Krzysztof BARTECKI
POLITECHNIKA OPOLSKA, WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI, INSTYTUT AUTOMATYKI I INFORMATYKI
ul. Gen. K. Sosnkowskiego 31, 45-272 Opole

Niektóre osobliwości aproksymacji neuronowej na przykładzie odwrotnego zadania kinematyki

Dr inż. Krzysztof BARTECKI

Stopień naukowy doktora nauk technicznych uzyskał w 2004 roku w Politechnice Opolskiej. Pracuje w Instytucie Automatyki i Informatyki Politechniki Opolskiej na stanowisku adiunkta. Główny kierunek badań naukowych obejmuje zastosowanie sztucznych sieci neuronowych w zagadnieniach modelowania, identyfikacji oraz sterowania obiektami dynamicznymi.

e-mail: k.bartecki@po.opole.pl

Streszczenie

W artykule wskazano na pewne charakterystyczne aspekty związane z zastosowaniem jednokierunkowych sieci neuronowych jako uniwersalnych układów aproksymujących złożone zależności nieliniowe. Zaprezentowany przykład dotyczy klasycznego problemu z dziedziny robotyki – tzw. *odwrotnego zadania kinematyki*. Zademonstrowano wpływ właściwego doboru struktury sieci, jej algorytmu uczenia oraz wzorców uczących na jakość aproksymacji neuronowej.

Słowa kluczowe: sztuczna sieć neuronowa, aproksymacja funkcji, odwrotne zadanie kinematyki.

Some peculiarities of neural approximation on example of inverse kinematic problem

Abstract

Characteristic features of feedforward artificial neural networks, acting as universal function approximators, are presented. The problem under consideration concerns inverse kinematics of a two-link planar manipulator (Fig. 1). As shown in this paper, a two-layer, feedforward neural network is able to learn the nonlinear mapping between the end effector position domain and the joint angle domain of the manipulator (Fig. 2). However, a necessary condition for achieving the required approximation quality is proper selection of the network structure, especially with respect to the number of nonlinear, sigmoidal units in its hidden layer. Using too few neurons in this layer results in underfitting (Fig. 3), while too many neurons bring the problem of overfitting (Figs 6 and 7). The effect of learning algorithm efficiency as well as proper choice of learning data set on the network performance is also demonstrated (Fig. 8). Apart from the general conclusions concerning neural approximation, the presented results show also the possibility of neural control of robotic manipulator trajectory.

Keywords: artificial neural network, function approximation, inverse kinematics.

1. Wstęp

Charakterystyczną cechą jednokierunkowych, wielowarstwowych sztucznych sieci neuronowych (ang. *multilayer feedforward artificial neural networks*) są ich uniwersalne właściwości aproksymacyjne, umożliwiające realizację praktycznie dowolnych nieliniowych odwzorowań wielowymiarowych [3, 4, 5, 9, 12]. Niezbędnym warunkiem, umożliwiającym efektywne wykorzystanie takiej sieci jako układu aproksymującego, jest właściwy dobór jej struktury. Przy ustalonej wymiarowości zadania aproksymacyjnego, tzn. dla określonej liczby wejść oraz wyjść sieci, dobór ten sprowadza się zwykle do ustalenia liczby jej warstw ukrytych, liczby neuronów w tych warstwach oraz ich funkcji aktywacji. Innym ważnym zagadnieniem jest przygotowanie odpowiedniego zbioru danych uczących oraz testowych, w oparciu o które zrealizowany zostanie algorytm uczenia sieci a także zweryfikowane zostanie jej działanie. Istotne znaczenie ma również wybór odpowiedniego algorytmu uczenia, realizującego zwykle iteracyjną procedurę modyfikacji współczynników wagowych w celu uzyskania założonej wartości funkcji błędu sieci.

W niniejszym artykule zaprezentowano pewne charakterystyczne aspekty zastosowania jednokierunkowej sieci neuronowej jako uniwersalnego narzędzia aproksymacyjnego. Stawiane przed nią zadanie dotyczy jednego z klasycznych problemów robotyki, tzw. *odwrotnego zadania kinematyki*, w odniesieniu do prostego manipulatora o dwóch przegubach.

