Sieci Neuronowe i Neurokomputery real. A

Projekt – aproksymacja funkcji
2018L

Imię i nazwisko studenta	Ocena
Igor Markiewicz	

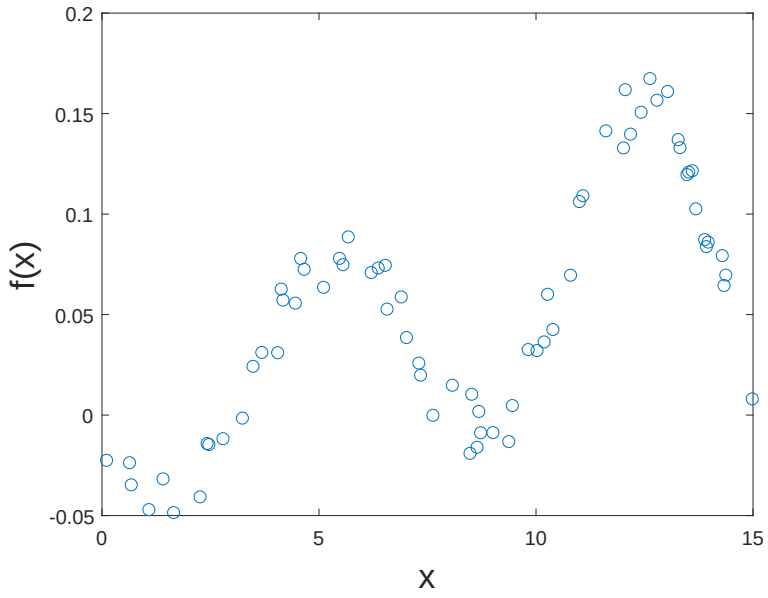
Spis treści

1 Cel projektu

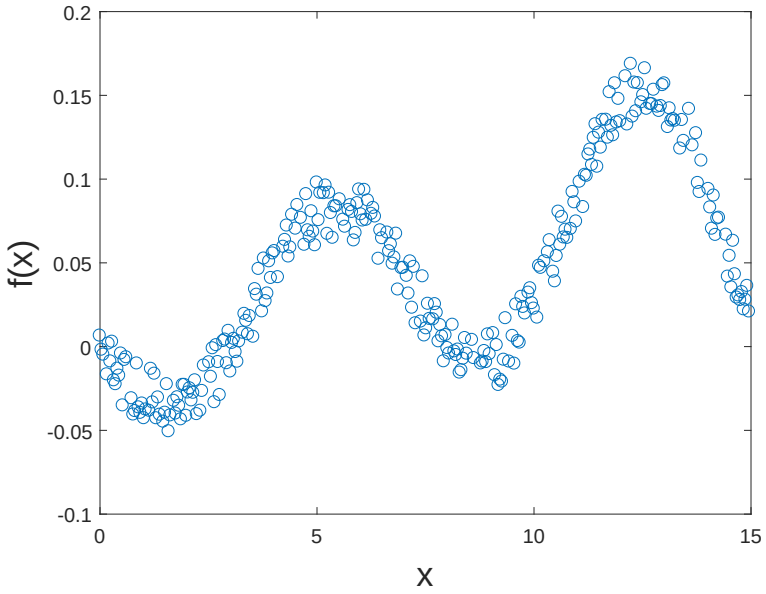
Celem projektu jest uzyskanie możliwie jak najbardziej optymalnej sieci neuronowej (optymalna liczba neuronów w warstwie ukrytej, dobra generalizacja) typu perceptron wielowarstwowy dla przydzielonych danych. Zadaniem sieci jest aproksymacja funkcji jednej zmiennej.

2 Opis i wizualizacja danych

Dostarczone zostały dwa zestawy danych – zbiór uczący(70 rekordów) oraz testowy(300 rekordów). Każdy rekord zawiera parę liczb – zmienną wejściową x oraz odpowiadającą dla tego argumentu wartość funkcji y. Dane znajdują się w plikach *zestaw_apr_2_train.txt* oraz *zestaw_apr_2_test.txt*.



