Algorytmy i metody uczenia sieci neuronowych w oparciu o zmodyfikowane twierdzenie Kołmogorowa

Jarosław Michalkiewicz promotor: prof. dr hab. inż. Michał Białko

kwiecień 2014

Spis treści

1 Wstęp **3**

1.1 Wprowadzenie 3

1.2 Teza i cele pracy 7

2 Aproksymacja za pomocą sieci neuronowych **8**

2.1 Wprowadzenie 8

2.2 Twierdzenie Cybenki [5] 8

2.3 Twierdzenie Stone'a [4] 9

2.4 Szacowanie liczby komórek sieci neuronowej 11

3 Kołmogorowska sieć neuronowa **13**

3.1 Wstęp 13

3.2 Twierdzenie Kołmogorowa i odpowiadająca mu sieć neuronowa 13

3.3 Uczenie kołmogorowskiej sieci neuronowej według Sprechera [7],[8] 15

3.4 Podsumowanie 17

4 Zmodyfikowana kołmogorowska sieć neuronowa **18**

4.1 Wstęp 18

4.2 **Zmodyfikowane podstawowe twierdzenie Kołmogorowa** 18

4.2.1 Sieć liniowa 18

4.2.2 Sieć potęgowa 24

4.2.3 Przykład sieci liniowej dla funkcji liniowej 26

4.2.4 Przykład sieci liniowej dla funkcji kwadratowej 29

4.2.5 Przykład sieci potęgowej dla funkcji kwadratowej 33

4.3 **Nowe koncepcje zmodyfikowanych kołmogorowskich sieci neuronowych** 34

4.3.1 Sieć indeksowana 36

4.4 Błąd aproksymacji funkcji ciągłej 44

4.5 Zestawienie właściwości kołmogorowskich sieci neuronowych 46

4.6 Podsumowanie 47

5 Algorytmy uczenia zmodyfikowanej sieci **49**

5.1 Wstęp 49

5.2 Algorytmy uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej . . 49
5.2.1 Wstęp 49
5.2.2 Opis algorytmów 49
5.3 Warunki zbieżności i szybkość zbieżności algorytmów uczenia sieci kołmogorowskiej 52
5.4 Podsumowanie 55

6 Eksperymenty ze zmodyfikowaną siecią 56
6.1 Wstęp 56
6.2 Identyfikacja charakterystyk obiektów dynamicznych za pomocą sigmoidalnej sieci neuronowej 57
6.2.1 Modele identyfikacji obiektów dynamicznych 57
6.2.2 Zarys identyfikacji systemów dynamicznych Hammersteina i Wienera 57
6.3 Identyfikacja charakterystyk nieliniowych obiektów dynamicznych przy zastosowaniu zmodyfikowanej indeksowanej sieci kołmogorowskiej 59
6.4 Identyfikacja systemu statycznego 61
6.5 Identyfikacja systemów dynamicznych Hammersteina 63
6.6 Identyfikacja systemów dynamicznych Wienera 66
6.7 Złożoność obliczeniowa algorytmu uczenia sieci kołmogorowskiej i algorytmu Sprechera 69
6.8 Porównanie złożoności algorytmu autora i algorytmu A. Janczaka 70
6.9 Podsumowanie 71

7 Zakończenie 72
7.1 Oryginalne wyniki naukowe rozprawy 72
7.2 Uwagi końcowe 73

8 Bibliografia 74

Rozdział 1

Wstęp

1.1 Wprowadzenie

Tematem pracy są metody uczenia sztucznych sieci neuronowych w oparciu o zmodyfikowane twierdzenie Kołmogorowa. Nazwa tych sieci o specyficznej strukturze pochodzi od nazwiska wybitnego matematyka rosyjskiego A. N. Kołmogorowa, który w latach pięćdziesiątych ubiegłego wieku podał twierdzenie o teorii ciągłych funkcji matematycznych. Twierdzenie Kołmogorowa zakłada, że funkcję ciągłą wielu zmiennych można przedstawić w postaci sumy funkcji jednej zmiennej [1].

Twierdzenie to okazało się przydatne w teorii sieci neuronowych ponieważ opierając się na nim, można udowodnić, że sieć neuronowa może aproksymować funkcję ciągłą. Każda funkcja ciągła zdefiniowana na n-wymiarowej kostce Euklidesa może być aproksymowana przez sumę funkcji ciągłych jednej zmiennej, natomiast funkcji jednej zmiennej odpowiada funkcja aktywacji w kołmogorowskiej sztucznej sieci neuronowej.

Prekursorem twierdzenia Kołmogorowa o funkcjach ciągłych jest amerykański matematyk Hilbert [23], który podał tak zwany trzynasty problem dotyczący funkcji rzeczywistych - "udowodnij, że funkcja ciągła trzech zmiennych nie może być reprezentowana przez funkcję ciągłą dwóch zmiennych". Kołmogorow podjął ten temat i udowodnił, że w matematyce nie ma funkcji ciągłych wielu zmiennych na zbiorze zwartym, których nie da się przedstawić w postaci sumy funkcji jednej zmiennej. Z powyższego wynika, że sztuczna sieć neuronowa typu kołmogorowskiego ma specyficzną strukturę wyznaczoną przez twierdzenie Kołmogorowa. Amerykański uczony Hecht-Nielsen [7],[8] jako pierwszy dowiódł, że sieć tego typu można zastosować w technice obliczeniowej. Kurkova [27] potwierdziła to założenie i przedstawiła sieć kołmogorowską w postaci klasycznej sieci sigmoidalnej.

Twierdzenie Kołmogorowa mówi o przedstawieniu dowolnej funkcji ciągłej n zmiennych h wzorem [1]

$$h(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} f_{p,q}(x_p)) \tag{1.1}$$

gdzie g i f są odpowiednio nazywane funkcją zewnętrzną i wewnętrzną.

Twierdzeniu Kołmogorowa odpowiada sieć neuronowa przedstawiona w dalszej części pracy.

Twierdzenie Kołmogorowa w oryginalnej postaci nie nadaje się do obliczeń numerycznych, gdyż orzeka ono tylko o istnieniu funkcji $f_{p,q}$ ale nie mówi jak te funkcje znaleźć.

Aby określić funkcje $f_{p,q}$ należy przyjąć odpowiednią strukturę sieci (liniową, całkowitoliczbową, potęgową). Aby określić funkcje g_q należy sieć wytrenować.

Autor zaproponował w pracy następującą metodę uczenia zmodyfikowanej (przez przyjęcie odpowiedniej struktury) sieci neuronowej kołmogorowskiej; polega ona na

1. przyjęciu jako wewnętrznych funkcji $f_{p,q}$ liniowych, potęgowych i całkowitoliczbowych tak, aby zapewnić zbieżność algorytmu;

2. określeniu zewnętrznych funkcji g_q na podstawie danej aproksymowanej funkcji h.

Zmodyfikowanemu twierdzeniu Kołmogorowa odpowiada sieć neuronowa rozważana w dalszej części pracy.

D. A. Sprecher pierwszy dokonał udanej próby modyfikacji twierdzenia Kołmogorowa, polegającej na określeniu funkcji wewnętrznej i podał algorytm uczenia sieci kołmogorowskich [7],[8]. Algorytm ten jest jednak skomplikowany i trudny w realizacji, a sieć składa się z dwóch warstw, co utrudnia obliczenia. Inny algorytm przedstawiony przez Igelnika [6] używa do modelowania i uczenia sieci neuronowej funkcji sklejanych i jest jeszcze bardziej skomplikowany od algorytmu Sprechera. Kurkowa udowodniła aproksymujące własności sieci kołmogorowskich, ale nie podała algorytmu ich uczenia [27].

Autor niniejszej pracy zmodyfikował twierdzenie Kołmogorowa [34],[36], przez wprowadzenie funkcji liniowej, potęgowej lub całkowitoliczbowej jako funkcji wewnętrznej $f_{p,q}$. Umożliwiło to budowę metod numerycznych, które są proste w obliczeniach, dobrze zbieżne i o dowolnie małym błędzie aproksymacji. Metody te są oparte o:

1. zmodyfikowane twierdzenie Kołmogorowa (liniowa, potęgowa funkcja wewnętrzna) oraz o:

2. zmodyfikowane indeksowane twierdzenie Kołmogorowa (całkowitoliczbowa funkcja wewnętrzna).