2. Jednokierunkowa sieć neuronowa jako uniwersalny aproksymator funkcji

Jednym z podstawowych twierdzeń, które gwarantują możliwość zastosowania jednokierunkowych sieci neuronowych w zagadnieniach aproksymacji funkcji ciągłych jest twierdzenie Cybenki, bazujące na wynikach wcześniejszych prac Kołmogorowa [5]. Twierdzenie to sformułowane zostało następująco [3]:

Niech $\sigma(t)$ będzie ciągłą, ograniczoną i monotonicznie rosnącą funkcją typu sigmoidalnego, tzn. funkcją, dla której zachodzi:

$$\lim \sigma(t) = \begin{cases} 1 & dla \quad t \to +\infty \\ 0 & dla \quad t \to -\infty. \end{cases} \tag{1}$$

Wówczas dla danej ciągłej funkcji rzeczywistej wielu zmiennych rzeczywistych $f(x_1,x_2,...,x_P)$, określonej wewnątrz obszaru $[0,1]^P$, oraz dla dowolnego $\varepsilon>0$, istnieje taka liczba całkowita N oraz zbiór takich stałych rzeczywistych a_i, b_i, w_{ij}, że dla:

$$g(x_1, x_2, \ldots, x_P) = \sum_{i=1}^{N} a_i \sigma\left(\sum_{j=1}^{P} w_{ij} x_j + b_i \right), \tag{2}$$

spełniona jest następująca nierówność:

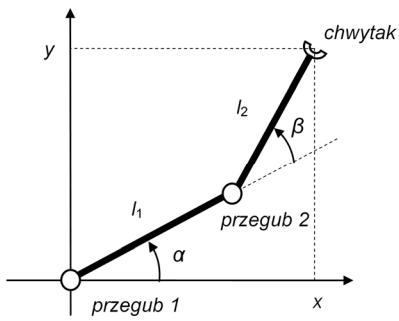
$$\left| f(x_1, x_2, \ldots, x_P) - g(x_1, x_2, \ldots, x_P) \right| < \varepsilon. \tag{3}$$

Analogiczne wnioski zaprezentowano w pracy [4], w której szczegółowo przeanalizowano aproksymacyjne właściwości jednokierunkowych sieci neuronowych. Na podstawie powyższego twierdzenia, jak również w oparciu o wyniki przedstawione m.in. w pracach [3, 4, 5], można stwierdzić, iż jednokierunkowa sieć neuronowa z pojedynczą (tzw. „ukrytą") warstwą neuronów o nieliniowej, sigmoidalnej funkcji aktywacji i z warstwą liniowych neuronów wyjściowych, jest w stanie aproksymować z założoną dokładnością daną ciągłą funkcję nieliniową $f(x_1,x_2,\ldots,x_P)$. Liczba wejść takiej sieci równa jest liczbie argumentów P aproksymowanej funkcji f. Z kolei liczba jej neuronów wyjściowych, w przypadku aproksymacji funkcji wielowartościowej, zależy od wymiarowości zagadnienia. Zatem jedynym otwartym problemem pozostaje kwestia doboru odpowiedniej liczby N neuronów w nieliniowej warstwie sieci, przy czym, jak zademonstrowano w dalszej części artykułu, liczba ta nie może być ani zbyt mała, ani zbyt duża w stosunku do stopnia złożoności danego zadania aproksymacyjnego.

Neuronowa aproksymacja funkcji nieciągłych jest trudniejsza, m.in. ze względu na występowanie efektu Gibbsa. Ciekawe wyniki dotyczące neuronowej aproksymacji funkcji odcinkami ciągłych w kontekście tego zjawiska zaprezentowano m.in. w pracy [5].

3. Odwrotne zadanie kinematyki dla manipulatora planarnego

Jednymi z ważniejszych, z punktu widzenia zastosowań praktycznych, obliczeniowych zagadnień robotyki są tzw. *zadania kinematyki* manipulatorów. W ogólności sprowadzają się one do określenia wzajemnych relacji między tzw. współrzędnymi zewnętrznymi manipulatora, określającymi położenie jego efektora (chwytaka), a współrzędnymi wewnętrznymi, opisującymi zazwyczaj nastawy kątowe jego przegubów [7].



