LABORATORUM 1
Układy logiki rozmytej

1. Cel ćwiczenia

Poznanie narzędzia: Fuzzy Logic Toolbox z systemu Matlab. Prosty układ wnioskowania rozmytego.

2. Wprowadzenie

Sterowanie rozmyte oferuje wygodne możliwości projektowania sterowania obiektami nieliniowymi, szczególnie w przypadku, gdy charakter nieliniowości utrudnia ich opisanie metodami analitycznymi, np. w formie równań różniczkowych lub algebraicznych, i wymagana jest zmiana parametrów regulacji w zależności od punktu pracy. Tradycyjną techniką stosowaną w takich przypadkach jest tzw. programowanie wzmocnienia (gain scheduling), ale analiza działania otrzymanego regulatora jest zwykle trudna. Ze względu na możliwość implementacji algorytmu sterowanie rozmyte należy do komputerowych (mikroprocesorowych) metod regulacji. Można wyróżnić następujące cechy sterowania rozmytego:

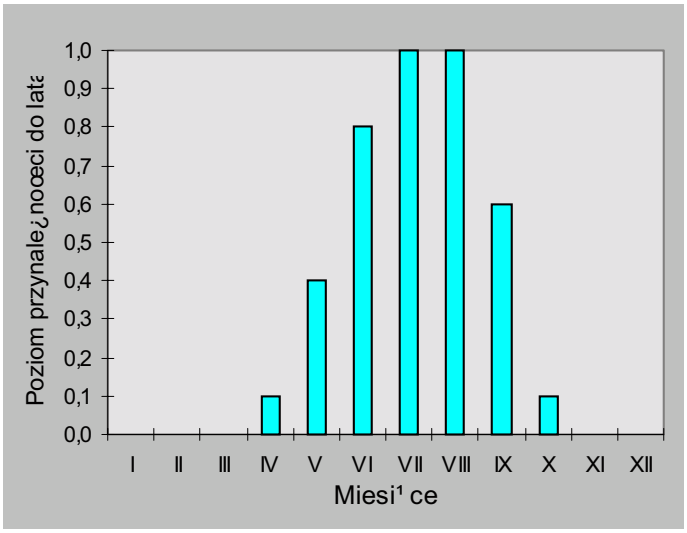
- umożliwia zapisanie problemu w języku naturalnym na podstawie doświadczenia "eksperta" (analizy zależności zbioru danych z wejścia i wyjścia procesu), co ułatwia jego zrozumienie,
- umożliwia modelowanie zależności nieliniowych o dużej złożoności, gdzie opis analityczny jest trudny lub niemożliwy,
- umożliwia zastosowanie adaptacyjnej techniki doboru parametrów na podstawie danych uczących (ANFIS - Adaptive Neuro-Fuzzy Inference Systems),
- jest elastyczne i odporne na nieprecyzyjne dane,
- nadaje się do stosowania obliczeń równoległych,
- może być łączone z konwencjonalnymi metodami sterowania.

Logika rozmyta opiera się na pojęciu zbioru rozmytego. Zbiór rozmyty różni się od klasycznego zbioru logiki dwuwartościowej tym, że nie ma ostrej, dobrze określonej granicy. W przypadku klasycznego zbioru A element x całkowicie należy do A (przynależność równa 1) albo całkowicie jest z A wyłączny (przynależność równa 0), czyli należy do zbioru nie-A (jest to tzw. zasada wyłączonego środka). W przypadku zbioru rozmytego przynależność elementu może być częściowa i przybierać dowolną wartość z przedziału [0,1]. Wartość ta jest określona przez tzw. funkcję przynależności (membership function). W przypadku pojęć nieostrych i nieprecyzyjnych logika rozmyta jest naturalnym sposobem opisu. Ilustruje to rys.1, na którym pokazane są przykłady pojedynczej dyskretnej funkcji przynależności (1a) oraz zbioru ciągłych funkcji pokrywających całą przestrzeń wartości wejściowych (1b). O konkretnym kształcie i położeniu funkcji przynależności decyduje "wiedza eksperta", którym może być doświadczony operator albo np. sieć neuronowa uczona danymi doświadczalnymi z procesu.

