Sprawozdanie 2

Metody inteligencji obliczeniowej – Informatyka Stosowana, WFIIS,
Jakub Salamon, II rok

Celem zajęć laboratoryjnych nr 2 była aproksymacja danych wejściowych.

Aproksymacja to proces określania rozwiązań przybliżonych na podstawie rozwiązań znanych, które są bliskie rozwiązaniom dokładnym.

Metoda “ręczna”

Zacząłem od analizy problemu w programie LibreOffice Calc. Wykonując odpowiednie działania znalazłem tylko 25 niepowtarzających się reguł opisujących układ. Metoda polegała na znalezieniu wartości minimalnej i maksymalnej w danych, podzieleniu obszaru danych wejściowych na trzy równe części i pokolorowaniu komórek na odpowiedni dla danej części tego przedziału kolor. Pozwoliło to na znalezienie zależności pomiędzy danymi wejściowymi a wyjściem. Reguły te wyglądają następująco:

1. If (input1 is y) and (input2 is y) and (input3 is g) then (output1 is y) (1)
2. If (input1 is r) and (input2 is y) and (input3 is g) then (output1 is y) (1)
3. If (input1 is r) and (input2 is y) and (input3 is y) then (output1 is y) (1)
4. If (input1 is r) and (input2 is r) and (input3 is y) then (output1 is y) (1)
5. If (input1 is r) and (input2 is r) and (input3 is y) then (output1 is r) (1)
6. If (input1 is r) and (input2 is y) and (input3 is y) then (output1 is r) (1)
7. If (input1 is y) and (input2 is r) and (input3 is y) then (output1 is y) (1)
8. If (input1 is y) and (input2 is r) and (input3 is g) then (output1 is y) (1)
9. If (input1 is y) and (input2 is y) and (input3 is g) then (output1 is g) (1)
10. If (input1 is y) and (input2 is y) and (input3 is y) then (output1 is g) (1)
11. If (input1 is y) and (input2 is y) and (input3 is y) then (output1 is y) (1)
12. If (input1 is y) and (input2 is g) and (input3 is g) then (output1 is g) (1)
13. If (input1 is y) and (input2 is g) and (input3 is g) then (output1 is y) (1)
14. If (input1 is g) and (input2 is g) and (input3 is g) then (output1 is g) (1)
15. If (input1 is g) and (input2 is g) and (input3 is y) then (output1 is g) (1)
16. If (input1 is g) and (input2 is y) and (input3 is y) then (output1 is g) (1)
17. If (input1 is g) and (input2 is y) and (input3 is y) then (output1 is y) (1)
18. If (input1 is g) and (input2 is g) and (input3 is y) then (output1 is y) (1)
19. If (input1 is y) and (input2 is y) and (input3 is r) then (output1 is y) (1)
20. If (input1 is r) and (input2 is r) and (input3 is r) then (output1 is r) (1)
21. If (input1 is r) and (input2 is y) and (input3 is r) then (output1 is r) (1)
22. If (input1 is r) and (input2 is y) and (input3 is r) then (output1 is y) (1)
23. If (input1 is r) and (input2 is g) and (input3 is r) then (output1 is y) (1)
24. If (input1 is y) and (input2 is g) and (input3 is r) then (output1 is y) (1)
25. If (input1 is r) and (input2 is g) and (input3 is r) then (output1 is r) (1)

r to pierwsza, **y** to druga, **g** to trzecia z trzech części

Stopień skomplikowania systemu rośnie wraz ze wzrostem ilości funkcji przynależności i liczby wejść