Rys. 1. Schemat poglądowy dwuczłonowego manipulatora planarnego [13]
Fig. 1. Schematic diagram of a two-link planar manipulator [13]

Na rys. 1. przedstawiono schemat poglądowy przykładowego manipulatora planarnego o dwóch przegubach. W tym przypadku tzw. *proste zadanie kinematyki* sprowadza się do wyznaczenia współrzędnych (x,y) chwytaka w sytuacji, gdy znane są wartości kątów (α,β) jego przegubów oraz długości (l_1,l_2) jego ramion. Odpowiednie zależności mają tu następującą postać:

$$\begin{aligned} x &= l_1 \cos\alpha + l_2 \cos(\alpha+\beta) \\ y &= l_1 \sin\alpha + l_2 \sin(\alpha+\beta). \end{aligned} \tag{4}$$

Z kolei *odwrotne zadanie kinematyki* polega na wyznaczeniu takich wartości kątów (α,β) przegubów manipulatora, aby jego chwytak znalazł się w punkcie o zadanych współrzędnych (x,y). Punkt ten w przypadku manipulatora przedstawionego na rys. 1 powinien znajdować się wewnątrz obszaru roboczego, wyznaczonego przez pierścień kołowy o środku w punkcie (0,0) i promieniach: zewnętrznym $r_1=l_1+l_2$ oraz wewnętrznym $r_2=|l_1-l_2|$. Odpowiednie zależności przyjmują wówczas następującą postać:

$$\begin{aligned} \alpha &= \text{atan2}(y,x) - \text{atan2}(k_2,k_1), \\ \beta &= \text{atan2}(s,c), \end{aligned} \tag{5}$$

gdzie:

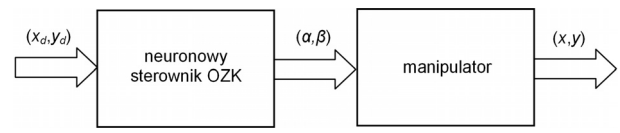
$$\begin{aligned} c &= \frac{x^2+y^2-l_1^2-l_2^2}{2l_1l_2}, \quad s = \sqrt{1-c^2}, \\ k_1 &= l_1 + l_2 c, \qquad\qquad k_2 = l_2 s, \end{aligned} \tag{6}$$

zaś atan2 jest dwuargumentową funkcją *arcus tangens*, w której znaki argumentów są wykorzystywane do wyznaczenia ćwiartki układu współrzędnych, do której należy wynik.

Rozwiązanie odwrotnego zadania kinematyki umożliwia realizację sterowania ruchem manipulatora. Dla danej sekwencji współrzędnych (x_i,y_i), odpowiadającej zadanej trajektorii chwytaka, układ sterujący powinien na podstawie zależności (5) oraz (6) wyznaczyć odpowiednią sekwencję kątów przegubów (α_i,β_i). Wynika stąd, że w ogólnym przypadku sterowanie położeniem chwytaka manipulatora wymaga rozwiązywania równań kinematyki odwrotnej w czasie rzeczywistym. Uzyskanie tych rozwiązań wymaga stosunkowo dużych mocy obliczeniowych, szczególnie w przypadku manipulatorów o wielu współrzędnych. Innym zagadnieniem jest problem tzw. manipulatorów redundantnych, dla których liczba niewiadomych w zadaniu kinematyki odwrotnej przewyższa liczbę jego równań. W literaturze proponuje się szereg rozwiązań tego zadania, w tym m.in. polegających na zastosowaniu tzw. algorytmów jakobianowych, charakteryzujących się stosunkowo dużą złożonością obliczeniową [11, 12]. W wielu pozycjach proponuje się wykorzystanie w tym celu aproksymacyjnych właściwości sztucznych sieci neuronowych [1, 2, 9, 11, 13].

