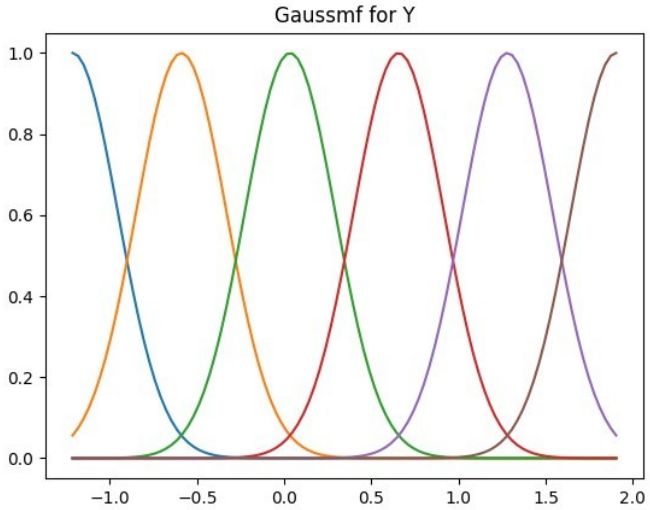
```
x_mf_triangle = [fuzz.trimf(x_values, [x_means[i] - 4, x_means[i], x_means[i] + 4])
    for i in range(mf_num_input)]
y_mf_triangle = [fuzz.trimf(np.linspace(min(y_values), max(y_values), 100),
    [y_means[i] - 0.5, y_means[i], y_means[i] + 0.5]) for i in range(mf_num_input)]
z_mf_triangle = [fuzz.trimf(np.linspace(min(z_values), max(z_values), 100),
    [z_means[i] - 0.1, z_means[i], z_means[i] + 0.1]) for i in range(mf_num_output)]

# Plot trifm for X
for i in range(mf_num_input):
    plt.plot(x_values, x_mf_triangle[i])
plt.title("Trimf for X")
plt.show()

# Plot trifm for Y
for i in range(mf_num_input):
    plt.plot(np.linspace(min(y_values), max(y_values), 100), y_mf_triangle[i])
plt.title("Trimf for Y")
plt.show()

# Plot trifm for Z
for i in range(mf_num_output):
     plt.plot(np.linspace(min(z_values), max(z_values), 100), z_mf_triangle[i])
plt.title("Trimf for Z")
plt.show()

# Calculate value for triangular mf
def calculate_trimf(x, a, b, c):
 if a <= x < b:
    return (x - a) / (b - a)
 elif b <= x <= c:
    return (c - x) / (c - b)
 else:
    return 0.0

# Get number of best function for value
def get_fun_num(value, means, diff):
    best_func_value = -float("inf")
    best_index = -1
    for index, mean in enumerate(means):
        if calculate_trimf(value, mean - diff, mean, mean + diff) > best_func_value:
            best_func_value = calculate_trimf(value, mean - diff, mean, mean + diff)
            best_index = index
    return best_index

# Table of values
```

