Akademia Górniczo-Hutnicza
WIMIiP, Inżynieria Obliczeniowa
grupa laboratoryjna 01
Agnieszka Kępka

29.11.2018,Kraków

Podstawy Sztucznej Inteligencji

Sprawozdanie numer 3

Budowa i działanie sieci wielowarstwowej typu feedforward

Cel ćwiczenia:

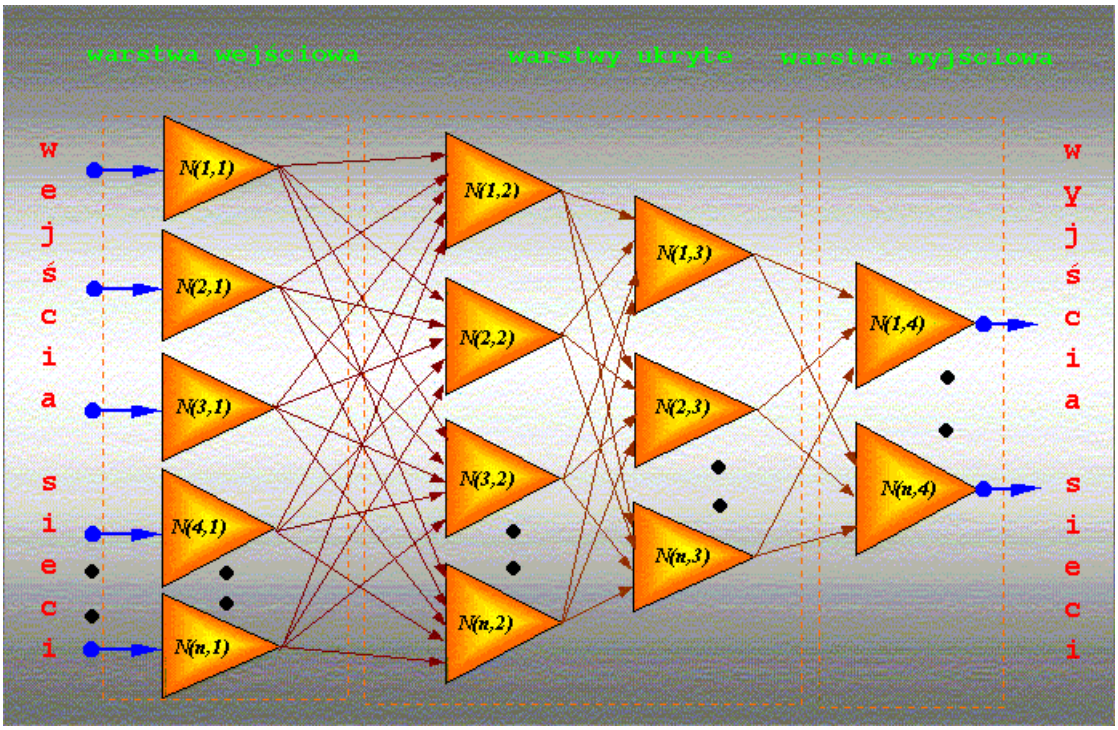
Celem ćwiczenia jest poznanie budowy i działania wielowarstwowych sieci neuronowych poprzez uczenie kształtu funkcji matematycznej z użyciem algorytmu wstecznej propagacji błędu.

Zadania do wykonania:

- Wygenerowanie danych uczących i testujących dla funkcji Rastrigin 3D dla danych wejściowych z przedziałów od -2 do 2.
- Przygotowanie wielowarstwowej sieci oraz algorytmu wstecznej propagacji błędu.
- Uczenie sieci dla różnych współczynników uczenia i bezwładności
- Testowanie sieci.

Zagadnienia teoretyczne:

Sieci wielowarstwowe to odpowiednio połączone warstwy neuronów zwykle o nieliniowych funkcjach aktywacji, aczkolwiek czasami w warstwie wyjściowej pojawiają się neurony liniowe. Sieci wielowarstwowe muszą posiadać minimalnie dwie warstwy neuronów: warstwę wejściową i warstwę wyjściową pomiędzy którymi może być jedna lub więcej warstw ukrytych. W każdej warstwie może występować różna ilość neuronów: W warstwie wejściowej odpowiada ona zwykle ilości parametrów opisujących dane wejściowe, w warstwie wyjściowej np. ilości klas, natomiast ilość neuronów w warstwach ukrytych odpowiada za możliwości modelu. Neurony łączą się tylko pomiędzy (zwykle) sąsiednimi warstwami, zaś wewnątrz warstw nie występują połączenia pomiędzy neuronami. Neurony zwykle łączymy pomiędzy sąsiednimi warstwami na zasadzie każdy z każdym.



