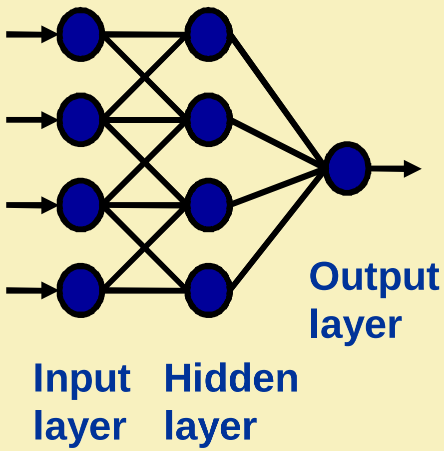


SIECI NIELINIOWE

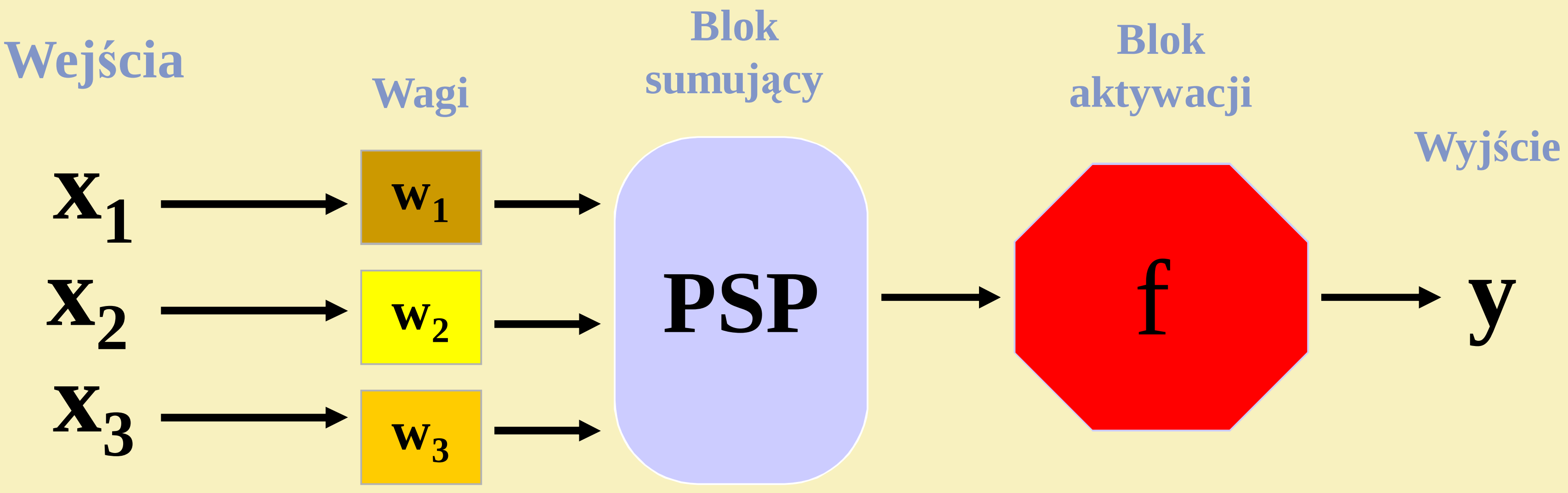
ALGORYTM WSTECZNEJ PROPAGACJI BŁĘDÓW

BP BACKPROPAGATION

Joanna Grabska- Chrząstowska

Wykłady w dużej mierze przygotowane w oparciu o materiały i pomysły
PROF. RYSZARDA TADEUSIEWICZA

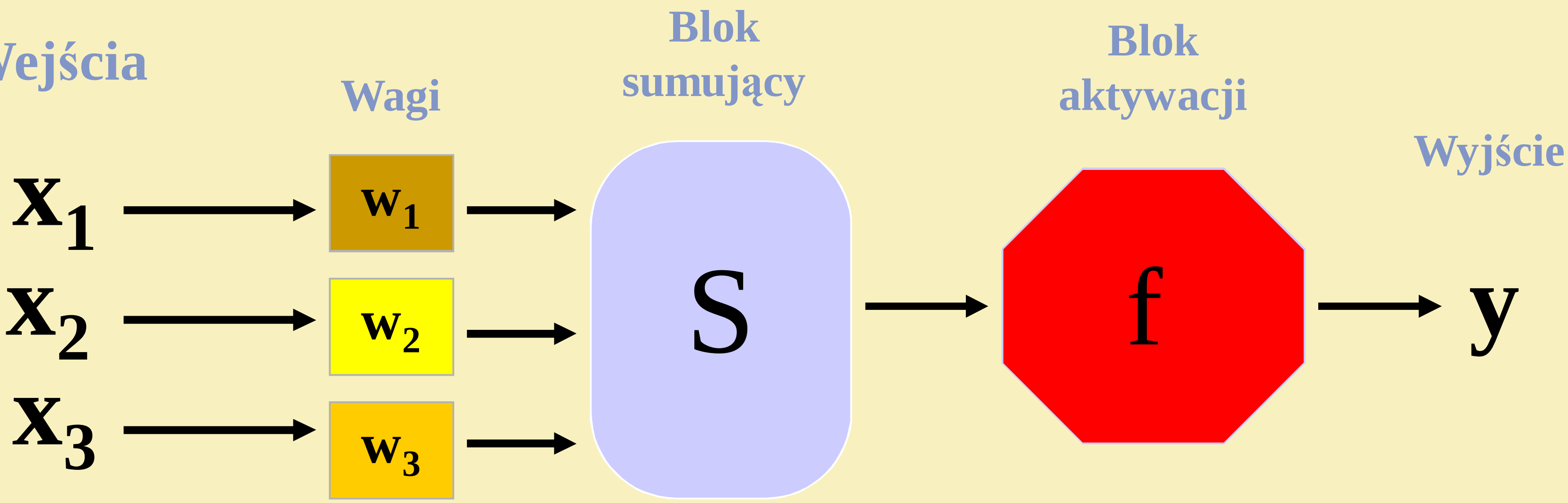
DEFINICJA NEURONU



$$y = f\ (\ PSP(\ x_i,\ w_i\ ))$$

PSP -Post **S**ynaptic **P**otential function

DEFINICJA NEURONU

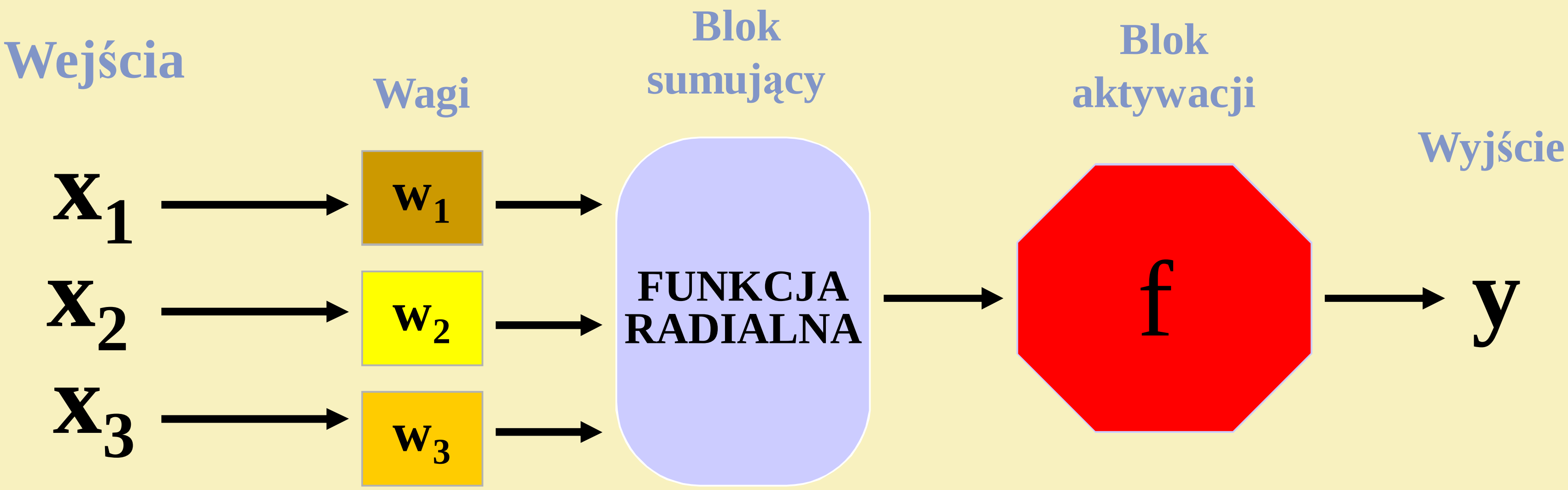


PSP - funkcja **liniowa** (iloczyn skalarny)

$$y = f\ (\ S\ (\ x_i * w_i\ ))$$

Liniowa funkcja PSP wyznacza ważoną sumę wszystkich wartości wejściowych. Ta suma następnie zostaje zmodyfikowana w taki sposób, że odejmuje się od niej wartość progową. W terminologii wektorowej można powiedzieć, że rozważana funkcja PSP jest to iloczyn skalarny wektora wag i wektora wejściowego - minus wartość progu. Neurony z liniową funkcją PSP generują liniowe funkcje dyskryminacyjne. Oznacza to, że identyczne wartości sygnału wyjściowego otrzymuje się dla sygnałów wejściowych znajdujących się po tej samej stronie hiperpłaszczyzny w przestrzeni wzorców. Położenie tej hiperpłaszczyzny w przestrzeni sygnałów wejściowych determinowane jest przez parametry neuronu (współczynniki wagowe i próg). Obserwując zachowanie neuronów z liniową funkcją PSP można stwierdzić, że próbują one rozwiązać stawiane im zadania poprzez odpowiednie manipulowanie wspomnianą hiperpłaszczyzną. Na przykład często podejmowane zadanie rozpoznawania wejściowych sygnałów neurony te usiłują zrealizować optymalizując klasyfikację wejściowych sygnałów poprzez stosowane podzielenie na części całej przestrzeni sygnałów wejściowych (na podstawie odpowiednich wzorców) za pomocą systemu przecinających się hiperpłaszczyzn. (źródło: StatSoft, materiały do programu Statistica, autor prof. Tadeusiewicz)

DEFINICJA NEURONU



PSP - funkcja **radialna** (różnica wektorowa)

$$y = f\ (\ \|x - w\|\ )$$

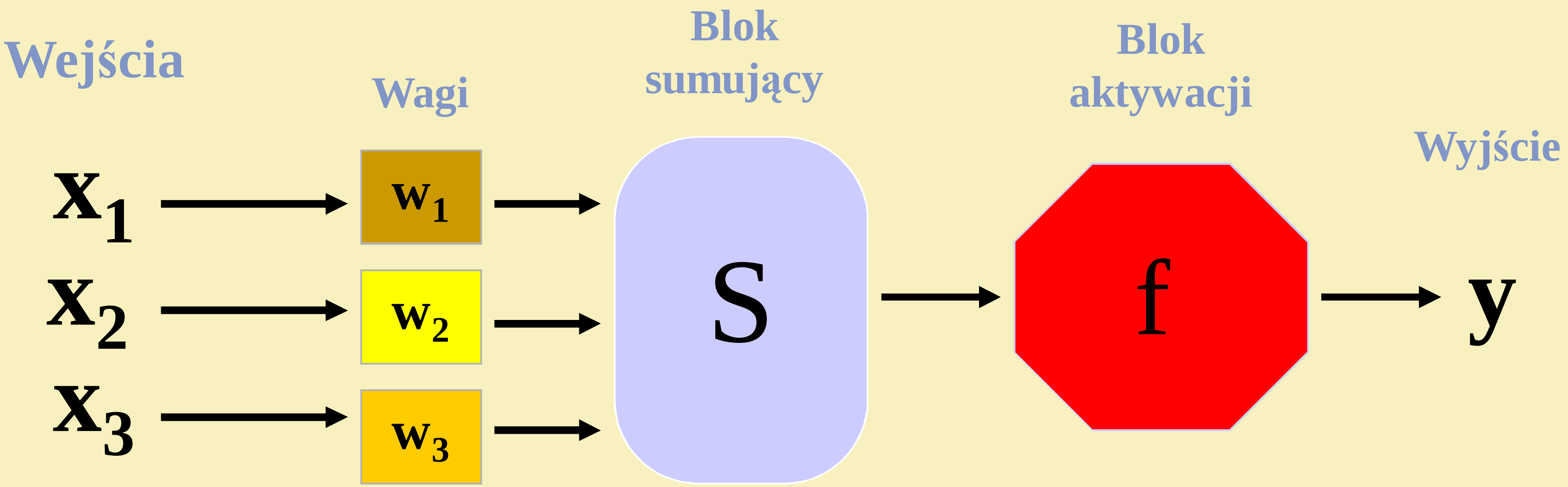
Radialna. Neurony wyposażone w radialną funkcję PSP wyznaczają kwadrat odległości pomiędzy dwoma punktami w N wymiarowej przestrzeni (gdzie N jest liczbą wejść). Punkty pomiędzy którymi wyznacza się odległość reprezentują odpowiednio wektor opisujący sygnał wejściowy oraz wektor wag neuronu. Neurony posiadające radialną funkcję PSP wytwarzają identyczne wartości wyjściowe dla wszystkich sygnałów wejściowych leżących na hipersferach wyznaczonych w przestrzeni tych sygnałów wejściowych. Środki tych hipersfer ulokowane są w punktach odpowiadających wektorom wag neuronów. Wektory te pełnią rolę wzorców sygnałów, na które dana sieć powinna szczególnie reagować. Neurony radialne próbują więc zrealizować klasyfikację wejściowych sygnałów poprzez pomiar odległości reprezentowanych przez nie punktów od wyznaczonych wzorców, które przechowywane są w postaci wektorów wag neuronów. Kwadrat odległości wyznaczany przez neurony radialne mnożony jest przez wartość progową (która w neuronach radialnych pełni rolę miary wartości dopuszczalnego odchylenia); w ten sposób wyznaczana jest wartość wejściowa rozważanego neuronu.

(źródło: StatSoft, materiały do programu Statistica, autor prof. Tadeusiewicz)

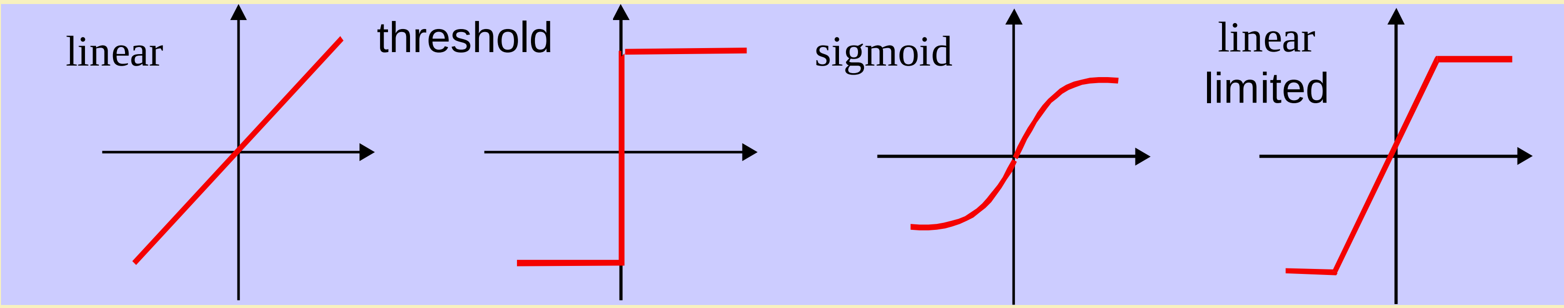
Ilorazowa.

Ten typ funkcji PSP został specjalnie zaprojektowany dla sieci regresyjnych i nie powinien być stosowany w innych przypadkach. W neuronach stosujących ten typ funkcji PSP oczekuje się, że waga skojarzona z jednym wejściem będzie równa +1, waga skojarzona z innym wejściem będzie równa -1, zaś wszystkie pozostałe wagi przyjmują wartość zero. Wartością generowaną przez tę funkcję jest wartość powstająca w ten sposób, że wartość sygnału na wejściu odpowiadającym wadze +1 podzielona jest przez wartość sygnału na wejściu o wadze -1.
(źródło: StatSoft, materiały do programu Statistica, autor prof. Tadeusiewicz)

DEFINICJA NEURONU



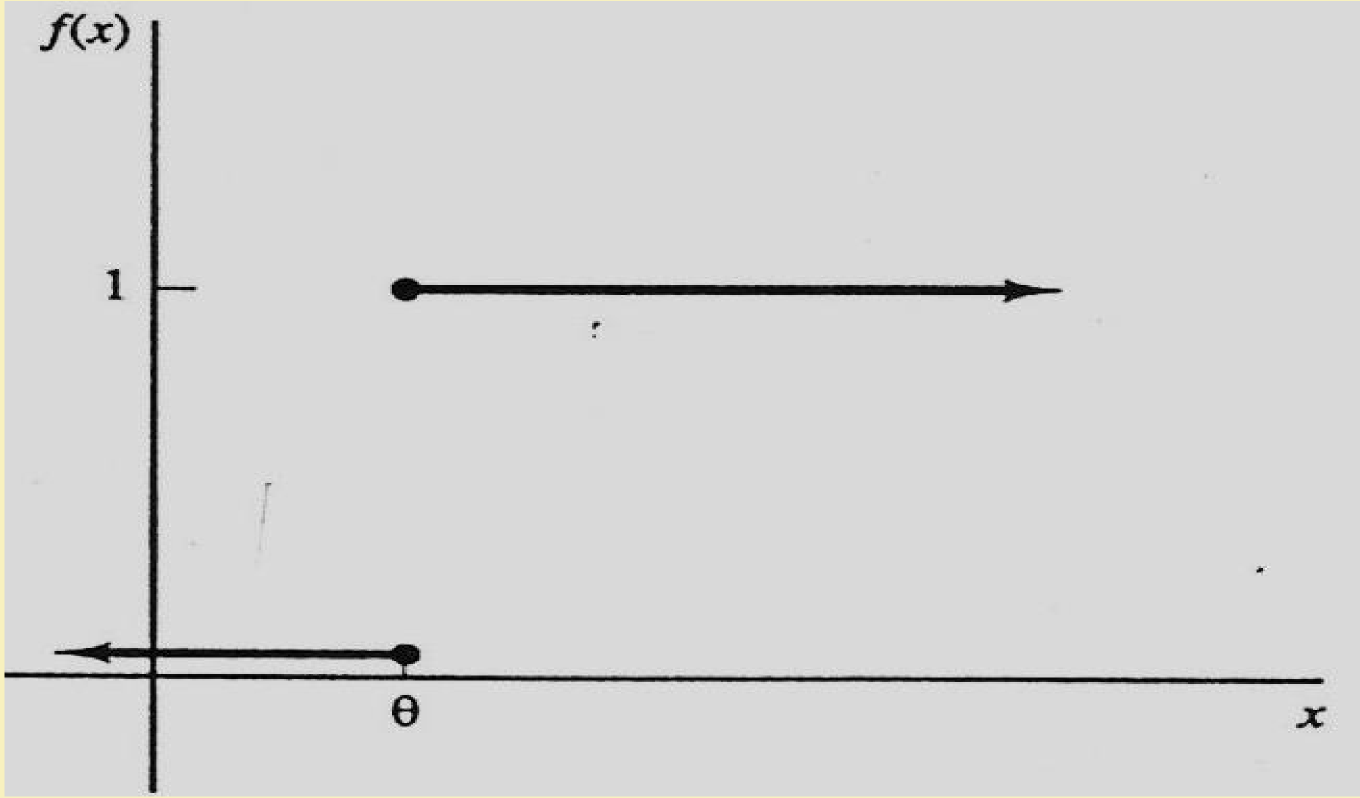
$$y = f\ (\ S\ (\ x_i * w_i\ ))$$



f – funkcja nieliniowa

FUNKCJE NIELINIOWE

FUNKCJA SKOKOWA
BINARY STEP FUNCTION



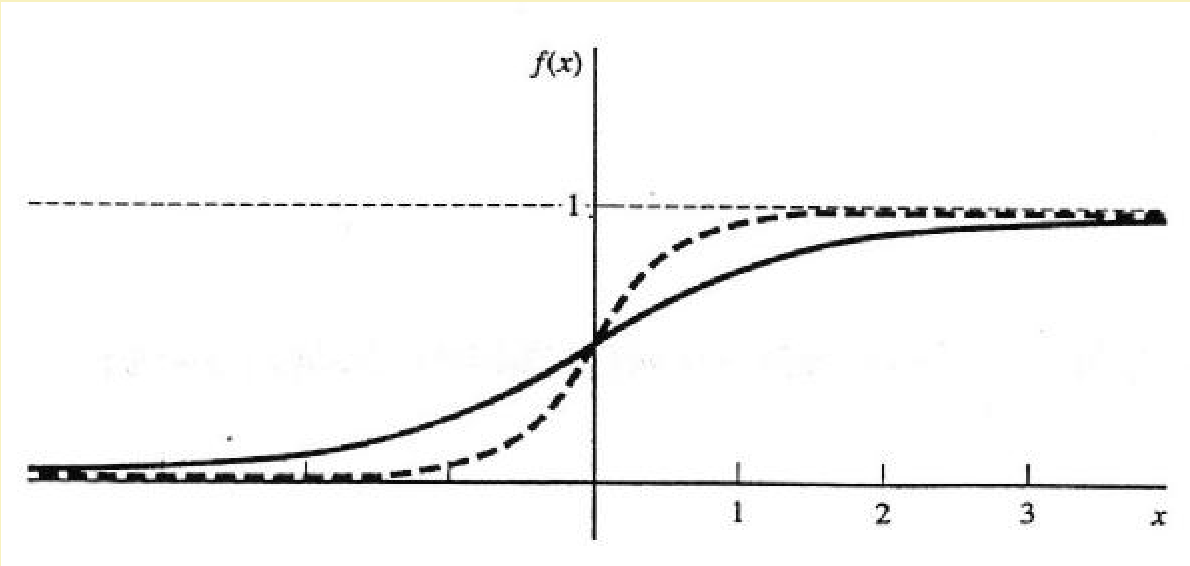
FUNKCJE NIELINIOWE

SIGMOIDA UNIPOLARNA (BINARNA)