4. Neuronowa aproksymacja odwrotnego zadania kinematyki

W niniejszym rozdziale zademonstrowano aproksymacyjne właściwości jednokierunkowej sieci neuronowej w odniesieniu do odwrotnego zadania kinematyki dla prostego manipulatora planarnego z rys. 1. Zgodnie ze schematem przedstawionym na rys. 2, odpowiednio nauczona sieć może pełnić rolę układu sterującego, generującego sygnały (α,β), oddziałujące na przeguby manipulatora w taki sposób, aby chwytak znalazł się w punkcie o współrzędnych (x,y), równych wartościom zadanym (x_d,y_d), podanym na wejście sieci.



Rys. 2. Zasada działania neuronowego sterownika manipulatora z rys. 1
Fig. 2. Neural network controller scheme for manipulator of Fig. 1

Do badań symulacyjnych przyjęto manipulator o jednostkowych długościach ramion. Założono, że jego chwytak będzie poruszał się wewnątrz ograniczonego obszaru roboczego, wyznaczonego przez kwadrat o wierzchołkach w punktach (0,0) i (1,1). W roli sterownika neuronowego wykorzystano dwuwarstwową sieć jednokierunkową o dwóch wejściach i dwóch liniowych neuronach w warstwie wyjściowej. Proces uczenia sieci przeprowadzono z wykorzystaniem 250 wzorców uczących, reprezentujących punkty rozmieszczone równomiernie na trajektorii w kształcie spirali Archimedesa o równaniu $r = 0.01\varphi$, której środek znajduje się w punkcie o współrzędnych (0.5,0.5) (rys. 3b).

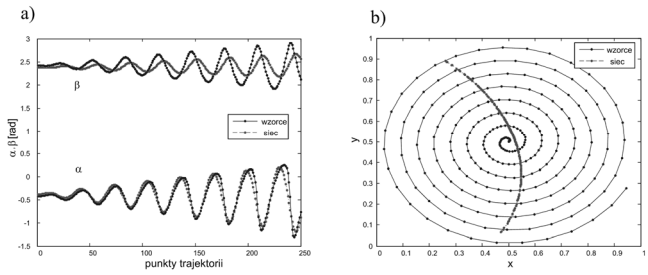
Rolę wzorców wejściowych pełniły współrzędne (x,y) kolejnych punktów trajektorii spiralnej, zaś w roli wzorców wyjściowych wykorzystano odpowiadające im wartości kątów (α,β) przegubów manipulatora. Po nauczeniu sieci jej działanie zostało zweryfikowane, najpierw w oparciu o wzorce uczące. Wygenerowane przez sieć wartości kątów (α,β) przegubów zostały porównane z ich wartościami wzorcowymi. Porównano także spiralną trajektorię wzorcową z trajektorią, jaką zakreśliłby chwytak manipulatora, gdyby do sterowania jego przegubami wykorzystano wartości kątów wygenerowane przez nauczoną sieć neuronową.

W ostatnim etapie badań, w celu określenia właściwości generalizacyjnych sieci, przetestowano jej działanie na trajektorii innej niż wzorcowa, leżącej w granicach obszaru roboczego. W tym celu przyjęto zamkniętą trajektorię w kształcie „klepsydry", złożonej z odcinków, łączących punkty o współrzędnych: A(0,1), B(1,0), C(1,1), D(0,0), A(0,1). Podobnie jak w przypadku trajektorii uczącej, porównano wartości kątów wygenerowane przez nauczoną sieć z ich wartościami wzorcowymi, oraz trajektorię wzorcową z trajektorią, jaką wykreśliłby chwytak manipulatora sterowanego przez sieć neuronową.

W trakcie symulacji zbadano wpływ liczby neuronów w warstwie nieliniowej sieci na jakość aproksymacji. Ponadto przetestowano skuteczność różnych algorytmów uczenia sieci, przy czym w artykule omówiono dwa najbardziej popularne – algorytm największego spadku z tzw. członem *momentum* oraz algorytm optymalizacyjny Levenberga-Marquardta. Badania przeprowadzono z wykorzystaniem m.in. funkcji biblioteki *Neural Network Toolbox*, wchodzącej w skład pakietu *Matlab*.