W metodach tych błąd aproksymacji uwzględnia zarówno błąd uczenia jak i błąd uogólnienia [36]. Z rozważań w [36] wynika że:

1. błąd całkowity można zmniejszyć przez zwiększenie liczby próbek uczących;

2. w sieciach kołmogorowskich nie ma optymalnej liczby próbek uczących, co różni je od klasycznych sieci sigmoidalnych (w których możliwe jest przeuczenie);

3. im więcej przyjmuje się próbek uczących tym mniejszy staje się błąd aproksymacji;

4. błąd aproksymacji zależy od zmienności funkcji aproksymowanych; im mniejsza jest zmienność tym mniejszy jest błąd.

Algorytmy uczenia sieci neuronowej w oparciu o zmodyfikowane indeksowane twierdzenie Kołmogorowa przedstawiono w formie schematu blokowego i kodu źródłowego w języku Delphi. Podane algorytmy mogą mieć zastosowanie przy realizacji i uczeniu sieci typu Kołmogorowa, która jest siecią nieliniową, niesigmoidalną, dwuwarstwową o dowolnie małym błędzie aproksymacji. Modeluje ona obiekty rzeczywiste i może być zastosowana przy identyfikacji nieliniowych obiektów dynamicznych. Sieć ta przewyższa jakościowo sigmoidalne sieci neuronowe ponieważ jest prostsza w realizacji (prosty algorytm) i w procesie uczenia.

Klasyczna sieć neuronowa wymaga zastosowania skomplikowanych algorytmów uczenia, takich jak metody gradientowe i metody propagacji wstecznej. Proponowane przez autora algorytmy uczenia sieci są prostsze numeryczne i prowadzą do znacznego skrócenia obliczeń.

Na podstawie podręcznika S. Osowskiego [3] sporządzono tablicę 1.1 różnych rodzajów algorytmów uczących, które są stosowane dla różnych sieci neuronowych. Z podanego

Rodzaje sieci	Algorytmy uczenia
Wielowarstwowe (sigmoidalne)	Propagacja wsteczna, metody gradientowe, metody największego spadku, Fahlmana, metody zmiennej metryki
Kołmogorowskie	Kołmogorowski
Radialne	Propagacja wsteczna
Rekurencyjne	Rekurencyjna metoda propagacji wstecznej, Fahlmana, Williama-Zipsera
Samoorganizujące	Kohonena, k-uśrednień, łagodnego współzawodnictwa, gazu neuronowego, reguła Hebba
Rozmyte	Propagacja wsteczna, samoorganizacja, metoda tablicowa

Tablica 1.1: Sieci i algorytmy

zestawienia wynika, że w klasycznych sieciach sigmoidalnych najczęściej stosowane są algorytmy propagacji wstecznej, metody gradientowe i algorytm największego spadku.

Według S. Osowskiego [3] wyróżnia się następujące rodzaje sztucznych sieci neuronowych:

1. ze względu na kierunek przepływu informacji: jednokierunkowe, rekurencyjne;
2. ze względu na liczbę warstw: jednowarstwowe, wielowarstwowe;
3. ze względu na przyjęty model matematyczny: sigmoidalne, radialne, kołmogorowskie;
4. ze względu na sposób organizacji: sieci uczone i samoorganizujące;
5. ze względu na model logiki: o logice klasycznej i rozmytej.

Sztuczne sieci neuronowe mają szerokie zastosowanie w nauce i technice. Informatyka i sztuczna inteligencja to jeden z głównych obszarów ich eksploracji. Sztuczne sieci neuronowe mogą mieć następując zastosowania [25]:

1. jako klasyfikatory neuronowe;
2. neuronowe pamięci asocjacyjne;
3. opisy właściwości obiektów i pojęć ze świata rzeczywistego;
4. neuronowe bazy wiedzy.

W literaturze technicznej i biomedycznej wymienia się jeszcze nastepujące obszary zastosowań sztucznych sieci neuronowych: są to biocybernetyka i modelowanie neuronowe, aproksymacja obiektów rzeczywistych, identyfikacja systemów, rozpoznowanie i przetwarzanie obrazów.

Sieci kołmogorowskie różnią się od klasycznych sieci sigmoidalnych następującymi cechami. Sieć kołmogorowska ma strukturę wyznaczoną przez twierdzenie Kołmogorowa. Zasadniczą funkcję pełnią w niej neurony z realizowaną wewnętrzną i zewnętrzną funkcją aktywacji. Budowę takiej sieci neuronowej zaczyna się od całości struktury to jest od zadanej liczby warstw i liczby neuronów w warstwach przechodząc do pojedynczych neuronów, natomiast przy budowie sieci sigmoidalnej proces przebiega odwrotnie od projektowania elementarnych komórek z funkcją sigmoidalną do określenia struktury całości sieci neuronowej. W sieci neuronowej kołmogorowskiej w miejscach wag występują wewnętrzne funkcje aktywacji $f_{p,q}$ i wagi są niejako ukryte, podczas gdy w sieci sigmoidalnej wagi pełnią ważną rolę i są widoczne.

Uczenie sieci kołmogorowskiej polega na określaniu zewnętrznych funkcji aktywacji g_q przy ustalonych wewnętrznych funkcjach aktywacji $f_{p,q}$ (stałe "ukryte"wagi), natomiast uczenie sieci sigmoidalych polega na wyznaczeniu wag przy ustalonych funkcjach aktywacji. Sieci kołmogorowskie składają się z dwóch warstw, natomiast sieci sigmoidalne są o dowolnej liczbie warstw i mają strukturę bardziej elastyczną (np. rekurencyjne). W sieciach sigmoidalnych sygnały wejściowe i wyjściowe są ograniczone do zakresu [0,1] lub [-1,1], a w sieciach kołmogorowskich sygnały wejściowe są z zakresu [0,1] a sygnały wyjściowe z zakresu [minimum h,maximum h] (h - aproksymowana funkcja).

Tabela 1.2 przedstawia zalety i wady sieci kołmogorowskiej.

Zalety	Wady
Proste metody uczenia	Potrzebna duża pamięć na funkcje aktywacji
Prosta implementacja metod identyfikacji obiektów dynamicznych	Mało elastyczna struktura sieci
Uniknięcie metod gradientowych i propagacji wstecznej	
Modelowanie funkcji nieróżniczkowalnych (z ostrzami)	

Tablica 1.2: Zalety i wady sieci kołmogorowskiej

Literatura dotycząca sztucznych sieci kołmogorowskich jest bardzo uboga w przeciwieństwie do obszernego piśmiennictwa o klasycznych sieciach sigmoidalnych. Nieliczne prace prezentują Sprecher [7],[8], Igelnik [6] i Kurkova [27]. Autor tej pracy zmodyfikował twierdzenie Kołmogorowa (przez wprowadzenie funkcji wewnętrznej: liniowej, całkowitoliczbowej, potęgowej) i podał wzory prostych i łatwych w obliczeniach algorytmów numerycznych służących do uczenia zmodyfikowanych sieci kołmogorowskich.

Tematyka dalszych rozdziałów jest następująca.

W rozdziałe drugim przedstawiono sigmoidalną sieć neuronową w aspekcie identyfikacji i aproksymacji funkcji ciągłej. Rozdział trzeci jest opisem sieci kołmogorowskiej [1] oraz przedstawieniem algorytmu numerycznego uczenia sieci według Sprechera [7] , [8]. Rozdział czwarty traktuje o sposobie modyfikacji twierdzenia Kołmogorowa. W rozdziale tym przedstawiono oryginalną koncepcję mająca na celu utworzenie prostych i łatwych w obli-

czeniach metod numerycznych uczenia sieci neuronowych. W rozdziale czwartym jest też analiza zbieżności skonstruowanych algorytmów w oparciu o zmodyfikowane twierdzenie Kołmogorowa i błędy aproksymacji funkcji dla różnych sieci: liniowej, potęgowej i indeksowanej. Rozdział piąty przedstawia opisy metod i algorytmów numerycznych autora. Także podane są warunki zbieżności algorytmów uczenia sieci . Rozdział szósty przedstawia badania eksperymentalne systemów statycznego i dynamicznego Hammersteina i Wienera, w których zastosowano zmodyfikowaną sieć neuronową kołmogorowską w celu identyfikacji obiektów nieliniowych oraz porównano algorytmy autora z algorytmem A. Janczaka.