BINARY SIGMOID

$$f(x) = \frac{1}{1 + \exp(-\sigma x)} .$$

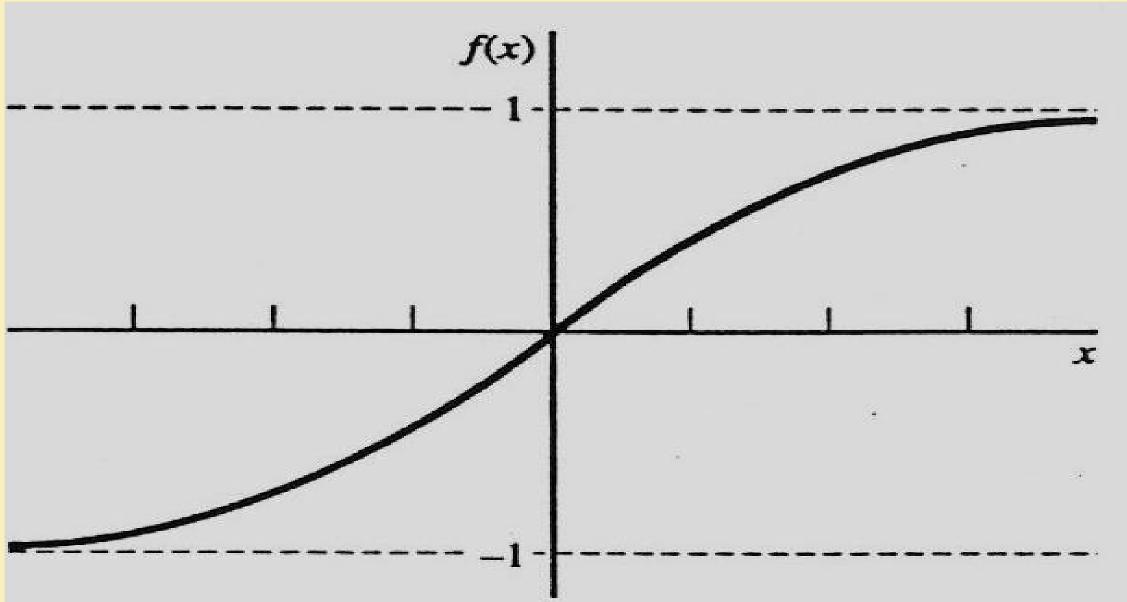
$$f'(x) = \sigma f(x)\,[1 - f(x)].$$



FUNKCJE NIELINIOWE

SIGMOIDA BIPOLARNA

BIPOLAR SIGMOID



$$g(x) = 2f(x) - 1 = \frac{2}{1 + \exp(-\sigma x)} - 1$$

$$= \frac{1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}.$$

$$g'(x) = \frac{\sigma}{2}[1 + g(x)][1 - g(x)].$$

FUNKCJE NIELINIOWE

TANGENS HIPERBOLICZNY

HYPERBOLIC TANGENS

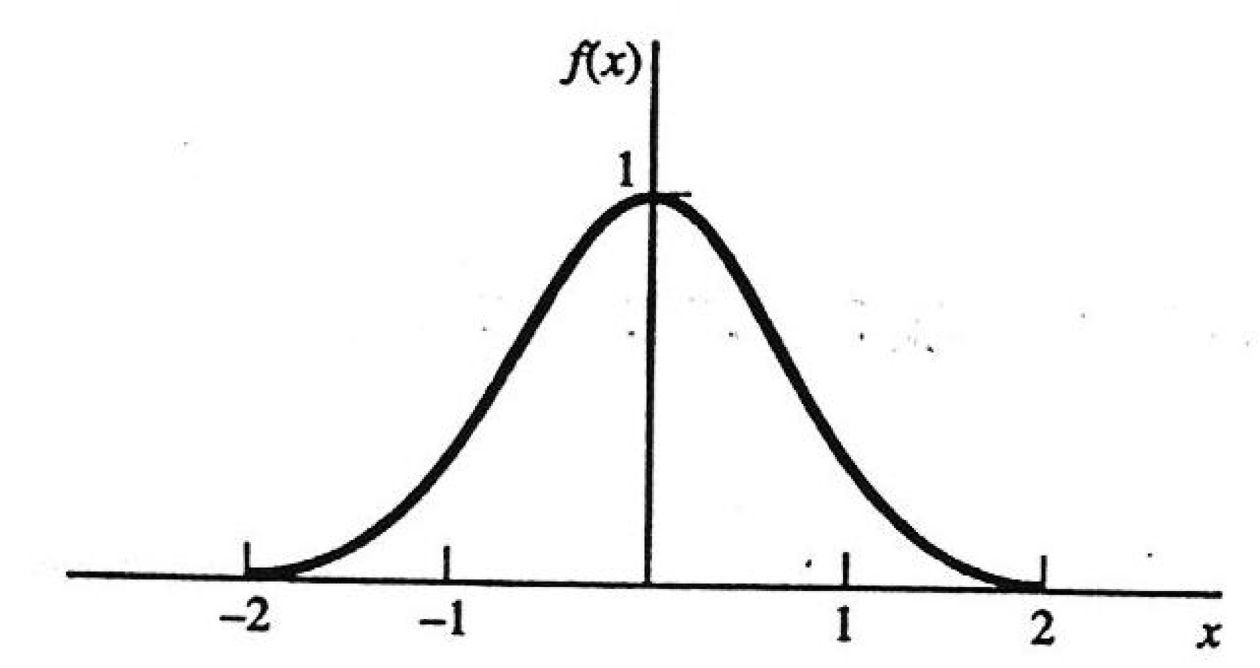
$$h(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

$$= \frac{1 - \exp(-2x)}{1 + \exp(-2x)}.$$

$$h'(x) = [1 + h(x)][1 - h(x)].$$

FUNKCJE NIELINIOWE

Funkcja Gaussa



$$f(x) = \exp(-x^2);$$

$$f'(x) = -2x \exp(-x^2) = -2xf(x).$$

Liniowa	x	(-inf,+inf)
Logistyczna	$\frac{1}{1+e^{-x}}$	(0,+1)
Hiperboliczna	$\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	(-1,+1)
Wykładnicza	e^{-x}	(0, +inf)
Softmax	$\frac{e^{x}}{\sum_i e^{x_i}}$	(0,+1)
Pierwiastek	$\sqrt{x}$	[0, +inf)
Sinus	$sin(x)$	[0,+1]
Liniowa z nasyceniem	$\begin{cases} -1 & x \le -1 \\ x & -1 < x < +1 \\ +1 & x \ge +1 \end{cases}$	[-1,+1]
Progowa	$\begin{cases} 0 & x < 0 \\ +1 & x \ge 0 \end{cases}$	[0,+1]

BIAS (przesunięcie) i PRÓG (threshold)

Jeżeli neuron posiada n wejść to wektor jego sygnałów wejściowych można przedstawić w postaci:

$$x = (x_1, x_2, \ldots, x_n)$$

a wektor wag połączeń jako wektor

$$w = (w_1, w_2, \ldots, w_n)$$

Neuron oblicza sumę ważoną tych wartości

$$e = b + \sum_{i=1}^{n} x_i\, w_i$$

gdzie b to waga połączenia bias .

Możemy przyjąć, że $w_0 = b$ i wtedy połączenie **bias** traktujemy dokładnie tak jak każdą inną wagę z tym, że sygnał wejściowy x_0 tego połączenia wynosi **1.**

Zatem łączne pobudzenie neuronu możemy zapisać w prostszej formie:

$$e = \sum_{i=0}^{n} x_i w_i$$

Funkcję aktywacji neuronu często przyjmujemy jako

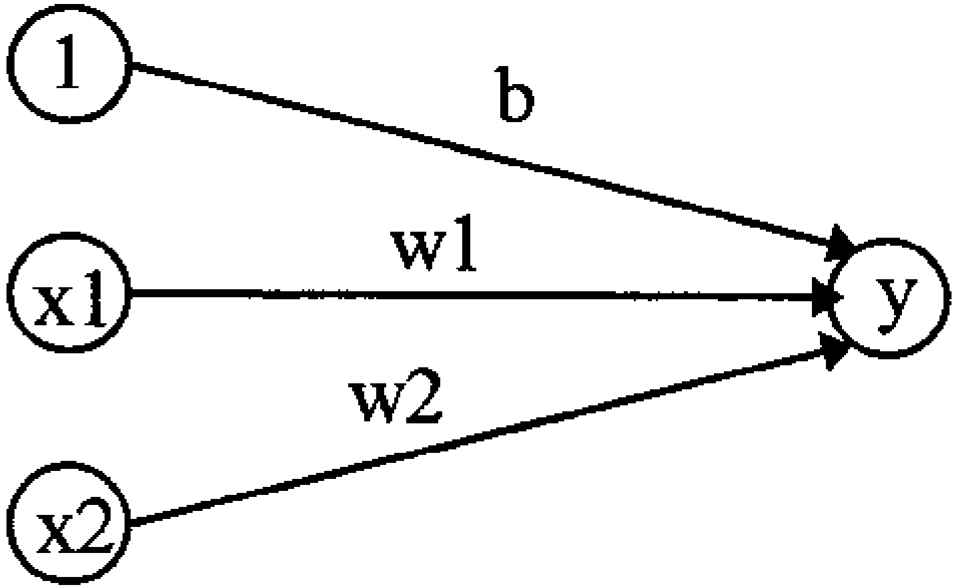
$$f(e) = \begin{cases} 1 & \text{gdy } e >= 0 \\ 0 & \text{gdy } e < 0 \end{cases}$$

Czasami zamiast elementu bias stosuje się stały **próg** i wtedy

$$f(e) = \begin{cases} 1 & \text{gdy } e >= \theta \\ 0 & \text{gdy } e < \theta \end{cases}$$

gdzie $e = \sum_{i=1}^{n} x_i w_i$

Zauważmy, że warunek $\mathbf{\Sigma\, x_i\, w_i = 0}$ i=1, ..., n dla n=2 oznacza równanie prostej, dla n=3 równanie płaszczyzny a ogólnie dla n>3 rozmaitość liniową stopnia n-1 czyli hiperpłaszczyznę w przestrzeni sygnałów wejściowych.

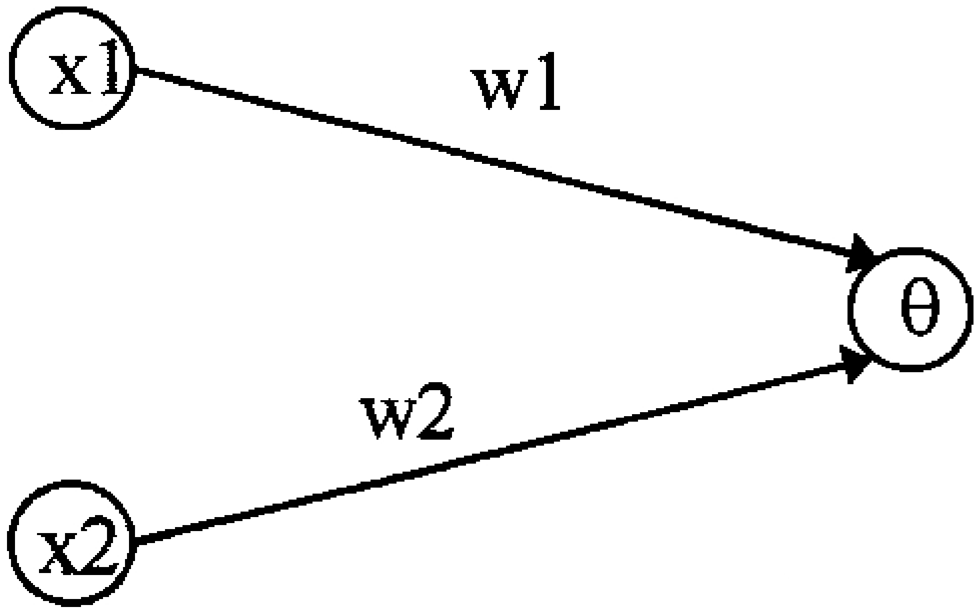


Granica pomiędzy wartościami, dla których sieć daje wartość 1 i 0 jest prostą

$$b + x_1 w_1 + x_2 w_2 = 0$$

lub (zakładając w_2 różne od 0)

$$x_2 = -w_1/w_2\, x_1 - b/w_2.$$



W przypadku wprowadzenia progu analogiczne równanie ma postać

$$x_1 w_1 + x_2 w_2 = \theta$$

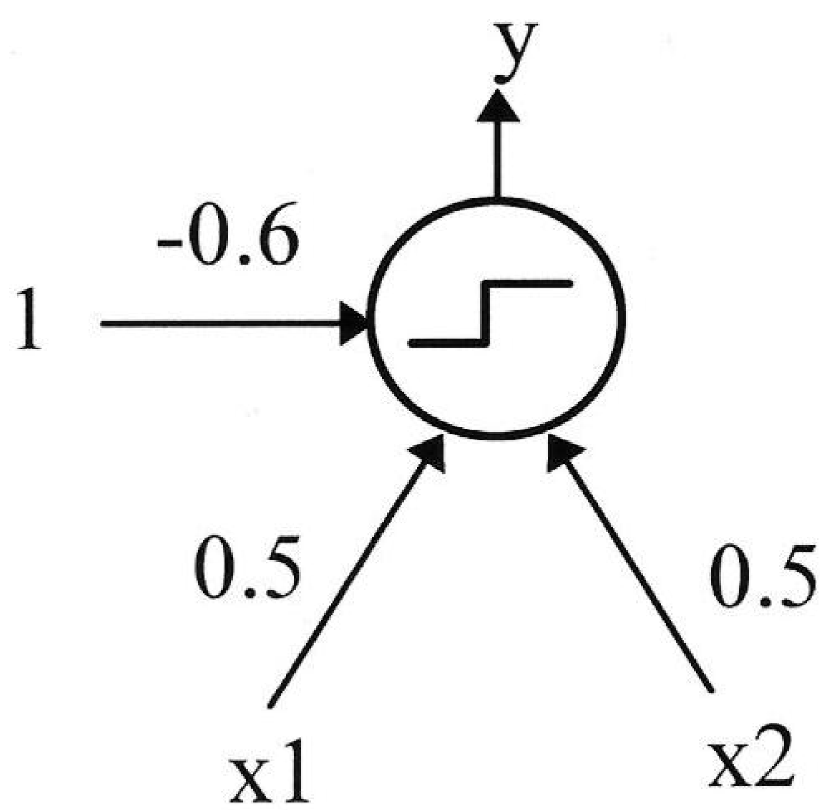
lub (zakładając w_2 różne od 0)

$$x_2 = -w_1/ w_2\, x_1 + \theta/ w_2.$$

$$x_2 = -w_1/w_2 * x_1 - b/w_2$$ **BIAS**

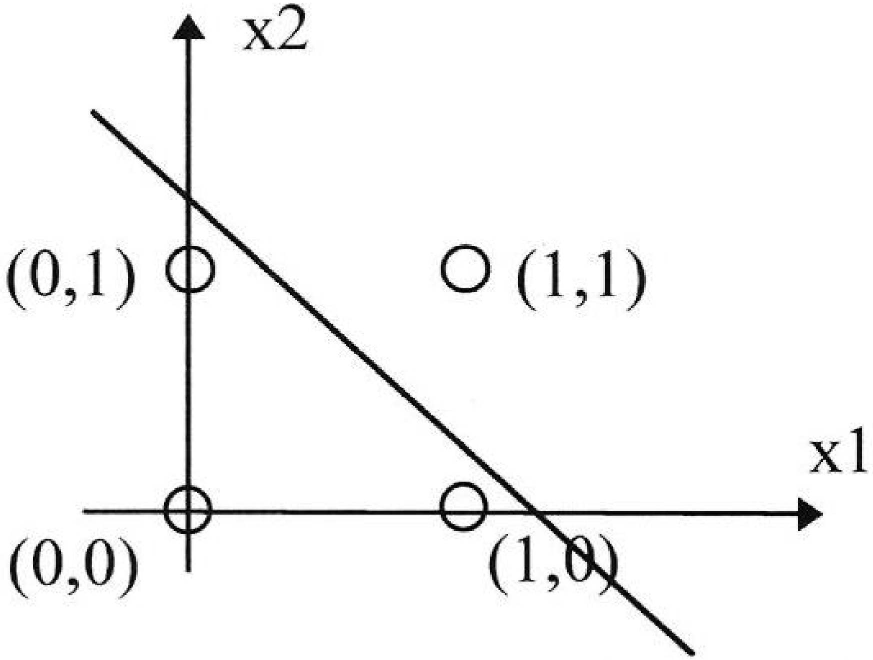
$$x_2 = -w_1/w_2 * x_1 + q/w_2$$ **PRÓG**

Przy porównaniu, równania z elementem bias i progiem wyglądają bardzo podobnie. Gdyby w obu sieciach wagi były takie same to b równałoby się -θ. Trzeba jednak pamiętać, że w trakcie uczenia wartość wagi bias jest modyfikowana natomiast próg θ pozostaje niezmienny. Korzystanie z jednej lub drugiej metody uzależnione jest od rozwiązywanego problemu.



Ta sieć realizuje funkcję logiczną AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



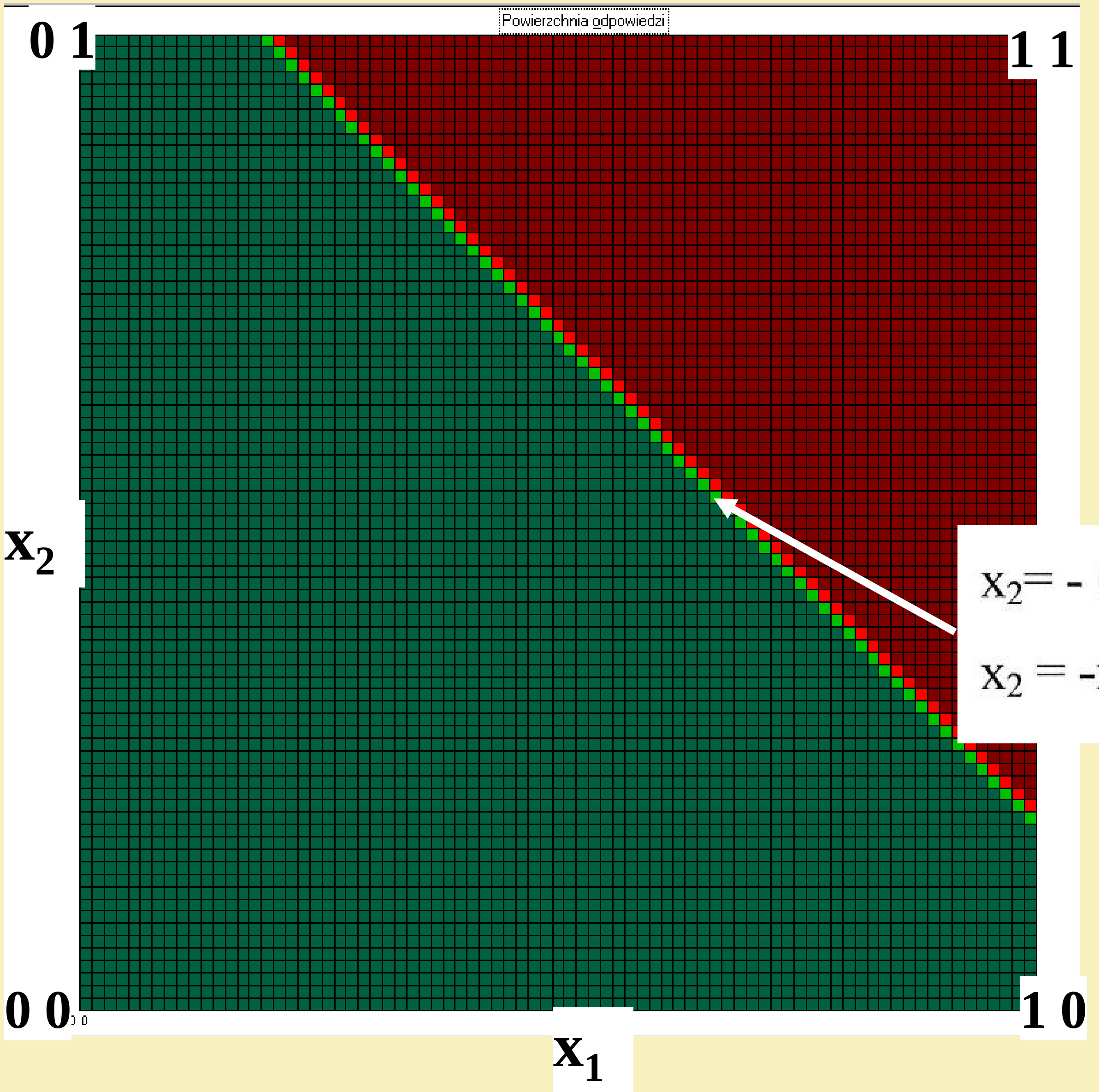
Prostą rozdzielającą punkty w przestrzeni sygnałów wejściowych równanie:

$$x_2 = -0.5 / 0.5\, x_1 - (-0.6) / 0.5$$

$$x_2 = -x_1 + 1.2$$

DYSKRYMINACJA LINIOWA

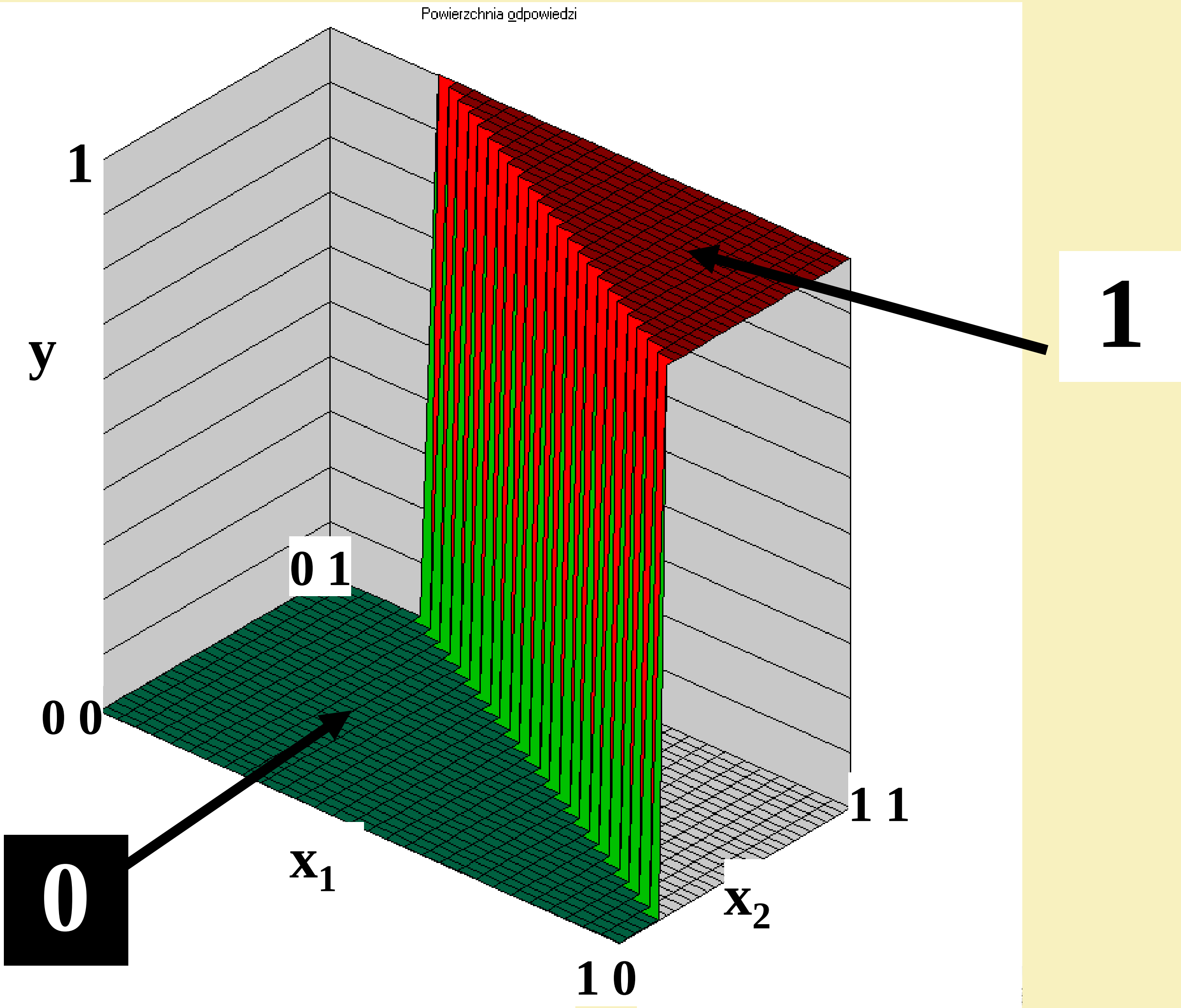
PRZEZ NEURON NIELINOWY



$$x_2 = -\,0.5 \,/\, 0.5\ x_1 - (-0.6) \,/\, 0.5$$

$$x_2 = -x_1 + 1.2$$

POWIERZCHNIA ODPOWIEDZI



ADALINE (ADAptive LINear Element)

Reguła WIDROW-HOFFA

$$\mathbf{W'} = \mathbf{W} + h\, d\, \mathbf{X}$$

$$d = \mathbf{z} - \mathbf{y}$$

h - szybkość uczenia

UCZENIE SIECI NIELINIOWYCH JEDNOWARSTWOWYCH

CEL: uzyskanie jak największej zgodności pomiędzy odpowiedzią neuronu a wymaganą wartością na wyjściu.

