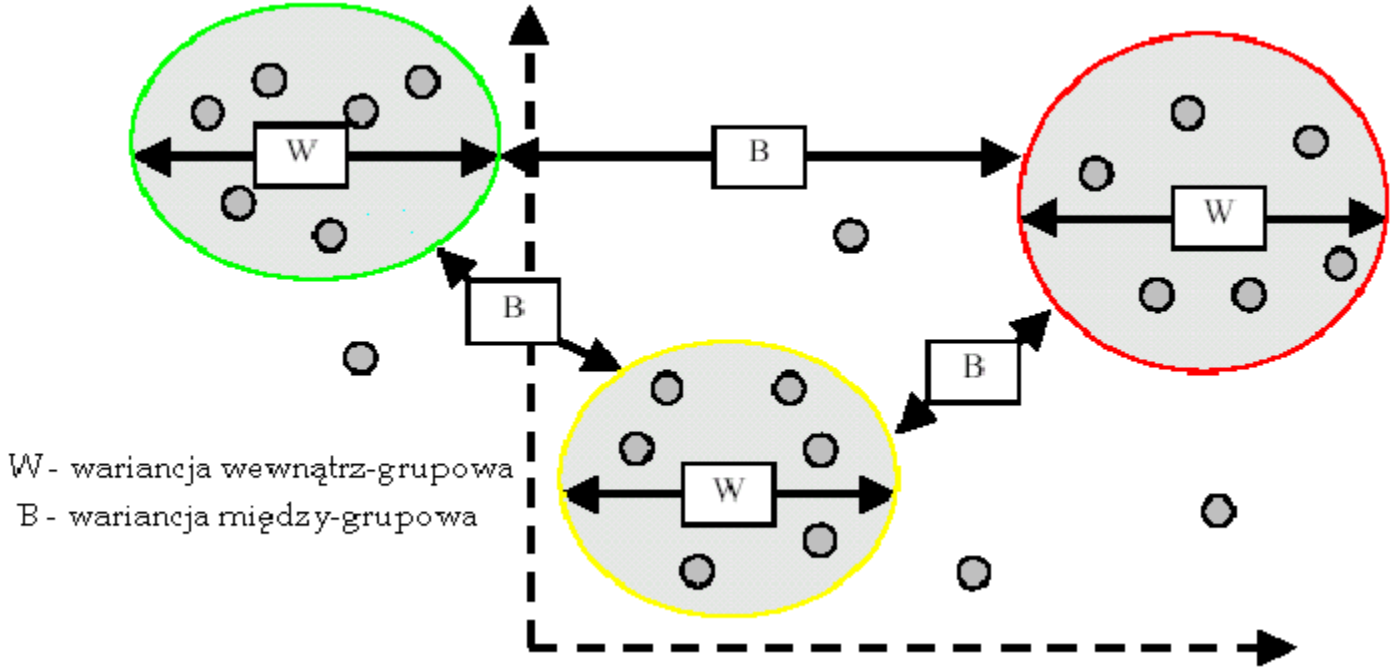
Analiza skupień

Grupowanie (analiza skupień) (ang. data clustering) – pojęcie z zakresu eksploracji danych oraz uczenia maszynowego, wywodzące się z szerszego pojęcia, jakim jest **klasyfikacja bezwzorcowa**.

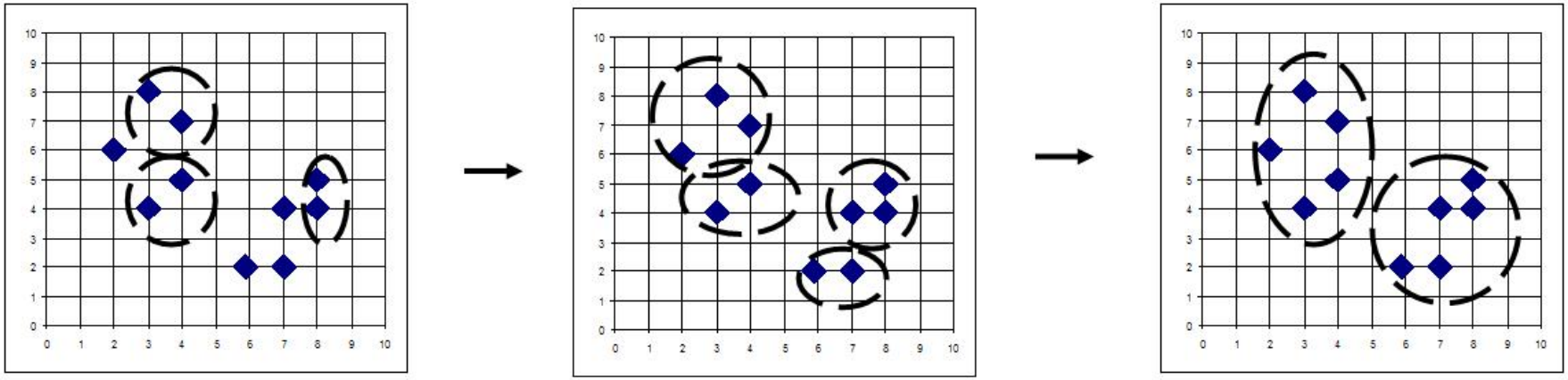
Analiza skupień jest metodą tzw. klasyfikacji bez nadzoru (ang. unsupervised learning). **Jest to metoda dokonująca grupowania elementów we względnie jednorodne klasy**. Podstawą grupowania w większości algorytmów jest **podobieństwo pomiędzy elementami** – wyrażone przy pomocy funkcji (metryki) podobieństwa.
Poprzez grupowanie można również rozwiązać problemy z gatunku odkrywania struktury w danych oraz dokonywanie uogólniania. Grupowanie polega na wyodrębnianiu grup (klas, podzbiorów). Grupowanie jako jedna z metod pozyskiwania wiedzy, a tym samym eksploracji danych, jest ściśle uwarunkowana źródłem danych oraz oczekiwaną postacią rezultatów.

Idea klasteryzacji

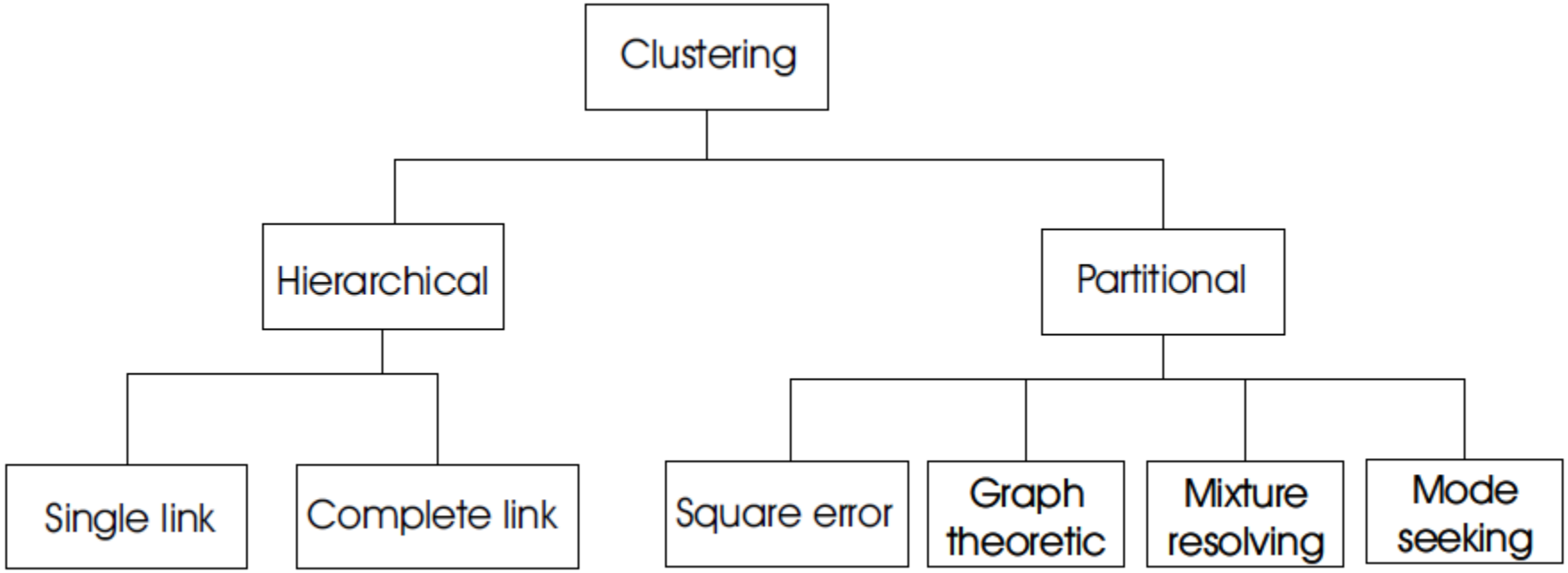
Dążymy do takiego podziału zbioru danych aby wariancja **wewnątrzgrupowa** (w każdym z klastrów) była możliwie **mała** a jednocześnie wariancja **miedzy-grupowa** możliwie **duża**.



Przykład



Techniki klasteryzacji

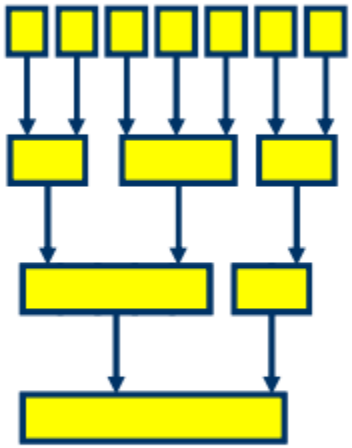


Metody hierarchiczne

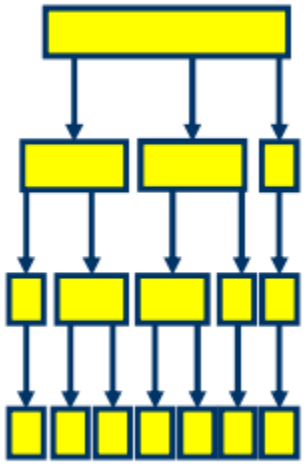
Metody hierarchiczne polegają na tym, że tworzy się hierarchię klasyfikacji. Jeśli mamy *n* obserwacji, to uzyskana hierarchia liczy *n* klasyfikacji składających się odpowiednio z 1, 2, 3, ..., *n* klas. Klasyfikacja zawierająca jedną klasę stanowi zbiór wszystkich obserwacji, natomiast złożona z *n* klas zawiera wyłącznie klasy jednoelementowe (klasa = pojedyncza obserwacja).

Metody hierarchiczne (cd)

W zależności od sposobu otrzymania ciągu klasyfikacji wyróżnia się dwa rodzaje metod hierarchicznych:



aglomeracyjne (grupowania)



podziału

Metody aglomeracyjne

1) Tworzymy macierz odległości wymiaru n x n.
2) Zakładamy, że każda obserwacja sama tworzy klasę, czyli mamy n klas jednoelementowych.
3) W każdym etapie grupowania znajdujemy taką parę klas, między którymi odległość jest najmniejsza. Obie klasy następnie łączymy w jedną, czyli liczba klas zmniejsza się o 1 (po r-tym etapie grupowania liczba klas jest równa $n - r$).
4) Następnie musimy określić odległość nowo powstałej klasy od pozostałych klas. Odległości zapisujemy w nowej macierzy odległości, która jest wymiaru $(n\text{-}r)$ x $(n\text{-}r)$.
5) Procedurę opisaną w punktach 3) i 4) powtarzamy aż do uzyskania klasy zawierającej wszystkie obserwacje (czyli n –1 krotnie).

Metody obliczenia odległości między grupami

Zauważmy, że przy **grupowaniu aglomeracyjnym czy rozdzielającym**, potrzebne są odległości między grupami elementów danych. Niżej przedstawiono najważniejsze miary odległości między grupami elementów.

Metoda pojedynczego połączenia (ang.single link). Odległość między dwoma skupieniami jest zdefiniowana jako **odległość między dwoma najbliższymi punktami**, po jednym z każdego skupienia:

$$D(C_i, C_j) = \min_{p \in C_i,\ p' \in C_j} d(p, p')$$

Metoda *całkowitego połączenia* (ang.*complete link*). Odległość między dwoma skupieniami bierze się **odległość między dwoma najbardziej oddalonymi punktami**, po jednym z każdego skupienia:

$$D(C_i, C_j) = \max_{p \in C_i,\ p' \in C_j} d(p, p')$$

Metody obliczenia odległości między grupami(cd)

Metoda *odległości między środkami* (ang.*mean distance*). Odległość między dwoma skupieniami jest zdefiniowana jako **odległość między dwoma ich środkami ciężkości**:

$$D(C_i, C_j) = d(m_i, m_j)$$

Metoda *średniej odległości* (ang.*average distance*). Odległość między dwoma skupieniami jest zdefiniowana jako **średnia odległość między każdymi dwoma punktami**, po jednym z każdego skupienia:

$$D(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} d(p, p')$$

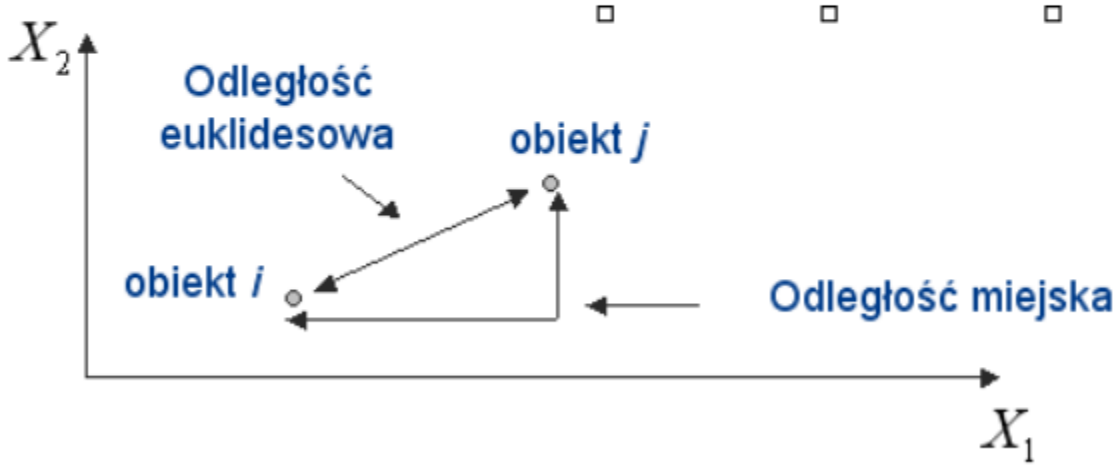
we wzorze n_i i n_j oznaczają liczbę elementów skupienia C_i i C_j, odpowiednio.

Metryki odległości

Najczęściej stosowane sposoby określania odległości opierają się na następujących metrykach:

1) odległość euklidesowa: $d_{ij} = \sqrt{\sum_{l=1}^{p} (x_{il} - x_{jl})^2}$

2) odległość miejska: $d_{ij} = \sum_{l=1}^{p} | x_{il} - x_{jl} |$



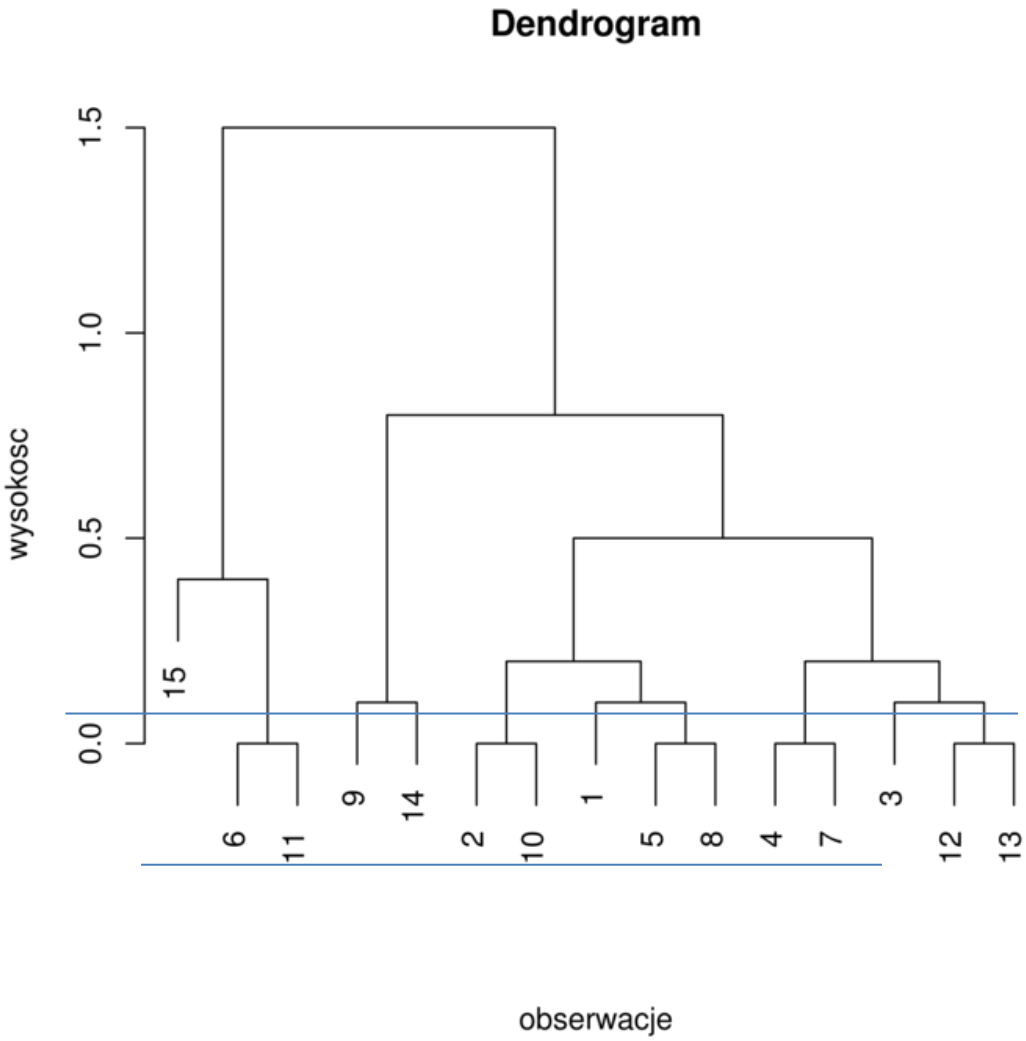
Metryki odległości (cd)

3) odległość Minkowskiego: $d_{ij} = (\sum_{l=1}^{n} | x_{il} - x_{jk} |^{p})^{1/p}$

Jeśli przyjmiemy p =1, to otrzymamy odległość miejską, a gdy p =2, otrzymujemy odległość Euklidesową.

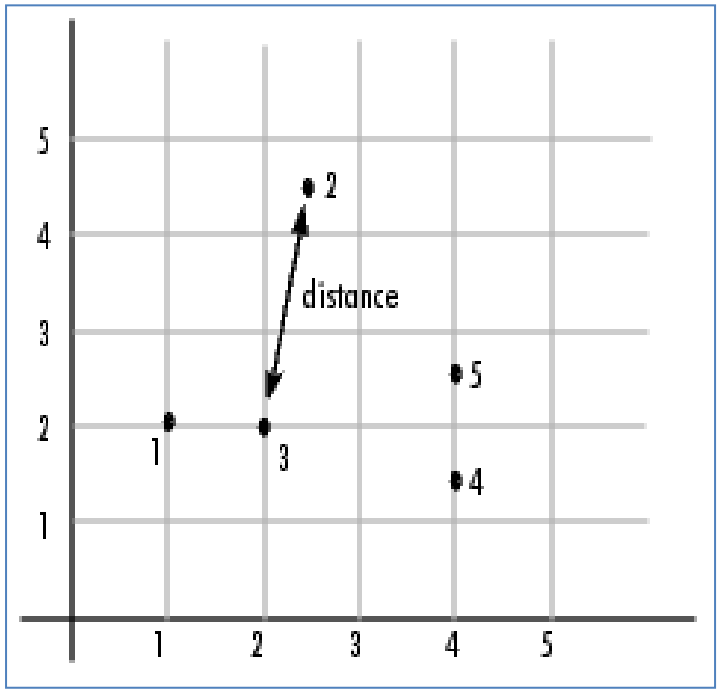
Dendrogram

Dendrogram jest metodą ilustracji wyników klasteryzacji hierarchicznej. Możemy obserwować od dołu dendrogramu jak kolejne klastry się łączą i dla jakiej wysokości (odległości klastrów) to zachodzi



Przykład

```
X = [1 2;2.5 4.5;2 2;4 1.5;4 2.5]
```




```
Y = pdist(X)
```

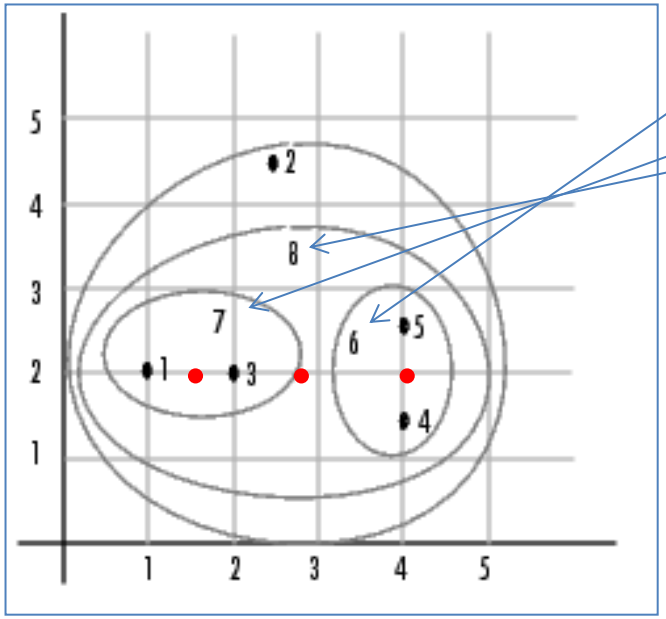
```
Y =
  Columns 1 through 5
    2.9155    1.0000    3.0414    3.0414    2.5495
  Columns 6 through 10
    3.3541    2.5000    2.0616    2.0616    1.0000
```

```
squareform(Y)
```

```
         0    2.9155    1.0000    3.0414    3.0414
    2.9155         0    2.5495    3.3541    2.5000
    1.0000    2.5495         0    2.0616    2.0616
    3.0414    3.3541    2.0616         0    1.0000
    3.0414    2.5000    2.0616    1.0000         0
```

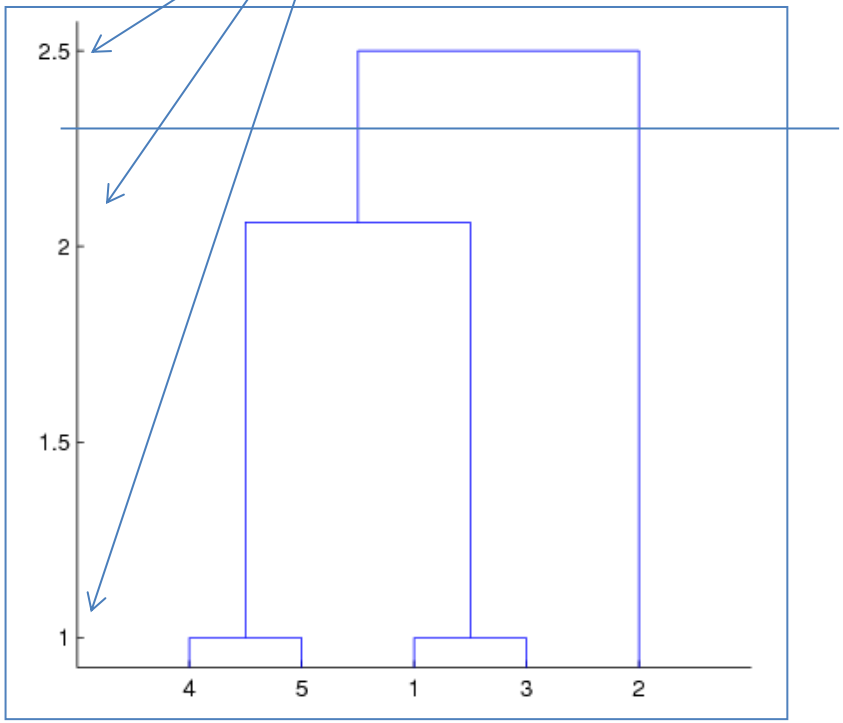
Połączenie

Z = linkage(Y)



4.0000	5.0000	1.0000
1.0000	3.0000	1.0000
6.0000	**7.0000**	2.0616
2.0000	**8.0000**	2.5000

dendrogram(Z)



Zagadnienie klasteryzacji klasycznej

Dane eksperymentalne

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & & & \\ x_{M1} & x_{M2} & \cdots & x_{Mn} \end{bmatrix}$$

Ilość punktów - M

Klastry danych

$$\bigcup_{i=1,c} A_i = \mathbf{X}$$

$$A_i \cap A_j = \varnothing, \quad i, j = \overline{1,c}, \quad i \neq j$$

$$\varnothing \subset A_i \subset \mathbf{X}, \quad i = \overline{1,c}$$

A_i – klastry

Ilość klastrów – c

Egzemplarz nie należy do kilku klastrów jednocześnie

Nie może być pustych klastrów

Funkcja charakterystyczna

$$\Phi = [\varphi_{ki}], \quad \varphi_{ki} \in \{0,1\}, \quad k = \overline{1,M}, \quad i = \overline{1,c}$$

$$c \ll M$$

$$\sum_{i=1,c} \varphi_{ki} = 1, \quad k = \overline{1,M}$$

$$0 < \sum_{k=1,M} \varphi_{ki} < M, \quad i = \overline{1,c}$$

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & & & \\ x_{M1} & x_{M2} & \cdots & x_{Mn} \end{bmatrix} \Rightarrow \Phi = \begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1c} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2c} \\ \cdots & & & \\ \varphi_{M1} & \varphi_{M2} & \cdots & \varphi_{Mc} \end{bmatrix}$$

Jakość klasteryzacji

$$J = \sum_{i=1,c} \sum_{X_k \in \mathrm{A}_i} \left\| V_i - X_k \right\|^2 \to \min$$

$$V_i = \frac{1}{\left| A_i \right|} \sum_{X_k \in A_i} X_k$$

centrum i-go klastra

$$A_i = \left\{ X_p : \varphi_{pi} = 1,\ p = \overline{1, M} \right\}$$

Zagadnienie optymalizacji

$$J = \sum_{i=1,c} \sum_{X_k \in \mathrm{A}_i} \left\| V_i - X_k \right\|^2 \to \min$$

Przykład

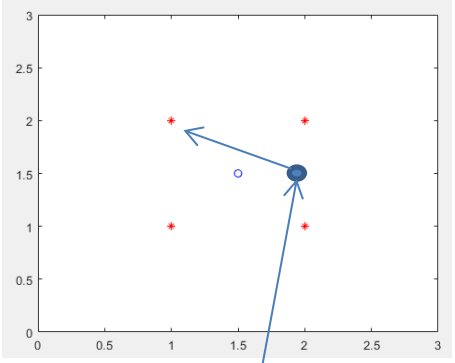
$$J = \sum_{i=1,c} \sum_{X_k \in A_i} \|V_i - X_k\|^2 \qquad J \to \min$$

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 2 & 1 \\ 2 & 2 \end{bmatrix} \Rightarrow \Phi = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

C=1

$$V_1 = \frac{1}{4} \sum_{X_k \in A_1} X_k = [1.5, 1.5]$$

$$A_1 = \{X_1, X_2, X_3, X_4\}$$



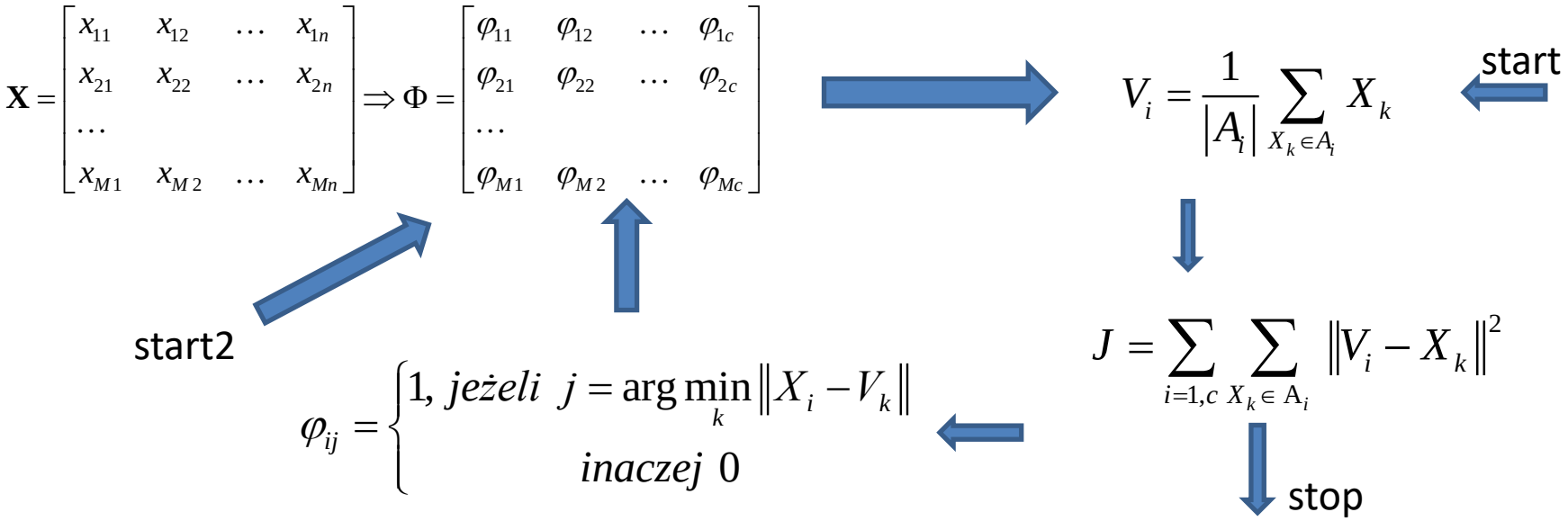

```
>> (pdist([1.5,1.5;1 1;1 2;2 1;2 2]).*2)'
    1.4142135623731
    1.4142135623731
    1.4142135623731
    1.4142135623731
                 2
                 2
   2.82842712474619
   2.82842712474619
                 2
```

J_1=5.65685424949238

6.47213595499958

Algorytm K-means

Ogólna idea algorytmu K-means (MacQueen,1967) polega na tym, że rozpoczyna się od losowo wybranego grupowania punktów, następnie **ponownie przypisuje się punkty tak, aby *otrzymać największy spadek w funkcji oceny***, po czym przelicza się zaktualizowane skupienia, **po raz kolejny przypisuje się punkty** i tak dalej aż do momentu, w którym nie ma już żadnych zmian w funkcji oceny lub w składzie skupień. To zachłanne podejście ma tę zaletę, że jest proste i gwarantuje otrzymanie co najmniej lokalnego maksimum (minimum) funkcji oceny.



Zasada działania algorytmu

1. **Ustalamy liczbę skupień.**
Jedną z metod ustalenia ilości skupień jest umowny jej wybór i ewentualna późniejsza zmiana tej liczby w celu uzyskania lepszych wyników. Wybór liczby skupień może być oparty również na wynikach innych analiz.

2. **Ustalamy wstępne środki skupień.**
Środki skupień tak zwane centroidy możemy dobrać na kilka sposobów: losowy wybór k obserwacji, wybór k pierwszych obserwacji, dobór w taki sposób, aby zmaksymalizować odległości skupień. Jedną z najczęściej stosowanych metod jest kilkakrotne uruchomienie algorytmu i wybór najlepszego modelu, gdy wstępnie środki skupień były wybierane losowo.

Zasada działania algorytmu (cd)

3**. Obliczamy odległości obiektów od środków skupień.**
Wybór metryki jest bardzo istotnym etapem w algorytmie. Wpływa ona na to, które z obserwacji będą uważane za podobne, a które za zbyt różniące się od siebie. Najczęściej stosowaną odległością jest odległość euklidesowa. Stosuje się również kwadrat tej odległości czy też odległość Czebyszewa.

4. **Przypisujemy obiekty do skupień**
Dla danej obserwacji porównujemy odległości od wszystkich skupień i przypisujemy ją do skupienia, do którego środka ma najbliżej.

Zasada działania algorytmu (cd)

5. **Ustalamy nowe środki skupień**
Najczęściej nowym środkiem skupienia jest punkt, którego współrzędne są średnią arytmetyczną współrzędnych punktów należących do danego skupienia.

6. **Wykonujemy kroki 3,4,5 do czasu, aż warunek zatrzymania zostanie spełniony**.
Najczęściej stosowanym warunkiem stopu jest ilość iteracji zadana na początku lub brak przesunięć obiektów pomiędzy skupieniami.

Algorytm K-means (cd)

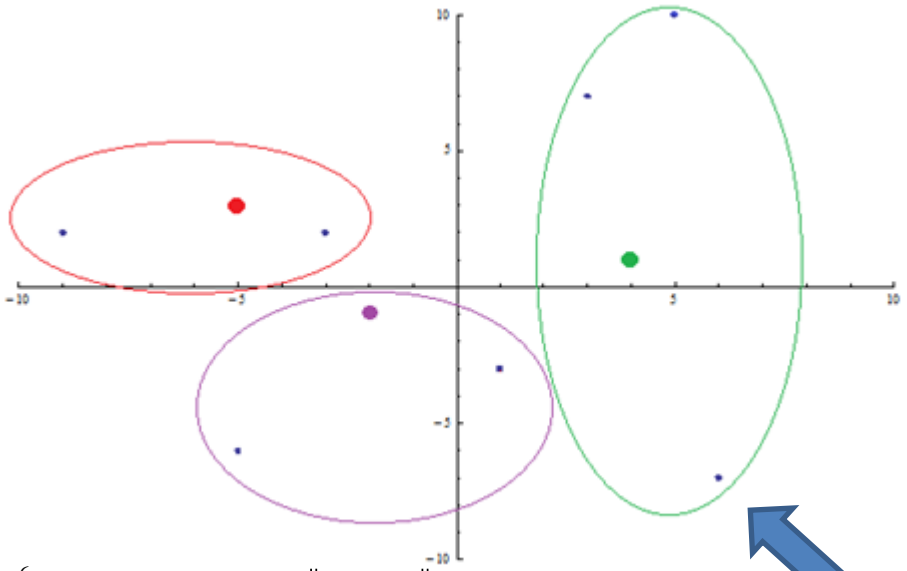
Algorithm 1: Algorytm K-means

wejście: P - zbiór przykładów P,
$K \in \Re$ – liczba skupień.
wyjście: $C = \{C_1, ...C_K\}$ - zbiór skupień.
begin
  while *są zmiany w skupieniach* C_k **do**
    Podziel zbiór danych na dowolnych k rozłącznych zbiorów;
    Dla każdej grupy wyznacz środek ciężkości;
    Podziel przykłady do najbliższego środka;
end

Przykład

Punkty pomiarowe (1,-3),(3,7),(−3,2),(−2,−1),(6,−7),(−5,−6),(−9,2),(−5,3),(4,1),(5,10)

Ustalmy, że chcemy nasze punkty zakwalifikować do **trzech skupień**. Wybierzmy losowo środki skupień: (-2,-1), (-5,3), (4,1).



$$\varphi_{ij} = \begin{cases} 1, \; jeżeli \;\; j = \arg\min_k \left\| X_i - V_k \right\| \\ inaczej \;\; 0 \end{cases}$$

Macierz odległości

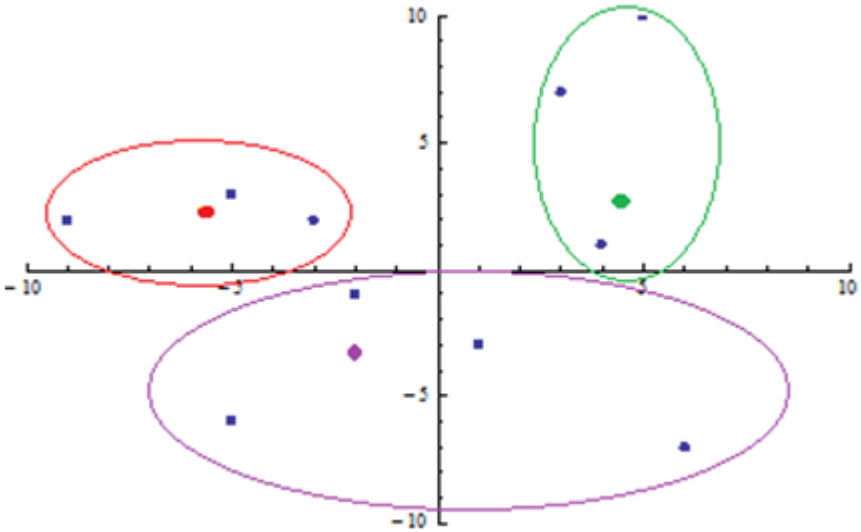
	C1 (-2,-1)	C2 (-5,3)	C3 (4,1)
(1,-3),	3.6	8.5	5.0
(3,7),	9.5	9.0	6.1
(−3,2),	3.1	2.2	7.1
(−2,−1),	0	5.0	6.3
(6,−7),	10	14	8.3
(−5,−6),	5.8	9.0	11.0
(−9,2),	7.6	4.1	13.0
(−5,3),	5.0	0	9.2
(4,1),	6.3	9.2	0
(5,10)	13.0	12.0	9.1

Teraz przyporządkujemy odpowiednim elementom ich skupienia.

min

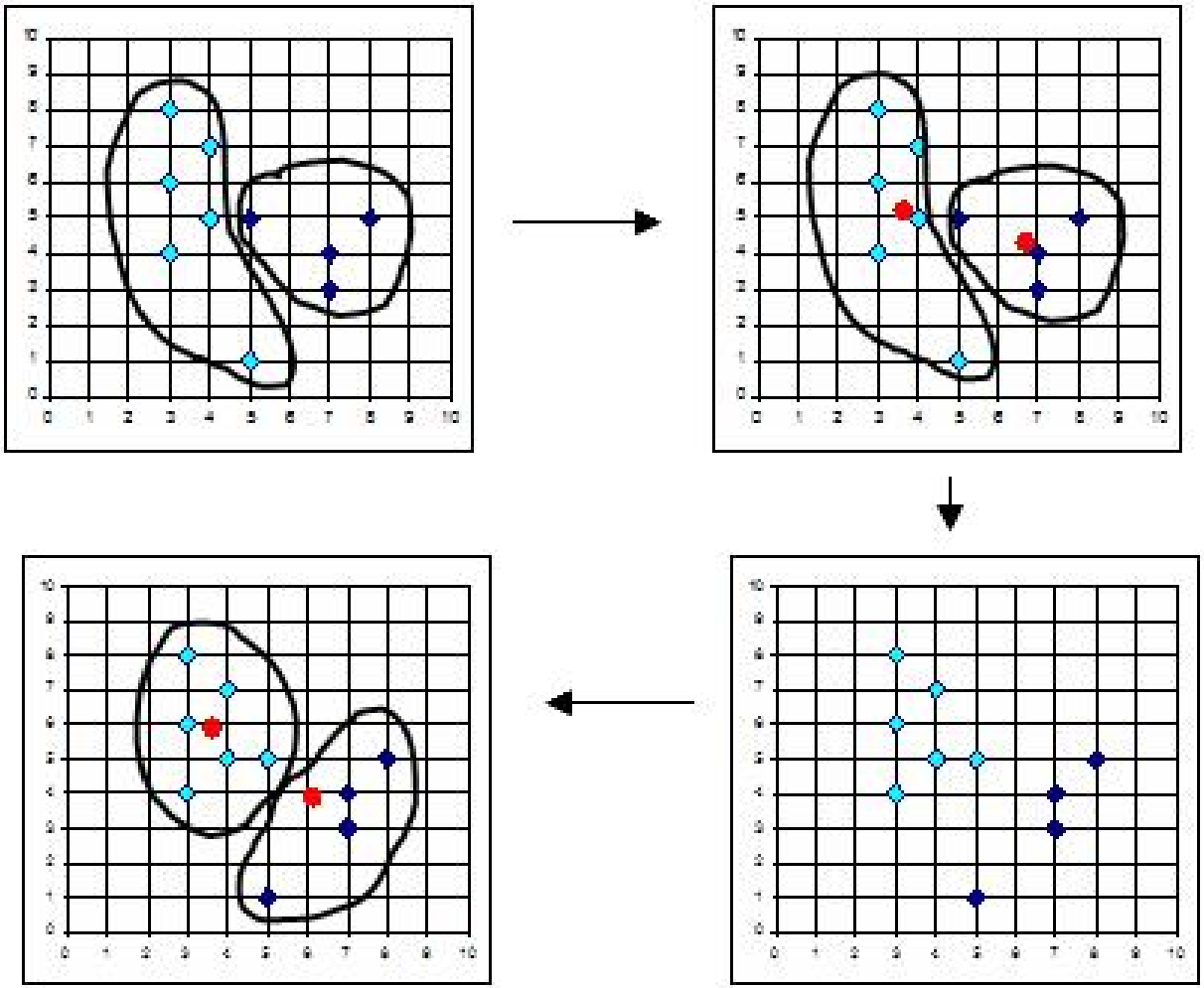
Przykład (cd)