1.2 Teza i cele pracy

Tezą pracy jest:

Zmodyfikowane twierdzenie Kołmogorowa umożliwia konstrukcję sieci neuronowych, nazywanych kołmogorowskimi oraz utworzenie metod uczenia sieci w szybszy i prostszy sposób w porównaniu do istniejących obecnie algorytmów trenowania sieci.

Wiąże się to z opracowaną modyfikacją twierdzenia Kołmogorowa na potrzeby obliczeń numerycznych oraz identyfikacji szerokiej klasy systemów w automatyce, trudnych do identyfikacji innymi algorytmami.

Autor realizuje to założenie przez modyfikację twierdzenia Kołmogorowa o funkcjach wielu zmiennych, co umożliwia osiągnięcie zamierzonych celów.

Celami pracy są:

1. modyfikacja twierdzenia Kołmogorowa i utworzenie na jego podstawie zmodyfikowanych kołmogorowskich sieci neuronowych;
2. utworzenie metod numerycznych uczenia sieci neuronowych w oparciu o zmodyfikowane twierdzenie Kołmogorowa;
3. implementacja algorytmu indeksowanych sieci kołmogorowskich do aproksymacji i identyfikacji obiektów statycznych i dynamicznych Hammersteina i Wienera;
4. porównanie sieci sigmoidalnych i kołmogorowskich z wykazaniem ich wad i zalet.

Rozdział 2

Aproksymacja za pomocą sieci neuronowych

2.1 Wprowadzenie

Sztuczne sieci neuronowe są inteligentym instrumentem matematycznym służącym między innymi do rozwiązywania problemu aproksymacji. Posiadają one zdolność przybliżania funkcji ciągłych. Za pomocą wzorów matematycznych jedne wielkości zastępujemy przez inne wielkości prostsze i bardziej dostępne przy opisywaniu obiektów wielowymiarowych. Oznacza to, że w trakcie procesu uczenia każdą funkcję ciągłą można przybliżyć siecią neuronową. O procesach aproksymacji za pomocą sieci neuronowej mówią twierdzenia Cybenki, Stone'a i Kołmogorowa. Bibliografia dotycząca aproksymujących właściwości sigmoidalnych sieci neuronowych jest bardzo obszerna, w przeciwieństwie do ubogiej literatury sieci kołmogorowskiej. Wyniki prac dotyczących sieci sigmoidalnych są mało konstruktywne ponieważ nie podają wzoru na funkcję aproksymującą (zadaną funkcję ciągłą) realizowaną przez sieć. Zagadnieniem aproksymacji za pomocą sigmoidalnych sieci neuronowych zajmowało się wielu autorów: Cybenko [5], Hornik, Stinchcombe, White [4], Mhaskar, Miccheli [42], Chui, Li [43], Leshno, Lin, Pinkus i Schoken [44], natomiast w odniesieniu do sieci kołmogorowskich wymienić można Sprechera [7],[8], Igelnika [6] oraz autora tej pracy, który podał wzór na aproksymację funkcji ciągłej kołmogorowską siecią neuronową i zbudował algorytmy uczenia tej sieci neuronowej.

Obecnie istnieją dwa twierdzenia dotyczące aproksymacji funkcji ciągłej za pomocą sigmoidalnej sieci neuronowej.

Są to:

1. aproksymacja w oparciu o twierdzenie Cybenki [5]
2. aproksymacja w oparciu o twierdzenie Stone'a [4]

2.2 Twierdzenie Cybenki [5]

Pełne twierdzenie Cybenki wraz z dowodem zostało przedstawione w [5]. Dalej podaję to twierdzenie wraz z komentarzem.

Niech $I_n = [0,1]^n$ będzie kostką n wymiarową liczb rzeczywistych w przestrzeni R^n na wejściu sieci. Przestrzeń funkcji ciągłych na I_n oznaczamy przez $C(I_n)$.

Niech $||f||$ oznacza supremum zbioru funkcji ciągłych $f \in C(I_n)$.

Niech $M(I_n)$ będzie przestrzenią miar Borela [19] na I_n. Miara Borela oznacza każdą miarą określoną na sigma ciele zbiorów borelowskich (rodzinie zbiorów w którym określone jest sumowanie zbiorów). Zbiorami borelowskimi są każde zbiory otwarte z przestrzeni I_n.

Definicja 1

Funkcja f nazywa się *dyskryminatorem* jeśli dla każdej miary $\mu \in M(I_n)$ zachodzi poniższa zależność

$$(\int_{I^n} f(\mathbf{w}_j^T \mathbf{x} + \Theta_j) d\mu(\mathbf{x}) = 0) \Rightarrow (\mu = 0) \tag{2.1}$$

to znaczy jeśli całka z funkcji f po mierze μ zmiennej $\mathbf{x}$ zbioru I^n jest równa zero to miara μ jest zerowa.

Wektory $\mathbf{w}_j \in R^n$ są wagami sigmoidalnej sieci neuronowej z funkcją aktywacji f a liczby $\Theta_j \in R$ są przesunięciem argumentów funkcji aktywacji.

Można pokazać że, jeśli f jest *sigmoidalna* tzn.

$$f(t) \to 1, t \to \infty \tag{2.2}$$

$$f(t) \to 0, t \to -\infty \tag{2.3}$$

to f jest *dyskryminatorem.*

Twierdzenie 1 (Cybenki)

Niech f będzie funkcją ciągłą i dyskryminatorem.

Wtedy klasa funkcji $G(\mathbf{x})$

$$G(\mathbf{x}) = \sum_{j=1}^{m} c_j f(\mathbf{w}_j^T \mathbf{x} + \Theta_j) \tag{2.4}$$

jest gęsta w $C(I_n)$,

c_j, $j = 1, ..., m$ oznaczają wagi na wyjściu. Inaczej mówiąc dla dowolnej funkcji $h \in C(I_n)$ istnieje $G(\mathbf{x})$ takie, że

$$|G(\mathbf{x}) - h(\mathbf{x})| < \varepsilon \tag{2.5}$$

dla $x \in I^n$ i $\varepsilon > 0$ jest dowolnie małą stałą.

To znaczy, że dla dowolnej funkcji ciągłej h można dobrać odpowiednią sigmoidalną sieć neuronową (przez wyznaczenie wag sieci), która ją przybliża (p. wzór (2.5)).

Twierdzenie Cybenki ma szczególne znaczenie, gdyż mówi, że sigmoidala sieć neuronowa ma własciowości aproksymujące.

Z podanego twierdzenia wynika, że sieć jednowarstwowa, jednokierunkowa, o n neuronach, aproksymuje dowolną funkcję ciagłą z dokładnościa zależną od liczby neuronów n, a neurony muszą mieć sigmoidalną funkcję aktywacji.

2.3 Twierdzenie Stone'a [4]

Hornik, Stinchcombe i White [4] zastosowali **twierdzenie Stone'a** [21] do udowodnienia aproksymacyjnych właściwości sigmoidalnej sieci neuronowej.

Kluczowe znaczenie dla zrozumienia twierdzenia Stone'a ma pojęcie algebry. Aby twierdzenie Stone'a było zrozumiałe autor podaje definicję struktury jaką jest algebra.

Definicja 2

Algebra - struktura zamknięta na dodawanie, mnożenie elementów zbioru i mnożenie przez skalary, to znaczy suma i iloczyn elementów algebry, a także iloczyn elementów algebry przez skalar będą należeć do algebry.

Pojęcie algebry występuje w twierdzeniu Stone'a.

Definicja 3

Algebra A rozdziela punkty zwartego zbioru K (np. $K = R^d$) gdy każda funkcja f algebry A dla dowolnych, różnych punktów zbioru K spełnia warunek $f(X) \neq f(Y)$ to znaczy, jeżeli $X, Y \in K, X \neq Y$ i $f \in A$ to $f(X) \neq f(Y)$.

Definicja 4

Algebra A nie znika w żadnym punkcie zwartego zbioru K (to znaczy elementy algebry nie zerują się) gdy każda funkcja f algebry A w dowolnym punkcie zbioru K jest różna od zera; to znaczy $X \in K, f \in A, f(X) \neq 0$.

Twierdzenie Stone'a [21]

Niech A będzie algebrą ciągłych, rzeczywistych funkcji na zwartym zbiorze K.

