Laporan Praktikum Kecerdasan Komputasional

Nama : Alif As'ad Ramadhan

NRP : 5054231007

```
!pip install scikit-fuzzy
import numpy as np
import skfuzzy as fuzz
from matplotlib import pyplot as plt
```

```
Collecting scikit-fuzzy
  Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl.metadata (2.6
kB)
Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl (920 kB)
   ---------------------------------------- 0.0/920.8 kB ? eta -:--:--
   - -------------------------------------- 30.7/920.8 kB ? eta
-:--:--
   - -------------------------------------- 41.0/920.8 kB 487.6 kB/s
eta 0:00:02
   - -------------------------------------- 41.0/920.8 kB 487.6 kB/s
eta 0:00:02
   - -------------------------------------- 41.0/920.8 kB 487.6 kB/s
eta 0:00:02
   --- ------------------------------------ 92.2/920.8 kB 403.5 kB/s
eta 0:00:03
   ------ --------------------------------- 153.6/920.8 kB 610.0 kB/s
eta 0:00:02
   --------- ------------------------------ 235.5/920.8 kB 758.5 kB/s
eta 0:00:01
   ------------------ --------------------- 450.6/920.8 kB 1.3 MB/s
eta 0:00:01
   ------------------------- ------------ 624.6/920.8 kB 1.6 MB/s
eta 0:00:01
   ------------------------------ -------- 727.0/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   ------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
```

```
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------- ------- 757.8/920.8 kB 1.8 MB/s
eta 0:00:01
   -------------------------------------  911.4/920.8 kB 860.7 kB/s
eta 0:00:01
   -------------------------------------- 920.8/920.8 kB 844.7 kB/s
eta 0:00:00
Installing collected packages: scikit-fuzzy
Successfully installed scikit-fuzzy-0.5.0
```

Fuzzy Logic

Fuzzy logic adalah sistem logika yang bisa menangani ketidakpastian atau `"ambiguitas"`, seperti yang sering kita hadapi dalam kehidupan sehari-hari. Berbeda dengan logika biner yang hanya mengenal nilai `0` atau `1` **(benar atau salah)**, fuzzy logic bisa punya nilai di antara, misalnya `0.3` atau `0.7`, yang menunjukkan tingkat kebenaran yang **"sebagian benar"** atau **"cukup benar".**

Contohnya, kalau kita menyebut suhu **"hangat"**, ini bisa berarti berbeda-beda tergantung orang. Fuzzy logic memungkinkan kita untuk mendefinisikan suhu **sebagai dingin**, **hangat**, atau **panas** dengan berbagai tingkat keanggotaan, tidak perlu tegas `100% dingin` atau `100% panas`. Ini membantu kita membuat keputusan yang lebih fleksibel, seperti dalam kontrol otomatis pada AC, di mana suhu bisa diatur dengan mempertimbangkan derajat kenyamanan, bukan hanya satu nilai tegas.

Keanggotaan (Membership function)

Keanggotaan (Membership function) dalam fuzzy logic adalah cara untuk menunjukkan seberapa kuat suatu nilai menjadi bagian dari suatu kelompok `(fuzzy set)`. Membership function mengubah nilai input menjadi derajat keanggotaan dengan rentang antara `0` hingga `1`, di mana:

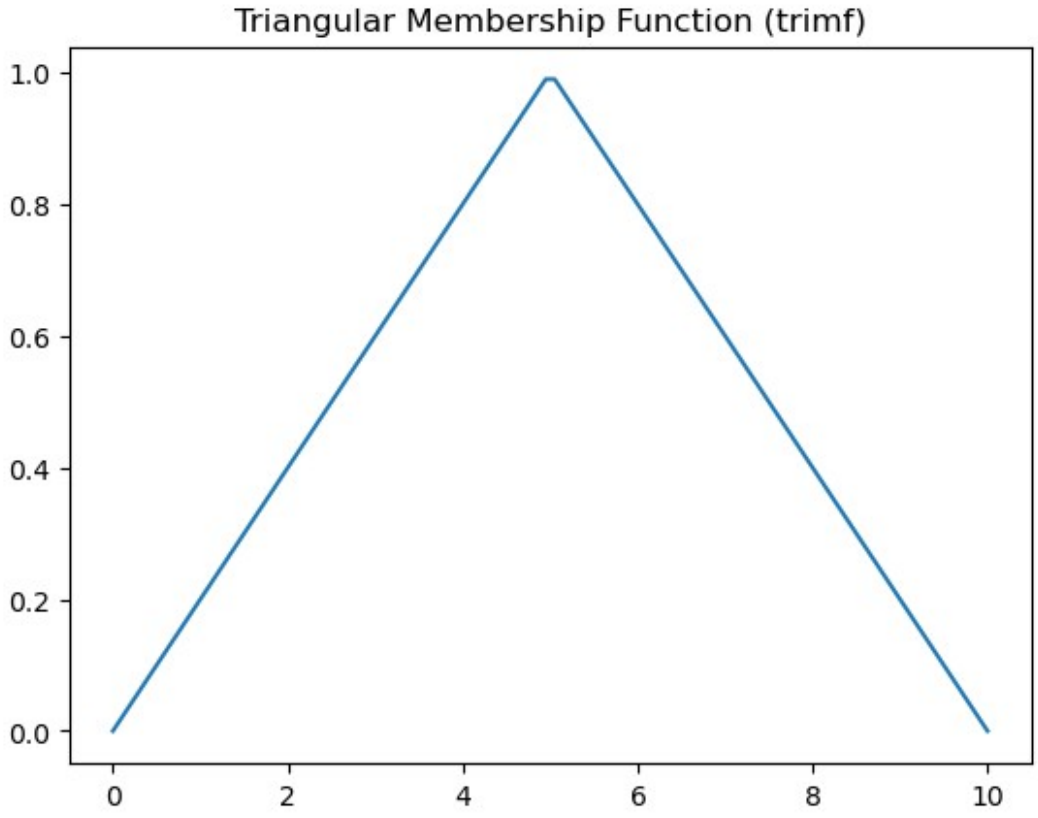
- `0` berarti tidak termasuk sama sekali dalam set.

- `1` berarti sepenuhnya termasuk dalam set.
- Nilai di antara `0` dan `1` menunjukkan bahwa nilai tersebut hanya sebagian termasuk.

Misalnya, jika kita punya fuzzy set "Tinggi", seseorang dengan tinggi 175 cm mungkin punya derajat keanggotaan 0.8 untuk "Tinggi", artinya cukup tinggi, tetapi tidak sepenuhnya. Membership function ini membantu merepresentasikan konsep yang tidak bisa dibagi secara tegas, seperti "agak tinggi" atau "cukup panas".

1. Triangular Membership Function (trimf)

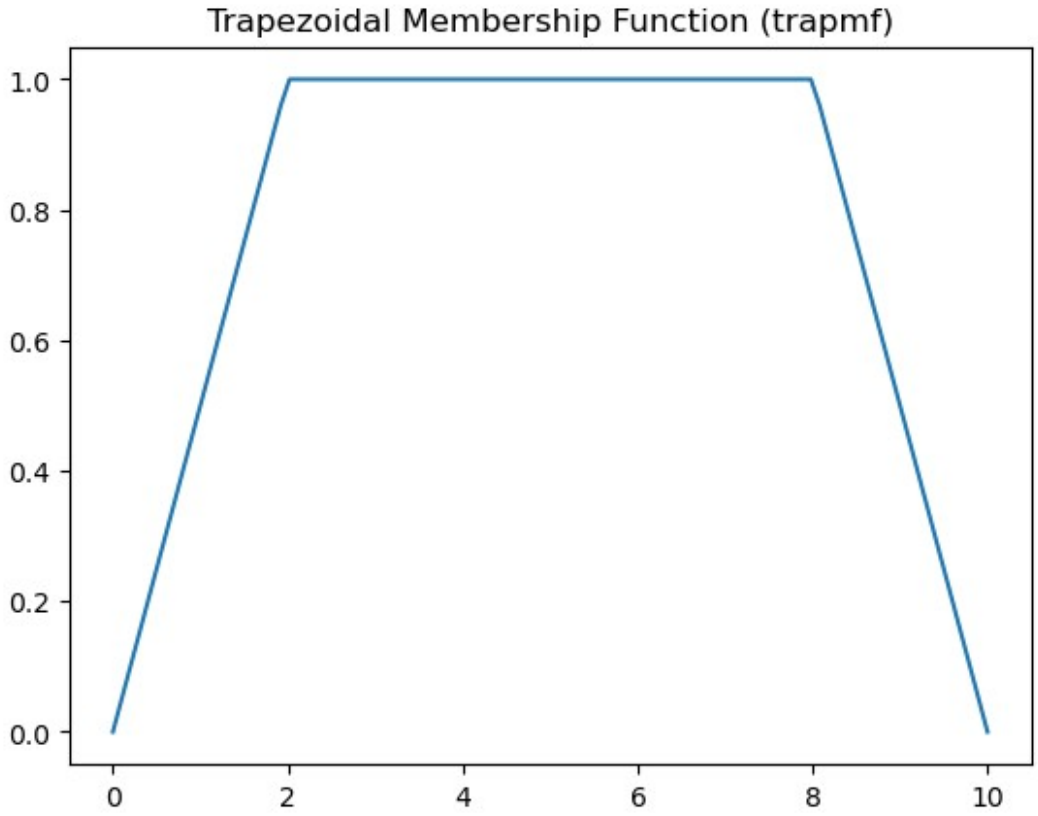
```
import numpy as np
import skfuzzy as fuzz
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 100)
trimf = fuzz.trimf(x, [0, 5, 10])
plt.plot(x, trimf)
plt.title("Triangular Membership Function (trimf)")
plt.show()
```