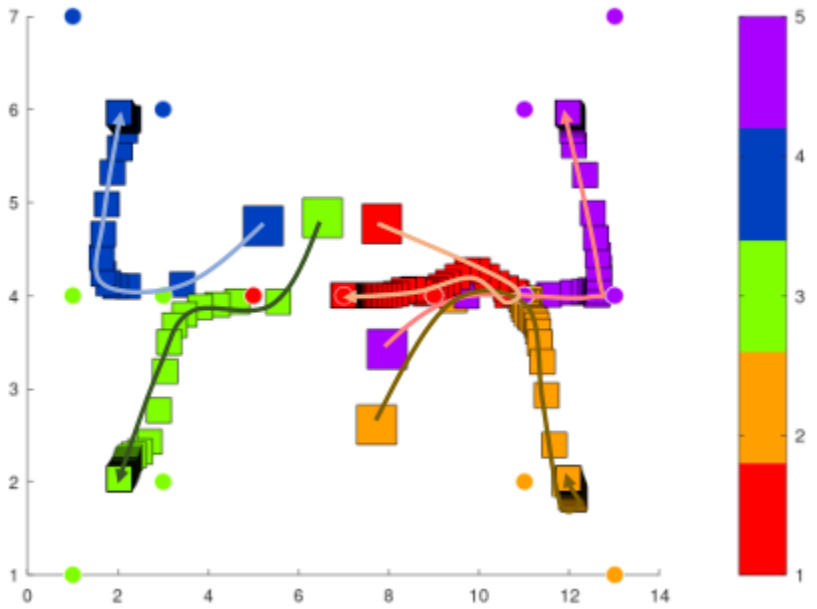
Następnie wyznaczamy nowe środki skupień przy pomocy średniej arytmetycznej. Nowe środki to $(-2,-10/3),(17/3,7/3),(9/2,11/4)$. Kroki powtarzamy tak długo, aż zakończy się przemieszczanie punktów pomiędzy skupieniami. Algorytm kończy się po 3 iteracjach i w rezultacie dostajemy skupienia przedstawione na poniższym rysunku:



Przykład

Kolejne kroki działania algorytmu



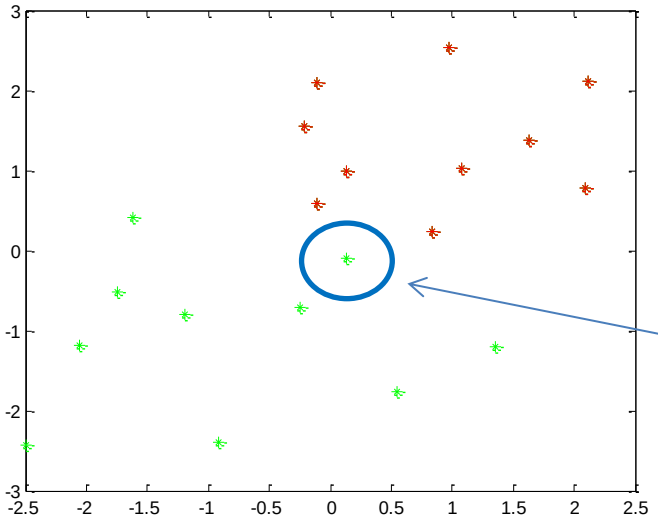


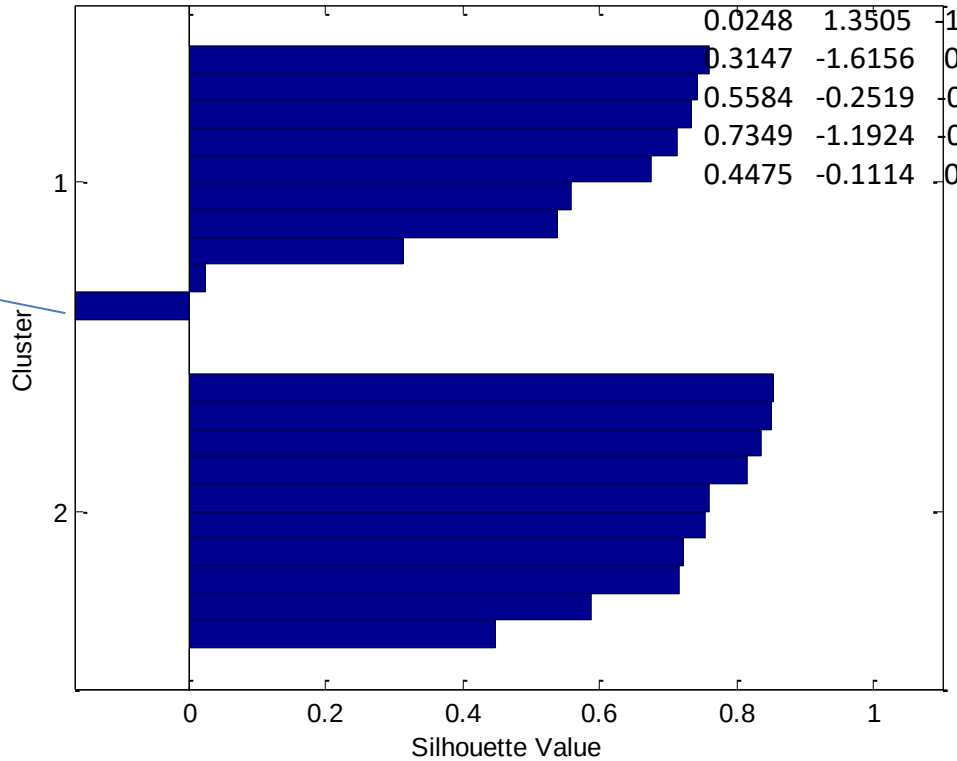
k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	1	1	1	3	3	3	5	7	9	11	11	11	13	13	13
x_2	1	4	7	2	4	6	4	4	4	2	4	6	1	4	7

K-means w Matlabie

```
X = [randn(10,2)+ones(10,2);
randn(10,2)-ones(10,2)];
cidx = kmeans(X,2,'distance','sqeuclid');
s = silhouette(X,cidx,'sqeuclid');
```

s	X1	X2
0.7207	0.1351	0.9932
0.8367	0.9699	2.5326
0.5893	0.8351	0.2303
0.8544	1.6277	1.3714
0.7546	2.0933	0.7744
0.8161	2.1093	2.1174
-0.1664	**0.1363**	**-0.0891**
0.8502	1.0774	1.0326
0.7169	-0.2141	1.5525
0.7606	-0.1135	2.1006
0.5389	0.5442	-1.7648
0.7591	-0.9141	-2.4023
0.7147	-2.4916	-2.4224
0.6744	-1.7423	-0.5118
0.7418	-2.0616	-1.1774
0.0248	1.3505	-1.1961
0.3147	-1.6156	0.4193
0.5584	-0.2519	-0.7084
0.7349	-1.1924	-0.8022
0.4475	-0.1114	-0.5877





Zagadnienie klasteryzacji miękkiej

Dane eksperymentalne

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \dots & & & \\ x_{M1} & x_{M2} & \dots & x_{Mn} \end{bmatrix}$$

Ilość punktów - M

Klastry danych

$$\bigcup_{i=1,c} A_i = \mathbf{X}$$

$$A_i \cap A_j \neq \varnothing, \quad i, j = \overline{1,c}, \quad i \neq j$$

$$\exists A_i = \varnothing, i = \overline{1,c}$$

A_i – klastry

Ilość klastrów – c

Egzemplarz należy do wszystkich klastrów jednocześnie

Mogą być puste klastry

Funkcja przynależności

$$\mathbf{F} = \left[\mu_{kj} \right], \quad \mu_{ki} \in [0,1], \quad k = \overline{1,M}, \quad i = \overline{1,c}$$

$$a) \quad \sum_{i=1,c} \mu_{ki} = 1, \quad k = \overline{1,M}$$

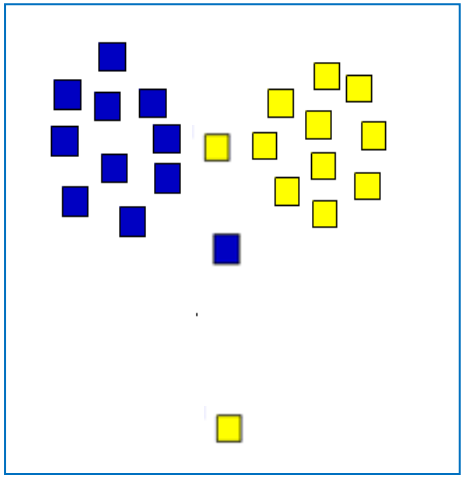
$$b) \quad \sum_{i=1,c} \mu_{ki} \neq 1, \exists i \mu_{ki} > 0, \forall k$$

$$0 \leq \sum_{k=1,M} \mu_{ki} \leq M, \quad i = \overline{1,c}$$

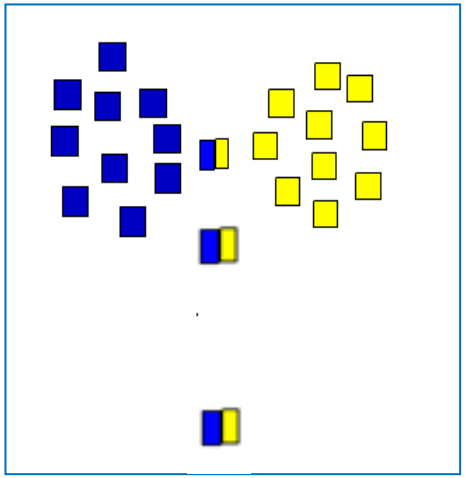
$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \dots & & & \\ x_{M1} & x_{M2} & \dots & x_{Mn} \end{bmatrix} \Rightarrow F = \begin{bmatrix} \mu_{11} & \mu_{12} & \dots & \mu_{1c} \\ \mu_{21} & \mu_{22} & \dots & \mu_{2c} \\ \dots & & & \\ \mu_{M1} & \mu_{M2} & \dots & \mu_{Mc} \end{bmatrix}$$

Uzasadnienie miękkiego podejścia

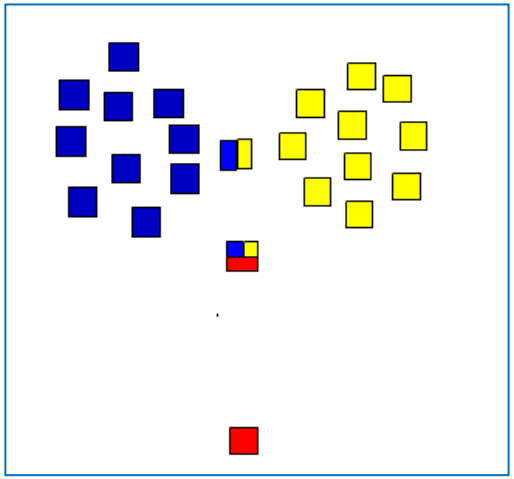
Kmeans



Cmeans, a)



Cmeans, b)



$$\varphi_{ki} \in \{0,1\}$$

$$a) \quad \sum_{i=1,c} \mu_{ki} = 1, \quad k = \overline{1,M}$$

$$b) \quad \sum_{i=1,c} \mu_{ki} \neq 1, \exists i \mu_{ki} > 0, \forall k$$

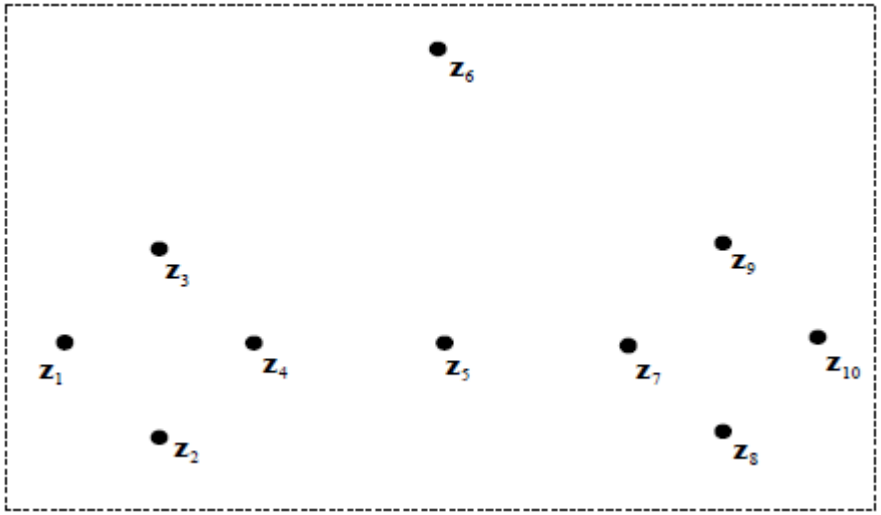
$$0 \leq \sum_{k=1,M} \mu_{ki} \leq M, \quad i = \overline{1,c}$$

Twarda klasteryzacja

‚Twarda' klasyczna klasteryzacja

$$M_{hc} = \left\{ \mathbf{U} \in \mathbb{R}^{c \times N} \middle| \mu_{ik} \in \{0,1\}, \forall i,k; \sum_{i=1}^{c} \mu_{ik} = 1, \forall k; \; 0 < \sum_{k=1}^{N} \mu_{ik} < N, \forall i \right\}$$

$$\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{10}\}$$



$$\mathbf{U} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

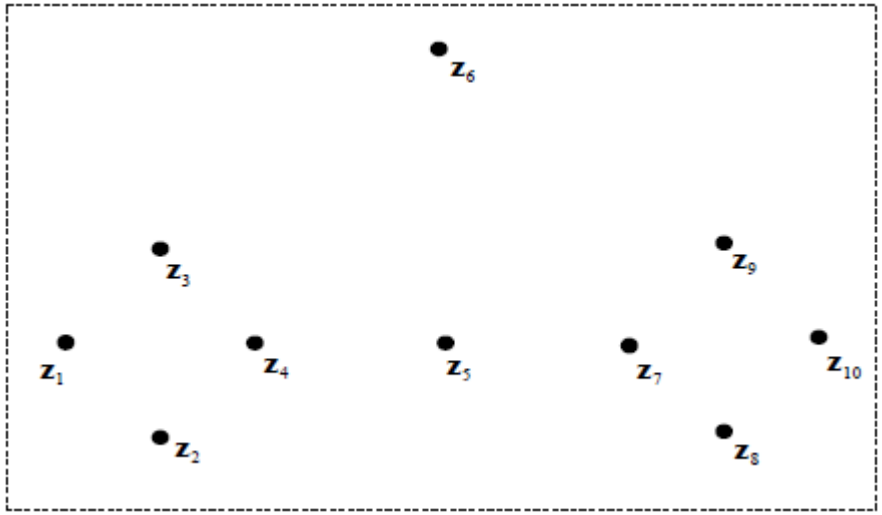
2^{10} possible hard partitions

Twarda klasteryzacja może nie dawać realistycznego obrazu danych bazowych. Graniczne punkty danych mogą reprezentować wzorce z mieszaniną właściwości danych w obydwu klastrach, a zatem nie mogą być w pełni przypisane do żadnej z tych klas, czy też stanowią oddzielną klasę. Tę wadę można złagodzić stosując rozmytej klasteryzacji

Miękka klasteryzacja typu 1

$$M_{fc} = \left\{ \mathbf{U} \in \mathbb{R}^{c \times N} \middle| \mu_{ik} \in [0,1], \forall i,k; \sum_{i=1}^{c} \mu_{ik} = 1, \forall k;\ 0 < \sum_{k=1}^{N} \mu_{ik} < N, \forall i \right\}$$

$$\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{10}\}$$

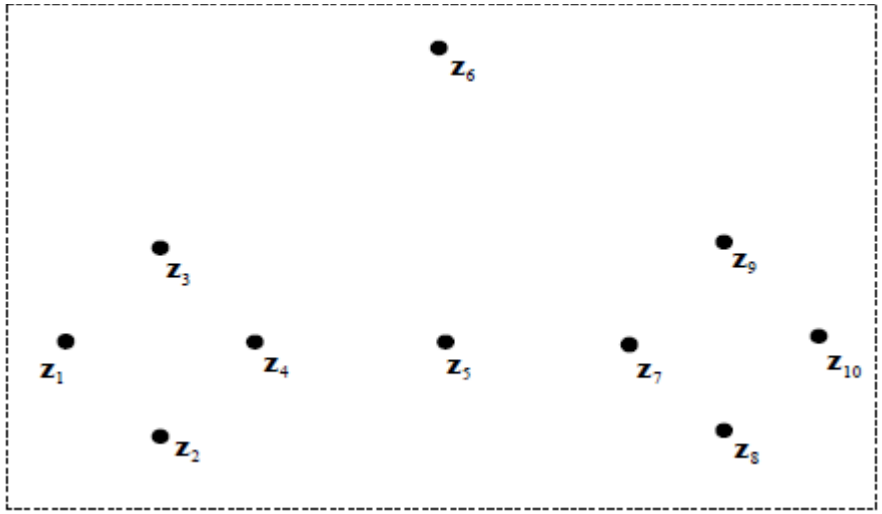


$$\mathbf{U} = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 0.8 & 0.5 & 0.5 & 0.2 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.2 & 0.5 & 0.5 & 0.8 & 1.0 & 1.0 & 1.0 \end{bmatrix}$$

Miękka klasteryzacja typu 2

$$M_{pc} = \left\{ \mathbf{U} \in \mathbb{R}^{c \times N} \middle| \mu_{ik} \in [0,1], \forall i,k;\ \forall k,\ \exists i,\ \mu_{ik} > 0;\ 0 < \sum_{k=1}^{N} \mu_{ik} < N, \forall i \right\}$$

$$\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{10}\}$$



$$\mathbf{U} = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 1.0 & 0.5 & 0.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.2 & 1.0 & 1.0 & 1.0 & 1.0 \end{bmatrix}$$

Jakość miękkiej klasteryzacji

$$J = \sum_{i=1,c} \sum_{X_k \in A_i} \|V_i - X_i\|^2$$

$$V_i = \frac{1}{|A_i|} \sum_{X_k \in A_i} X_k$$

$$A_i = \left\{ X_p : \varphi_{pi} = 1,\ p = \overline{1, M} \right\}$$

$$J = \sum_{i=1,c} \sum_{k=1,M} (\mu_{ki})^m \|V_i - X_k\|^2 \qquad m \in (1, \infty)$$

$$V_i = \frac{\sum_{k=1,M} (\mu_{ki})^m X_k}{\sum_{k=1,M} (\mu_{ki})^m}$$

centrum i-go klastra

$$A_i = \left\{ \mu_{i,k} / X_k \right\}, k = 1, M$$

Zagadnienie optymalizacji

$$J = \sum_{i=1,c} \sum_{X_k \in A_i} \|V_i - X_k\|^2 \to \min$$

$$J = \sum_{i=1,c} \sum_{k=1,M} (\mu_{ki})^m \|V_i - X_k\|^2 \to \min$$

Algorytm C-means

1. Ustalamy C – ilość klastrów, m - waga; ε – kryterium zakończenia

2. Losowa generacja macierzy F $\quad \mathbf{F} = [\mu_{ik}], \quad \mu_{ik} \in [0,1], \quad k = \overline{1,M}, \quad i = \overline{1,c}$

3. Obliczenia centrów klastrów $\quad V_i = \dfrac{\sum_{k=1,M} (\mu_{ik})^m X_k}{\sum_{k=1,M} (\mu_{ik})^m}, \quad i = 1,c$

4. Obliczenia odległości punktów do centrów klastrów

$$D_{ik} = \sqrt{\|X_k - V_i\|^2}, \quad k = \overline{1,M}, \quad i = \overline{1,c}$$

5. Obliczenia jakości klasteryzacji

$$\chi = \sum_{i=1,c} \sum_{k=1,M} (\mu_{ik})^m \|X_k - V_i\|^2$$

Algorytm C-means (cd)

5. Obliczenie nowej macierzy F $D_{ik} = \sqrt{\left\| X_k - V_i \right\|^2}, \quad k = \overline{1, M}, \quad i = \overline{1, c}$

Jeżeli $D_{ik} > 0$ to $\mu_{ik} = \dfrac{1}{D_{ik}^2 \sum_{j=1,c} \left(\dfrac{1}{D_{jk}^2} \right)^{2/(m-1)}}$

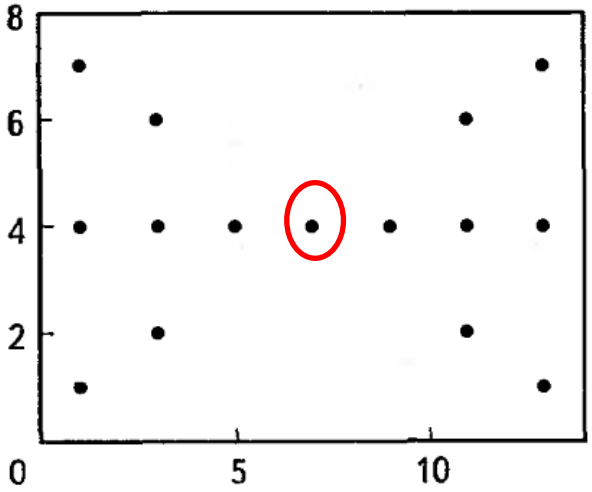
Jeżeli $D_{ik} = 0$ to $\mu_{ik} = 1$

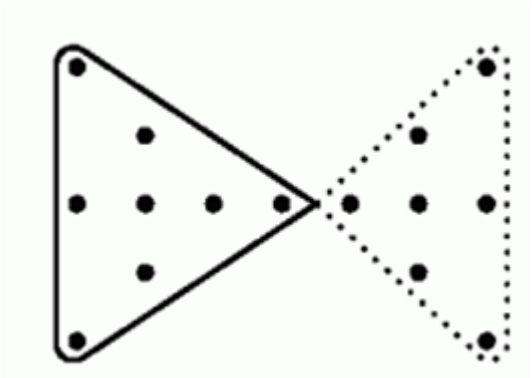
6. Sprawdzanie warunku zakończenia $\left\| F - F^* \right\|^2 < \varepsilon$

cmeansOS.m

Przykład

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	1	1	1	3	3	3	5	7	9	11	11	11	13	13	13
x_2	1	4	7	2	4	6	4	4	4	2	4	6	1	4	7



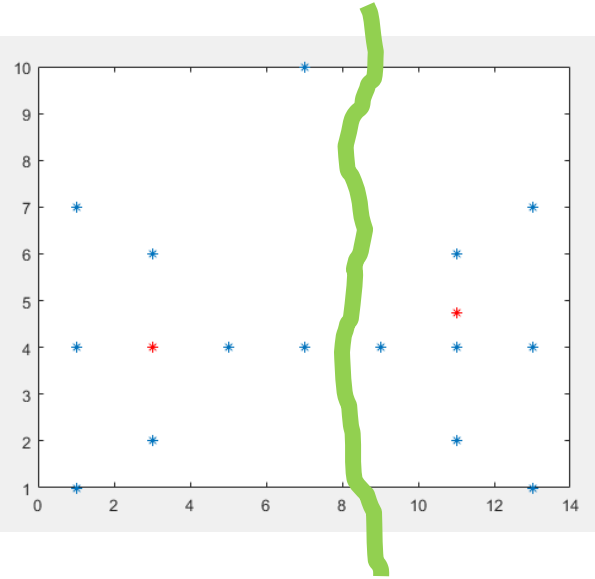


K-means

1	1	1
1	4	1
1	7	1
3	2	1
3	4	1
3	6	1
5	4	1
7	**4**	**1**
9	4	2
11	2	2
11	4	2
11	6	2
13	1	2
13	4	2
13	7	2

klasterSztowbaOS.m

Przykład (cd)



1	1	2
1	4	2
1	7	2
3	2	2
3	4	2
3	6	2
5	4	2
7	**4**	**2**
9	4	1
11	2	1
11	4	1
11	6	1
13	1	1
13	4	1
13	7	1
7	**10**	**2**

Przykład(cd)

C-means

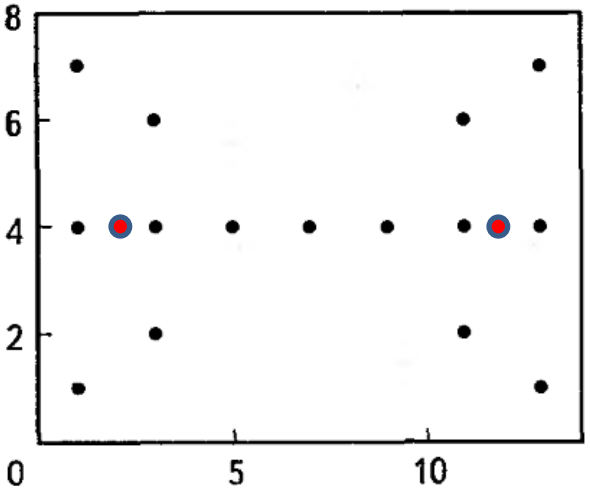
k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
μ_{k1}	0,909	0,976	0,909	0,947	0,998	0,947	0,879	0,500	0,121	0,053	0,002	0,053	0,091	0,024	0,091
μ_{k2}	0,091	0,024	0,091	0,053	0,002	0,053	0,121	0,500	0,879	0,947	0,998	0,947	0,909	0,976	0,909

[centers,U] = fcm(D,2);

C1	C2
2.63442208891946	11.3655618630795
4.0000000128359	3.99999998720547

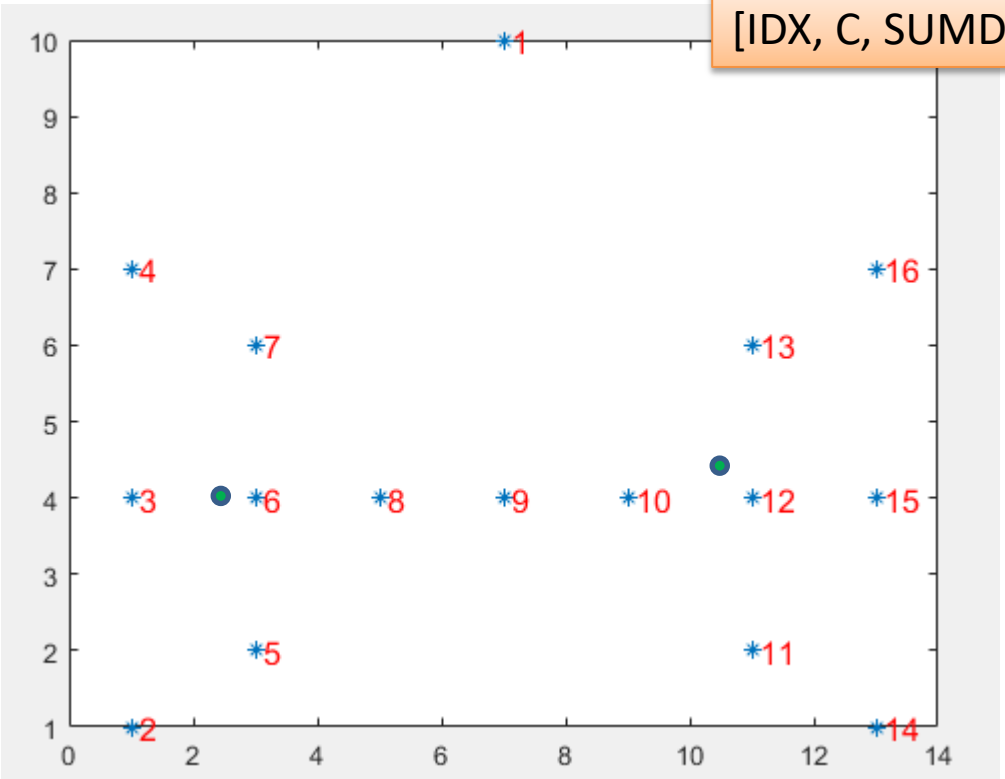
Iteration count = 10, obj. fcn = 82.934805

0.908900397565453	0.0910996024345467
0.975740759639855	0.0242592403601451
0.908900398767374	0.091099601232626
0.947083403643996	0.0529165963560043
0.998093920222566	0.00190607977743397
0.947083404958314	0.052916595041686
0.878655607632406	0.121344392367593
0.499998161980966	**0.500001838019034**
0.121342408173539	0.878657591826461
0.0529162709044848	0.947083729095515
0.00190590545688004	0.99809409454312
0.0529162722150112	0.947083727784989
0.0910997368253619	0.908900263174638
0.0242596319012884	0.975740368098712
0.0910997380241107	0.908900261975889



K-means

[IDX, C, SUMD] = kmeans(z', 2);

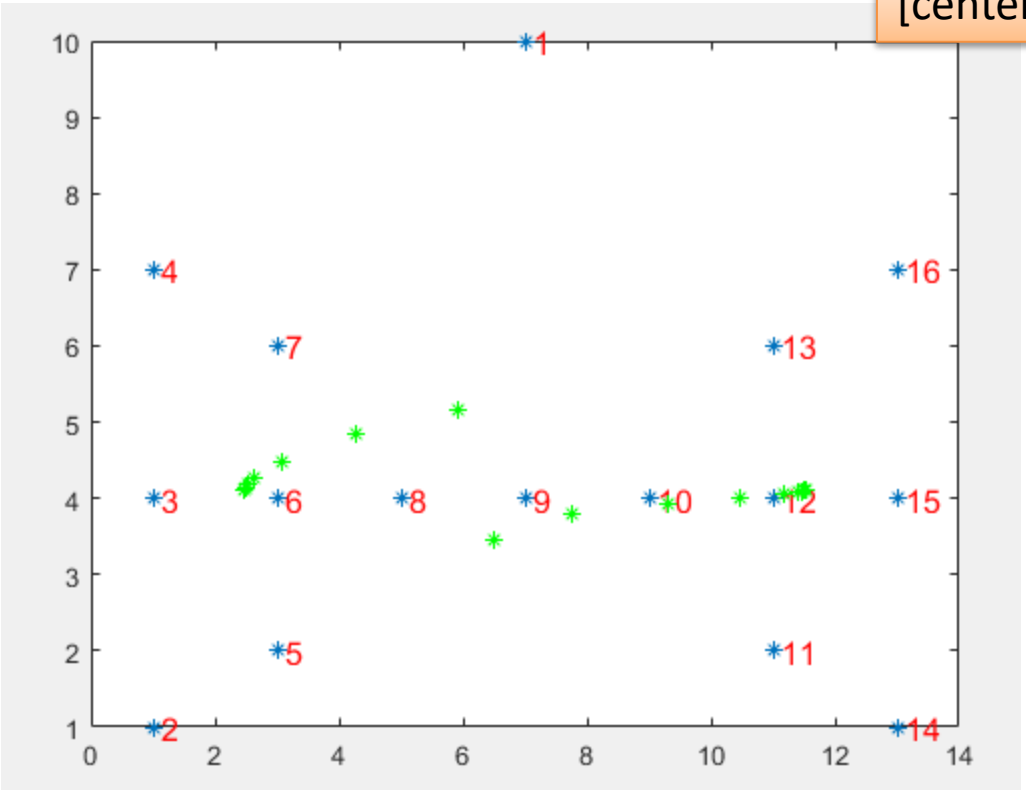


c = 10.5555555555556 4.66666666666667
2.42857142857143 4

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2

Cmeans I

[centers,U] = fcm(z',2)



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0.5000	0.2390	0.1238	0.2295	0.1984	0.0585	0.1834	0.2788	0.5000	0.7212	0.8016	0.9415	0.8166	0.7610	0.8762	0.7705
0.5000	0.7610	0.8762	0.7705	0.8016	0.9415	0.8166	0.7212	0.5000	0.2788	0.1984	0.0585	0.1834	0.2390	0.1238	0.2295

Ostre, rozmyte i posybilistyczne grupowanie

$$Z_H = \left\{ U \in R^{cxM} \right\} \mid \mu_{ik} \in \{0,1\}, \forall i,k;\ \sum_{i=1}^{c} \mu_{ik} = 1,\ \forall k;\ 0 < \sum_{k=1}^{M} \mu_{ik} < M, \forall i$$

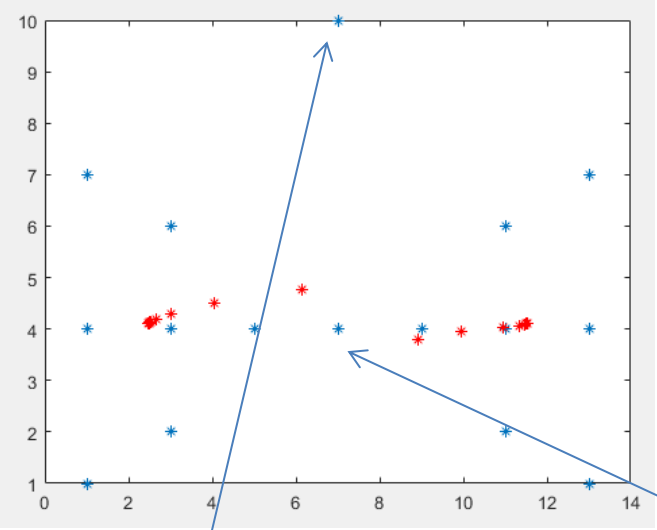
$$Z_F = \left\{ U \in R^{cxM} \right\} \mid \mu_{ik} \in [0,1], \forall i,k;\ \sum_{i=1}^{c} \mu_{ik} = 1,\ \forall k;\ 0 < \sum_{k=1}^{M} \mu_{ik} < M, \forall i$$

$$Z_P = \left\{ U \in R^{cxM} \right\} \mid \mu_{ik} \in [0,1], \forall i,k; \forall k,\ \exists i,\ \mu_{ik} > 0;\ 0 < \sum_{k=1}^{M} \mu_{ik} < M, \forall i$$

Algorytm Cmeans I zakłada że suma stopni przynależności danego obiektu do każdej z grup jest zawsze równa 1. Może to spowodować niepożądane przesunięcia środków w przypadku występowania pojedynczych „przypadkowych” danych (outliers). Rezygnacja z tego ograniczenia prowadzi do Cmeans II (Possibilistic CM) .

Wynik grupowania rozmytego Cmeans I

[centers,U] = fcm(z',2)



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0.5000	0.2390	0.1238	0.2295	0.1984	0.0585	0.1834	0.2788	0.5000	0.7212	0.8016	0.9415	0.8166	0.7610	0.8762	0.77(
2	0.5000	0.7610	0.8762	0.7705	0.8016	0.9415	0.8166	0.7212	0.5000	0.2788	0.1984	0.0585	0.1834	0.2390	0.1238	0.22

Jakość grupowania rozmytego Cmeans I

$$J(X;U,V)=\sum_{i=1}^{c}\sum_{k=1}^{M}(\mu_{ik})^{m}\left\|x_k-v_i\right\|_A^2$$

$$U=[\mu_{ik}]\in Z_F$$

$$m\to 1\Rightarrow FCM\to HCM$$

Jakość grupowania posybilistycznego Cmeans II

$$J(X;U,V)=\sum_{i=1}^{c}\sum_{k=1}^{M}(\mu_{ik})^m\left\|x_k-v_i\right\|_A^2+\sum_{i=1}^{c}\eta_i\sum_{k=1}^{M}(1-\mu_{ik})^m$$

$D_{ik}>0$

1

$$D_{ik}=\sqrt{\left\|X_k-V_i\right\|^2},\quad k=\overline{1,M},\quad i=\overline{1,c}$$

2

$$\mu_{ik}=\frac{1}{D_{ik}^2\sum_{j=1,c}\left(\frac{1}{D_{jk}^2}\right)^{2/(m-1)}}$$

W razie $D_{ik}=0$ $\mu_{ik}=1$

3

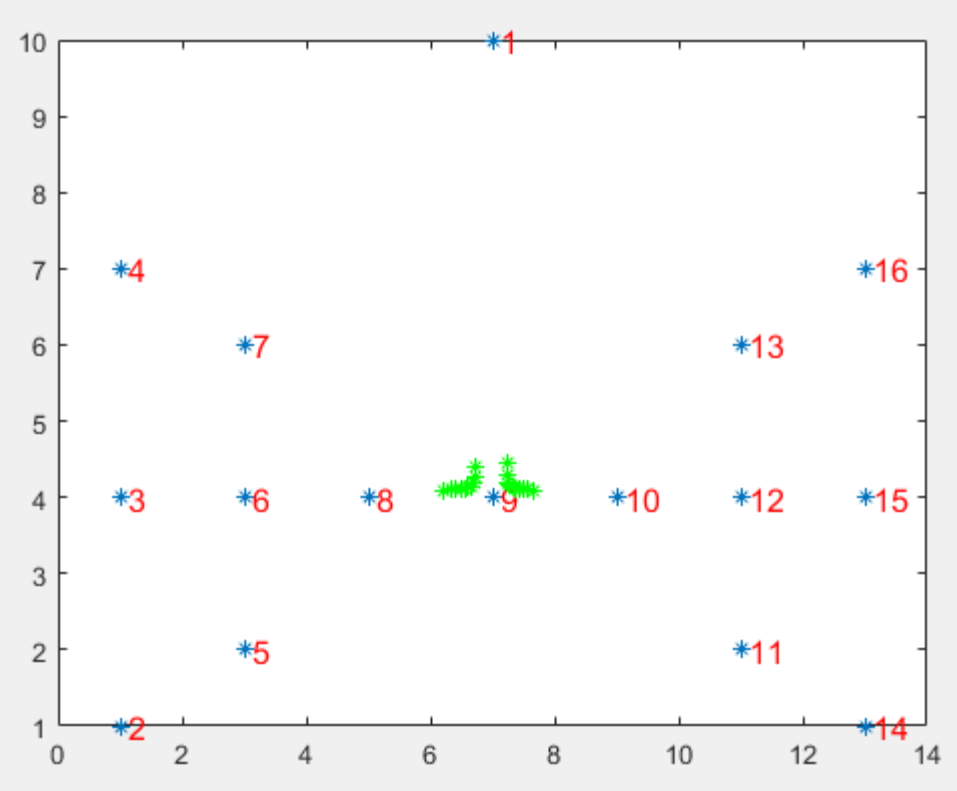
$$\eta_i=K\frac{\sum_{k=1}^{M}\mu_{ik}^m D_{ik}^2}{\sum_{k=1}^{M}\mu_{ik}^m},1\le i\le c$$

4

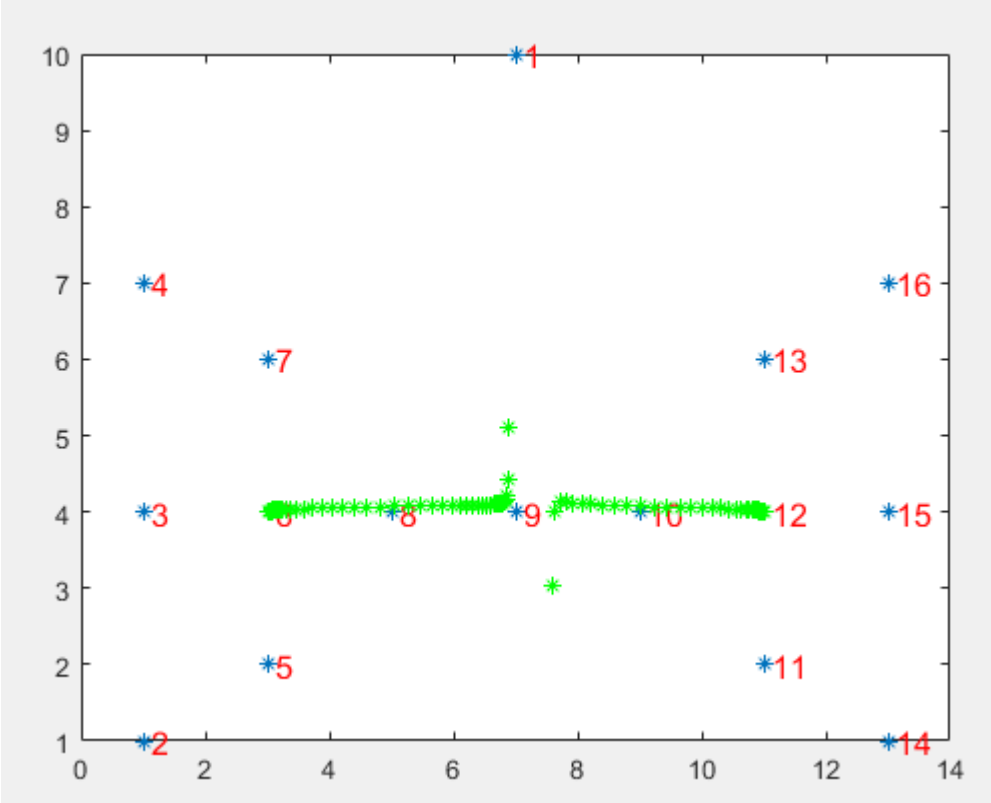
$$t_{ik}=\frac{1}{1+\left(\frac{D_{ik}^2}{\eta_i}\right)^{1/(m-1)}}1\le i\le c;1\le k\le M$$