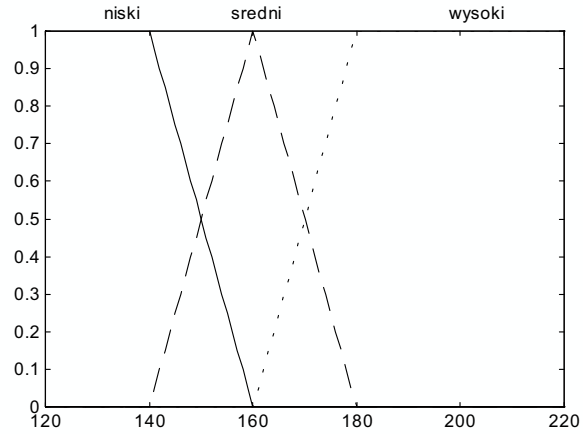
Poziomy przynależności do zbiorów rozmytych różne od 0 (*false*) lub 1 (*true*) wymagają rozszerzenia definicji *operacji logicznych*. I tak najprostszym rozszerzeniem operacji iloczynu logicznego *A* AND *B*, gdzie $A,B \in [0,1]$ są poziomami przynależności, jest zastosowanie funkcji *min*(*A,B*) wybierającej mniejszą z wartości funkcji przynależności do *A* i *B*, dla operacji sumy *A* OR *B* można zastosować funkcę *max*(*A,B*), a dla negacji NOT *A* funkcję 1-*A*. Tworzy się w ten sposób tablice prawdy logiki rozmytej. W ogólności, funkcje dla operatorów logiki rozmytej można wybierać w sposób bardzo dowolny przy zachowaniu ogólnych zasad, w szegółności zgodności z logiką klasyczną dla wartości 0 i 1. Alternatywą dla funkcji AND jest często iloczyn *prod(A,B),* a dla funkcji OR suma probabilistyczna *probor(A,B)=A+B-A*B*.

Zbiory i operatory rozmyte pełnią funkcje odpowiednio podmiotu i orzeczenia zdań logiki rozmytej. Do konstruowania algorytmów rozmytych wykorzystuje się zdania warunkowe typu *if-then*. W najprostszym przypadku ma ono formę *if x is A then y is B,* gdzie *A* i *B* są wartościami lingwstycznymi określonymi przez zbiory rozmyte na przestrzeniach *X* i *Y*, z których pochodzą elementy *x* i *y*. Zdanie po *if* nazywa się przesłanką, a zdanie po *then* - następstwem lub konkluzją.

a)



b)



Rys1. Przykłady funcji przynależności: a) miesiące należące do lata, b) podział wzrostu człowieka na 3 kategorie: niski, średni i wysoki (trapezowe funkcje przynależności)

Przykład: *if* ***temperatura*** *is* ***niska*** *then* ***zawór wody gorącej*** *is* ***otwarty***

Przesłanka zwraca *liczbę* określającą poziom przynależności konkretnej wartości wejściowej *x* do zbioru *A*, natomiast w konkluzji wartości wyjściowej *y* przyporządkowuje się *zbiór* rozmyty *B* (a właściwie funkcję przynależności do *B)*, co można wyrazić w języku C czy MATLAB przez różnicę symboli: *if x == A then y = B.* Klasyczna metoda z rozmytym zbiorem wyjściowym nosi nazwę metody Mamdani'ego. W wielu przypadkach bardziej efektywne jest zastosowanie jako wyjściowej funkcji przynależności pojedynczego piku (tzw. singletona), co ułatwia opisaną w pkt.II.5 defuzzyfikację wyjścia. Takie układy rozmyte nazywają sie układami Sugeno.

Przesłanka może składać się z wielu części połączonych operatorami, np.

if (***temperatura*** *is* ***niska***) *or* (***ciśnienie*** *is* ***niskie***) *then ...*

Wszystkie składowe przesłanki mogą być obliczane jednocześnie, a wynik liczbowy otrzymuje się po zastosowaniu operatora logicznego OR. Podobnie złożona może być konkluzja, np.