Definisi: Fungsi ini berbentuk segitiga, dengan tiga titik yang mendefinisikan batas awal, puncak, dan akhir (misalnya, `[0, 5, 10]`).

2. Trapezoidal Membership Function (trapmf)

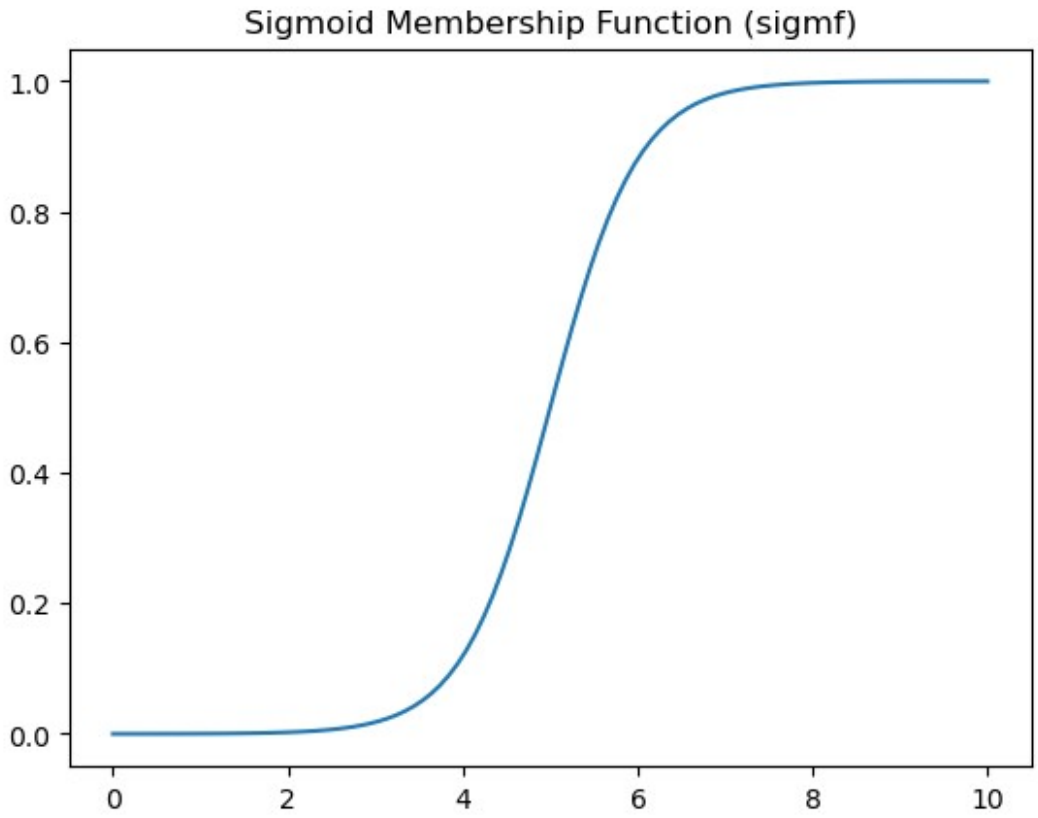
```
trapmf = fuzz.trapmf(x, [0, 2, 8, 10])
plt.plot(x, trapmf)
plt.title("Trapezoidal Membership Function (trapmf)")
plt.show()
```



Definisi: Fungsi ini berbentuk trapesium dengan empat titik yang mendefinisikan awal, awal penuh, akhir penuh, dan akhir (misalnya, `[0, 2, 8, 10]`). Ada area datar di tengah yang mewakili keanggotaan penuh.

3. Sigmoid Membership Function (sigmf)

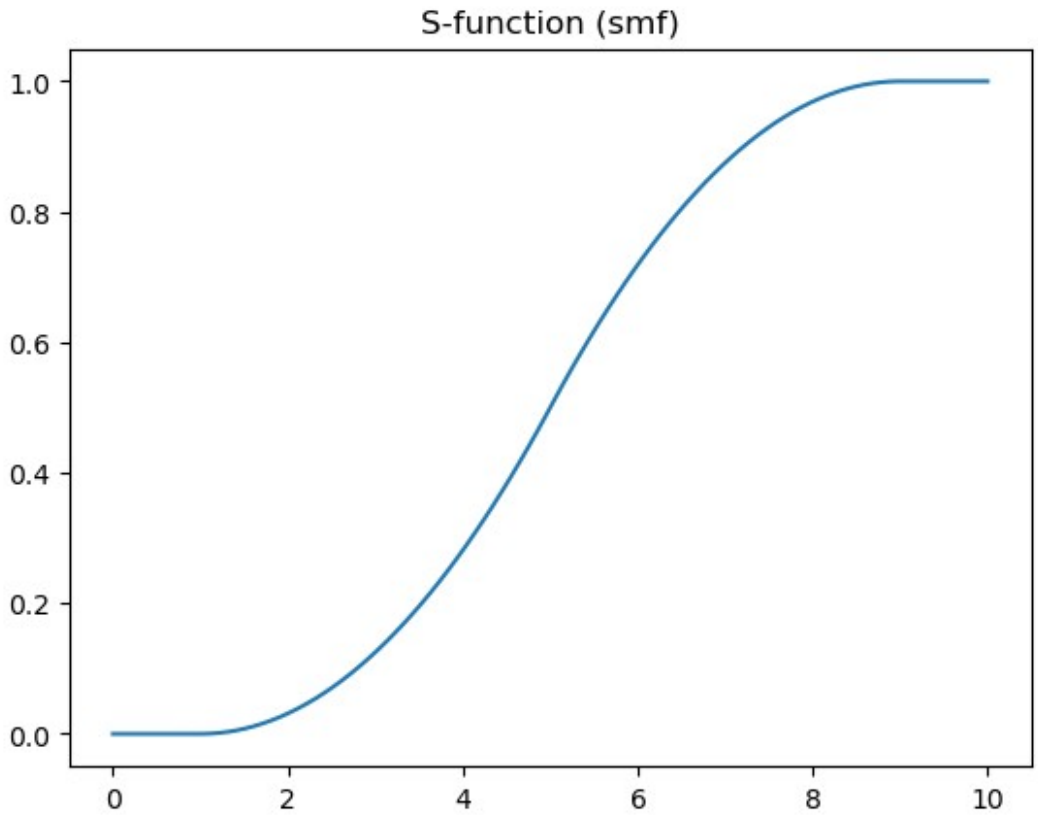
```
sigmf = fuzz.sigmf(x, 5.0, 2.0)
plt.plot(x, sigmf)
plt.title("Sigmoid Membership Function (sigmf)")
plt.show()
```



Definisi: Kurva berbentuk S yang halus, ditentukan oleh pusat (`misalnya 5.0`) dan lebar (`2.0`). Biasanya digunakan untuk menggambarkan perubahan bertahap.