5

$$\mathbf{v}_i=\frac{\sum_{k=1}^{M}t_{ik}^m\mathbf{x}_k}{\sum_{k=1}^{M}t_{ik}^m},\quad 1\le i\le c;1\le k\le M$$



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0.1961	0.1378	0.1628	0.1404	0.2484	0.2844	0.2540	0.5497	0.9517	0.8254	0.3558	0.4345	0.3674	0.1840	0.2314	0.1887
0.1939	0.1879	0.2381	0.1927	0.3659	0.4505	0.3782	0.8498	0.9332	0.5251	0.2388	0.2721	0.2440	0.1333	0.1567	0.1357

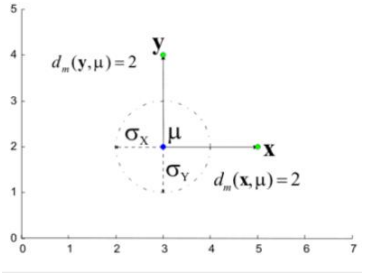

10
9
8
7
6
5
4
3
2
1
0
2
4
6
8
10
12
14
1
4
16
7
13
3
9
12
15
5
11
2
14

Odległość Mahalanobisa

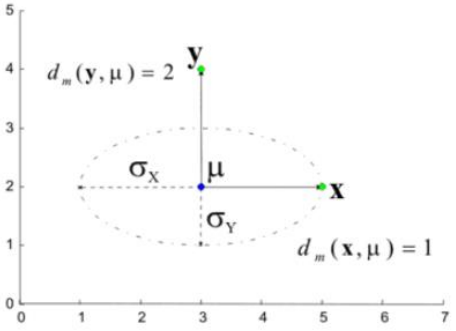
$$D^2 = \|X - V\|^2 \quad \Rightarrow \quad \|X - V\|_B^2 = (X - V)\,B\,(V - X)^T$$

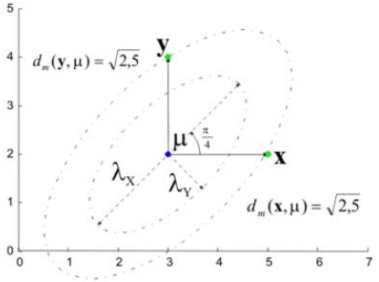
$$B = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

Odległość Euklidesowa



$$B = \begin{bmatrix} \omega_1 & 0 & \dots & 0 \\ 0 & \omega_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \omega_n \end{bmatrix}$$

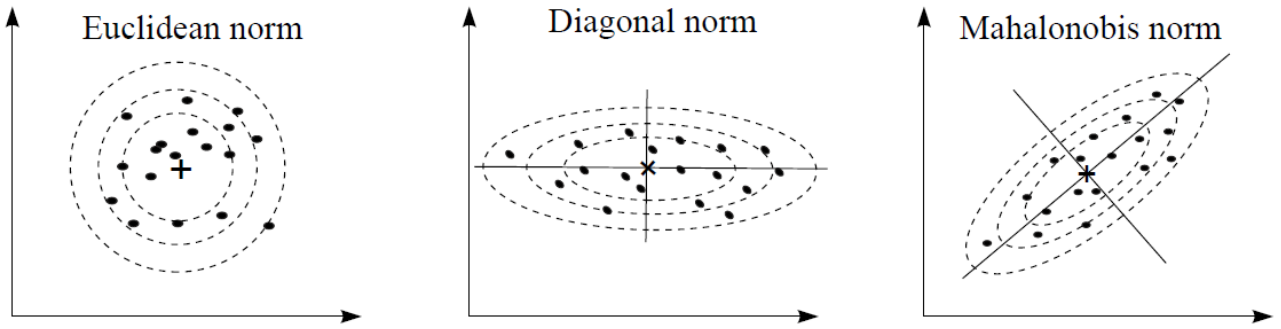




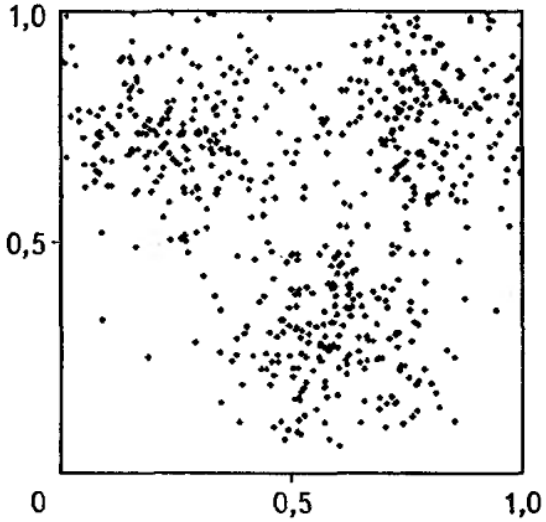
Długości półosi hiper elipsoidy są określone przez odwrotności pierwiastków kwadratowych wartości własnych macierzy kowariancji C, utworzonej przez punkty pomiarowe

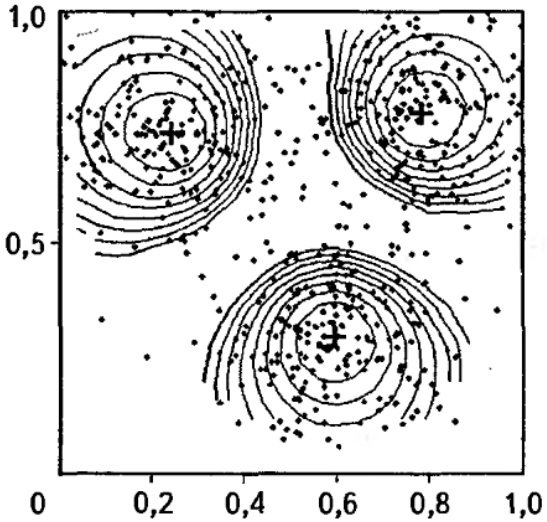
Normy

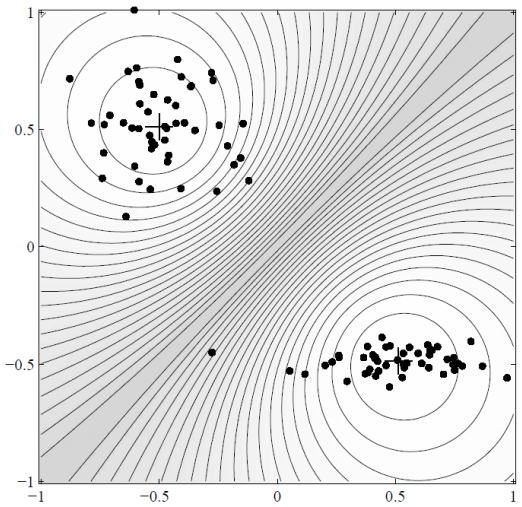
$$\mathbf{A} = \begin{bmatrix} (1/\sigma_1)^2 & 0 & \cdots & 0 \\ 0 & (1/\sigma_2)^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (1/\sigma_n)^2 \end{bmatrix}$$



Przykłady







Algorytm Gustafsona–Kessela

Każdy klaster ma swoją macierz odległości dla adaptacji topologicznej struktury danych

$$D^2_{ik\mathbf{A}_i} = (\mathbf{z}_k - \mathbf{v}_i)^T \mathbf{A}_i (\mathbf{z}_k - \mathbf{v}_i)$$

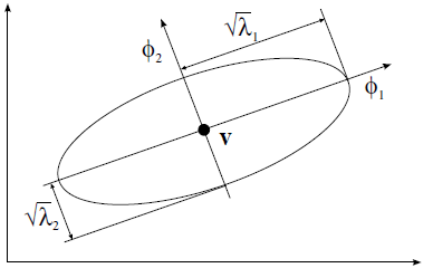
$$\mathbf{Z} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1N} \\ z_{21} & z_{22} & \cdots & z_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ z_{n1} & z_{n2} & \cdots & z_{nN} \end{bmatrix}$$

$$J(\mathbf{Z}; \mathbf{U}, \mathbf{V}, \{\mathbf{A}_i\}) = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ik})^m D^2_{ik\mathbf{A}_i}$$

$$\mathbf{A}_i = [\rho_i \det(\mathbf{F}_i)]^{1/n} \mathbf{F}_i^{-1}$$

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{z}_k - \mathbf{v}_i)(\mathbf{z}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m}$$

Macierz kowariancji klastra
(informacja o kształcie i orientacji klastra)



Algorytm GK

Repeat for $l = 1, 2, \ldots$

Step 1: Compute cluster prototypes (means):

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m \mathbf{z}_k}{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m}, \quad 1 \le i \le c\,.$$

Step 2: Compute the cluster covariance matrices:

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m (\mathbf{z}_k - \mathbf{v}_i^{(l)})(\mathbf{z}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m}, \quad 1 \le i \le c\,.$$

Step 3: Compute the distances:

$$D_{ik\mathbf{A}_i}^2 = (\mathbf{z}_k - \mathbf{v}_i^{(l)})^T \left[\rho_i \det(\mathbf{F}_i)^{1/n} \mathbf{F}_i^{-1}\right] (\mathbf{z}_k - \mathbf{v}_i^{(l)}),$$
$$1 \le i \le c, \quad 1 \le k \le N\,.$$

Step 4: Update the partition matrix:
for $1 \le k \le N$
if $D_{ik\mathbf{A}_i} > 0$ for all $i = 1, 2, \ldots, c$

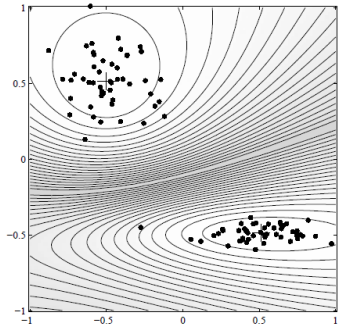
$$\mu_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{c} (D_{ik\mathbf{A}_i} / D_{jk\mathbf{A}_i})^{2/(m-1)}},$$

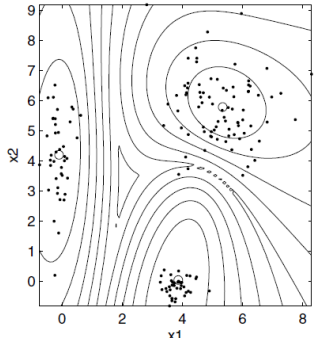
otherwise

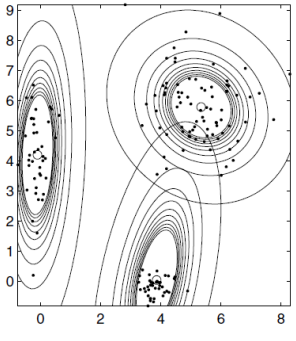
$$\mu_{ik}^{(l)} = 0 \text{ if } D_{ik\mathbf{A}_i} > 0, \text{ and } \mu_{ik}^{(l)} \in [0, 1] \text{ with } \sum_{i=1}^{c} \mu_{ik}^{(l)} = 1\,.$$

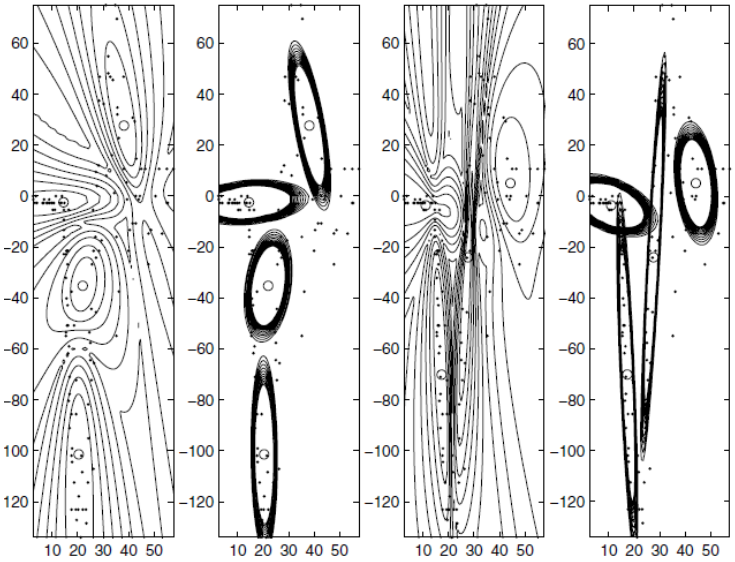
until $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \epsilon$.

Przykłady









Górny algorytm Jagera

$D(x_i, x_j)$ Odległość między punktami

$Z_h = x_k$ Potencjalne klastry

Potencjał klastra $P_1(Z_h) = \sum_{k=1}^{M} \exp(-\alpha D(Z_h, x_k))$ $\alpha > 0$

Szukamy punkt z największym potencjałem V_1

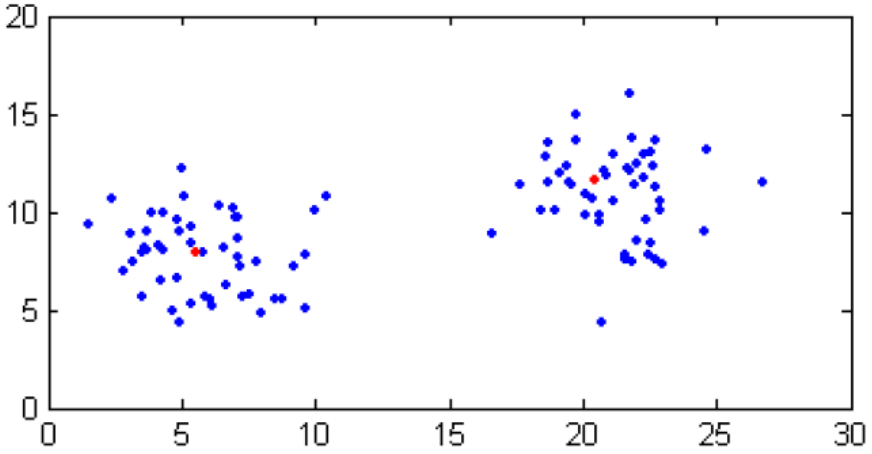
Eliminujemy wpływ tego punktu

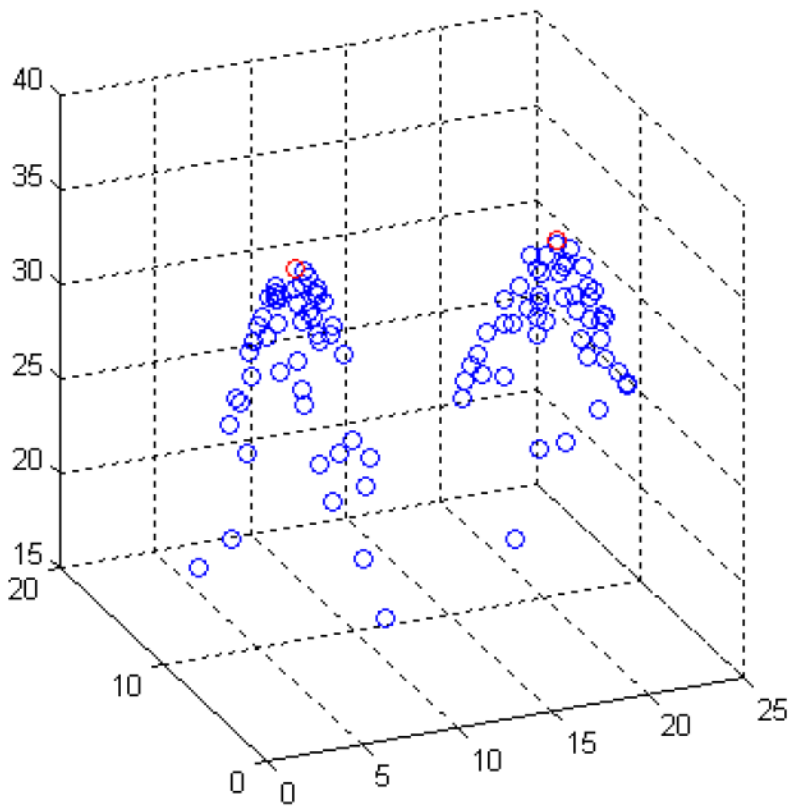
$$P_2(Z_h) = P_1(Z_h) - P_1(V_1) \cdot \exp(-\beta \cdot D(Z_h, V_1))$$

Szukamy punkt z największym potencjałem V_2

$$P_3(Z_h) = P_2(Z_h) - P_2(V_2) \cdot \exp(-\beta \cdot D(Z_h, V_2))$$

Przykład





Potencjały punktów i centrów klastrów

$$P_1(Z_h) = \sum_{k=1}^{M} \exp(-\alpha D(Z_h, x_k))$$

Funkcje Fuzzy Logic Toolbox

Fuzzy Logic Toolbox zawiera funkcje **fcm** i **subclust**, które realizują algorytmy klasteryzacji odpowiednio metodą c-środków (c-means clustering) i klasteryzacji różnicowej (subtractive clustering).
Funkcja **genfis1** generuje modele rozmyte typu Sugeno na podstawie siatki podziału (grid partition) – bez klasteryzacji. Wartości współczynników wielomianu w konkluzjach reguł mają wartości zerowe, a liczba reguł jest duża.
Fuzzy Logic Toolbox zawiera też dwie funkcje (**genfis2** i **genfis3**), które generują struktury reprezentujące układy rozmyte typu Sugeno. Liczba i parametry funkcji przynależności w przesłankach są określane poprzez algorytmy odpowiednio: klasteryzacji różnicowej i klasteryzacji metodą c-środków. Wartości współczynników wielomianu w konkluzjach są wyznaczane metodą najmniejszych kwadratów.
Wygenerowane za pomocą funkcji genfis1, genfis2 i genfis3 modele rozmyte są zazwyczaj stosowane jako modele początkowe (wstępne) dla funkcji lub interfejsu ANFIS.

ANALIZA SKUPIEŃ W MatLabie

Findcluster Toobox

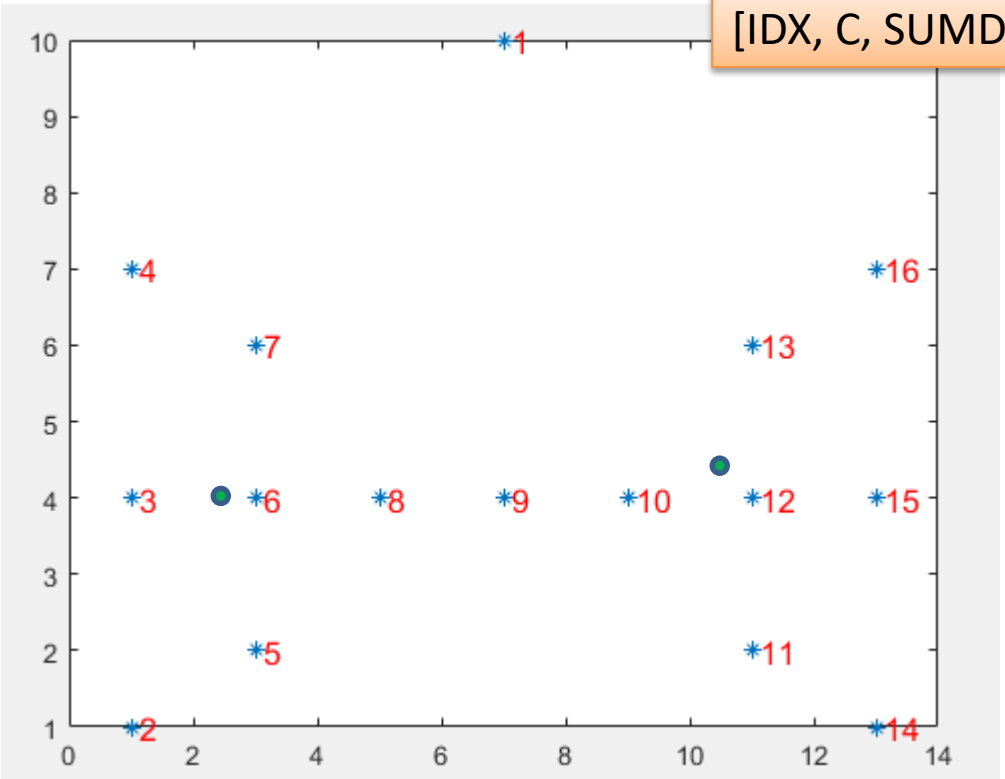
findcluster(,testOS.dat')

testOS.dat

1	1 1
2	1 4
3	1 7
4	3 2
5	3 4
6	3 6
7	5 4
8	7 4
9	9 4
10	11 2
11	11 4
12	11 6
13	13 1
14	13 4
15	13 7

K-means

[IDX, C, SUMD] = kmeans(z', 2);

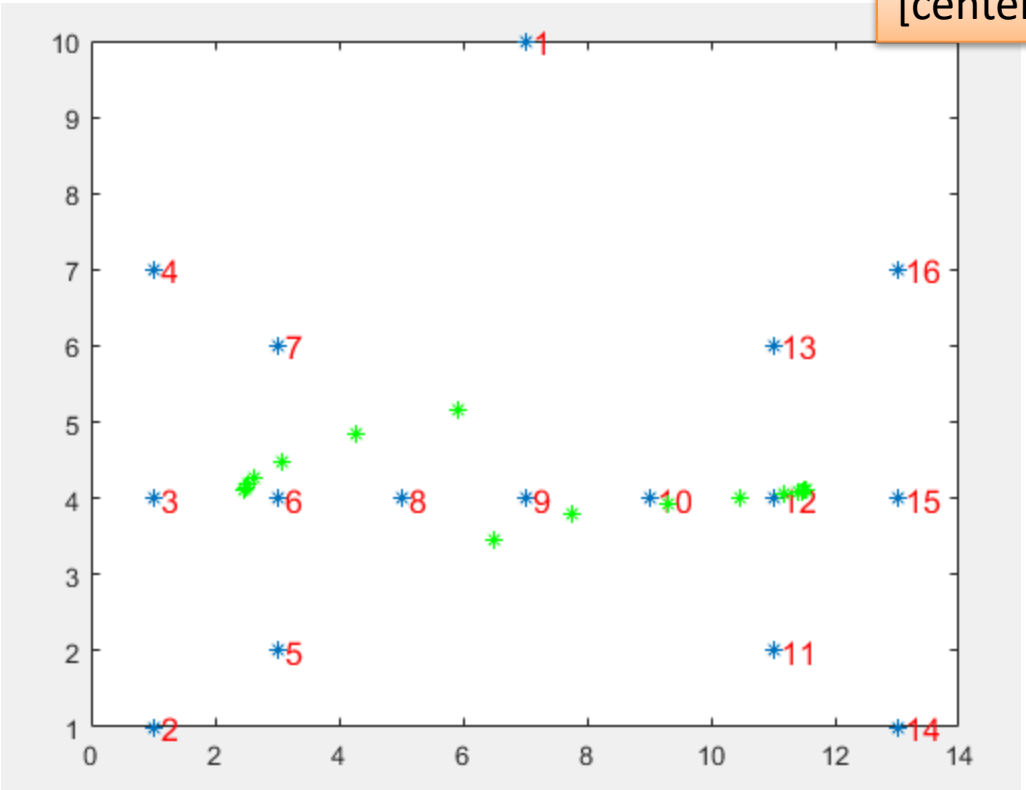


c = 10.5555555555556 4.66666666666667
2.42857142857143 4

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2

Cmeans I

[centers,U] = fcm(z',2)



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0.5000	0.2390	0.1238	0.2295	0.1984	0.0585	0.1834	0.2788	0.5000	0.7212	0.8016	0.9415	0.8166	0.7610	0.8762	0.7705
0.5000	0.7610	0.8762	0.7705	0.8016	0.9415	0.8166	0.7212	0.5000	0.2788	0.1984	0.0585	0.1834	0.2390	0.1238	0.2295

Funkcia FindCluster

Description

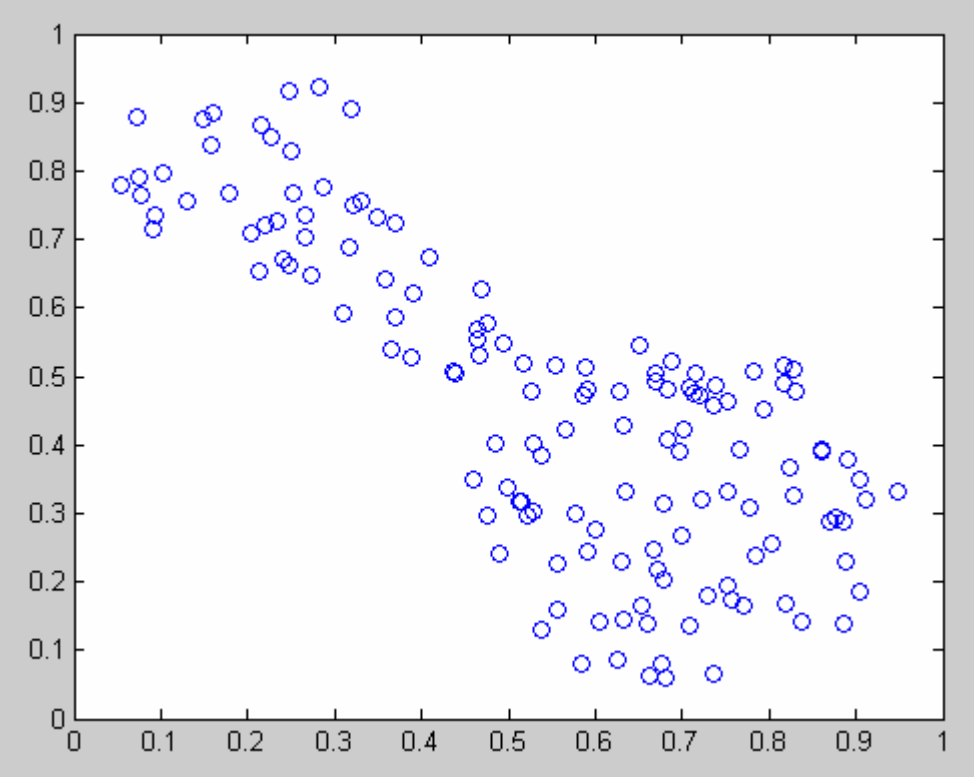
findcluster opens a GUI to implement either the **fuzzy c-means** (fcm), the **fuzzy subtractive clustering** (subtractiv) using the pull-down tab under Method on the GUI, or both. Data is entered using the Load Data button. The options for each of these methods are set to default values. These default values can be changed. See fcm reference page for a description of the options for fuzzy c-means. The subclust reference page provides a description of the options for fuzzy subclustering.

This tool works on multidimensional data sets, but only displays two of those dimensions. Use the pull-down tabs under X-axis and Y-axis to select which data dimension you want to view. For example, if you have data that is five-dimensional, this tool labels the data as data_1, data_2, data_3, data_4, data_5, in the order in which the data appears in the data set. Start to perform the clustering, and Save Center to save the cluster center.

When operating on a data set called file.dat, findcluster (file.dat) loads the data set automatically, plotting up to the first two dimensions of the data only. You can still choose which two dimensions of the data you want to cluster after the GUI appears.

Przykład

```
load fcmdata.dat
plot(fcmdata(:,1),fcmdata(:,2),'o')
```




```
Iteration count = 1, obj. fcn = 8.794048
Iteration count = 2, obj. fcn = 6.986628
.....
Iteration count = 12, obj. fcn = 3.797430
```

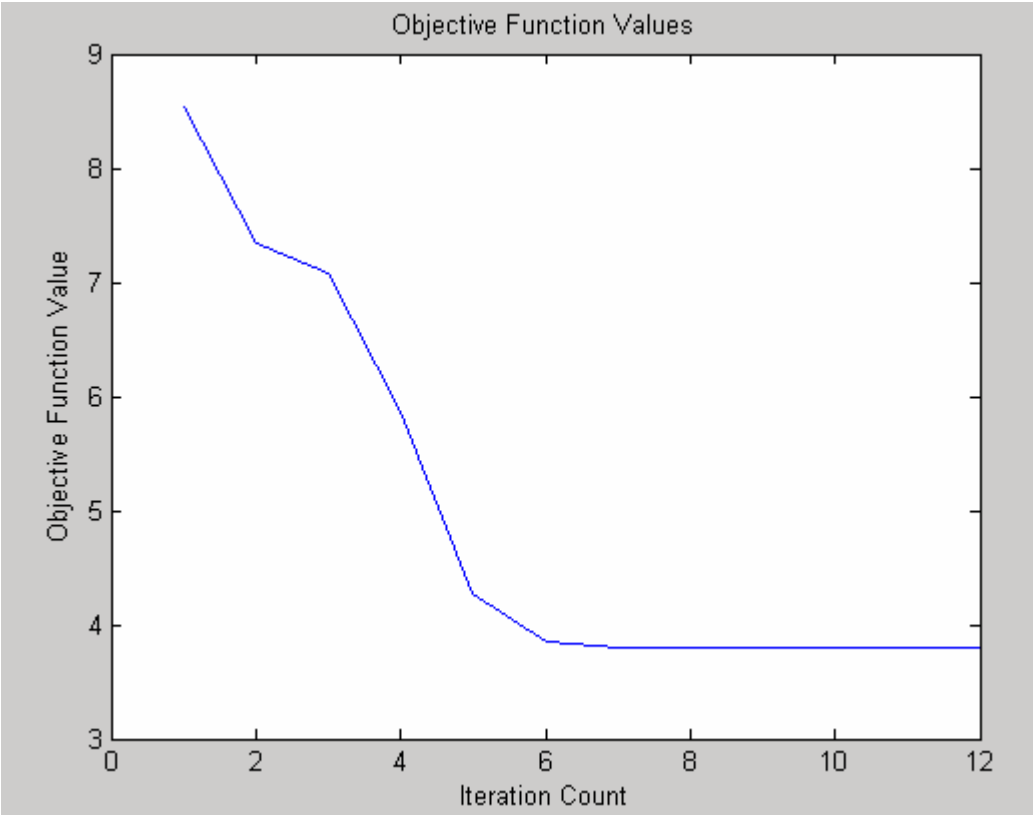
```
[center,U,objFcn] = fcm(fcmdata,2);
```

```
[c6,U,obj_fcn] = fcm(data,2,[2,1,1e-5,6])
```

← Ilość iteracji

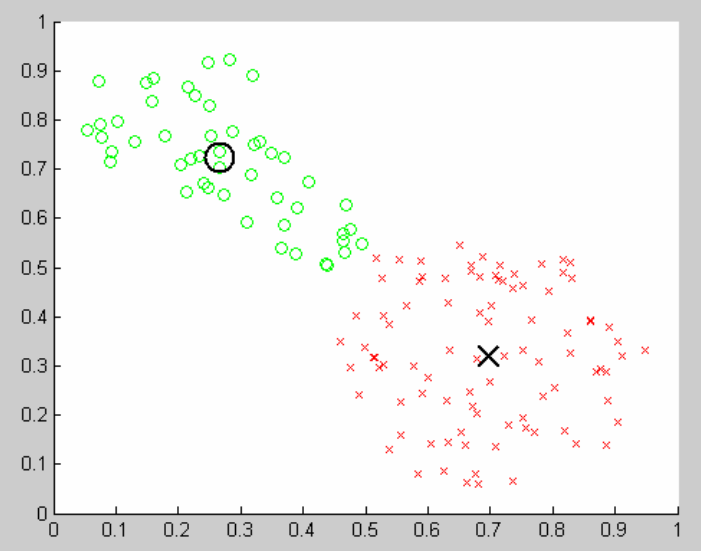
Iteracje

```
figure
plot(objFcn)
title('Objective Function Values')
xlabel('Iteration Count')
ylabel('Objective Function Value')
```



Wizualizacja

```
maxU = max(U);
index1 = find(U(1, :) == maxU);
index2 = find(U(2, :) == maxU);
figure
line(fcmdata(index1, 1), fcmdata(index1, 2), 'linestyle‘,'none','marker', 'o','color','g');
line(fcmdata(index2,1),fcmdata(index2,2),'linestyle‘,'none','marker', 'x','color','r');
hold on
plot(center(1,1),center(1,2),'ko','markersize',15,'LineWidth',2)
plot(center(2,1),center(2,2),'kx','markersize',15,'LineWidth',2)
```



Dynamika centrów

[**c1**,U,obj_fcn] = fcm(data,2,[2,1,1e-5,**1**])

[**c2**,U,obj_fcn] = fcm(data,2,[2,1,1e-5,**2**])

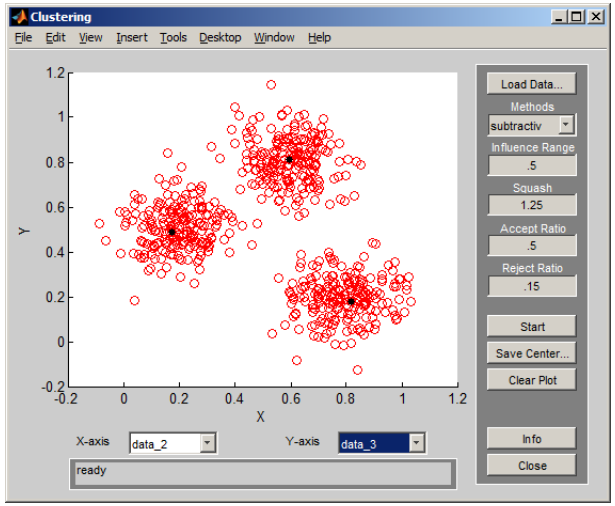
[**c3**,U,obj_fcn] = fcm(data,2,[2,1,1e-5,**3**])

findcluster('clusterdemo.dat')

Subtractive Clustering

If you do not have a clear idea how many clusters there should be for a given set of data, subtractive clustering is a fast, one-pass algorithm for estimating the number of clusters and the cluster centers for a set of data [2]. The cluster estimates, which are obtained from the **subclust** function, can be used to initialize **iterative optimization-based clustering methods** (fcm) and **model identification methods** (like anfis). The **subclust** function finds the clusters using the subtractive clustering method.

To generate a Sugeno-type fuzzy inference system that models the behavior of input/output data, you can configure the genfis command to use subtractive clustering.



Specify the following clustering options:

- Squash factor of `2.0` - Only find clusters that are far from each other.
- Accept ratio `0.8` - Only accept data points with a strong potential for being cluster centers.
- Reject ratio of `0.7` - Reject data points if they do not have a strong potential for being cluster centers.
- Verbosity flag of `0` - Do not print progress information to the command window.

Funkcja Subclust

Find cluster centers with subtractive clustering

Syntax

[C,S] = subclust(X,radii,xBounds,options)

Description

[C,S] = subclust(X,radii,xBounds,options) estimates the cluster centers in a set of data by using the subtractive clustering method.

The function returns the cluster centers in the matrix C. Each row of C contains the position of a cluster center. The returned S vector contains the sigma values that specify the range of influence of a cluster center in each of the data dimensions. All cluster centers share the same set of sigma values.

The subtractive clustering method assumes each data point is a potential cluster center and calculates a measure of the likelihood that each data point would define the cluster center, based on the density of surrounding data points. The algorithm does the following:

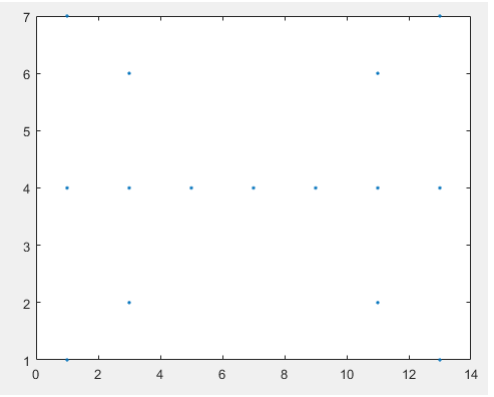
Selects the data point with the highest potential to be the first cluster center

Removes all data points in the vicinity of the first cluster center (as determined by radii), in order to determine the next data cluster and its center location

Iterates on this process until all of the data is within radii of a cluster center

Przykład

```
z =
    1    1    1    3    3    3    5    7    9   11   11   11   13   13   13
    1    4    7    2    4    6    4    4    4    2    4    6    1    4    7
```

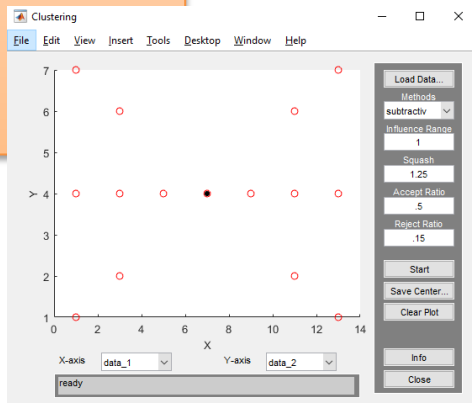


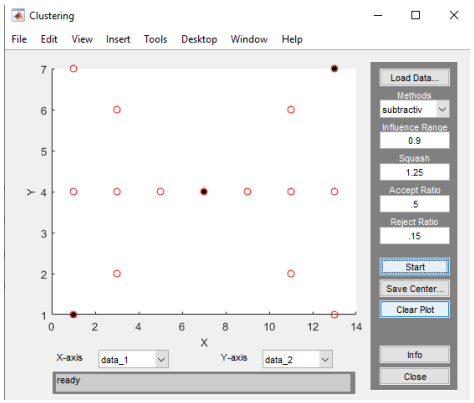

```
>>
c=subclust(z',0.5)

c =

    5    4
   11    4
    1    1
    1    7
   13    7
   13    1
```

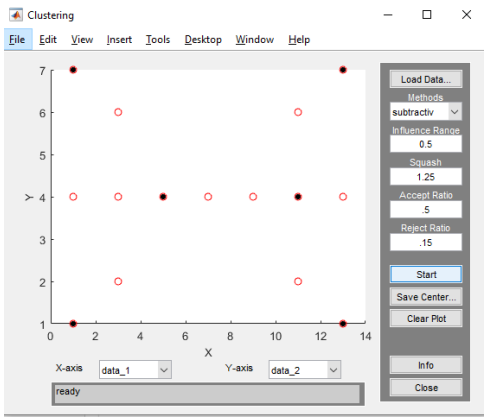
```
c=subclust(z',1)
c =   7    4
```

```
c=subclust(z',0.9)
c =
     7    4
     1    1
    13    7
```

[centers,sigma] = subclust(data,clusterInfluenceRange)

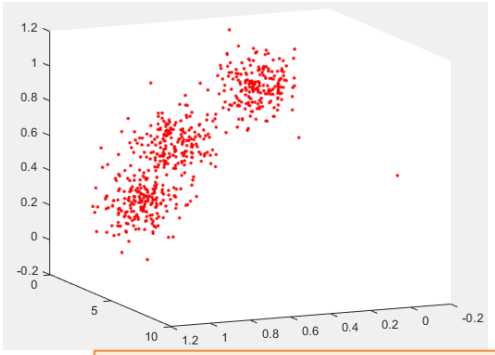


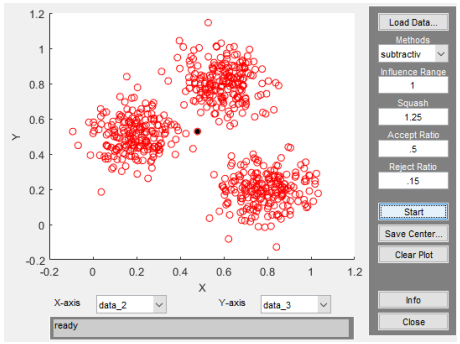


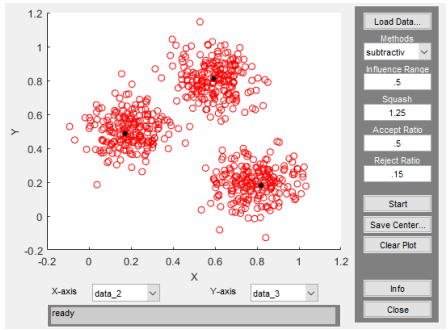


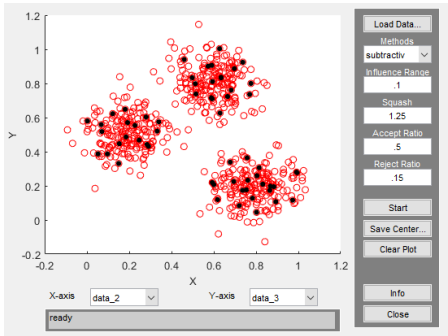
Przykład 2

```
plot3(clusterdemo(:,1),clusterdemo(:,2),clusterdemo(:,3),'.r')
```










```
>> [c s]=subclust(clusterdemo,1)
c =          0.559247          0.328423          0.513069
s =    0.380726443534101    2.86100706968936    0.45015266218473
```

```
>> [c s]=subclust(clusterdemo,0.5)

c =
          0.643778          0.276279          0.401561
          0.198491           0.53765          0.800492
          0.879269          0.979664          0.096464
s =
     0.190363221767051    1.43050353484468    0.225076331092365
```

Funkcja genfis1

Funkcja **genfis1** generuje modele rozmyte typu Sugeno na podstawie siatki podziału (grid partition) – bez klasteryzacji. **Wartości współczynników wielomianu w konkluzjach reguł mają wartości zerowe**, a liczba reguł jest duża.