4.1. Sieć z pojedynczym neuronem nieliniowym

Na rys. 3a porównano wzorcowe wartości kątów α i β dla trajektorii uczącej (spiralnej) z odpowiednimi wartości kątów, uzyskanymi na wyjściach sieci z pojedynczym neuronem o sigmoidalnej funkcji aktywacji (N=1). Z kolei na rys. 3b dokonano porównania wzorcowej trajektorii spiralnej z trajektorią, jaką zakreśliłby chwytak manipulatora sterowanego przez nauczoną sieć, generującą kąty przegubów przedstawione na rys. 3a.



Rys. 3. Wykres porównawczy kątów przegubów (a) oraz trajektorii chwytaka (b) dla danych uczących – sieć z pojedynczym neuronem nieliniowym
Fig. 3. Comparative plot of joint angles (a) and end effector trajectory (b) for learning data set – network with single nonlinear neuron

Przedstawiona na rys. 3b trajektoria chwytaka manipulatora, otrzymana na podstawie kątów wygenerowanych przez nauczoną sieć, w niczym nie przypomina trajektorii wzorcowej. Wynika to z faktu, że sieć z pojedynczym neuronem nieliniowym nie jest w stanie dokonać poprawnej aproksymacji stosunkowo złożonych zależności nieliniowych (5) oraz (6). W tym celu konieczne jest zwiększenie liczby neuronów w warstwie ukrytej sieci.

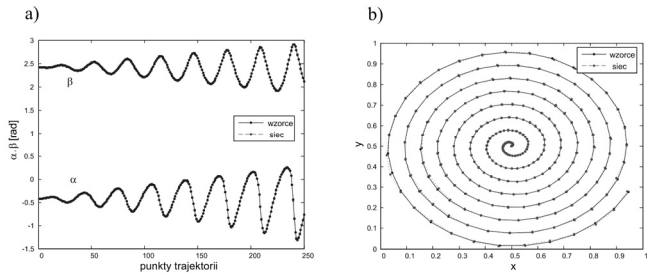
4.2. Osiem neuronów nieliniowych

Zwiększenie liczby neuronów w nieliniowej warstwie sieci znacząco wpłynęło na poprawę jej zdolności aproksymacyjnych. Na rys. 4 zaprezentowano wykresy porównawcze trajektorii uczących, uzyskane dla sieci z ośmioma neuronami sigmoidalnymi (N=8). Z kolei odpowiednie przebiegi dla trajektorii testowej przedstawiono na rys. 5. Jak wynika z otrzymanych wykresów, jakość aproksymacji neuronowej jest bardzo wysoka w przypadku trajektorii uczącej oraz zadowalająca w przypadku trajektorii testowej. Pogorszenie jakości aproksymacji jest widoczne w przypadku jej bocznych krawędzi i „narożników". Wynika to z faktu, że te punkty obszaru roboczego manipulatora nie były reprezentowane w zbiorze danych wejściowych. W celu poprawy jakości działania sterownika neuronowego na cały obszar roboczy, należałoby rozszerzyć zbiór wzorców uczących o dodatkowe punkty.

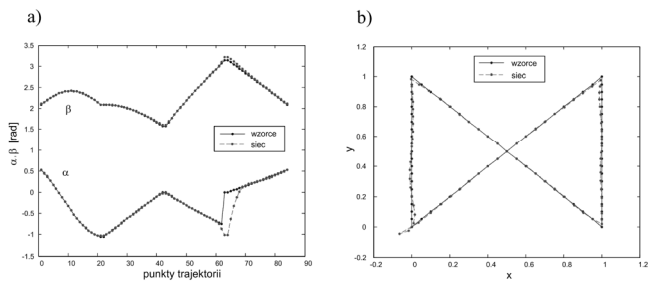
4.3. Zbyt duża liczba neuronów nieliniowych

Na kolejnych rysunkach przedstawiono trajektorie uzyskane dla sieci zawierającej 50 neuronów w warstwie nieliniowej. Ocena jakości działania sieci dla wzorców rozmieszczonych na trajektorii uczącej (rys. 6) nie budzi najmniejszych zastrzeżeń. Jednak przebieg trajektorii testowej (rys. 7) jest nieakceptowany z punktu widzenia jakości działania sieci neuronowej jako układu sterującego omawianym manipulatorem. Przyczyną jest tu nadmierne dopasowanie (ang. *overfitting*), spowodowane zbyt dużą liczbą neuronów nieliniowych w stosunku do stopnia złożoności zadania aproksymacyjnego. Sieć uczy się tu niejako „na pamięć" trajektorii wzorcowej, tracąc przy tym zdolność uogólniania, co odbija się na jakości działania dla danych, które nie były prezentowane w trakcie uczenia.