Uczenie sieci neuronowych - wymuszenie na niej określonej reakcji na zadane sygnały wejściowe. Uczenie jest konieczne tam, gdzie brak jest informacji doświadczalnych o powiązaniu wejścia z wyjściem lub jest ona niekompletna, co uniemożliwia szczegółowe zaprojektowanie sieci. Uczenie może być realizowane krok po kroku lub poprzez pojedynczy zapis. Istotnym czynnikiem przy uczeniu jest wybór odpowiedniej strategii (metody) uczenia. Wyróżnić możemy dwa podstawowe podejścia: uczenie z nauczycielem (supervised learning) i uczenie bez nauczyciela (unsupervised learning).

Uczenie metodą wstecznej propagacji błędów - uczenie z nauczycielem - pierwszą czynnością w procesie uczenia jest przygotowanie dwóch ciągów danych: uczącego i weryfikującego. Ciąg uczący jest to zbiór takich danych, które w miarę dokładnie charakteryzują dany problem. Jednorazowa porcja danych nazywana jest wektorem uczącym. W jego skład wchodzi wektor wejściowy czyli te dane wejściowe, które podawane są na wejścia sieci i wektor wyjściowy czyli

takie dane oczekiwane, jakie sieć powinna wygenerować na swoich wyjściach. Po przetworzeniu wektora wejściowego, nauczyciel porównuje wartości otrzymane z wartościami oczekiwanymi i informuje sieć czy odpowiedź jest poprawna, a jeżeli nie, to jaki powstał błąd odpowiedzi. Błąd ten jest następnie propagowany do sieci ale w odwrotnej niż wektor wejściowy kolejności (od warstwy wyjściowej do wejściowej) i na jego podstawie następuje taka korekcja wag w każdym neuronie, aby ponowne przetworzenie tego samego wektora wejściowego spowodowało zmniejszenie błędu odpowiedzi. Procedurę taką powtarza się do momentu wygenerowania przez sieć błędu mniejszego niż założony. Wtedy na wejście sieci podaje się kolejny wektor wejściowy i powtarza te czynności. Po przetworzeniu całego ciągu uczącego (proces ten nazywany jest epoką) oblicza się błąd dla epoki i cały cykl powtarzany jest do momentu, aż błąd ten spadnie poniżej dopuszczalnego.

Algorytm uczenia sieci wielowarstwowych metodą propagacji błędu:

1. Podajemy dane wejściowe.
2. Inicjujemy losowo, odpowiednio małe wagi.
3. Dla danego wektora uczącego obliczamy odpowiedź sieci (warstwa po warstwie).
4. Każdy neuron wyjściowy oblicza swój błąd, oparty na różnicy pomiędzy obliczoną odpowiedzią *y* oraz poprawną odpowiedzią *t*. Błędy propagowane są do wcześniejszych warstw.
5. Każdy neuron modyfikuje wagi na podstawie wartości błędu i wielkości przetwarzanych w tym kroku sygnałów.
6. Dla kolejnych wektorów uczących powtarzamy operacje od kroku 3. Gdy wszystkie wektory zostaną użyte, losowo zmieniamy ich kolejność i zaczynamy wykorzystywać powtórnie.
7. Jeśli średni błąd na danych treningowych przestanie maleć, przerywamy działanie algorytmu.

Funkcja obliczająca błąd:

$$d(w_1,\ldots,w_K)=\frac{1}{2}\left(f(w_1z_1+\ldots+w_Kz_K)-t\right)^2$$

Część Praktyczna:

Do realizacji zadania został wykorzystany pakiet MATLAB i narzędzie Neural Networking Training Tool.

Zadanie polegało na wygenerowaniu danych uczących i testujących dla funkcji Rastrign 3D dla danych wejściowych z przedziału [-2; 2]. Funkcja Rastrign3D przyjmowała następującą postać:

$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i)\right]$$

gdzie: A = 10, xi∈ [-5.12; 5.12] oraz dla f(0) = 0.

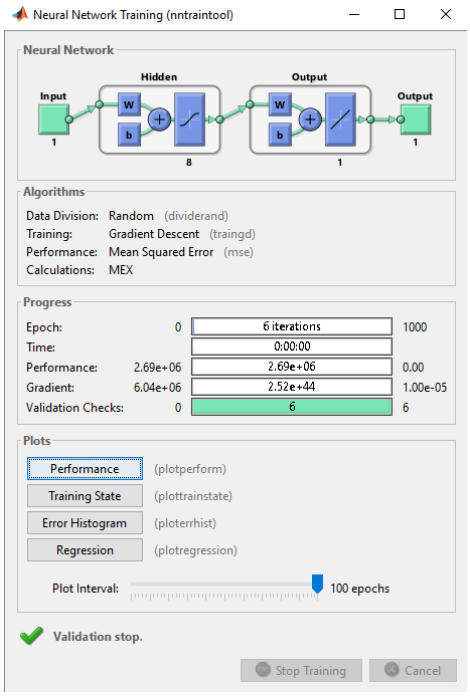
Danymi wejściowymi jest 21 elementowa tablica liczb zmiennoprzecinkowych znajdująca się w przedziale [-2,2], gdzie krok wraz z końcami przedziałów wynosi 0.2.
Jako dane wyjściowe przyjmuję 21 elementowa tablicę, która ma wyniki funkcji Rastrign 3D.