Funkcja **genfis1**: - tworzy FIS typu Takagi Sugeno, na podstawie tzn "grid partition"
fismat = genfis1(data,numMFs,inmftype,outmftype)
gdzie:
fismat – wynikowa struktura FIS
data – zbiór danych na podstawie którego struktura będzie tworzona
numMFs – liczba funkcji przynależności generowana dla każdego z osobna lub wszystkich wejść
inmftype – typ funkcji przynależności (najlepiej trójkątna/Gaussowska)
outmftype – typ wyjściowej funkcji – uwaga – w FIS typu Takagi-Sugeno wyjściem nie jest zbiór rozmyty a tylko funkcja, dlatego też możliwe opcje to linear lub constant

Inicjalizacja - Funkcja genfis1

>> **help genfis1**

GENFIS1 Generates an initial Sugeno-type FIS for ANFIS training using a grid partition.

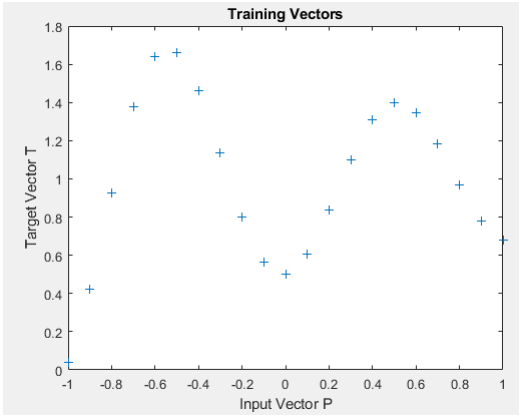
FIS = GENFIS1(DATA) **generates a single-output Sugeno-type fuzzy inference system (FIS)** using a grid partition on the data (no clustering). FIS is used to provide **initial conditions for ANFIS training**. **DATA is a matrix with N+1 columns where the first N columns contain data for each FIS input, and the last column contains the output data.** By default, GENFIS1 uses two 'gbellmf' type membership functions for each input. Each rule generated by GENFIS1 has one output membership function, which is of type 'linear' by default.

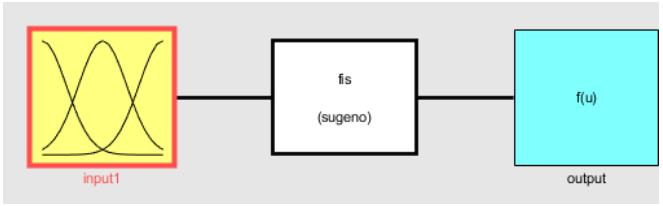
FIS = GENFIS1(DATA, NUMMFS, INPUTMF, OUTPUTMF) explicitly specifies:

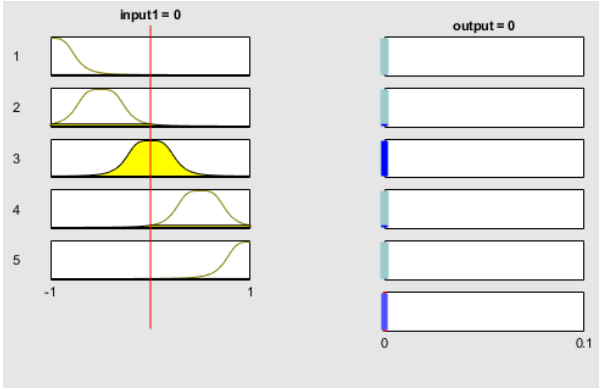
* NUMMFS number of membership functions per input. A scalar value, specifies the same number for all inputs and a vector value specifies the number for each input individually.
* INPUTMF type of membership function for each input. A single string specifies the same type for all inputs, a string array specifies the type for each input individually.
* OUTPUTMF output membership function type, either 'linear' or 'constant'

Przykład

```
X = -1:.1:1;
T = [-.9602 -.5770 -.0729 .3771 .6405 .6600
.4609 ...
.1336 -.2013 -.4344 -.5000 -.3930 -.1647 .0988
...
.3072 .3960 .3449 .1816 -.0312 -.2189 -
.3201]+1;
data=[X' T'];
plot(X,T,'+');
title('Training Vectors');
xlabel('Input Vector P');
ylabel('Target Vector T');
F=genfis1(data,5);
fuzzy(F);
```

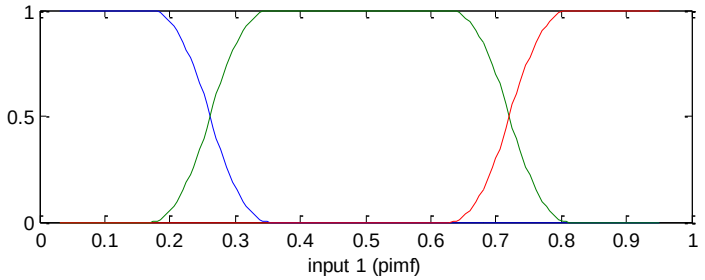


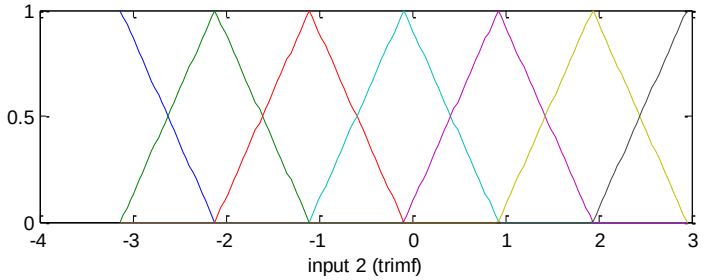


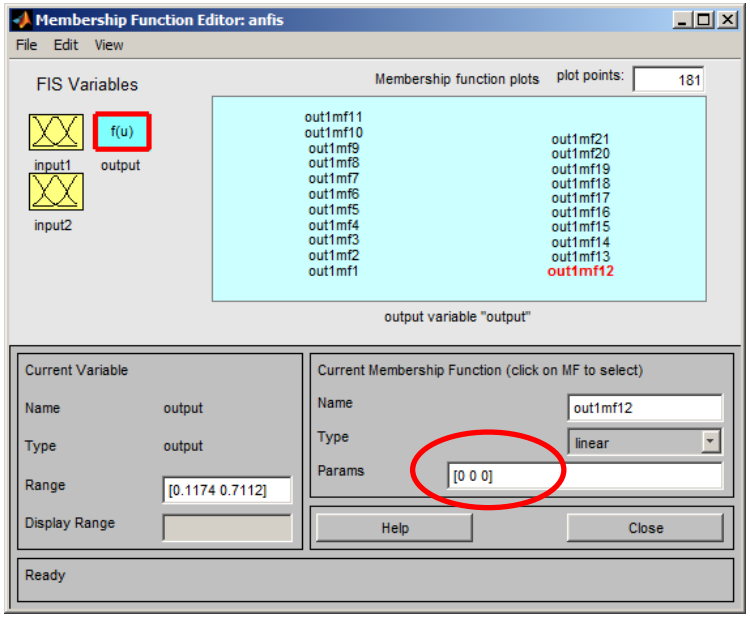


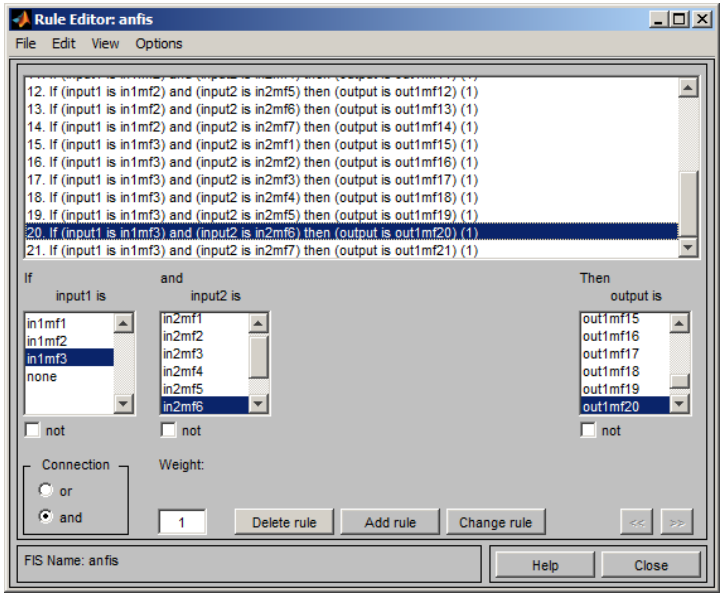
Przykład 2

```
data = [rand(10,1) 10*rand(10,1)-5
rand(10,1)];
    fis = genfis1(data,[3 7],char('pimf','trimf'));
    [x,mf] = plotmf(fis,'input',1);
    subplot(2,1,1), plot(x,mf);
    xlabel('input 1 (pimf)');
    [x,mf] = plotmf(fis,'input',2);
    subplot(2,1,2), plot(x,mf);
    xlabel('input 2 (trimf)');
```









Funkcja genfis2

Funkcja *genfis2: - tworzy FIS typu Takagi Sugeno, na podstawie grupowania danych w oparciu o algorytmu* **subtractive clustering**. *Domyślnie tworzony system zbudowany jest w oparciu o gaussowskie funkcje przynależności.*

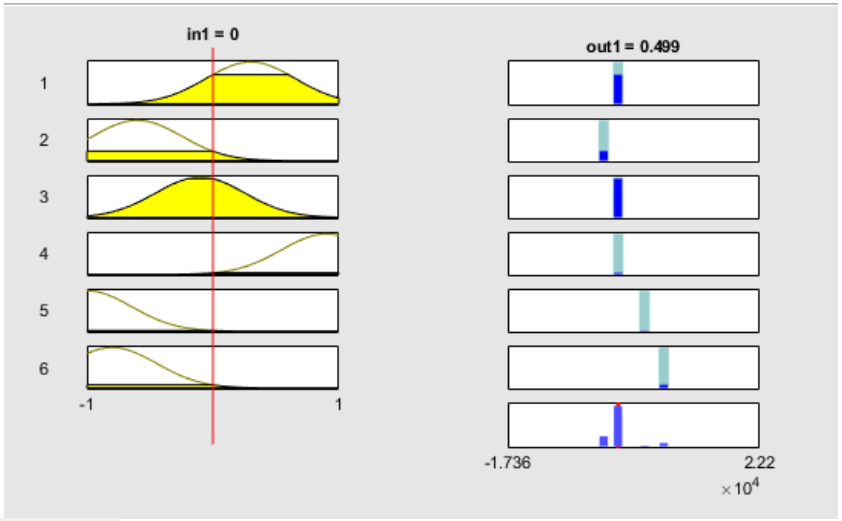
fismat = genfis2(Xin,Xout,radii)

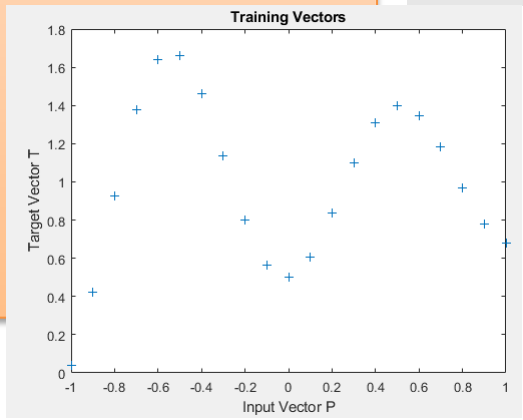
gdzie:
fismat– wynikowa struktura FIS
Xin – zbiór danych wejściowych (zmienne używane do predykcji)
Xout –zmienne wyjściowe – dla nas wyjściem jest kolumna z etykietami
Radii - jest wektorem określającym zakres wpływu centrum klastra w każdym z wymiarów danych, przy założeniu, że dane mieszczą się w jednostkowym hiper boksie.

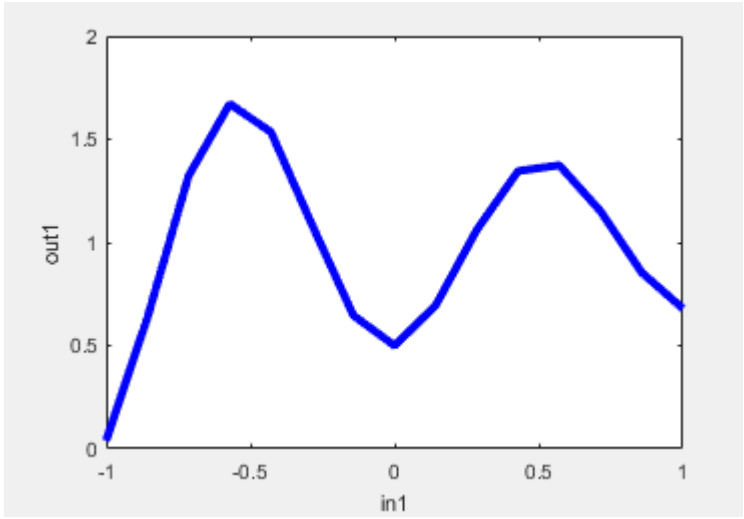
Stworzona przez obydwa inicjatory struktur FIS może następnie zostać poddana optymalizacji w oparciu a algorytm ANFIS, dzięki czemu położenie, oraz parametry funkcji przynależności oraz odpowiednie parametry funkcji wyjściowej zostaną automatycznie zoptymalizowane na podstawie danych.

Przykład

```
X = -1:.1:1;
T = [-.9602 -.5770 -.0729 .3771 .6405 .6600
.4609 ...
.1336 -.2013 -.4344 -.5000 -.3930 -.1647 .0988
...
.3072 .3960 .3449 .1816 -.0312 -.2189 -
.3201]+1;
data=[X' T'];
plot(X,T,'+');
title('Training Vectors');
xlabel('Input Vector P');
ylabel('Target Vector T');
F2=genfis2(X',T',0.5);
fuzzy(F2);
```







Przykład 2

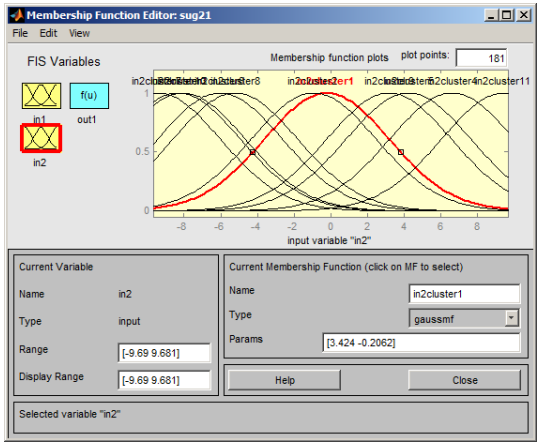
Xin1 = 7*rand(50,1);

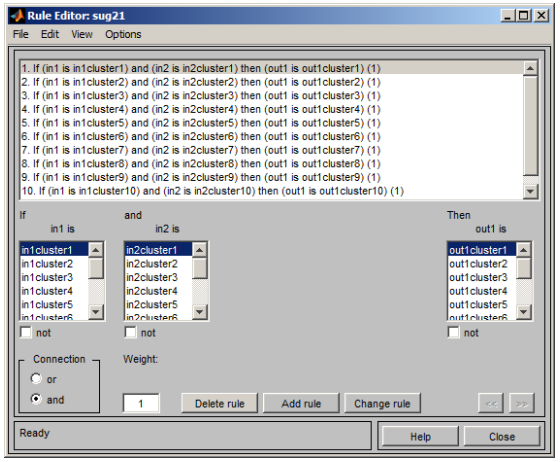
Xin2 = 20*rand(50,1)-10;

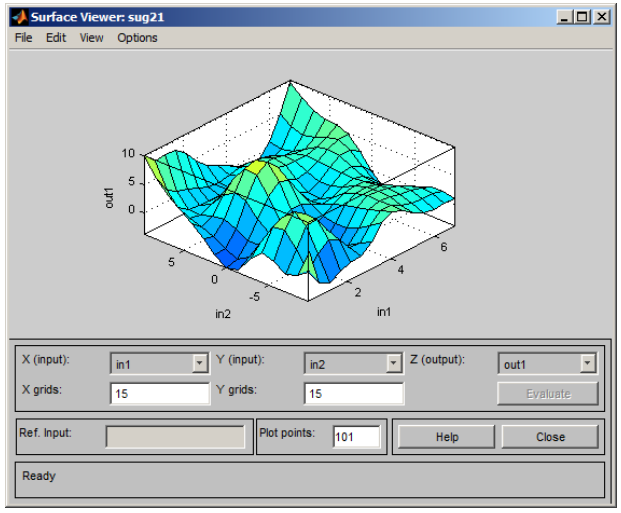
Xin = [Xin1 Xin2];

Xout = 5*rand(50,1);

fis = genfis2(Xin,Xout,0.5) specifies a range of influence of 0.5 for all data dimensions.



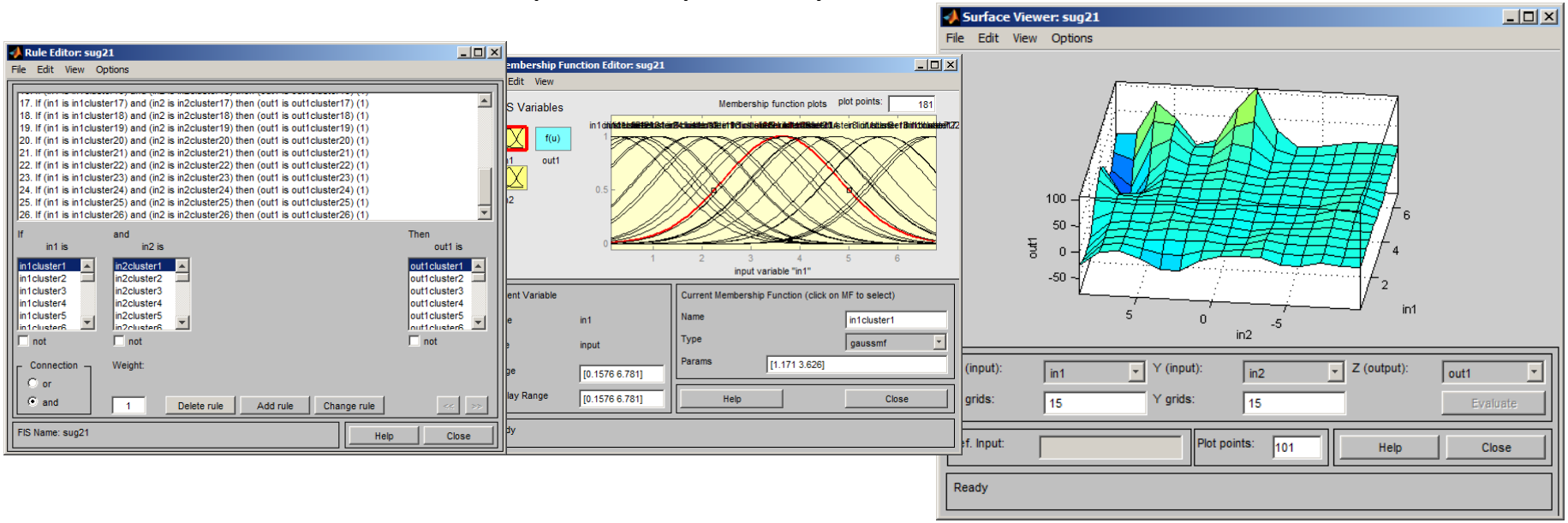




Przykład3

```
Xin1 =  7*rand(50,1);
    Xin2 = 20*rand(50,1)-10;
    Xin  = [Xin1 Xin2];
    Xout =  5*rand(50,1);
```

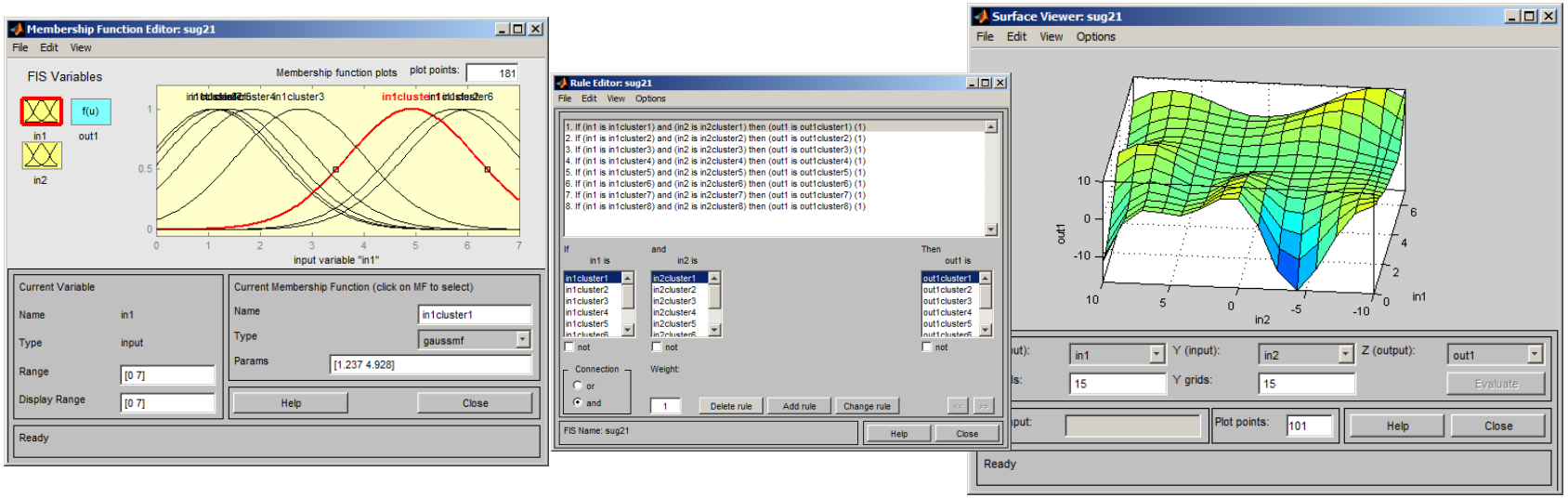
fis = genfis2(Xin,Xout,[0.5 0.25 0.3]) specifies the ranges of influence in the first, second, and third data dimensions are 0.5, 0.25, and 0.3 times the width of the data space, respectively.



Przykład4

```
Xin1 =  7*rand(50,1);
    Xin2 = 20*rand(50,1)-10;
    Xin  = [Xin1 Xin2];
    Xout =  5*rand(50,1);
```

fis = genfis2(Xin,Xout,0.5,[0 -10 0; 7 10 5]) specifies the data in the first column of Xin are scaled from [0 7], the data in the second column of Xin are scaled from [-10 10], and the data in Xout are scaled from [0 5].

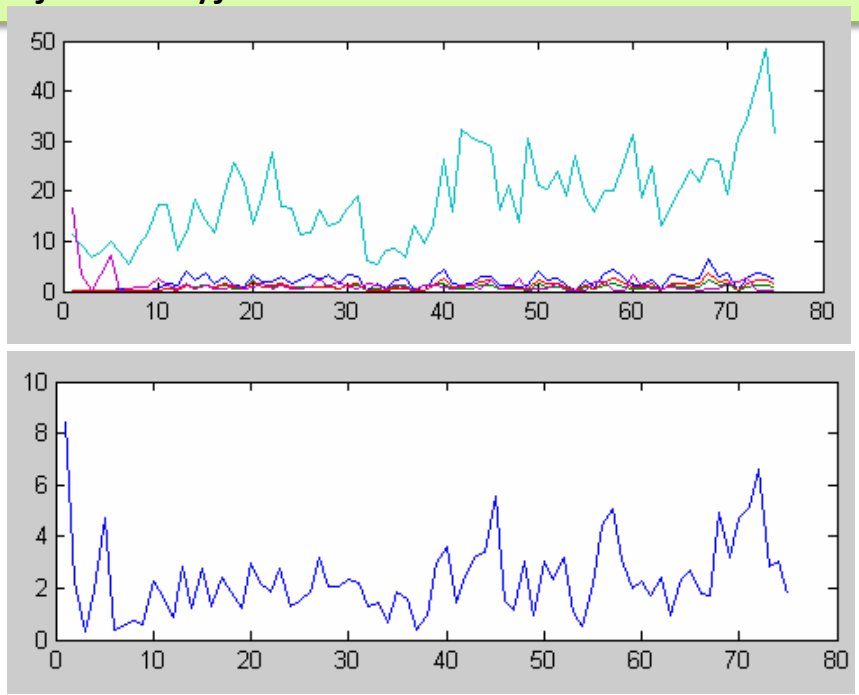


Przykład 5

Modelowanie relacji między liczbą wyjazdów samochodowych a demografią. Dane demograficzne i podróży są od analizy 100 stref drogowych. Mamy pięć demograficznych czynników:
ludność,
liczby jednostek mieszkalnych,
własności pojazdu,
mediana dochodów gospodarstw domowych, a łączne zatrudnienie.
Stąd model ma pięć zmiennych wejściowych i jedno wyjście.

```
clear
close all
mytripdata
subplot(2,1,1), plot(datin)
subplot(2,1,2), plot(datout)
```

chkdata1	25x6 double
chkdatin	25x5 double
chkdatout	25x1 double
datin	75x5 double
datout	75x1 double
trndata1	75x6 double



Tworzenie i sprawdzenie FIS modeli

```
fismat=genfis2(datin,datout,0.5);
```

```
fuzout=evalfis(datin,fismat);
trnRMSE=norm(fuzout-
datout)/sqrt(length(fuzout))
```

```
chkfuzout=evalfis(chkdatin,fismat);
chkRMSE=norm(chkfuzout-
chkdatout)/sqrt(length(chkfuzout))
```

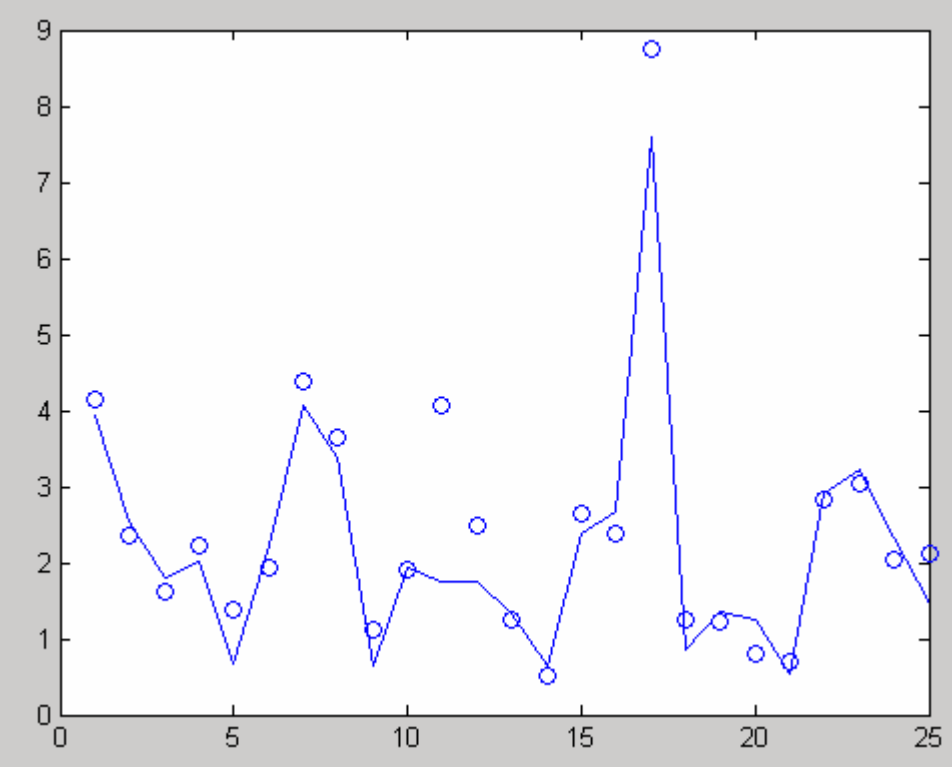
```
trnRMSE =
    0.5276
```

```
chkRMSE =
    0.6179
```

GENFIS2 może być używany do generowania początkowych FIS szkolenia ANFIS najpierw stosując odjemnikowego grupowania na danych.
GENFIS2 osiąga to poprzez wyodrębnianie zbioru przepisów, które modeluje zachowania danych. Metoda ekstrakcji zasada najpierw używa się SUBCLUST funkcję określającą liczbę reguł i funkcji przynależności przesyłek a następnie używa ocenę liniowej najmniejszych kwadratów do określenia każdej reguły wynikające z tego równania.

Sprawdzenie

```
figure
plot(chkdatout)
hold on
plot(chkfuzout,'o')
hold off
```



Ulepszenie jakości modelu

```
fismat2=anfis([datin datout],fismat,[20 0 0.1]);
```

20 jest liczbą epok, 0 błędów jest celem szkolenia i 0.1 jest początkowa wielkość kroku.

```
ANFIS info:
        Number of nodes: 44
        Number of linear parameters: 18
        Number of nonlinear parameters: 30
        Total number of parameters: 48
        Number of training data pairs: 75
        Number of checking data pairs: 0
        Number of fuzzy rules: 3

Start training ANFIS ...

 1          0.527607
.
.
 20         0.420275
```

```
fuzout2=evalfis(datin,fismat2);
trnRMSE2=norm(fuzout2-
datout)/sqrt(length(fuzout2))
chkfuzout2=evalfis(chkdatin,fismat2);
chkRMSE2=norm(chkfuzout2-
chkdatout)/sqrt(length(chkfuzout2))
```

```
trnRMSE2 =
        0.4203
chkRMSE2 =
        0.5894
```

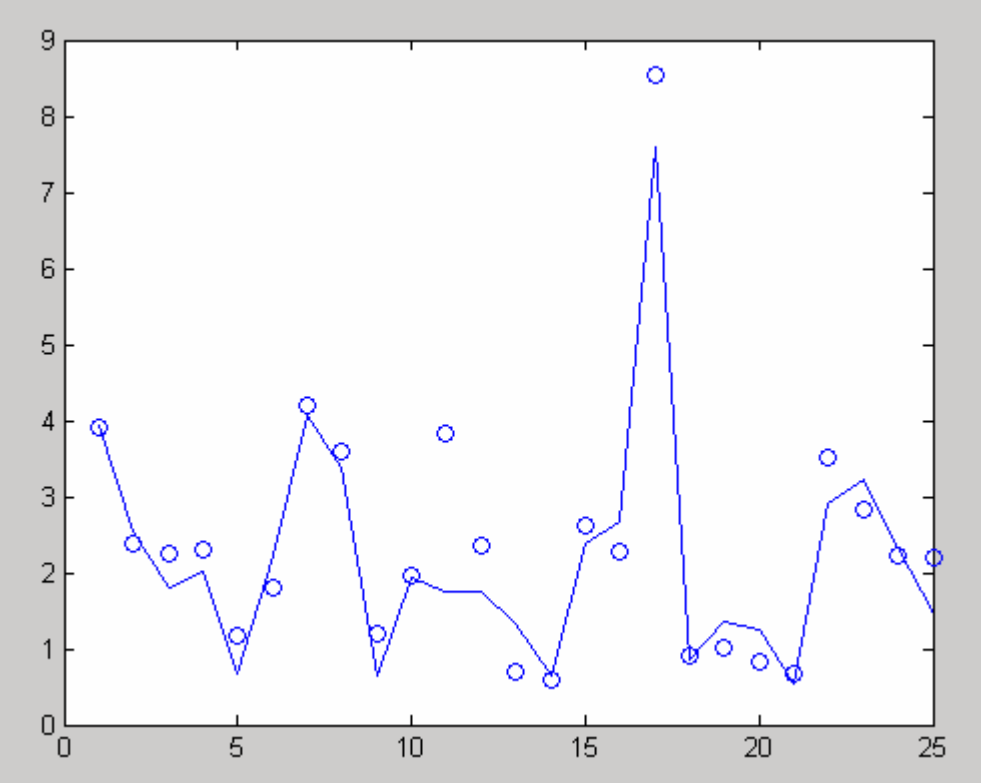
było

0.5276

0.6179

Sprawdzanie

```
figure
plot(chkdatout)
hold on
plot(chkfuzout2,'o')
hold off
```



Nadmiarowe kształcenie (overfitting)

Nadmiarowe kształcenie można wykryć po kontroli błędów danych sprawdzających które zaczyna się **wzrastać**, podczas gdy błąd danych szkolenia nadal **maleje**.

```
[fismat3,trnErr,stepSize,fismat4,chkErr]= ...
  anfis([datin datout],fismat,[200 0 0.1],[], ...
  [chkdatin chkdatout]);
```

fismat3 jest FIS struktura gdy błąd danych szkolenia osiąga minimum. **fismat4** jest struktura FIS podczas sprawdzania błąd danych sprawdzających osiąga minimum.

```
1      0.527607      0.617875
 2     0.513727      0.615487
 .
 .
 200   0.326576      0.601531
```

```
fuzout4=evalfis(datin,fismat4);
trnRMSE4=norm(fuzout4-
datout)/sqrt(length(fuzout4))
chkfuzout4=evalfis(chkdatin,fismat4);
chkRMSE4=norm(chkfuzout4-
chkdatout)/sqrt(length(chkfuzout4))
```

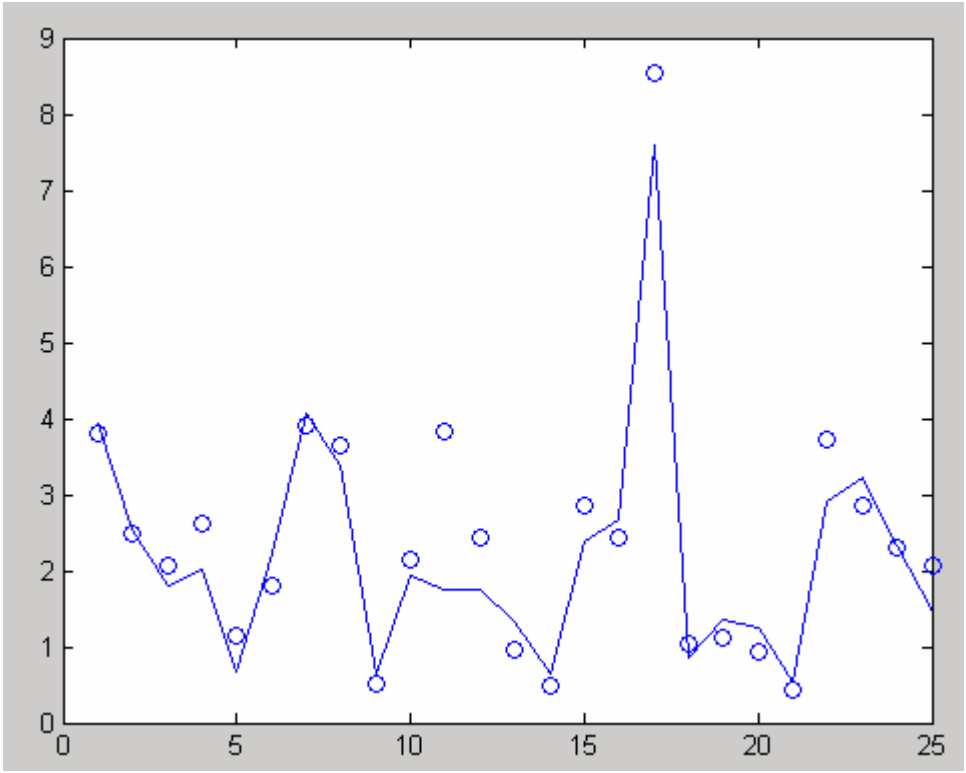
Błąd w danych szkoleniowych jest **najniższy** a błąd w danych kontrolnych jest **nieco niższa** niż było.

To jest **nadmiarowe kształcenie**

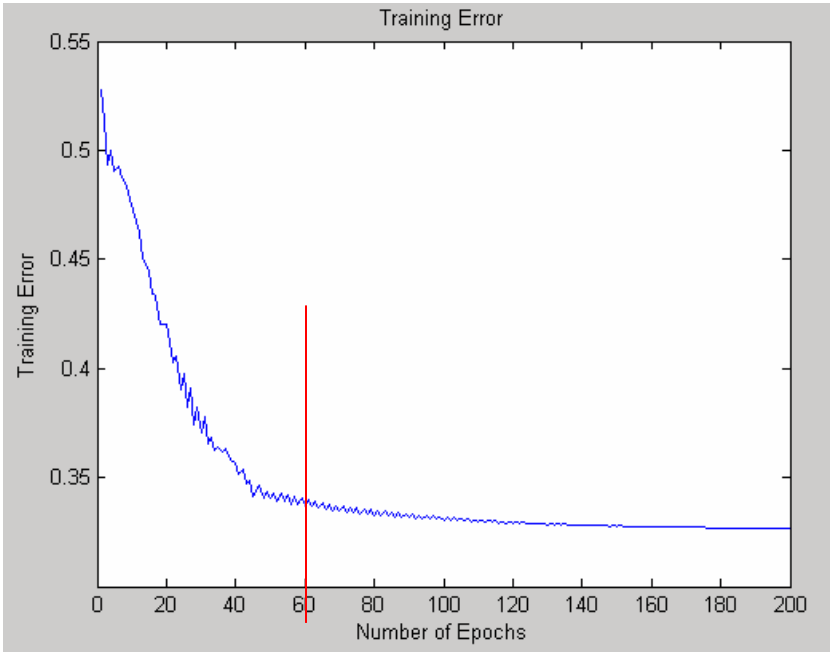
```
trnRMSE4 =
   0.3393
chkRMSE4 =
   0.5833
```

Sprawdzanie

```
figure
plot(chkdatout)
hold on
plot(chkfuzout4,'o')
hold off
```

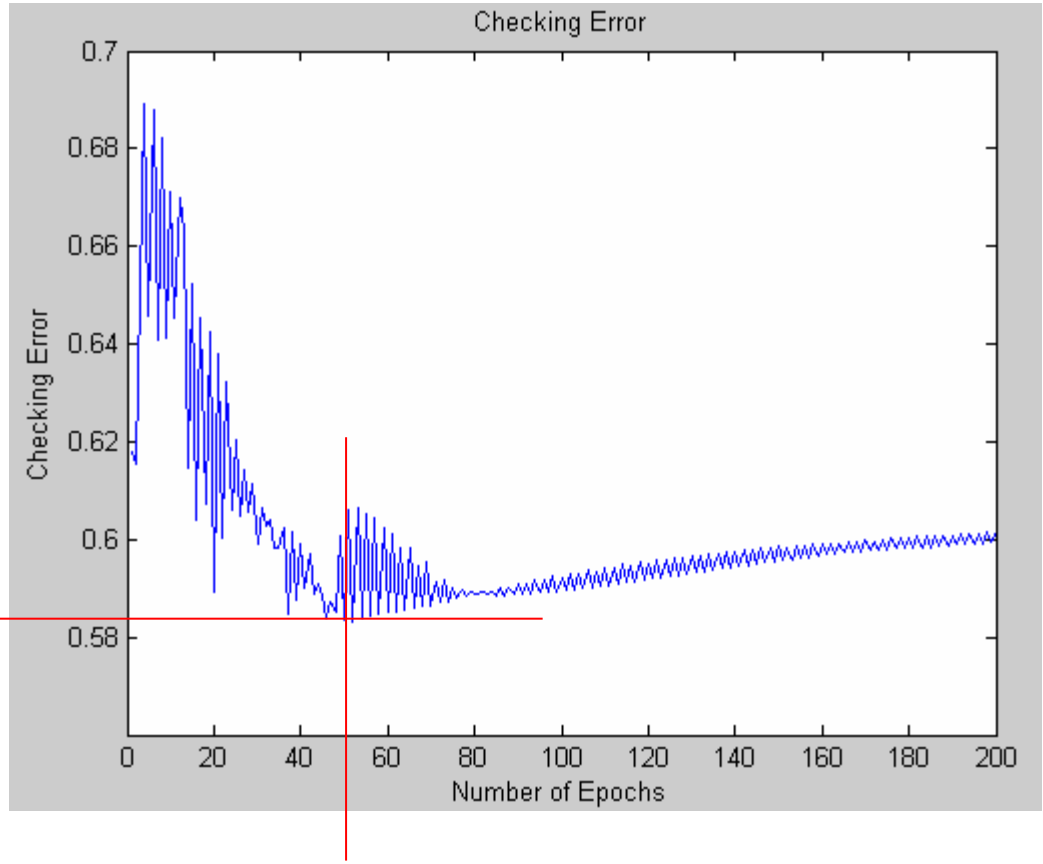



```
figure
plot(trnErr)
Title('Training Error')
xlabel('Number of Epochs')
ylabel('Training Error')
```



Sprawdzanie

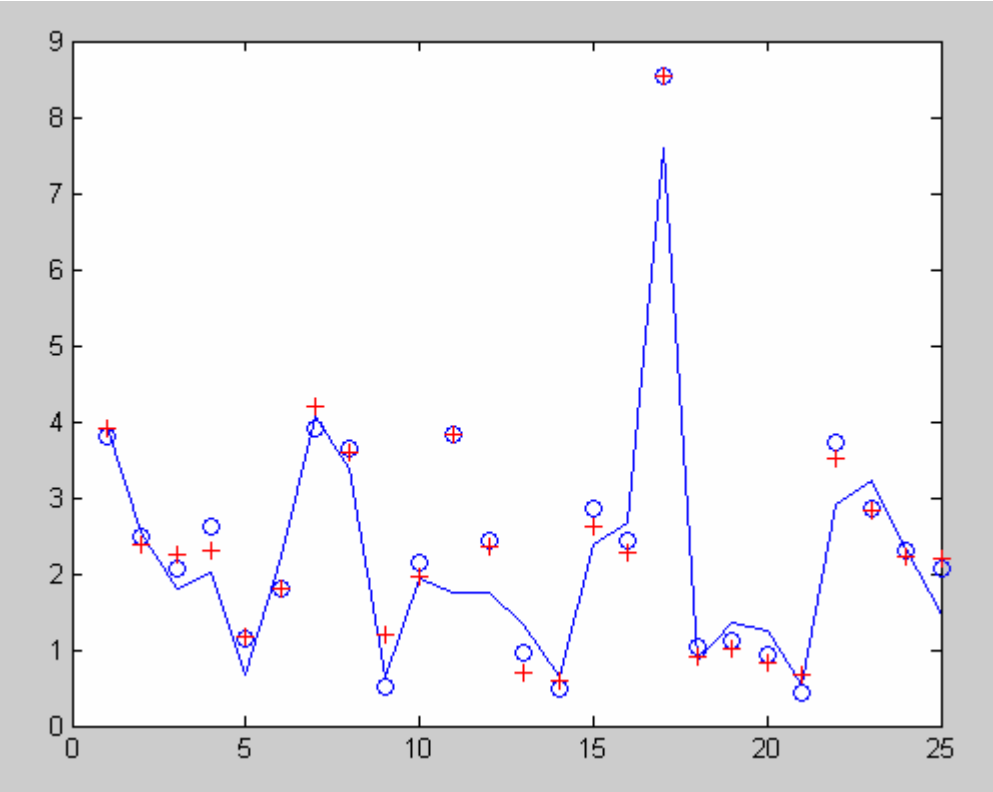
```
figure
plot(chkErr)
Title('Checking Error')
xlabel('Number of
Epochs')
ylabel('Checking Error')
```



minimum

Sprawdzanie modeli

```
figure
plot(chkdatout)
hold on
plot(chkfuzout4,'ob')
plot(chkfuzout2,'+r')
```



Funkcja genfis3

Funkcja *genfis3: - tworzy FIS typu Takagi Sugeno, na podstawie grupowania danych w oparciu o algorytmu* ***rozmytych c-średnich (FCM).*** *Domyślnie tworzony system zbudowany jest w oparciu o gaussowskie funkcje przynależności.*

fismat = genfis3(Xin,Xout,type,cluster_n)

gdzie:
fismat– wynikowa struktura FIS
Xin – zbiór danych wejściowych (zmienne używane do predykcji)
Xout –zmienne wyjściowe – dla nas wyjściem jest kolumna z etykietami
type – typ struktury FIS, możliwe opcje to Mamdani/Sugeno
cluster_n – liczba klastrów na które **zbiór danych wejściowych** zostanie podzielony

Stworzona przez obydwa inicjatory struktur FIS może następnie zostać poddana optymalizacji w oparciu a algorytm ANFIS, dzięki czemu położenie, oraz parametry funkcji przynależności oraz odpowiednie parametry funkcji wyjściowej zostaną automatycznie zoptymalizowane na podstawie danych.