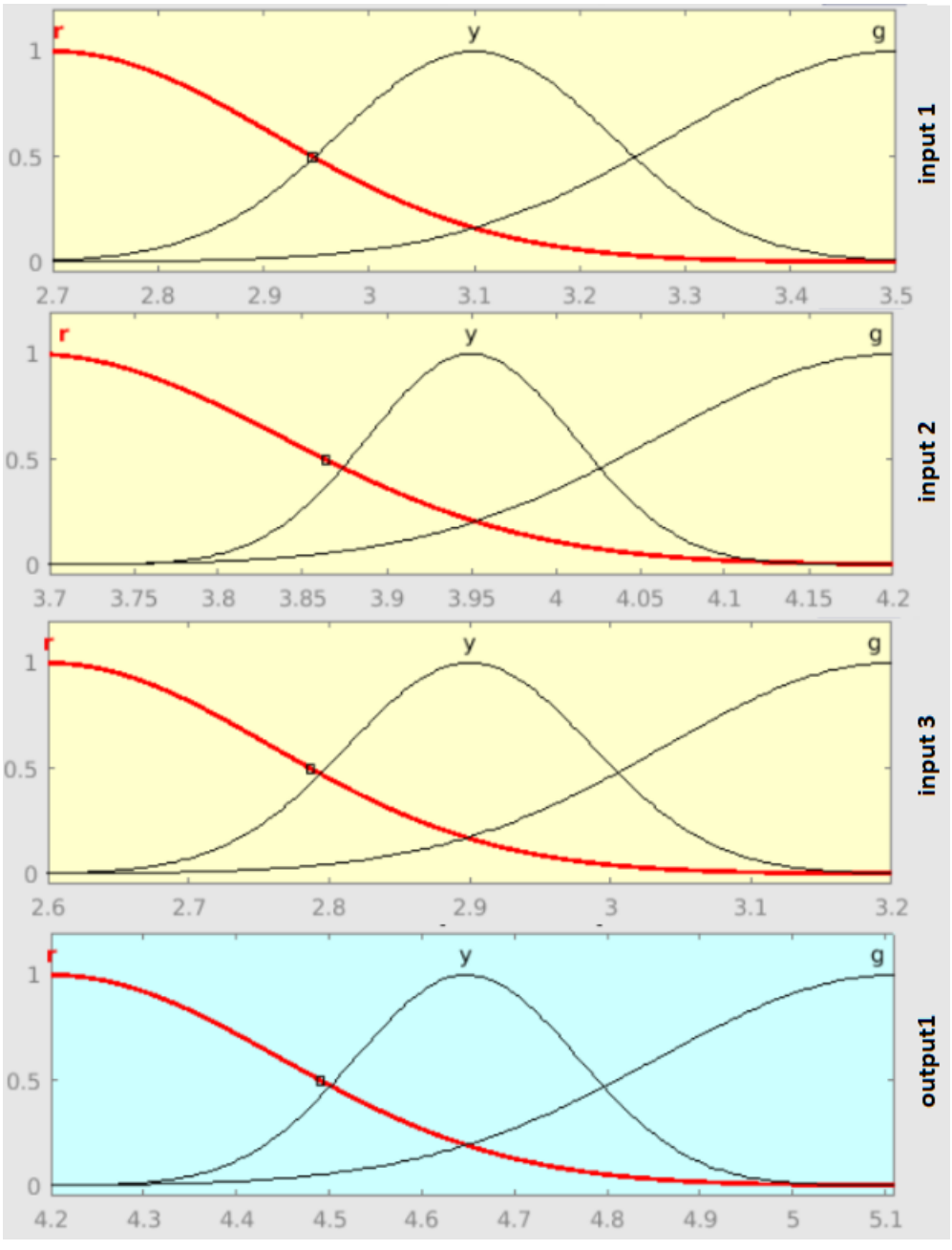
Według poniższego wzoru powinniśmy otrzymać $3^3 = 27$ reguł opisujących ten układ.

$$\mathbf{R = (F_p)^w}$$

R – liczba reguł
F_p - liczba funkcji przynależności
w – liczba wejść

Nie dało się jednak wyznaczyć dodatkowych dwóch, niepowtarzających się reguł tą metodą.

Następnie wyznaczyłem po 3 funkcje przynależności typu Gaussa dla każdego wejścia i wyjścia:



Kolejnym krokiem było wyznaczenie błędu średniokwadratowego dla tej metody.

Błąd średniokwadratowy

wzór

$$Y = \frac{1}{n} \sum_{1}^{n} (X_i - \overline{X_i})^2$$

gdzie ,
n – *liczba elementow*
X_i – *konkretny pomiar*
$\overline{X}$ – *pomiar , z którym porównujemy*

Kod matlab do wyznaczania błędu średniokwadratowego z ręcznej aproksymacji

---

```
% wczytywanie danych z pliku lab02_data.csv
dane = csvread('lab02_data.csv');

input_data = dane;
input_data(:,4) = [];
output_data = dane(:, 4);

% import danych wejsciowych do FIS
fis_manual = readfis('reczne_reguly.fis');
output_manual_fis = evalfis(input_data, fis_manual);

% obliczanie bledu sredniokwadratowego dla recznych regul, podpunkt 1
blad_sredniokwadratowy_manual_rules = 0;
suma = 0;

for i = 1 : size(output_manual_fis(:,1))
   suma = suma + ((output_manual_fis(i, 1) - output_data(i, 1))^2);
end

blad_sredniokwadratowy_manual_rules = suma / 255
```

Błąd średniokwadratowy dla ręcznego wyznaczania reguł i funkcji przynależności wyniósł:

0.0104

Metoda analizy skupień

Analiza skupień to metoda klasyfikacji, dokonująca grupowania elementów we względnie jednorodne klasy. Podstawą grupowania jest głównie podobieństwo pomiędzy elementami.