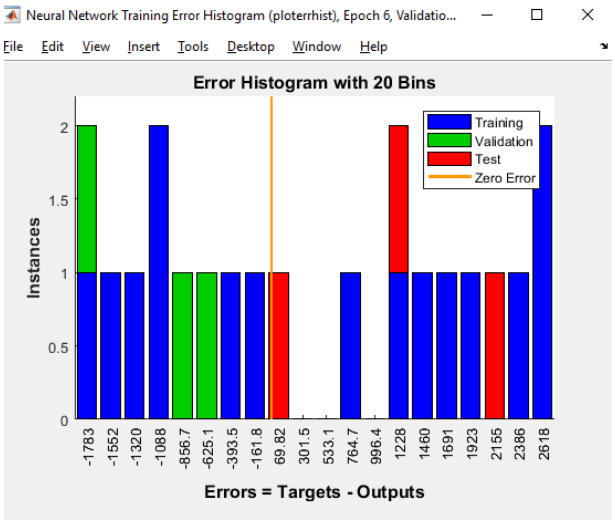
Sieć wielowarstwowa została zaimplementowana przez polecenie feedforwardnet(8), gdzie 8 to liczba warstw ukrytych, których liczba jest zmienna w zależności od potrzeb danego problemu. Pakiet Matlab przyjmuje domyślnie 10 warstw ukrytych ale ten problem nie jest na tyle złożony, aby wykorzystać aż tyle warstw. W przypadku użycia więcej niż dwóch warstw istnieje ryzyko przeuczenia sieci. Z tego względu ilość warstw sieci została zmieniona na dwie warstwy.

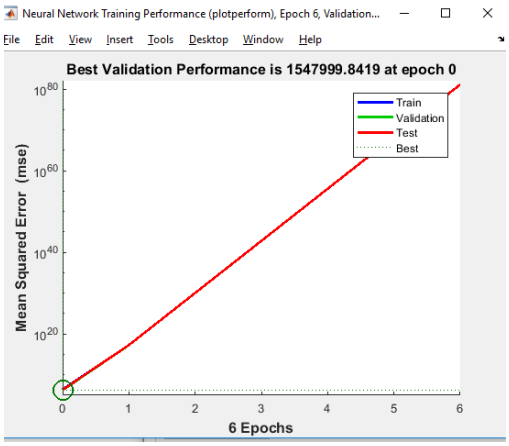
Współczynnik uczenia i bezwładności (momentum) wpływa na działanie sieci, z kolei by móc testować działanie sieci uczącej z algorytmem wstecznej propagacji i bez niego należy dodać parametr treningu net.trainFcn=’traingd’.

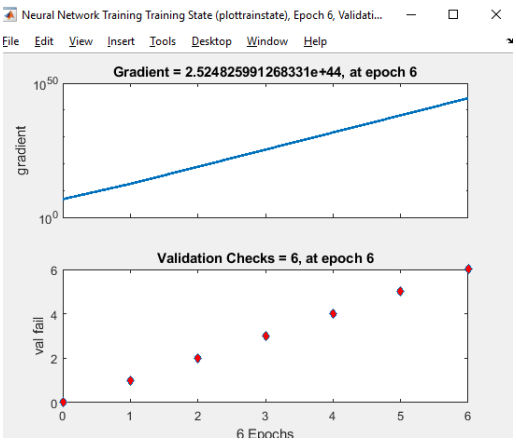
Analiza wyników:

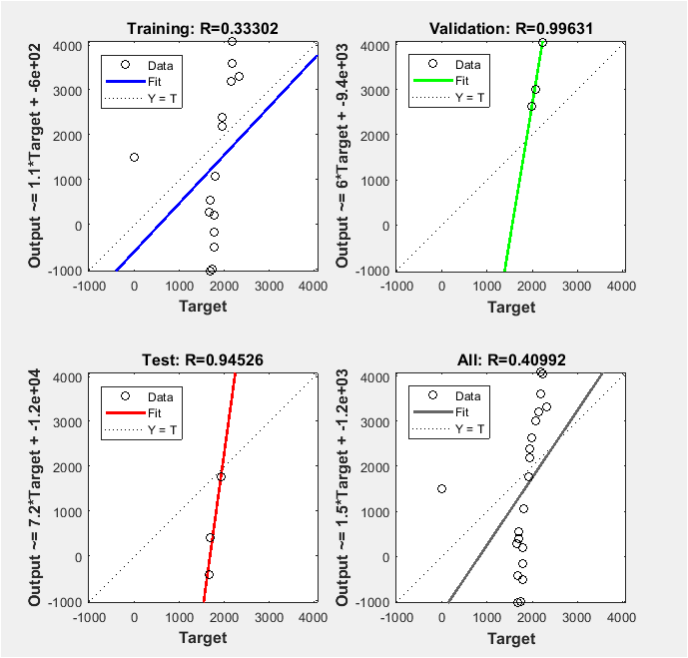
sieć z 8 warstwami ukrytymi,wsp uczenia 0.1, wsp bezwładności 0.5



