ĆWICZENIE 1

SYNTEZA UKŁADU WNIOSKUJĄCEGO

Cel ćwiczenia

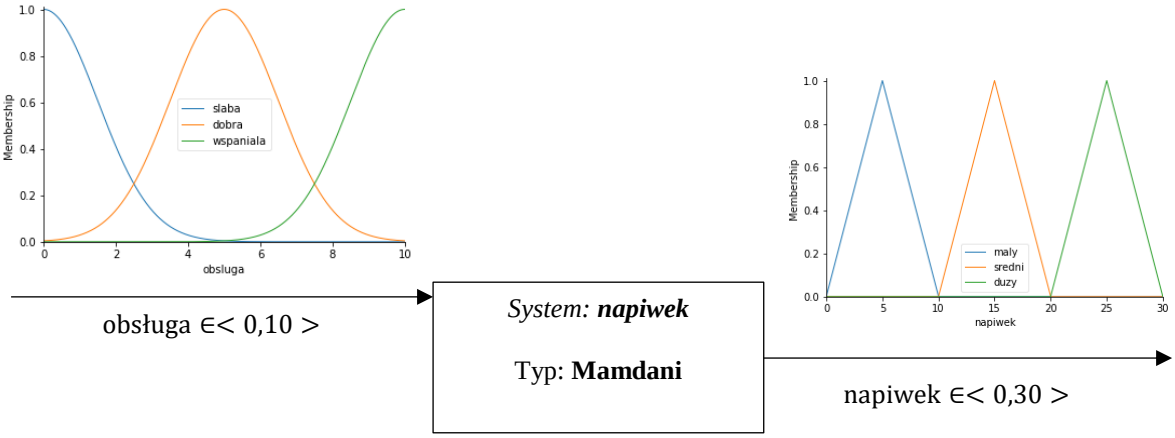
Laboratorium składa się z trzech zasadniczych części. Część I ma na celu zapoznanie się ze sposobem syntezy rozmytego systemu ekspertowego typu Mamdaniego z wykorzystaniem biblioteki **scikit-fuzzy**. W części II, należy zapoznać się z ideą działania systemu Mamdaniego a następnie dokonać modyfikacji systemu wykonanego w części I. Część III laboratorium polega na wykonaniu przykładowego zadania zaliczeniowego.

Część I: Wykonanie przykładowego systemu ekspertowego typu Mamdaniego.

Problem

Zaprojektować rozmyty układ ekspertowy doradzający ile napiwku pozostawić w restauracji na podstawie oceny jakości obsługi oraz jakości jedzenia. Jakość obsługi i jakość jedzenia będzie oceniana w skali od 1 do 10, gdzie 10 reprezentuję ocenę maksymalną, natomiast napiwek będzie liczbą z przedziału [0,30] reprezentującą procent wartości rachunku.
Baza reguł będzie składała się z 5 reguł. System zostanie wykonany w dwóch etapach. W **etapie pierwszym** (Rys. 1) system będzie zbudowany z jednego wejścia (*obsługi*) i jednego wyjścia (*napiwek*) oraz 3 reguł postaci:

R1. jeżeli obsługa jest słaba, to napiwek jest mały
R2. jeżeli obsługa jest dobra, to napiwek jest średni
R3. jeżeli obsługa jest wspaniała, to napiwek jest duży

