Name: **Harsh Pramod Padyal**

Roll No : **40**

Div: **D20B**

Experiment 06

AIM: To implement Fuzzy Membership Functions.

Theory

Fuzzy membership functions are fundamental to fuzzy logic systems, providing a way to represent imprecise or "fuzzy" data. Unlike classical sets where an element is either in or out of a set, fuzzy sets allow for partial membership. The membership value, ranging from 0 to 1, indicates the degree to which an element belongs to a fuzzy set. A value of 0 means no membership, and a value of 1 means full membership.

Different Types of Membership Functions

1. **Singleton Membership Function** This is the simplest type of membership function. It assigns a membership value of 1 to a single, specific point (x_0) and a value of 0 to all other points. It is useful when a particular value has full membership in a fuzzy set.

   Equation:

$$\mu(x) = \begin{cases} 1 & \text{if } x = x_0 \\ 0 & \text{if } x \neq x_0 \end{cases}$$

2. **Triangular Membership Function** This function has a triangular shape and is defined by three points: a left endpoint (a), a peak (b), and a right endpoint (c). It is a popular choice due to its simplicity and ease of calculation. The membership value increases linearly from 0 at point a to 1 at point b, and then decreases linearly back to 0 at point c.

   Equation:

$$\mu(x) = \begin{cases} 0 & \text{if } x \leq a \text{ or } x \geq c \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{if } b \leq x \leq c \end{cases}$$

3. **Trapezoidal Membership Function** Similar to the triangular function, but it has a flat top, meaning all values within a specific range have a full membership of 1. This function is defined by four points: the left endpoint (a), the start of the flat top (b), the end of the flat top (c), and the right endpoint (d).

Equation:

$$\mu(x) = \begin{cases} 0 & \text{if } x \leq a \text{ or } x \geq d \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ 1 & \text{if } b \leq x \leq c \\ \frac{d-x}{d-c} & \text{if } c \leq x \leq d \end{cases}$$

4. **Gaussian Membership Function** This function is characterized by a smooth, bell-shaped curve. It is useful in systems that require smooth transitions from a membership value of 0 to 1 and back to 0. The shape is determined by the center of the curve (c) and the standard deviation (σ), which controls the width of the bell curve.