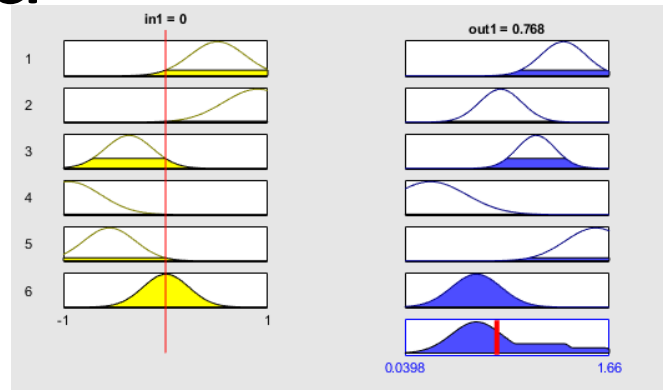
Funkcja genfis3

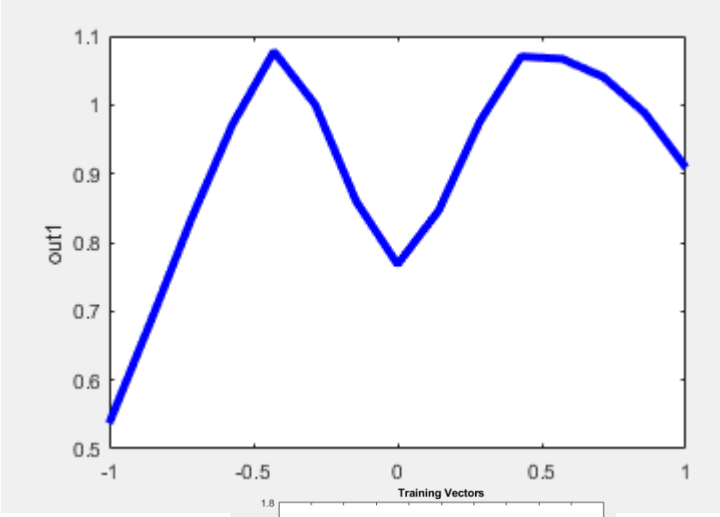
FIS = GENFIS3(XIN, XOUT, TYPE, CLUSTER_N) allows you to specify the number of clusters to be generated by FCM in the argument CLUSTER_N. **The number of clusters determines the number of rules and membership functions in the generated FIS**. This argument also takes the value 'auto' in which case it uses the subclust algorithm with a radii of 0.5 to find the number of clusters.

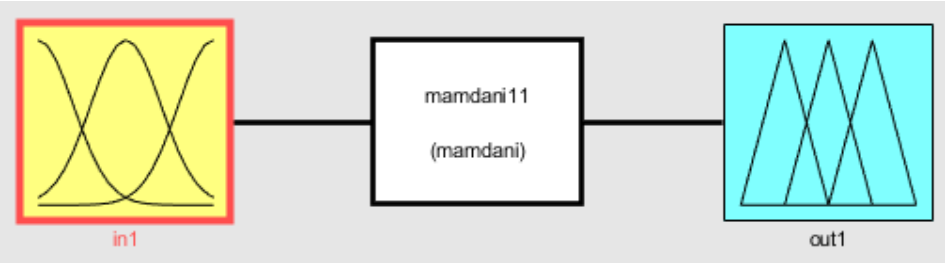
FIS = GENFIS3(XIN, XOUT, TYPE, CLUSTER_N, FCMOPTIONS) allows you to specify options for the FCM algorithm. Type HELP FCM for a list of options that can be specified for the FCM algorithm.

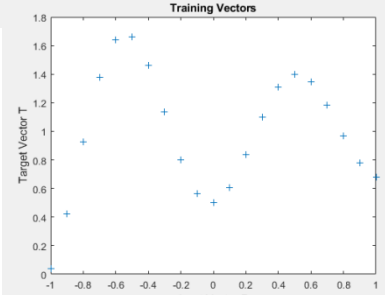
Przykład

```
X = -1:.1:1;
T = [-.9602 -.5770 -.0729 .3771 .6405 .6600
.4609 ...
.1336 -.2013 -.4344 -.5000 -.3930 -.1647 .0988
...
.3072 .3960 .3449 .1816 -.0312 -.2189 -
.3201]+1;
data=[X' T'];
plot(X,T,'+');
title('Training Vectors');
xlabel('Input Vector P');
ylabel('Target Vector T');
F3M=genfis3(X',T','mamdani');
```

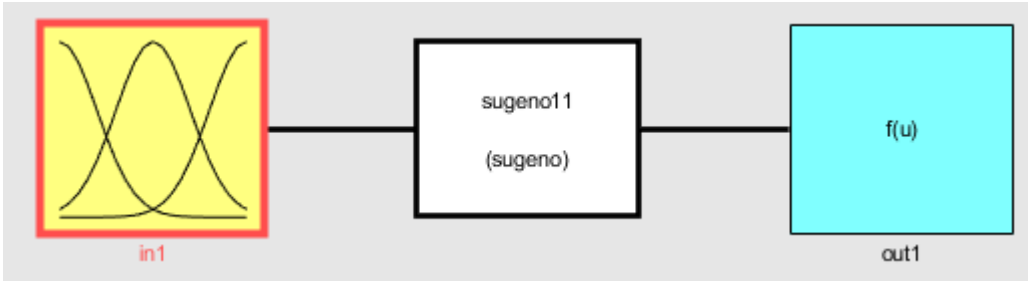


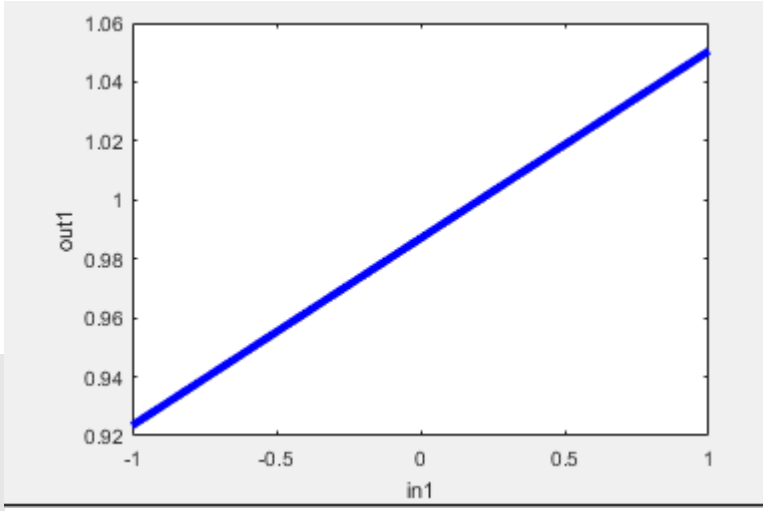


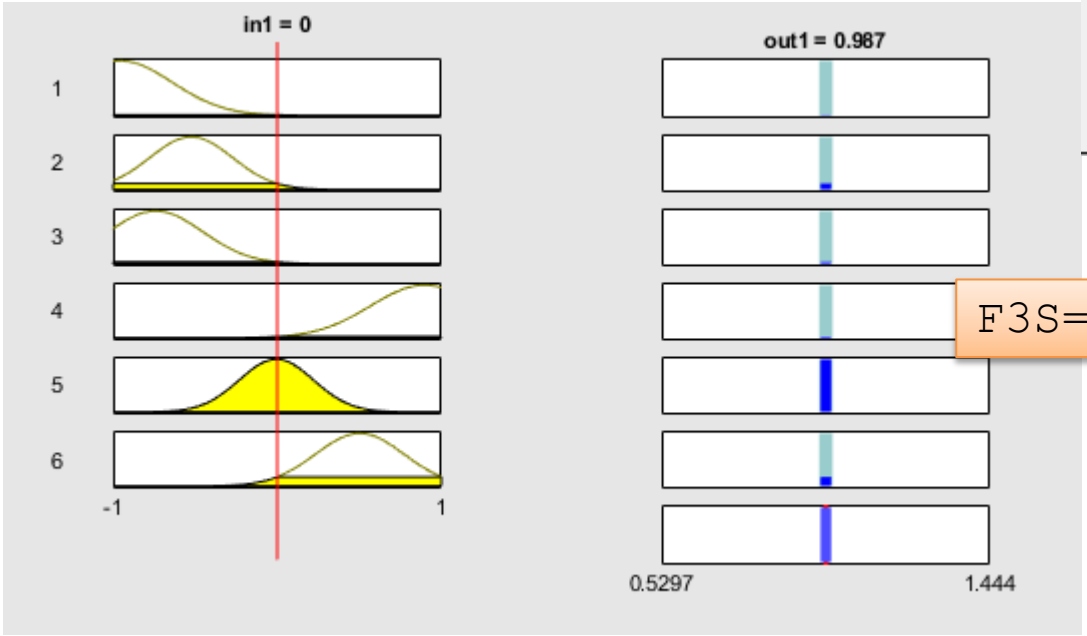




Przykład (cd)

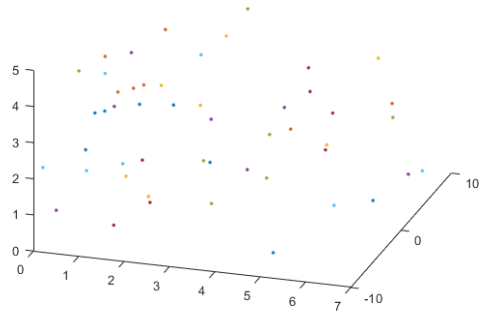


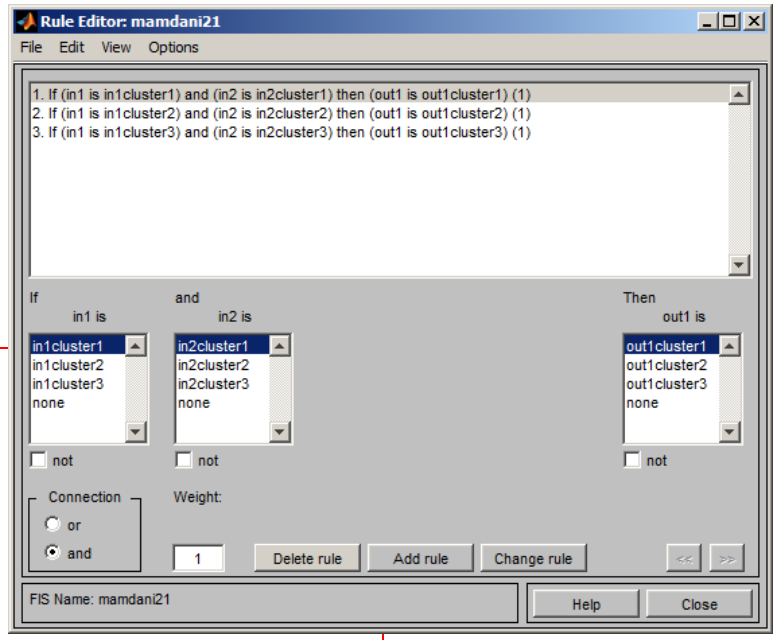




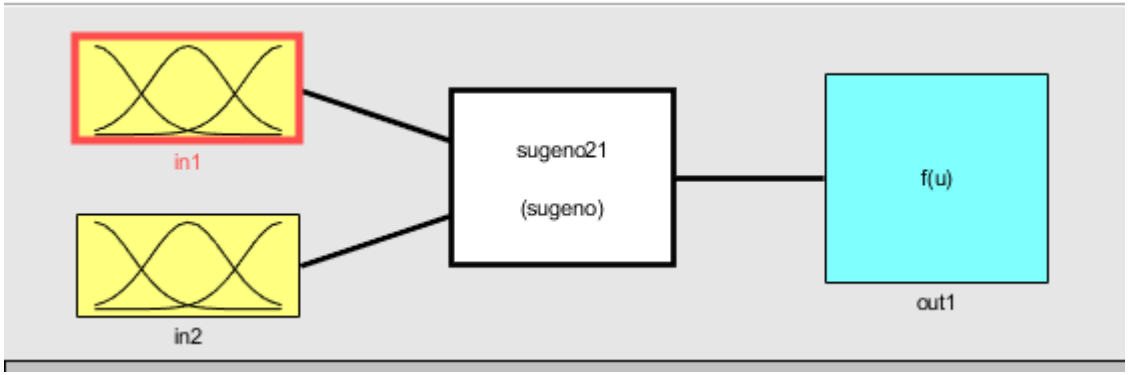

```
F3S=genfis3(X',T','sugeno');
```

Przykład 2






```
Xin1 =  7*rand(50,1);
      Xin2 = 20*rand(50,1)-10;
      Xin  = [Xin1 Xin2];
      Xout =  5*rand(50,1);
      fis = genfis3(Xin,Xout); %Sugeno

      fis = genfis3(Xin,Xout,'mamdani',3)
      specifies the type of FIS and the number of clusters desired.

      fis = genfis3(Xin,Xout,'mamdani',3,[2,100,1e-5,1])
      specifies the type of FIS, the number of clusters desired and
FCM
      options
```


in1
in2
sugeno21
(sugeno)
f(u)
out1

Adaptive Neuro-Fuzzy Inference System (ANFIS)

- Dowolna sieć ma właściwości adaptacyjne, jedynym warunkiem jest kierunek pracy.

- ANFIS - klasa sieci adaptacyjnych, o właściwościach ekwiwalentnych do układów rozmytych

Wprowadzenie

- Systemy rozmyte modelują procesy, które trudno opisać metodami klasycznymi (n.p. równaniami różniczkowymi)
- Słabości klasycznych układów rozmytych:
  - nie ma standardowej metody do przekształcenia wiedzy eksperckiej na reguły wnioskowania rozmytego
  - potrzebna jest efektywna metoda dostrajania funkcji przynależności (MF), tak by minimalizować błędy wnioskowania

Architektura ANFIS

- Wnioskowanie Takagi -Sugeno

Rule 1: If x is A_1 and y is B_1, then $f_1 = p_1 x + q_1 y + r_1$
Rule 2: If x is A_2 and y is B_2, then $f_2 = p_2 x + q_2 y + r_2$

System wnioskowania rozmytego

- Wnioskowanie rozmyte:

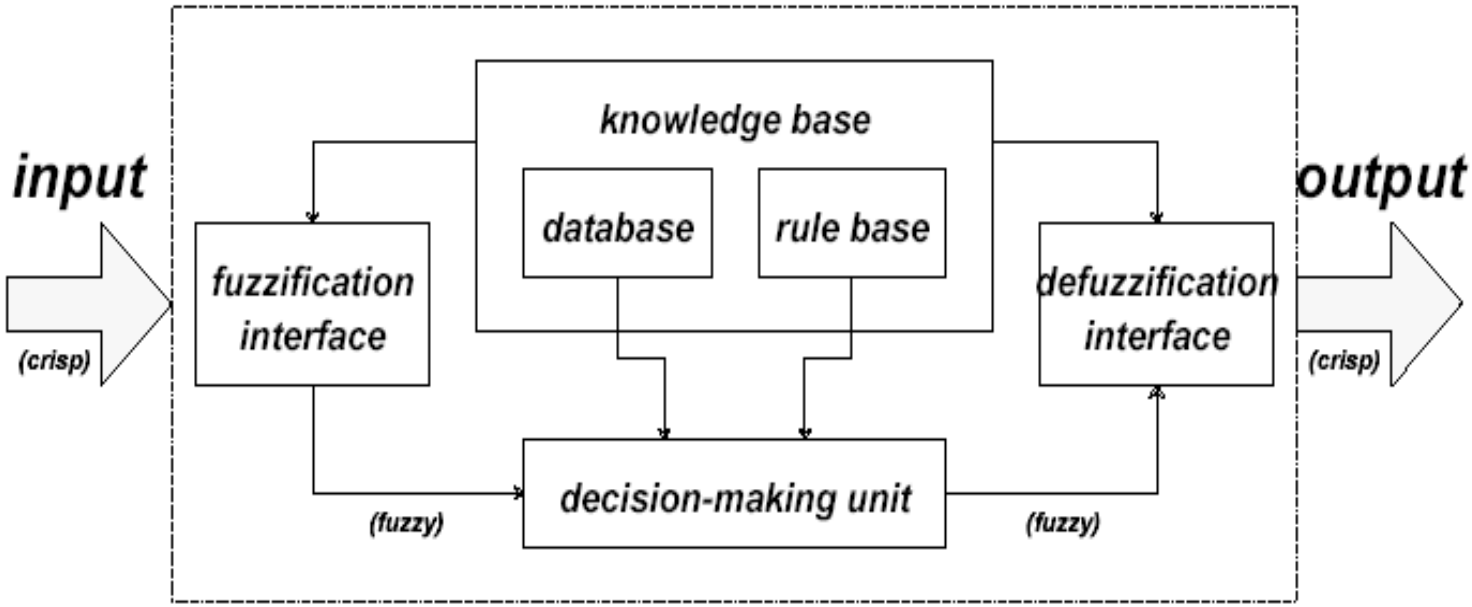
if A then B

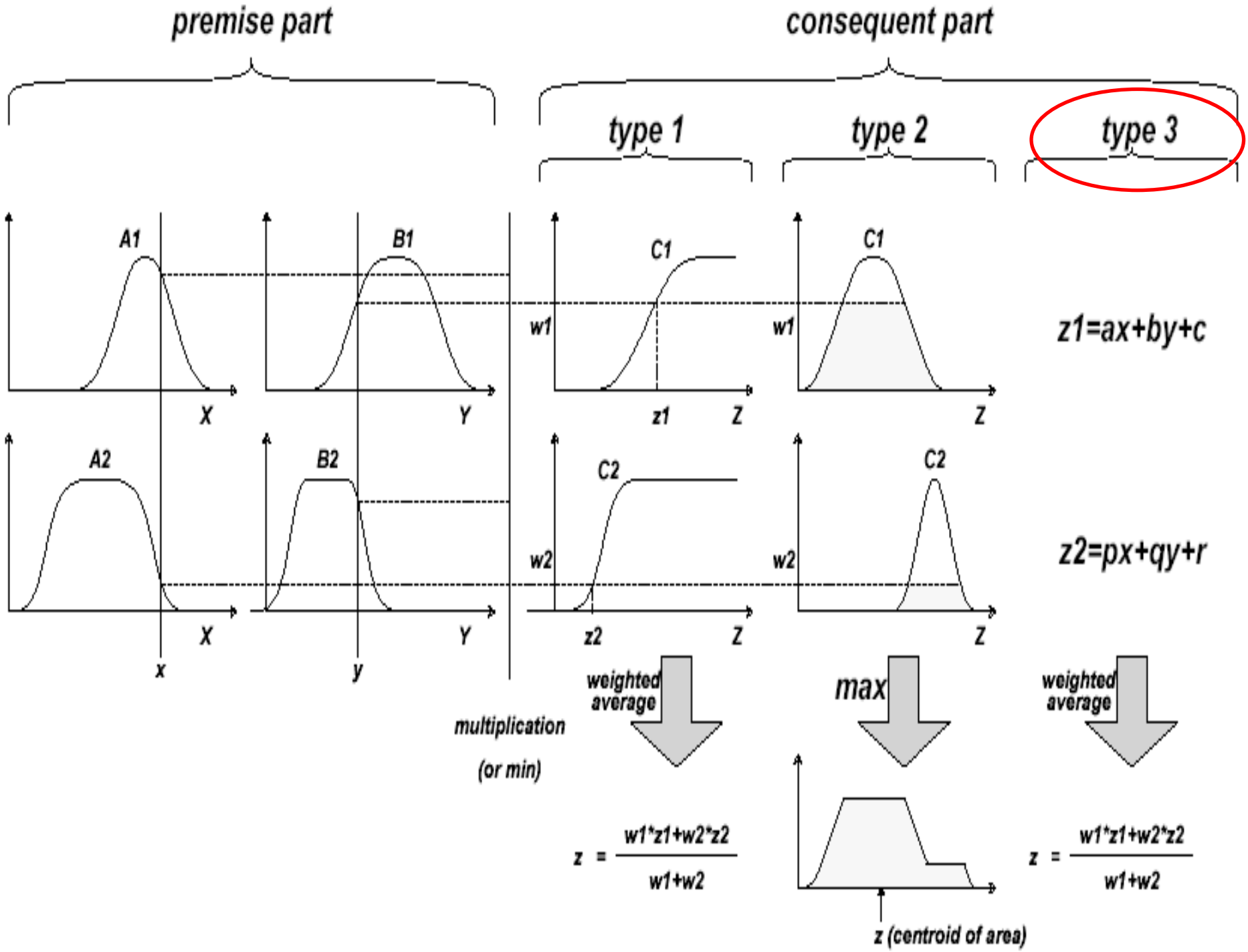
przesłanka *wniosek*

- **if** *temperatura jest niska* **then** *załącz dużą moc*

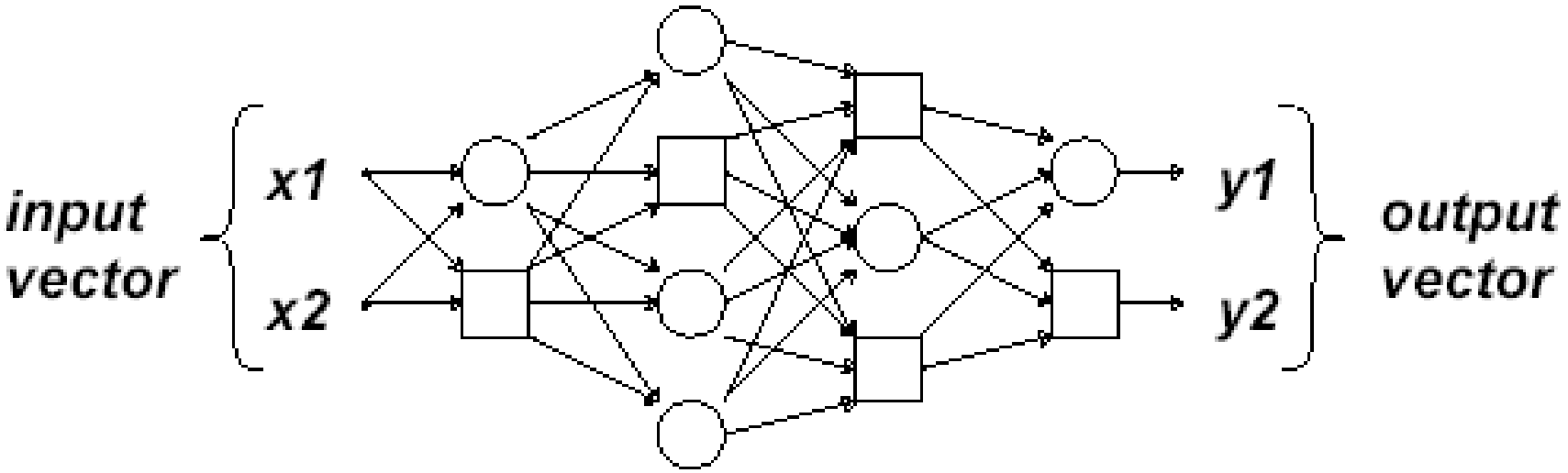
- wnioskowanie Takagi-Sugeno:
  if *prędkośc jest duża* **then** *siła=k*(prędkość)*^{2}

System wnioskowania rozmytego




premise part
consequent part
type 1
type 2
type 3
A1
B1
C1
C1
w1
w1
X
Y
z1
Z
Z
z1=ax+by+c
A2
B2
C2
C2
w2
w2
X
Y
z2
Z
Z
z2=px+qy+r
x
y
multiplication
(or min)
weighted average
max
weighted average
z = w1*z1+w2*z2 / w1+w2
z = w1*z1+w2*z2 / w1+w2
z (centroid of area)

Sieć adaptacyjna



Węzły adaptacyjne (kwadraty) opisane zestawem parametrów i węzły stałe - bez parametrów (okręgi)

$$O_i^k = O_i^k(O_1^{k-1}, \ldots O_{\#(k-1)}^{k-1}, a, b, c, \ldots);$$

Zarys metody uczenia

Błąd dla p-tego wektora uczącego

$$E_p = \sum_{m=1}^{\#(L)} (T_{m,p} - O_{m,p}^{L})^2$$

Zależność błędu od parametrów węzła - warstwa wyjściowa

$$\frac{\partial E_p}{\partial O_{i,p}^{L}} = -2(T_{i,p} - O_{i,p}^{L})$$

Zależność błędu od parametrów węzła - dla węzłów warstw ukrytych

$$\frac{\partial E_p}{\partial O_{i,p}^{k}} = \sum_{m=1}^{\#(k+1)} \frac{\partial E_p}{\partial O_{m,p}^{k+1}} \frac{\partial O_{m,p}^{k+1}}{\partial O_{i,p}^{k}}$$

Zarys metody uczenia

Zależność błędu od parametru α

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \frac{\partial O^*}{\partial \alpha}$$

Zależność sumarycznego błędu od parametru α

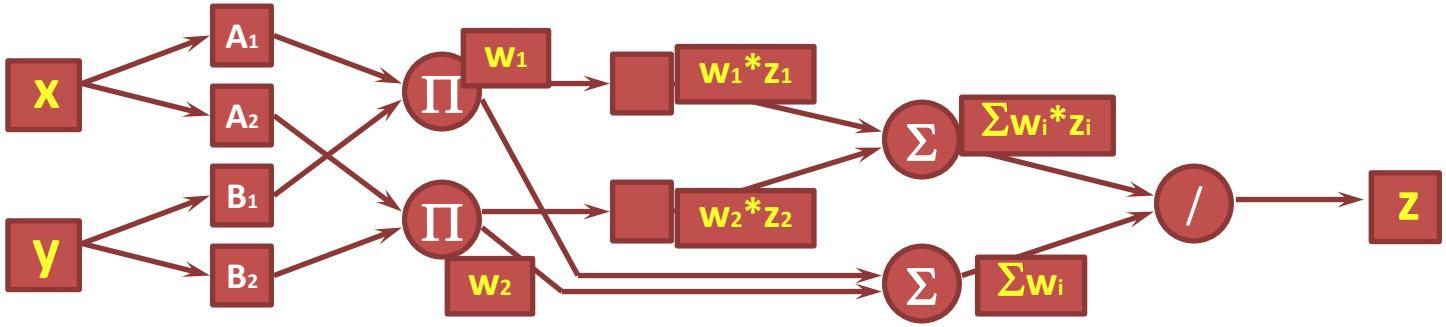
$$\frac{\partial E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial E_p}{\partial \alpha} \qquad \triangle \alpha = -\eta \frac{\partial E}{\partial \alpha}$$

Adaptive Neuro-Fuzzy Inference System

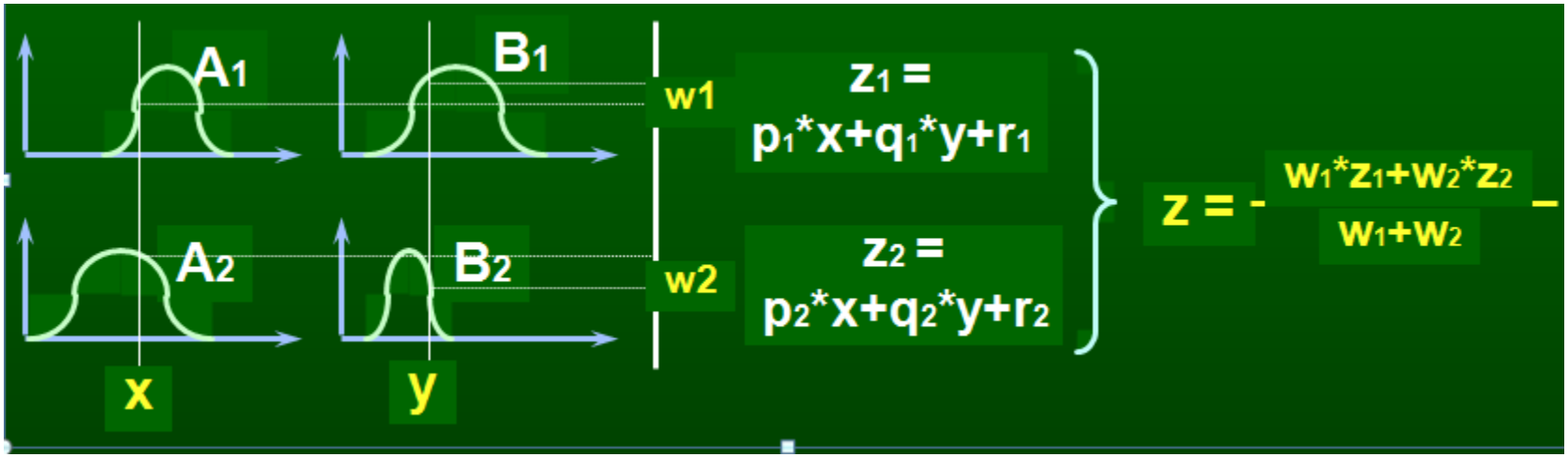
- Dowolna sieć ma właściwości adaptacyjne, jedynym warunkiem jest kierunek pracy.

- ANFIS - klasa sieci adaptacyjnych, o właściwościach ekwiwalentnych do układów rozmytych

ANFIS (Adaptive Neuro-Fuzzy Inference System)

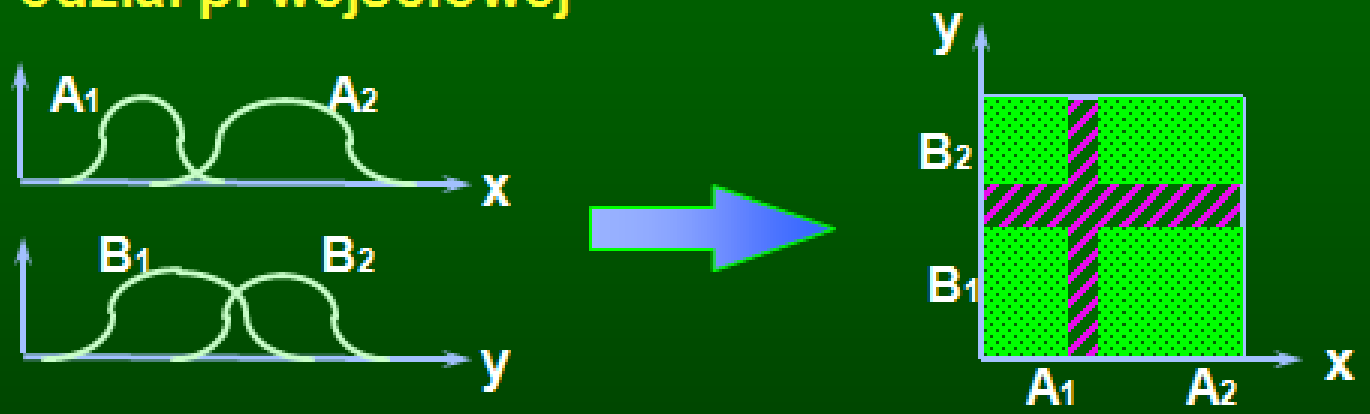


wnioskowanie

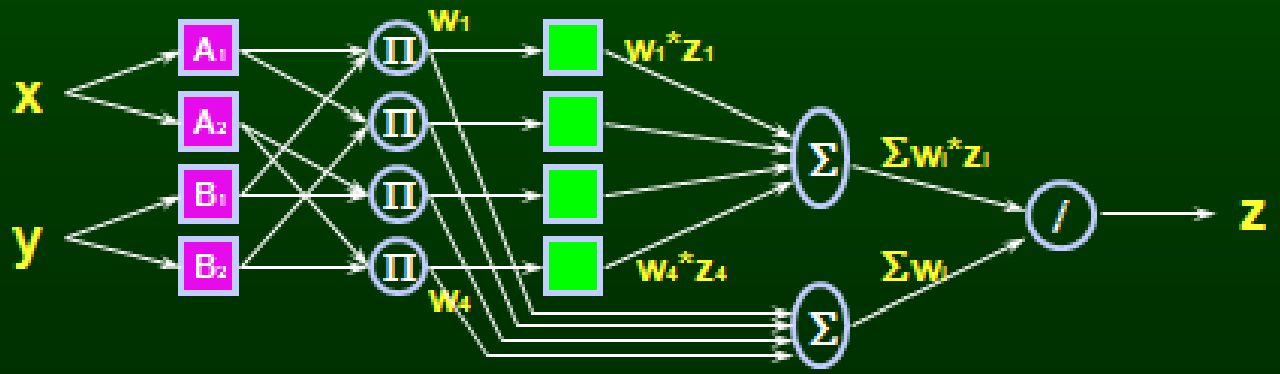


ANFIS z 4 regułami

- Podział p. wejściowej

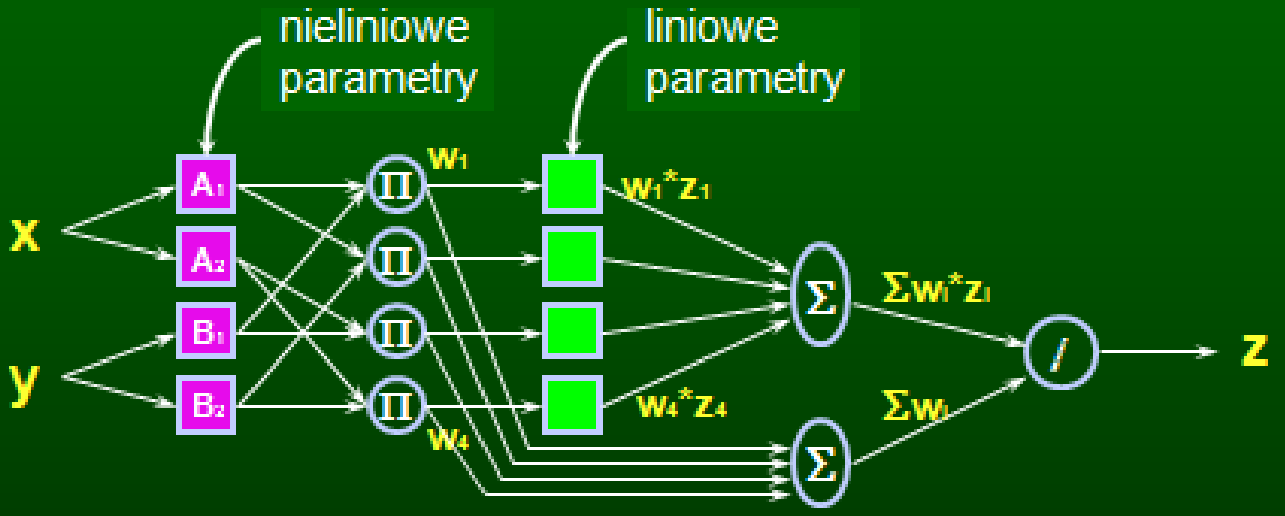


- ANFIS (Adaptive Neuro-Fuzzy Inference System)

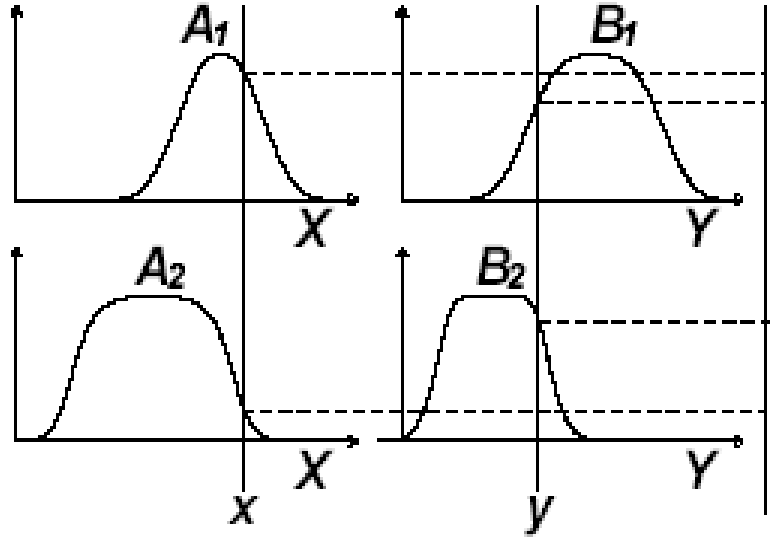


ANFIS: identyfikacja param.