4. S-function (smf)

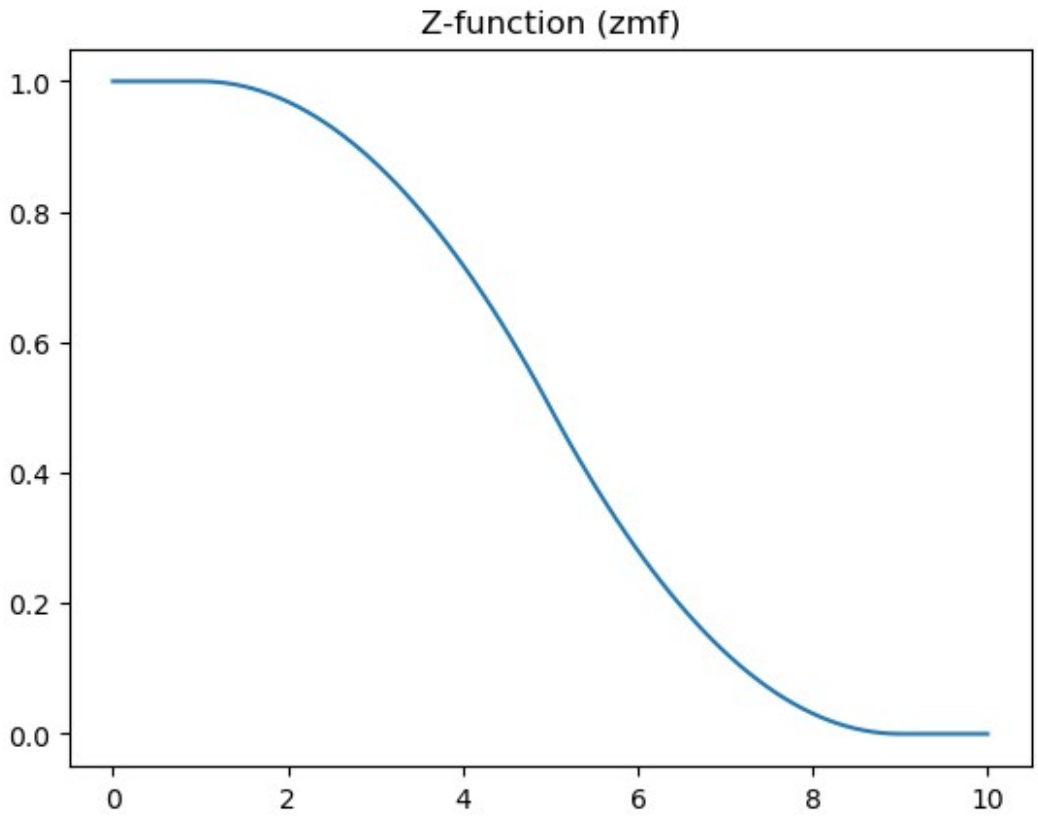
```
smf = fuzz.smf(x, 1.0, 9.0)
plt.plot(x, smf)
plt.title("S-function (smf)")
plt.show()
```



Definisi: Mirip dengan sigmoid, namun menggunakan dua titik untuk menggambarkan kenaikan halus dari satu titik ke titik lainnya (misalnya (`1.0, 9.0`)).

5. Z-function (zmf)

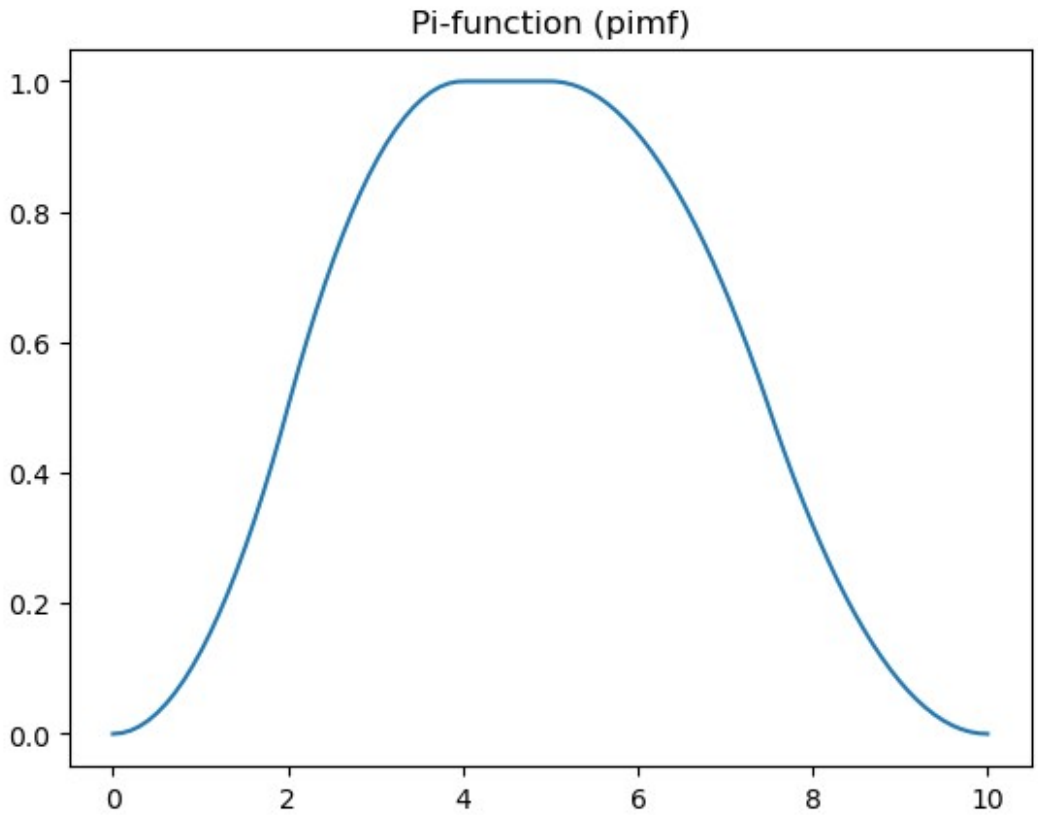
```
zmf = fuzz.zmf(x, 1.0, 9.0)
plt.plot(x, zmf)
plt.title("Z-function (zmf)")
plt.show()
```



Definisi: Kebalikan dari S-function, dimulai dengan keanggotaan penuh lalu menurun dengan halus menuju 0.

6. Pi-function (pimf)

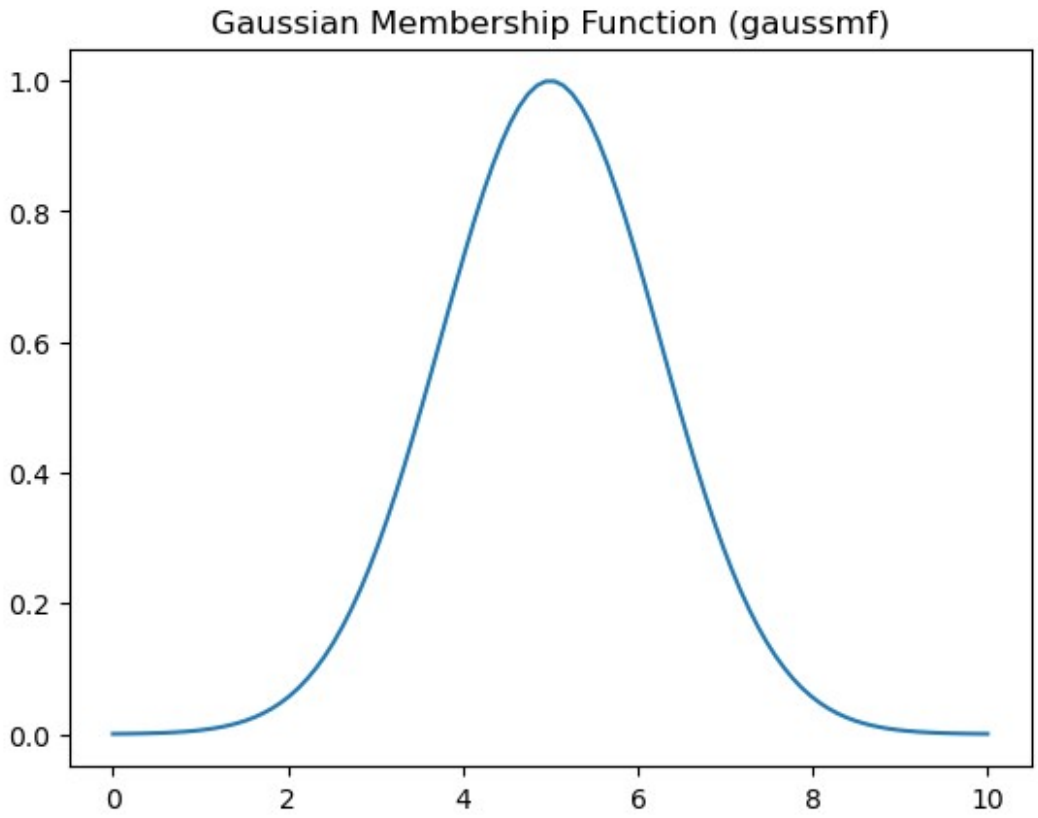
```
pimf = fuzz.pimf(x, 0.0, 4.0, 5.0, 10.0)
plt.plot(x, pimf)
plt.title("Pi-function (pimf)")
plt.show()
```



Definisi: Fungsi berbentuk lonceng, ditentukan oleh empat titik (`0.0, 4.0, 5.0, 10.0`) dengan transisi halus di kedua sisi.