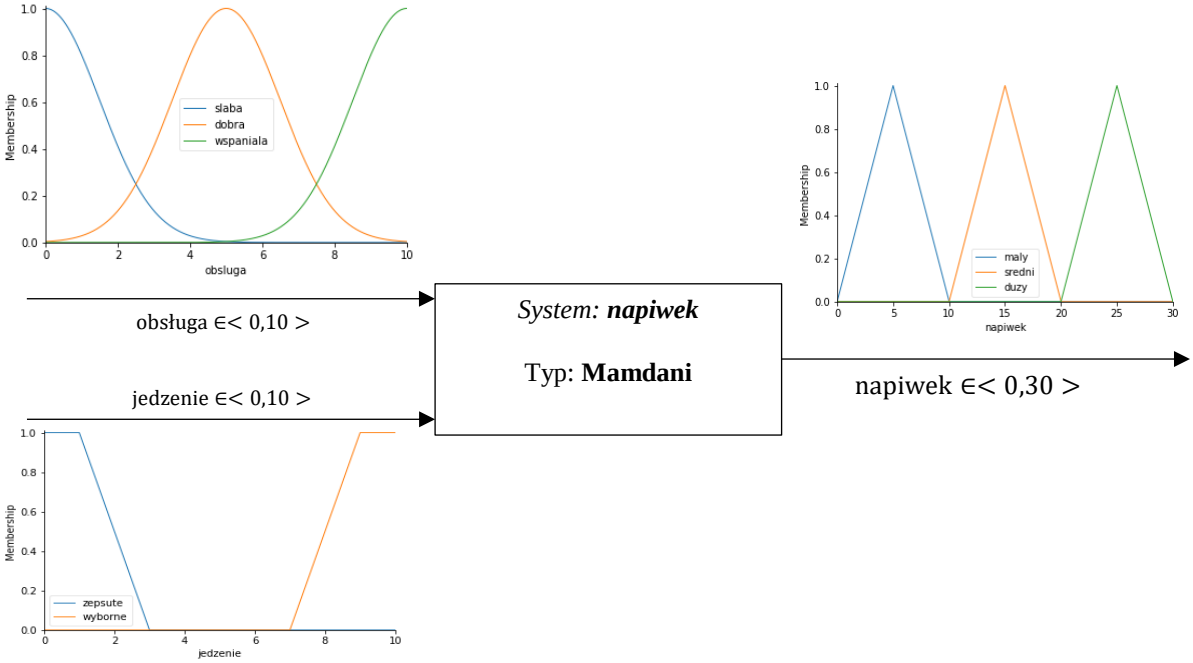


Rys. 1. System napiwek o 1 wejściu

Nad strzałkami (patrz Rys. 1) reprezentującymi zmienne wejściowe i wyjściowe zamieszczono kształty zbiorów rozmytych, którymi należy pokryć dziedziny poszczególnych zmiennych.

W **etapie drugim** do systemu zostanie dodane druga zmienna wejściowa *jedzenie* (Rys. 2) oraz 2 dodatkowe reguły:

R4. jeżeli jedzenie jest zepsute, to napiwek jest mały
R5. jeżeli jedzenie jest wyborne, to napiwek jest duży



Rys. 2. System *napiwek* o 2 wejściach

Dziedziny zmiennych stanu zostały pokryte trzema typami zbiorów rozmytych (Rys. 2):

- *gaussa* – parametryzowanymi za pomocą dwóch liczb: centrum i rozpiętość funkcji gaussa. Przykładowo, parametrami zbioru *słaba* są: 0 (centrum zbioru) i 1.5 (rozpiętość),
- *trójkątnym* – opisywany za pomocą tzw. trzech punktów załamania, a więc tymi punktami na osi argumentu funkcji, w których ma miejsce zmiana przebiegu funkcji zbioru. Przykładowo, parametrami zbioru *mały* jest trójka liczb 0, 5 oraz 10,
- *trapezoidalnym* - opisywany za pomocą czerech punktów załamania, np. parametrami zbioru *zepsute* jest czwórka liczb 0, 0, 1 oraz 3.

Przebieg ćwiczenia

1. Inicjowanie modułów
```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
import matplotlib.pyplot as plt
```

2. Utworzenie zmiennych stanu poprzednika „jedzenie” oraz następnika „napiwek”
```
obsluga  = ctrl.Antecedent(np.arange(0,10.01,0.01), 'obsluga')
napiwek  = ctrl.Consequent(np.arange(0,30.01,0.01), 'napiwek')
```

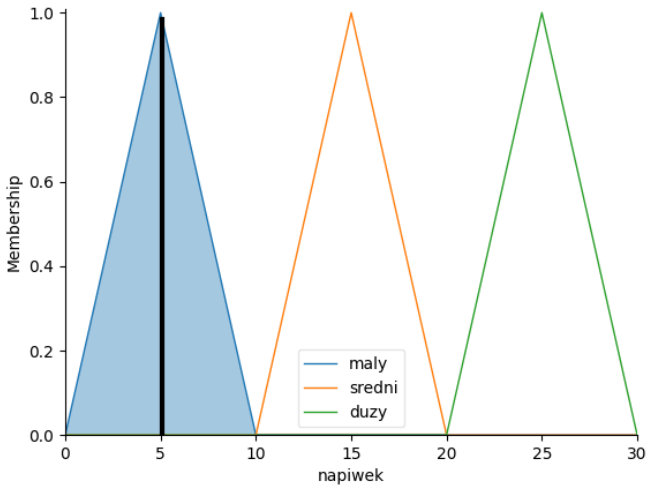
3. Dodanie zbiorów rozmytych
```
obsluga['slaba'] = fuzz.gaussmf(obsluga.universe, 0, 1.5)
```
Powyżej dodano zbiór „słaba” o centrum umieszczonym w punkcie uniwersum równym **0** i rozpiętości wynoszącej **1.5**. Na podobnej zasadzie należy dodać zmiennej „obsluga” zbiory „dobra” i „wspaniała” o centrach ulokowanych w punktach odpowiednio **5** oraz **10** i rozpiętości wynoszącej **1.5**, zaś zmiennej „napiwek” 3 zbiory trójkątne (`trimf`) o nazwach „maly”, „sredni” i „duzy” i parametrach `[0, 5, 10]`, `[10, 15, 20]` i `[20, 25, 30]`.