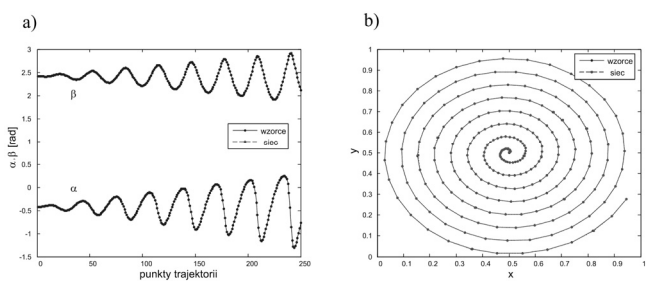
Pewną poprawę właściwości generalizacyjnych sieci można uzyskać, wydzielając ze zbioru dostępnych wzorców uczących dodatkowy podzbiór tzw. danych walidacyjnych [8, 10, 13]. Proces uczenia sieci przeprowadza się w sposób klasyczny, tzn. minimalizując wartość funkcji błędu sieci, określonej na zbiorze danych uczących.



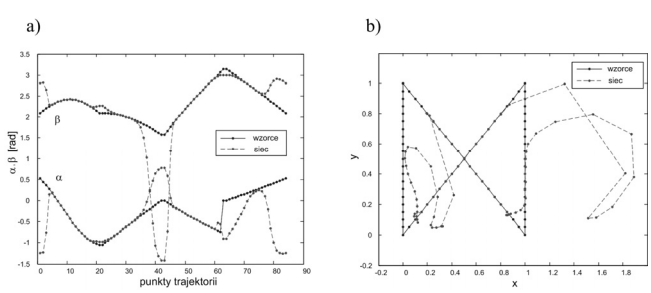
Rys. 4. Wykres porównawczy kątów przegubów (a) oraz trajektorii chwytaka (b) dla danych uczących – sieć z ośmioma neuronami nieliniowymi
Fig. 4. Comparative plot of joint angles (a) and end effector trajectory (b) for learning data set – network with eight nonlinear neurons



Rys. 5. Wykres porównawczy kątów przegubów (a) oraz trajektorii chwytaka (b) dla danych testowych – sieć z ośmioma neuronami nieliniowymi
Fig. 5. Comparative plot of joint angles (a) and end effector trajectory (b) for testing data set – network with eight nonlinear neurons



Rys. 6. Wykres porównawczy kątów przegubów (a) oraz trajektorii chwytaka (b) dla danych uczących – sieć z pięćdziesięcioma neuronami nieliniowymi
Fig. 6. Comparative plot of joint angles (a) and end effector trajectory (b) for learning data set – network with fifty nonlinear neurons



Rys. 7. Wykres porównawczy kątów przegubów (a) oraz trajektorii chwytaka (b) dla danych testowych – sieć z pięćdziesięcioma neuronami nieliniowymi
Fig. 7. Comparative plot of joint angles (a) and end effector trajectory (b) for testing data set – network with fifty nonlinear neurons

Jednak w trakcie uczenia sieci dodatkowo kontroluje się wartość funkcji błędu, określonej dla podzbioru danych walidacyjnych. Uczenie sieci przerywa się w momencie, gdy jej wartość zacznie wzrastać ponad pewien określony poziom. Strategia ta zapobiega zjawisku nadmiernego dopasowania sieci do wzorców uczących, zapewniając jej lepsze właściwości generalizacyjne. Jednak niezależnie od przyjętej strategii uczenia, liczba neuronów w warstwie nieliniowej powinna być dostosowana do stopnia złożoności zagadnienia aproksymacyjnego.