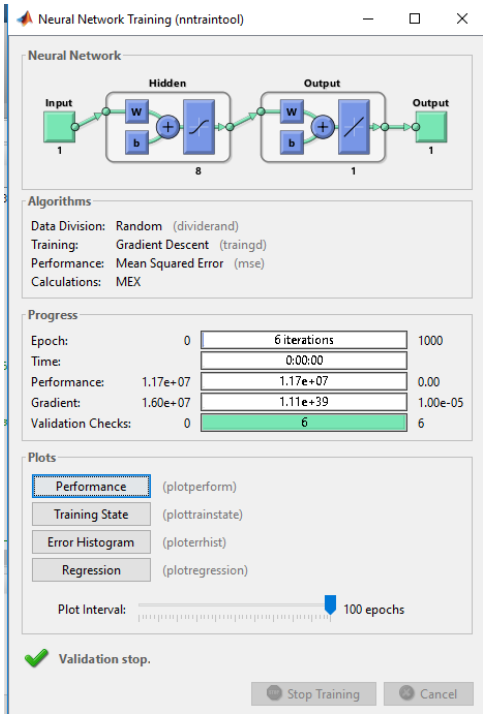


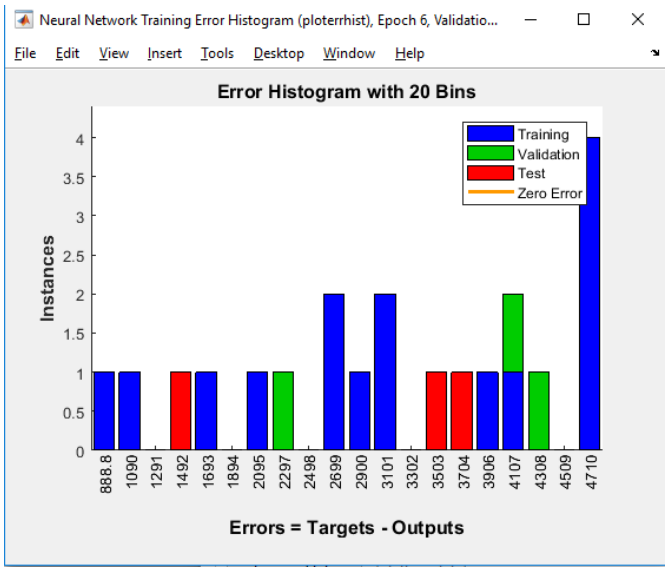


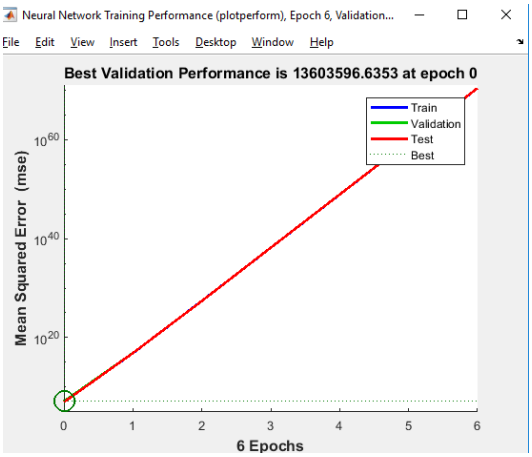


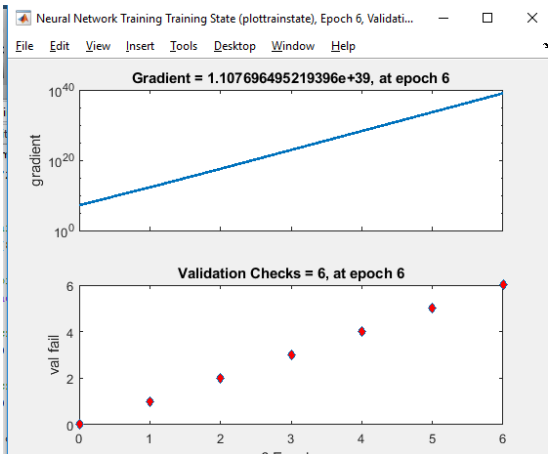


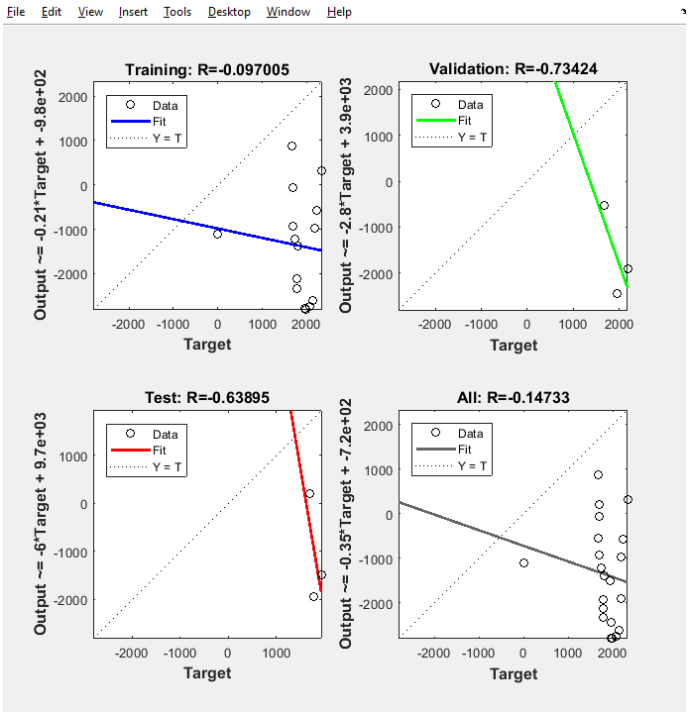
sieć z 8 warstwami ukrytymi,wsp uczenia 0.01, wsp bezwładności 0.0