4. Podgląd zbiorów rozmytych można zrealizować metodą view() dla poszczególnych zmiennych stanu, np.:
```
obsluga.view()
```

5. Definicja reguł, np. dla reguły pierwszej ma postać:
```
regula1 = ctrl.Rule(obsluga['slaba'], napiwek['maly'])
```
Podobnie zależy zdefiniować regułę 2 i 3 z pierwszej strony instrukcji.

6. Dodanie reguł do systemu rozmytego:
```
napiwek_ctr = ctrl.ControlSystem([regula1,regula2,regula3])
napiwek_sym = ctrl.ControlSystemSimulation(napiwek_ctr)
```

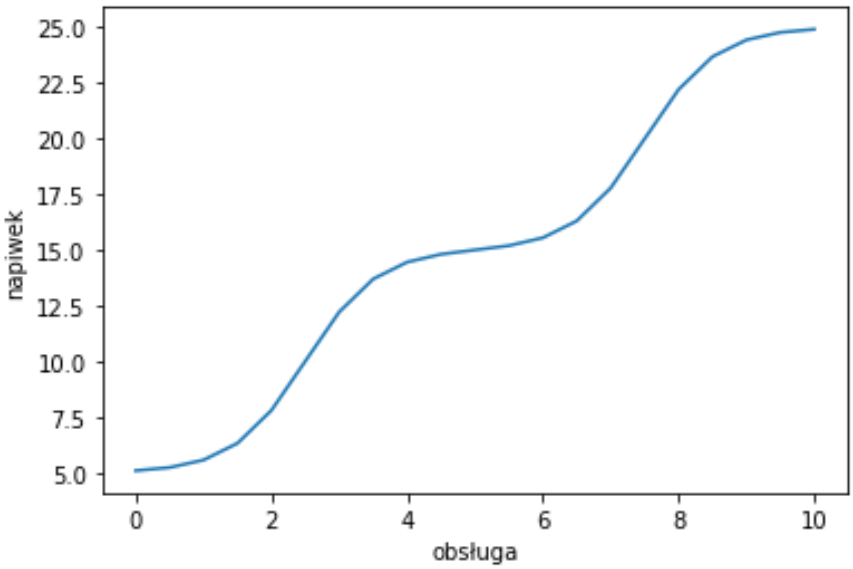
7. Sprawdzenie działania systemu dla wartości obsługi równej **0**
```
napiwek_sym.input['obsluga']     = 0
napiwek_sym.compute()
print('Wynik',napiwek_sym.output['napiwek'])
napiwek.view(sim=napiwek_sym)
```
Wartość napiwku powinna wynosić **5.076578**.



Rys. 3. Wyostrzanie metodą środka ciężkości dla wejścia obsluga = 0.

Proszę sprawdzić również wartość napiwku dla wartości wejścia równej **10**.

8. Sprawdzenie działania systemu dla wartości obsługi od 0 do 10 (Rys. 4 - wykres funkcji napiwek=f(obsluga))
```
n_points = 21
x = np.linspace(0, 10, n_points)
z = np.zeros_like(x)
for i in range(n_points):
    napiwek_sym.input['obsluga'] = x[i]
    napiwek_sym.compute()
    z[i] = napiwek_sym.output['napiwek']

fig, ax = plt.subplots()
ax.set_xlabel('obsługa')
ax.set_ylabel('napiwek')
ax.plot(x,z)
fig.show()
```