- **Hybrydowe metody trenowania: BP + LMS**



	wprzód	wstecz
Param. FP (nieliniowe)	stałe	gradientowe
Współczynniki (liniowe)	LMS	stałe

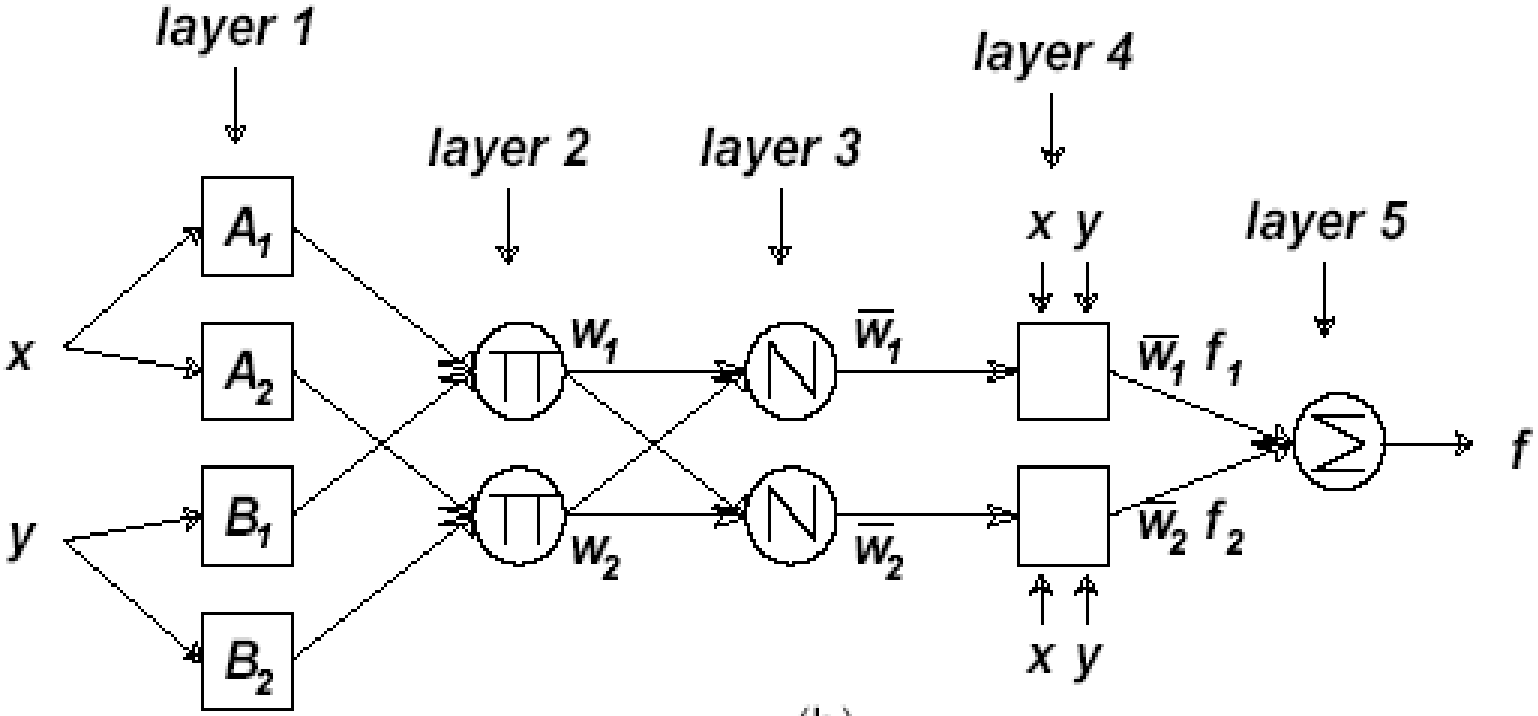


$f_1 = p_1x + q_1y + r_1$

$f_2 = p_2x + q_2y + r_2$

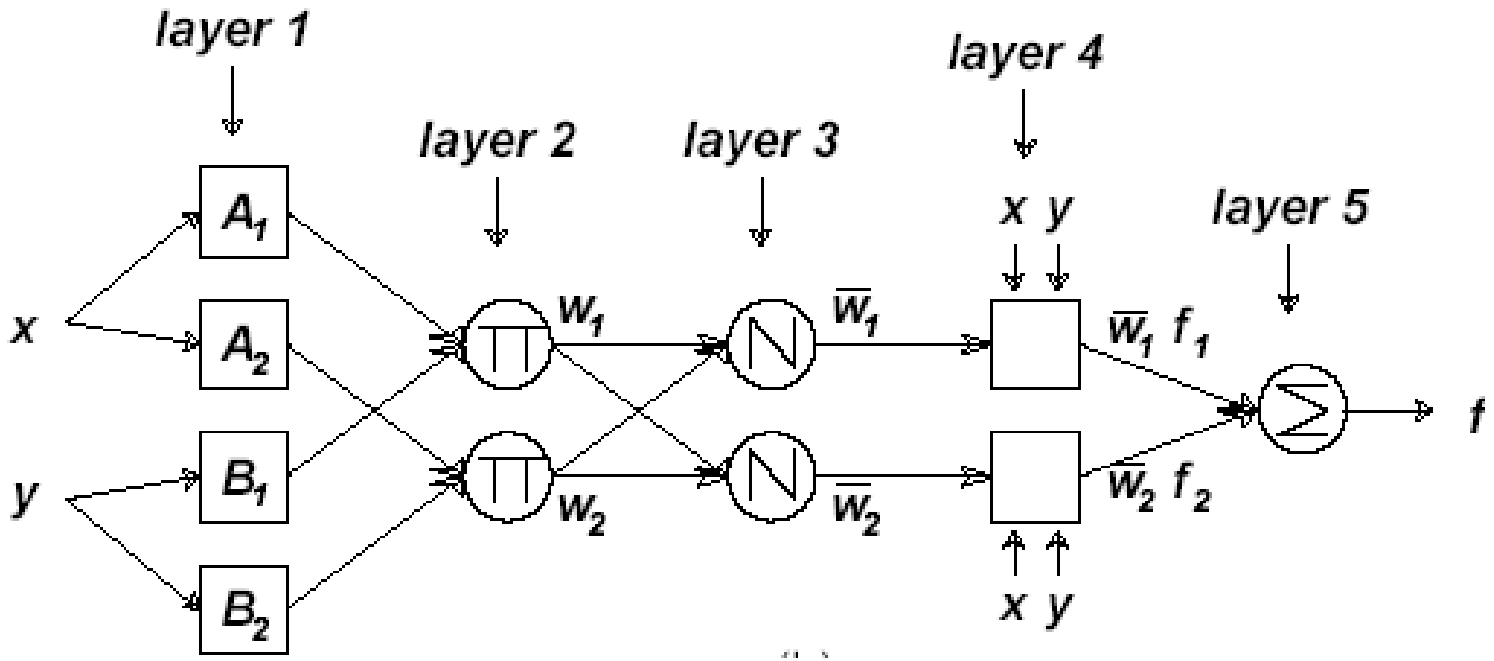
$$f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \overline{w}_1 f_1 + \overline{w}_2 f_2$$

(a)



(b)

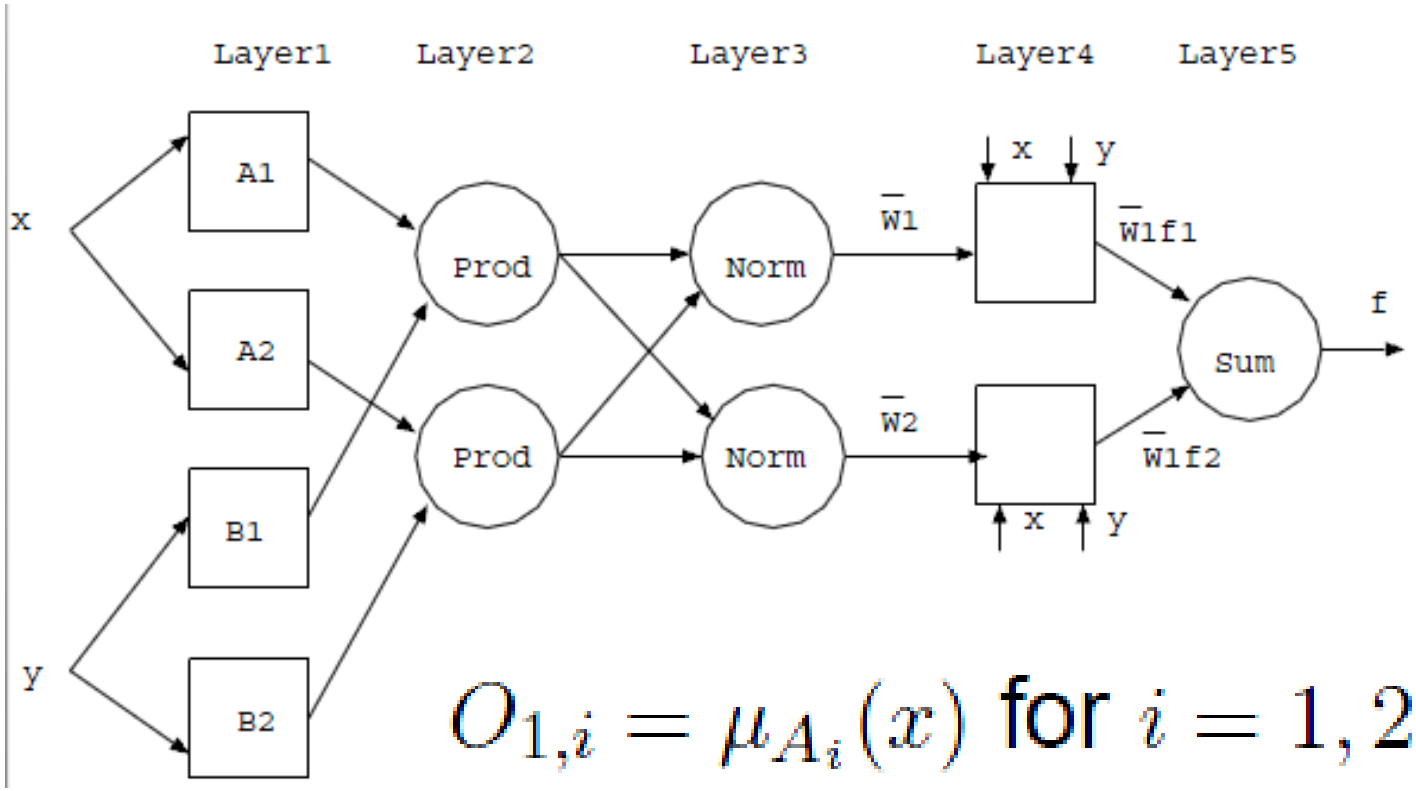
ANFIS warstwa 1



$$O_i^1 = \mu_{A_i}(x) \qquad \mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}}$$

$$\mu_{A_i}(x) = exp\left[-\left(\frac{x - c_i}{a_i}\right)^2\right]$$

Warstwa 1

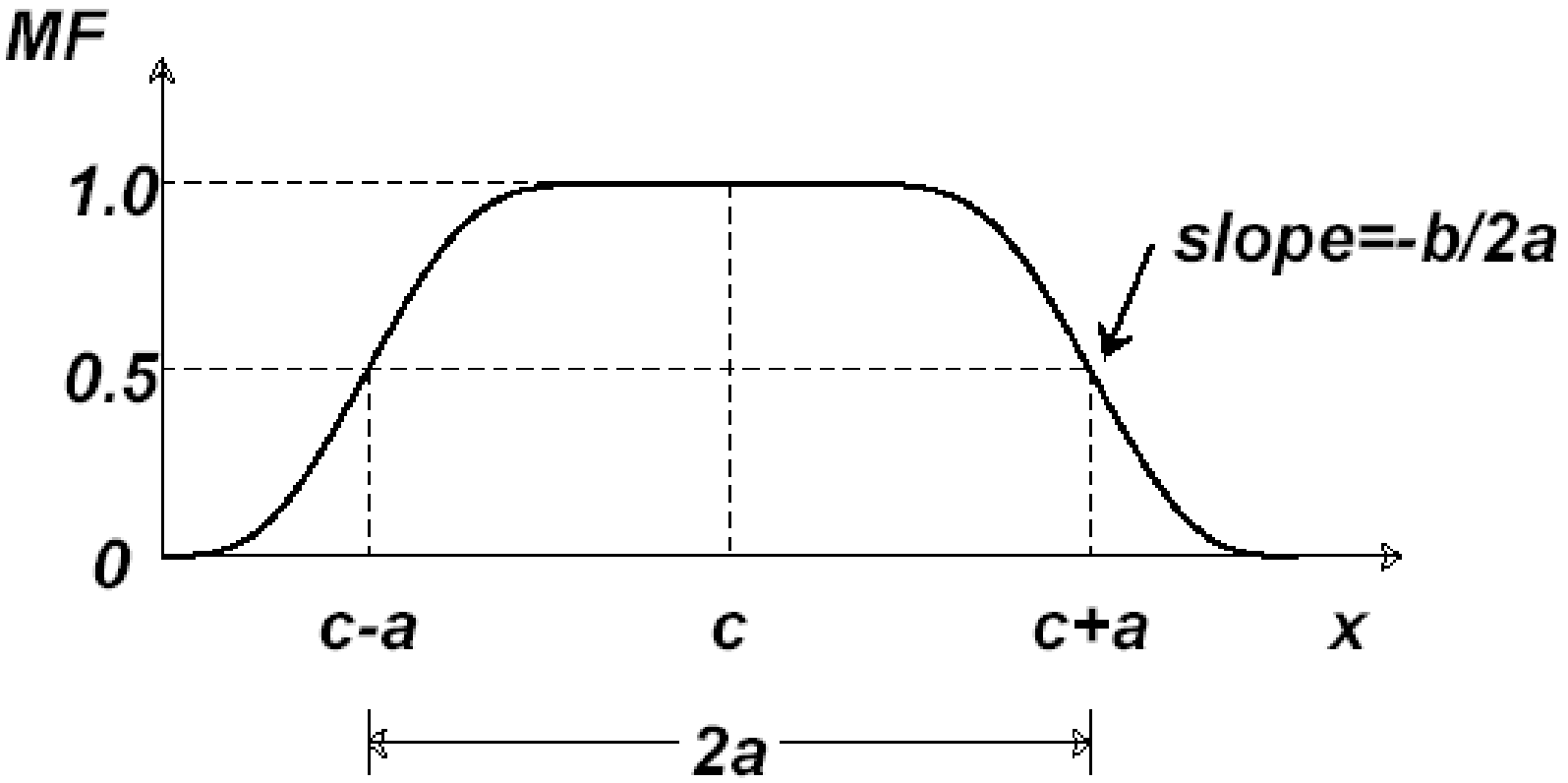


$$O_{1,i} = \mu_{A_i}(x) \text{ for } i = 1, 2, \text{ or}$$

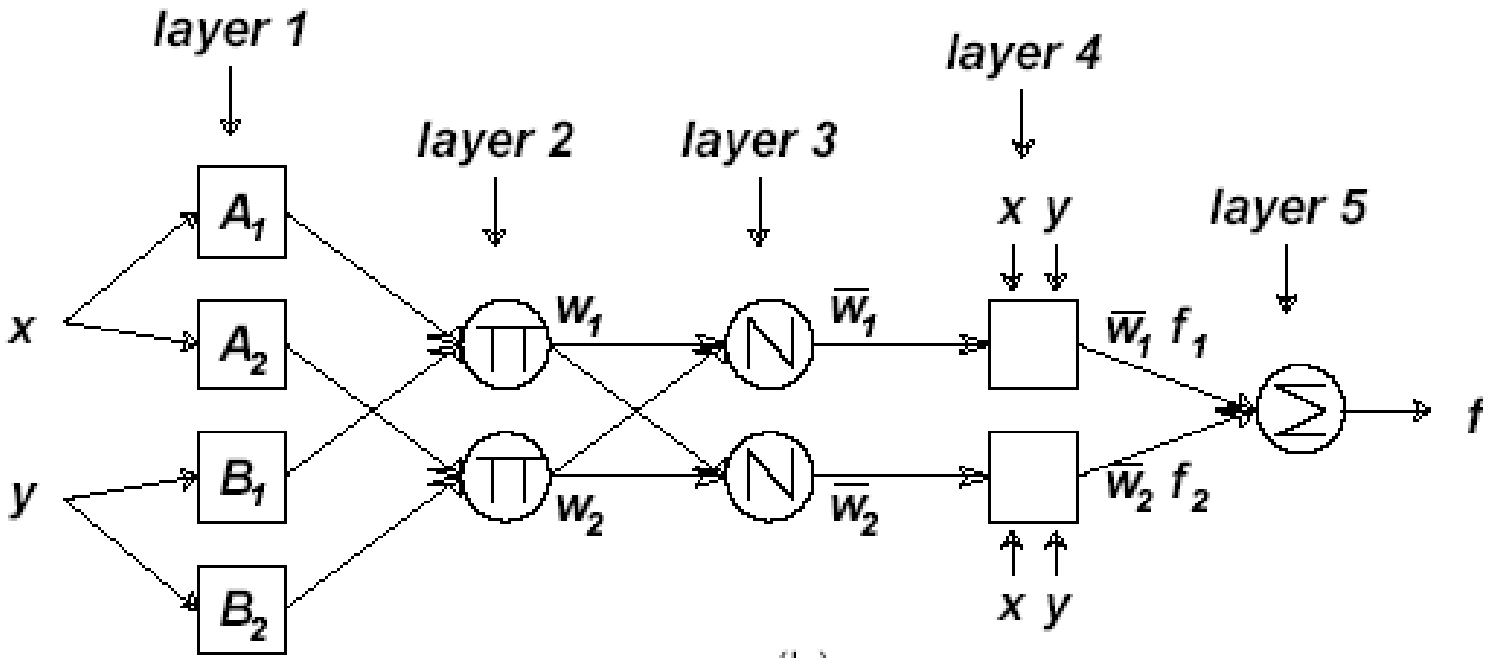
$$O_{1,i} = \mu_{B_{i-2}}(x) \text{ for } i = 3, 4$$

Funkcja dzwonowa

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}}$$



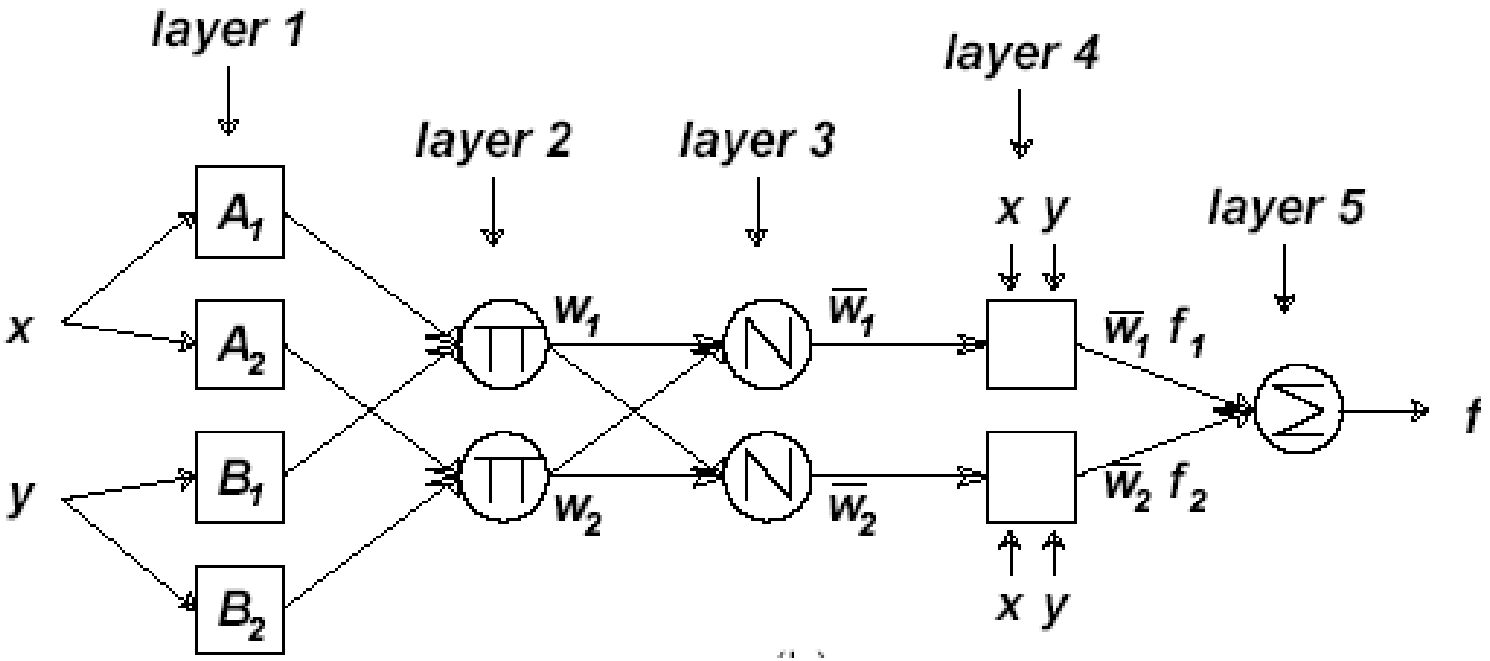
ANFIS warstwa 1



Parametry węzła - ***a,b,c*** - funkcja dzwonowa, lub *a,c* - funkcja gaussowska

Są to **parametry przesłanki**

ANFIS warstwa 2

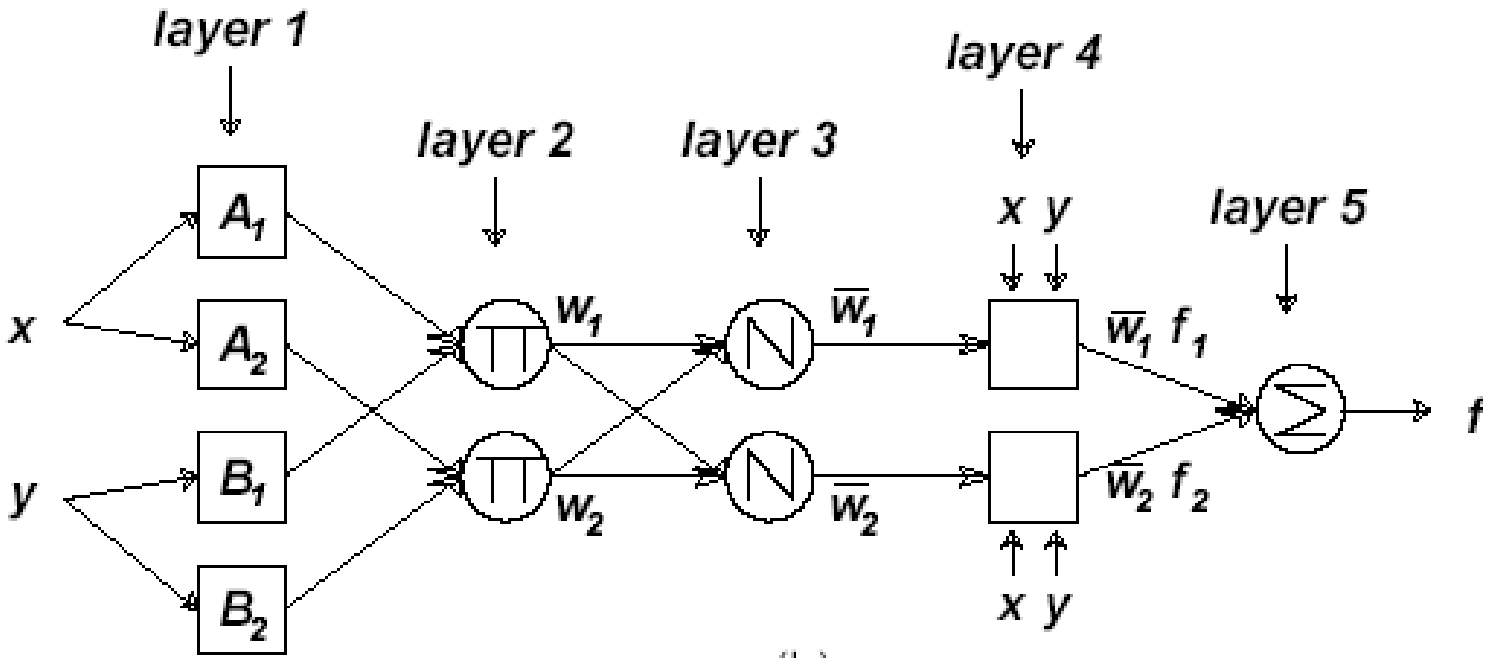


$$O_{2,i} = w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y), \quad i = 1, 2$$

Warstwa bez parametrów. Wyjście odpowiada poziomowi zapłonu poszczególnych reguł

Może to być dowolna T-funkcja (uogólnione AND)

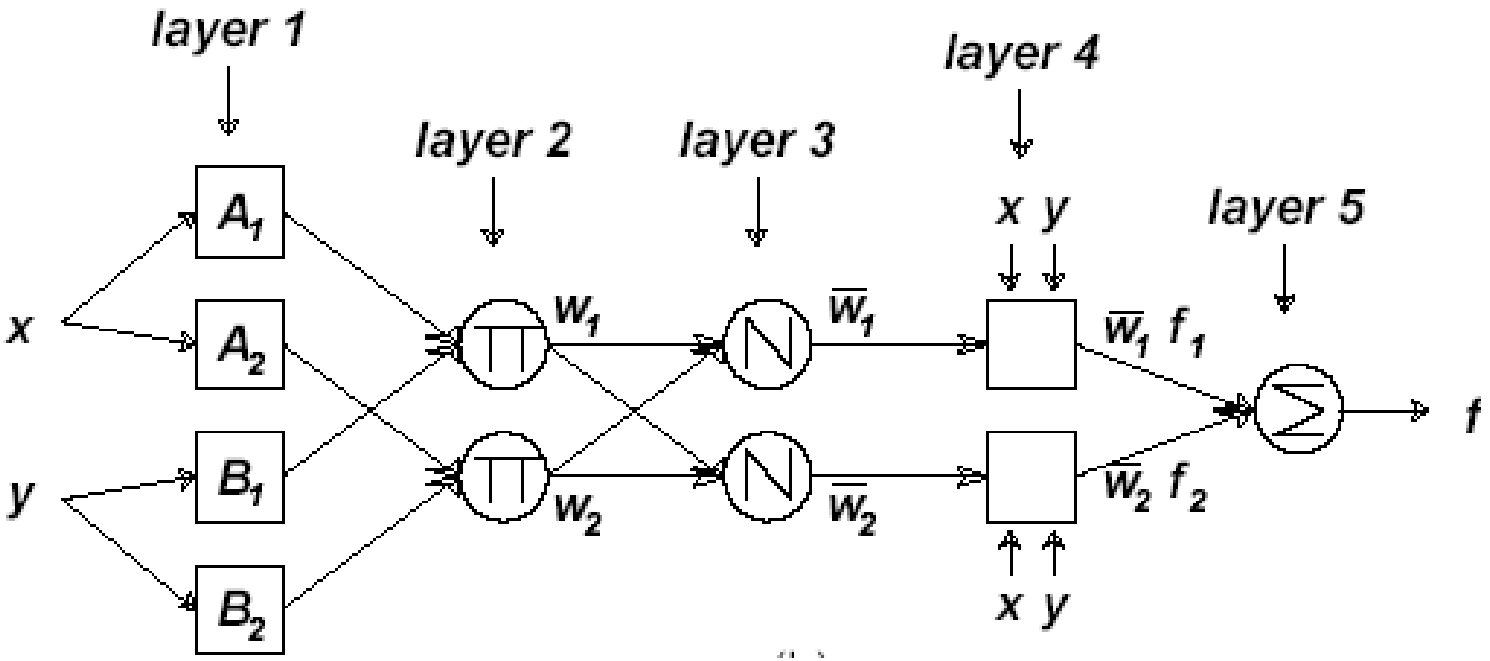
ANFIS warstwa 3



$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2$$

Warstwa bez parametrów. Wyjście odpowiada unormowanemu poziomowi zapłonu poszczególnych reguł

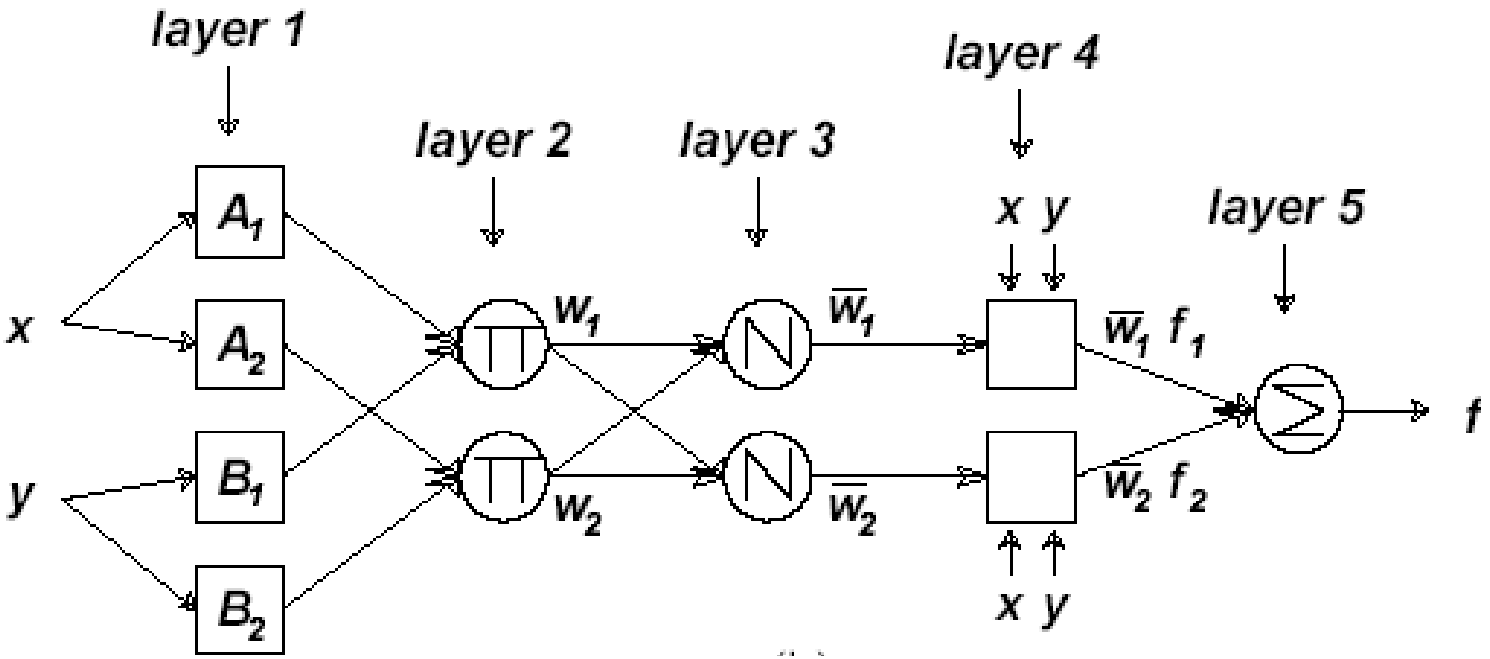
ANFIS warstwa 4



$$O_i^4 = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i)$$

Warstwa z parametrami ***p,q,r***. Są to **parametry wnioskowania**

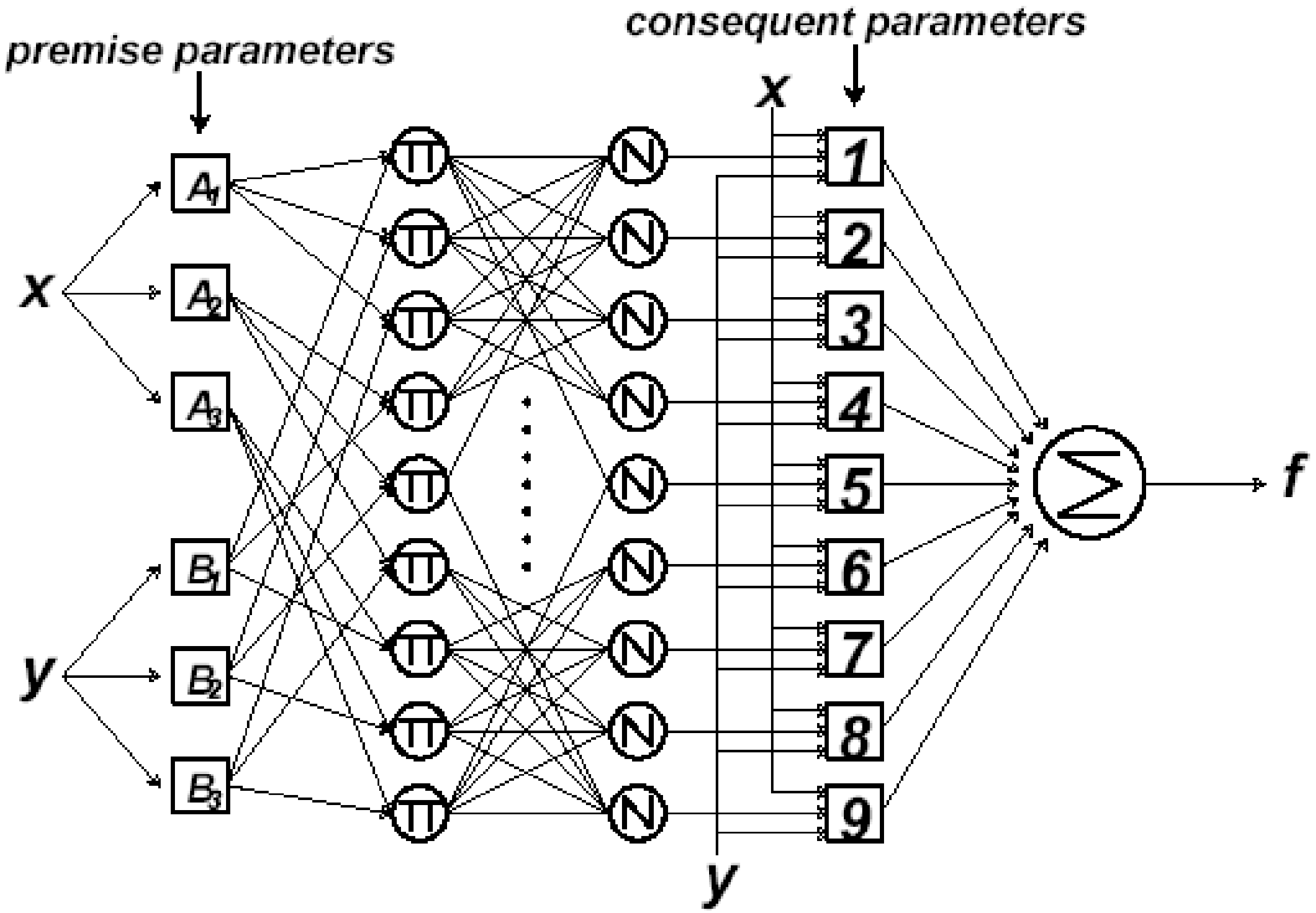
ANFIS warstwa 5



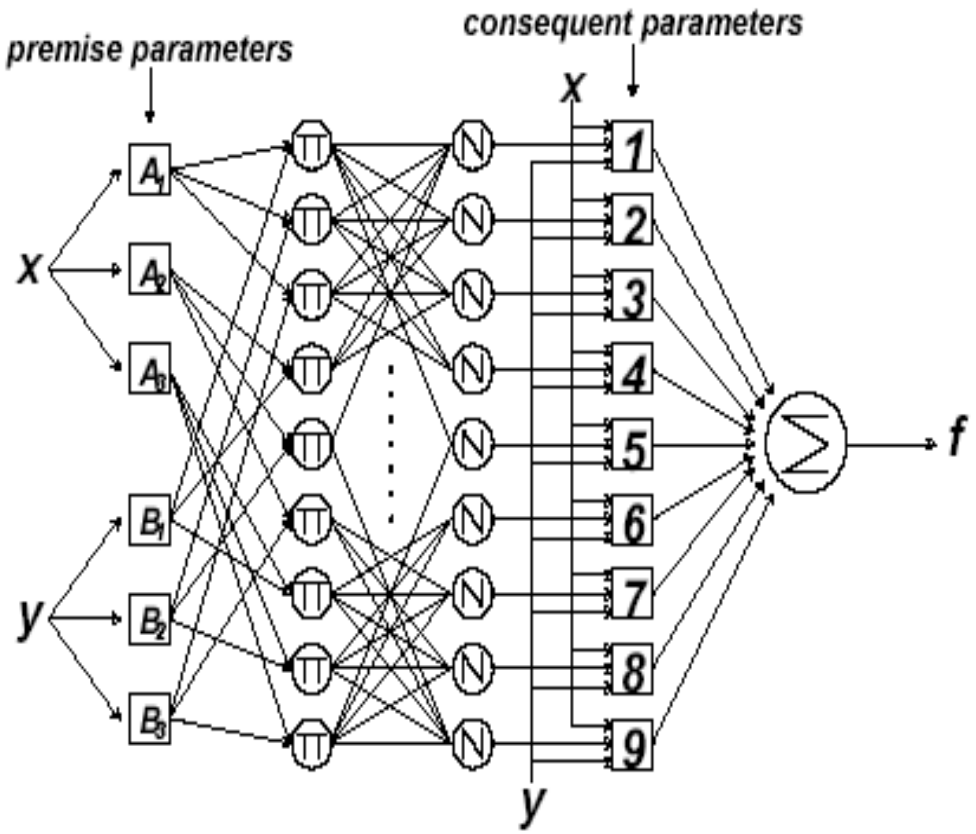
$$O_1^5 = overall\ output = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

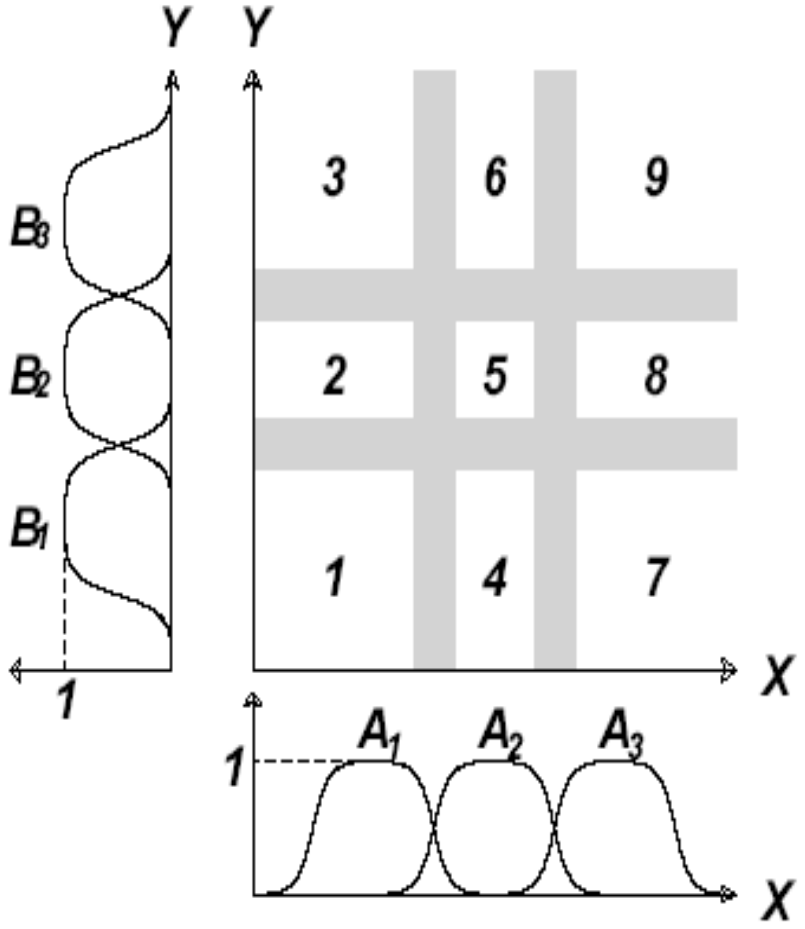
Warstwa bez parametrów. Wyjściem jest sygnał wyostrzony

ANFIS - 2 wejścia, 9 reguł



ANFIS - 2 wejścia, 9 reguł





Hybrydowe uczenie układu ANFIS

- Faza obliczeń do przodu: *forward pass*
  - **dobierane są parametry warstwy wnioskowania metodą najmniejszych kwadratów**
- Faza propagacji błędu - *backward pass* - **dobierane są parametry warstwy przesłanek metodą gradientową.**

-	forward pass	backward pass
premise parameters	fixed	gradient descent
consequent parameters	least squares estimate	fixed
signals	node outputs	error rates