METODA: minimalizacja (przy pomocy metody gradientowej) funkcji kryterialnej.

$$\mathbf{Q = ½ \overset{N}{\underset{1}{S}} (z – y)^{2} = ½ \overset{N}{\underset{1}{S}} (z – f(e))^{2}}$$

$$\mathrm{D}\, \mathbf{w_i} = -\, \mathrm{h} \frac{\delta Q}{\delta\, \mathbf{w_i}}$$

Oznaczamy jako uogólniony błąd

$d_{\mathbf{uog}}$

$$\frac{\delta Q}{\delta\, \mathbf{w_i}} = \frac{\delta Q}{\delta\, \mathbf{e}} \frac{\delta e}{\delta\, \mathbf{w_i}}$$

W efekcie otrzymujemy: $\mathrm{D}\, \mathbf{w_i} = \mathrm{h}\, d_{\mathbf{uog}}\, \mathbf{x_i}$

czyli REGUŁĘ DELTA

REGUŁA UCZENIA DELTA DLA SIECI JEDNOWARSTWOWYCH

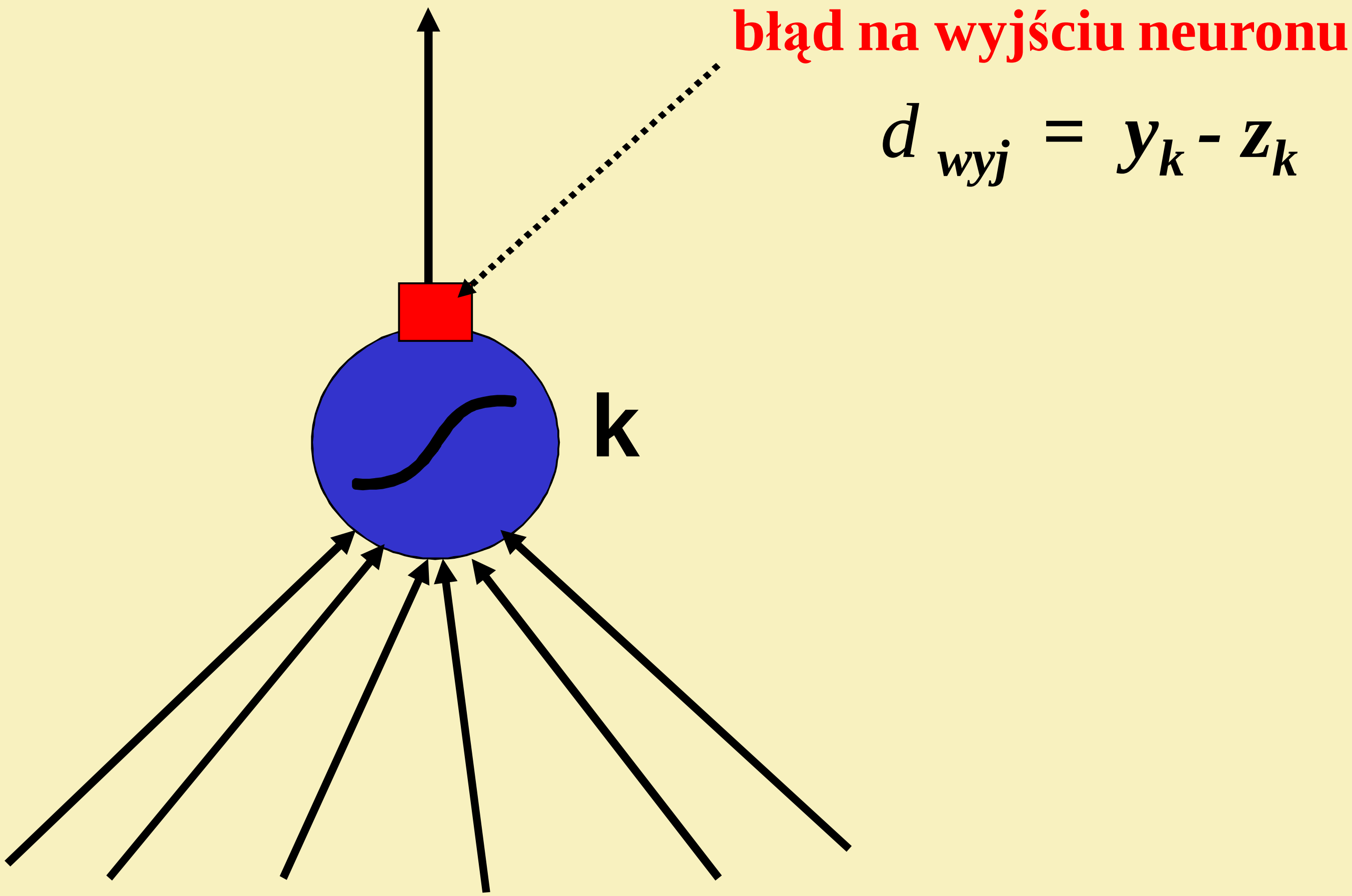
$$W' = W + \eta\, \delta\, x^T$$

$$\delta_m \overset{def}{=} \frac{\partial Q}{\partial e_k} = -\frac{1}{2}\frac{\partial (z_k - y_k)^2}{\partial e_k} = (z_k - y_k)\frac{\partial y_k}{\partial e_k}$$

błąd
k-tego neuronu
(na jego wejściu)

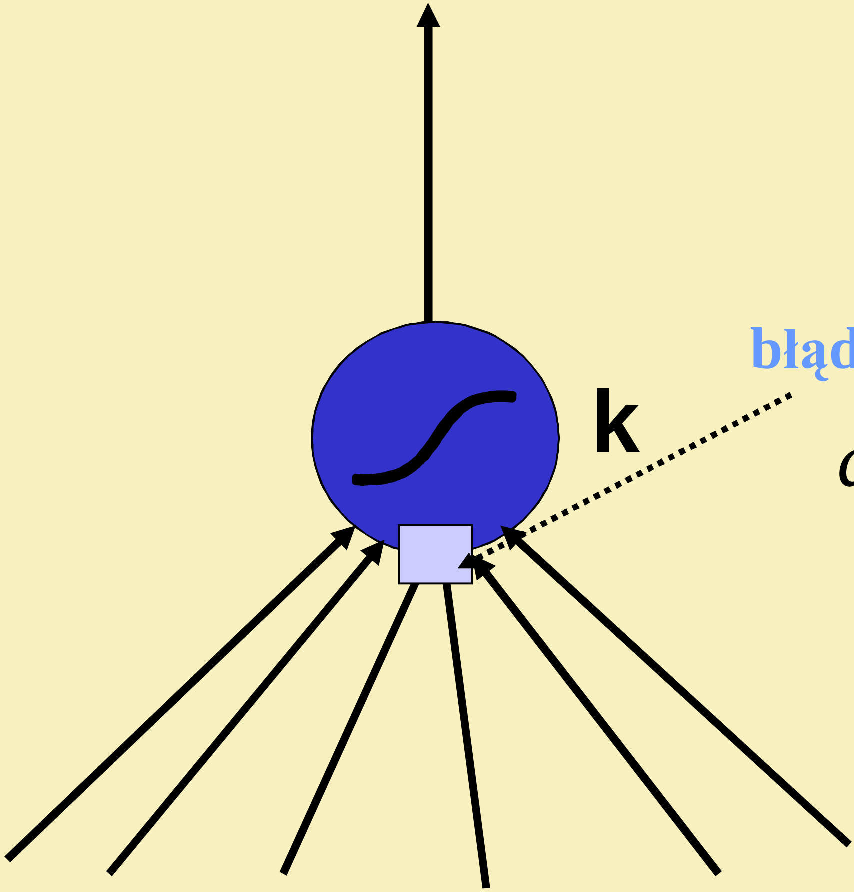
$$\delta_k = (z_k - y_k) f'(e_k)$$

BŁĄD NA WYJŚCIU I NA WEJŚCIU NIELINIOWEGO NEURONU



$$d_{wyj} = y_k - z_k$$

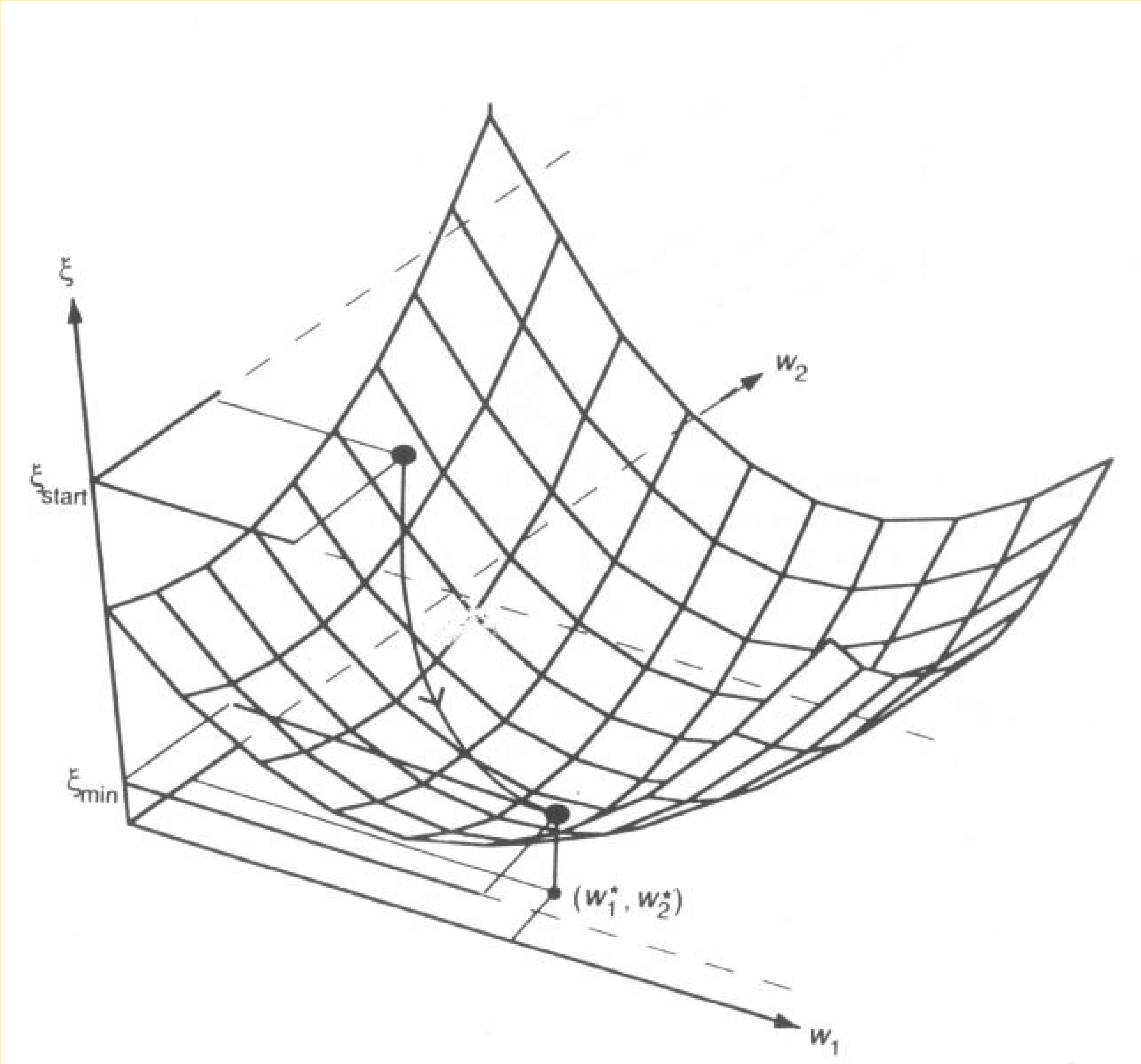
BŁĄD NA WYJŚCIU I NA WEJŚCIU NIELINIOWEGO NEURONU



błąd na wejściu neuronu

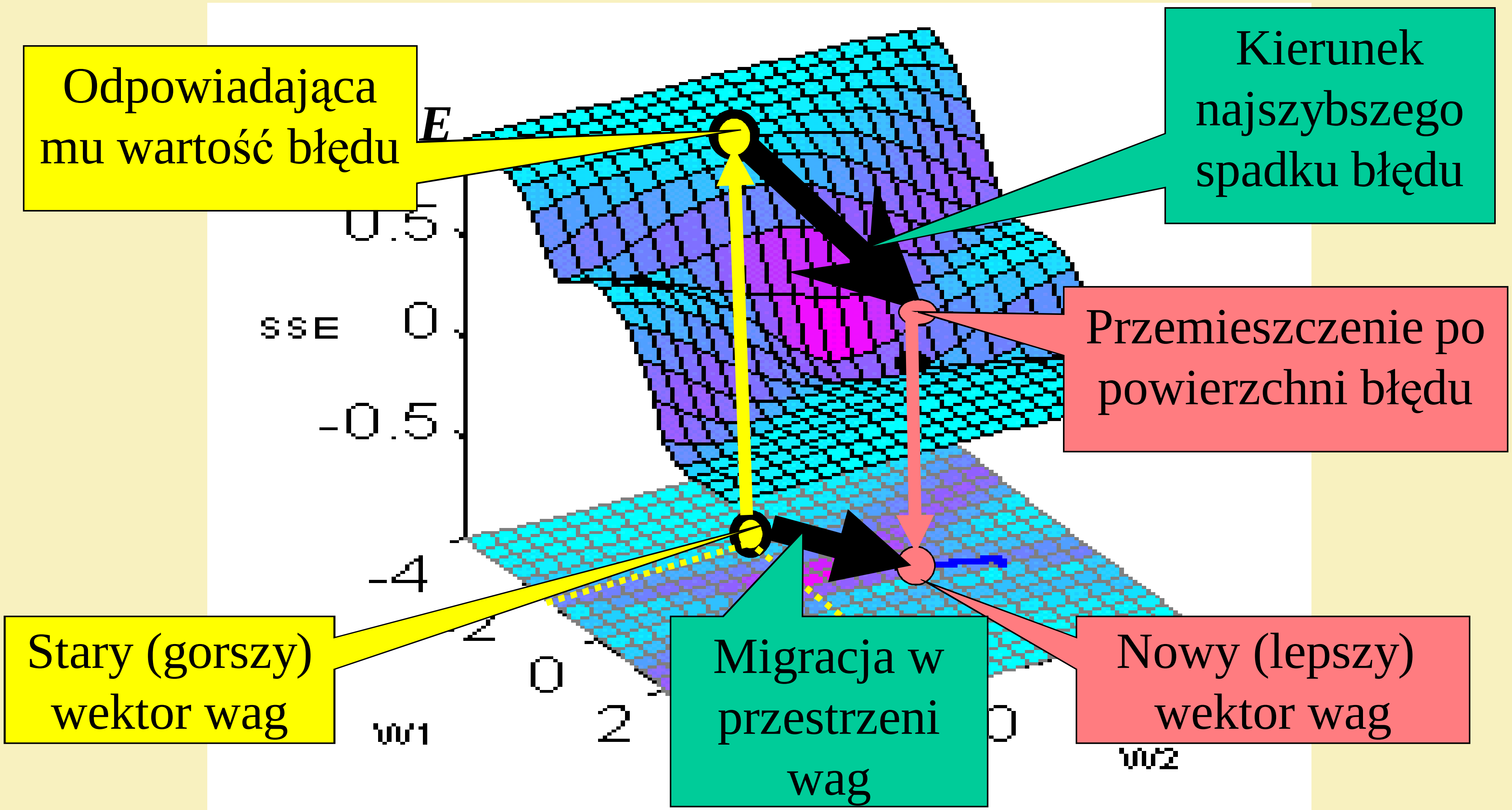
$$d_{wej} = (y_k - z_k)\; f'(e_k)$$

KSZTAŁT POWIERZCHNI BŁĘDU I METODA NAJSZYBSZEGO SPADKU

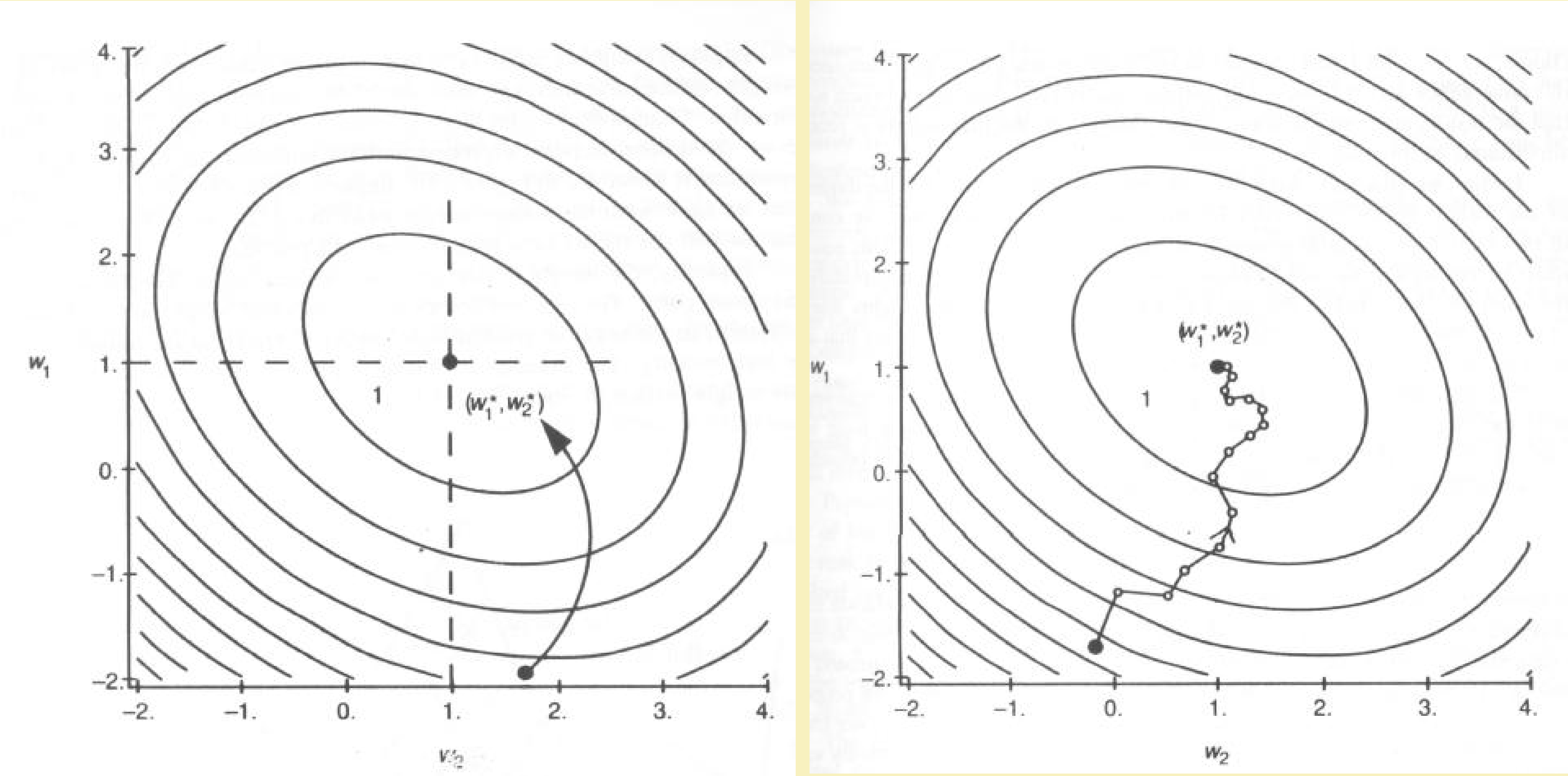


ALGORYTM WSTECZNEJ PROPAGACJI BŁĘDÓW

polega na szukaniu kierunku spadku *E* i na takim zmienianiu wartości wag w_1, w_2, żeby zmniejszać wartość funkcji błędu w kierunku jej najszybszego spadku



KSZTAŁT POWIERZCHNI BŁĘDU I METODA NAJSZYBSZEGO SPADKU



a) idealna b) rzeczywista

trajektoria końca wektora wag w procesie uczenia sieci

REGUŁA UCZENIA DELTA DLA SIECI JEDNOWARSTWOWYCH

Reguła DELTA dla konkretnej wagi przyjmuje postać:

$$w_{ik}^{(j+1)} = w_{ik}^{(j)} + \eta\, f'\left(e_k^{(j)}\right)\left(z_k^{(j)} - y_k^{(j)}\right)x_i^{(j)}$$

$w_{ik}^{(j)}$ - waga połączenia i-tego wejścia z k-tym neuronem w i-tym kroku

$y_k^{(j)}$ - sygnał wyjściowy k-tego neuronu w j-tym kroku równy:

$$y_k^{(j)} = f\left(e_k^j\right) = f\left(\sum_{l=0}^{L} w_{lk}^{(j)} x_l^{(j)}\right)$$

$e_k^{(j)}$ - pobudzenie k-tego neuronu

f‘ - pochodna funkcja aktywacji neuronu

$z_k^{(j)}$ - sygnał wymaganej odpowiedzi k-tego neuronu w j-tym kroku

η - współczynnik szybkości uczenia

BACKPROPAGATION

Algorytm uczenia sieci nieliniowych **backpropagation** czyli **metoda wstecznej propagacji błędów** polega na odtwarzaniu przypuszczalnej wartości błędów głębszych warstw sieci (do których nie ma bezpośredniego dostępu) na podstawie rzutowania wstecz błędów wykrytych na wyjściu. Rozważając pojedynczy neuron warstwy ukrytej bierze się pod uwagę błędy wszystkich tych neuronów, do których wysłał swój sygnał wyjściowy, sumuje się je uwzględniając wagi.