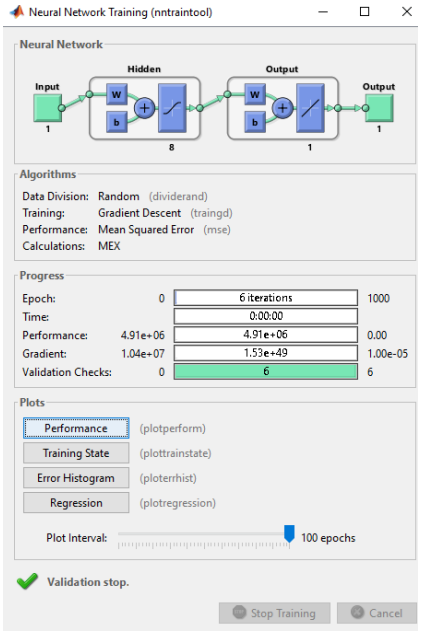


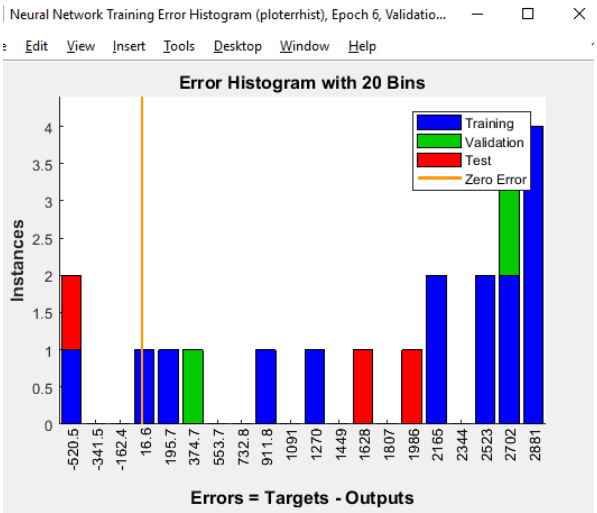


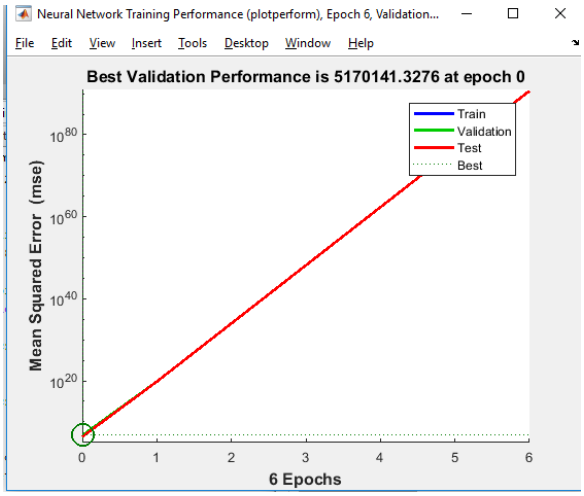


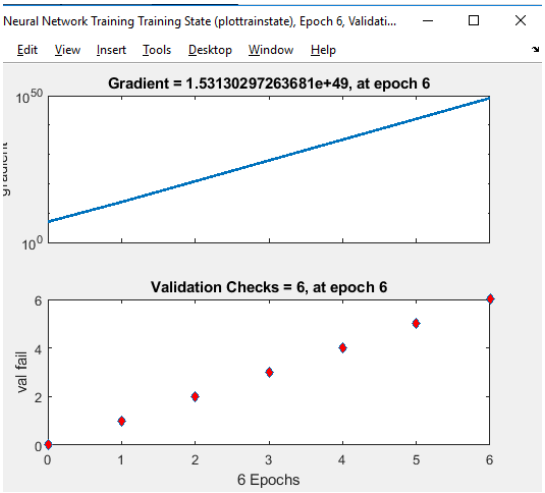


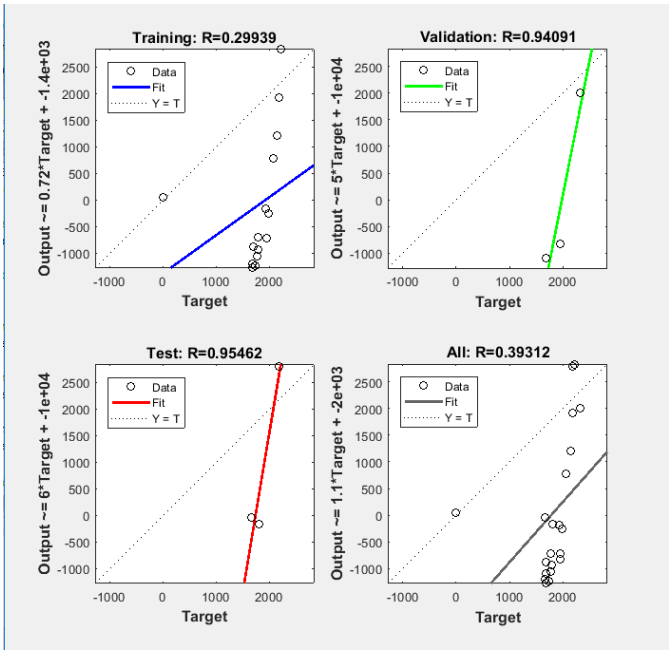
sieć z 8 warstwami ukrytymi,wsp uczenia 0.5, wsp bezwładności 1.0