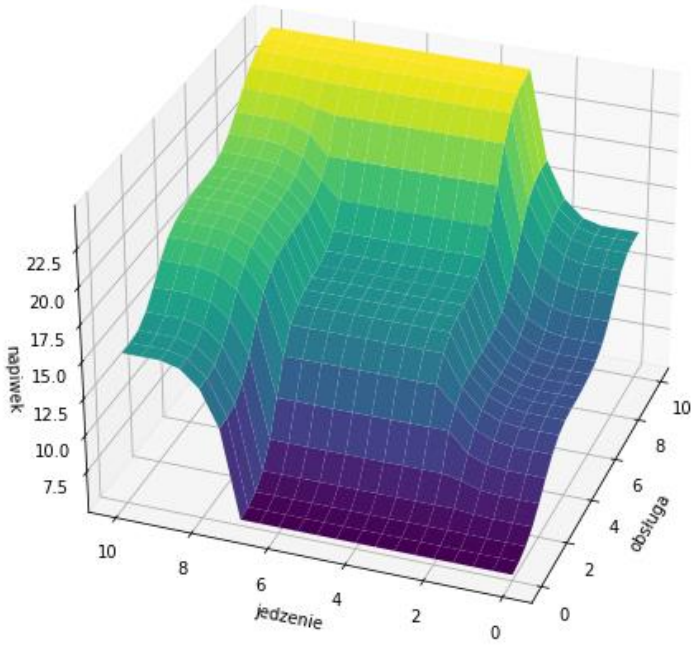
Rys. 4. Powierzchnia przejścia systemu „napiwek" o jednym wejściu.

9. Dodanie drugiej wejściowej zmiennej stanu „jedzenie" o trapezoidalnych (`trapmf`) zbiorach rozmytych „zepsute" oraz „wyborne". Patrz punkt 3. Parametry zbiorów trapezoidalnych to: `[-2, 0, 1, 3]` oraz `[7, 9, 10, 12]`.
10. Dodanie reguł 4 i 5 będzie odbywało się analogicznie do punktu 5 z trzeciej strony instrukcji.
11. Sprawdzenie działania systemu dla wartości obsługi równej **0** oraz wartości jedzenia równej **0**.

```
napiwek_sym.input['obsluga']     = 0
napiwek_sym.input['jedzenie']    = 0
napiwek_sym.compute()
print('Wynik',napiwek_sym.output['napiwek'])
napiwek.view(sim=napiwek_sym)
```

12. Sprawdzenie działania systemu dla wartości obsługi i jedzenia od 0 do 10 (Rys. 5 - wykres funkcji napiwek=f(obsluga, jedzenie))

```
n_points = 21
upsampled = np.linspace(0, 10, n_points)
x, y = np.meshgrid(upsampled, upsampled)
z = np.zeros_like(x)

# Loop through the system 21*21 times to collect the control surface
for i in range(n_points):
    for j in range(n_points):
        napiwek_sym.input['obsluga'] = x[i, j]
        napiwek_sym.input['jedzenie'] = y[i, j]
        napiwek_sym.compute()
        z[i, j] = napiwek_sym.output['napiwek']

fig = plt.figure(figsize=(8, 8))
ax = fig.add_subplot(111, projection='3d')

surf = ax.plot_surface(x, y, z, cmap='viridis')

ax.set_xlabel('obsługa')
ax.set_ylabel('jedzenie')
ax.set_zlabel('napiwek')

ax.view_init(30, 200)
```



Rys. 5. Powierzchnia przejścia systemu napiwek o dwóch wejściach.

Część II: Modyfikacja systemu

Jak można było zauważyć, wyjście systemu napiwek zawierało się w przedziale (**5.08**,**24.9**), a zatem niemożliwym było uzyskanie napiwku mniejszego od **5.08** i większego od **24.9**. Celem tej części laboratorium będzie taka modyfikacja systemu, aby uzyskać wartość napiwku z przedziału (**0**,**30**). Zostanie to osiągnięte jedynie poprzez modyfikację kształtów zbiorów rozmytych zmiennej wejściowej „obsługa" oraz wyjściowej „napiwek". Uzasadnieniem modyfikacji systemu jest zrozumienie zasady działania systemu Mamdaniego. Przyjmijmy na potrzeby niniejszego laboratorium następującą szczególną definicję systemu Mamdaniego postaci:

Def.: System rozmyty typu Mamdaniego definiowany jest przez piątkę operatorów:

1. Rozmywania: **singleton rozmyty** (najpopularniejszy)
2. Wyznaczenia stopnia aktywności reguł(y): **min**
3. Implikacji: **min**
4. Agregacji/złożenia: **max**
5. Wyostrzania: **środek ciężkości** (najpopularniejszy)

Przykład:

Rozpatrzmy następnie działanie systemu napiwek o wejściach ustalonych na **0**, **0** (Rys. 6.). W pięciu wierszach zwizualizowano kolejne reguły R1 .. R5. Reguły posiadają poprzedniki będące zdaniami prostymi, stąd czerwonym znakiem X zaznaczono, że druga (R1-R3) bądź pierwsza (R4, R5) ze zmiennych stanu w danej regule nie jest uwzględniana.

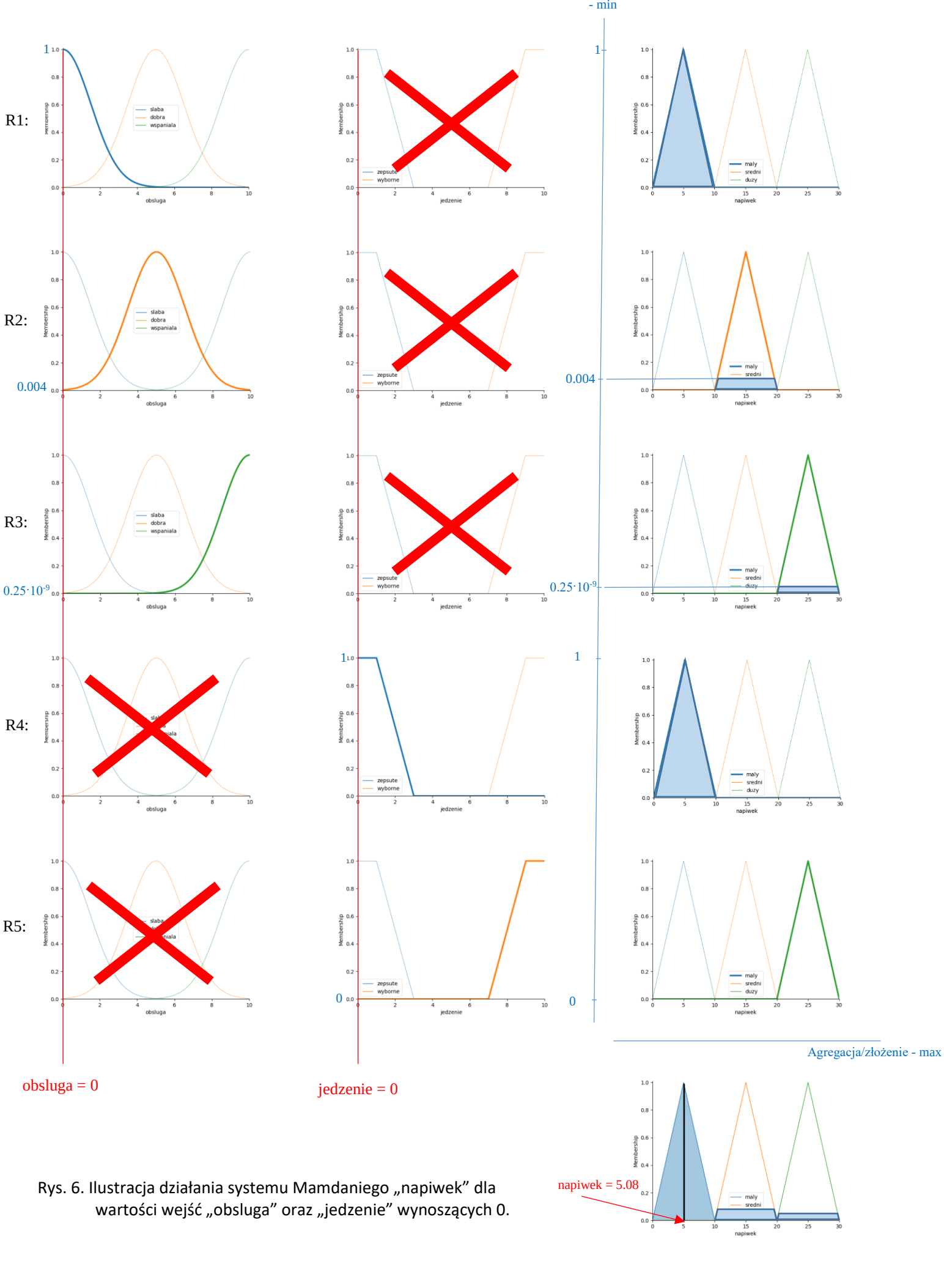
Efektem działania **operatora rozmywania singleton rozmyty** będą wartości funkcji przynależności zbiorów poprzedników poszczególnych reguł. I tak przynależność wartości wejścia obsluga = 0 do zbioru „slaba" jest maksymalna i wynosi 1 (Reguła1). W regułach 2 i 3 przynależność wartości wejścia obsluga = 0 do zbiorów „dobra" i „wspaniała" wynosi odpowiednio 0.004 oraz $0.25 \cdot 10^{-9}$. Przynależność wartości wejścia jedzenie = 0 do zbiorów „zepsute" i „wyborne" wynosi odpowiednio 1 oraz 0.