Rys. 1: Dane uczące



Rys. 2: Dane testowe

3 Algorytm uczenia i opis zastosowanego programu komputerowego (środowisko, język programowania, procedury numeryczne i graficzne, interfejs użytkownika)

Do rozwiązania postawionego zadania użyto środowiska Matlab w wersji 2016b oraz pakietu Neural Network Toolbox. Jako algorytm uczenia zastosowano zgodnie z zaleceniami najpierw algorytm wstecznej propagacji błędu (100 epok) a następnie algorytm Levenberga – Marquardta (200 epok). Normalizację (i denormalizację) danych przeprowadzono w następujący sposób :

$$x_{znorm} = \frac{x_i - \overline{x}}{x_{max} - x_{min}} \in [-1, 1]$$

4 Raport z wykonania zadań projektu i wybór optymalnego modelu – analiza testu

Architektura sieci jest następująca :

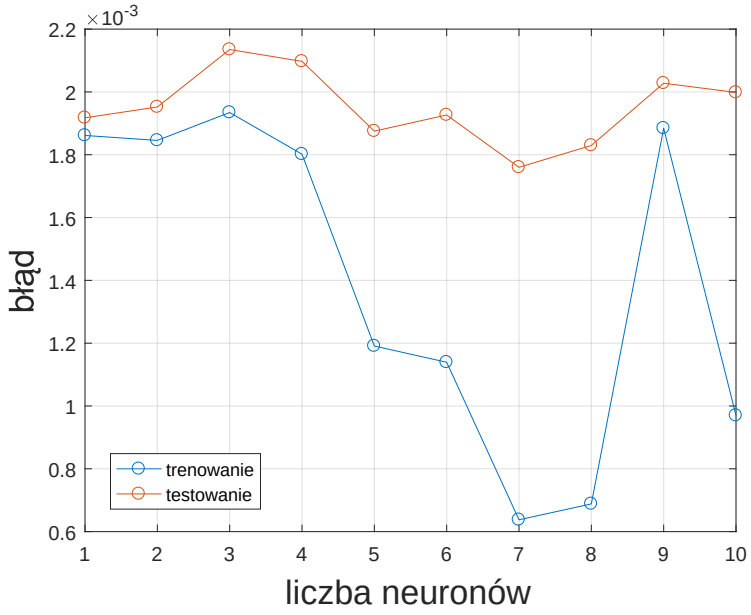
- jedno wejście (zmienna x)
- n neuronów ukrytych z n biasami oraz funkcją aktywacji typu tangens hiperboliczny

- jeden neuron wyjściowy z jednym biasem oraz liniową funkcją aktywacji (zmienna y)

W programie w przypadku macierzy niepełnego rzędu, wyniki zostały oznaczone symbolem *NaN*.

4.1 Dobór liczby neuronów ukrytych metodą porównania błędu średniokwadratowego aproksymacji na zbiorze uczącym i na zbiorze testowym po zakończeniu uczenia

Testy przeprowadzone zostały dla sieci zawierających od 1. do 10. neuronów ukrytych, w każdym przypadku powtarzając uczenie i testowanie 10 razy, a następnie uśredniając błędy.



Rys. 3: Błędy dla uczenia i testowania w zależności o d liczby neuronów ukrytych

Najmniejszą wartość błędu uzyskano dla 7. neuronów ukrytych zarówno w przypadku uczenia $(6,3762 \cdot 10^{-4})$ jak i testowania (ok. $18 \cdot 10^{-4}$). Zaobserwowano przy tym wahania błędu dla uczenia dla większej liczby neuronów.

4.2 Dobór liczby neuronów ukrytych metodą wirtualnej skrajnej oceny krzyżowej (ang. *virtual leave-one-out*)

Wyniki wariancji rozkładów dźwigni dla poszczególnej liczby neuronów oraz zbiorów znajdują się w pliku *var_hkk.xlsx*. Jedynym modelem dla którego dla wszystkich badanych zbiorów macierz Jacobiego była pełnego rzędu jest sieć z jednym neuronem. Dla dwóch neuronów istniej jedna sytuacja w której macierz Jacobiego jest niepełnego rzędu, natomiast w pozostałych przypadkach liczba takich sytuacji zaczyna rosnąć.

Liczba neuronów ukrytych	minimalna	średnia	mediana	maksymalna
1	$6,6495 \cdot 10^{-4}$	0,0019	0,0017	0,0166
2	0,0022	0,0129	0,0116	0,0381

Tabela 1: Parametry rozkładu wariancji wag dla sieci pełnego rzędu (**1**) lub prawie pełnego rzędu (**2** – pominięto jedną wartość dla macierzy niepełnego rzędu)