Jeśli A rozdziela punkty K (p. def. 3) i A nie znika w żadnym punkcie K, to

$$A = C(K) \tag{2.6}$$

gdzie $C(K)$ jest zbiorem funkcji ciągłych na zbiorze zwartym K.

Inaczej mówiąc każda funkcja ciągła na zbiorze zwartym K może być aproksymowana przez funkcje algebry A. Oznacza to, że dla każdej funkcji ciągłej h można dobrać funkcję f algebry A (aby ją przybliżyć). Funkcja f (p. wzór (2.7)) reprezentuje sieć neuronową przybliżającą funkcje h.

Twierdzenie to można zastosować bezpośrednio dla sigma-Π sieci neuronowej to znaczy dla takiej, która ma d wejść i jedno wyjście wyrażone wzorem (2.7)

$$\left\{ f : R^d \to R : f(\mathbf{x}) = \sum_{j=1}^{q} \beta_j \prod_{k=1}^{l_j} G(W_{j,k}(\mathbf{x})) : \mathbf{x} \in R^d, l_j \in N, \beta_j \in R \right\} \tag{2.7}$$

gdzie $W_{j,k}$ jest macierzą wag. Jak wynika ze wzoru (2.7) jest to sieć dwuwarstwowa o q blokach z neuronami z funkcjami aktywcji G i wejsciach będących iloczynem l_j sygnałów wejściowych, co przedstawiono na rysunku 2.1

Funkcje f wyrażone wzorem (2.7) tworzą algebrę. O G zakłada się, że jest funkcją mierzalną i ciągłą.

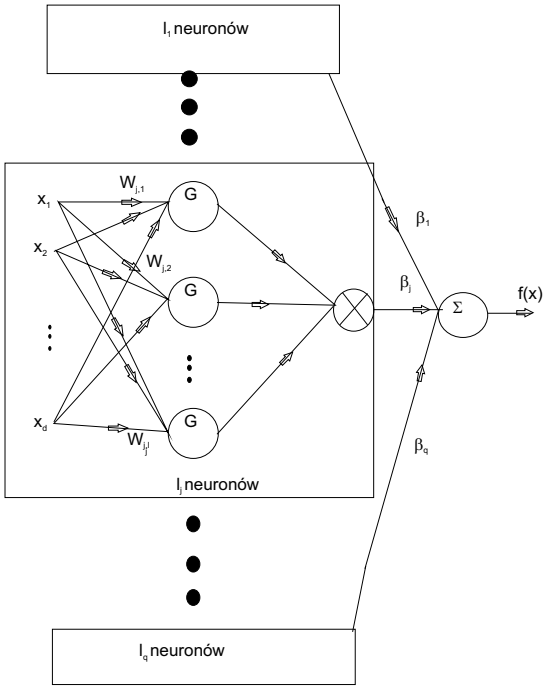
Aby to twierdzenie zastosować dla sieci z sumatorami, to znaczy dla "klasycznej" sieci sigmoidalnej rozpatruje się $G = cos(.)$.

Następnie

1. zamienia się iloczyn funkcji na ich sumę korzystając z tego, że $\cos(a)\cos(b) = \cos(a+b) - \cos(a-b)$; w ten sposób zamieniamy mnożenie na sumę;

2. cosinus przybliżamy funkcjami Gallanta-White'a

$$\psi(\lambda) = (1 + \cos(\lambda + \frac{3\Pi}{2}))\frac{1}{2}\mathbf{1}_{-\frac{\Pi}{2} \leq \lambda \leq \frac{\Pi}{2}} + \mathbf{1}_{\lambda > \frac{\Pi}{2}} \tag{2.8}$$



Rysunek 2.1: Sieć neuronowa sigma pi

gdzie **1** jest funkcją skokową przyjmujacą wartość 1 dla zaznaczonego przedziału;

3. funkcje Gallanta-White'a zastępujemy dowolną funkcją aktywacji ciągłą, sigmoidalną i niemalejącą zwaną w literaturze angielskiej funkcją *squashing*.

Twierdzenie Stone'a podobnie jak twierdzenie Cybenki podkreśla właściwości aproksymujące sieci, ponieważ wskazuje, że dla dowolnej funkcji ciągłej można dobrać (aproksymującą) sigmoidalną sieć neuronową.

Jednak w tym przypadku problem aproksymacji jest formułowany inaczej niż w twierdzeniu Kołmogorowa, bo dobiera się wagi, a nie funkcję aktywacji dla zadanej funkcji ciągłej.

Jednowarstwowa sieć neuronowa o n neuronach i funkcjach aktywacji ciągłych, sigmoidalnych i niemalejących aproksymuje dowolną funkcję ciągłą z dowolnie małym błędem. Jest to wniosek z twierdzenia Cybenki lub Stone'a.

Na przykład funkcja $y = sin(x)$ na przedziale [0,1] może być aproksymowana przez jednowarstwową sieć neuronową o n neuronach (dokładność aproksymacji zależy od ilości neuronów).

2.4 Szacowanie liczby komórek sieci neuronowej

Należy stwierdzić, że istnieje różnica w określeniu funkcji aktywacji w twierdzeniach Stone'a i Cybenki. Stone przyjmuje, że funkcja aktywacji musi być niemalejąca, natomiast Cybenko nie stawia takiego wymagania. W związku z tym sieć neuronowa z inną funkcją aktywacji będzie przybliżała dowolną funkcję ciągłą w twierdzeniu Cybenki a z inną funkcją aktywacji w twierdzeniu Stone'a. Jednak w obu przypadkach aproksymacja jest możliwa.

Twierdzenia te nie wyjaśniają jaka ma być liczba n komórek w sieci neuronowej. Inaczej mówiąc nie wiadomo jak złożona ma być sieć neuronowa, aby przybliżała funkcję ciągłą z przyjętym błędem.

Problem ten rozważał Barron [22], który oszacował błąd aproksymacji ϵ dla n komórek w sieci neuronowej

Wzór Barrona umożliwia oszacowanie liczby neuronów n w jednowarstwowej sigmoidalnej sieci neuronowej tak, aby błąd aproksymacji był mniejszy od pewnej dowolnie małej stałej ϵ.

Rozdział 3

Kołmogorowska sieć neuronowa

3.1 Wstęp

Sieć neuronowa z sigmoidalną funkcją aktywacji wymaga zastosowania skomplikowanych algorytmów uczenia. Proponowana przez autora nowa postać kołmogorowskiej sieci neuronowej pozwala na zastosowanie prostych numerycznie algorytmów uczenia sieci. Dzięki temu możemy uniknąć skomplikowanych i nie zawsze zbieżnych algorytmów gradientowych, a w tym propagacji wstecznej. Prezentowane w literaturze algorytmy uczenia sieci sigmoidalnych wymagają użycia w modelu sieci o co najmniej dwóch warstwach i są skomplikowane w obliczeniach.

W niniejszym rozdziale podaje się oryginalne twierdzenie Kołmogorowa, dotyczące przedstawienia funkcji wielu zmiennych przy pomocy funkcji jednej zmiennej. Umożliwia to prezentację wniosków na temat zmodyfikowanego twierdzenia Kołmogorowa, które jest omówione w rozdziale 4.

3.2 Twierdzenie Kołmogorowa i odpowiadająca mu sieć neuronowa

Twierdzenie Kołmogorowa

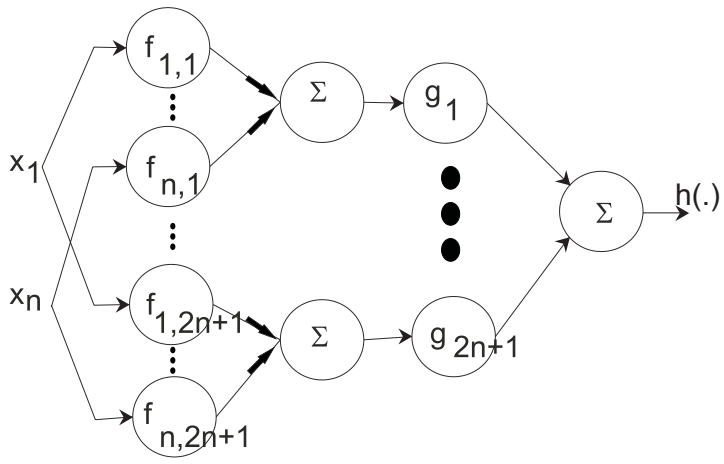
Twierdzenie to brzmi następująco [1]:

Istnieją ciągłe rosnące funkcje $f_{p,q}(x)$ określone na przedziale wejściowym $E = [0, 1]$, czyli o wartościach rzeczywistych w granicach $[0, 1]$ tak, że każda ciągła funkcja $h(x_1, ..., x_n)$ na E^n, to jest na iloczynie kartezjańskim n przedziałów $[0, 1]$, może być przedstawiona w postaci

$$h(\mathbf{x}) = h(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} f_{p,q}(x_p)) \tag{3.1}$$

gdzie g_q są ciągłymi funkcjami jednej zmiennej.