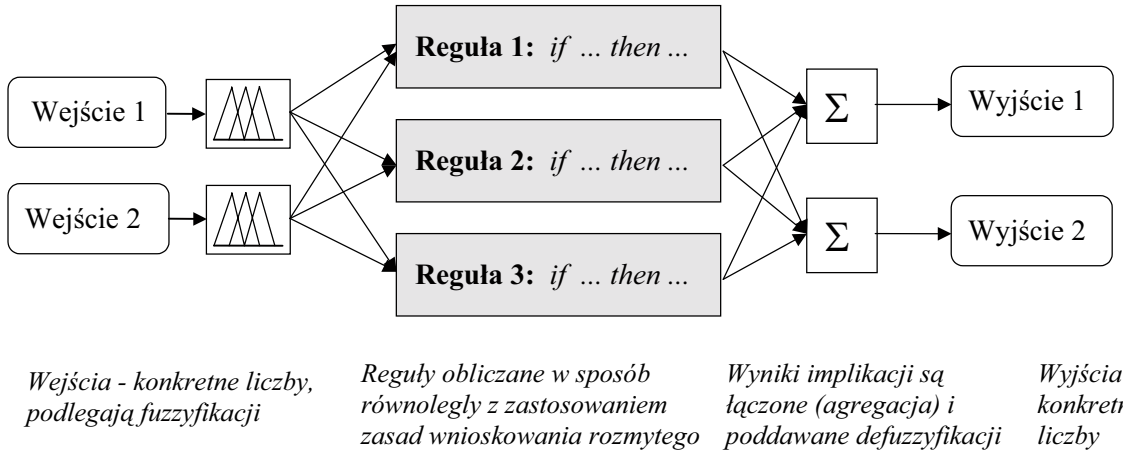
if ***temperatura*** *is* ***niska*** *then* (***zawór wody gorącej*** *is* ***otwarty***) *and* (***zawór wody zimnej is zamknięty***)

W tym przypadku wynik przesłanki ma jednakowy wpływ na wszystkie składowe konkluzji.

W logice dwuwartościowej implikacja $p \rightarrow q$ ma wartość 0 lub 1 zależnie od wartości przesłanki. W logice rozmytej jeśli przesłanka spełniona jest częściowo, to konkluzja będąca rezultatem implikacji również, np. $0.5p \rightarrow 0.5q$. Wartość przesłanki *modyfikuje* funkcję przynależności do zbioru rozmytego *B* przyporządkowanego do wyjścia *y* przez zastosowanie przyjętej *metody (funkcji) implikacji.* Najczęściej stosowane metody to: - obcięcie B na poziomie spełnienia przesłanki (funcja *min*) lub przeskalowanie przez czynnik spełnienia przesłanki (funkcja *prod*).

3. Etapy projektowania układu rozmytego

Typowy schemat działania klasycznego układu rozmytego pokazuje rys.2.



Rys.2. Schemat działania układu rozmytego

Projektowanie układu sprowadza się do zdefiniowania operacji wykonywanych w poszczególnych krokach.

1. Fuzzyfikacja wejść. Polega ona na określeniu stopnia przynależności danej wartości wielkości wejściowej do każdego z odpowiadających jej zbiorów rozmytych pokrywających zakres możliwych wartości wejściowych (np. do jakiego stopnia temperatura jest niska, a do jakiego średnia). Operacja ta sprowadza się na obliczaniu funkcji lub wyszukiwaniu odpowiednich wartości w tabelach.

2. Zastosowanie operatorów logiki rozmytej do określenia stopnia, w jakim spełniona jest przesłanka w każdej z reguł. Wartościami wejściowymi są wartości przynależności