4.3 Określić optymalną liczbę neuronów ukrytych

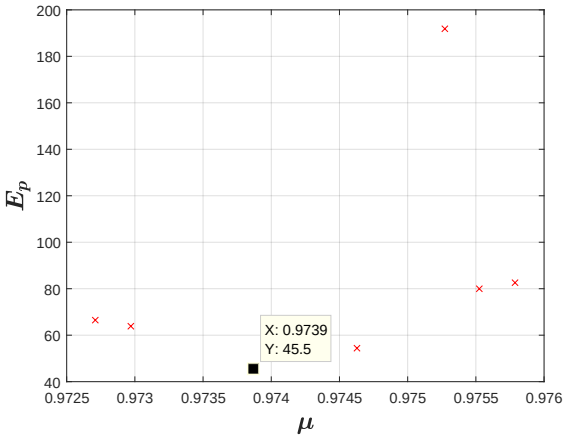
Po kilku testach i obserwacji wyników, zdecydowano że najlepsze skutki są uzyskiwane dla ok 5 – 8 neuronów ukrytych, dlatego zdecydowano ostatecznie że decydującym czynnikiem zostanie błąd średniokwadratowy, w wyniku czego ustalono optymalną liczbę neuronów ukrytych na 7.

4.4 Dla optymalnej liczby neuronów ukrytych wykonać symulację serii 50 sieci neuronowych dla różnych losowych wag początkowych dla tego samego zbioru uczącego. Wybrać sieci, dla których macierz $\boldsymbol{Z}$ jest pełnego rzędu ($rank\boldsymbol{Z} = q$) i wykonać następujące czynności

4.4.1 Obliczyć wielkości E_p oraz μ, przedstawić wyniki obliczeń w tabeli, sporządzić wykres w postaci punktów we współrzędnych (μ, E_p) i wybrać najlepszy model

Do wyboru modelu we współrzędnych (μ, E_p) została zastosowana metryka Euklidesa dla której wyznaczono wartość minimalną :

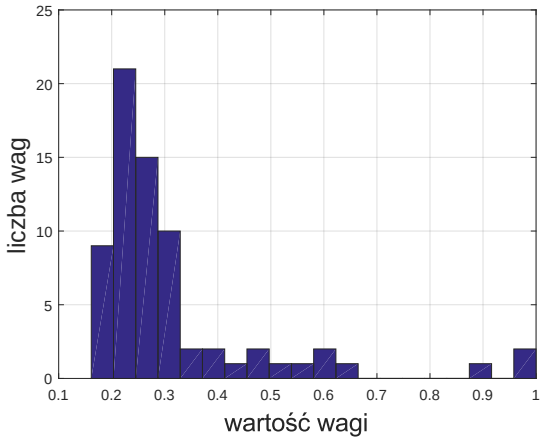
$$i_{min} = \arg\min_i \left(\sqrt{E_p^2(i) + (1 - \mu(i))^2} \right)$$



Rys. 4: Wykers (μ, E_p) z zaznaczeniem najlepszej sieci wg. minimum kryterium Euklidesa

Wyznaczono że najlepszy model odpowiada $(\mu; E_p) = (0,9739; 45,5044)$

4.4.2 Sporządzić histogram dźwigni dla najlepszej sieci

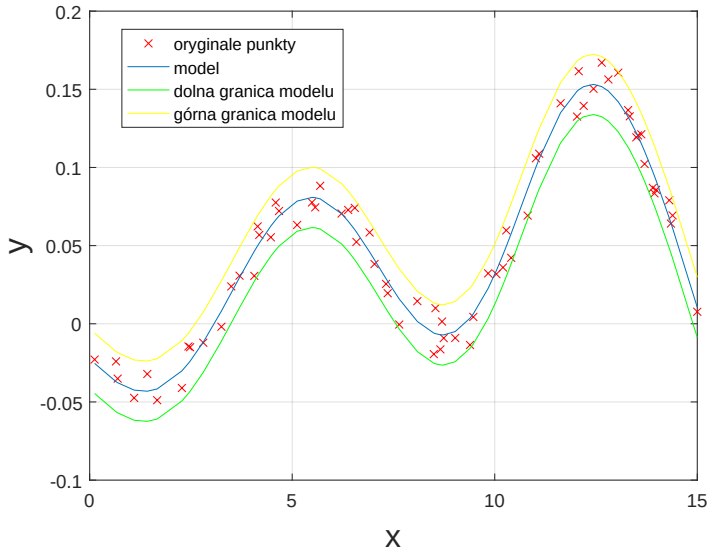


Rys. 5: Histogram dźwigni dla najlepszej sieci

Możemy zauważyć że duża wartość μ i mała E_p zapewniają stosunkowo duże skupienie wartości dźwigni wokół średniej, co wpływa pozytywnie na wartość błędu średniokwadratowego.