Twierdzenie to ma zastosowanie w teorii sieci neuronowych, gdyż twierdzeniu Kołmogorowa odpowiada sieć neuronowa jak na rys. 3.1. Jest to sieć dwuwawarstwowa, jednokierunkowa. W warstwie wejściowej znajduje się n razy po $2n+1$ neuronów. Wielkościami wejściowymi są zmienne $x_1, x_2, ..., x_n$. Sumatory wewnętrzne sumują wielkości do warstwy



Rysunek 3.1: Neuronowa sieć kołmogorowska.

wyjsciowej g_q, w której mamy $2n+1$ neuronów. Wyjścia są sumowane w końcowym sumatorze. Jest to sieć neuronowa z jednym wyjściem. Sieć ta różni się od sigmoidalych sieci neuronowych, ponieważ nie występują w niej wagi. Miejsce wag zajmują wewnętrzne funkcje aktywacji $f_{p,q}$. Podobnie jak w sieciach sigmoidalnych istnieją funkcje aktywacji g_q zwane tutaj zewnętrznymi funkcjami aktywacji. Uczenie sieci przebiega inaczej niż w sigmoidalnych sieciach neuronowych i polega na wyznaczaniu zewnętrznych funkcji aktywacji przy określonych wewnętrznych funkcjach aktywacji. W oryginalnej wersji twierdzenia Kołmogorowa wagi nie występują.

Z twierdzenia Kołmogorowa wynika wniosek, że sieć neuronowa kołmogorowska może aproksymować funkcję wielu zmiennych.

Lorentz [3] wykazał, że można zastąpić $2n+1$ funkcji g_q przez jedną funkcje g , a Sprecher [3] zmodyfikował funkcje $f_{p,q}$ do $\lambda_p f_q$. Przyjmując to otrzymujemy zmodyfikowany wzór na funkcję aproksymowaną

$$h(\mathbf{x}) = h(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g(\sum_{p=1}^{n} \lambda_p f_q(x_p)) \tag{3.2}$$

Ponadto Sprecher [3] wprowdza wagi między warstwą wewnętrzną a warstwą zewnętrzną, przez co sieć neuronowa staje się podobna do sieci sigmoidalnej, reprezentowanej przez zależność:

$$h(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} \lambda_p f(x_p + (q-1)a)) \tag{3.3}$$

gdzie a jest pewną, odpowiednio dobraną stałą (p. dowód twierdzenia Kołmogorowa [1]) np. $a = \frac{1}{((9n)^k)3n}$ gdzie stała k określa zagęszczenie siatki przestrzennej.

Osowski [3] podaje jeszcze inną postać sieci kołmogorowskiej, w której wprowadza zmienne współczynniki wagowe A_j, C_k, dobierane w procesie uczenia sieci kołmogorowskiej:

$$z_k = \sum_{j=1}^{n} A_j \psi(x_j + b_j) + B_{0,k} \tag{3.4}$$

$$y_i = \sum_{k=1}^{2n+1} C_k g(z_k + d_k) + D_{0,i} \tag{3.5}$$

Wielkości x_i, z_i, y_i oznaczają odpowiednio wielkości wejściowe, wielkości pośrednie i wielkości wyjściowe sieci neuronowej. Ciągłe funkcje ψ, g nie zmieniają się w procesie uczenia. Stałe b_j, d_k, $B_{0,k}$, $D_{0,i}$ wyznaczają przesunięcia odpowiednio argumentu i wartości funkcji.

Sieć neuronowa zbudowana według wzorów (3.4) i (3.5) jest podobna do sieci sigmoidalnej.

3.3 Uczenie kołmogorowskiej sieci neuronowej według Sprechera [7],[8]

Algorytm Sprechera jest jedynym algorytmem uczenia sieci neuronowej kołmogorowskiej dostępnym w literaturze, jednak jest on uwikłany i trudny w użyciu. Uczenie sieci polega na znalezieniu funkcji g we wzorze (3.1).

Wewnętrzne funkcje aktywacji $f_{q,p}$ zostają określone na zbiorze wejściowym w postaci skończonych rozwinięć liczb przedstawionych wzorem (3.6)

$$d_k = \sum_{r=1}^{k} i_r \gamma^{-r} \tag{3.6}$$

gdzie γ jest podstawą rozwinięć liczb wejściowych; na przykład dla rozwinięć dziesiętnych $\gamma = 10$. Funkcje f określa się [7],[8] ze wzoru

$$f(\sum_{r=1}^{k} i_r \gamma^{-r}) = \sum_{r=1}^{k} i'_r 2^{-m_r} \gamma^{-\frac{n^{r-m_r}-1}{n-1}} \tag{3.7}$$

gdzie

$$i'_r = i_r - (\gamma - 2) < i_r > \tag{3.8}$$

i

$$m_r =< i_r > (1 + \sum_{s=1}^{r-1} [i_s].....[i_{r-1}]) \tag{3.9}$$

dla $r = 1, 2, ..., k$

gdzie $< . >$ i $[.]$ są określone następująco

$$\begin{aligned} < \quad & i_1 >= 0 \\ < \quad & i_r >= 0, i_r = 0, 1, .., \gamma - 2 \\ < \quad & i_r >= 1, i_r = \gamma - 1 \\ r \quad & \geq \quad 2 \end{aligned} \tag{3.10}$$

$$[i_1] \quad = \quad 0 \tag{3.11}$$

$$\begin{aligned} [i_r] &= i_r,\ i_r = 0, 1, ..., \gamma - 3 \\ [i_r] &= 1,\ i_r = \gamma - 2, \gamma - 1 \\ r &\geq 2 \end{aligned}$$

Przy obliczaniu wartości

$$f(d_k + (q-1)a) \tag{3.12}$$

przyjmujemy $a = \frac{1}{(\gamma-1)\gamma}$
następnie korzystamy ze wzoru

$$f(d_k + \frac{q-1}{(\gamma-1)\gamma}) = f(\varpi_q(d_k)) + (q-1)\sum_{r=k+1}^{\infty} \gamma^{-\frac{n^r-1}{n-1}} \tag{3.13}$$

gdzie

$$\varpi_q(d_k) = \frac{i_1}{\gamma} + \sum_{r=2}^{k} \frac{i_r + q - 1}{\gamma^r} \tag{3.14}$$

Tak więc funkcje f zadane są a'priori (por. wzór 3.7). Określanie funkcji g_q odbywa się iteracyjnie .

Problemem pozostaje dobór k , które tak należy dobrać, aby

$$|h_{r-1}(\mathbf{x}) - h_{r-1}(\mathbf{x}')| \leq \epsilon \, \|h_{r-1}\| \tag{3.15}$$

gdzie $\mathbf{x} = (x_1, ..., x_n)$, a $\mathbf{x}'$ jest punktem bliskim $\mathbf{x}$ i h_r jest funkcją aproksymującą h w r tym kroku
a wtedy,

$$\left|x_p - x'_p\right| \leq \gamma^{-k}, p = 1, 2, ..., n \tag{3.16}$$

Niech funkcje h_r będą zdefiniowane następująco

$$h_r(\mathbf{x}) = h(\mathbf{x}) - \sum_{q=0}^{2n} \sum_{j=1}^{r} g_q^j \cdot \zeta(\mathbf{x}_q) \tag{3.17}$$

gdzie

$$\zeta(\mathbf{x}_q) = \sum_{p=1}^{n} \lambda_p f(x_p + qa) \tag{3.18}$$

Stałe λ wyznaczamy następująco $\lambda_1 = 1$ i $\lambda_p = \sum_{r=1}^{\infty} \gamma^{-(p-1)\frac{n^r-1}{n-1}}$ dla $p = 2, 3, ..., n$ Stałe te są wykorzystywane we wzorze (3.18).