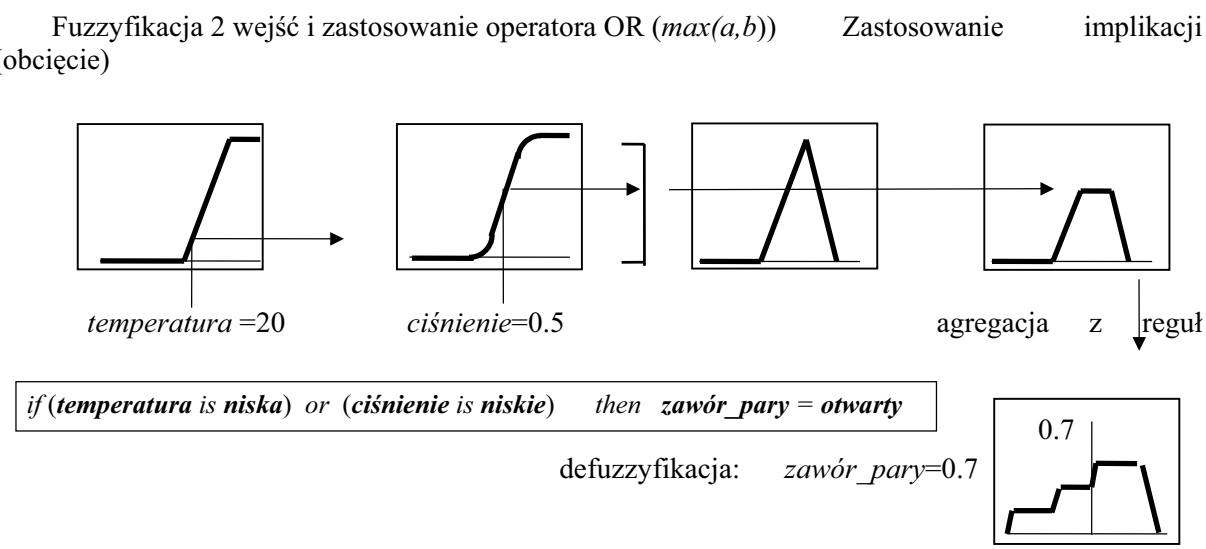
sfuzzyfikowanych wejść, na których wykonywane są rozmyte operacje logiczne (AND, OR itp.) tworzące przesłankę. Jako wynik otrzymuje się pojedynczy poziom prawdy spełnienia przesłanki.

3. Zastosowanie metody implikacji. Operacja ta sprowadza się do zmiany kształtu funkcji przynależności zbioru rozmytego konkluzji zgodnie z poziomem prawdy spełnienia przesłanki (przez obcięcie lub skalowanie). Dodatkowo przesłance każdej z reguł można nadać wagę z zakresu od 0 do 1 wyrażającą jej ważność w porównaniu z innymi. Wynikiem operacji są zbiory rozmyte odpowiadające każdej wielkości wyjściowej występującej w konkluzji.

4. Agregacja wszystkich wyjść. Polega ona na połączeniu dla każdej wielkości wyjściowej odpowiadających jej zbiorów wyjściowych ze wszystkich reguł w jeden zbiór rozmyty. Na wejścu procesu agregacji mamy listę obciętych lub przeskalowanych w wyniku implikacji funkcji przynależności danej wielkości wyjściowej w poszczególnych regułach (niekoniecznie wszystkich).

5. Defuzzyfikacja. Polega na wyznaczeniu konkretej wartości dla każdej wielkości wyjściowej ze zbioru rozmytego otrzymanego po agregacji. Najczęściej stosowaną metodą defuzzyfikacji jest obliczanie środka ciężkości obszaru pod krzywą zagregowanej funkcji przynależności (*centroid method*). Inne możliwości to średnia maksimów funkcji zbioru wyjściowego, wybór największego lub najmniejszego z maksimów czy metoda bisekcji. W układach Sugeno defuzyfikacja polega na prostym wyznaczeniu średniej ważonej singletonów wyjściowych.

Przykładowy przebieg opisanych operacji ilustruje rys.3. Warto zwrócić uwagę na to, że zaprojektowany w opisany sposób regulator rozmyty realizuje *statyczną* funkcję przejścia. Działanie dynamiczne można otrzymać przez wykonanie różniczkowania lub całkowania przed układem rozmytym i podanie otrzymanych w ten sposób sygnałów na jego wejścia.