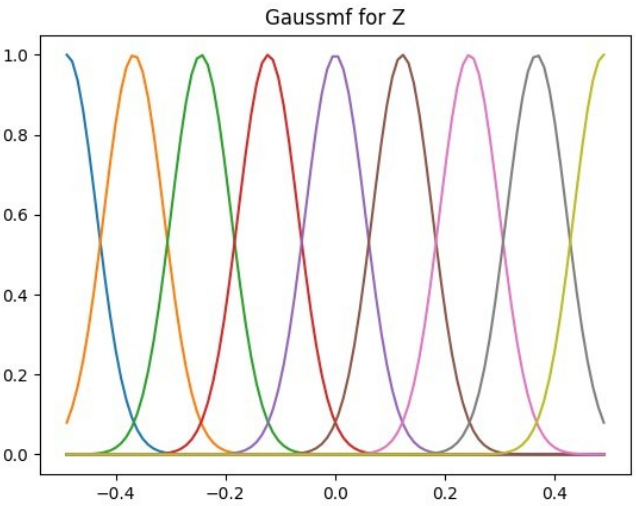
```
print("Values' table:")
table = [["y\\x"] + [str(x) for x in x_means]]
for y_value in y_means:
    row = [round(y_value, 2)]
    for x in x_means:
        z = get_z(x, y_value)
        row.append(round(z, 2))
    table.append(row)
print(tabulate(table, tablefmt="grid"))

# Table of function's name
rules = {}
print("Function's name's table:")
table = [["y\\x"] + ["mx" + str(i) for i in range(1, mf_num_input + 1)]]
for i in range(mf_num_input):
    row = ["my" + str(i + 1)]
    for j in range(mf_num_input):
        z = get_z(x_means[j], y_means[i])
        best_func = get_fun_num(z, z_means, 0.1)
        row.append("mf" + str(best_func + 1))
        rules[(j, i)] = best_func
    table.append(row)
print(tabulate(table, tablefmt="grid"))

# Print rules
print("\nRules:")
for rule in rules:
    print(f"if (x is mx{rule[0] + 1}) and (y is my{rule[1] + 1}) then (z is mf{rules[rule]
+ 1})")

# Model
z_output = []
for x in x_values:
    best_x_func = get_fun_num(x, x_means, 4)
    best_y_func = get_fun_num(get_y(x), y_means, 0.5)
    best_z_func = rules[(best_x_func, best_y_func)]
    z_output.append(z_means[best_z_func])

# Plot results
plt.plot(x_values, z_output, label="Model")
plt.plot(x_values, z_values, label="True")
plt.title("True func and model func")
plt.legend()
plt.show()
```

```
# Get scores
mse = mean_squared_error(z_values, z_output)
mae = mean_absolute_error(z_values, z_output)
print(f"\nMean Squared Error (MSE): {mse}")
print(f"Mean Absolute Error (MAE): {mae}")
```

Повторимо дії на Гауссівській функції приналежності:







Values' table:

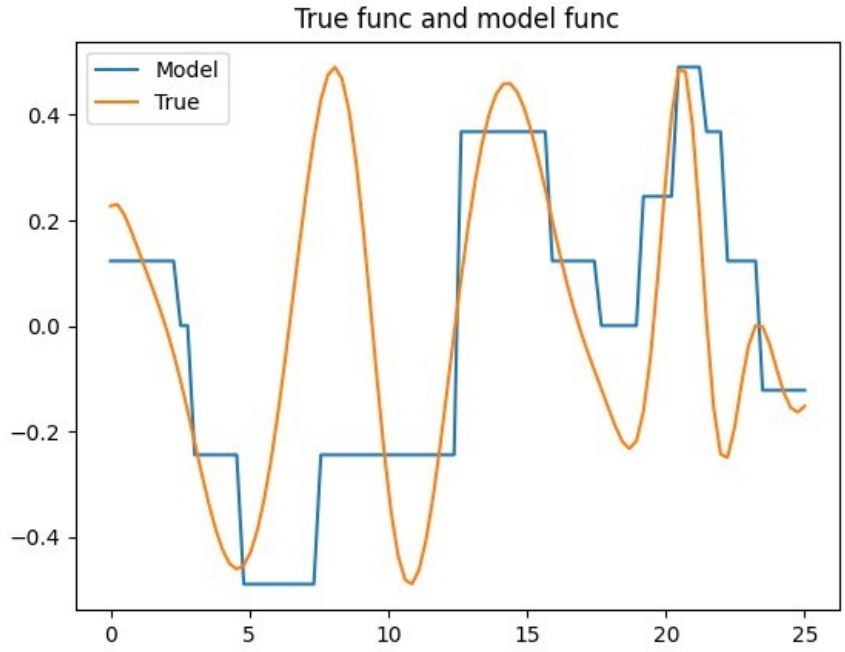
y\x	0	5	10	15	20	25
-1.21	-0.16	-0.1	0.1	0.16	-0.01	-0.17
-0.59	-0.23	-0.4	0.01	0.4	0.22	-0.27
0.03	0.02	-0.47	-0.29	0.31	0.46	-0.05
0.66	0.24	-0.23	-0.37	0.02	0.38	0.2
1.28	0.14	-0	-0.14	-0.08	0.09	0.13
1.9	-0.15	-0.1	0.1	0.15	-0.01	-0.16