Widać więc, że notacja algorytmu Sprechera różni się od notacji twierdzenia Kołmogorowa.

Niech funkcja Θ potrzebna do obliczenia zewnętrznych funkcji aktywacji będzie zdefiniowana następująco:

$$\Theta(\mathbf{d}_k^q, y_q) = \sigma(\gamma^{\beta(k+1)}(y_q - \zeta(\mathbf{d}_k^q) + 1)) - \sigma(\gamma^{\beta(k+1)}(y_q - \zeta(\mathbf{d}_k^q) - (\gamma - 2)b_k)) \tag{3.19}$$

a $\sigma : R \to [0,1]$ jest ciągłą funkcją taką, że

$$\begin{aligned} \sigma(x) &\equiv 0, x \leq 0 \\ \sigma(x) &\equiv 1, x \geq 1 \end{aligned} \tag{3.20}$$

natomiast $b_k = \varepsilon_k \sum_{p=1}^{n} \lambda_p$ i $\varepsilon_k = \sum_{r=k+1}^{\infty} \gamma^{-\beta(r)}$ gdzie $\beta(r) = \frac{n^r - 1}{n-1}$
oraz

$$d_{k,p}^q = d_{k,p} + (q-1) \sum_{r=2}^{k} \gamma^{-r} \tag{3.21}$$

gdzie $d_{k,p}^q$ są współrzędnymi wektora $\mathbf{d}_k^q$ (wektor ten ma n współrzędnych).
Aby nauczyć sieć należy określić funkcje g_q^j ze wzoru

$$g_q^j(y_q) = \frac{1}{2n+1} \sum_{\mathbf{d}_k^q} h_{r-1}(\mathbf{d}_k^q) \Theta(\mathbf{d}_k^q, y_q) \tag{3.22}$$

Algorytm Sprechera umożliwił uczenie sieci kołmogorowskiej. Sprecher podał wzór na funkcje wewnętrzne w twierdzeniu Kołmogorowa, czego nie dokonał Kołmogorow. Jednak algorytm uczenia sieci jest bardzo skomplikowany i to zarówno w części wyznaczającej funkcję wewnętrzną (arbitralnie, niezależnie od funkcji h) jak i funkcję zewnętrzną (zależnej od funkcji h). Funkcje wewnętrzne określone są na dziedzinie w postaci skończonych rozwinięć liczb. Takie przeliczanie komplikuje pracę algorytmu. Algorytm do obliczania wewnętrznej funkcji aktywacji wymaga użycia zestawu bardzo skomplikowanych wzorów. Funkcje zewnętrzne są określane iteracyjnie w procesie uczenia. Algorytm ich wyznaczenia wymaga wcześniejszego określenia funkcji wewnętrznych co jest procesem bardzo żmudnym. Również wyznaczanie zewnętrznych funkcji aktywacji jest określone zestawem bardzo skomplikowanych wzorów. Sam warunek zatrzymania (p. wzór (3.15)-(3.16)) jest skomplikowany w obliczeniach. Reasumując algorytm Sprechera jest bardzo skomplikowanym algorytmem nie tylko w obliczeniach ale także w zrozumieniu.

3.4 Podsumowanie

W niniejszym rozdziale przedstawiono klasyczne twierdzenie Kołmogorowa dotyczące aproksymacji funkcji wielu zmiennych przez funkcję jednej zmiennej. Jest to twierdzenie fundamentalne przy konstruowaniu sieci neuronowych ponieważ sieć neuronowa realizuje "sztuczną" funkcję, która przybliża się do zadanej funkcji wielu zmiennych.

Sieć kołmogorowska jest siecią z wyznaczaną podczas uczenia zewnętrzną funkcją aktywacji. Dotychczasowe próby modyfikacji twierdzenia Kołmogorowa przez Lorentza czy Sprechera nie podały prostego algorytmu uczenia sieci. Dlatego w następnym rozdziale przedstawiono modyfikację twierdzenia Kołmogorowa oraz proste algorytmy uczenia sieci neuronowej na jego podstawie.

Rozdział 4

Zmodyfikowana kołmogorowska sieć neuronowa

4.1 Wstęp

Rozdział ten zawiera oryginalną koncepcję autora mającą na celu modyfikację twierdzenia Kołmogorowa. Modyfikacja polega na wprowadzeniu do twierdzenia Kołmogorowa wewnętrznych funkcji: liniowej, potęgowej oraz całkowitoliczbowej. Autor proponuje dwie wersje tego twierdzenia:

1. zmodyfikowaną-podstawową (funkcja wewnętrzna liniowa lub potęgowa),
2. indeksowaną (funkcja wewnętrzna całkowitoliczbowa).

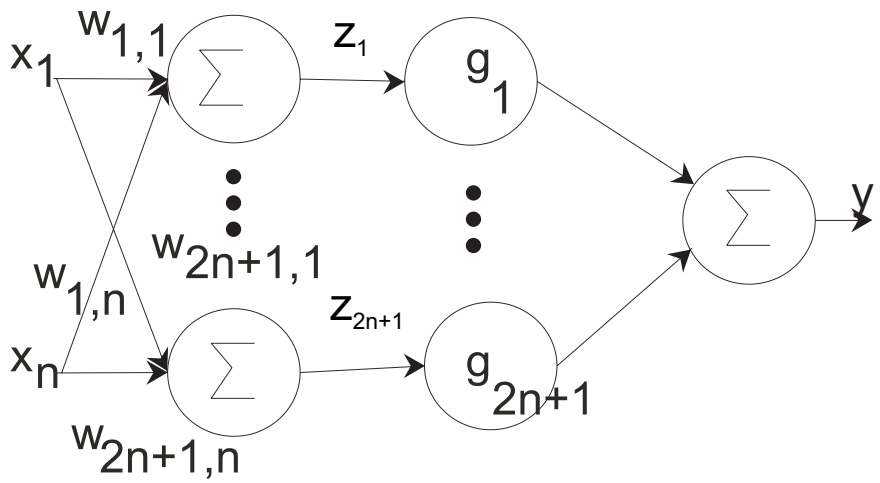
Umożliwia to budowę prostych i łatwych do obliczeń algorytmów numeryczych służących do trenowania sieci neuronowych. Autor przedstawia także właściwości neuronowej sieci liniowej, sieci indeksowanej i sieci potęgowej. Rozdział ten zawiera też opis błędów aproksymacji funkcji ciągłej.

4.2 Zmodyfikowane podstawowe twierdzenie Kołmogorowa

Modyfikacja twierdzenia i odpowiadająca mu sieci neuronowa wynika z potrzeby konstrukcji nowego bardziej efektywnego algorytmu numerycznego uczenia kołmogorowskiej sieci neuronowej. Cel ten osiągnięto wprowadzając do twierdzenia Kołmogorowa wewnętrzne funkcje aktywacji: liniową lub potęgową oraz całkowitoliczbową. Przyjęcie określonych funkcji wewnętrznych umożliwia obliczenie zewnętrznych funkcji aktywacji, które zależą od zadanej przy uczeniu ciągłej funkcji h. W ten sposób otrzymuje się zestaw funkcji zewnętrznych nazwanych przez autora "portretem Kołmogorowa"funkcji h. Jest on uzyskany w wyniku uczenia sieci kołmogorowskiej.

4.2.1 Sieć liniowa

Sieć kołmogorowską z rozdziału 3 można sprowadzić do zmodyfikowanej liniowej kołmogorowskiej sieci neuronowej przyjmując:



Rysunek 4.1: Zmodyfikowana podstawowa liniowa neuronowa sieć kołmogorowska.

$$f_{p,q}(x_p) = w_{q,p}x_p \tag{4.1}$$

dla

$$1 \leq p \leq n \tag{4.2}$$

i

$$1 \leq q \leq 2n+1 \tag{4.3}$$

Tak zmodyfikowana podstawowa sieć neuronowa kołmogorowska przedstawiona jest na rys. 4.1. W zmodyfikowanej postaci sieci zamiast neuronów wewnętrznych wprowadzono wagi $w_{p,q}$, które zastępują wewnętrzne funkcje aktywacji $f_{p,q}$, a neurony realizujące zewnętrzną funkcję aktywacji są takie same jak na rysunku 3.1.