Rys.3. Kroki działania układu rozmytego

4. *Fuzzy Logic Toolbox* do pakietu MATLAB®

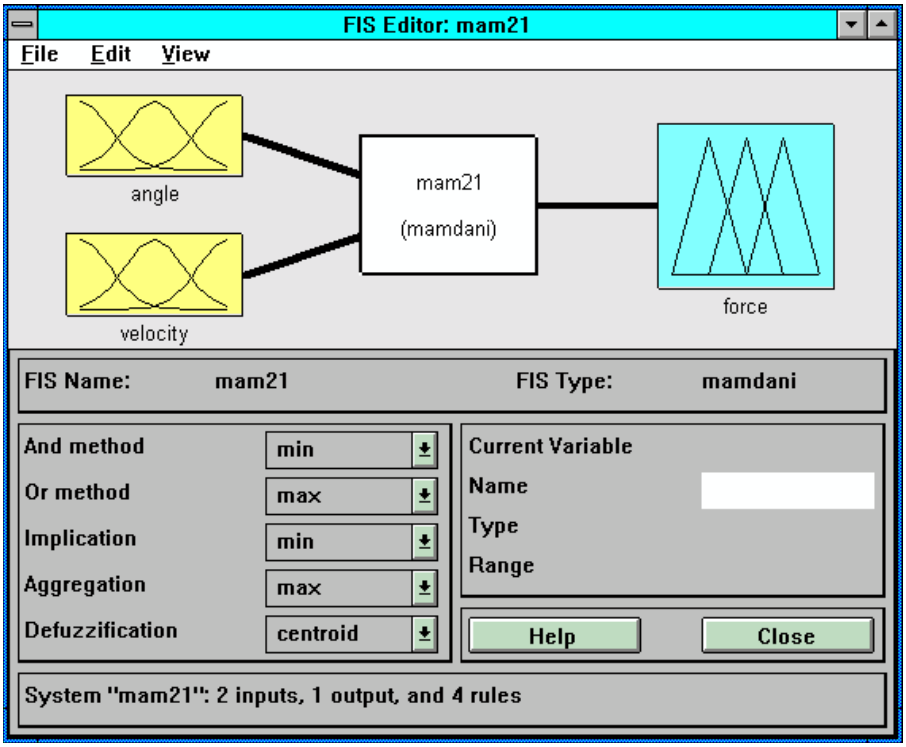
Fuzzy Logic Toolbox jest biblioteką funkcji do projektowania układów rozmytych, tzw. FIS (*Fuzzy Inference Systems*), w środowisku MATLAB. Z narzędzia tego można korzystać poprzez interfejs graficzny albo wydawanie poleceń z linii komend MATLABa. Informacja o tworzonym lub modyfikowanym układzie rozmytym jest przechowywana w pojedynczej macierzy, tzw. *FIS matrix*, i może zostać zapisana w pliku `*.fis`. Edycja układu przebiega najprościej w graficznym edytorze FIS (rys.4.), który wywołuje się z linii komend poleceniem:

```
>> fuzzy <nazwa FIS (bez rozszerzenia) >
```

Edytor FIS dysponuje szerokim zestawem możliwych do zastosowania kształtów funkcji przynależności, rozmytych operatorów logicznych, metod implikacji i agregacji. każdy z tych

elementów może być również zdefiniowany przez użytkownika w postaci funkcji (pliku skryptowego `*.m`) MATLABa. Oprócz układów klasycznych (Mamdani'ego) toolbox umożliwia projektowanie układów Takagi-Sugeno z wykorzystaniem procedury ANFIS adaptacyjnego doboru parametrów na podstawie danych uczących.

Zmienne wyjściowe i wyjściowe (podobnie jak funkcje przynależności w oknie niższego rzędu) dodaje sie lub usuwa się przy pomocy polecenia menu **Edit/Add/Remove**. Funkcje przynależności można edytować po dwukrotnym kliknięciu na okienku wybranej zmiennej lub korzystając z menu. Dostępnych jest ponad 10 różnych funkcji: trójkątne, trapezowe, gaussowskie, sigmoidalne itp. Reguły podaje się korzystając z edytora reguł (rys.6), który otwiera się po kliknięciu na środkowe okno (mam21) na rys.4. Reguły mogą być podane w formie językowej lub symbolicznej oraz mieć różne wagi (wszystkie równe 1 na rys.6).



Rys.4. Główne okno edytora FIS *Fuzzy Logic Toolbox*

Bardzo poglądowymi elementami edytora FIS są: okno Rule Viewer (rys.7) pokazujące działanie reguł, agregację zbiorów i stan wyjścia dla podanych wartości wejść (można je zmieniać przeciągając myszką pionowe linie) oraz wykres powierzchni sterowania (zmiennej wyjściowej) dla 2 wybranych zmiennych wejściowych.