7. Gaussian Membership Function (gaussmf)

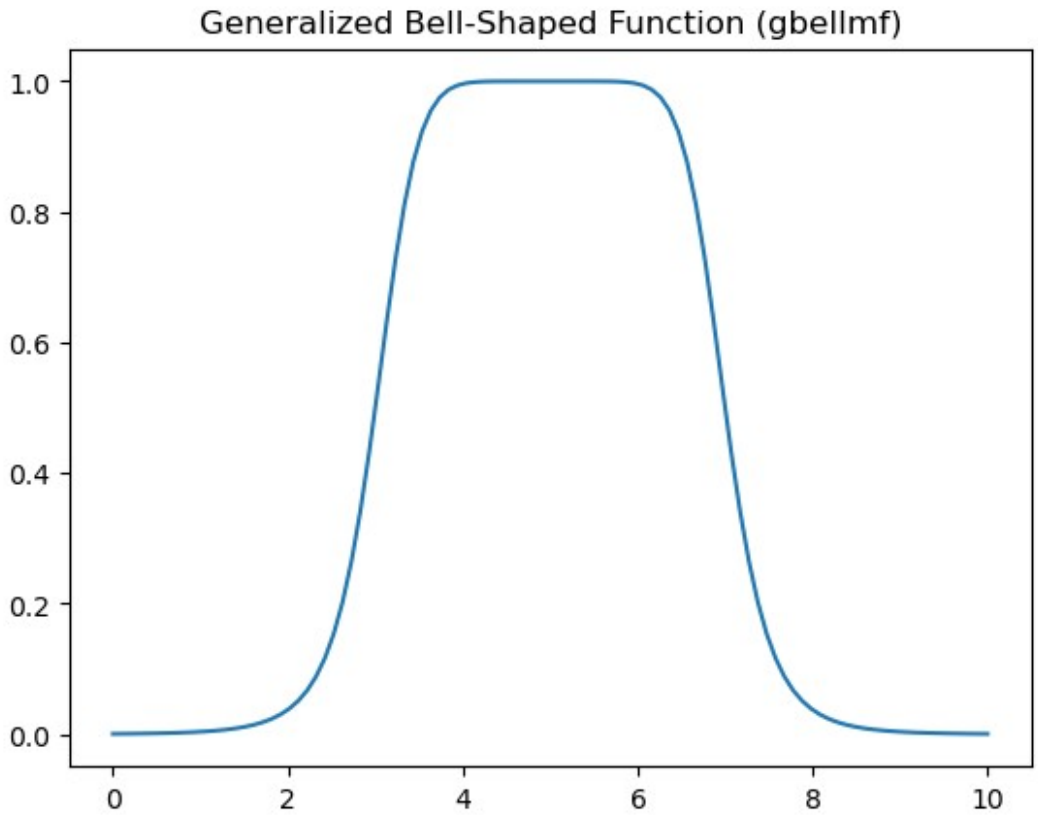
```
gaussmf = fuzz.gaussmf(x, 5.0, 1.25)
plt.plot(x, gaussmf)
plt.title("Gaussian Membership Function (gaussmf)")
plt.show()
```



Definisi: Kurva simetris dan halus, dikendalikan oleh nilai tengah (`mean`, `misalnya 5.0`) dan standar deviasi (`1.25`), membentuk distribusi normal.

8. Generalized Bell-Shaped Function (gbellmf)

```
gbellmf = fuzz.gbellmf(x, 2.0, 4.0, 5.0)
plt.plot(x, gbellmf)
plt.title("Generalized Bell-Shaped Function (gbellmf)")
plt.show()
```

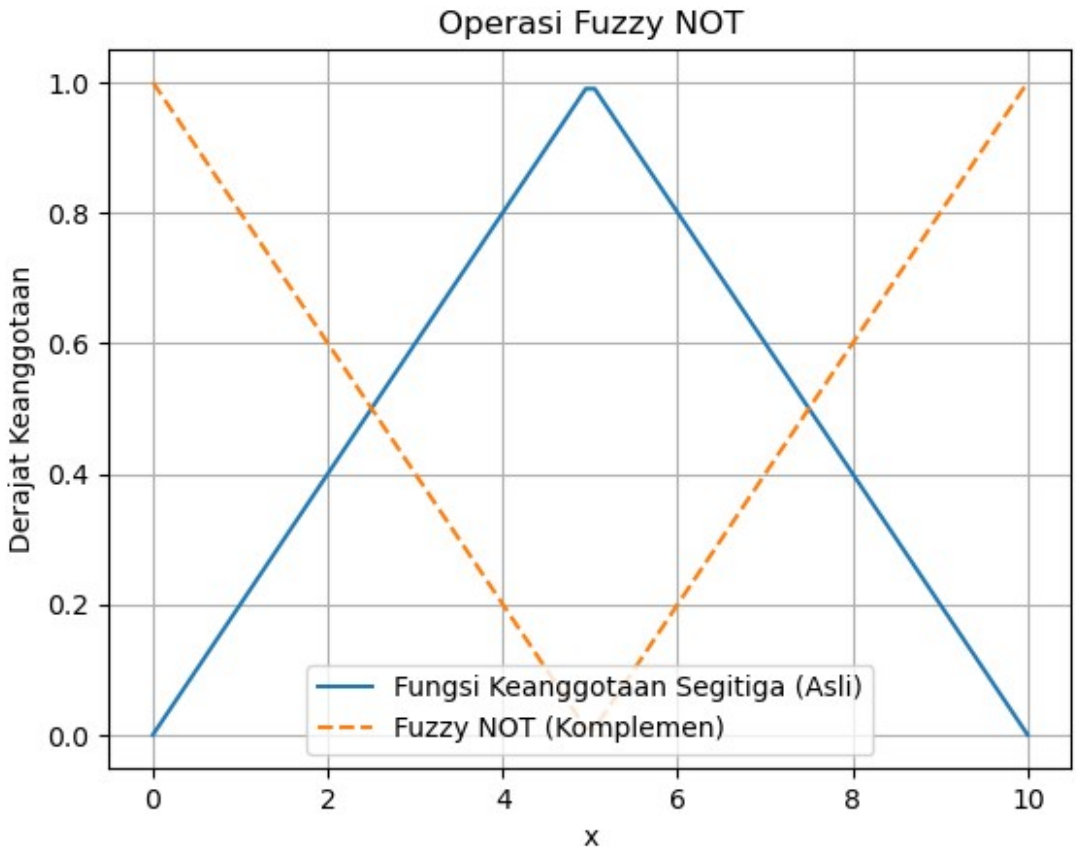


Definisi: Fungsi berbentuk lonceng yang lebih fleksibel, dikendalikan oleh lebar, kemiringan, dan pusatnya.

Fuzzy Operation

```
x = np.linspace(0, 10, 100)

# Definisikan dua fungsi keanggotaan untuk demonstrasi (triangular dan trapezoidal)
trimf1 = fuzz.trimf(x, [0, 5, 10])  # Fungsi keanggotaan segitiga
trapmf1 = fuzz.trapmf(x, [0, 2, 8, 10])  # Fungsi keanggotaan trapesium

# Fuzzy NOT: Komplemen dari fungsi keanggotaan segitiga
fuzzy_not = 1 - trimf1  # Menghitung komplemen secara manual

# Fuzzy AND: Irisan antara fungsi keanggotaan segitiga dan trapesium
fuzzy_and = np.minimum(trimf1, trapmf1)  # Menghitung nilai minimum di setiap titik

# Fuzzy OR: Gabungan antara fungsi keanggotaan segitiga dan trapesium
fuzzy_or = np.maximum(trimf1, trapmf1)  # Menghitung nilai maksimum di setiap titik
```