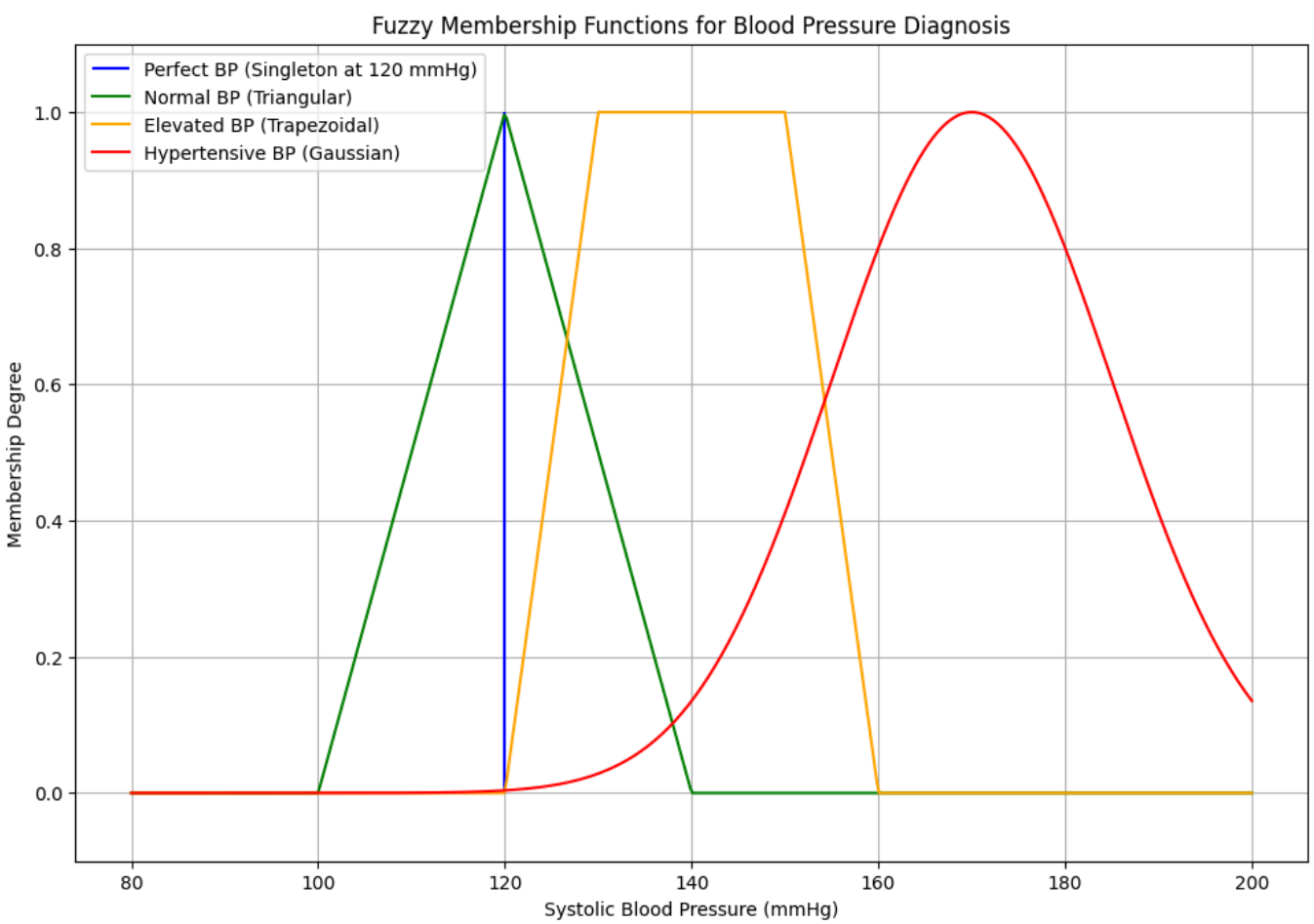
   Equation:

$$\mu(x) = e^{-\frac{(x-c)^2}{2\sigma^2}}$$

```
import numpy as np
import matplotlib.pyplot as plt

# Singleton Membership Function: Assigns a membership value of 1 to a single point and 0 to all others
def singleton_mf(x, x0):
    """
    Args:
        x: The input array or value.
        x0: The specific point with full membership.

    Returns:
        An array of membership values.
    """
    return np.where(x == x0, 1, 0)

# Triangular Membership Function: Defined by a triangular shape with three points (a, b, c).
def triangular_mf(x, a, b, c):
    """
    Args:
        x: The input array or value.
        a: The left endpoint.
        b: The peak (membership value of 1).
        c: The right endpoint.

    Returns:
        An array of membership values.
    """
    return np.maximum(np.minimum((x - a) / (b - a), (c - x) / (c - b)), 0)

# Trapezoidal Membership Function: Similar to triangular but with a flat top (b, c) where membership i
def trapezoidal_mf(x, a, b, c, d):
    """
    Args:
        x: The input array or value.
        a: Left endpoint.
        b: Start of the top.
        c: End of the top.
        d: Right endpoint.
```