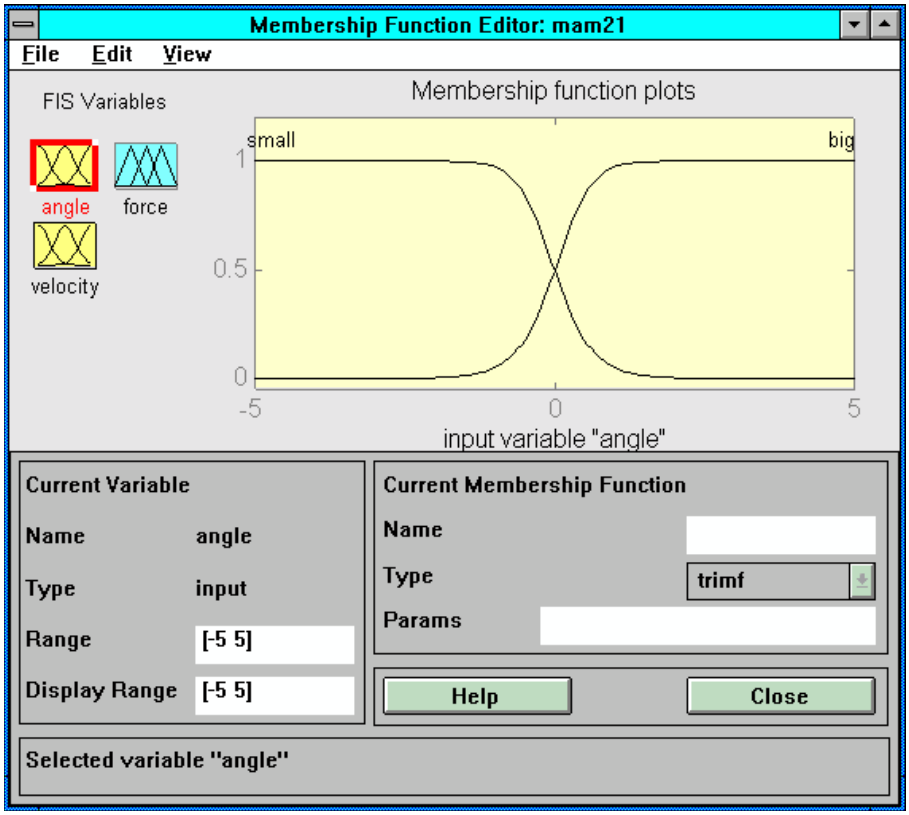
Wybrane funkcje Fuzzy Logic Toolbox jako polecenia linii komend MATLABa:

`>>fismat=readfis('filename')` - wczytanie układu `filename.fis` do zmiennej `fismat`

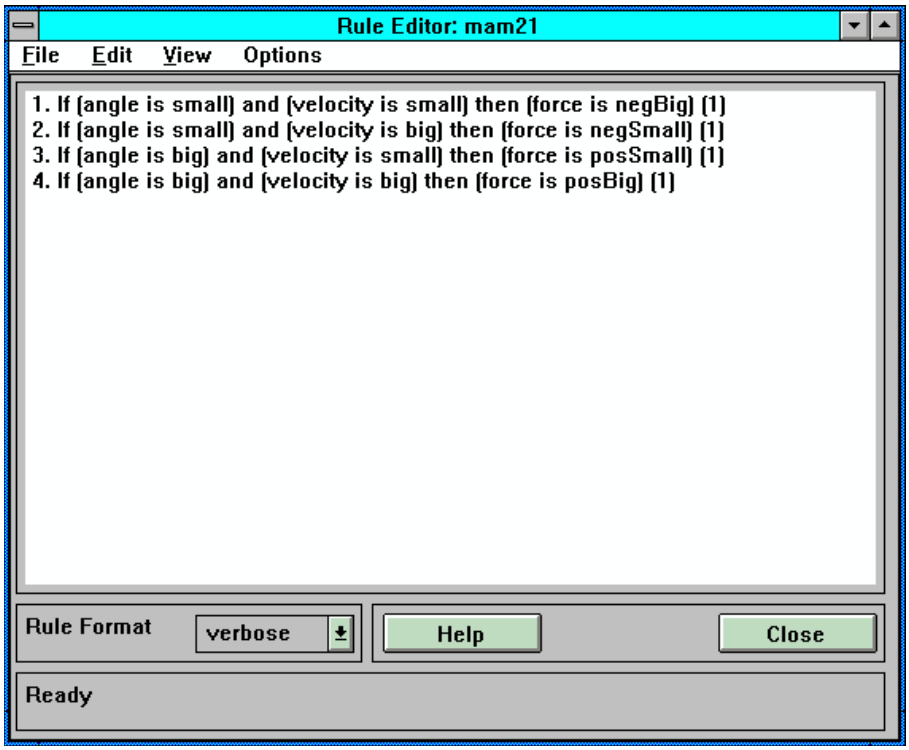
`>>plotfis(fismat)` - drukowanie diagramu wejście-wyjście układu `fismat`