Reguły uczenia BACKPROPAGATION wielowarstwowych sieci nieliniowych (uogólniona reguła DELTA)

Reguła zmiany wag dla k-tego neuronu w m-tej warstwie jest identyczna jak reguła DELTA:

$$w_{ik}^{(m)'} = w_{ik}^{(m)} + \eta \, \delta_k^{(m)} \, x_i^{(m)}$$

gdzie:

$x_i^{(m)}$ - sygnał wejściowy do k-tego neuronu z m-tej warstwy pochodzący od i-tego neuronu z warstwy m-1

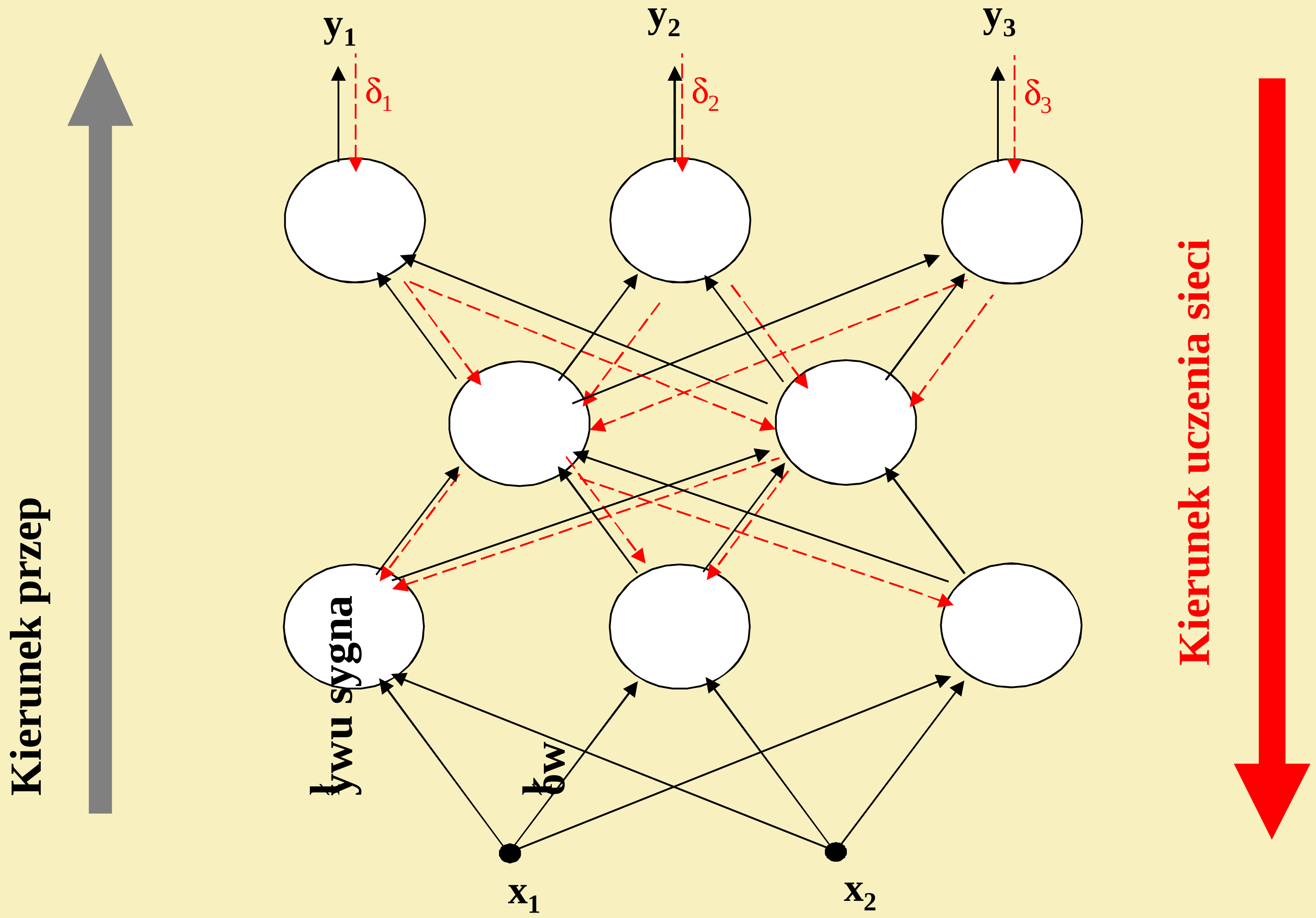
$\delta_i^{(m)}$ - przypuszczalny błąd na jego wejściu

$$\delta_k^{(m)} = \begin{cases} f'(e_k)(z_k - y_k) & \text{dla warstwy wyjściowej} \\ f'\left(e_k^{(m)}\right) \sum_{l=1}^{M^{(m+1)}} w_{kl}^{(m+1)} \delta_l^{m+1} & \text{dla warstw ukrytych} \end{cases}$$

gdzie

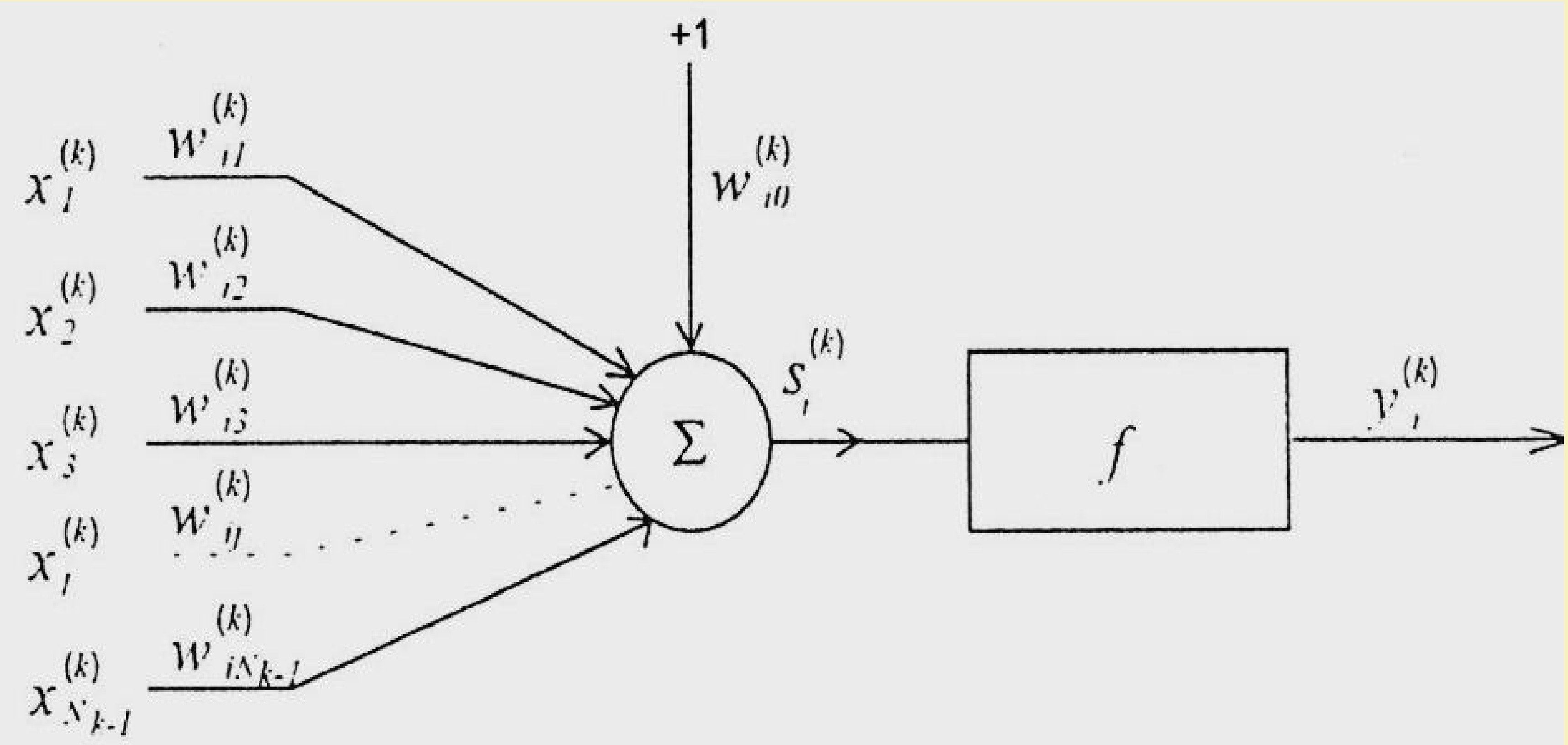
$M^{(m+1)}$ – liczba neuronów w m+1 warstwie

BACKPROPAGATION



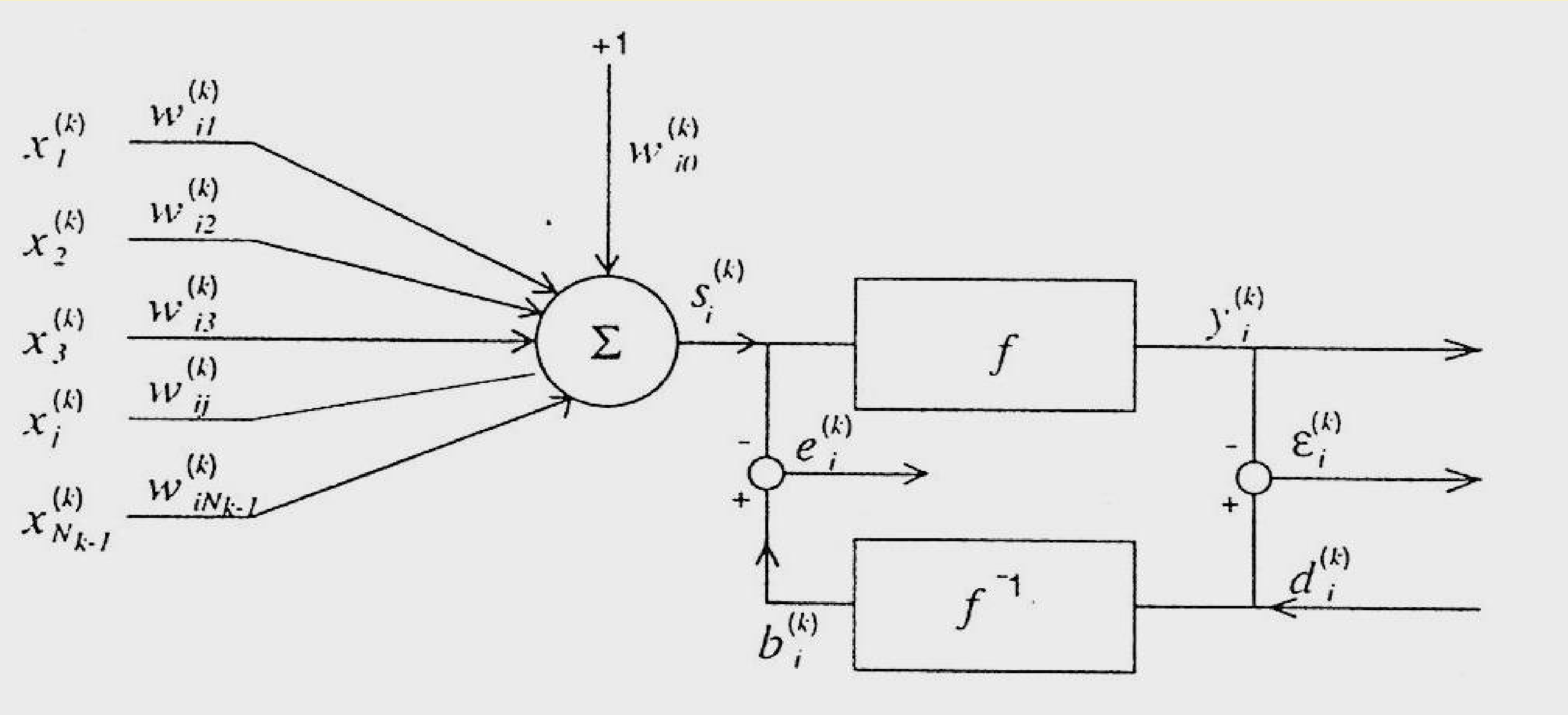
BACKPROPAGATION

MODEL NEURONU

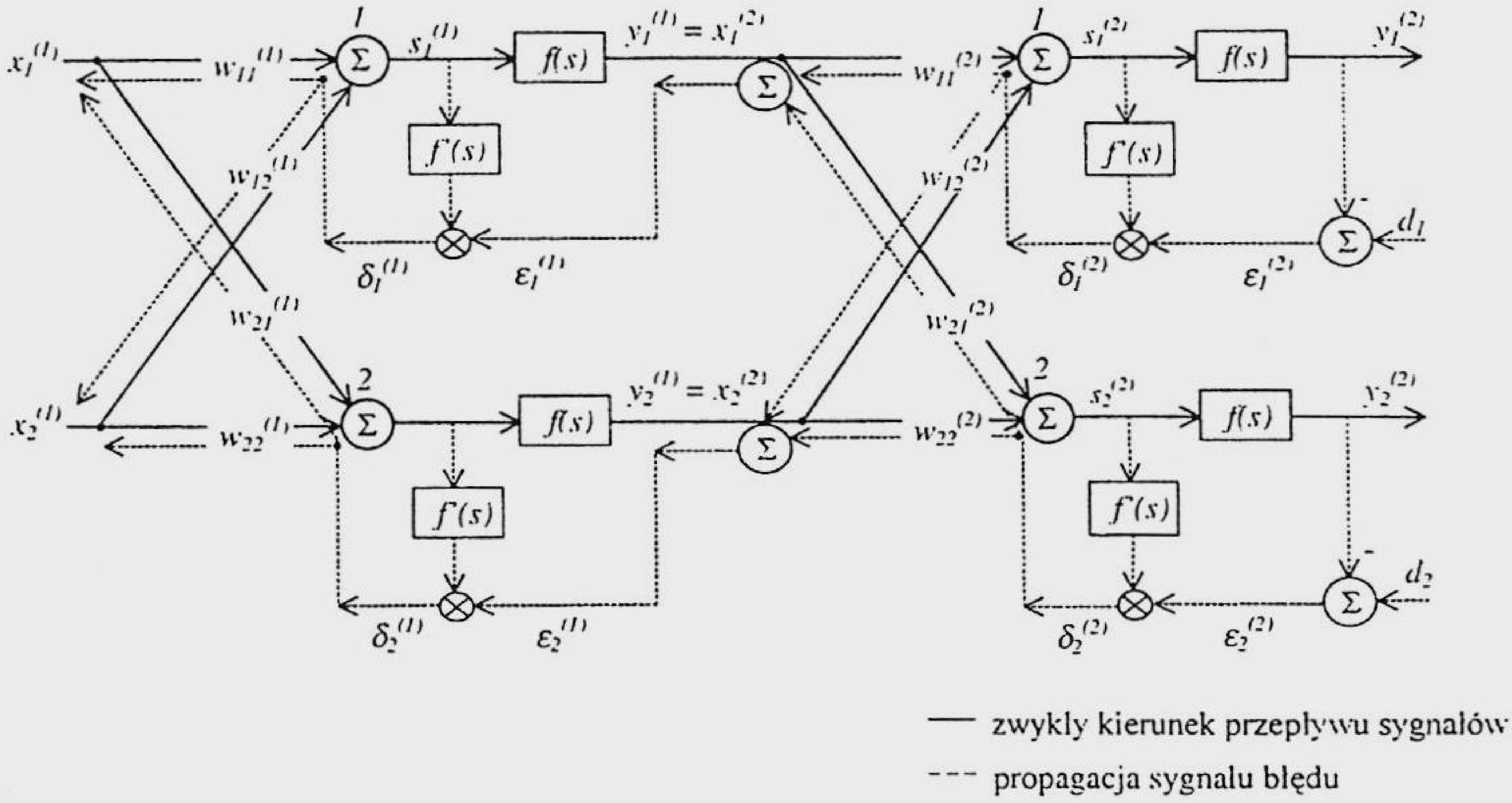


BACKPROPAGATION

MODEL NEURONU



Metoda BACKPROPAGATION dla przykładowej sieci dwuwarstwowej



Reguły uczenia BACKPROPAGATION sieci nieliniowych wielowarstwowych (uogólniona reguła DELTA) (5)

KOREKTA WAG

sposób przyrostowy - aktualizacja wag następuje bespośrednio po podaniu każdej pary uczącej. Funkcja błędu zmienia się w każdym kolejnym kroku. Jeżeli pary uczące podawane są w losowej kolejności to scieżka w przestrzeni wag jest stochastyczna, co pozwala lepiej wykorzystać powierzchnię błędu.

sposób grupowy – obliczany jest gradient błędu łącznego. Korekta wag następuje po podaniu całego zestawu uczącego. Ten sam efekt można uzyskać obliczając poprawki wag dla każdej pary uczącej, ale bez dokonywania jej aktualizacji. Zmiana wagi następuje po prezentacji wszytkich par uczących poprzez dodanie wszytkich poprawek.

Metoda BACKPROPAGATION – obliczanie błędu

Błąd średniokwadratowy

Root **M**ean **S**quare error

$$RMS = \sqrt{\frac{\sum_{i=1}^{N}(z_i - y_i)^2}{N}}$$

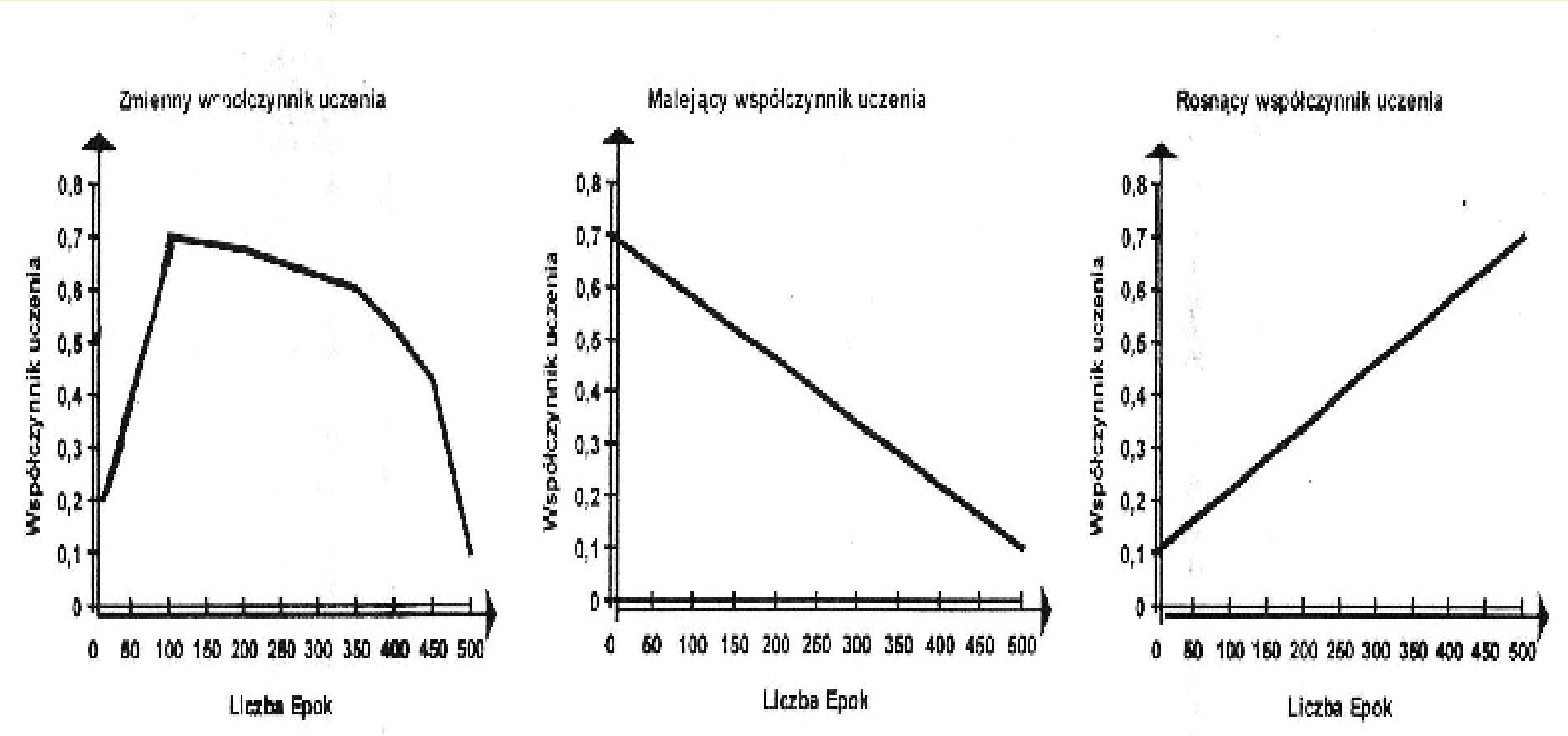
Suma kwadratów błędów w epoce

Total **S**um of **S**quares

$$tss = \sum_{k=1}^{K}\sum_{i=1}^{N}(z_i - y_i)^2$$

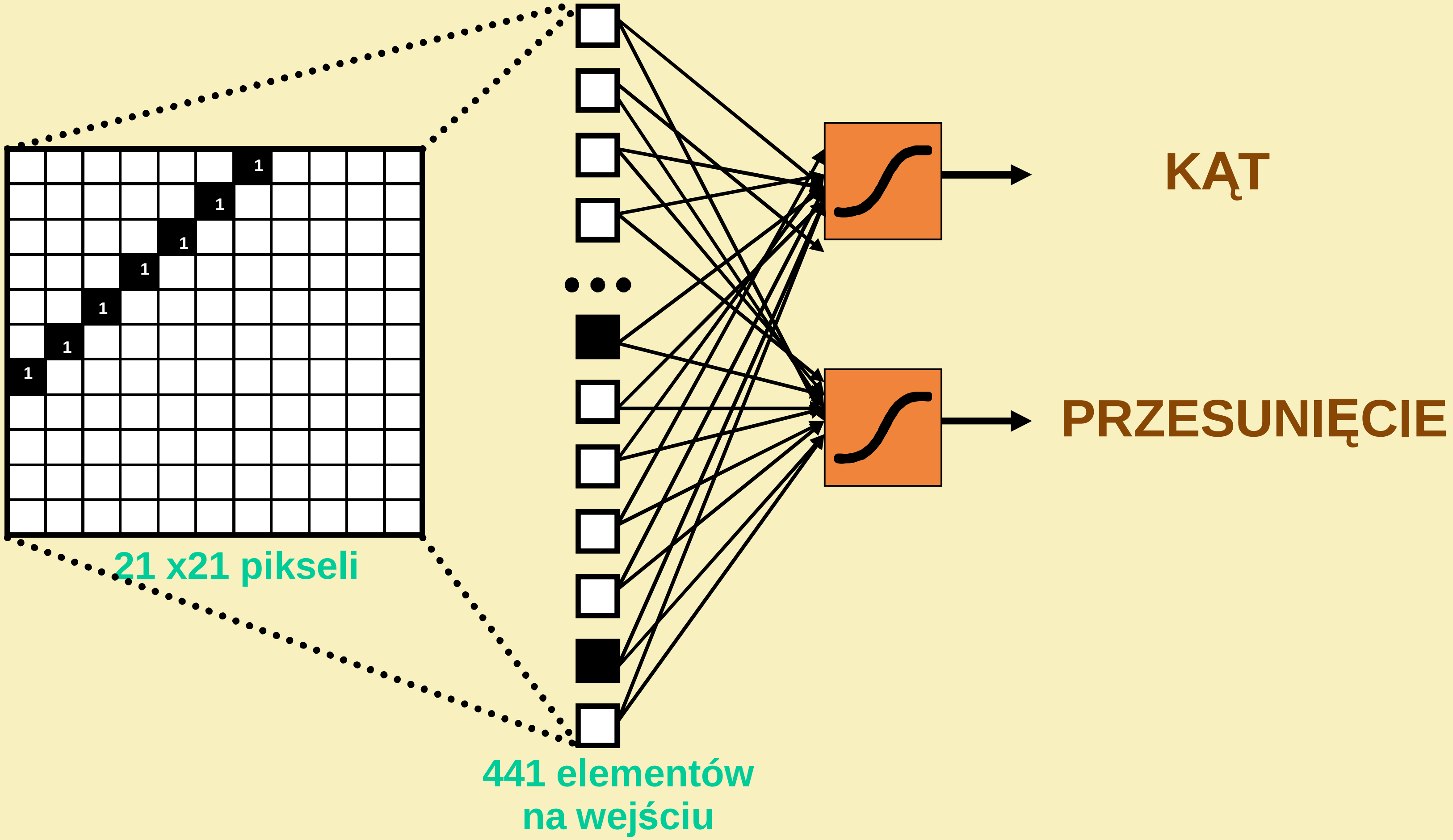
Reguły uczenia BACKPROPAGATION (6)