Za pomocą programu MATLAB stworzyłem system logiki rozmytej dopasowany do otrzymanych danych. Funkcja genfis3 generuje system logiki rozmytej typu Mamdaniego. Otrzymujemy funkcje przynależności typu gaussmf, a funkcje tą wyznacza t-norma. Funkcja genfis3 korzysta z algorytmu fuzzy c-means. Stosując ten typ systemu otrzymujemy przejrzystą i intuicyjną baze reguł.

Algorytm fuzzy c-means jest formą klasteryzacji danych, w którym każda informacja wejściowa może należeć do więcej niż do jednego klastra. Dane w jednym klastrze są do siebie podobne jak tylko się da, a dane należące do innych klastrów są od siebie maksymalnie różne.

Algorytm fuzzy c-means:

Próbuje podzielić skończoną liczbe n danych:

$X = \{x_1, \ldots, x_n\}$

do zbioru klastrów c

Daje nam skończony zestaw danych. Algorytm zwraca liste centrów klastrów c

$C = \{c_1, \ldots, c_c\}$

oraz dzieli macierz

$W = w_{i,j} \in [0, 1]$, $i = 1, \ldots, n$, $j = 1, \ldots, c$, gdzie każdy element $w_{i,j}$ mówi nam o stopniu w jakim każdy z elementów x_i przynależy do klastra c_j

Celem Fuzzy c-means jest zminimalizowanie funkcji celu.

$$\arg\min_{C} \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij}^{m} \|\mathbf{x}_i - \mathbf{c}_j\|^2$$