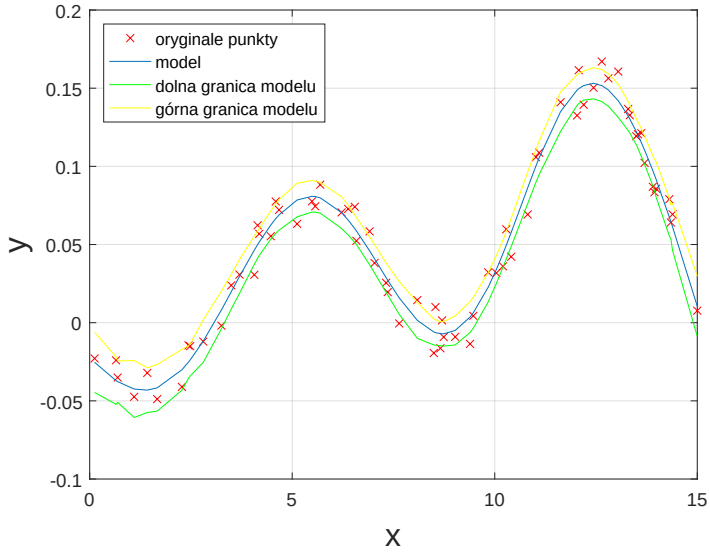
4.4.3 Sporządzić wykres otrzymanej funkcji $y = g(\boldsymbol{x}, \boldsymbol{w}_{SN})$ wraz z przedziałami ufności

- $g(\boldsymbol{x}^k, \boldsymbol{w}_{SN}) \pm t_\alpha^{N-q} s$



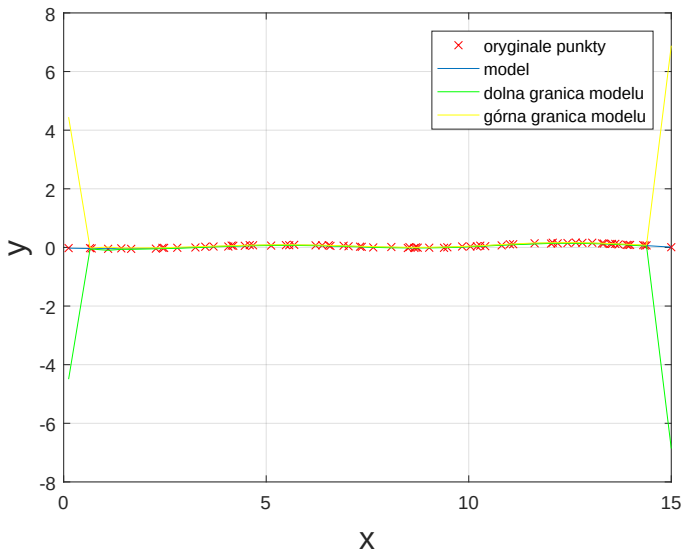
Rys. 6: Przedział ufności pomiaru wyjścia sieci neuronowej

- $g(\boldsymbol{x}^k, \boldsymbol{w}_{SN}) \pm t_\alpha^{N-q} s\sqrt{h_{kk}}$



Rys. 7: Przedział ufności wielkości wyjściowej sieci neuronowej dla k – tego przykładu ze zbioru uczącego na poziomie $1 - \alpha$

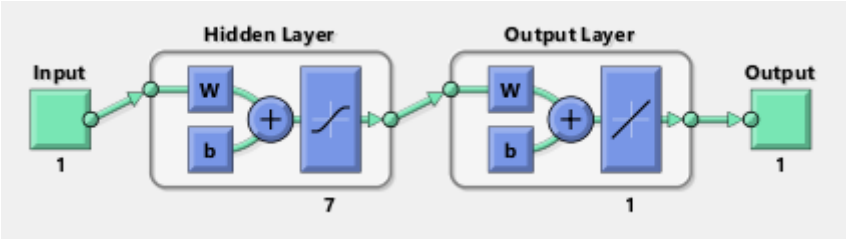
- $g(\boldsymbol{x}^k, \boldsymbol{w}_{SN}) \pm t_\alpha^{N-q} s \sqrt{\frac{h_{kk}}{1-h_{kk}}}$



Rys. 8: Przedział ufności predykcji dla k – tego przykładu ze zbioru uczącego na poziomie $1 - \alpha$

Możemy zauważyć (Rys. 8) że dla przedziałów ufności predykcji dla k – tego przykładu istnieją wartości powodujące znaczne odchylenie na krańcach aproksymowanej funkcji, co oznacza że przewidywanie wartości wyjścia na ich podstawie jest dużo bardziej niepewne w stosunku do pozostałych przykładów.