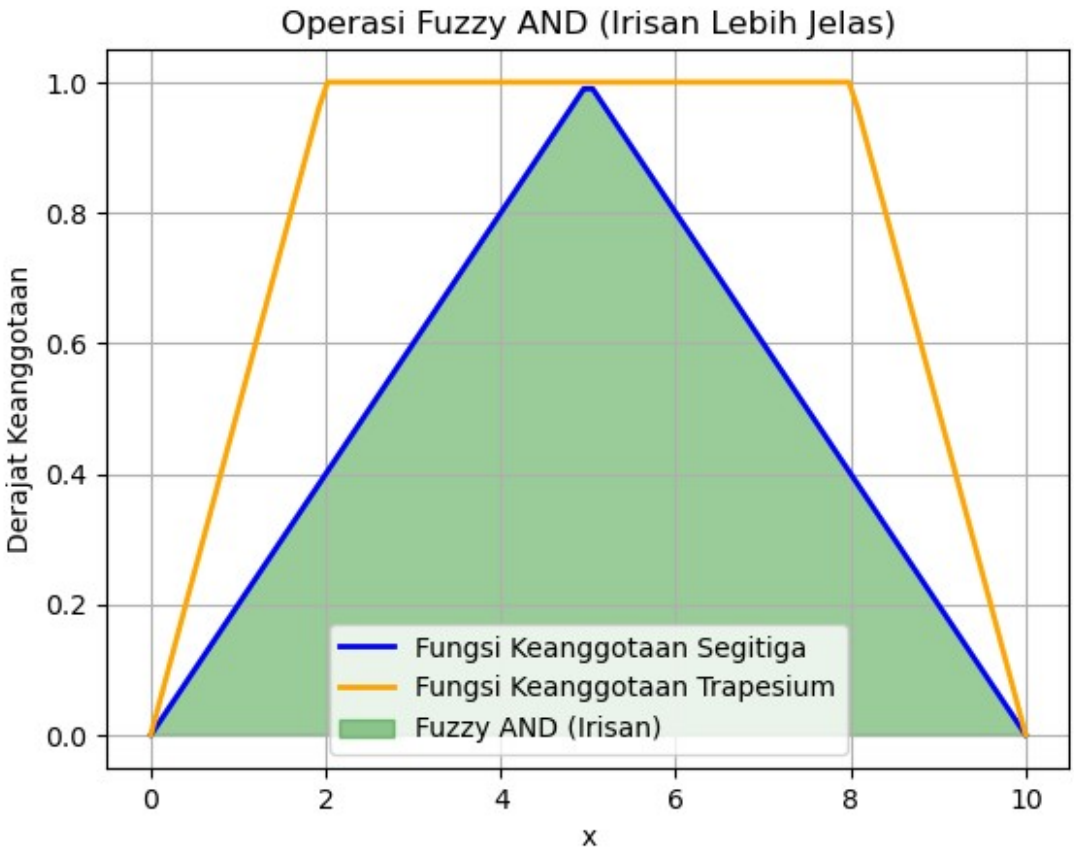
```
# Plot Fuzzy NOT
plt.figure()
plt.plot(x, trimf1, label="Fungsi Keanggotaan Segitiga (Asli)")
plt.plot(x, fuzzy_not, label="Fuzzy NOT (Komplemen)", linestyle='--')
plt.title("Operasi Fuzzy NOT")
plt.xlabel("x")
plt.ylabel("Derajat Keanggotaan")
plt.legend()
plt.grid()

plt.show()
```

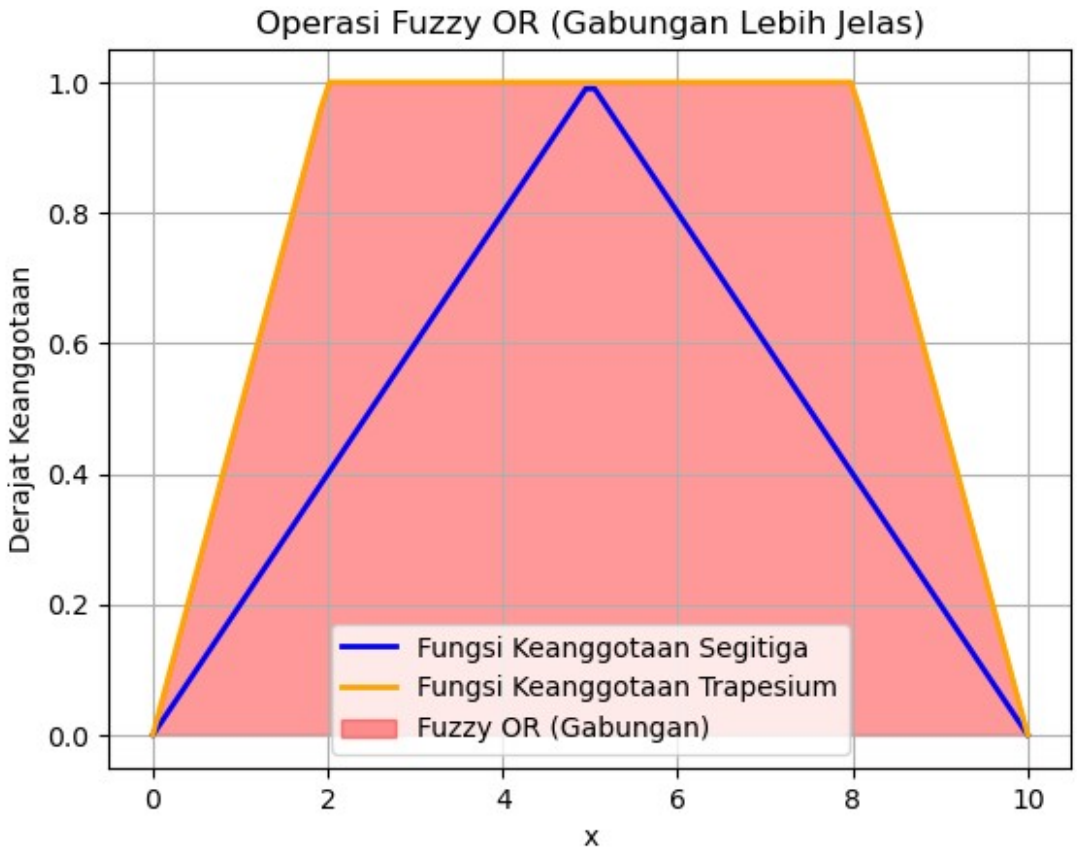


Deskripsi: Operasi **Fuzzy NOT** menghasilkan inversi dari fungsi keanggotaan segitiga. Jika suatu nilai awalnya memiliki derajat keanggotaan tinggi, maka setelah **NOT**, nilai tersebut menjadi rendah, dan sebaliknya.

```
# Mengimpor ulang library yang dibutuhkan
import numpy as np
import matplotlib.pyplot as plt

# Definisi rentang untuk sumbu x
x = np.linspace(0, 10, 100)
```

```
# Definisikan dua fungsi keanggotaan untuk demonstrasi (segitiga dan
trapesium)
trimf1 = np.maximum(0, np.minimum((x - 0) / (5 - 0), (10 - x) / (10 -
5)))  # Fungsi keanggotaan segitiga
trapmf1 = np.maximum(0, np.minimum(np.minimum((x - 0) / (2 - 0), 1),
(10 - x) / (10 - 8)))  # Fungsi keanggotaan trapesium

# Fuzzy AND: Irisan antara fungsi keanggotaan segitiga dan trapesium
fuzzy_and = np.minimum(trimf1, trapmf1)  # Menghitung nilai minimum di
setiap titik

# Fuzzy OR: Gabungan antara fungsi keanggotaan segitiga dan trapesium
fuzzy_or = np.maximum(trimf1, trapmf1)  # Menghitung nilai maksimum di
setiap titik

# Plot Fuzzy AND dengan irisan yang lebih jelas
plt.figure()
plt.plot(x, trimf1, label="Fungsi Keanggotaan Segitiga", color='blue',
linewidth=2)
plt.plot(x, trapmf1, label="Fungsi Keanggotaan Trapesium",
color='orange', linewidth=2)
plt.fill_between(x, fuzzy_and, color='green', alpha=0.4, label="Fuzzy
AND (Irisan)")
plt.title("Operasi Fuzzy AND (Irisan Lebih Jelas)")
plt.xlabel("x")
plt.ylabel("Derajat Keanggotaan")
plt.legend()
plt.grid()

# Plot Fuzzy OR dengan gabungan yang lebih jelas
plt.figure()
plt.plot(x, trimf1, label="Fungsi Keanggotaan Segitiga", color='blue',
linewidth=2)
plt.plot(x, trapmf1, label="Fungsi Keanggotaan Trapesium",
color='orange', linewidth=2)
plt.fill_between(x, fuzzy_or, color='red', alpha=0.4, label="Fuzzy OR
(Gabungan)")
plt.title("Operasi Fuzzy OR (Gabungan Lebih Jelas)")
plt.xlabel("x")
plt.ylabel("Derajat Keanggotaan")
plt.legend()
plt.grid()

# Tampilkan semua plot
plt.show()
```


Operasi Fuzzy AND (Irisan Lebih Jelas)
Derajat Keanggotaan
x
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
Fungsi Keanggotaan Segitiga
Fungsi Keanggotaan Trapesium
Fuzzy AND (Irisan)



Ringkasan Perbedaan Visualisasi Fuzzy AND dan Fuzzy OR:

1. Fuzzy AND (Irisan)

- Definisi: Mengambil nilai minimum di setiap titik antara dua fungsi keanggotaan, yaitu segitiga dan trapesium.