4.4. Wpływ algorytmu uczenia sieci na jakość aproksymacji

Wyniki przedstawione w poprzednich punktach dotyczą sieci neuronowej uczonej klasyczną, gradientową metodą największego spadku z adaptacyjnymi zmianami współczynnika prędkości uczenia oraz z tzw. członem *momentum*. Modyfikacja wartości współczynników wagowych sieci przeprowadzana jest tu w kolejnych cyklach uczących w oparciu o następującą zależność [8, 10, 13]:

$$\boldsymbol{w}(i+1) = \boldsymbol{w}(i) - \eta(i)\nabla E(\boldsymbol{w}(i)) + \alpha(i)(\boldsymbol{w}(i) - \boldsymbol{w}(i-1)), \quad (7)$$

gdzie:

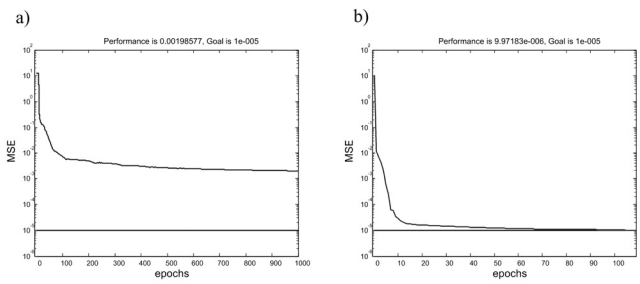
$\boldsymbol{w}(i)$ – wektor współczynników wagowych sieci w *i*-tym cyklu,
$\nabla E(\boldsymbol{w}(i))$ – gradient funkcji błędu sieci względem jej wag,
$\eta(i)$ – współczynnik prędkości uczenia,
$\alpha(i)$ – stała *momentum*.

Znacznie korzystniejszy przebieg procedury uczenia sieci uzyskano stosując quasi-newtonowskie metody optymalizacyjne, a w szczególności algorytm Levenberga-Marquardta. Modyfikacja wartości współczynników wagowych sieci realizowana jest tu według następującego wzoru [8, 10]:

$$\boldsymbol{w}(i+1) = \boldsymbol{w}(i) - \eta(i)\left[\widetilde{\boldsymbol{H}}(\boldsymbol{w}(i))\right]^{-1}\nabla E(\boldsymbol{w}(i)), \quad (8)$$

gdzie $\widetilde{H}(\boldsymbol{w}(i))$ jest aproksymowaną macierzą hesjanu, czyli macierzą drugich pochodnych funkcji błędu sieci $E(\boldsymbol{w}(i))$ względem jej wszystkich współczynników wagowych.

Na rys. 8 przedstawiono uzyskane charakterystyki, ilustrujące przebieg zmian wartości funkcji błędu sieci w kolejnych cyklach uczenia w oparciu o metodę największego spadku (a) oraz Levenberga-Marquardta (b).



Rys. 8. Przebieg zmian wartości funkcji błędu sieci uczonej algorytmem największego spadku (a) oraz Levenberga-Marquardta (b)
Fig. 8. Process of network error minimisation for gradient descent (a) and Levenberg-Marquardt (b) learning algorithm

Podsumowując otrzymane wyniki, można stwierdzić, iż algorytm optymalizacyjny Levenberga-Marquardta umożliwia bardziej efektywne uczenie sieci neuronowej realizującej zadanie aproksymacyjne, głównie ze względu na liczbę cykli treningowych, wymaganych w celu osiągnięcia założonej wartości funkcji błędu. Wynika to z faktu, że algorytm ten charakteryzuje się zbieżnością kwadratową, podczas gdy metoda najszybszego spadku jest metodą o zbieżności liniowej [8]. Za wadę algorytmu Levenberga-Marquardta można uznać spory koszt obliczeniowy, głównie ze względu na konieczność wyznaczania odwrotności aproksymowanej macierzy hesjanu.