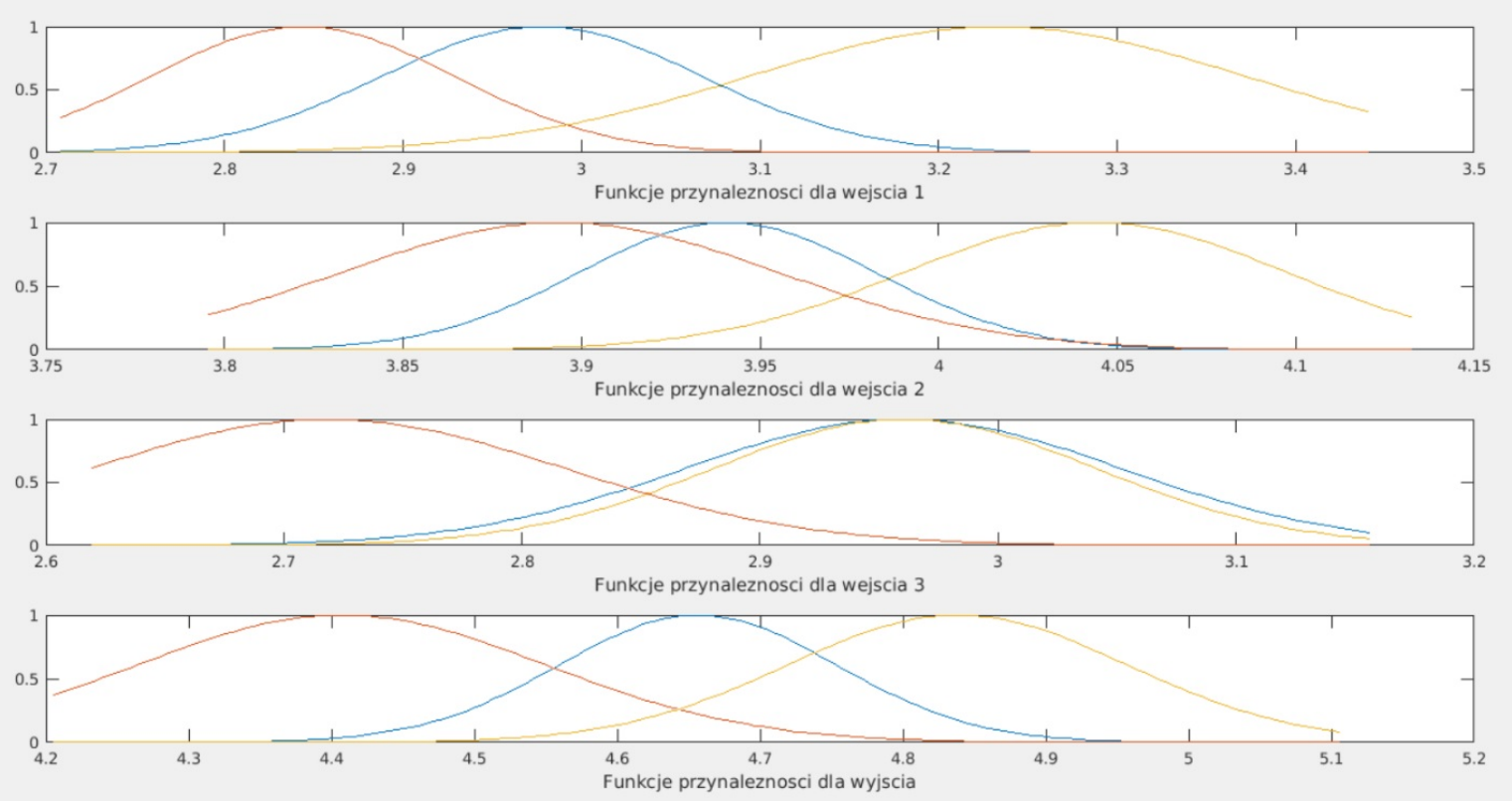
gdzie

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|\mathbf{x}_i - \mathbf{c}_j\|}{\|\mathbf{x}_i - \mathbf{c}_k\|} \right)^{\frac{2}{m-1}}}$$

Kod MATLAB

```
dane = csvread('lab02_data.csv');

% liczba wejsc
N = 3;

inputData = dane(:,1:N);
outputData = dane(:,N+1);
options = [NaN 25 0.001 0];

opt = NaN(3, 1);
opt(4) = 0;
%generowanie fis
fis = genfis3(inputData, outputData, 'mamdani', N, opt);
%pokazuje reguly
showrule(fis)

%rysowanie wykresow funkcji przynaleznosci
[x, mf] = plotmf(fis, 'input', 1);
subplot(4, 1, 1)
plot(x, mf)
xlabel('Funkcje przynaleznosci dla wejscia 1');

[x,mf] = plotmf(fis,'input',2);
subplot(4,1,2)
plot(x,mf)
xlabel('Funkcje przynaleznosci dla wejscia 2')

[x,mf] = plotmf(fis,'input',3);
subplot(4,1,3)
plot(x,mf)
xlabel('Funkcje przynaleznosci dla wejscia 3')

[x,mf] = plotmf(fis,'output',1);
subplot(4,1,4)
plot(x,mf)
xlabel('Funkcje przynaleznosci dla wyjscia')

% import danych wejsciowych do FIS
output_fis = evalfis(inputData, fis);

% obliczanie bledu sredniokwadratowego dla recznych regul, podpunkt 1
blad_sredniokwadratowy_manual_rules = 0;
suma = 0;

for i = 1 : size(output_fis(:,1))
   suma = suma + ((output_fis(i, 1) - outputData(i, 1))^2);
end
blad_sredniokwadratowy_manual_rules = suma / 255
```

Program wygenerował następujące funkcje przynależności:



oraz takie reguły:

```
1. If (in1 is in1cluster1) and (in2 is in2cluster1) and (in3 is in3cluster1) then (out1 is out1cluster1) (1)
2. If (in1 is in1cluster2) and (in2 is in2cluster2) and (in3 is in3cluster2) then (out1 is out1cluster2) (1)
3. If (in1 is in1cluster3) and (in2 is in2cluster3) and (in3 is in3cluster3) then (out1 is out1cluster3) (1)
```

Przy użyciu metody analizy skupień algorytm wygenerował system logiki rozmytej z takim błędem średniokwadratowym dla aproksymowanych danych:

0.0085

Błąd ten maleje dla większej liczby funkcji przynależności co widać po małej zmianie w programie i przetestowaniu tego.
Dla 5 funkcji przynależności błąd wyniósł 0.0054, dla 10 – 0.0036, a dla 15 – 0.0033.

Różnica w błędzie średniokwadratowym wyniosła 0.0019. Wynika z tego, że błąd przy aproksymacji automatycznej był o około 18% niższy niż dla aproksymacji ręcznej.

Jak widać metoda automatyczna jest lepsza. Otrzymujemy lepszą aproksymacje danych. Jest dokładniejsza i szybsza. Przy ręcznym sposobie trzeba wykonać dużo więcej pracy, co nie skutkuje lepszymi wynikami niż w przypadku procesu automatycznej aproksymacji. Wynika z tego, że ręczna aproksymacja danych jest nieopłacalna i lepiej stosować automatyczną.

Jakub Salamon
Informatyka Stosowana, WFIIS, II rok

Źródła:
1. Dokumentacja MATLAB
2. wikipedia.org