- Representasi: Hasil irisan ditampilkan sebagai area hijau pada plot, yang menunjukkan bagian di mana kedua kondisi memiliki derajat keanggotaan yang saling tumpang tindih.

- Makna: Irisan ini merepresentasikan kondisi di mana kedua syarat (segitiga dan trapesium) perlu terpenuhi, dengan derajat keanggotaan yang paling rendah di antara keduanya.

1. Fuzzy OR (Gabungan)

- Definisi: Mengambil nilai maksimum di setiap titik antara dua fungsi keanggotaan, yaitu segitiga dan trapesium.

- Representasi: Hasil gabungan ditampilkan sebagai area merah pada plot, yang menunjukkan bagian dengan derajat keanggotaan tertinggi di antara kedua fungsi.

- Makna: Gabungan ini merepresentasikan kondisi di mana salah satu dari kedua syarat harus terpenuhi, dengan mempertahankan derajat keanggotaan tertinggi di setiap titik, memberikan cakupan penuh dari kedua set fuzzy.

Defuzzification

Defuzzification adalah proses mengubah hasil fuzzy menjadi nilai pasti (crisp) yang dapat digunakan dalam aplikasi nyata. Dalam sistem logika fuzzy, input fuzzy diproses untuk menghasilkan output fuzzy dengan derajat keanggotaan tertentu. Namun, hasil ini perlu diubah menjadi satu nilai yang jelas agar dapat digunakan dalam pengambilan keputusan atau kontrol.

Tujuan Defuzzification:

- Mengonversi data fuzzy menjadi nilai konkret.
- Menghasilkan output yang dapat digunakan dalam sistem kontrol atau pengambilan keputusan.

Metode Defuzzification

Beberapa metode defuzzifikasi yang umum digunakan adalah:

`Centroid (Center of Gravity):`

- Metode ini menghitung titik di mana seluruh area fuzzy set seimbang, mirip dengan pusat massa dari objek fisik.
- Ini memberikan nilai yang dianggap paling representatif dari seluruh fuzzy set.

`Bisector:`

- Membagi fuzzy set menjadi dua bagian yang sama besar, yaitu dengan luas yang sama di kedua sisi.
- Titik hasil defuzzifikasi dengan metode ini adalah titik yang membagi fuzzy set secara merata. `Mean of Maximum (MOM):`
- Mengambil rata-rata dari semua nilai x yang memiliki nilai keanggotaan maksimum.
- Memberikan hasil yang berada di tengah puncak derajat keanggotaan.

`Smallest of Maximum (SOM):`

- Memilih nilai x terkecil dari titik-titik yang memiliki nilai keanggotaan maksimum.

-Menghasilkan nilai yang cenderung lebih konservatif, karena memilih puncak paling kiri.

`Largest of Maximum (LOM):`

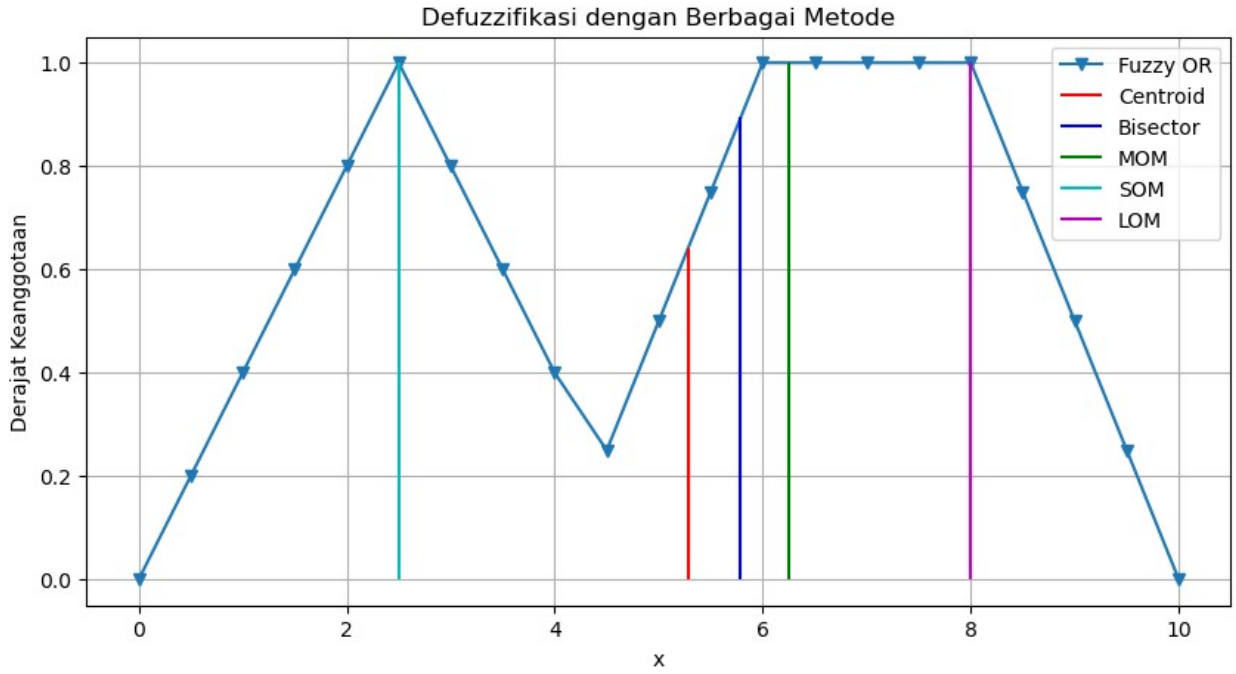
- Memilih nilai x terbesar dari titik-titik dengan nilai keanggotaan maksimum.

-Menghasilkan nilai yang lebih optimis, karena memilih puncak paling kanan.

```
# Mengimpor ulang library yang dibutuhkan
import numpy as np
import skfuzzy as fuzz
from matplotlib import pyplot as plt
```

```
# Definisi rentang nilai x
delta = 0.001
start = 0
stop = 10 + delta
step = 0.5
x = np.arange(start, stop + delta, step)

# Definisi Fungsi Keanggotaan
# Triangular Membership Function
x1 = np.arange(0, 5 + delta, step)
trimf = fuzz.trimf(x1, [0, 2.5, 5])

# Trapezoidal Membership Function
x2 = np.arange(4, 10 + delta, step)
trapmf = fuzz.trapmf(x2, [4, 6, 8, 10])

# Fuzzy OR
x3, tri_trap_or = fuzz.fuzzy_or(x1, trimf, x2, trapmf)

# Defuzzifikasi
# Centroid
centroid_x = fuzz.defuzz(x3, tri_trap_or, "centroid")
centroid_y = fuzz.interp_membership(x3, tri_trap_or, centroid_x)

# Bisector
bisector_x = fuzz.defuzz(x3, tri_trap_or, "bisector")
bisector_y = fuzz.interp_membership(x3, tri_trap_or, bisector_x)

# Mean of Maximum (MOM)
mom_x = fuzz.defuzz(x3, tri_trap_or, "mom")
mom_y = fuzz.interp_membership(x3, tri_trap_or, mom_x)

# Smallest of Maximum (SOM)
som_x = fuzz.defuzz(x3, tri_trap_or, "som")
som_y = fuzz.interp_membership(x3, tri_trap_or, som_x)

# Largest of Maximum (LOM)
lom_x = fuzz.defuzz(x3, tri_trap_or, "lom")
lom_y = fuzz.interp_membership(x3, tri_trap_or, lom_x)

# Visualisasi Hasil Defuzzifikasi Saja
plt.figure(figsize=(10, 5))
plt.title("Defuzzifikasi dengan Berbagai Metode")
plt.plot(x3, tri_trap_or, label="Fuzzy OR", marker="v")
plt.vlines(centroid_x, 0.0, centroid_y, label="Centroid", color="r")
plt.vlines(bisector_x, 0.0, bisector_y, label="Bisector", color="b")
plt.vlines(mom_x, 0.0, mom_y, label="MOM", color="g")
plt.vlines(som_x, 0.0, som_y, label="SOM", color="c")
plt.vlines(lom_x, 0.0, lom_y, label="LOM", color="m")
```

```
plt.xlabel("x")
plt.ylabel("Derajat Keanggotaan")
plt.legend(loc="upper right")
plt.grid()
plt.show()
```



Ringkasan Perbedaan Metode Defuzzifikasi:

Centroid memberikan hasil rata-rata berdasarkan "pusat massa".