Modyfikacja twierdzenia Kołmogorowa jest konieczna, ze względu na to, że Kołmogorow [1] nie podał wzoru na funkcje wewnętrzne, mówiąc jedynie o ich istnieniu. Poza tym twierdzenie Kołmogorowa jest twierdzeniem o reprezentacji funkcji ciągłej wielu zmiennych w postaci złożenia i sumy funkcji jednej zmiennej, w której funkcje zewnętrzne wyznacza się w nieskończenie długim procesie iteracyjnym, co jest niemożliwe w praktyce. Modyfikacja twierdzenia Kołmogorowa pozwoliła na sformułowanie twierdzenia o aproksymacji funkcji ciągłej wielu zmiennych,w której funkcje zewnętrzne wyznacza się w jednym kroku.

Poniżej przedstawiono twierdzenia dotyczące sieci liniowej. Nazwa sieć liniowa bierze się od sposobu wyznaczenia wyjść warstwy wewnętrznej, które są liniowo proporcjonalne do wejść warstwy wewnętrznej,(patrz rys. 4.1 i wzór (4.1)).

Niech E oznacza przedział liczb rzeczywistych x_p od 0 do 1, to jest $E = [0, 1]$ na wejściu sieci.

Niech dana będzie aproksymowana funkcja n zmiennych $h(x_1, ..., x_n)$.

Zbiór liczb rzeczywistych na którym aproksymowana jest funkcja h dany jest wzorem $D_K = \left\{\frac{1}{K}i : i = 0, 1, ...K\right\}$ gdzie $''$:$''$ oznacza zwrot "takie, że". Wskaźnik i wskazuje numer porządkowy liczby w zbiorze D_K. Zbiór D_K wyznacza "ziarno", to jest zbiór punktów na odcinku liczb rzeczywistych [0,1] i jak łatwo zauważyć ze wzrostem K "ziarno" będzie coraz gęstsze. Liczba K przyjmuje wartości całkowite dodatnie to jest 1,2,... .

Zbiór $D_K^n = D_K \times D_K \times ... \times D_K$ n razy jest produktem kartezjańskim zbiorów D_K.

Podobnie jak w pracy Kołmogorowa [1] przyjmuje się przedziały liczb rzeczywistych $A^q_{K,i}$ w obrębie przedziału E:

$$A^q_{K,i} = \frac{1}{K}[i - 1 - \frac{q}{3n}, i - \frac{1}{3n} - \frac{q}{3n}] \tag{4.4}$$

Wskaźnik i pokazuje numer przedziału $A^q_{K,i}$ w przedziale E. Wskaźnik i zmienia się od 0 do K. Prawy koniec przedziału jest przesunięty o $\frac{1}{3n}$ względem lewego końca kolejnego przedziału. Przesunięcie to zapewnia, że przedziały nie mają punktów wspólnych ze sobą, a z drugiej strony zapewniają przecinanie się ze zbiorami D^n_K, co jest potrzebne dla prawdziwości przedstawionego niżej lematu 1. Z kolei przesunięcie całego przedziału o $\frac{q}{3n}$ jest potrzebne aby zewnętrzne funkcje aktywacji przybliżały funkcje aproksymowaną w różnych miejscach i nie za bardzo odległych od siebie (stąd ułamek $\frac{q}{3n}$ jest mniejszy od 1).

Iloczyn zbiorów $A^q_{K,i}$ oznaczmy przez $S^q_{K,i_1,...,i_n}$:

$$S^q_{K,i_1,...,i_n} = \prod_{p=1}^{n} A^q_{K,i_p} \tag{4.5}$$

Zbiór $S^q_{K,i_1,...,i_n}$ jest zbiorem wejściowym przekształcanym przez wewnętrzne funkcje aktywacji $f_{p,q}$ (p. wzór (4.1)).

Poniższy lemat jest potrzebny do dowodu zmodyfikowanego twierdzenia Kołmogorowa.
Lemat 1 (o pokryciu)

Zbiór hipersześcianów $S^q_{K,i_1,...,i_n}$ przy ustalonym K i zmiennym q oraz $i_1, ..., i_n$ pokrywa zbiór D^n_K tak, że każdy punkt z D^n_K jest pokryty $2n + 1$ razy

Dowód

Dla $n = 2$ długość przedziału $A^q_{K,i}$ wynosi $\frac{5}{6}\frac{1}{K}$ (patrz wzór (4.4)), a odległość lewego końca przedziału $A^q_{K,i}$ od najbliższej liczby ze zbioru D_K wynosi $\frac{5}{6}\frac{1}{K}$; D^n_K zawsze jest pokryte hipersześcianem $S^q_{K,i_1,...,i_n}$. Dla $n > 2$ dowód jest analogiczny.

♠

Na podstawie lematu 1 można powiedzieć, ze zbiór D^n_K przecina się z przedziałami $A^q_{K,i}$ dla $q = 1, 2, ..., 2n + 1$. Liczby zbioru D^n_K stanowią "siatkę przestrzenną" na której aproksymowana jest funkcja h. Punkty przedziałów $A^q_{K,i}$ są bliskie punktom siatki przestrzennej D^n_K przy dużych wartościach K (p. wzór (4.4)).

Autor proponuje następującą modyfikację twierdzenia Kołmogorowa.

Twierdzenie 1(zmodyfikowane twierdzenie Kołmogorowa)

Dowolną funkcje $h(x_1, ..., x_n)$ ciągłą i rzeczywistą na $E \times ... \times E$ $(n - razy)$ w punktach zbioru D^n_K można z dowolnie małym błędem aproksymować wzorem

$$h_a(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} w_{q,p} x_p) \tag{4.6}$$

gdzie g_q są rzeczywiste i wyrażone wzorem $g_q = \frac{1}{2n+1} h(\varsigma^q_{K,i_1}, ..., \varsigma^q_{K,i_n})$, a lewe końce przedziałów A^q_{K,i_p} oznaczono przez ς^q_{K,i_p},

i $w_{q,p}$ są są wagami synaptycznymi spełniającymi warunek (wyjaśnienie doboru wag w dowodzie twierdzenia):

$$w_{q,p} \geq w_{q,p-1}(K+1) \tag{4.7}$$

gdzie waga $w_{q,1} > 0$ może być dowolnie mała, a liczba K definiuje zbiór D_K

Dowód

Zgodnie z rys. 4.1 przyjmijmy, że

$$z_q = \sum_{p=1}^{n} w_{q,p} x_p \tag{4.8}$$

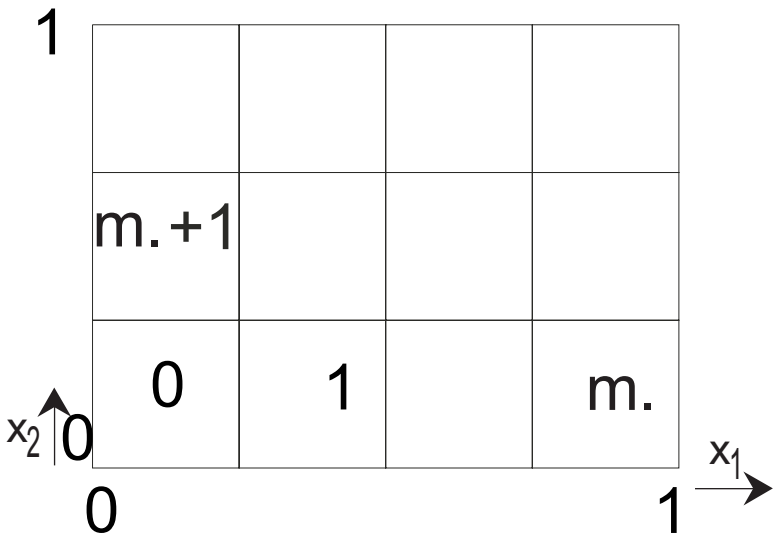
Wadą warunku (4.7) jest niekorzystne narastanie wag przy dużej dokładności obliczeń. Wynika to z warunku nałożonego na wagi $w_{q,p} \geq w_{q,p-1}(K+1)$. Przy dużej dokładności czyli przy dużym K, kolejne wagi $w_{q,p}$ przy zmieniającym się p będą narastały do dużych wartości, co niekorzystnie odbije się na dokładności obliczeń zmiennej z_q, bo pewne składniki z_q będą małe a inne duże czy bardzo duże.