`>>plotmf(fismat,vartype,varindex)`- rysowanie funkcji przynależności zmiennej o nr `varindex`; `vartype=input` lub `output` określa typ zmiennej

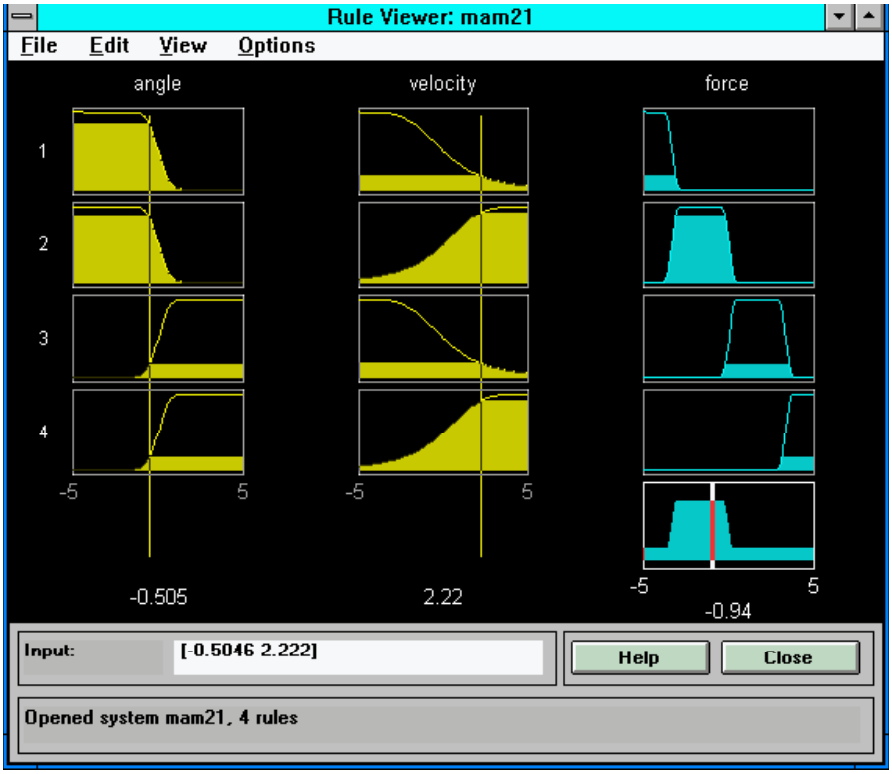
`>>gensurf(fismat,inputs,outputs), surfview(fismat)` - generowanie i rysowanie powierzchni sterowania modelu `fismat` dla 2 zmiennych wejściowych (np. `inputs=[1,3]`) i zmiennej wyjściowej (np.`output=2`)



Rys.5. Okno edycji zmiennej *angle* modelu *mam21* z rys.4



Rys.6. Okno edycji reguł modelu *mam21* z rys.4 (reguły zatwierdza się klawiszami Ctrl+Enter)



Rys.7. Okno Rule Viewer pokazujące działanie reguł dla modelu *mam21* z rys.4

Program Ćwiczenia:

I. Synteza układu wnioskowania z jedną zmienną wejściową i jedną zmienną wyjściową

II. Rozbudowa układu wnioskowania do dwóch zmiennych wejściowych

III. Rozbudowa układu wnioskowania do trzech zmiennych wejściowych

Zagadnienie:

Zaprojektować rozmyty model pomagający ocenić wysokość napiwku dołączanego do rachunku w restauracji. Wysokość napiwku ma być uzależniona od: jakości obsługi, jakości jedzenia i innych czynników subiektywnych. Klient dokonuje ostrej oceny tych kryteriów w skali od 0 do 10 punktów . Zmienne te podawane są na wejście modelu. Wyjściem jest natomiast wysokość napiwku od 5 do 25%.