Bisector membagi fuzzy set menjadi dua bagian dengan luas yang sama.

MOM mengukur rata-rata dari puncak-puncak dengan nilai maksimum.

SOM memilih puncak paling kiri, memberikan hasil yang lebih konservatif.

LOM memilih puncak paling kanan, memberikan hasil yang lebih optimis.

Contoh Problem

The Tipping Problem 'tipping problem'biasanya digunakan untuk menggambarkan kekuatan prinsip logika fuzzy untuk menghasilkan perilaku kompleks dari seperangkat aturan ahli yang ringkas dan intuitif.

Variabel Input Ada beberapa variabel yang mempengaruhi keputusan tentang seberapa besar tip yang akan diberikan saat makan di restoran. Pertimbangkan dua hal berikut ini:

- Kualitas Makanan (Quality): Seberapa enak makanan yang disajikan.

- Pelayanan (Service): Seberapa baik kualitas pelayanan yang diberikan oleh staf.

Variabel Output Variabel outputnya adalah jumlah tip yang akan diberikan, dalam bentuk persentase dari total tagihan:

- Tip: Persentase dari tagihan yang ditambahkan sebagai tip. Untuk diskusi ini, kita akan menggunakan fungsi keanggotaan **"tinggi"**, **"sedang"**, dan **"rendah"** untuk kedua variabel input (`Kualitas Makanan` dan `Pelayanan`) serta untuk variabel output (`Tip`). Fungsi-fungsi ini akan kita definisikan menggunakan pustaka scikit-fuzzy seperti berikut.

Source Code

- Membuat variabel

```
x_qual = np.arange(0, 11, 1)
x_serv = np.arange(0, 11, 1)
x_tip  = np.arange(0, 26, 1)
```

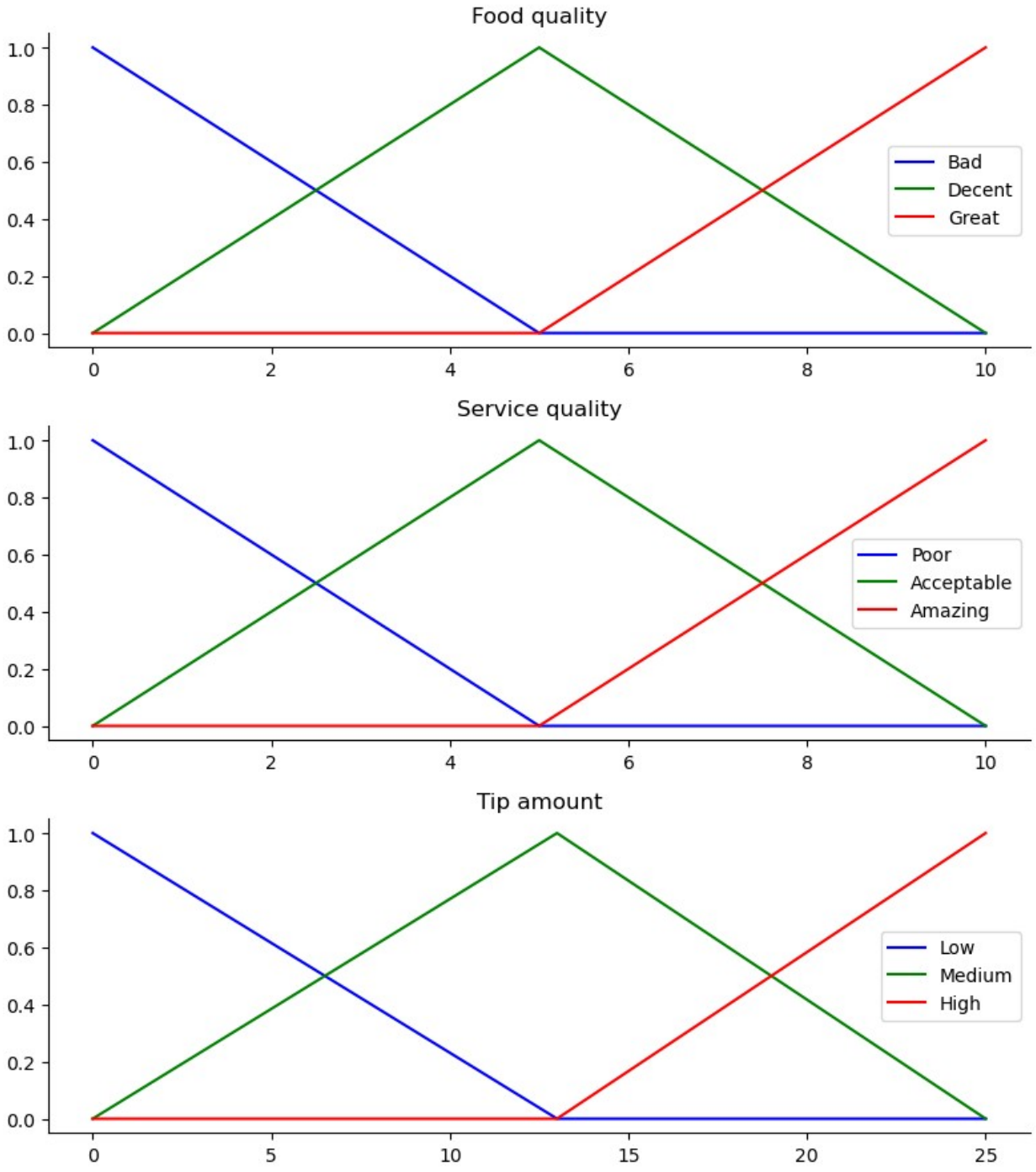
- Membuat fuzzy membership functions

```
qual_lo = fuzz.trimf(x_qual, [0, 0, 5])
qual_md = fuzz.trimf(x_qual, [0, 5, 10])
qual_hi = fuzz.trimf(x_qual, [5, 10, 10])
serv_lo = fuzz.trimf(x_serv, [0, 0, 5])
serv_md = fuzz.trimf(x_serv, [0, 5, 10])
serv_hi = fuzz.trimf(x_serv, [5, 10, 10])
tip_lo = fuzz.trimf(x_tip, [0, 0, 13])
tip_md = fuzz.trimf(x_tip, [0, 13, 25])
tip_hi = fuzz.trimf(x_tip, [13, 25, 25])
```

- Visualisasi Membership Function

```
fig, (ax0, ax1, ax2) = plt.subplots(nrows=3, figsize=(8, 9))

ax0.plot(x_qual, qual_lo, 'b', linewidth=1.5, label='Bad')
ax0.plot(x_qual, qual_md, 'g', linewidth=1.5, label='Decent')
ax0.plot(x_qual, qual_hi, 'r', linewidth=1.5, label='Great')
ax0.set_title('Food quality')
ax0.legend()

ax1.plot(x_serv, serv_lo, 'b', linewidth=1.5, label='Poor')
ax1.plot(x_serv, serv_md, 'g', linewidth=1.5, label='Acceptable')
ax1.plot(x_serv, serv_hi, 'r', linewidth=1.5, label='Amazing')
ax1.set_title('Service quality')
ax1.legend()

ax2.plot(x_tip, tip_lo, 'b', linewidth=1.5, label='Low')
ax2.plot(x_tip, tip_md, 'g', linewidth=1.5, label='Medium')
ax2.plot(x_tip, tip_hi, 'r', linewidth=1.5, label='High')
```

```
ax2.set_title('Tip amount')
ax2.legend()

# Turn off top/right axes
for ax in (ax0, ax1, ax2):
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

plt.tight_layout()
```



Aturan Fuzzy

Untuk membuat fungsi keanggotaan ini berguna, kita perlu mendefinisikan hubungan fuzzy antara variabel input dan output. Untuk contoh ini, kita gunakan tiga aturan sederhana berikut:

- Jika kualitas makanan buruk atau pelayanan jelek, maka tip akan rendah.
- Jika pelayanannya cukup baik, maka tip akan sedang.

- Jika kualitas makanan sangat bagus atau pelayanan luar biasa, maka tip akan tinggi.