5. Podsumowanie

W artykule wskazano na pewne charakterystyczne cechy jednokierunkowej sieci neuronowej jako uniwersalnego narzędzia aproksymacyjnego. Zademonstrowano je na przykładzie aproksymacji odwrotnego zadania kinematyki, umożliwiającej zastosowanie sieci neuronowej jako układu sterującego prostym manipulatorem planarnym o dwóch ramionach. Na podstawie przedstawionych wyników można stwierdzić, że niezbędnym warunkiem poprawnej realizacji zadania jest dobór odpowiedniej struktury sieci, sprowadzający się przede wszystkim do ustalenia właściwej liczby neuronów w jej warstwie nieliniowej. Istotne znaczenie ma również odpowiednia selekcja wzorców uczących oraz wybór efektywnego algorytmu uczenia sieci. W przypadku zbyt małej liczby neuronów nieliniowych poprawna aproksymacja funkcji nie jest możliwa. Z drugiej strony, zbyt duża ich liczba powoduje nadmierne dopasowanie sieci do wzorców uczących, przy jednoczesnym pogorszeniu jej właściwości generalizacyjnych. Istotne znaczenie dla zdolności uogólniania sieci ma również właściwy dobór wzorców uczących – jak zademonstrowano w artykule, powinny one reprezentować cały zakres obszaru roboczego manipulatora.

Wykorzystanie w roli algorytmu uczącego gradientowej metody największego spadku jest nieefektywne ze względu na jej liniową zbieżność. Znacznie lepsze efekty uzyskuje się w przypadku zastosowania algorytmu Levenberga-Marquardta, łączącego w sobie cechy metody największego spadku i algorytmu Newtona, charakteryzującego się zbieżnością kwadratową.

Pomimo charakteru uniwersalnego aproksymatora, jednokierunkowa sieć neuronowa nie może być bezkrytycznie traktowana jako panaceum na wszelkie problemy związane z aproksymacją złożonych zależności nieliniowych. Dopiero uwzględnienie wszystkich wymienionych w artykule elementów: właściwego doboru wzorców uczących, odpowiedniej struktury sieci a także efektywnego algorytmu jej uczenia, sprawia, iż aproksymacja wielowymiarowych odwzorowań nieliniowych z wykorzystaniem sztucznej sieci neuronowej może dać pożądane rezultaty.

6. Literatura

[1] Arteaga-Bravo F. J.: Multilayer back-propagation network for learning the forward and inverse kinematics equations. Proc. of the IEEE Int. Neural Network Conf. (2): 319-321, 1990.
[2] Bingul Z., Ertunc H. M., Oysu C.: Applying Neural Network to Inverse Kinematic Problem for 6R Robot Manipulator with Offset Wrist. Adaptive and Natural Computing Algorithms, Springer, Vienna, 2005.
[3] Cybenko G.: Approximation by Superpositions of a Sigmoidal Function. Mathematical Control Signals Systems (2): 303-314 (1989).
[4] Hornik K., Stinchcombe M., White H.: Mulitilayer Feedforward Networks are Universal Approximators. Neural Networks (2): 359-366 (1989).
[5] Kolmogorov A. N.: On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Doklady Akademii Nauk SSR (114): 953-956 (1957).
[6] Llanas B., Lantarón S., Sáinz F. J.: Constructive Approximation of Discontinuous Functions by Neural Networks. Neural Processing Letters 27(3): 209-226 (2008).
[7] Morecki A., Knapczyk J. (red.): Podstawy robotyki. WNT, Warszawa, 1993.
[8] Osowski S.: Sieci neuronowe w ujęciu algorytmicznym. WNT, Warszawa, 1996.
[9] Oyama E., Chong N. Y., Agah A., Maeda T., Tachi S.: Inverse Kinematics Learning by Modular Architecture Neural Networks with Performance Prediction Networks. Proceedings of the IEEE International Conference on Robotics & Automation, Seoul, 2001.
[10] Rojas R.: Neural networks: a systematic introduction. Springer-Verlag, Berlin, 1996.
[11] Sciavicco L., Siciliano B.: A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulators. IEEE Journal for Robotics and Automation 4(4): 403-410 (1988).
[12] Tchoń K., Karpińska J., Janiak M.: Approximation of Jacobian Inverse Kinematics Algorithms. International Journal of Applied Mathematics and Computer Science 19(4): 519-531 (2009).
[13] Żurada J., Barski M., Jędruch W.: Sztuczne Sieci Neuronowe. Wydawnictwo Naukowe PWN, Warszawa, 1996.

otrzymano / received: 02.04.2010
przyjęto do druku / accepted: 04.05.2010 *artykuł recenzowany*