Operator **wyznaczenia stopnia aktywności reguł min** będzie jednoargumentowym (np. dla reguły R1 mamy min(1)=1), stąd na pionowym odcinku obrazującym stopnie aktywności poszczególnych reguł mamy wartości przynależności wartości chwilowych zmiennych stanu do zbiorów rozmytych poszczególnych reguł.

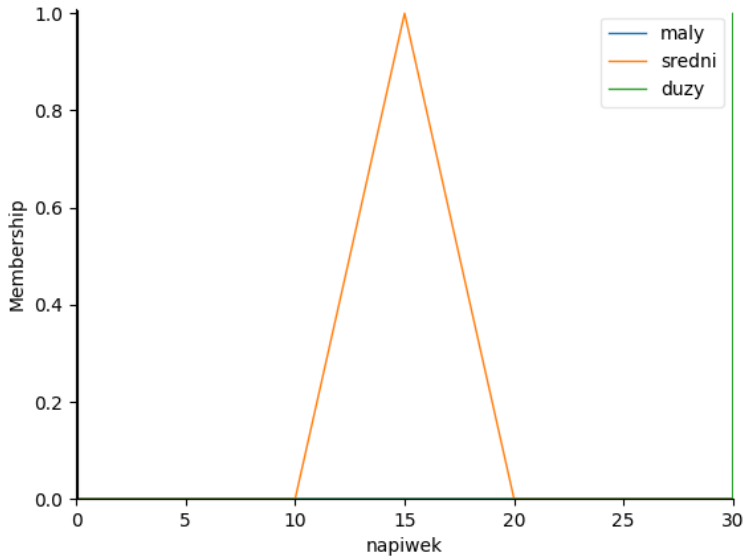
Operator **implikacji min** działa pomiędzy liczbą będącą stopniem aktywności reguły a poszczególnymi wartościami funkcji przynależności zbioru będącego następnikiem reguły. Wizualnie prowadzi to do „przycięcia" zbioru będącego następnikiem do wysokości wynikającej ze stopnia aktywności reguły. Stopnie przynależności reguły R2 i R3 wynoszące odpowiednio ok 0.004 oraz $0.25 \cdot 10^{-9}$ zostały w istotny sposób powiększone celem podkreślenia wpływu reguł R2 i R3 na wynik końcowy. W rzeczywistości stopnie aktywności tych reguł nie byłyby widoczne na powyższym rysunku.
Operator **agregacji/złożenia max** został zwizualizowany za pomocą poziomego odcinka pod następnikiem reguły R5. Efektem jego działania jest graficzna suma następników reguł R1-R5 zwizualizowanych trójkątami i trapezami w kolorze niebieskim. Figura przedstawiona w prawym dolnym rogu rysunku jest rozmytą konkluzją.
Zadaniem operatora **wyostrzania** jest zaproponowanie ostrej reprezentacji konkluzji rozmytej (zbioru rozmytego) będącego wynikiem złożeniowej reguły wnioskowania. W niniejszym przykładzie ograniczymy się jedynie do operatora środka ciężkości, który dał rezultat pod postacią liczby ok **5.08**.
To kończy ilustrację działania operatorów składających się na rozmyty system rozmyty typu Mamdaniego. Można teraz powrócić do celu tej części laboratorium, którym jest uzyskanie wyjścia równego **0**. Znając zasadę działania systemu przeprowadźmy rozumowanie odwrotne celem uzyskania oczekiwanego rozwiązania. Jeżeli środek ciężkości zbioru będącego wynikiem wnioskowania ma być ulokowany w punkcie uniwersum równym 0, to jedyny zbiór, który spełnia zadość takiemu wymaganiu (przy założeniu, że nie rozszerza się dziedziny zmiennej napiwek o wartości poniżej 0 i powyżej 30) ma kształt singletonu, tzn. zbioru, który przyjmuje wartość równą 1 w punkcie uniwersum równym 0. Aby uzyskać taki zbiór należy wykonać dwie modyfikacje systemu podstawowego:

1. Zmienić kształty zbiorów rozmytych dla zmiennej „obsluga" na takie, których nośnikiem nie jest całe uniwersum (np. zbiory trójkątne). Taka modyfikacja sprawi, że w punkcie obsluga = 0 niezerowa wartość funkcji przynależności będzie miała miejsce jedynie dla zbioru „slaba". Zerowa wartość funkcji przynależności zbiorów „dobra" i „wspaniała" przełoży się na zerową aktywność reguł R2 i R3 i w konsekwencji na brak przyczynków pod postacią trapezów do zbioru wynikowego.
2. Modyfikacja druga dotyczy zbioru „mały" zmiennej napiwek. Jak wspomniano powyżej powinien to być zbiór typu singleton ulokowany w punkcie 0, a zatem jego parametry powinny być następujące: **[**0**,** 0**,** 0**]**

Uwzględniwszy powyższe dwa punkty w kodzie programu powinniśmy otrzymać zbiór wynikowy postaci (Rys. 7) i wartość napiwku w przybliżeniu **0**.



Rys. 6. Ilustracja działania systemu Mamdaniego „napiwek” dla wartości wejść „obsluga” oraz „jedzenie” wynoszących 0.

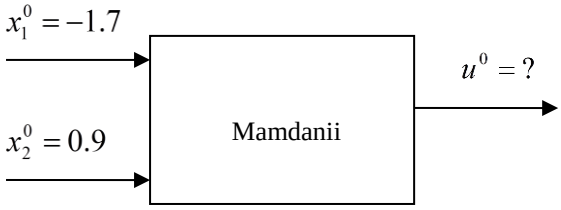


Rys. 7. Konkluzja wynikowa zmodyfikowanego systemu „napiwek”. Zbiory „maly” oraz „duzy” są singletonami. Ze względu na brak aktywności reguł innych niż R1 oraz R4 konkluzją wynikową jest jedynie sigleton ulokowany w punkcie 0, którego środek ciężkości jest również ulokowany w tym samym punkcie.

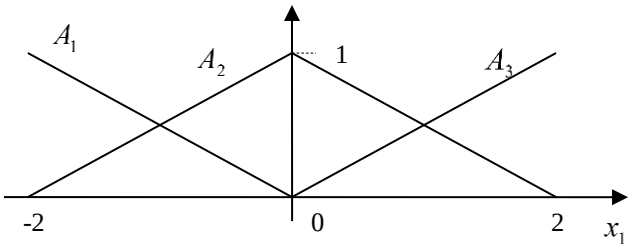
Część III: Przykładowe zadanie zaliczeniowe

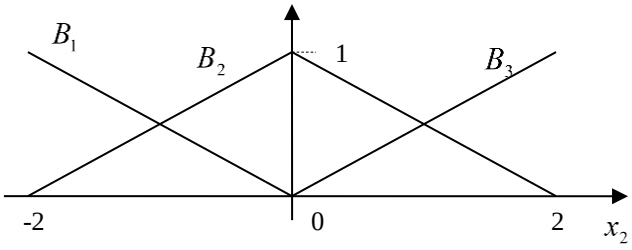
Zadanie na ocenę 3:

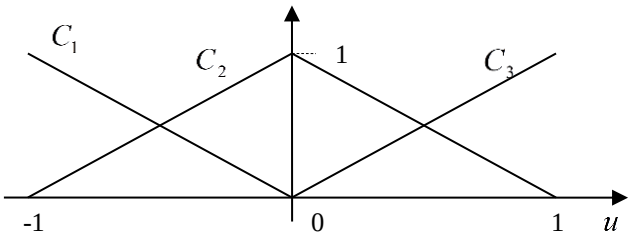
Wyznaczyć wyjście systemu wnioskowania rozmytego Mamdaniego przedstawionego na rysunku 8.