Modelowanie 2-wejściowej funkcji nieliniowej

$$z = sinc(x, y) = \frac{sin(x)}{x} \times \frac{sin(y)}{y}$$

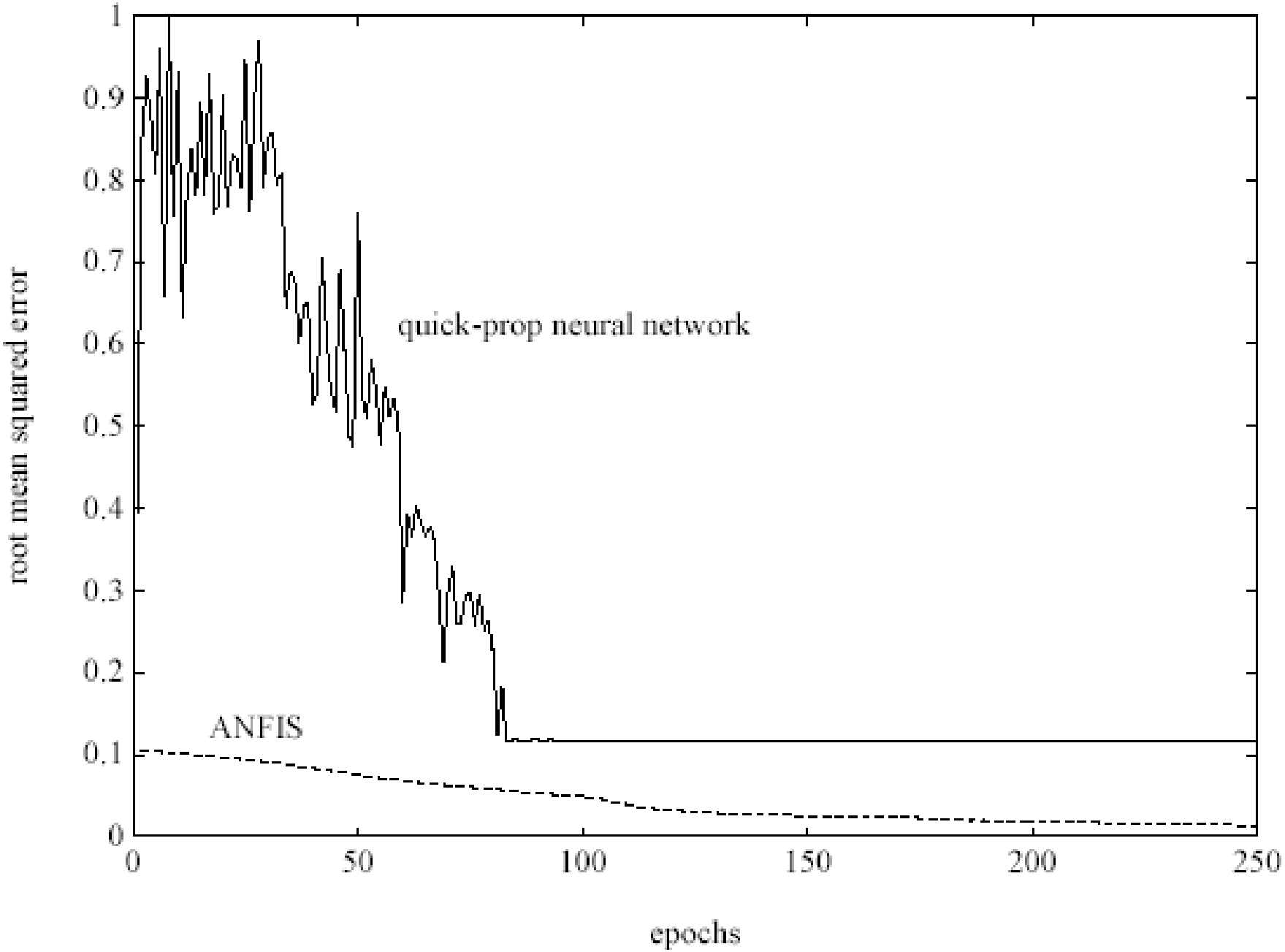
Zakres wejść: -10...+10, -10...+10

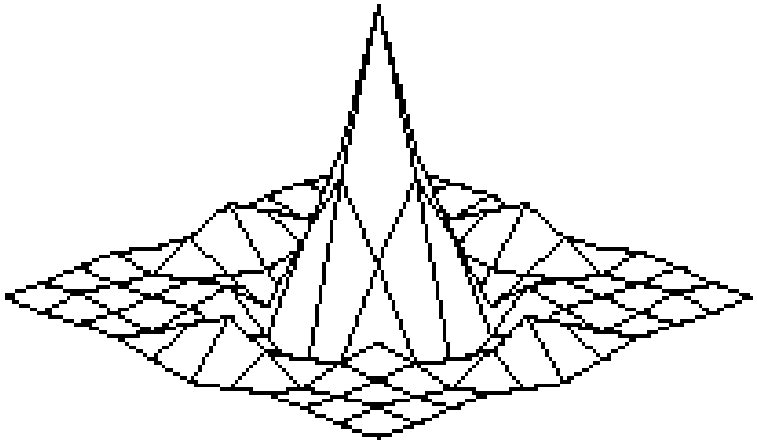
121 par uczących

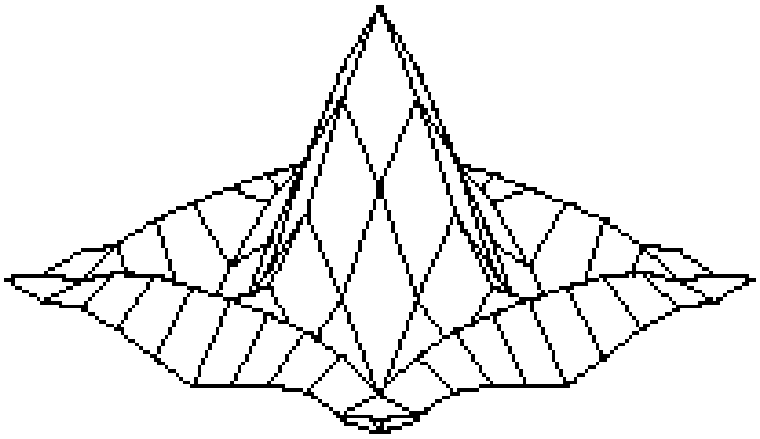
ANFIS: 16 reguł, 4 - funkcje przynależności dla każdego wejścia

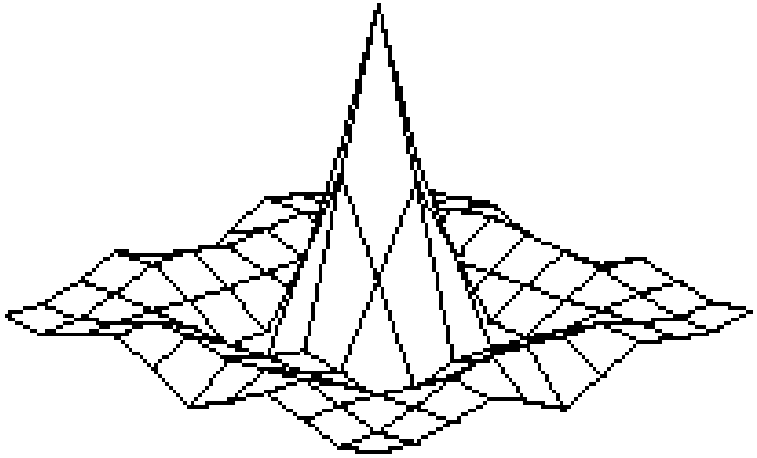
Łącznie 72 parametry - 24 przesłanki i 48 wnioskowania

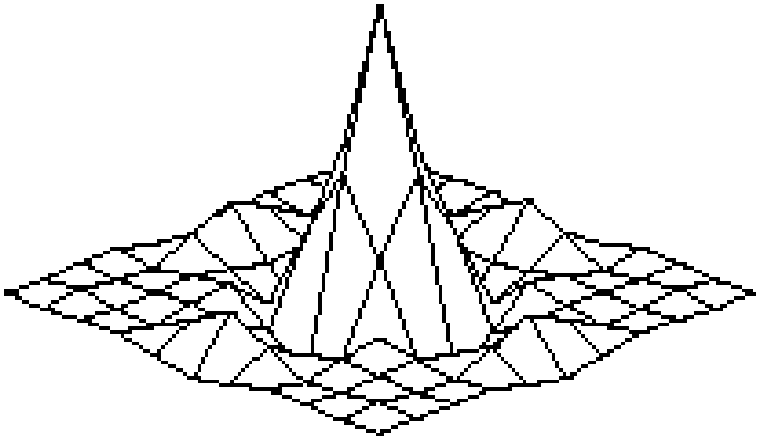
Sieć neuronowa - 2-18-1 - 73 parametry (wagi i bias)

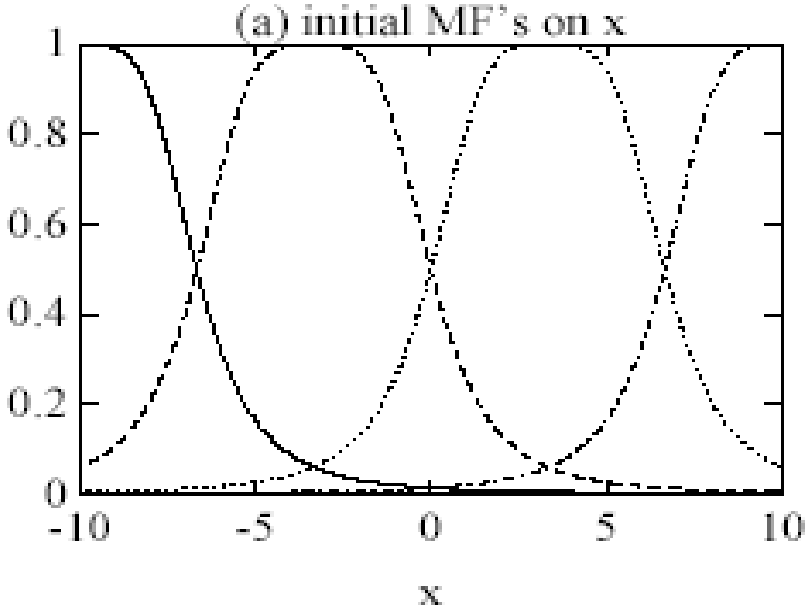

root mean squared error
epochs
quick-prop neural network
ANFIS
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
50
100
150
200
250

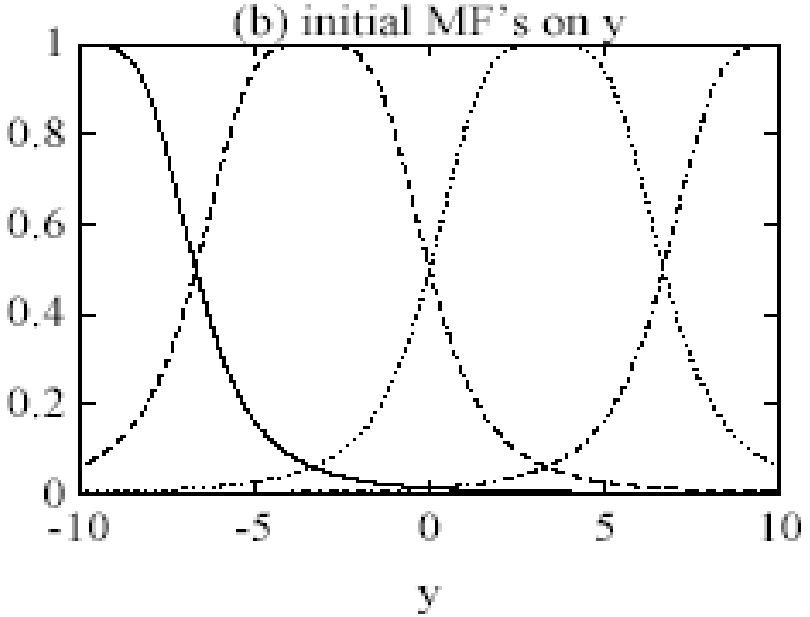


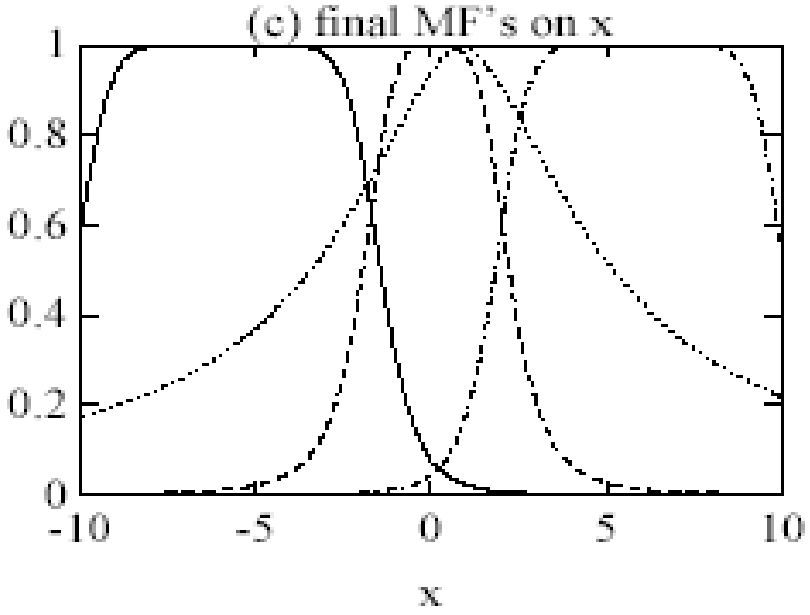


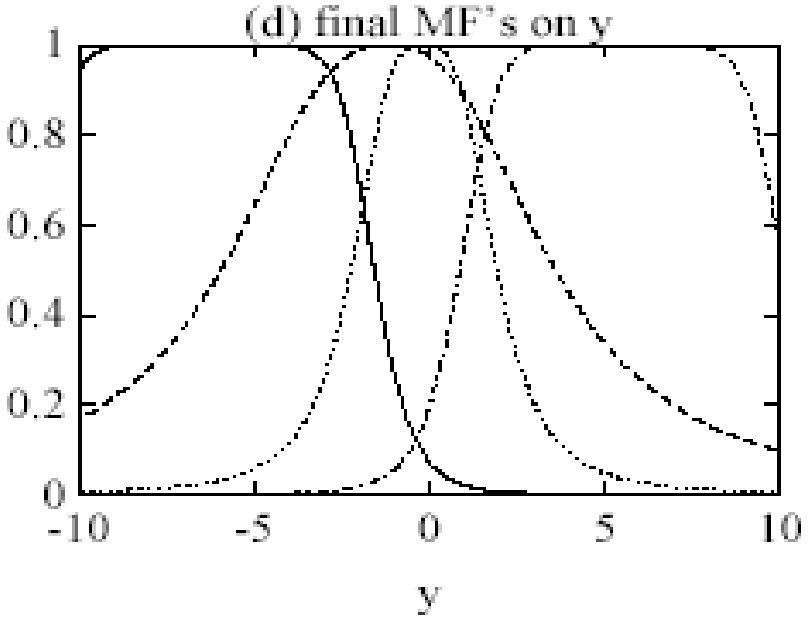





(a) initial MF's on x
1
0.8
0.6
0.4
0.2
0
-10
-5
0
5
10
x


(b) initial MF's on y
1
0.8
0.6
0.4
0.2
0
-10
-5
0
5
10
y


(c) final MF's on x
1
0.8
0.6
0.4
0.2
0
-10
-5
0
5
10
x


(d) final MF's on y
1
0.8
0.6
0.4
0.2
0
-10
-5
0
5
10
y

Modelowanie 3-wejściowej funkcji nieliniowej

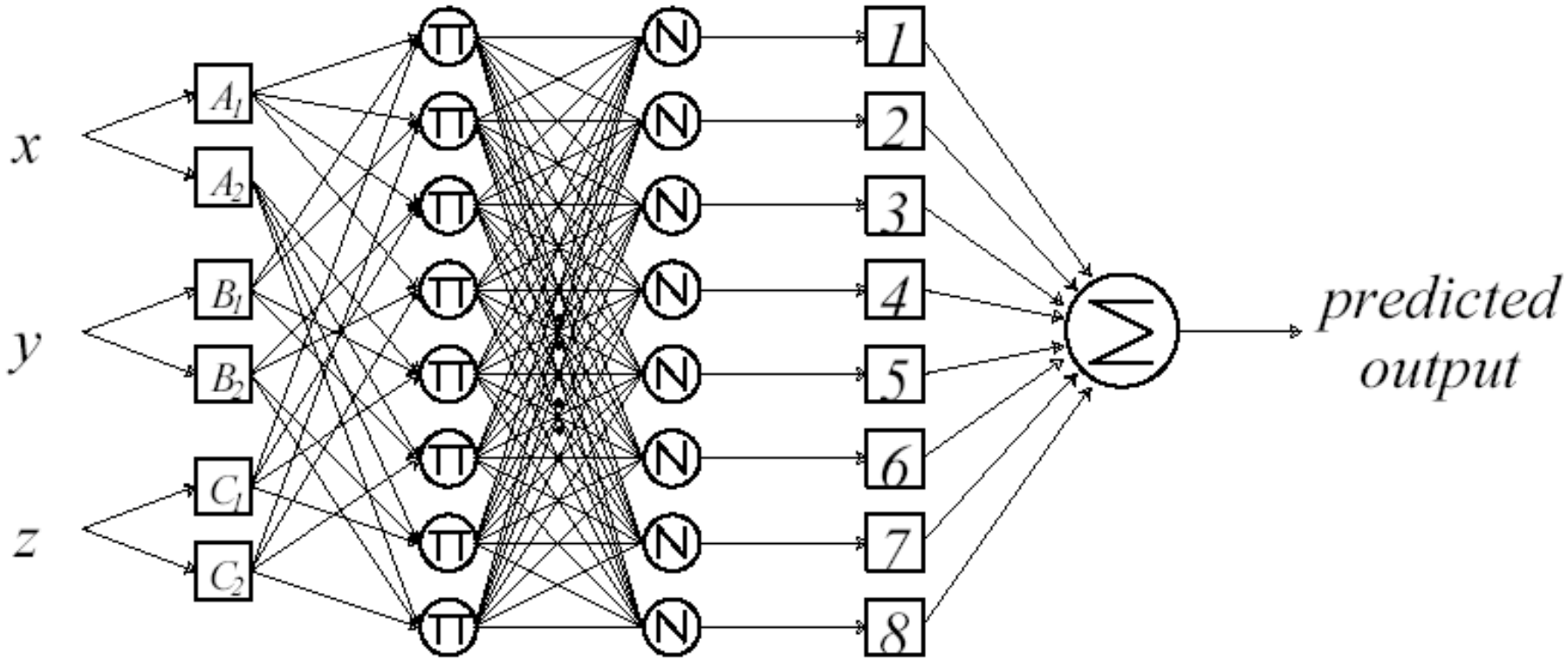
$$output = (1 + x^{0.5} + y^{-1} + z^{-1.5})^2$$

Zakres wejść: 1...6, 1...6, 1...6,

216 zestawów uczących

ANFIS: 8 reguł, 2 - funkcje przynależności dla każdego wejścia

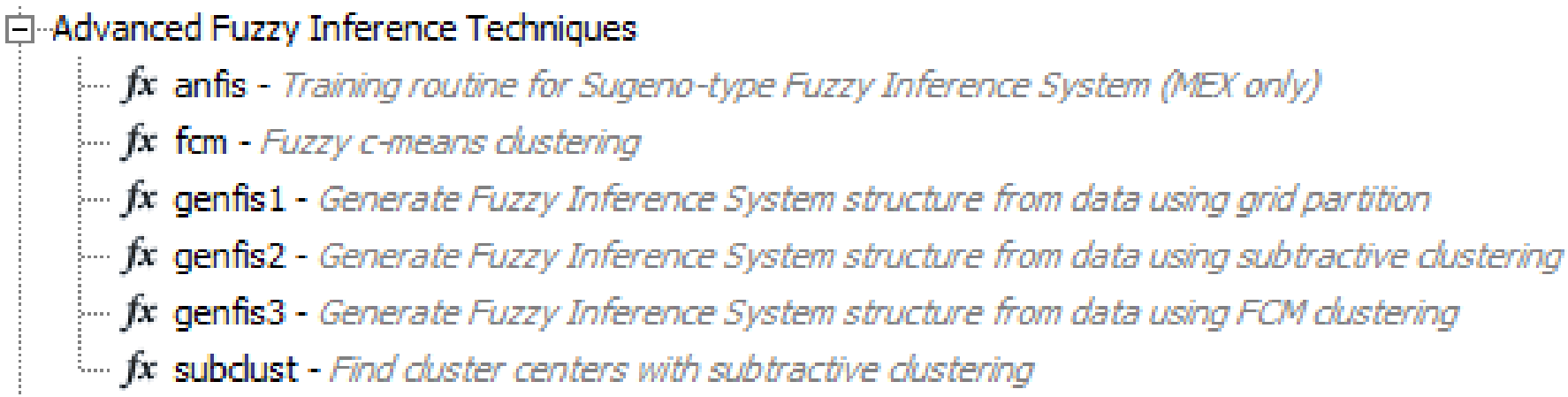
Modelowanie 3-wejściowej funkcji nieliniowej



ANFIS w Matlabie

- Fragment Fuzzy Logic Toolbox
- Dostępne podobne toolboxy, na zasadzie freeware
- Cel : zastosowanie układu rozmytego w przypadku, gdy dostępny jest zestaw danych (wejście/wyjście) układu.
- Nie znana jest struktura układu, nie ma reguł eksperckich
- Wykorzystywane: wsteczna propagacja i metoda najmniejszych kwadratów

Funkcje w Matlabie



Postać funkcji ANFIS

[fis,error] = anfis(trnData,initFis,trnOpt,dispOpt,chkData,optMethod)

Gdzie:

- fis – nauczony system FIS
- error – błąd uzyskany podczas uczenia
- trnData – zbiór danych uczących
- initFis – zainisjowana struktura FIS przez algorytm genfis1 I genfis2
- trnOpt – opcje procesu uczenia – wektor o 5 elementach zawierający w poszczególnych komórkach wartości opisujące parametry (w przypadku potrzeby użycia wartości domyślnych należy wpisad NaN):
- trnOpt(1) – liczba epok uczenia, im więcej tym dłuzej system będzie uczony, ale będzie miał mniejszy błąd. Typowa wartośd to 100, domyślna wart. 10
- trnOpt(2) - maksymalna dopuszczalna wartośd błędu, domyślna wart. 0
- trnOpt(3) – początkowy rozmiar kroku uczenia – domyslna wart. 0.01
- trnOpt(4) – rozmiar zmniejszania kroku uczenia – domyslna wart. 0.9
- trnOpt(5) – rozmiar zwiększania kroku uczenia – domyslna wart. 1.1
- dispOpt – opcje wyświetlania wyników cząstkowych na ekranie (na potrzeby lab. nieistotne)
- chkData – zbiór danych testowych umozliwiający ocenę przed przeuczeniem system (na potrzeby lab. nieistotne)
- optMethod – sposób optymalizacji funkcji przynależności, możliwe opcje to 0 lub 1, gdzie 1 – optymalizacja hybrydowa, 0 – optymalizacja w oparciu o alg. Wstecznej propagacji błędu.

Nauczanie

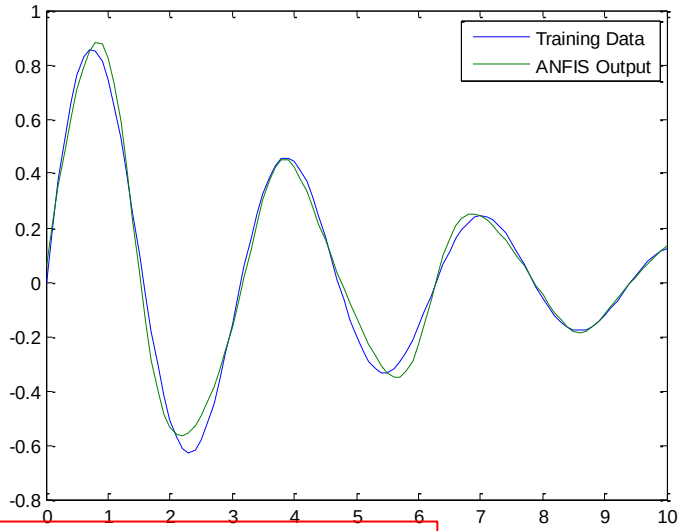
ANFIS uses a hybrid learning algorithm to identify the membership function parameters of single-output, Sugeno type fuzzy inference systems (FIS). **A combination of least-squares and backpropagation gradient descent methods are used for training FIS membership function parameters to model a given set of input/output data.**

[FIS,ERROR] = ANFIS(TRNDATA) tunes the FIS parameters using the input/output training data stored in TRNDATA. For an FIS with N inputs, TRNDATA is a matrix with N+1 columns where the first N columns contain data for each FIS input and the last column contains the output data. ERROR is the array of root mean square training errors (difference between the FIS output and the training data output) at each epoch. ANFIS uses GENFIS1 to create a default FIS that is used as the starting point for ANFIS training.

[FIS,ERROR] = ANFIS(TRNDATA,INITFIS) uses the FIS structure, INITFIS as the starting point for ANFIS training.

Przykład

```
x = (0:0.1:10)';
    y = sin(2*x)./exp(x/5);
    epoch_n = 20;
    in_fis  = genfis1([x y],5,'gbellmf');
    out_fis = anfis([x y],in_fis,epoch_n);
    plot(x,y,x,evalfis(x,out_fis));
    legend('Training Data','ANFIS Output');
```



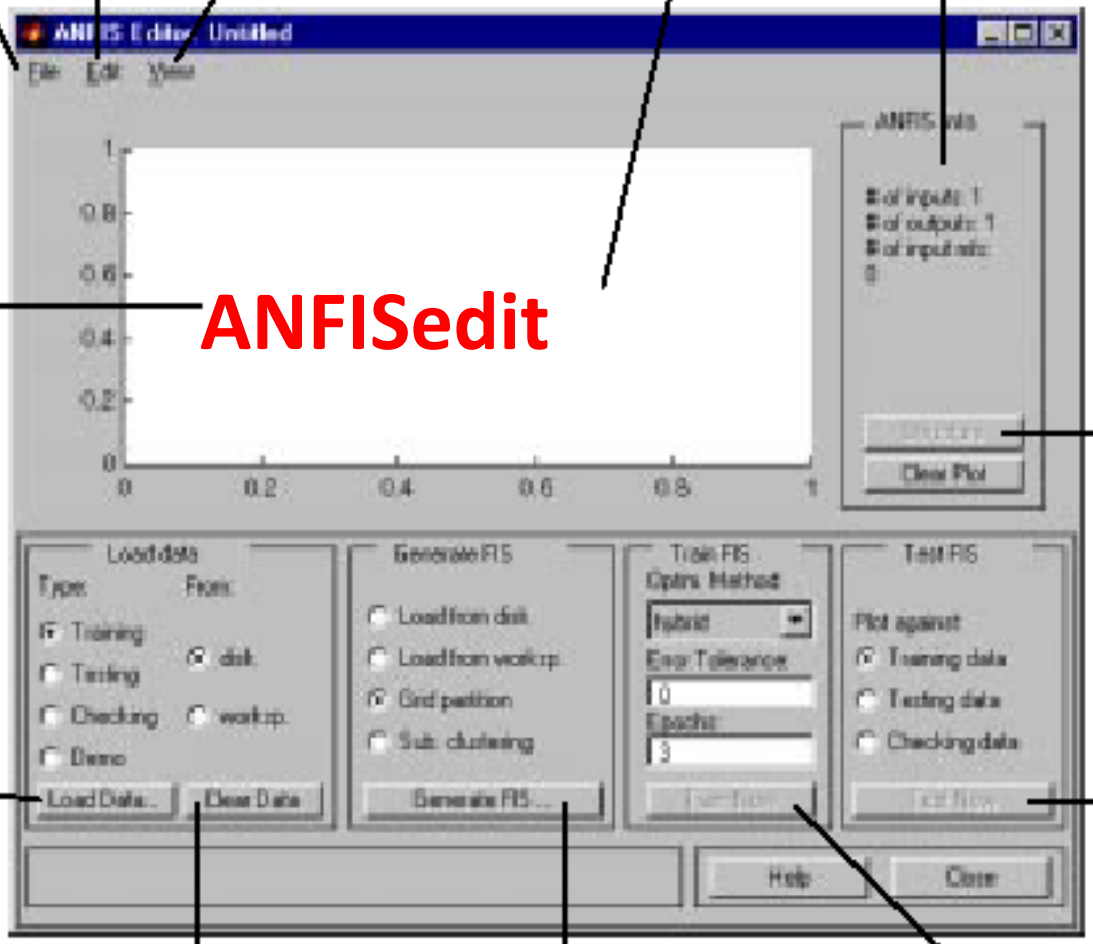
Start training ANFIS ...

1	0.0694086
2	0.0680259
3	0.066663
4	0.0653198
5	0.0639961

ANFIS info:

Number of nodes: 24
Number of linear parameters: 10
Number of nonlinear parameters: 15
Total number of parameters: 25
Number of training data pairs: 101
Number of checking data pairs: 0
Number of fuzzy rules: 5

```
Step size increases to 0.011000 after epoch 5.
   6      0.0626917
   7      0.0612787
   8      0.0598881
   9      0.0585193
```



Load or save a fuzzy Sugeno system, or open new Sugeno system.

Undo

Open or edit a FIS with any of the other GUIs.

Plot region

Status of the number of inputs, outputs, input membership functions, and output membership functions

Testing data appears on the plot in blue as · ·

Training data appears on the plot in blue as o o

Checking data appears on the plot in blue as + +

FIS output appears on the plot in red as * *

After you generate or load a FIS, this button allows you to open a graphical representation of its input/output structure.

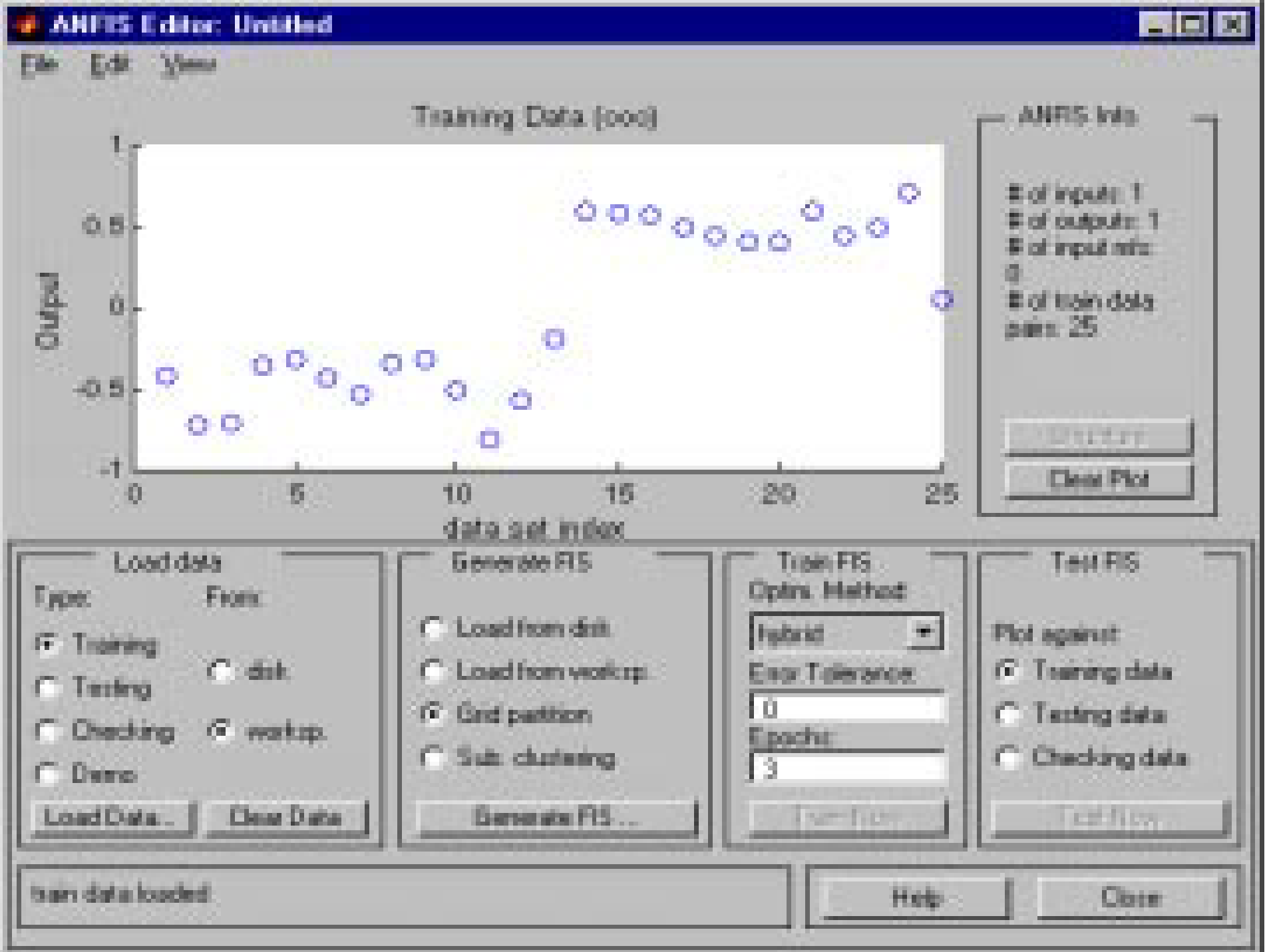
Load either training, testing, or checking data from disk or workspace, or load demo data. Data appears in the plot region.

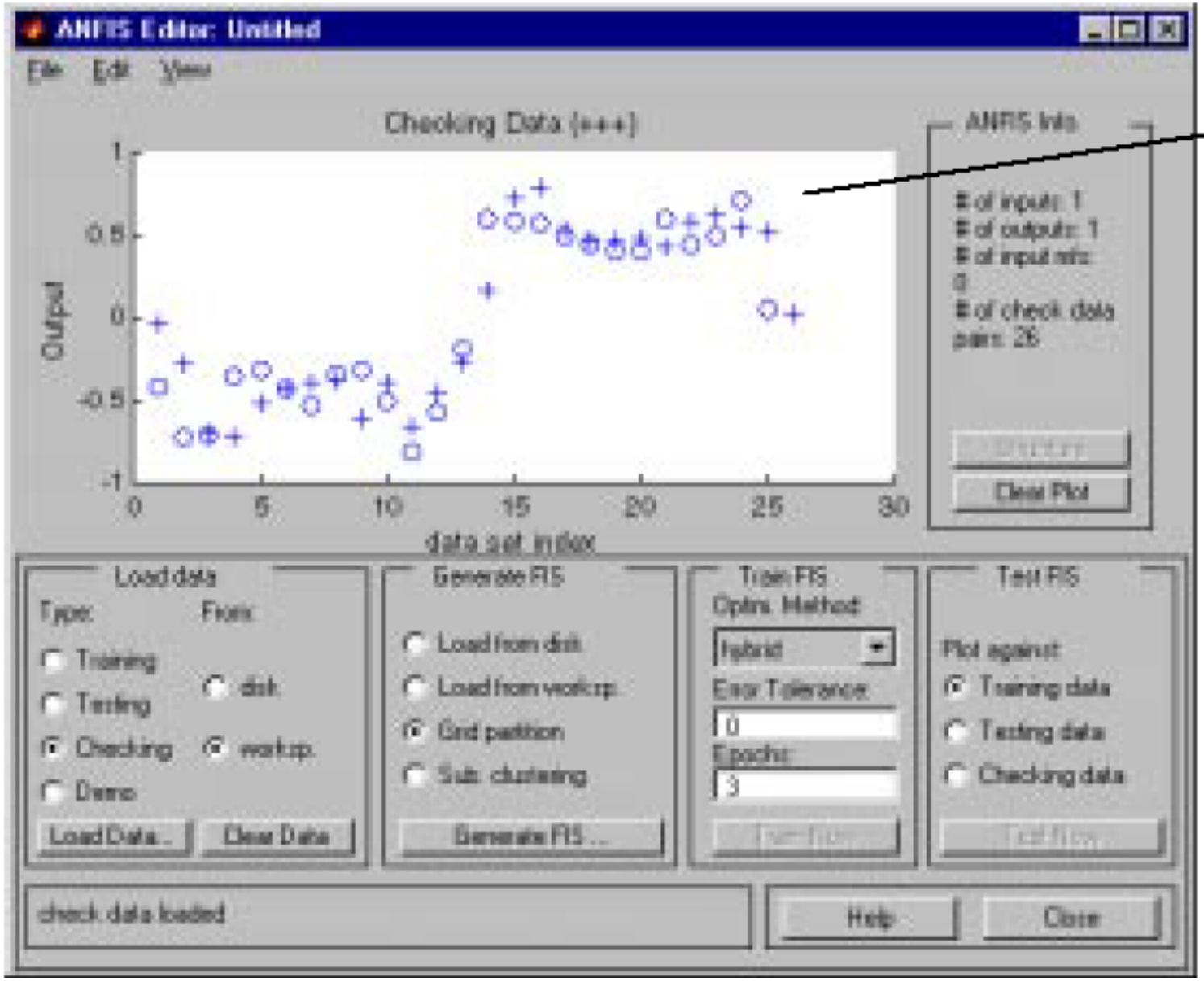
Test data against the FIS model. The plot appears in the plot region.

Clear Data unloads the data set selected under **Type:** and clears the plot region.

Load FIS or generate FIS from loaded data using your chosen number of MFs and rules or fuzzy.

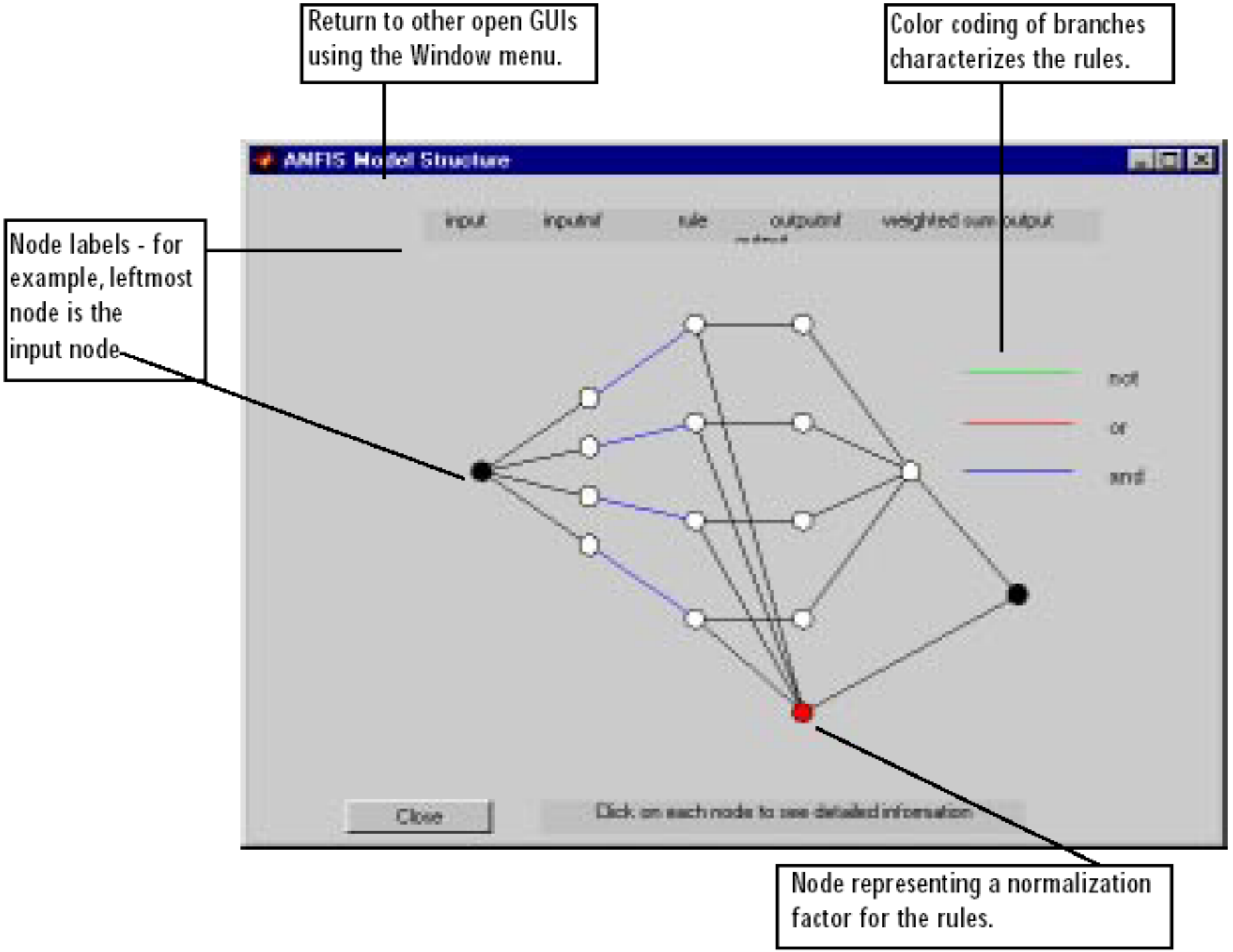
Train FIS after setting optimization method, error tolerance, and number of epochs. This generates error plots in the plot region.