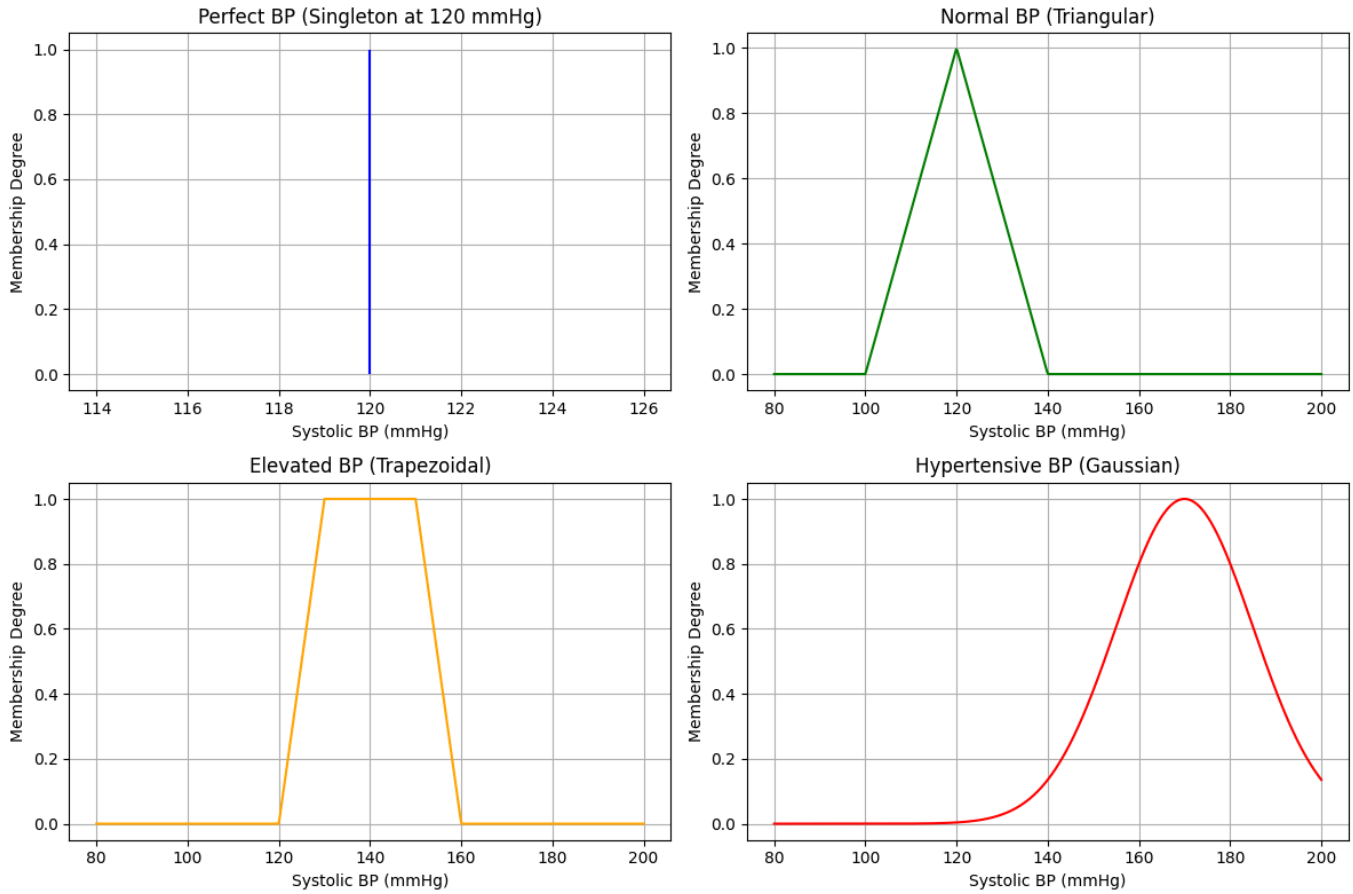
```
    Returns:
        An array of membership values.
    """
    return np.maximum(np.minimum(np.minimum((x - a) / (b - a), 1), (d - x) / (d - c)), 0)

# Gaussian Membership Function: Characterized by a bell-shaped curve.
def gaussian_mf(x, c, sigma):
    """
    Args:
        x: The input array or value.
        c: The center of the curve.
        sigma: The standard deviation, which controls the width of the curve.

    Returns:
        An array of membership values.
    """
    return np.exp(-((x - c)**2) / (2 * sigma**2))
```

```
# Medical Diagnosis Example: Blood Pressure

# The universe of discourse (input space) is the blood pressure range (e.g., from 80 to 200 mmHg)
bp = np.linspace(80, 200, 500)

# Defining fuzzy sets for different blood pressure categories
# Singleton: A specific reading of 120 mmHg is considered "Perfectly Normal".
perfect_bp = singleton_mf(bp, 120)

# Triangular: Normal blood pressure, peaking at 120 mmHg.
normal_bp = triangular_mf(bp, 100, 120, 140)

# Trapezoidal: Elevated blood pressure, with a range of 130-150 mmHg considered fully elevated.
elevated_bp = trapezoidal_mf(bp, 120, 130, 150, 160)

# Gaussian: Hypertensive blood pressure, centered around 170 mmHg.
hypertensive_bp = gaussian_mf(bp, 170, 15)

# --- Plotting the Membership Functions ---
plt.figure(figsize=(12, 8))

# Plotting the membership functions for the medical diagnosis example
plt.vlines(120, 0, 1, colors='blue', linestyles='solid', label="Perfect BP (Singleton at 120 mmHg)")
plt.plot(bp, normal_bp, label="Normal BP (Triangular)", color='green')
plt.plot(bp, elevated_bp, label="Elevated BP (Trapezoidal)", color='orange')
plt.plot(bp, hypertensive_bp, label="Hypertensive BP (Gaussian)", color='red')

# Adding plot details
plt.title('Fuzzy Membership Functions for Blood Pressure Diagnosis')