ZMIANY WSPÓŁCZYNNIKA SZYBKOŚCI UCZENIA



UCZENIE SIECI JEDNOWARSTWOWEJ PRZYKŁAD

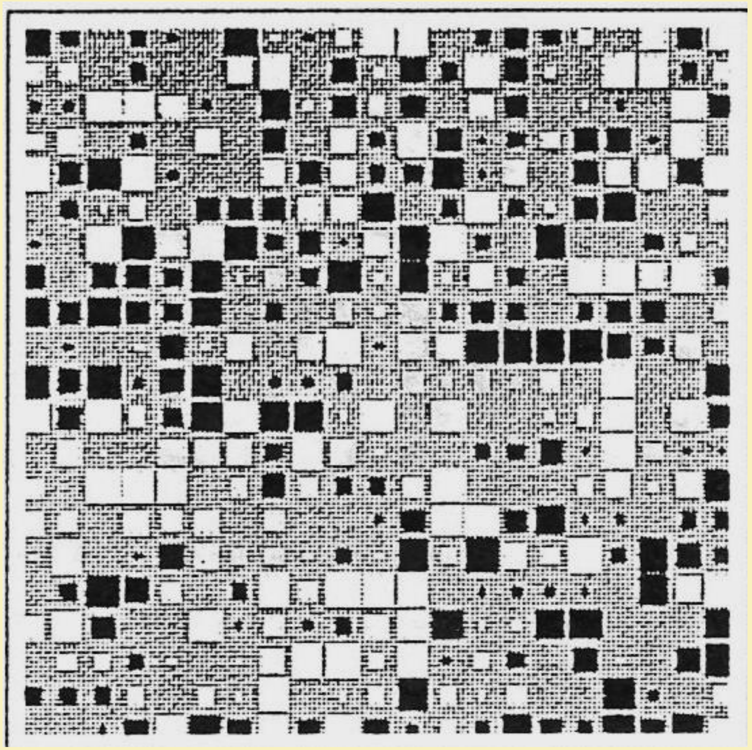
Rozpoznawanie kąta i przesunięcia linii prostych



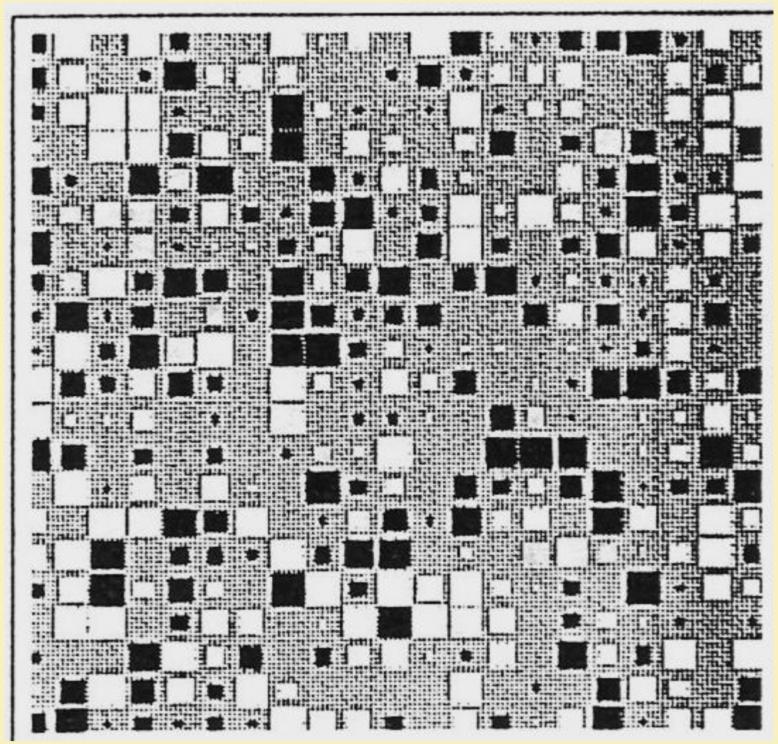
UCZENIE SIECI JEDNOWARSTWOWEJ PRZYKŁAD

WAGI POCZĄTKOWE

KĄT



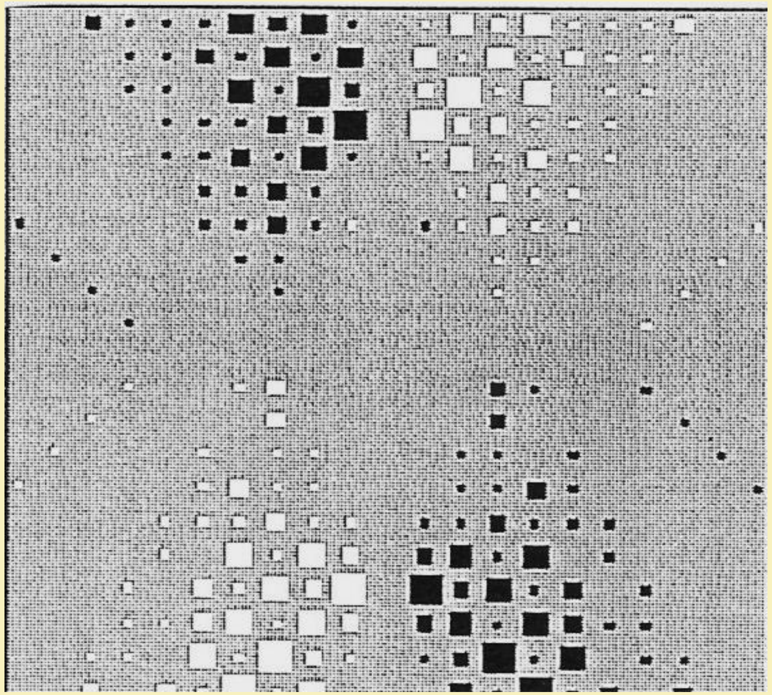
PRZESUNIĘCIE



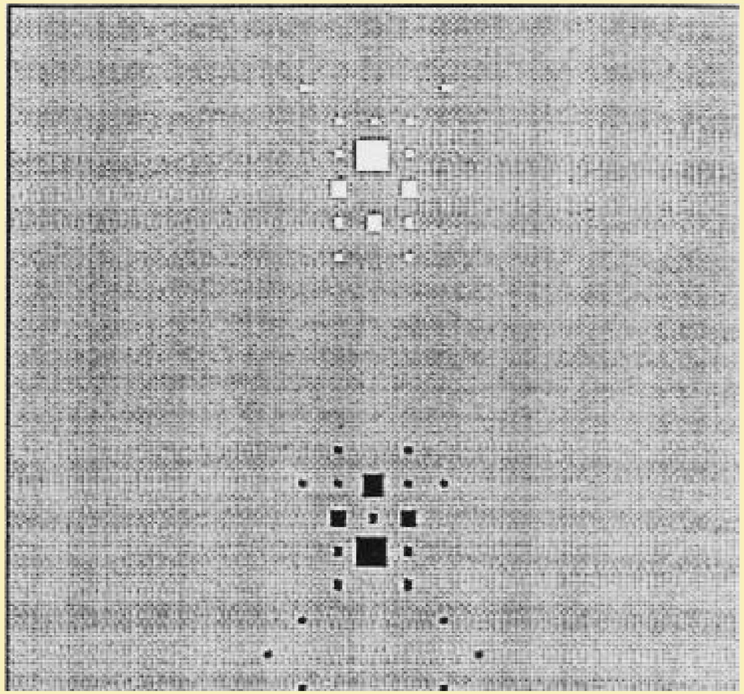
UCZENIE SIECI JEDNOWARSTWOWEJ
PRZYKŁAD

WAGI PO NAUCZENIU PRAWIDŁOWYM

KĄT



PRZESUNIĘCIE

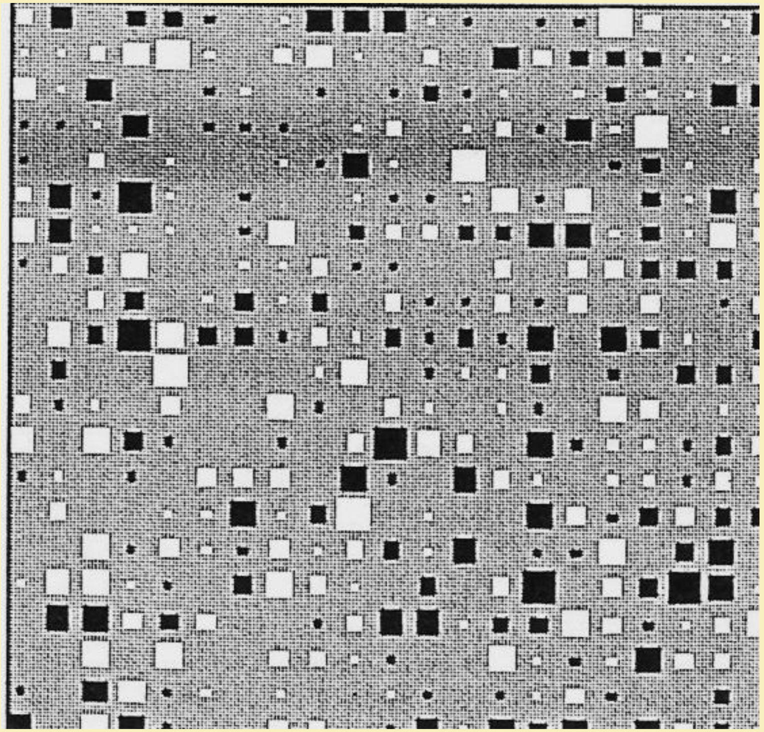


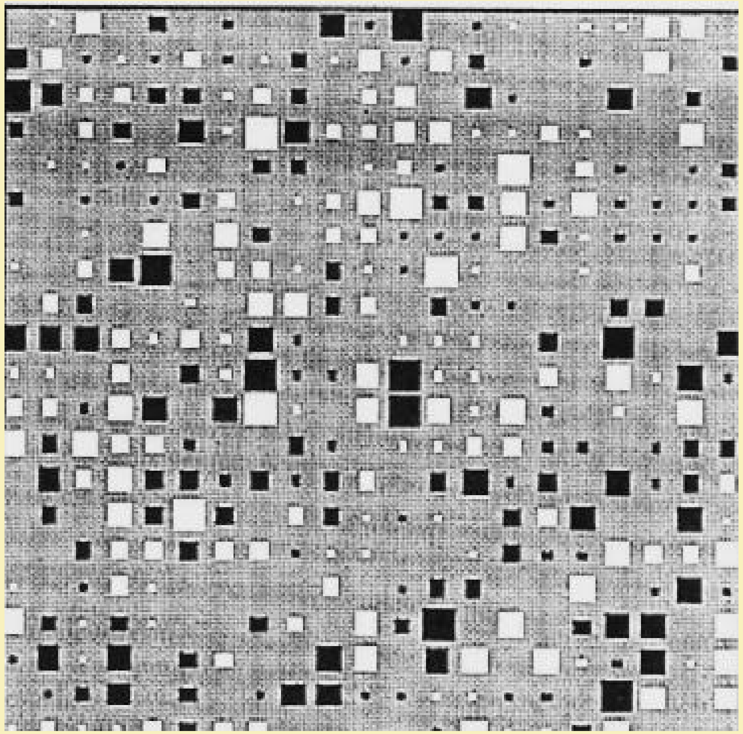
- wagi dodatnie

- wagi ujemne

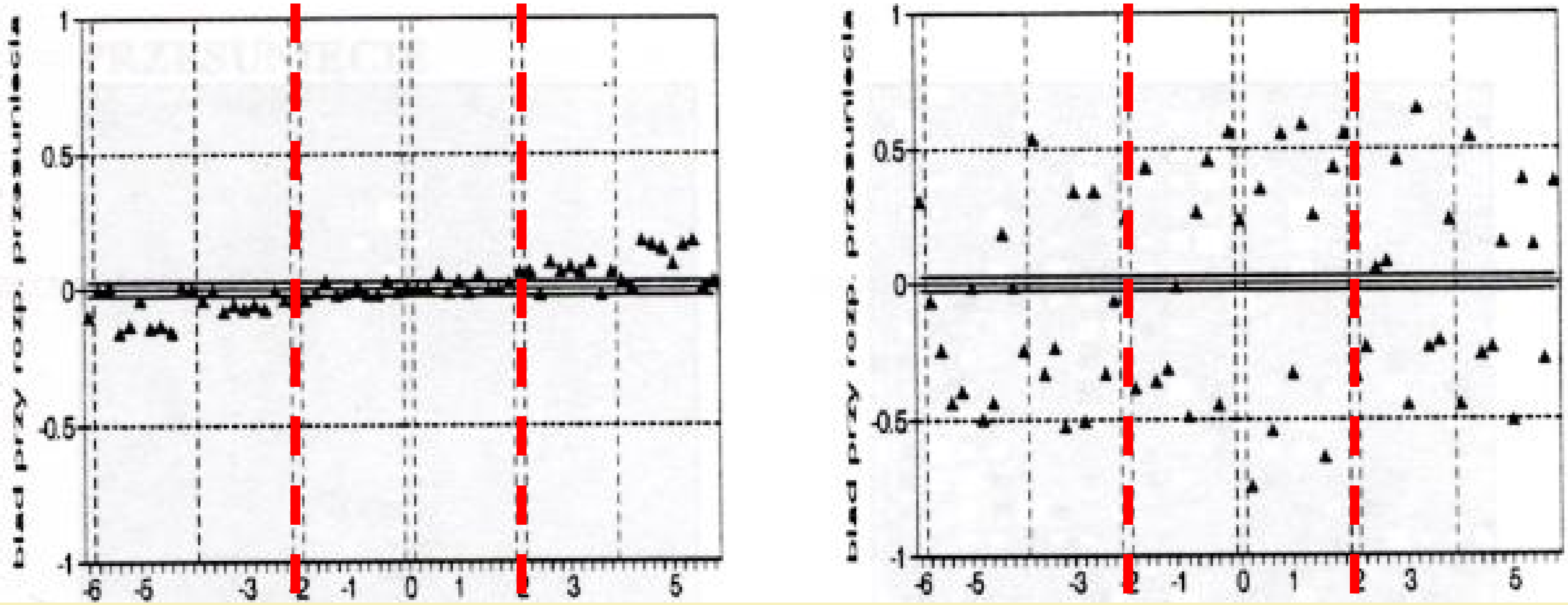
UCZENIE SIECI JEDNOWARSTWOWEJ PRZYKŁAD

WAGI PO NAUCZENIU NA PAMIĘĆ
(sieć w sposób idealny rozpoznaje ciąg uczący)





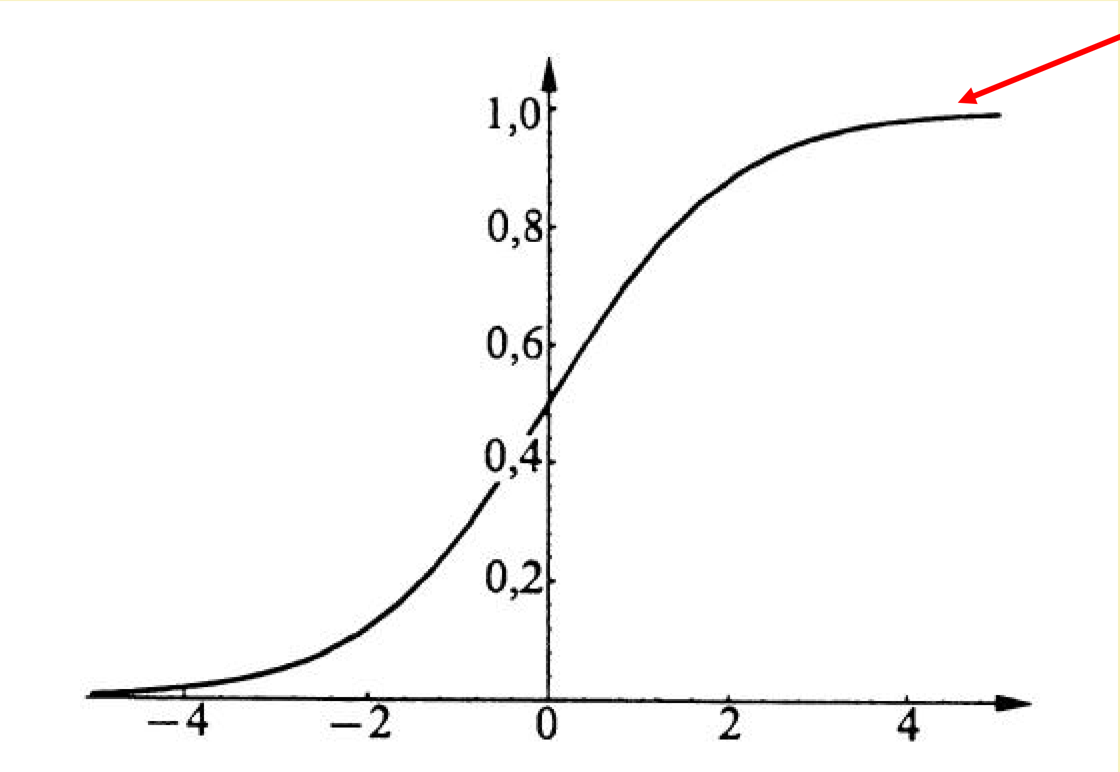
UCZENIE SIECI JEDNOWARSTWOWEJ
ROZPOZNANIE PRZY UCZENIU NA PAMIĘĆ



PRZESUNIĘCIE

MOŻLIWE PRZYCZYNY „PARALIŻU” SIECI

Niekorzystny punkt pracy

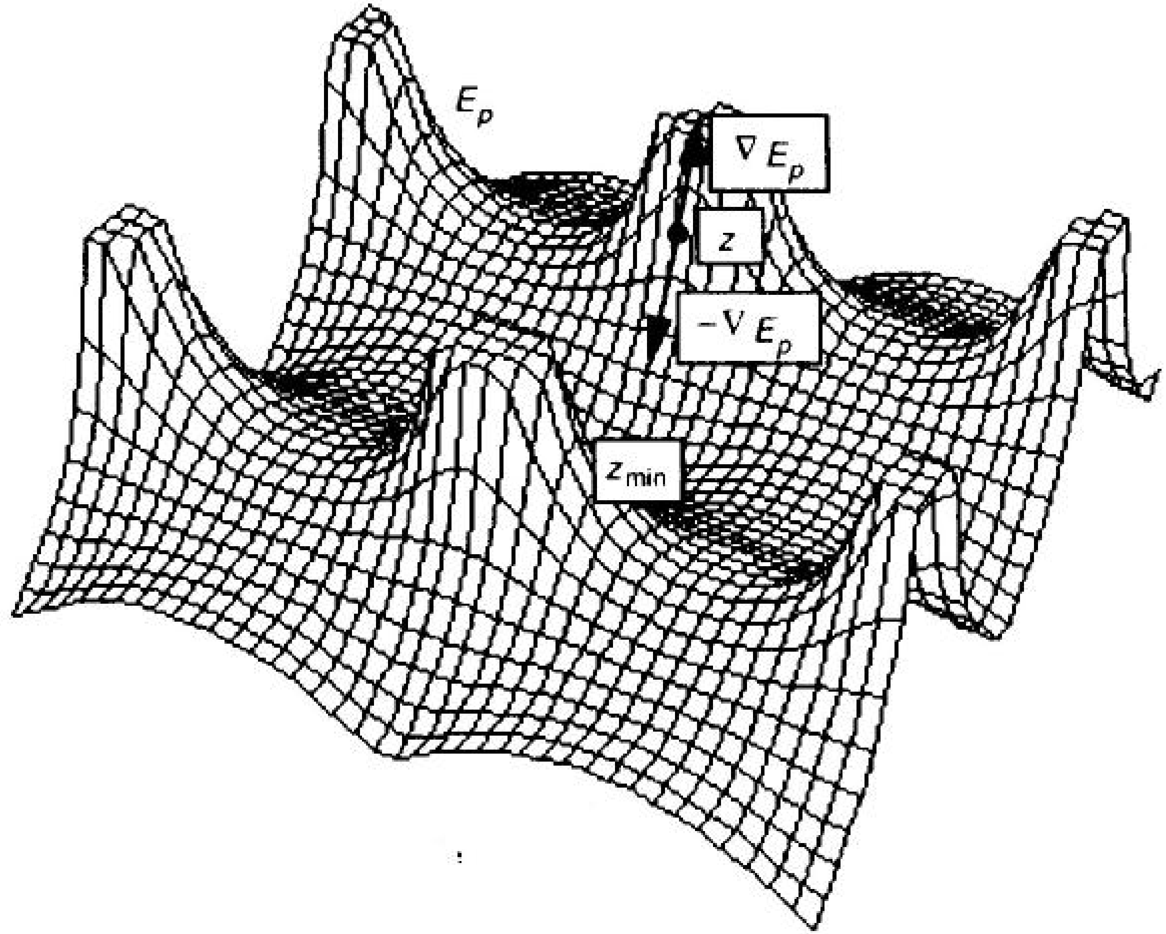


Rozwiązanie:

właściwa inicjalizacja wag

MINIMA LOKALNE

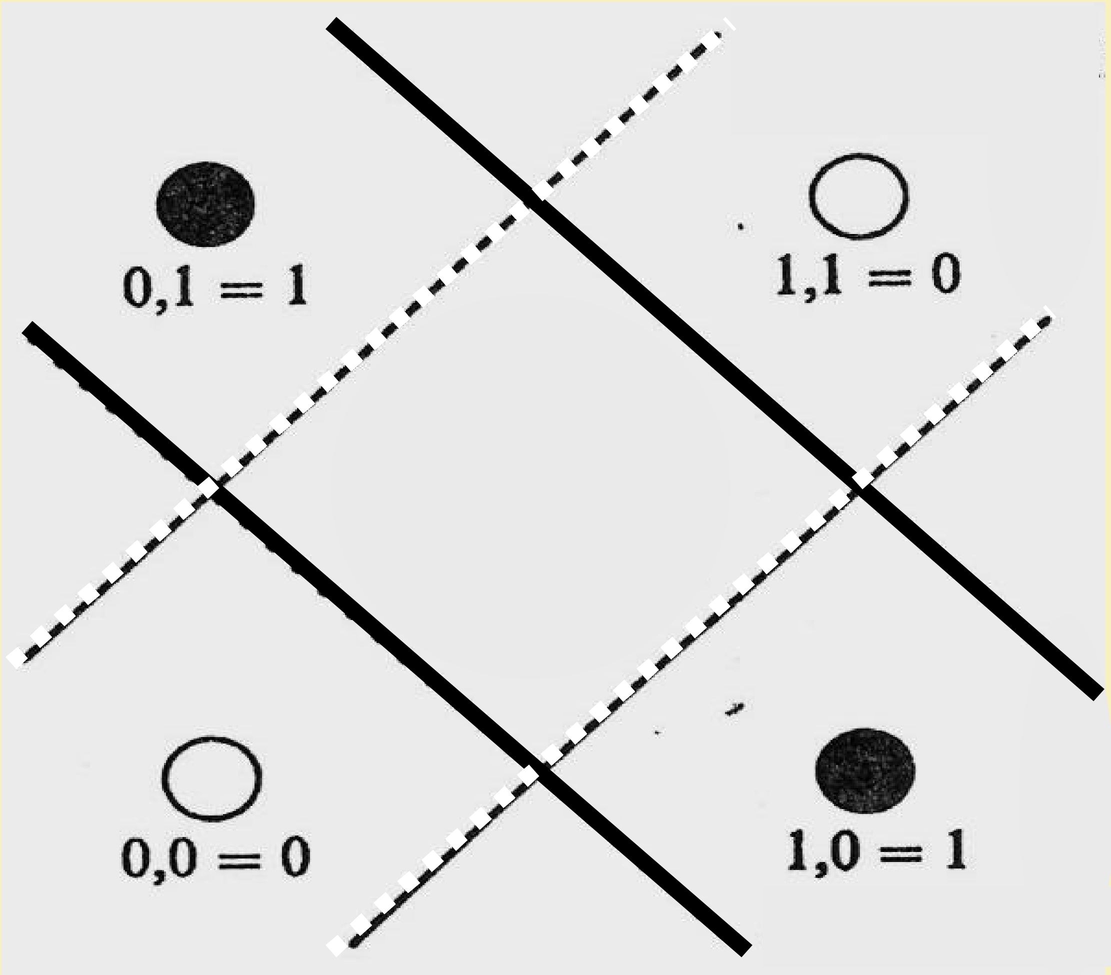
Przykładowy „krajobraz” funkcji błędu



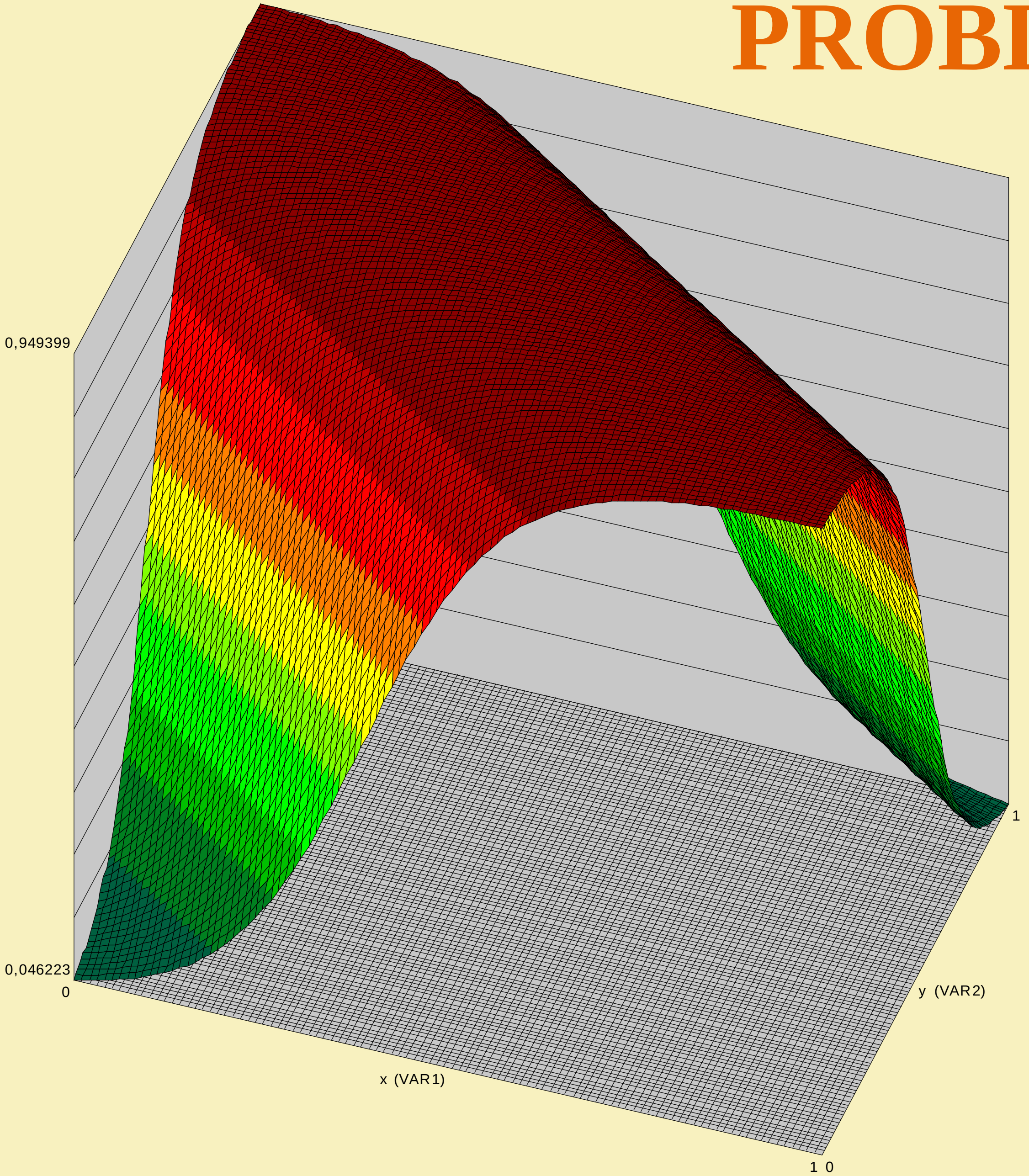
Rozwiązania:

- **Wprowadzenie „bezwładności”;**
- **Metoda symulowanego wyżarzania;**

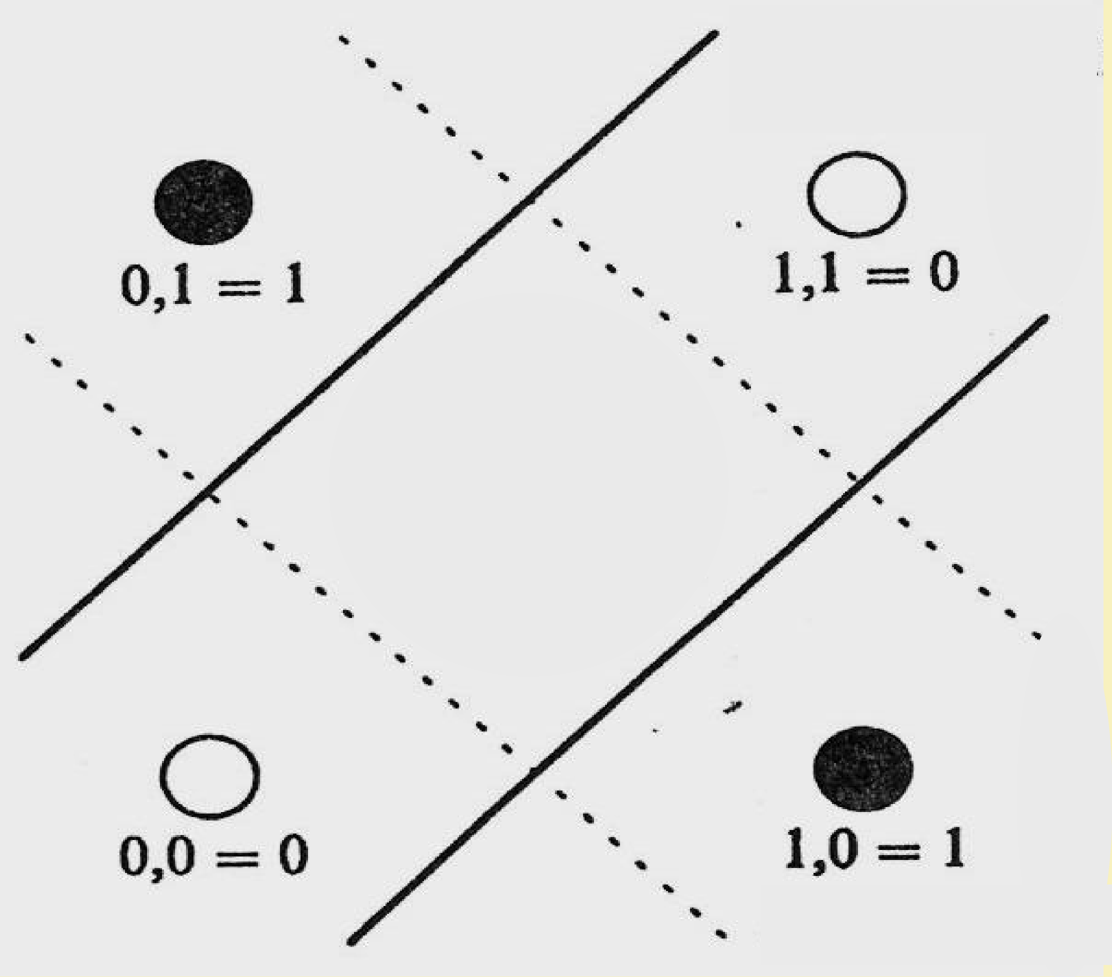
PROBLEM XOR



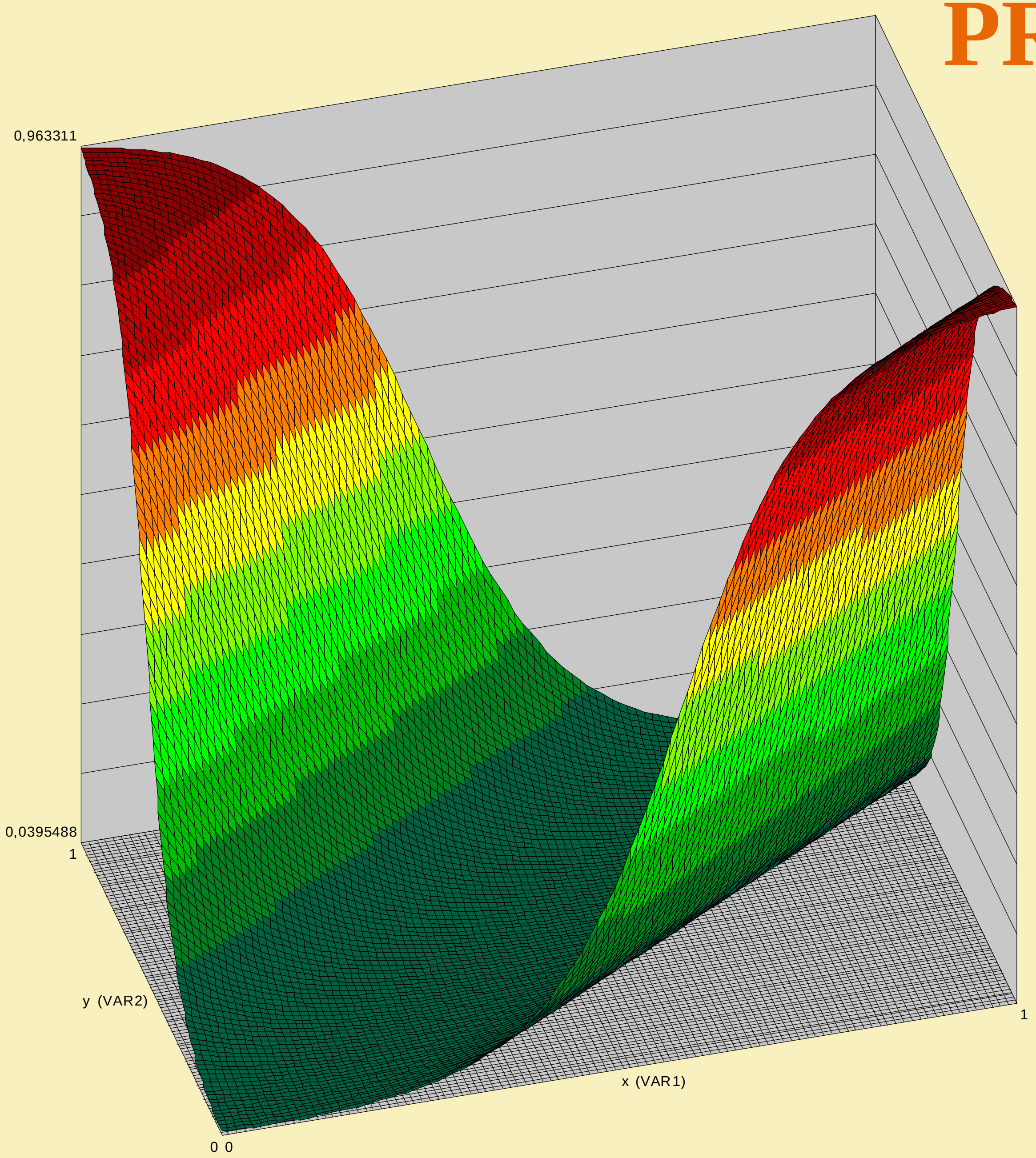
PROBLEM XOR



PROBLEM XOR

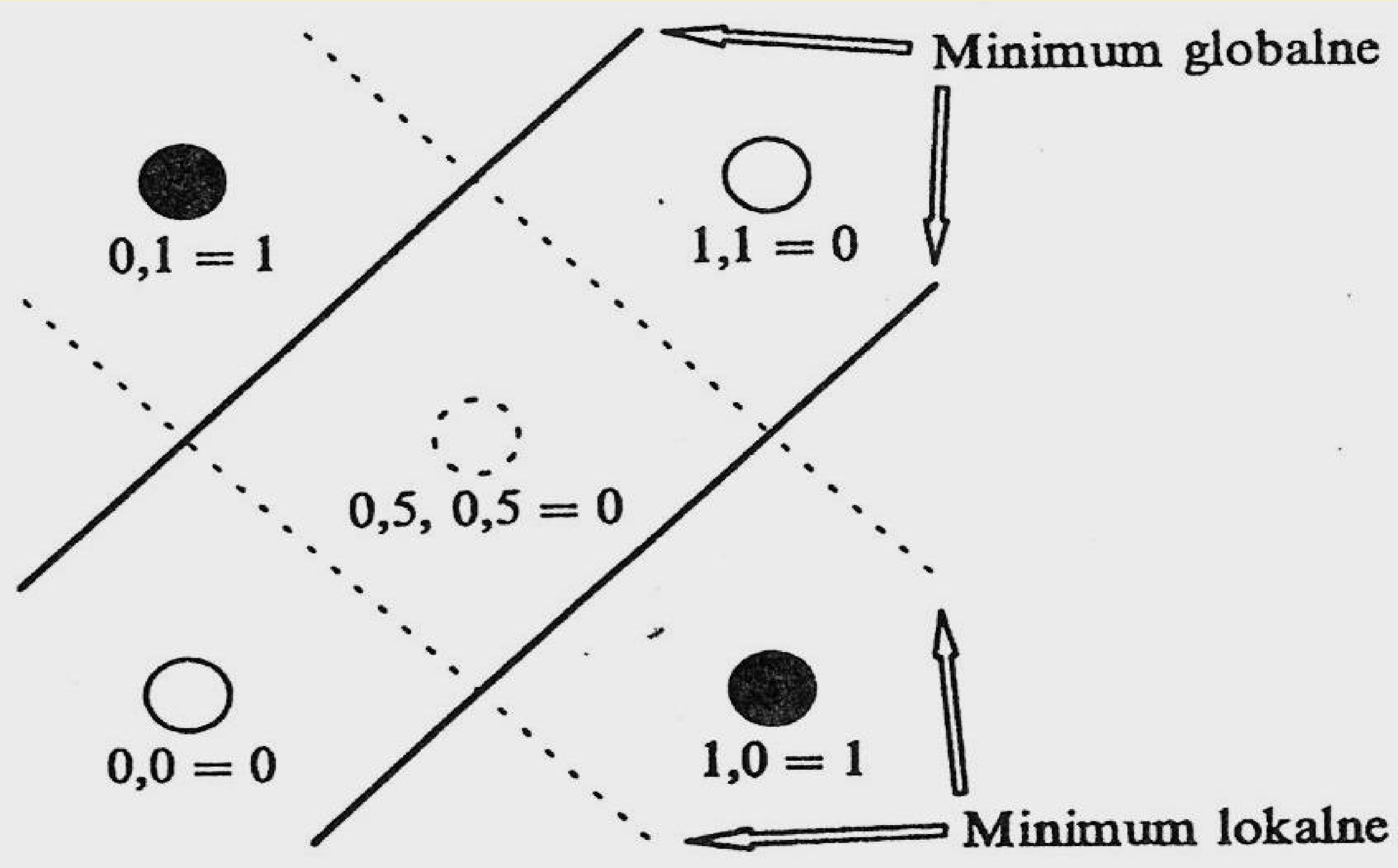


PROBLEM XOR



MINIMA LOKALNE

PROBLEM XOR



PODSTAWOWA MODYFIKACJA ALGORYTMU WSTECZNEJ PROPAGACJI BŁĘDÓW

METODA MOMENTUM

$$w_{ik}^{(m)(j)} = w_{ik}^{(m)(j)} + \eta_1 \; \delta_k^{(m)(j)} \; x_i^{(m)(j)} + \eta_2 \; \Delta w_{ik}^{(m)(j-1)}$$

gdzie:

η_1 - współczynnik szybkiego uczenia (learning rate)
η_2 - współczynnik momentum (bezwładność uczenia)
$\Delta w_{ik}^{(m)(j-1)}$ - zmiana wagi z poprzedniego kroku

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + \eta_1 \, \delta_i^{(k)}(n) \, x_j^{(k)} + \eta_2 \, [w_{ij}^{(k)}(n) - w_{ij}^{(k)}(n-1)]$$

ODRĘCZNA NOTATKA PROF. TADEUSIEWICZA

Współczynnik η_2 (momentum) jest miarą bezwładności procesu uczenia, chroniącą algorytm przed niestabilnym działaniem w warunkach silnie niemonotonicznej charakterystyki hiperpowierzchni błędu. W związku z tym wzrost wartości tego współczynnika prowadzi do wygładzania lokalnych oscylacji zmian współczynników wagowych i zwiększa prawdopodobieństwo osiągnięcia globalnego minimum funkcji błędu mimo obecności ~~lokalnych~~ pasożytniczych atraktorów w formie drobnych ale głębokich minimów lokalnych tej funkcji.

Modyfikacja algorytmu wstecznej propagacji błędów (1)

METODA MOMENTUM

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + \eta_1\,\delta_i^{(k)}(n)\,x_j^{(k)} + \eta_2\,[w_{ij}^{(k)}(n) - w_{ij}^{(k)}(n-1)]$$

INNA WERSJA METODY MOMENTUM

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + (1-\eta)\,\delta_i^{(k)}(n)\,x_j^{(k)} + \eta\,[w_{ij}^{(k)}(n) - w_{ij}^{(k)}(n-1)]$$

KOREKTA WAG W DWÓCH ETAPACH

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + \eta_1\,\delta_i^{(k)}(n)\,x_j^{(k)}$$

$$w_{ij}^{(k)}(n+2) = w_{ij}^{(k)}(n+1) + \eta_2\,[w_{ij}^{(k)}(n+1) - w_{ij}^{(k)}(n-1)]$$

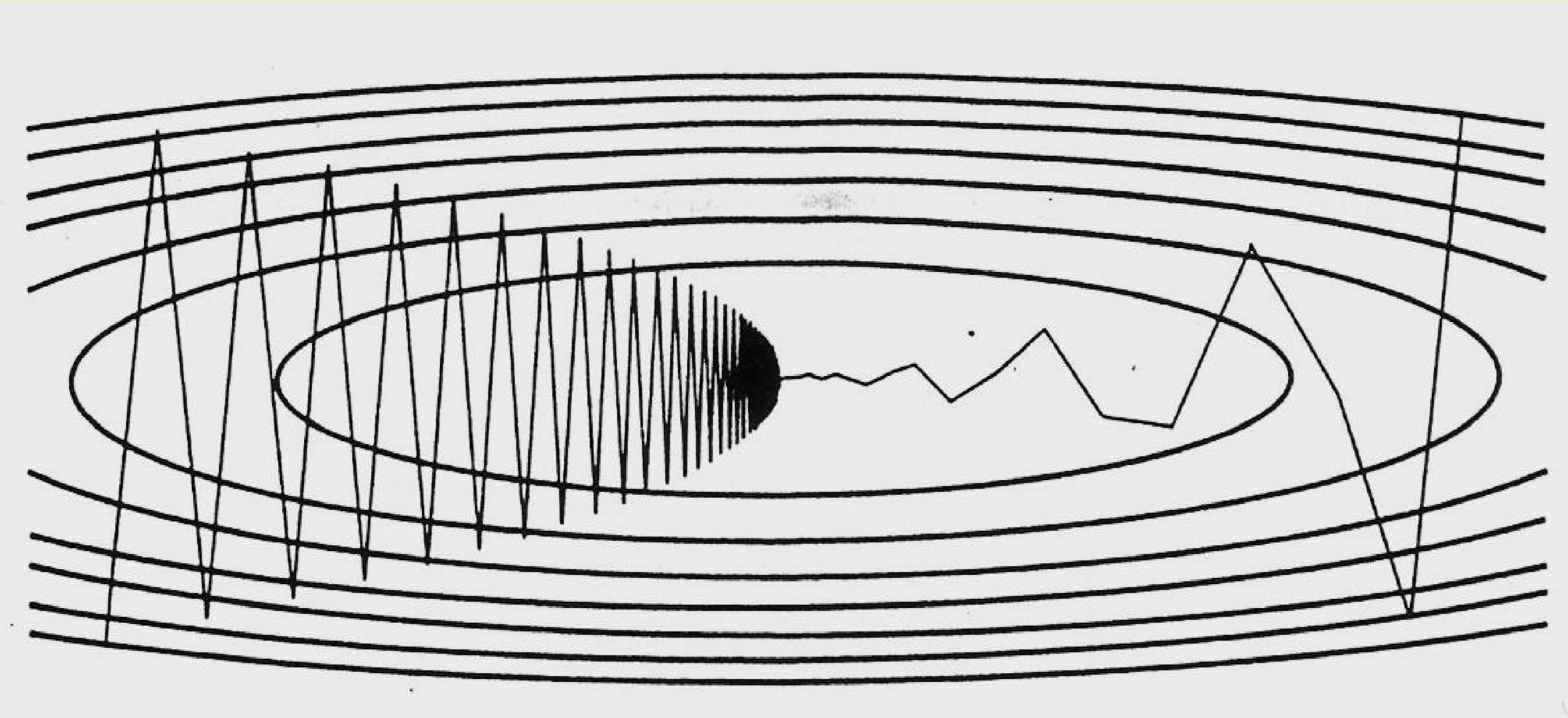
BACKPROPAGATION
WSPÓŁCZYNNIKI UCZENIA

Poniższa tablica podsumowuje liczbę kroków dla różnych wartości: η_1 η_2

Metoda propagacji wstecznej	momentum h_2	szybkość uczenia h_1	LICZBA EPOK
Bez momentu	0,0	0,5	10 367
Z momentem	0,5	0,5	5 180
	0,9	0,5	1 007
	0,9	0,05	10 339

METODA MOMENTUM

Kolejne kroki przy minimalizacji funkcji metodą gradientową bez i z momentum.



METODY ZE ZMIENNYMI WSPÓŁCZYNNIKAMI

Optymalne wartości współczynnika uczenia mogą być różne dla róznych iteracji. Są one zależne od hiperpowierzchni funkcji błędu w przestrzeni wag i od punktu trajektorii, w której aktualnie znajduje się sieć poruszając sie po tej hiperpowierzchni.

METODY ZE ZMIENNYMI WSPÓŁCZYNNIKAMI

Zależność współczynnika uczenia od kształtu powierzchni funkcji błędu.