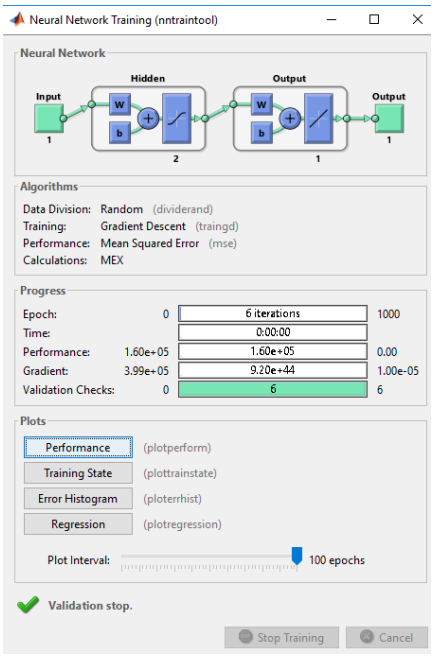


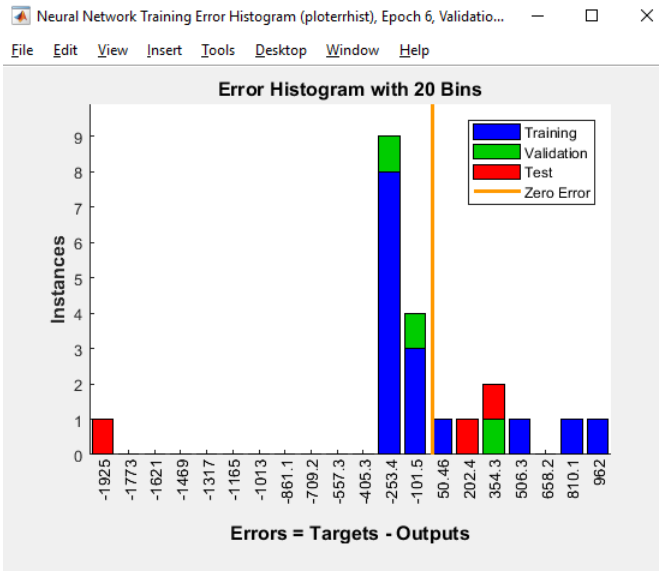


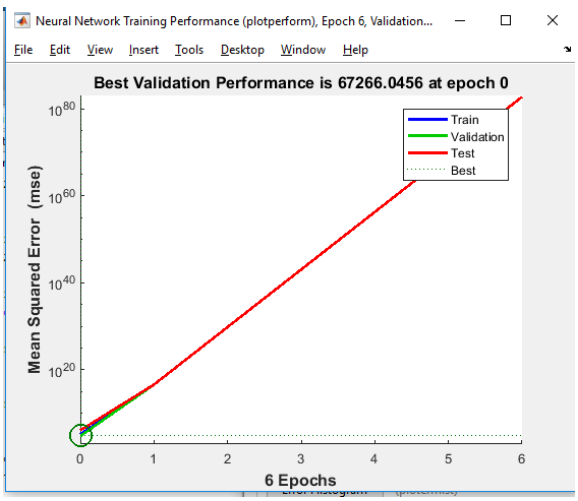


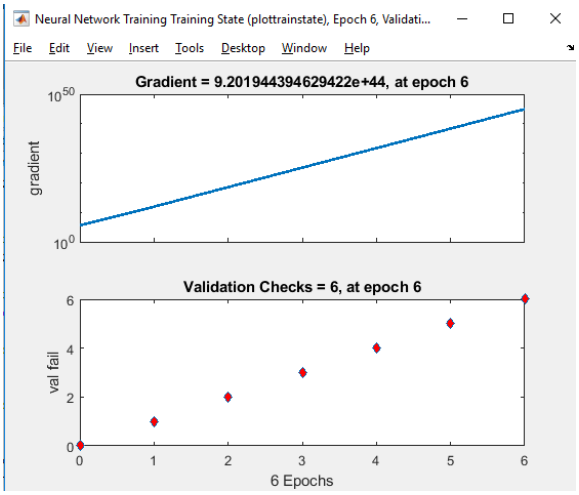


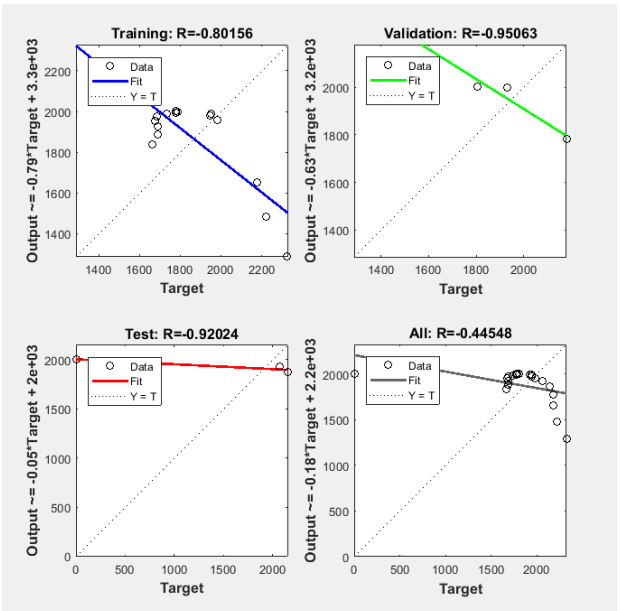
sieć z 2 warstwami ukrytymi,wsp uczenia 0.5, wsp bezwładności 0.5