plt.xlabel('Systolic Blood Pressure (mmHg)')
plt.ylabel('Membership Degree')
plt.ylim(-0.1, 1.1)
plt.legend()
plt.grid(True)
plt.show()
```




```
# Universe of discourse
bp = np.linspace(80, 200, 500)

# Defining fuzzy sets
perfect_bp = singleton_mf(bp, 120)
normal_bp = triangular_mf(bp, 100, 120, 140)
elevated_bp = trapezoidal_mf(bp, 120, 130, 150, 160)
hypertensive_bp = gaussian_mf(bp, 170, 15)

# Plotting
fig, axs = plt.subplots(2, 2, figsize=(12, 8))

# Perfect BP (Singleton)
axs[0, 0].vlines(120, 0, 1, colors='blue')
axs[0, 0].set_title("Perfect BP (Singleton at 120 mmHg)")
axs[0, 0].set_xlabel("Systolic BP (mmHg)")
axs[0, 0].set_ylabel("Membership Degree")
axs[0, 0].grid(True)
```

```
# Normal BP (Triangular)
axs[0, 1].plot(bp, normal_bp, color='green')
axs[0, 1].set_title("Normal BP (Triangular)")
axs[0, 1].set_xlabel("Systolic BP (mmHg)")
axs[0, 1].set_ylabel("Membership Degree")
axs[0, 1].grid(True)

# Elevated BP (Trapezoidal)
axs[1, 0].plot(bp, elevated_bp, color='orange')
axs[1, 0].set_title("Elevated BP (Trapezoidal)")
axs[1, 0].set_xlabel("Systolic BP (mmHg)")
axs[1, 0].set_ylabel("Membership Degree")
axs[1, 0].grid(True)

# Hypertensive BP (Gaussian)
axs[1, 1].plot(bp, hypertensive_bp, color='red')
axs[1, 1].set_title("Hypertensive BP (Gaussian)")
axs[1, 1].set_xlabel("Systolic BP (mmHg)")
axs[1, 1].set_ylabel("Membership Degree")
axs[1, 1].grid(True)

plt.tight_layout()
plt.show()
```



Conclusion

We successfully implemented and understood the concept of Fuzzy Membership Functions by using Python to create and visualize Singleton, Triangular, Trapezoidal, and Gaussian types. This experiment helped us learn how to represent imprecise data and make flexible, human-like decisions in fuzzy logic systems.

Start coding or generate with AI.