4.5 Schemat zaprojektowanej optymalnej sieci neuronowej



Rys. 9: Schemat sieci neuronowej

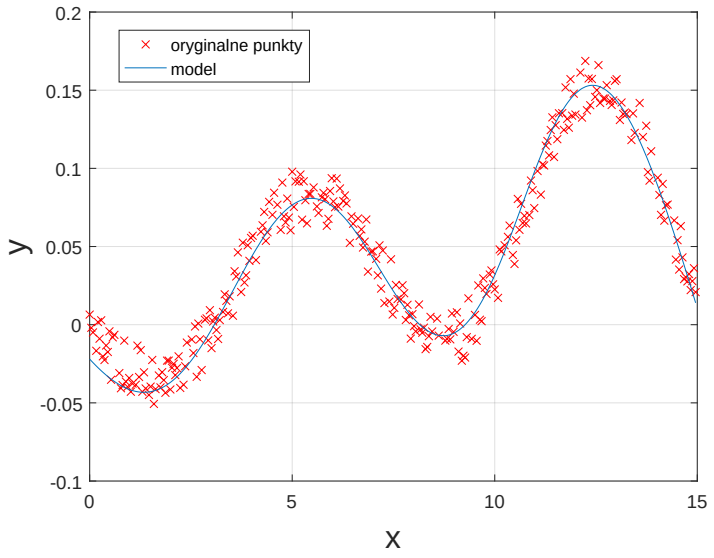
4.6 Sporządzić dokumentację najlepszej sieci neuronowej – tabela wartości wag, plik zawierający wartości wag

Wartości wag znajdują się również w pliku *weights.txt*.

Neuron	waga wejściowa warstwy ukrytej	waga wyjściowa warstwy ukrytej	bias warstwy ukrytej	bias neuronu wyjściowego
1	3,515836	-1,027158	0,226637	-1,687196
2	2,502502	1,232415	1,272526	
3	-3,484641	-2,393289	1,515724	
4	6,863343	-1,802285	-0,6278805	
5	-2,062809	2,562049	2,002011	
6	1,492680	-2,436188	1,333224	
7	2,892343	1,410638	1,651771	

Tabela 2: Tabela z wartościami wag

4.7 Ocenić jakość najlepszej sieci wykorzystując zbiór testowy



Rys. 10: Wynik działania sieci neuronowej dla zbioru testowego

Wyuczona sieć w pomyślny sposób realizuje aproksymację funkcji na zbiorze testowym, nie zaobserwowano niepokojących zjawisk.

5 Sformułować wnioski końcowe i ocenić jakość otrzymanego modelu neuronowego

Zaprojektowana sieć dobrze realizuje swoje zadanie i zapewnia niezbędny poziom generalizacji. Zauważono jednocześnie że istnieją przykłady powodujące poszerzenie przedziałów ufności predykcji wyznaczonych na ich podstawie, wynika to jednak ze specyfiki dostarczonych danych.

6 Źródła

- materiały wykładowe
- dokumentacja języka Matlab
- *Sieci neuronowe do przetwarzania informacji* – Ossowski Stanisław, Oficyna Wydawnicza Politechniki Warszawskiej, 2013
- źródła internetowe

7 Opis załączonych plików: plik zawierający wartości wag sieci neuronowej, programy

- *zestaw_apr_2_train.txt* – zbiór danych uczących
- *zestaw_apr_2_test.txt* – zbiór danych testowych
- *var_hkk.xlsx* – plik z wariancjami dźwigni
- *weights.txt* – plik z wartościami wag optymalnej sieci
- *data_visualization.m* – wizualizacja danych uczących i testowych
- *regularize_data.m* – normalizacja i centrowanie danych
- *deregularize_data.m* – odwrotna transformata w stosunku do normalizacji i centrowania
- *calc_jacobian.m* – dostarczony przez wykładowcę plik z wyznaczaniem macierzy Jacobiego sieci neuronowej
- *train_net.m* – budowa sieci neuronowej
- *optimal_size_of_hidden_layer.m* – określenie optymalnej liczby neuronów w warstwie ukrytej
- *networks_50.m* – budowa i testowanie 50 modeli sieci określonych dla optymalnej liczby neuronów ukrytych
- *getModel.m* – wybór optymalnej sieci wg. minimum kryterium Euklidesa we współrzędnych (μ, E_p)