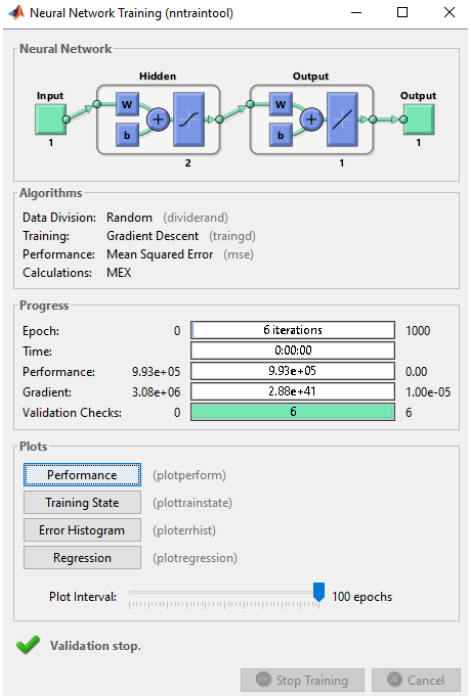


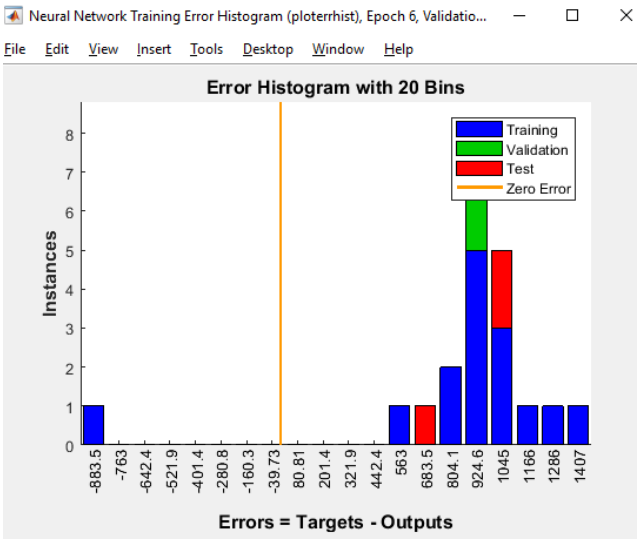


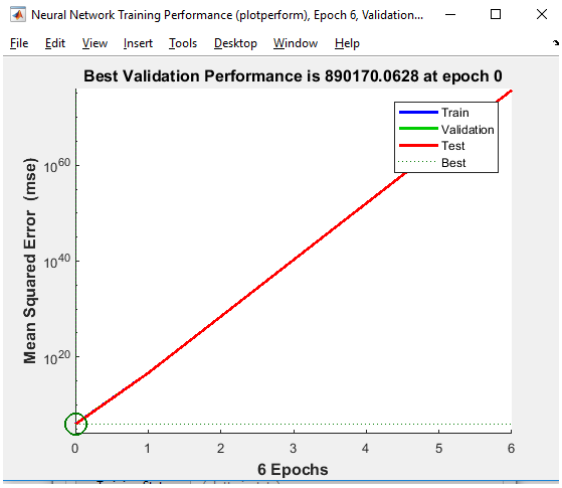


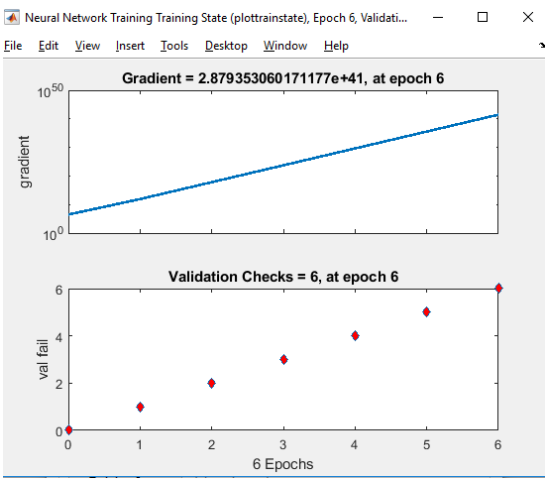


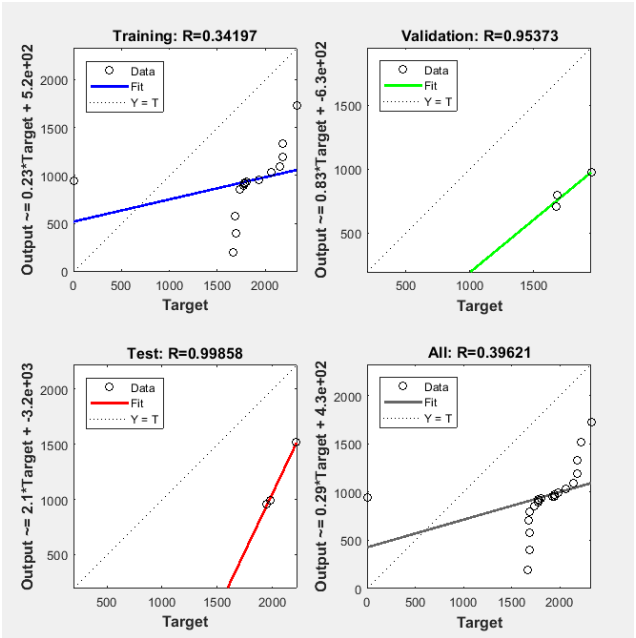
sieć z 2 warstwami ukrytymi,wsp uczenia 0.1, wsp bezwładności 0.5











Listing kodu programu:

```
in = [-2.0 -1.8 -1.6 -1.4 -1.2 -1.0 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1.0
1.2 1.4 1.6 1.8 2.0]
out = [1663.3144032672324 1690.1132374370839 1688.034998342087 1674.099059643066
1686.7210309127113 1732.9225480289258 1776.4412985191298 1786.0418485521361
1779.9521156718527 1803.5731229031892 0 1928.860217750531 1949.5489457731007
1948.9857411333403 1982.511002459129 2065.507872849039 2147.692111352819
2178.4193810575716 2179.078110374434 2220.2957248379535 2326.150912968639 ]
test = zeros(1);

net = feedforwardnet(8); %tworzenie sieci
net.trainFcn = 'traingd'; % wsteczna propagacja
net.trainParam.lr = 0.1; % wsp uczenia 0.5 0.1 0.01
net.trainParam.mc = 0.5; %wsp.bezwladnosci

net = train(net, in, out);
efect = zeros(size(net));

% generowanie wyników dla funkcji rstrigin_3d
%testowanie dzialania sieci
for k = 1:21
    test(k) = rstrigin_3d(in(k));
    efect(k) = sim(net, in(k));
end


function fx = rstrigin_3d(x)