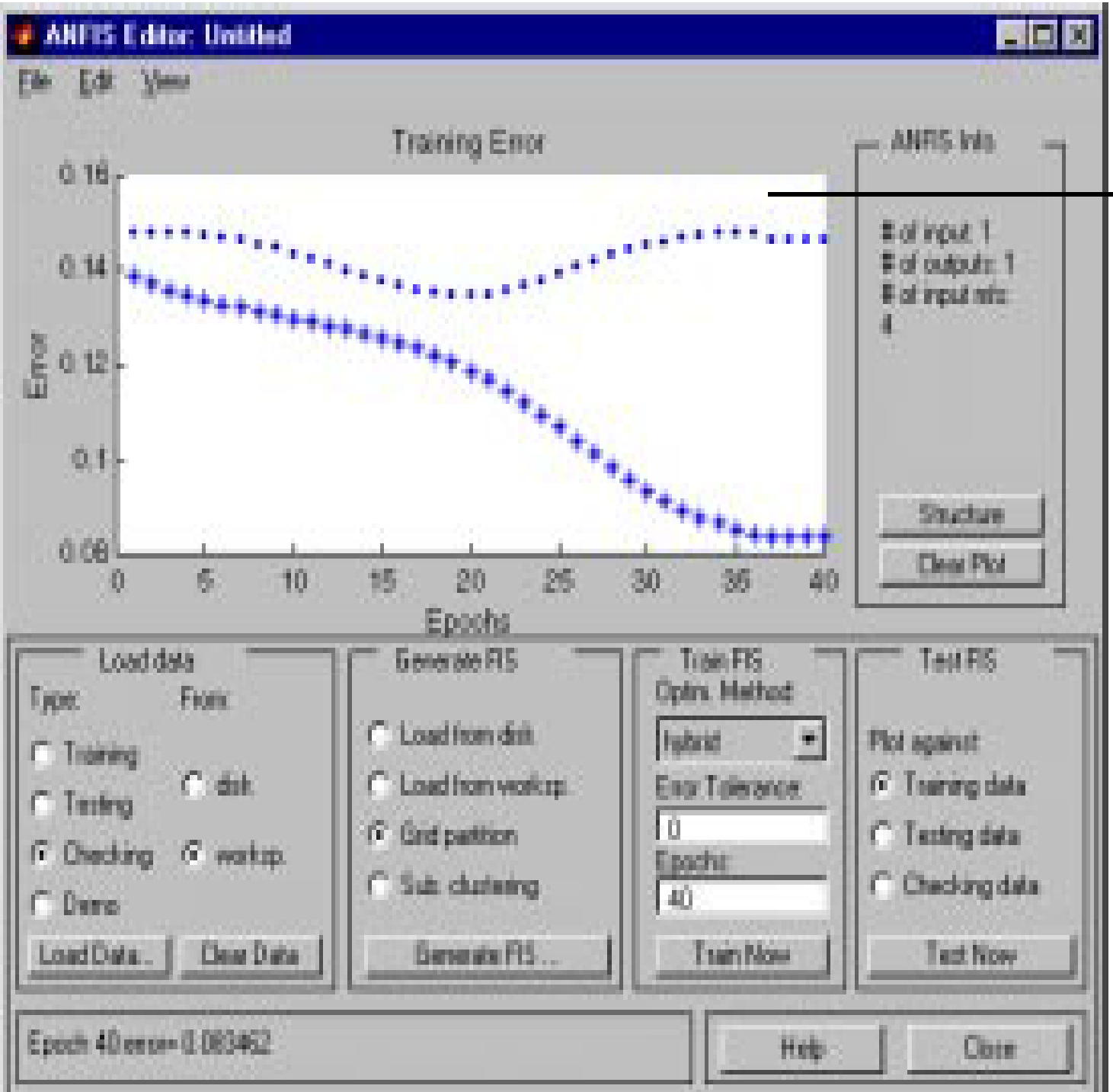

ANFIS Editor: Untitled
File Edit View
Training Data (ooo)
Output
data set index
ANFIS Info
# of inputs: 1
# of outputs: 1
# of input mfs: 0
# of train data pairs: 25
Structure
Close Plot
Load data
Type:
Training
Testing
Checking
Demo
From:
disk
workspace
Load Data...
Clear Data
Generate FIS
Load from disk
Load from workspace
Grid partition
Sub. clustering
Generate FIS...
Train FIS
Optim. Method:
Hybrid
Error Tolerance:
0
Epochs:
3
Train Now
Test FIS
Plot against:
Training data
Testing data
Checking data
Test Now
Train data loaded.
Help
Close



+++ Checking data

ooo Training data


Return to other open GUIs using the Window menu.
Color coding of branches characterizes the rules.
ANFIS Model Structure
input
inputmf
rule
outputmf
weighted sum output
Node labels - for example, leftmost node is the input node
not
or
and
Close
Click on each node to see detailed information
Node representing a normalization factor for the rules.



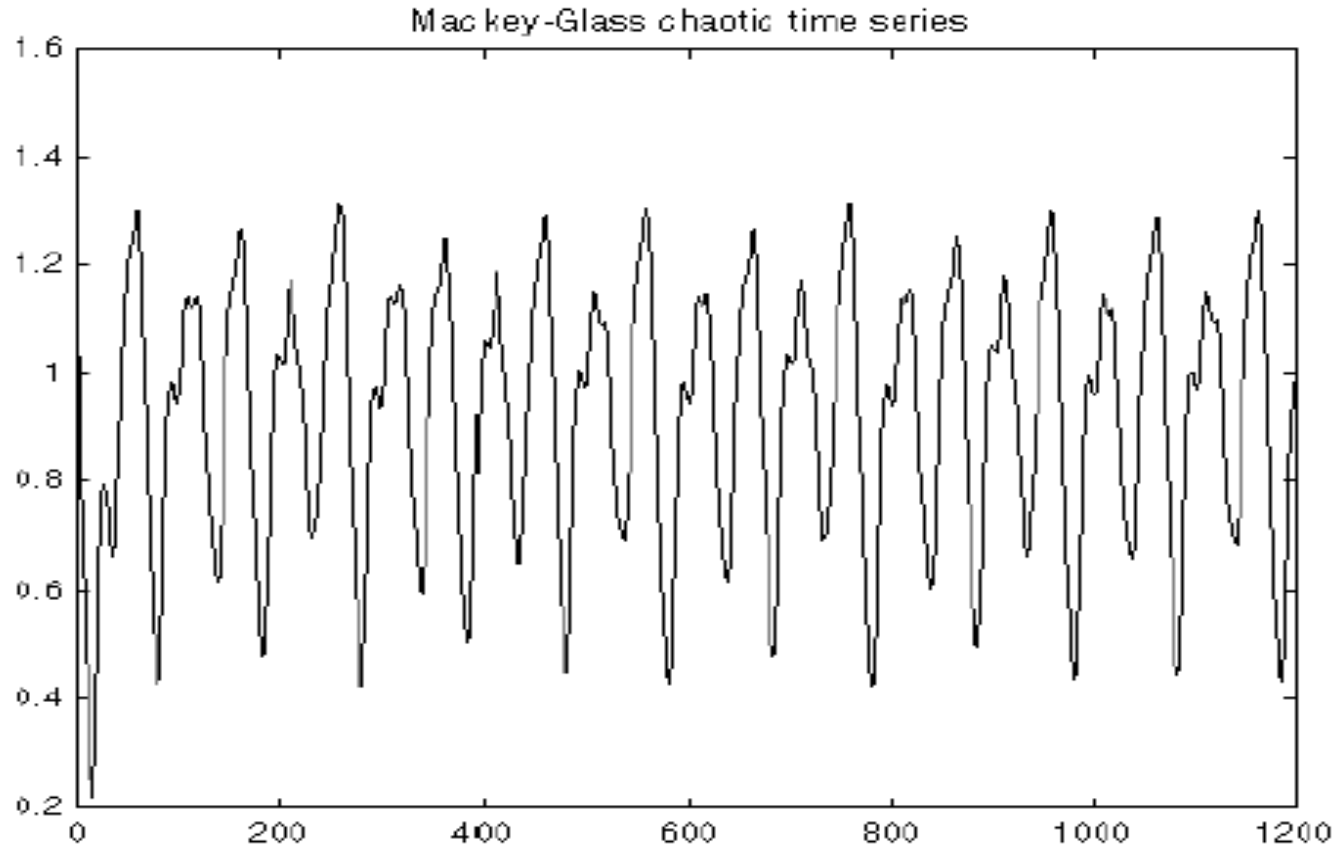
*** Training error

... Checking error

mgtsdemo - dane chaotyczne

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \qquad s(t) = x(t+6)$$

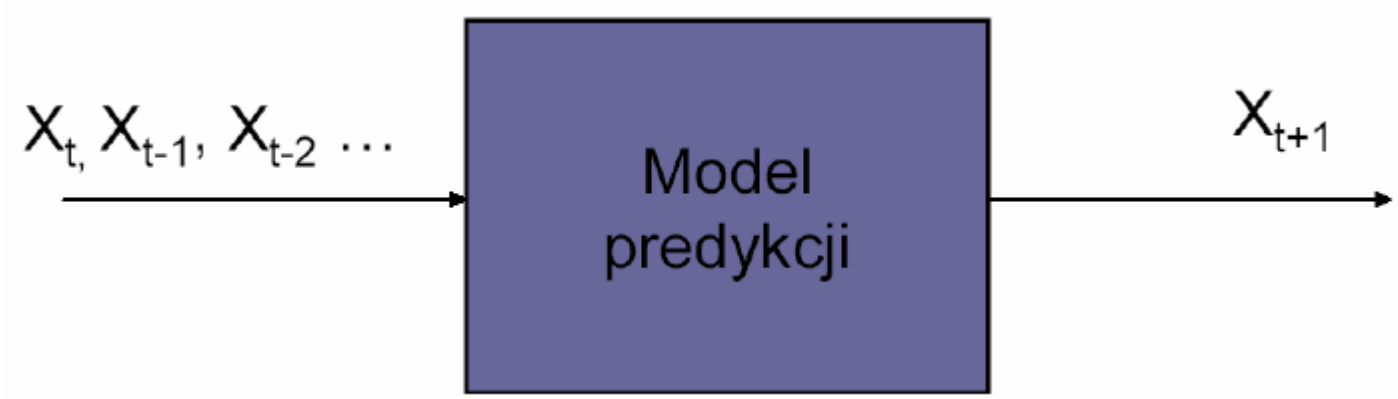
$$w(t) = [x(t-18)\ \ x(t-12)\ \ x(t-6)\ \ x(t)]$$



Prognoza czasowych przebiegów sygnałów

Prognoza czasowych przebiegów sygnałów jest oparta na ich wcześniejszych wartościach Dlatego też niezbędne jest zebranie i analiza zgromadzonych danych. Podczas analizy tych danych w sposób obiektywny niezbędne jest posiadanie danych zawierających maksymalną ilość informacji oraz adekwatną liczbę pomiarów danych wejściowych. Przyszłe wartości przebiegów czasowych $x(t)$ mogą być przedstawione jako funkcja minionych wartości $x(t-1), x(t-2), .., x(t-\varphi)$

$$x(t+\tau) = f(x(t-1), x(t-2), .., x(t-\varphi))$$



Szeregi czasowe

Szereg czasowy – ciąg obserwacji pewnego zjawiska w kolejnych jednostkach czasu.

Szeregi czasowe są podstawą analizy **dynamiki zjawisk.**

Cel analizy szeregów czasowych
Zbudowanie modelu pewnego
zjawiska/procesu w oparciu
o obserwowane zmiany w czasie pewnych
mierzalnych
wielkości opisujących ten proces.

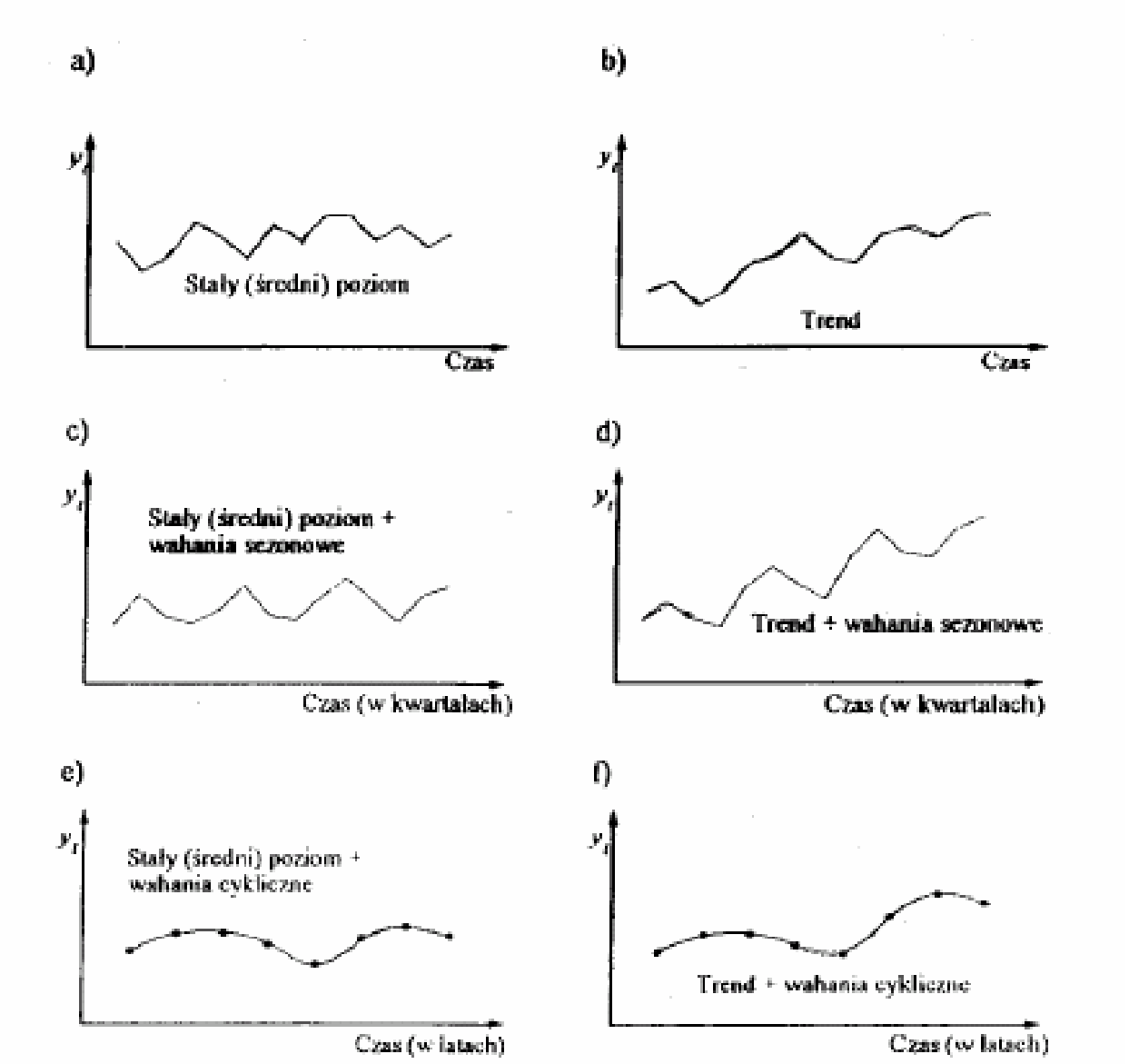
Podstawowa struktura szeregów czasowych

Stały (przeciętny) poziom zmiennej.
Trend (tendencja rozwojowa) – reprezentuje ogólny kierunek rozwoju zjawiska (systematyczne zmiany, jakim podlega zjawisko); rozróżnia się, np., trend liniowy lub nieliniowy.
Składowa okresowa (wahania okresowe / regularne odchylenia od tendencji rozwojowej) – składnik powtarzający się cyklicznie.
Szum (zakłócenia, wahania przypadkowe).

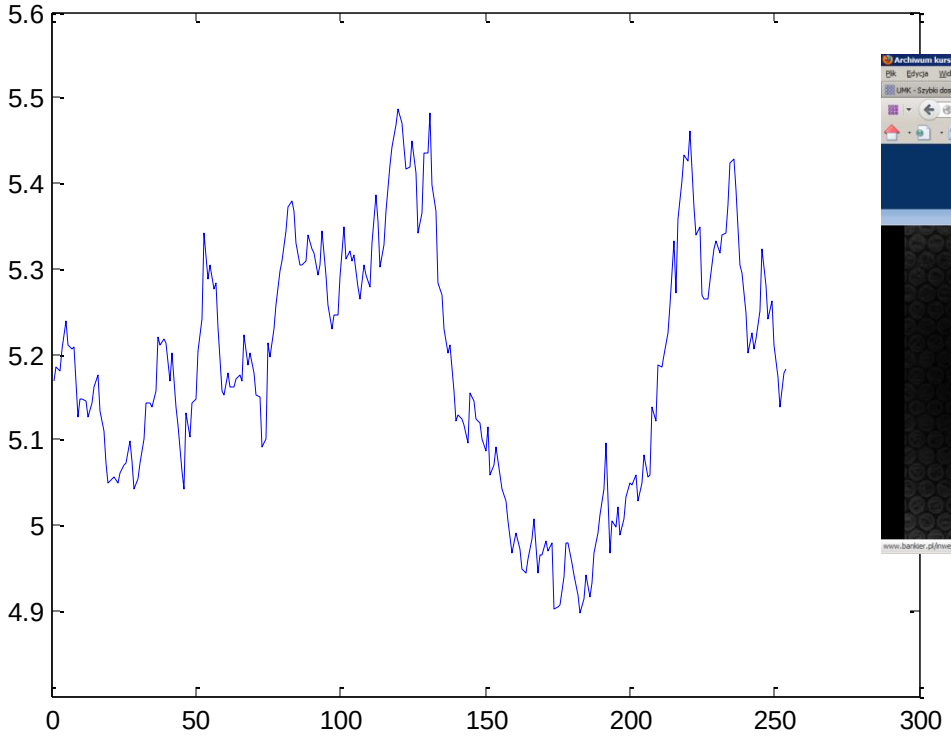
Składowa okresowa

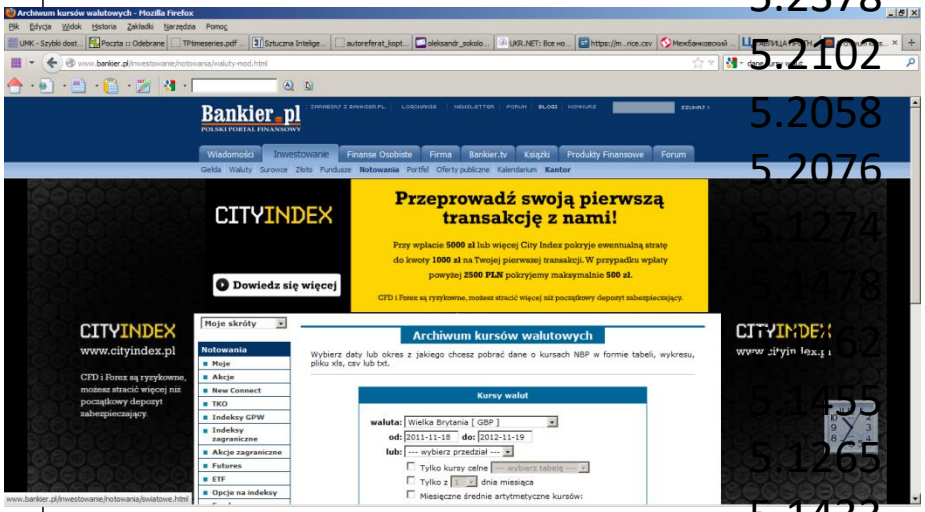
Składowa okresowa może wystąpić w postaci wahań:
cyklicznych - długookresowe, rytmiczne wahania (cykl koniunkturalny gospodarki, cykl rozwoju populacji nabywców danego produktu, itp.),
sezonowych – krótkookresowe do 1 roku, odzwierciedlają wpływ zachowań wynikający z „kalendarza” (np. rytm pracy w skali tygodnia, dnia, pory roku, świąt, ...).

Przykłady


a)
Stały (średni) poziom
Czas
b)
Trend
Czas
c)
Stały (średni) poziom +
wahania sezonowe
Czas (w kwartałach)
d)
Trend + wahania sezonowe
Czas (w kwartałach)
e)
Stały (średni) poziom +
wahania cykliczne
Czas (w latach)
f)
Trend + wahania cykliczne
Czas (w latach)

Prognoza kursów waluty



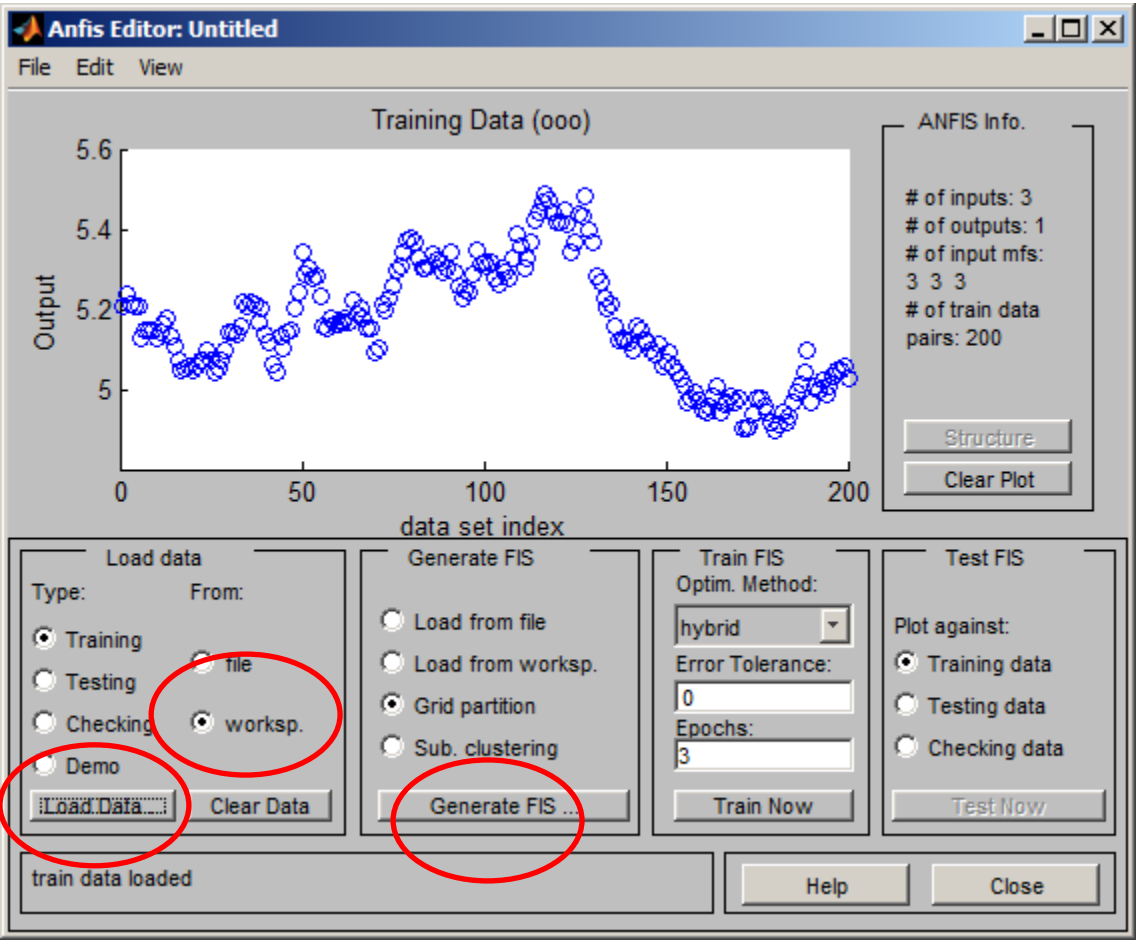


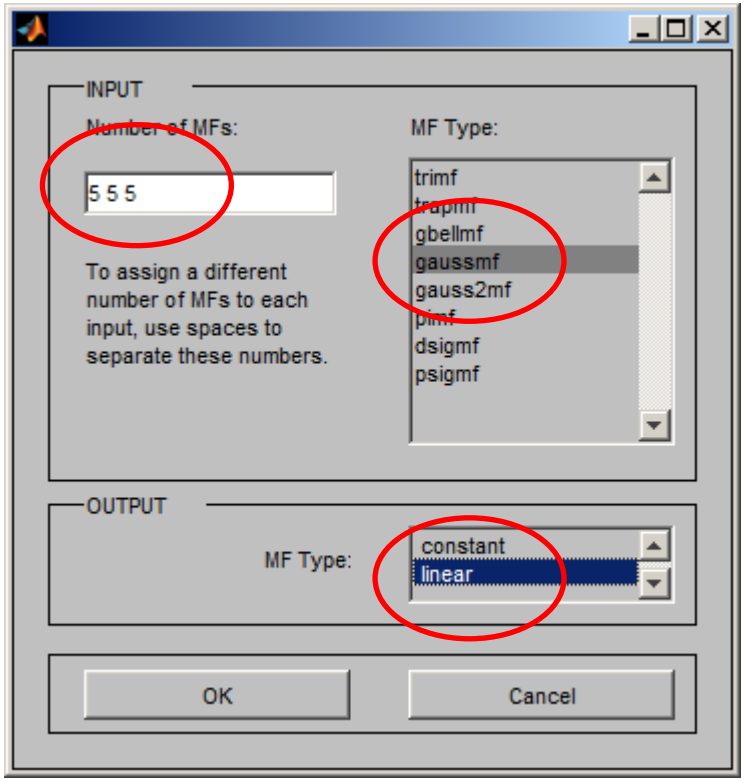
http://www.bankier.pl/inwestowanie/notowania/waluty-mod.html

5.1682
5.1859
5.1794
5.2059
5.2378
5.2102
5.2058
5.2076
5.1274
5.1478
5.162
5.1455
5.1265
5.1433
5.1605
5.1751
5.1341
5.1098
5.0714
5.0493
5.0532

Przygotowanie danych

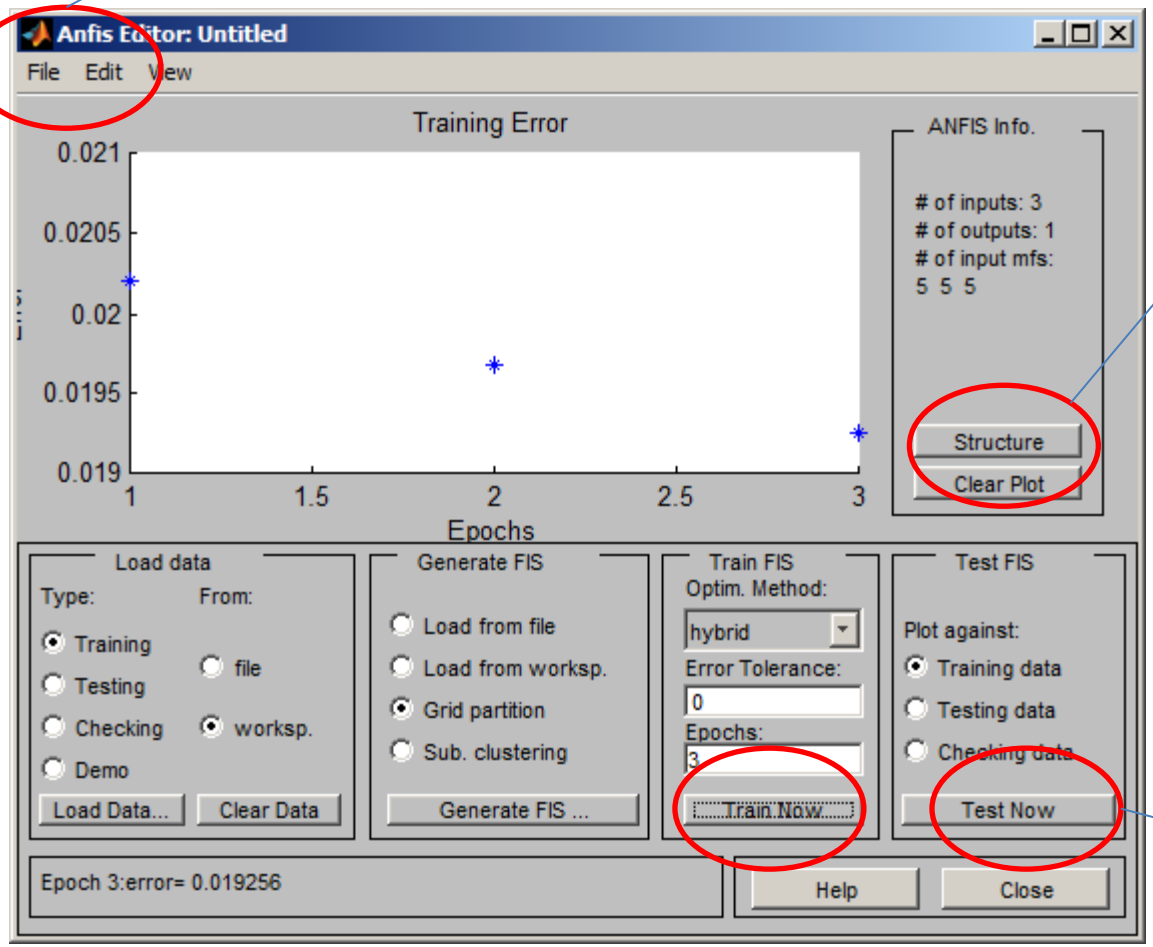
y=[x(1:200) x(2:201) x(3:202) x(4:203)]

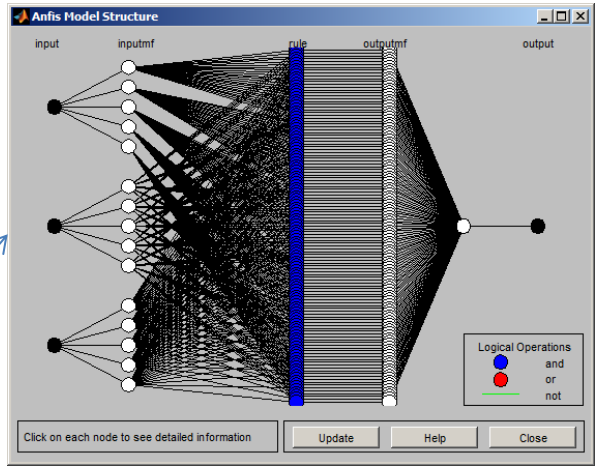


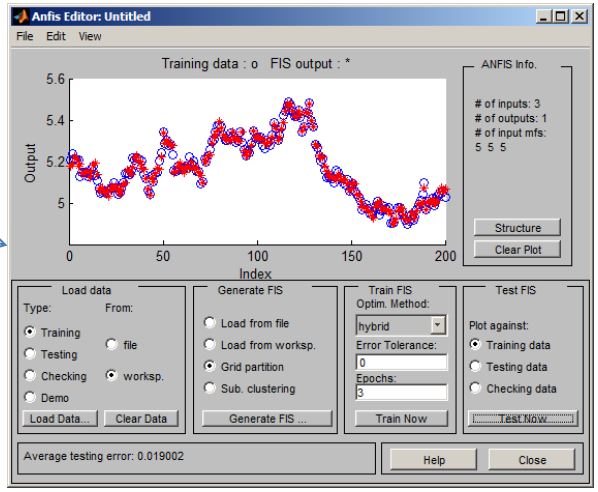


Nauczanie

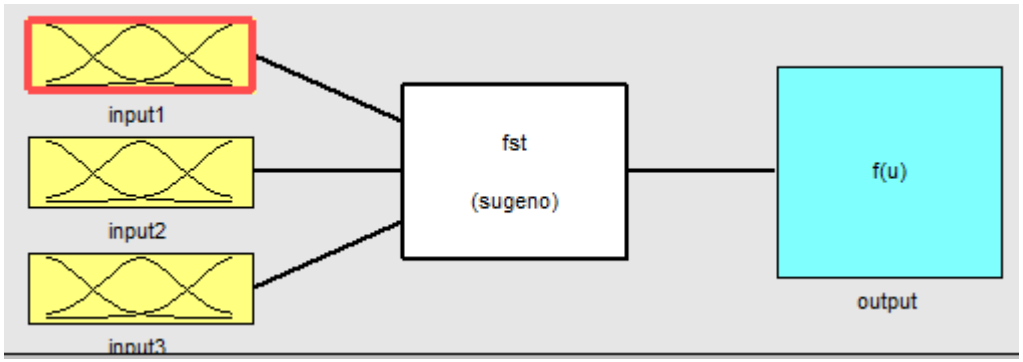
Export FIS

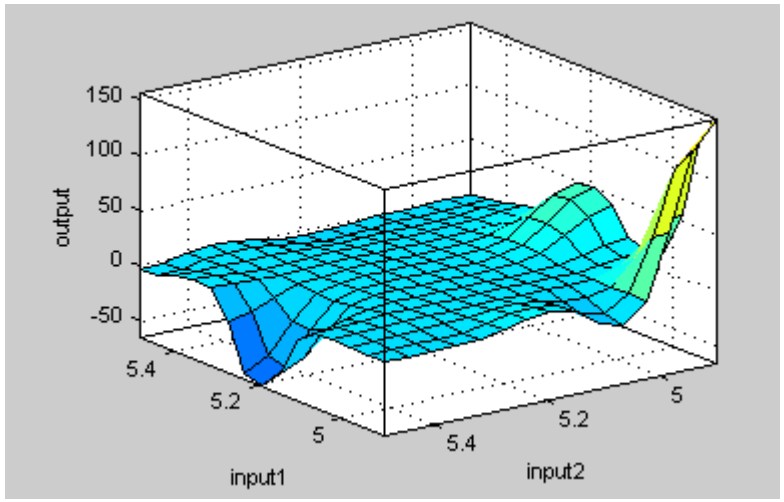






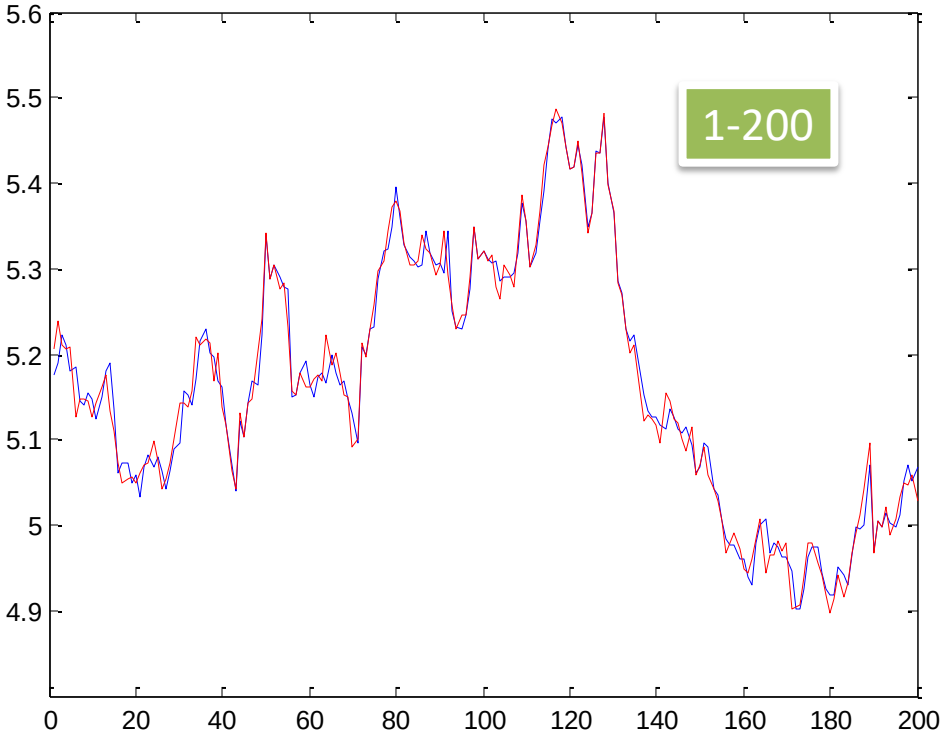
FIS model



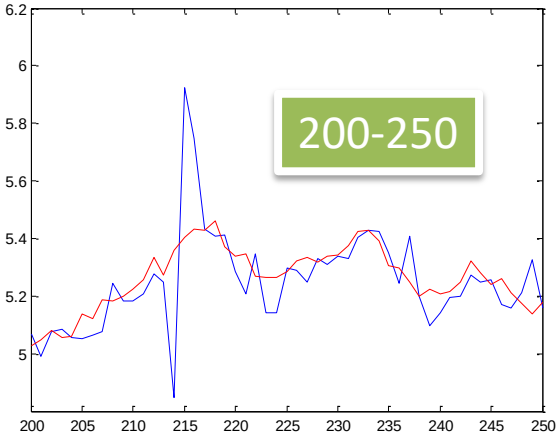


116. If (input1 is in1mf5) and (input2 is in2mf4) and (input3 is in3mf1) then (output is out1mf116) (1)
117. If (input1 is in1mf5) and (input2 is in2mf4) and (input3 is in3mf2) then (output is out1mf117) (1)
118. If (input1 is in1mf5) and (input2 is in2mf4) and (input3 is in3mf3) then (output is out1mf118) (1)
119. If (input1 is in1mf5) and (input2 is in2mf4) and (input3 is in3mf4) then (output is out1mf119) (1)
120. If (input1 is in1mf5) and (input2 is in2mf4) and (input3 is in3mf5) then (output is out1mf120) (1)
121. If (input1 is in1mf5) and (input2 is in2mf5) and (input3 is in3mf1) then (output is out1mf121) (1)
122. If (input1 is in1mf5) and (input2 is in2mf5) and (input3 is in3mf2) then (output is out1mf122) (1)
123. If (input1 is in1mf5) and (input2 is in2mf5) and (input3 is in3mf3) then (output is out1mf123) (1)
124. If (input1 is in1mf5) and (input2 is in2mf5) and (input3 is in3mf4) then (output is out1mf124) (1)
125. If (input1 is in1mf5) and (input2 is in2mf5) and (input3 is in3mf5) then (output is out1mf125) (1)

Wyniki prognozy

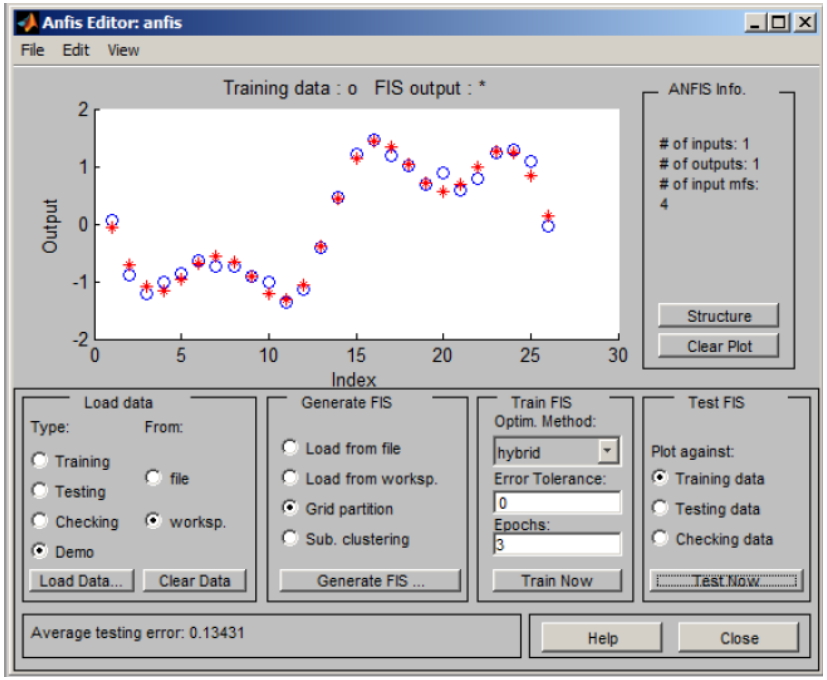



```
pr=[];
M=1;
N=200;
for i=M:N

pr=[pr,evalfis(x(i:i+2),fst)];
end;
plot(M:N,pr);
hold on;
plot(M:N,x(M+3:N+3),'r')
```



Ograniczenia funkcji i interfejsu ANFIS

Interfejs i funkcja anfis działa poprawnie tylko na modelach rozmytych typu Sugeno rzędu zerowego lub pierwszego. Oznacza to, że w konkluzjach modelu mogą występować tylko stałe lub funkcje liniowe.



Inne ograniczenia:
wszystkie funkcje wyjściowe w konkluzjach muszą być tego samego typu i muszą być albo stałymi albo funkcjami liniowymi
funkcje w konkluzjach nie mogą się powtarzać, ich liczba musi być równa liczbie reguł
należy określić funkcje przynależności dla wszystkich zmiennych wejściowych
układ rozmyty musi mieć wagę jednostkową dla każdej reguły.

Materiały dodatkowe

The Algorithm

In the clustering problem, we are given a training set ${x^{(1)}, ... , x^{(m)}}$, and want to group the data into a few cohesive "clusters." Here, we are given feature vectors for each data point $x^{(i)} \in \mathbb{R}^n$ as usual; but no labels $y^{(i)}$ (making this an unsupervised learning problem). Our goal is to predict k centroids **and** a label $c^{(i)}$ for each datapoint. The k-means clustering algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

   For every i, set

   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

   $$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}.$$

   }

METODY

c:\Users\KIS-OS\OneDrive\saszasokolov\Nauka\LR\lit\Klaster\János Abonyi, Balázs Feil Cluster Analysis for Data Mining and System Identification 2007.pdf

Babuszka
c:\Users\KIS-OS\OneDrive\saszasokolov\Nauka\LR\lit\Klaster\Fuzzy-Clustering-lecture-Babuska.pdf

Algorithms

Fuzzy c-means (FCM) is a clustering method that allows each data point to belong to multiple clusters with varying degrees of membership.

FCM is based on the minimization of the following objective function

$$J_m = \sum_{i=1}^{D}\sum_{j=1}^{N} \mu_{ij}^{m} \|x_i - c_j\|^2,$$

where

- D is the number of data points.
- N is the number of clusters.
- m is fuzzy partition matrix exponent for controlling the degree of fuzzy overlap, with $m > 1$. Fuzzy overlap refers to how fuzzy the boundaries between clusters are, that is the number of data points that have significant membership in more than one cluster.
- x_i is the ith data point.
- c_j is the center of the jth cluster.
- μ_{ij} is the degree of membership of x_i in the jth cluster. For a given data point, x_i, the sum of the membership values for all clusters is one.

`fcm` performs the following steps during clustering:

1. Randomly initialize the cluster membership values, μ_{ij}.
2. Calculate the cluster centers:

$$c_j = \frac{\sum_{i=1}^{D} \mu_{ij}^{m} x_i}{\sum_{i=1}^{D} \mu_{ij}^{m}}.$$

3. Update μ_{ij} according to the following:

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{N} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}}.$$

4. Calculate the objective function, J_m.
5. Repeat steps 2–4 until J_m improves by less than a specified minimum threshold or until after a specified maximum number of iterations.

References

[1] Bezdec, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.