Kebanyakan orang akan setuju dengan aturan-aturan ini, tetapi aturan tersebut sebenarnya fuzzy (tidak pasti). Mengubah aturan yang tidak pasti ini menjadi tip yang jelas dan dapat diambil tindakan adalah tantangan tersendiri. Di sinilah fuzzy logic sangat unggul.

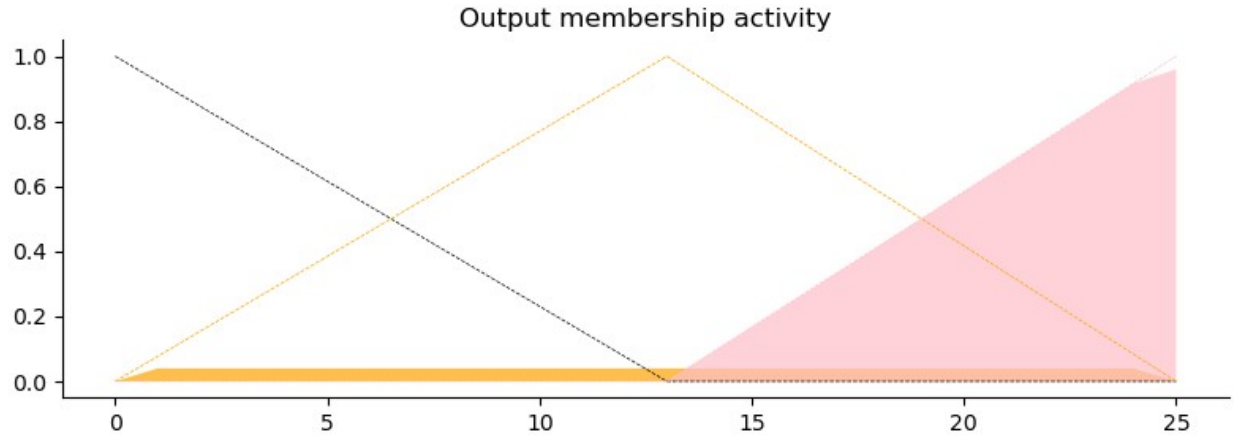
Penerapan Aturan

Berapa tip yang akan diberikan dalam situasi berikut:

- Kualitas makanan: 6,5
- Pelayanan: 9,8

```
# We need the activation of our fuzzy membership functions at these
values.
# The exact values 6.5 and 9.8 do not exist on our universes...
# This is what fuzz.interp_membership exists for!
qual_level_lo = fuzz.interp_membership(x_qual, qual_lo, 6.5)
qual_level_md = fuzz.interp_membership(x_qual, qual_md, 6.5)
qual_level_hi = fuzz.interp_membership(x_qual, qual_hi, 6.5)

serv_level_lo = fuzz.interp_membership(x_serv, serv_lo, 9.8)
serv_level_md = fuzz.interp_membership(x_serv, serv_md, 9.8)
serv_level_hi = fuzz.interp_membership(x_serv, serv_hi, 9.8)

# Now we take our rules and apply them. Rule 1 concerns bad food OR
service.
# The OR operator means we take the maximum of these two.
active_rule1 = np.fmax(qual_level_lo, serv_level_lo)

# Now we apply this by clipping the top off the corresponding output
# membership function with `np.fmin`
tip_activation_lo = np.fmin(active_rule1, tip_lo)  # removed entirely
to 0

# For rule 2 we connect acceptable service to medium tipping
tip_activation_md = np.fmin(serv_level_md, tip_md)

# For rule 3 we connect high service OR high food with high tipping
active_rule3 = np.fmax(qual_level_hi, serv_level_hi)
tip_activation_hi = np.fmin(active_rule3, tip_hi)
tip0 = np.zeros_like(x_tip)

# Visualize this
fig, ax0 = plt.subplots(figsize=(8, 3))

ax0.fill_between(x_tip, tip0, tip_activation_lo, facecolor='black',
alpha=0.7)
ax0.plot(x_tip, tip_lo, 'black', linewidth=0.5, linestyle='--', )
```

```
ax0.fill_between(x_tip, tip0, tip_activation_md, facecolor='orange',
alpha=0.7)
ax0.plot(x_tip, tip_md, 'orange', linewidth=0.5, linestyle='--')
ax0.fill_between(x_tip, tip0, tip_activation_hi, facecolor='pink',
alpha=0.7)
ax0.plot(x_tip, tip_hi, 'pink', linewidth=0.5, linestyle='--')
ax0.set_title('Output membership activity')

# Turn off top/right axes
for ax in (ax0,):
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

plt.tight_layout()
```



Aturan Agregasi

Dengan diketahuinya aktivitas setiap fungsi keanggotaan keluaran, maka seluruh fungsi keanggotaan keluaran harus digabungkan. Ini biasanya dilakukan dengan menggunakan operator maksimum. Langkah ini juga dikenal sebagai agregasi.

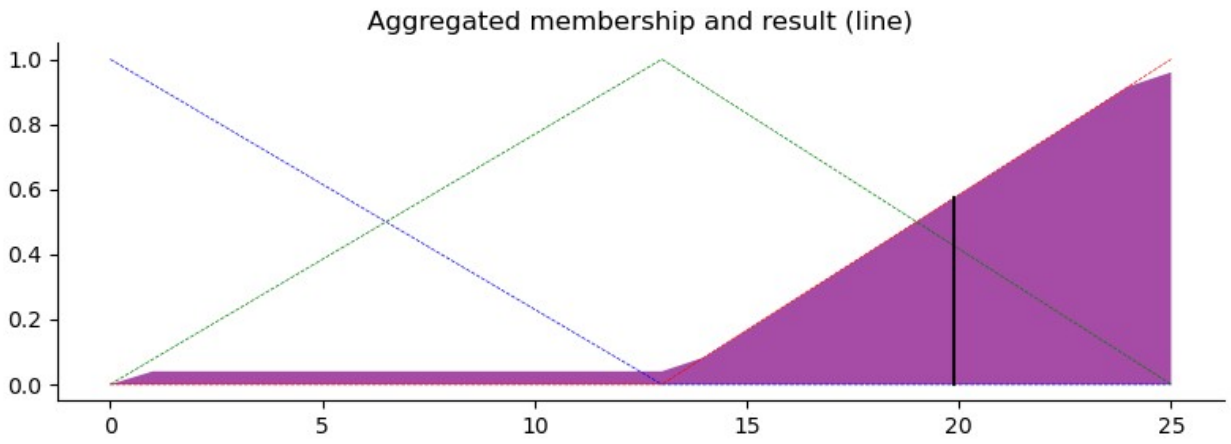
Defuzzification

Terakhir, untuk mendapatkan jawaban dunia nyata, kita kembali ke logika tajam dari dunia fungsi keanggotaan fuzzy. Untuk keperluan contoh ini metode centroid akan digunakan.

Hasilnya adalah tip sebesar 20,2%

```
# Aggregate all three output membership functions together
aggregated = np.fmax(tip_activation_lo,
                     np.fmax(tip_activation_md, tip_activation_hi))

# Calculate defuzzified result
```

```
tip = fuzz.defuzz(x_tip, aggregated, 'centroid')
tip_activation = fuzz.interp_membership(x_tip, aggregated, tip)  # for
plot

# Visualize this
fig, ax0 = plt.subplots(figsize=(8, 3))

ax0.plot(x_tip, tip_lo, 'b', linewidth=0.5, linestyle='--', )
ax0.plot(x_tip, tip_md, 'g', linewidth=0.5, linestyle='--')
ax0.plot(x_tip, tip_hi, 'r', linewidth=0.5, linestyle='--')
ax0.fill_between(x_tip, tip0, aggregated, facecolor='purple',
alpha=0.7)
ax0.plot([tip, tip], [0, tip_activation], 'k', linewidth=1.5,
alpha=0.9)
ax0.set_title('Aggregated membership and result (line)')

# Turn off top/right axes
for ax in (ax0,):
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

plt.tight_layout()
```



Kesimpulan

Kekuatan dari sistem fuzzy adalah kemampuannya untuk menghasilkan perilaku yang rumit dan intuitif hanya dengan menggunakan sistem aturan yang sederhana dan dengan beban kerja yang minimal. Perhatikan bahwa himpunan semesta dari fungsi keanggotaan kita hanya didefinisikan pada bilangan bulat, namun `fuzz.interp_membership` memungkinkan resolusi yang lebih tinggi sesuai kebutuhan. Sistem ini dapat merespons perubahan input sekecil apa pun, dan beban pemrosesannya tetap minimal.