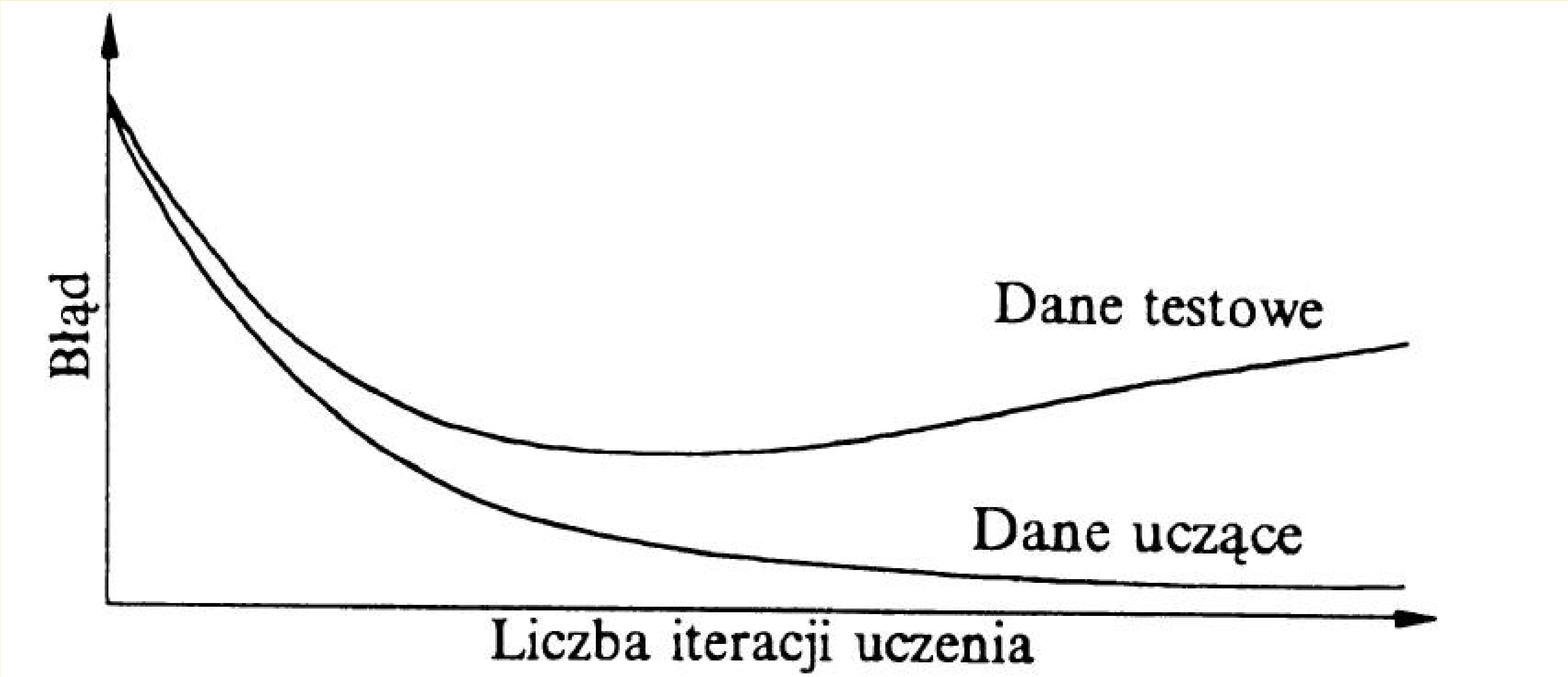
Ukształtowanie powierzchni	Wartość korzystna	Wartość niekorzystna
płaskowyż	*duża* efekt: większe poprawki wag powodują szybsze przesuwanie w kierunku minimum	*mała* efekt: niewielkie zmiany wag w poszczególnych krokach, wolny przebieg procesu minimalizacji
wąwóz	*mała* efekt: trajektoria będzie przebiegać w dół dokładnie po linii najszybszego spadku	*duża* efekt: możliwe oscylacje trajektorii pomiędzy jedną, a drugą ścianą wąwozu

METODY ZE ZMIENNYMI WSPÓŁCZYNNIKAMI

Zależność współczynnika momentu od kształtu powierzchi funkcji błędu.

Ukształtowanie powierzchni	Wartość korzystna	Wartość niekorzystna
płaskowyż	*stosunkowo duża* efekt: nadanie procesowi minimalizacji dodatkowego „pędu", zwiększenie efektywnego tempa uczenia	*mała* efekt: jedynie znikomy wpływ na poprawę efektywnego tempa uczenia
wąwóz	*niewielka* efekt: tłumienie oscylacji — rola filtru LP dla zmian składowych gradientu	*zbyt duża* efekt: odejście trajektorii zbyt daleko od linii najszybszego spadku, możliwy wzrost funkcji błędu w kolejnych krokach

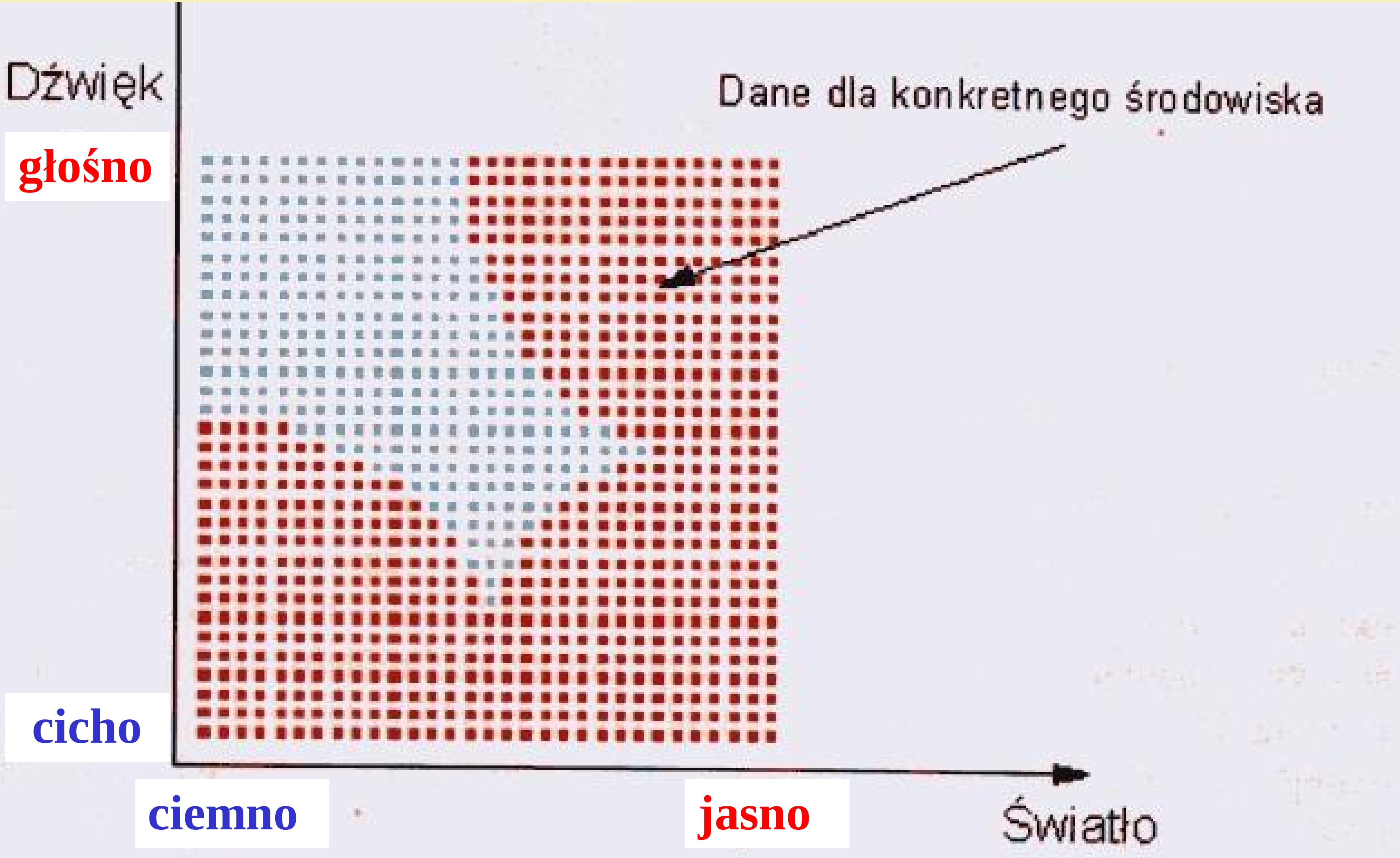
PRZETRENOWANIE SIECI



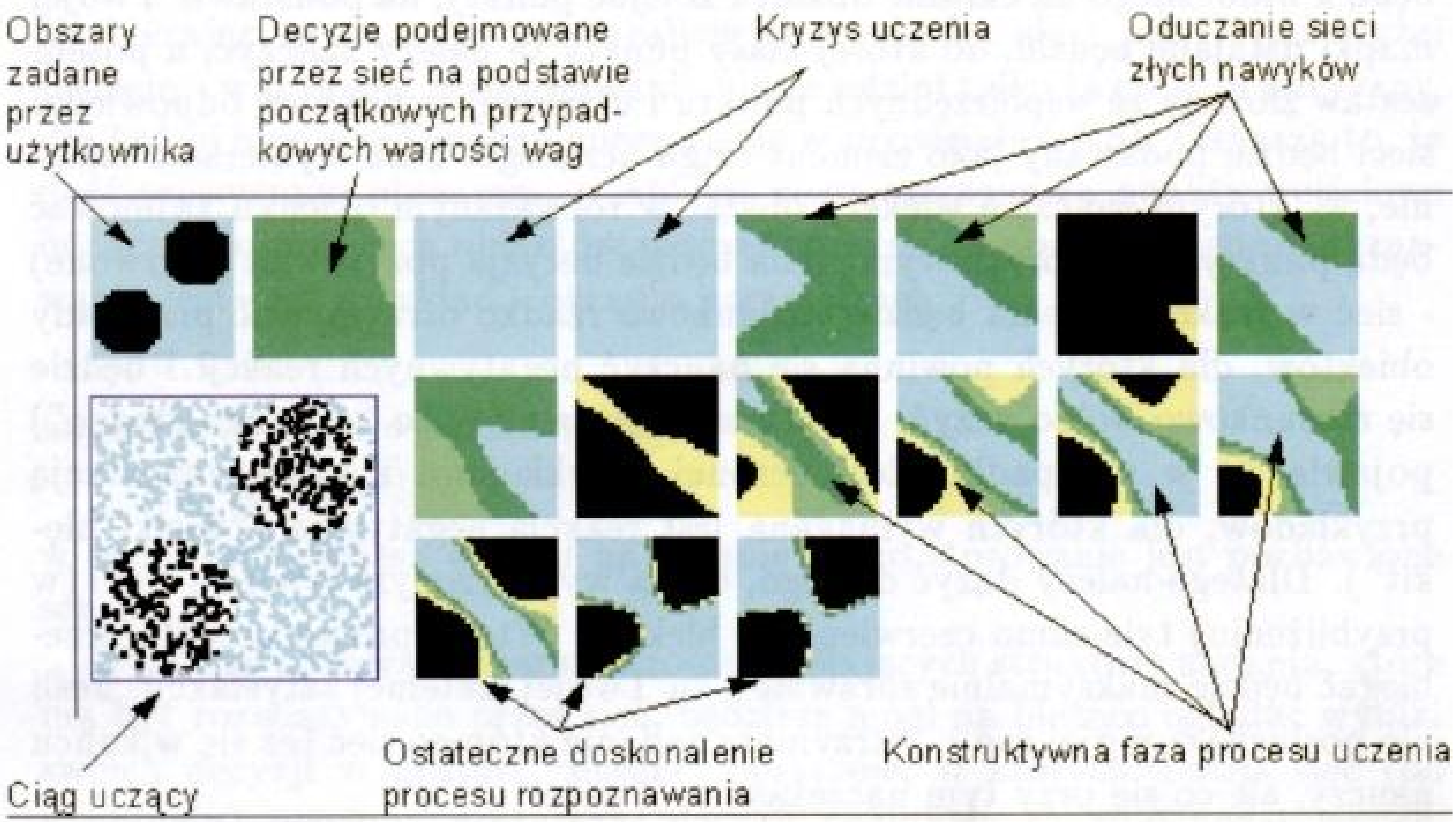
Rozwiązanie:

wprowadzenie zbioru walidacyjnego
(oprócz zbioru uczącego i testowego)