Function's name's table:

y\x	mx1	mx2	mx3	mx4	mx5	mx6
my1	mf4	mf4	mf6	mf6	mf5	mf4
my2	mf3	mf2	mf5	mf8	mf7	mf3
my3	mf5	mf1	mf3	mf8	mf9	mf5
my4	mf7	mf3	mf2	mf5	mf8	mf7
my5	mf6	mf5	mf4	mf4	mf6	mf6
my6	mf4	mf4	mf6	mf6	mf5	mf4

Rules:

if (x is mx1) and (y is my1) then (z is mf4)
if (x is mx2) and (y is my1) then (z is mf4)
if (x is mx3) and (y is my1) then (z is mf6)
if (x is mx4) and (y is my1) then (z is mf6)
if (x is mx5) and (y is my1) then (z is mf5)
if (x is mx6) and (y is my1) then (z is mf4)
if (x is mx1) and (y is my2) then (z is mf3)
if (x is mx2) and (y is my2) then (z is mf2)
if (x is mx3) and (y is my2) then (z is mf5)
if (x is mx4) and (y is my2) then (z is mf8)
if (x is mx5) and (y is my2) then (z is mf7)
if (x is mx6) and (y is my2) then (z is mf3)
if (x is mx1) and (y is my3) then (z is mf5)
if (x is mx2) and (y is my3) then (z is mf1)
if (x is mx3) and (y is my3) then (z is mf3)
if (x is mx4) and (y is my3) then (z is mf8)
if (x is mx5) and (y is my3) then (z is mf9)
if (x is mx6) and (y is my3) then (z is mf5)
if (x is mx1) and (y is my4) then (z is mf7)
if (x is mx2) and (y is my4) then (z is mf3)
if (x is mx3) and (y is my4) then (z is mf2)
if (x is mx4) and (y is my4) then (z is mf5)
if (x is mx5) and (y is my4) then (z is mf8)
if (x is mx6) and (y is my4) then (z is mf7)
if (x is mx1) and (y is my5) then (z is mf6)
if (x is mx2) and (y is my5) then (z is mf5)
if (x is mx3) and (y is my5) then (z is mf4)
if (x is mx4) and (y is my5) then (z is mf4)
if (x is mx5) and (y is my5) then (z is mf6)
if (x is mx6) and (y is my5) then (z is mf6)
if (x is mx1) and (y is my6) then (z is mf4)
if (x is mx2) and (y is my6) then (z is mf4)
if (x is mx3) and (y is my6) then (z is mf6)
if (x is mx4) and (y is my6) then (z is mf6)
if (x is mx5) and (y is my6) then (z is mf5)
if (x is mx6) and (y is my6) then (z is mf4)




```
Mean Squared Error (MSE): 0.08080520865489307
Mean Absolute Error (MAE): 0.20208938433125034
Press any key to continue . . .
```

Бачимо, що похибки невеликі, тому ця модель теж працює добре**.**

Лістинг коду програми:

lab2_Gaussian.py

```
import numpy as np
import matplotlib.pyplot as plt
import skfuzzy as fuzz
from tabulate import tabulate
from sklearn.metrics import mean_squared_error, mean_absolute_error

def get_y(x):
    return np.cos(x/2) + np.sin(x/3)

def get_z(x, y):
    return 0.5 * np.sin(x + y) * np.cos(y)

# Get values
x_values = np.linspace(0, 25, 100)
y_values = get_y(x_values)
z_values = get_z(x_values, y_values)

# Plot function Y
plt.plot(x_values, y_values)
plt.title("Function Y")
```

```
plt.show()

# Plot function Z
plt.plot(x_values, z_values)
plt.title("Function Z")
plt.show()

# Numbers of membership functions
mf_num_input = 6
mf_num_output = 9

# Means
x_means = np.linspace(min(x_values), max(x_values), mf_num_input)
y_means = np.linspace(min(y_values), max(y_values), mf_num_input)
z_means = np.linspace(min(z_values), max(z_values), mf_num_output)

# Sigmas
x_sigma = (max(x_values) - min(x_values)) / mf_num_input / 2
y_sigma = (max(y_values) - min(y_values)) / mf_num_input / 2
z_sigma = (max(z_values) - min(z_values)) / mf_num_output / 2

# Gaussian fuzzy membership functions
x_mf_gaussian = [fuzz.gaussmf(x_values, x_means[i], x_sigma) for i in
    range(mf_num_input)]
y_mf_gaussian = [fuzz.gaussmf(np.linspace(min(y_values), max(y_values), 100),
    y_means[i], y_sigma) for i in range(mf_num_input)]
z_mf_gaussian = [fuzz.gaussmf(np.linspace(min(z_values), max(z_values), 100),
    z_means[i], z_sigma) for i in range(mf_num_output)]

# Plot gaussfm for X
for i in range(mf_num_input):
    plt.plot(x_values, x_mf_gaussian[i])
plt.title("Gaussmf for X")
plt.show()

# Plot gaussfm for Y
for i in range(mf_num_input):
    plt.plot(np.linspace(min(y_values), max(y_values), 100), y_mf_gaussian[i])
plt.title("Gaussmf for Y")
plt.show()

# Plot gaussfm for Z
for i in range(mf_num_output):
    plt.plot(np.linspace(min(z_values), max(z_values), 100), z_mf_gaussian[i])
plt.title("Gaussmf for Z")
```

```
plt.show()

# Get number of best function for value
def get_fun_num(value, means, sigma):
    best_func_value = -float("inf")
    best_index = -1
    for index, mean in enumerate(means):
        if fuzz.gaussmf(value, mean, sigma) > best_func_value:
            best_func_value = fuzz.gaussmf(value, mean, sigma)
            best_index = index
    return best_index

# Table of values
print("Values' table:")
table = [["y\\x"] + [str(x) for x in x_means]]
for y_value in y_means:
    row = [round(y_value, 2)]
    for x in x_means:
      z = get_z(x, y_value)
      row.append(round(z, 2))
    table.append(row)
print(tabulate(table, tablefmt="grid"))

# Table of function's name
rules = {}
print("Function's name's table:")
table = [["y\\x"] + ["mx" + str(i) for i in range(1, mf_num_input + 1)]]
for i in range(mf_num_input):
    row = ["my" + str(i + 1)]
    for j in range(mf_num_input):
        z = get_z(x_means[j], y_means[i])
        best_func = get_fun_num(z, z_means, z_sigma)
        row.append("mf" + str(best_func + 1))
        rules[(j, i)] = best_func
    table.append(row)
print(tabulate(table, tablefmt="grid"))

# Print rules
print("\nRules:")
for rule in rules.keys():
    print(f"if (x is mx{rule[0] + 1}) and (y is my{rule[1] + 1}) then (z is mf{rules[rule]
+ 1})")

# Model
z_output = []
```

```
for x in x_values:
    best_x_func = get_fun_num(x, x_means, x_sigma)
    best_y_func = get_fun_num(get_y(x), y_means, y_sigma)
    best_z_func = rules[(best_x_func, best_y_func)]
    z_output.append(z_means[best_z_func])

# Plot results
plt.plot(x_values, z_output, label="Model")
plt.plot(x_values, z_values, label="True")
plt.title("True func and model func")
plt.legend()
plt.show()
# Get scores
mse = mean_squared_error(z_values, z_output)
mae = mean_absolute_error(z_values, z_output)
print(f"\nMean Squared Error (MSE): {mse}")
print(f"Mean Absolute Error (MAE): {mae}")
```

Якщо зменшити число правил (викинути деякі із існуючих), то тоді можуть бути значеня вихідної функції z, які не описуються жодним правилом. Експериментально було встановлено, що це призводить до помилки програми на етапі моделювання. Якщо зменшити число термінів приналежності, то можна зменшити число правил без можливих значень z, які не будуть описані, але тоді постраждає точність моделі.

Висновок: Під час виконання лабораторної роботи я побудував нечітку модель функції двох змінних. Як функції приналежності я обрав трикутну та Гаусівську. Для вхідних змінних я брав 6 функцій приналежності, для вихідної – 9. По результатах моделювання можна побачити, що для обраних функцій приналежності результати є однаковими.