Jednokierunkowe sieci wielowarstwowe 2

Polecenie tworzące sieć

```
net = newff(we,wy,[h1 h2],{'tansig', 'logsig','purelin'});
```

we – macierz danych wejściowych
wy – wektor zadanych danych wyjściowych
[h1 h2] – wektor określający ilość neuronów na kolejnych warstwach ukrytych
{' ',' ',' '} - wektor komórkowy określający typy funkcji aktywacji na warstwach ukrytych i wyjściowej

Przykłady:

Sieć z 1 warstwą ukrytą na której jest 5 neuronów z funkcją aktywacji typu tangens hiperboliczny i z neuronami liniowymi na warstwie wyjściowej:

```
net = newff(we,wy,[5],{'tansig', 'purelin'});
```

Sieć z dwoma warstwami ukrytymi. Na pierwszej mamy 10 neuronów z funkcją aktywacji typu tangens hiperboliczny, na drugiej 4 neurony z funkcją sigmoidalną i na warstwie wyjściowej mamy neurony liniowe:

```
net = newff(we,wy,[10 4],{'tansig', 'logsig','purelin'});
```

Przykład dla danych 1-wejściowych

```
we = load('dane_1D_sin1_i.txt'); we = we';
wy = load('dane_1D_sin1_o.txt'); wy = wy';

net = newff(we,wy,[4],{'logsig', 'purelin'});
%net = newff(minmax(we),[4 1],{'logsig', 'purelin'});
net.inputweights{1,1}.initFcn = 'rands';
net.biases{1}.initFcn = 'rands';
net = init(net); % inicjujemy wagi losowo
net.trainParam.epochs = 100;

figure(1)
we1 = linspace(min(we),max(we),1000);
wy1 = sim(net, we1);
plot(we1,wy1,'r'); hold on;
plot(we,wy,'.b')
title('Charakterystyka modelu przed uczeniem')

net = train(net, we, wy);

figure(2)
wy2 = sim(net, we1);
plot(we1,wy2,'g'); hold on;
plot(we,wy,'.b')
title('Charakterystyka modelu po uczeniu')
```

Przykład dla danych 2-wejściowych:

```
we = load('dane_2D_5_i.txt'); we = we';
wy = load('dane_2D_5_o.txt'); wy = wy';

net = newff(we,wy,[5],{'tansig', 'logsig'});
net.outputs{2}.processFcns = {};
%net = newff(minmax(we),[4 1],{'logsig', 'purelin'});
net.inputweights{1,1}.initFcn = 'rands';
net.biases{1}.initFcn = 'rands';
net = init(net); % inicjujemy wagi losowo
net.trainParam.epochs = 100;

figure(1)
wy1 = sim(net, we);
x_lin = linspace(min(we(1,:)),max(we(1,:)),50);
y_lin = linspace(min(we(2,:)),max(we(2,:)),50);
[X,Y] = meshgrid(x_lin,y_lin);
Z = griddata(we(1,:), we(2,:), wy1, X, Y, 'cubic');
mesh(X,Y,Z)
hold on
plot3(we(1,:), we(2,:), wy,'.');
title('Powierzchnia modelu przed uczeniem')

net = train(net, we, wy);

figure(2)
wy2 = sim(net, we);
x_lin = linspace(min(we(1,:)),max(we(1,:)),50);
y_lin = linspace(min(we(2,:)),max(we(2,:)),50);
[X,Y] = meshgrid(x_lin,y_lin);
Z = griddata(we(1,:), we(2,:), wy2, X, Y, 'cubic');
mesh(X,Y,Z)
hold on
plot3(we(1,:), we(2,:), wy,'.');
title('Powierzchnia modelu po uczeniu')

figure(3) % wykres tylko dla zadania klasyfikacji binarnej
%plot(we(1,:), we(2,:),'.'); hold on
plotpv(we,wy); hold on
contour(X,Y,Z,[0.5])
title('Linia separujaca klasy')

figure(4)
x_lin = linspace(min(we(1,:)),max(we(1,:)),50);
y_lin = linspace(min(we(2,:)),max(we(2,:)),50);
[X,Y] = meshgrid(x_lin,y_lin);
Z = griddata(we(1,:), we(2,:), wy, X, Y, 'cubic');
mesh(X,Y,Z)
hold on
plot3(we(1,:), we(2,:), wy,'.');
title('Interpolowana powierzchnia danych')
```

Zadania do wykonania:

1. Dla wybranych plików z danymi:
   a. dobrać strukturę sieci neuronowej (ilość wejść i wyjść zależy od danych, dobrać ilość warstw ukrytych – 1 lub 2, dobrać ilość neuronów w warstwach ukrytych – np. metodą addytywną, dobrać typy funkcji aktywacji w warstwach ukrytych i w warstwie wyjściowej – zależnie od problemu);
   b. dla badanych sieci należy obserwować kształt charakterystyki modelu i określić błąd sieci.

2. Opracować samodzielnie skrypt, uczący sieć metodą wstecznej propagacji błędu.

W sprawozdaniu zamieszczamy raport z przeprowadzonych eksperymentów dla każdych badanych danych.