    if x == 0
        fx = 0;
    else
        x1 = x;
        A = 10;
        n = 100;
        dx = (5.12-x)/n;

        fx = A * n;

        for i = 1:n
            x = x1 + (i * dx);
            fx = fx + (x^2) - (A * cos(2 * pi * x));
        end
    end
end
```

Wyniki:

Sieci wielowarstwowe posiadają błędy w stosunku do sieci jednowarstwowych.

Najlepsze wyniki zostały uzyskane w momencie gdy współczynnik uczenia był różny od współczynnika bezwładności.

Sieć wielowarstwowa bez algorytmu propagacji danych działają lepiej niż sieć z wspomnianym algorytmem.

Należy zachować duże różnice między wsp. uczenia a wsp. bezładności przy nie używaniu algorytmu propagacji danych.

Najgorsze wyniki osiągają sieci których wsp.uczenia jest równy wsp.bezwładności.

Sieci wielowarstwowe wykorzystujące algorytm propagacji danych posiadają nieznaczny odsetek błędów. W tym przypadku im większy wsp. uczenia tym wyższe prawdopodobieństwo wystąpienia błędu.

Algorytm propagacji danych przyspiesza nawet o połowę proces uczenia epok.

Gradienty uczenia przy sieciach z algorytmem propagacji błędów posiadają charakterystykę niemalże liniową, dzięki czemu można łatwiej przewidzieć, jakich współczynników do nauki sieci najlepiej użyć, by otrzymane rezultaty były zadowalające.