Przebieg ćwiczenia:

I. Synteza układu wnioskowania z jedną zmienną wejściową i jedną zmienną wyjściową

Należy zbudować układ wnioskowania, realizujący następujące reguły:

jeżeli obsługa jest słaba to napiwek jest mały

jeżeli obsługa jest dobra to napiwek jest średni

jeżeli obsługa jest wspaniała to napiwek jest duży

1. Uruchom FIS Editor poleceniem fuzzy
2. Kliknij na żółtym polu oznaczonym input1 (pole to zostanie otoczone czerwoną ramką).
3. W białym polu po prawej stronie okna zmień input1 na obsluga i <ENTER>
4. Kliknij na niebieskim polu oznaczonym output1.
5. Zmień output1 na napiwek.
6. Zapisz system pod nazwą napiwek1 menu: **File/Export/To Workspace**
7. Podwójnie kliknij na okno obsługa
8. Ustaw **Range** i **Display** na [0 10]
9. Usuń wszystkie funkcje przynależności: menu: **Edit/Remove All MFs**
10. Dodaj 3 gaussowskie funkcje przynależności menu: **Edit/Add MFs,** type: gaussmf
11. Zmień nazwę mf1 na slaba i parametry na [1.5 0]
12. Podobnie mf2 na srednia i mf3 na wspaniala z parametrami odpowiednio [1.5 5] i [1.5 10]
13. Przejdź do edycji funkcji przynależności dla wyjścia napiwek
14. Ustaw **Range** i **Display** na [0 30]
15. Wprowadź 3 trójkątne (type: trimf) funkcje przynależności o nazwach i parametrach odpowiednio: maly, sredni, duzy, [0 5 10], [10 15 20], [20 25 30]. Parametry

można ustawiać graficznie myszką.

16. Wybierz menu: **Edit/Rules** i wprowadź 3 reguły zgodnie z założoną bazą reguł.

17. Z menu wybierz **View/View rules...**

18. Zmieniając wartość wejścia obserwuj zmiany wyjścia

19. Z menu wybierz **View/View surface...?**

II. Synteza układu wnioskowania z dwoma zmiennymi wejściowymi i jedną zmienną wyjściową

Należy zbudować układ wnioskowania, realizujący następujące reguły:

jeżeli obsługa jest słaba LUB jedzenie jest złe to napiwek jest mały

jeżeli obsługa jest dobra to napiwek jest średni

jeżeli obsługa jest wspaniała I jedzenie jest dobre to napiwek jest duży

1. Ponownie zapamiętaj poprzedni układ, zmieniając nazwę na napiwek2.
2. Dodaj dodatkową zmienną wejściową: menu: Edit/Add Variable/Input
3. Ustal nazwę na jedzenie i wprowadź dwie trójkątne funkcje przynależności w zakresie [0 10], o parametrach: zle [0 0 4] i dobre [6 10 10]
4. Zmodyfikuj bazę reguł, zgodnie z przyjętym sposobem wnioskowania.
5. Przeanalizuj działanie układu, zaobserwuj powierzchnie wnioskowania.
6. Oceń wpływ metody defuzyfikacji na działanie układu.
7. Oceń wpływ metody wnioskowania na działanie układu.

III. Rozbudowa układu wnioskowania do trzech zmiennych wejściowych

Wprowadź dodatkowy czynnik wpływający wysokość napiwku, na przykład:
atmosfera; wartości lingwistyczne zmiennej – dołująca, frapująca, obojętna, przyjazna, fantastyczna; dziedzina (wartości numeryczne zmiennej) [0, 10]
uroda kelnerki/kelnera; wartości lingwistyczne zmiennej – lepiej nie patrzeć, można byłoby zatrudnić kogoś lepszego, jest na czym zawiesić oko, przyjemnie popatrzeć, oszałamiająca; dziedzina (wartości numeryczne zmiennej) [0, 20],
samopoczucie; wartości lingwistyczne zmiennej – rozdrażnienie, obojętność, pogoda, euforia; dziedzina (wartości numeryczne zmiennej) [0, 10]

Dla precyzyjnego opisu wysokości napiwków rozbuduj funkcje przynależności, na przykład:

a) *śmieszny* oznacza napiwek wynoszący mniej, bądź około 1 % ceny posiłku
b) *niski* oznacza napiwek wynoszący około 3 % ceny posiłku
c) *średni* oznacza napiwek wynoszący około 10 % ceny posiłku
d) *wysoki* oznacza napiwek wynoszący około 20 % ceny posiłku
e) *ekscentryczny* oznacza napiwek wynoszący około, bądź powyżej 30 % ceny posiłku

Bazę reguł zaproponuj samodzielnie!