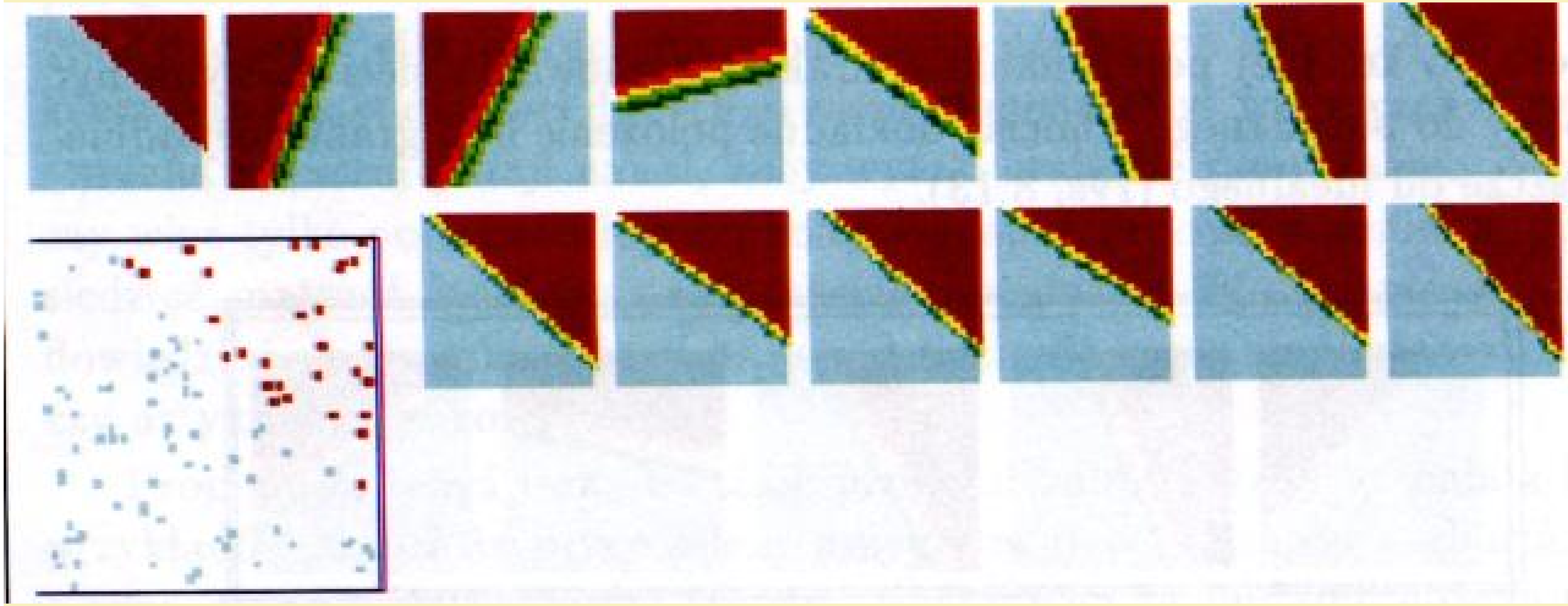
PROSTY PRZYKŁAD
BAŁWANKI



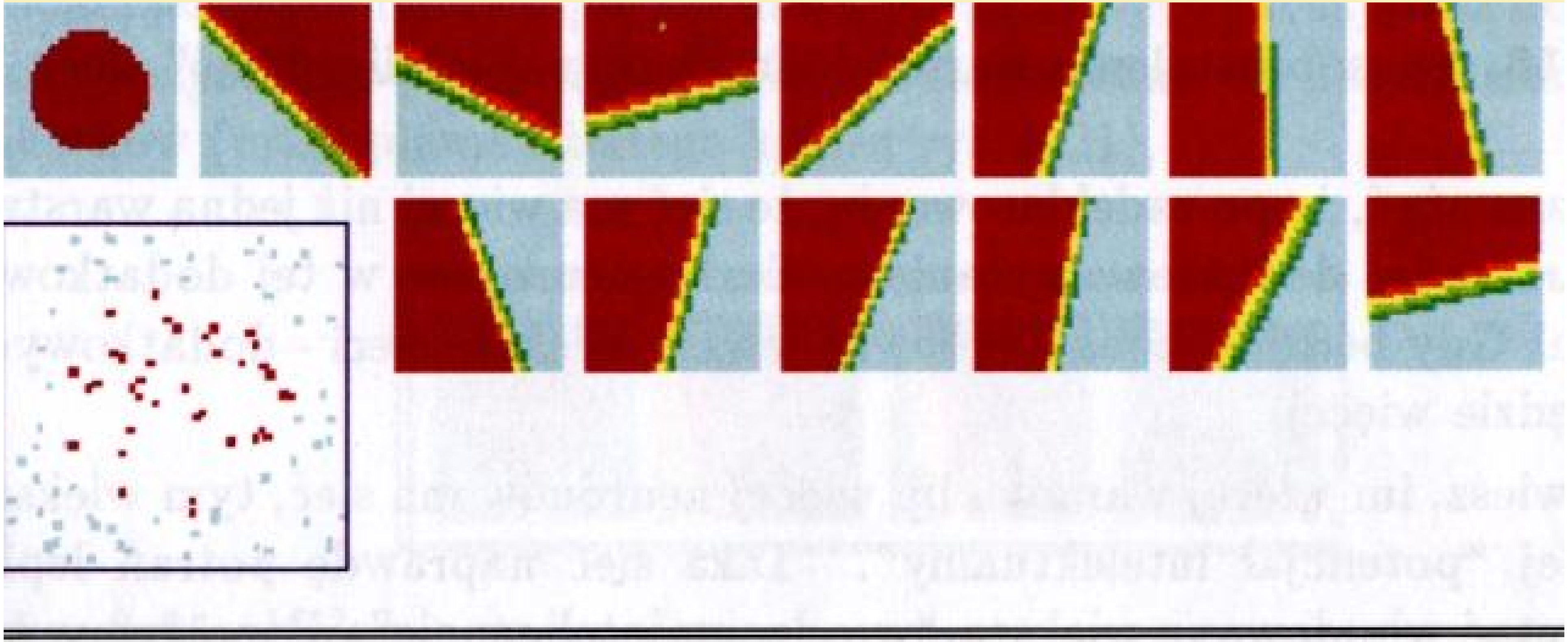
OPIS UCZENIA SIECI



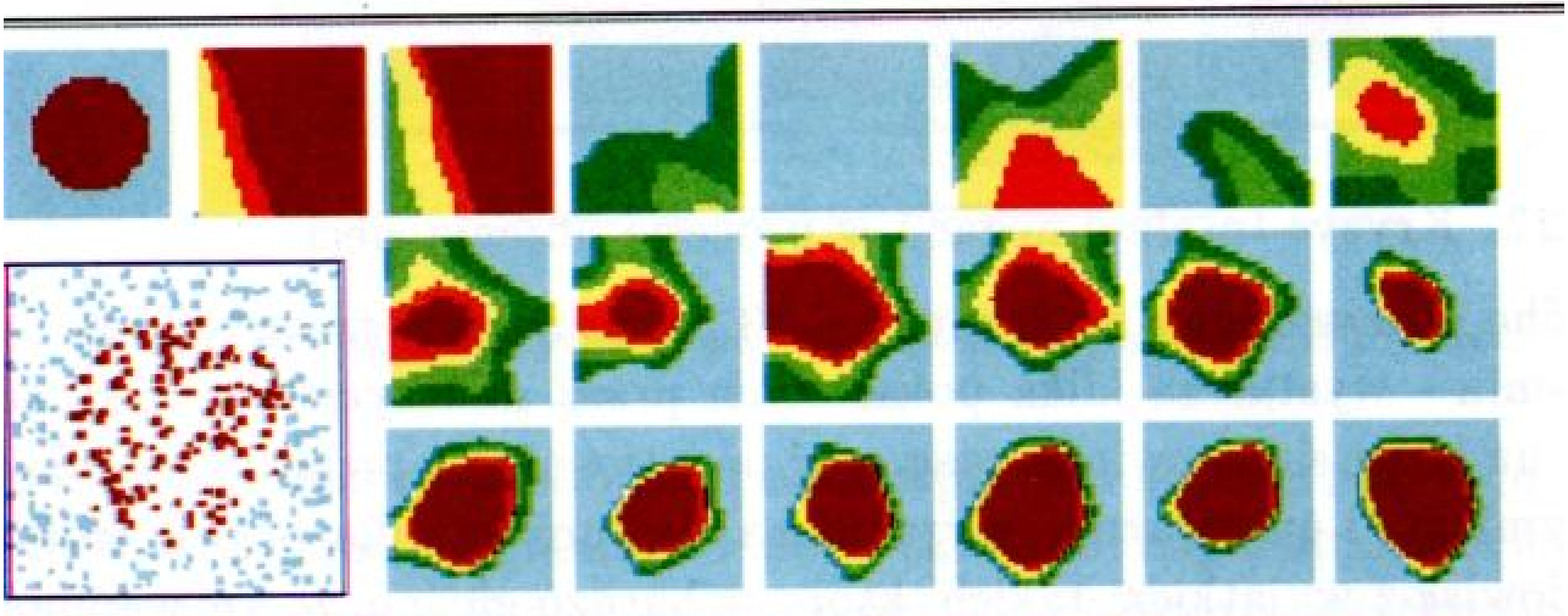
SIEĆ JEDNOWARSTWOWA
PROSTE ZADANIE



SIEĆ JEDNOWARSTWOWA
TRUDNE ZADANIE

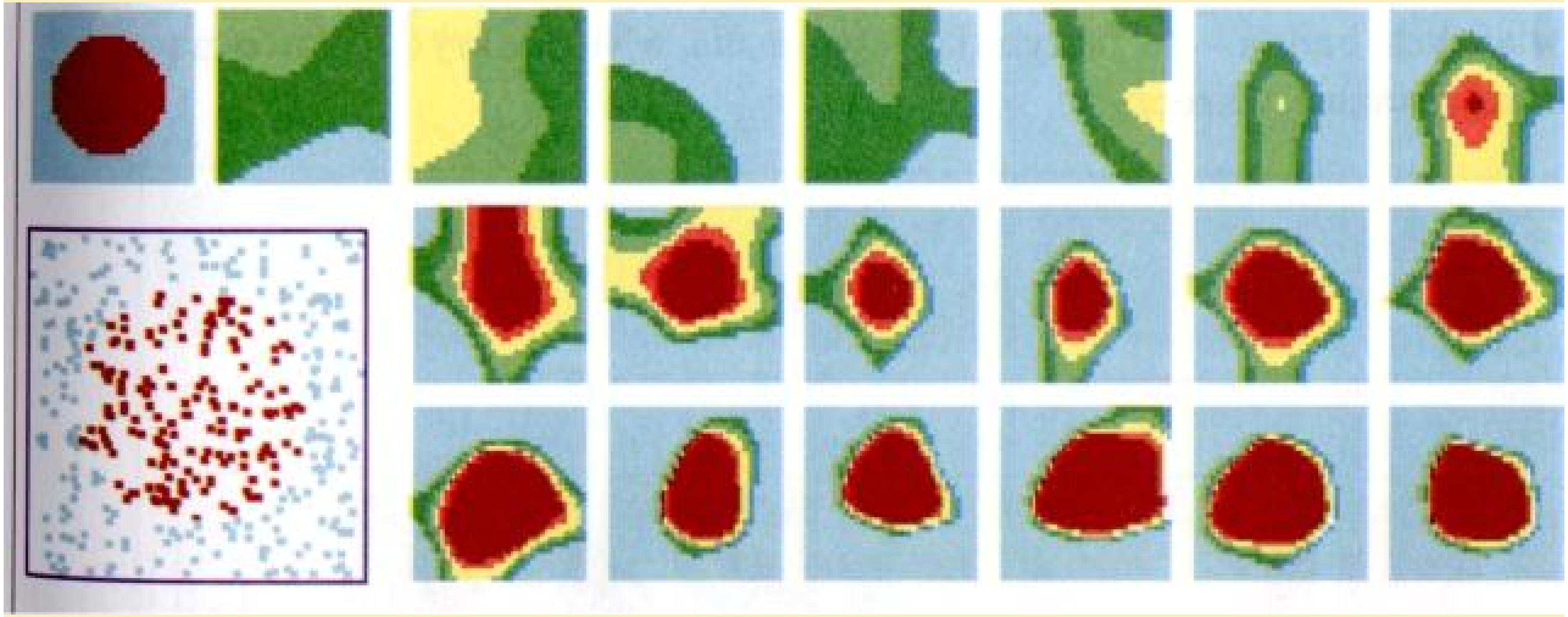


SIEĆ DWUWARSTWOWA
TRUDNE ZADANIE



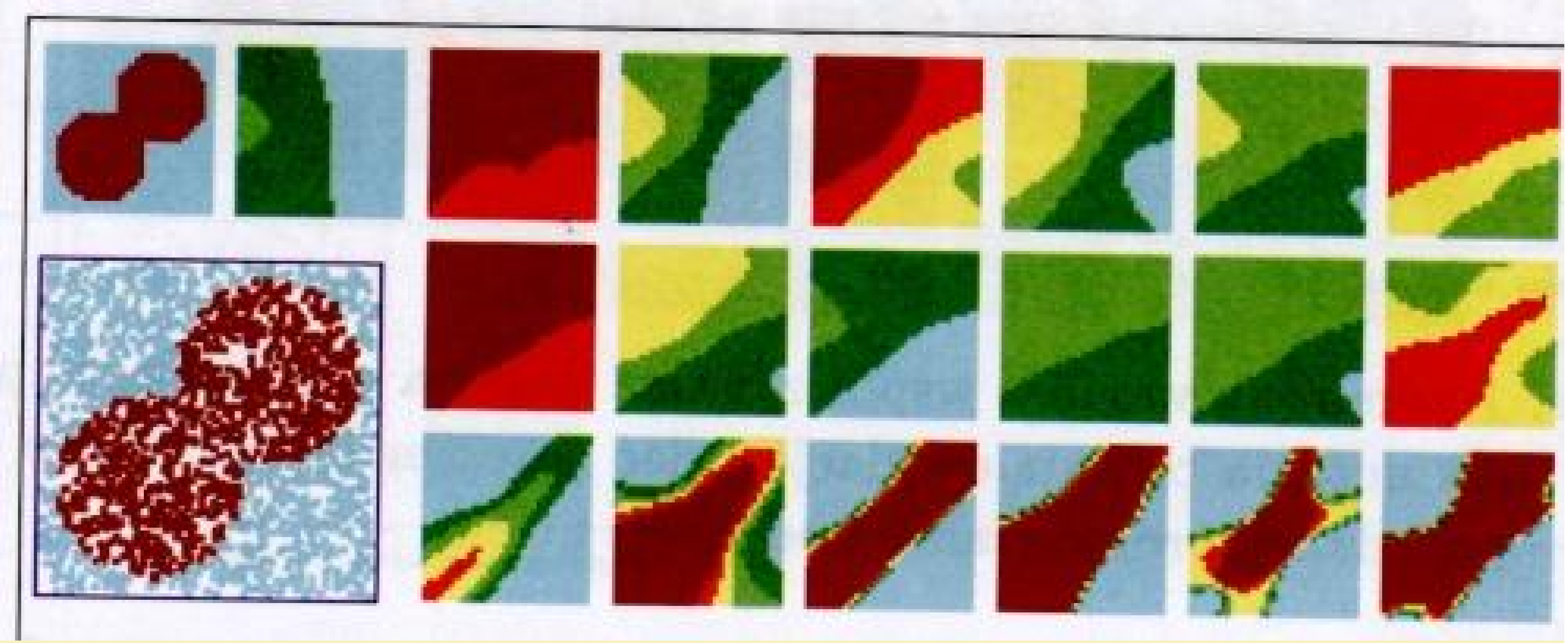
SIEĆ ENTUZJASTYCZNA !!!

SIEĆ DWUWARSTWOWA
TRUDNE ZADANIE

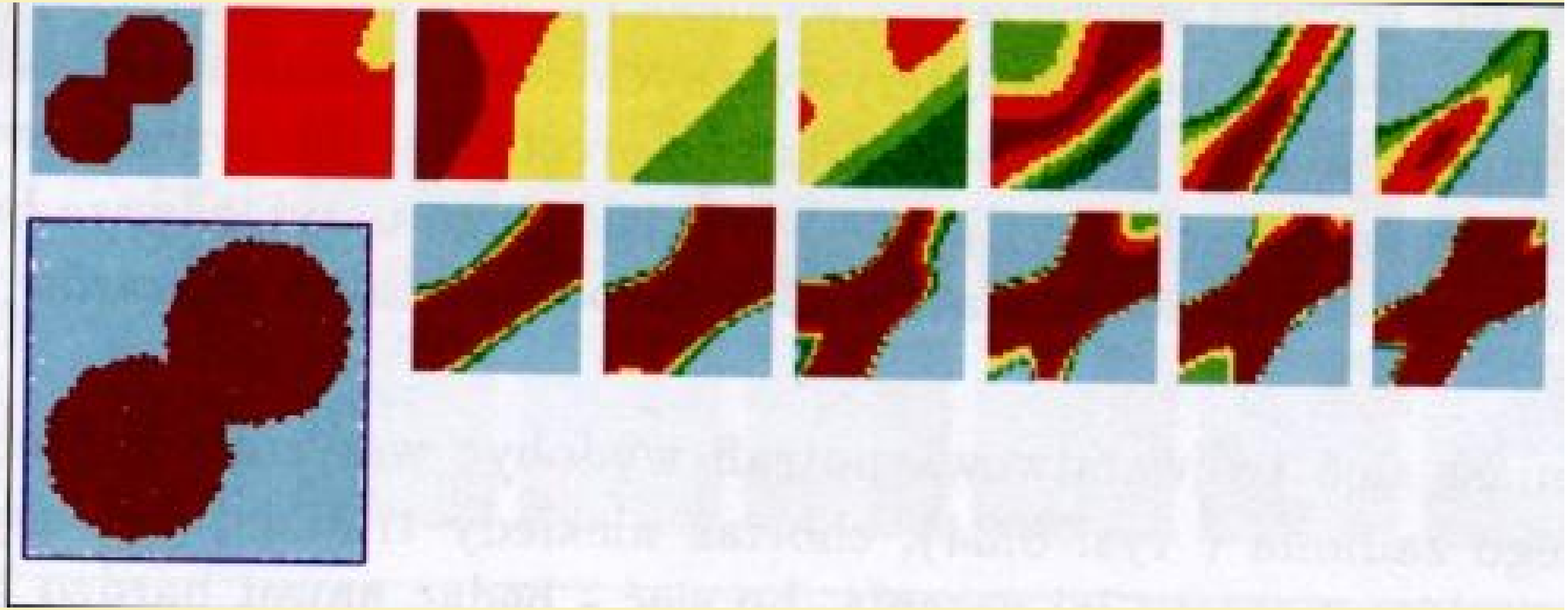


SIEĆ MELANCHOLIJNA !!!

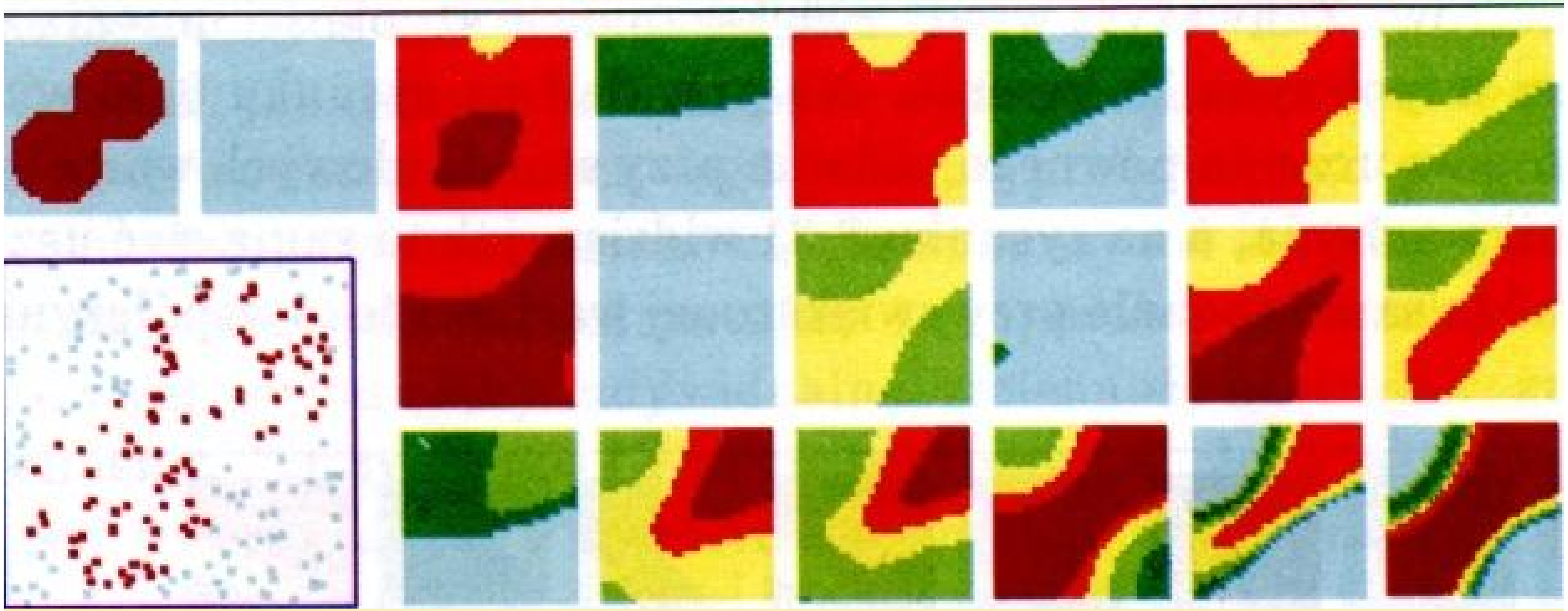
STANDARDOWY WSPÓŁCZYNNIK UCZENIA



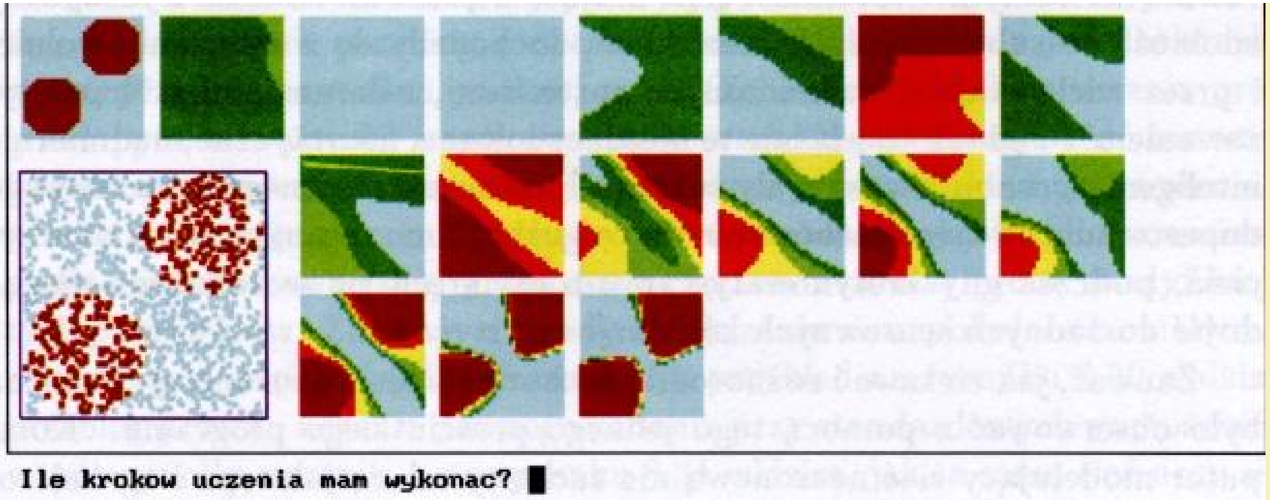
ZWIĘKSZONY WSPÓŁCZYNNIK SZYBKOŚCI UCZENIA



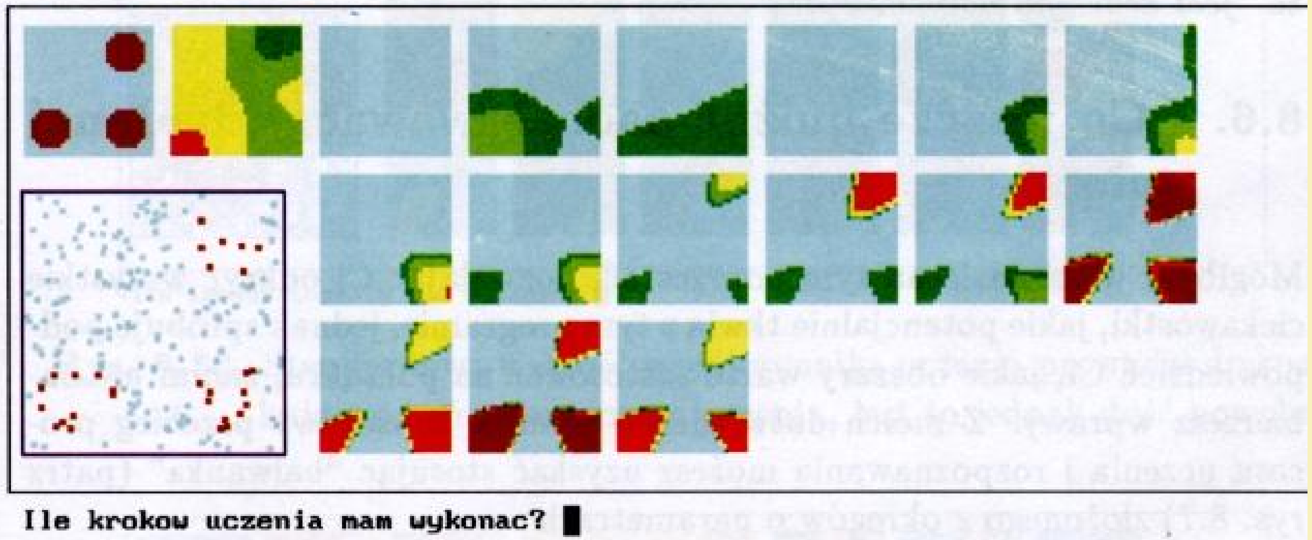
NADMIERNIE DUŻY WSPÓŁCZYNNIK SZYBKOŚCI UCZENIA



SIEĆ DWUWARSTWOWA

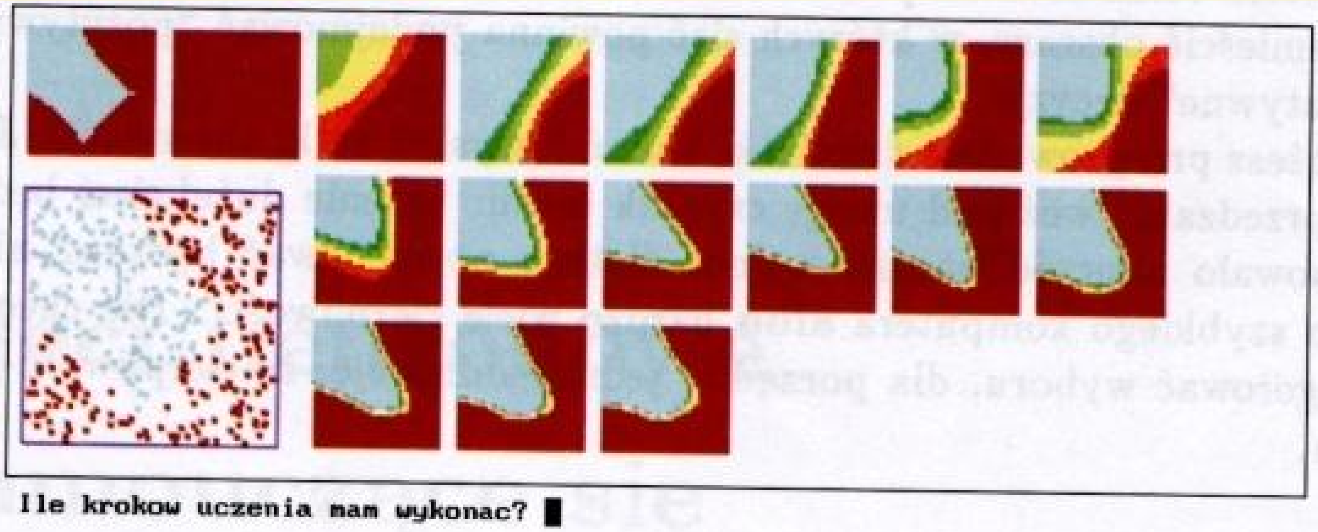


Rys. 8.28. Przykład zadania , które nie każda dwuwarstwowa sieć zdoła rozwiązać

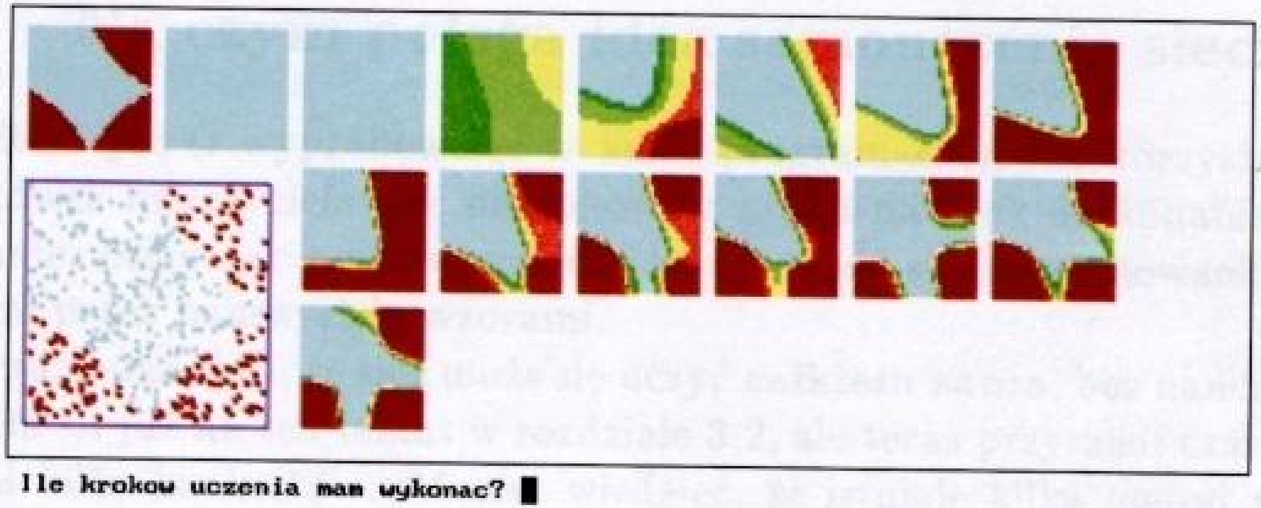


Rys. 8.29. Zadanie o dużym stopniu trudności wyłącznie dla trójwarstwowej sieci

SIEĆ TRÓJWARSTWOWA

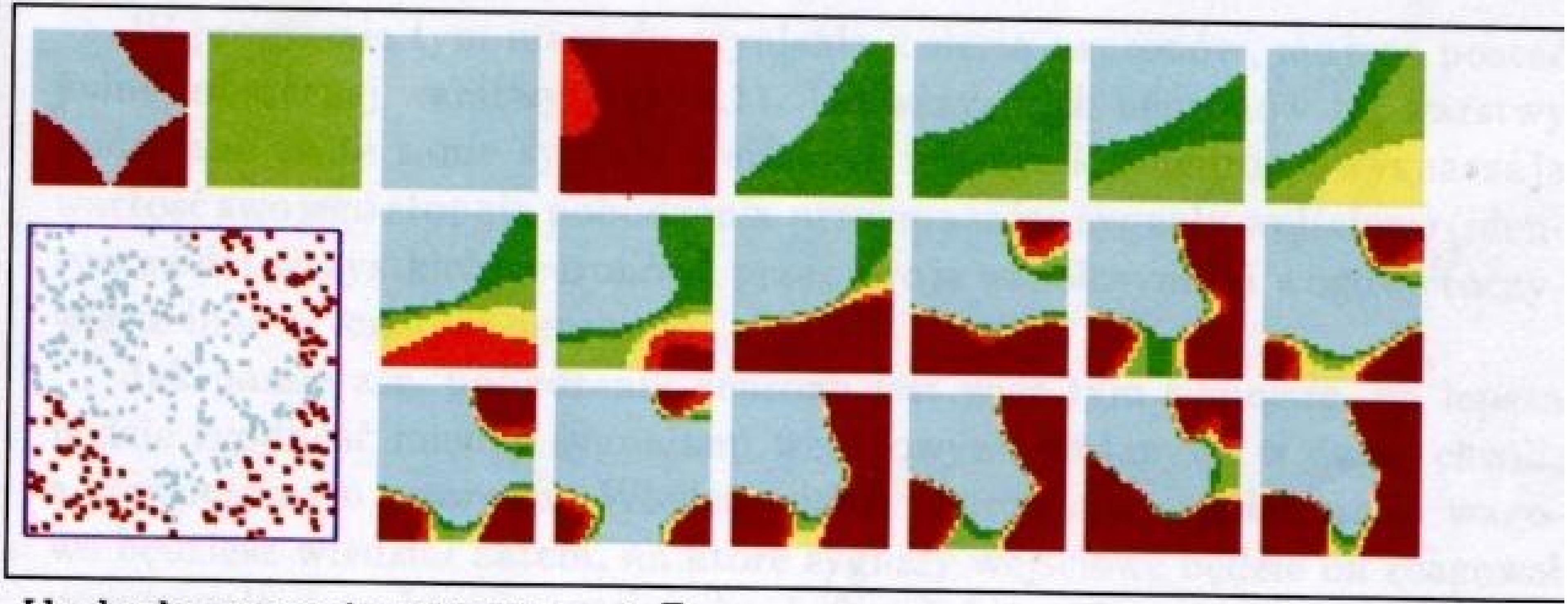


Rys. 8.33. Słabe dopasowanie obszarów w przypadku sieci dwuwymiarowej



Rys. 8.34. Poprawne i szybkie rozwiązanie trudnego zadnia przy użyciu sieci trójwarstwowej

SIEĆ TRÓJWARSTWOWA



Ile krokow uczenia mam wykonac?

Rys. 8.35. Znamiona niestabilności uczenia pojawiające się w sieci trójwarstwowej