Zbiory rozmyte







Tablica reguł

$x_1 \setminus x_2$	B_1	B_2	B_3
A_1	-	C_1	C_2
A_2	-	C_2	C_3
A_3	-	-	-

Rys. 8. Treść przykładowego zadania zaliczeniowego

Odp.: u^0=-0.131

Uwaga: Odpowiedzi nie będziecie Państwo znali w trakcie zaliczenia.

Sprawozdanie

W sprawozdaniu na ocenę **dobrą** wystarczy udokumentować wykonanie wszystkich 3 części laboratorium. Na ocenę **bardzo dobrą** należy:

1. Zrealizować treści laboratorium w innym języku/bibliotece, niż zaproponowane na laboratorium. Poniższa tabela pochodząca z artykułu Simone Spolaor, Caro Fuchs, Paolo Cazzaniga, Uzay Kaymak, Daniela Besozzi, Marco S. Nobile, *Simpful: A User-Friendly Python Library for Fuzzy Logic*, International Journal of Computational Intelligence Systems, Vol. 13(1), 2020, pp. 1687–1698, DOI: https://doi.org/10.2991/ijcis.d.201012.002; ISSN: 1875-6891; eISSN: 1875-6883, https://www.atlantis-press.com/journals/ijcis/ może stanowić pomoc w wyborze alternatywnego narzędzia pozwalającego na zrealizowanie laboratorium.

Table 1 Software for the design of FISs.

Name	Language	Latest Release	Description
FuzzyLite [14]	C++	2017	A collection of C++ libraries designed for fuzzy control, compatible with the FCL standard
FisPro [15]	C++	2019	A general-purpose software provided with a GUI, designed to facilitate the learning of fuzzy inference systems from data
Juzzy [16]	Java	2013	A Java based toolkit, implementing type-2 fuzzy reasoning
JFML [17]	Java	2018	A Java library implementing the FML standard
FuzzyR [18]	R	2019	An R toolkit, provided with a GUI, for the design of type-1 and type-2 fuzzy inference systems
Fuzzy Toolbox for Matlab [19]	Matlab	1994	General-purpose toolbox implemented in the Matlab environment
Fuzzy Logic Toolbox [20]	Matlab	2020	Commercially distributed toolbox, provided with a GUI and available inside the Matlab environment
PyFuzzy [21]	Python 2	2014	A Python 2 library, compatible with the FCL standard. The development of the library was discontinued and Python 2 is no longer officially supported. This library depends on the ANTLR 3 runtime
Fuzzylab [22]	Python 3	2019	Python library based on the Octave Fuzzy Logic Toolkit
Scikit-Fuzzy [23]	Python 3	2019	General-purpose API meant to work in the scipy stack, offering classes and methods to support the definition of fuzzy systems
Py4JFML [24]	Python 3	2019	A Python wrapper for the JFML java library

Note: FIS, fuzzy inference system; FCL, Fuzzy Control Language; GUI, graphical user interface; FML, Fuzzy Markup Language; API, Application Programming Interface.

Name	Mamdani	Zero-order Takagi–Sugeno	First-order Takagi–Sugeno	Higher Order Takagi–Sugeno	Open Source
FuzzyLite [14]	✓	✓	✓	✓	✓
FisPro [15]	✓	✓			✓
Juzzy [16]	✓				✓
JFML [17]	✓	✓	✓		✓
FuzzyR [18]	✓				✓
Fuzzy Toolbox for Matlab [19]	✓	✓	✓		
Fuzzy Logic Toolbox [20]	✓	✓	✓		
PyFuzzy [21]	✓	✓			✓
Fuzzylab [22]	✓	✓			✓
Scikit-Fuzzy [23]	✓				✓
Py4JFML [24]	✓	✓	✓		✓
Simpful	✓	✓	✓	✓	✓

Note: FIS, fuzzy inference system; FML, Fuzzy Markup Language.

2. Wykazać w sposób formalny, że rozwiązanie zadania z części III laboratorium jest poprawne. W tym celu należy przeprowadzić wnioskowanie zgodnie z definicją systemu Mamdaniego przytoczoną na stronie 5 a następnie przykładowo zilustrowaną na rys. 6, zaś wyostrzanie metodą środka ciężkości przeprowadzić wg zależności:

$$u^0 = \frac{\int_u u \cdot C\prime(u)du}{\int_u C\prime(u)du},$$

w której $C'(u)$ jest zbiorem rozmytym powstałym w wyniku złożeniowej reguły wnioskowania.