Funkcja określona wzorem (4.8) przekształca zbiór $S^q_{K,i_1,...,i_n}$ w zbiór $\Delta^q_{K,i_1,...,i_n}$. Ponieważ funkcja o wartościach z_q jest ciągła (jako suma funkcji ciągłych), zatem na mocy twierdzenia o przekształceniu ciągłym przestrzeni spójnej ([2] tw.1.7.3 str.63) wnioskujemy, że obraz spójnego zbioru $S^q_{K,i_1,...,i_n}$, to jest $\Delta^q_{K,i_1,...,i_n}$, jest zbiorem spójnym. Czyli każdy wielowymiarowy zbiór dziedziny na wejściu $S^q_{K,i_1,...,i_n}$ funkcji h przejdzie w przedział jednowymiarowy co wynika z właściwości funkcji (4.8). Redukcja wymiaru jest konieczna, bo funkcje g_q są określone na zbiorze jednowymiarowym z (funkcje jednej zmiennej).

Wagi synaptyczne $w_{p,q}$ należy tak dobrać, aby przedziały $\Delta^q_{K,i_1,...,i_n}$ nie pokrywały się wzajemnie (to znaczy aby przekrój przedziałów był zbiorem pustym) a funkcje wyrażone wzorem (4.8) były różnowartościowe. Gdyby przedziały się pokryły to nie byłaby możliwa aproksymacja dla tych przedziałów według wzoru $g_q(z(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})) = \frac{1}{2n+1} h(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})$ (z takie samo bo punkt $(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})$ i punkt $(\varsigma^q_{K,i'_1},...,\varsigma^q_{K,i'_n})$ przechodziłby w punkt z zgodnie ze wzorem (4.8) a prawa strona wzoru różna). Wówczas mamy sprzeczny układ równań: $g_q(z(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})) = \frac{1}{2n+1} h(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})$ i $g_q(z(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})) = \frac{1}{2n+1} h(\varsigma^q_{K,i'_1},...,\varsigma^q_{K,i'_n})$.

Zauważmy, że rozpatrujemy wartości funkcji z_q wyrażonej wzorem (4.8) na "siatce przestrzennej" D^n_K. Obrazem tych punktów są punkty przedziałów $\Delta^q_{K,i_1,...,i_n}$. Aby punkty z przedziałów $\Delta^q_{K,i_1,...,i_n}$ nie pokryły się wzajemnie wystarczy, aby obrazy punktów "siatki przestrzennej" nie nakładały się. Współczynniki $w_{p,q}$ w sieci liniowej, które są stałe w procesie uczenia sieci można dobrać na kilka podanych niżej sposobów (aby punkty z przedziałów Δ nie pokryły się). Jednym z nich jest sposób określony przez wzór (4.7).

Skutek doboru wag według wzoru $w_{q,p} \geq w_{q,p-1}(m+1)$ ilustruje schematycznie rysunek 4.2. W zależności od doboru wag mamy odpowiednio ponumerowaną dziedzinę funkcji h. Wagi sieci liniowej powinny spełniać warunek $w_{2,1} = (m+1)w_{1,1}$ aby zmiana wielkości wejściowej x_2 o jedną jednostkę (zmiana minimalna) odpowiadała zmianie wielkości wejściowej x_1 o $m+1$ jednostek (cały zakres zmian). Wtedy obszar na rysunku 4.2 będzie ponumerowany wzajemnie jednoznacznie. Autor dąży do tego, aby obszar wielowymiarowy, (na rysunku 4.2 dwuwymiarowy), odwzorować w przedział jednowymiarowy o wartościach całkowitych w sposób wzajemnie jednoznaczny (bijekcja). Jeden z takich sposobów jest przedstawiony na rys. 4.2. Punkt wielkości wejściowej o dwóch współrzędnych rzeczywistych po transformacji ma tylko jedną współrzędną całkowitą. Wagi sieci liniowej muszą mieć taką wartość, aby po wymnożeniu wielkości wejściowych przez te wagi (patrz wzór



Rysunek 4.2: Transformacja zbioru dwuwymiarowego x_1, x_2 w liczby całkowite.

(4.8)) "zamienić" obszar wielowymiarowy w zbiór jednowymiarowy (w sposób wzajemnie jednoznaczny - bijekcja), a w szczególnym przypadku (p. sieci indeksowane), w podzbiór zbioru liczb całkowitych.

Rozważmy przypadek, gdy warunek (4.7) nie jest spełniony dla $n = 2$ i $k = 1$. Gdyby przyjąć $w_{2,1} = 18$ i $w_{1,1} = 18$, to punkt zbioru D_k^2 $(\frac{1}{2}, \frac{1}{18})$ przejdzie w punkt $9+1 = 10$, bo $z = w_{1,1}x_1 + w_{2,1}x_2 = 18 * \frac{1}{2} + 18 * \frac{1}{18}$ a punkt $(0, \frac{10}{18})$ przejdzie w punkt 10 i odwzorowanie wyrażone wzorem (4.8) nie będzie różnowartościowe (bijekcja).

Dalej po przedstawieniu uwag o wagach kontynuujemy dowód twierdzenia.

Niech dana będzie funkcja aproksymująca

$$h_a(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} w_{q,p}x_p) \tag{4.9}$$

Lewe końce przedziałów A_{K,i_p}^q oznaczono przez ς_{K,i_p}^q

$$\varsigma_{K,i_p}^q \in A_{K,i_p}^q \tag{4.10}$$

wtedy

$$z_q \in \Delta_{K,i_1,...,i_n}^q \tag{4.11}$$

Wartości z_q funkcji wyrażonej wzorem (4.8) są lewymi końcami przedziału $\Delta_{K,i_1,...,i_n}^q$.

Wartość funkcji g_q jest wyrażona wzorem (4.12). W dowodzie oryginalnego twierdzenia Kołmogorowa proponowano podobny wzór do (4.12) z mianownikiem ułamka równym $n + 1$). Autor udowodnił zmodyfikowane twierdzenie Kołmogorowa ze wzorem (4.12). Wartość mianownika ułamka równa $2n + 1$ wynika z tego, że mamy $2n + 1$ zewnętrznych funkcji aktywacji. Funkcja aproksymująca h_a składa się z $2n + 1$ zewnętrznych funkcji aktywacji $g_q(z(\varsigma_{K,i_1}^q, ..., \varsigma_{K,i_n}^q))$.

$$g_q(z(\varsigma_{K,i_1}^q, ..., \varsigma_{K,i_n}^q)) = \frac{1}{2n+1} h(\varsigma_{K,i_1}^q, ..., \varsigma_{K,i_n}^q) \tag{4.12}$$

Dalej, podobnie jak w oryginalnym dowodzie twierdzenia Kołmogorowa [1], szacuje się różnicę $h(x_1, ..., x_n) - h_a(x_1, ..., x_n)$.

Każdy składnik sumy zewnętrznej (4.9) można wyrazić wzorem

$$g_q(z(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})) = \frac{1}{2n+1}h(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n}) = \frac{1}{2n+1}h(x_1,...,x_n) + \delta \tag{4.13}$$

gdzie

$$|\delta| \leq \frac{1}{2n+1}\epsilon \tag{4.14}$$

bo można dobrać K tak duże aby zagęścić "siatkę przestrzenną" aby

$$\left|h(x_1,...,x_n) - h(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})\right| \leq \frac{1}{2n+1}\epsilon \tag{4.15}$$

gdzie ϵ oznacza dodatnią, dowolnie małą stałą (w idealnym przypadku $\epsilon = 0$). Rozumowanie w podanym dowodzie jest prawdziwe, bo funkcja h jest ciągła, a więc przez zwiększenie K, a tym samym przez zawężenie $A^q_{K,i}$ uzyskamy dowolnie małą wartość zmian wartości funkcji h na przedziale $A^q_{K,i}$

Stąd po przekształceniach (4.9),(4.13) i (4.15) otrzymamy

$$|h(x_1,...,x_n) - h_a(x_1,...,x_n)| \leq \epsilon \tag{4.16}$$

Ponieważ ϵ jest dowolnie małą stałą, a tym samym

$$h_a \to h \tag{4.17}$$

czyli funkcja aproksymująca wyrażona wzorem (4.9) dąży do funkcji h, co kończy dowód zmodyfikowanego twierdzenia Kołmogorowa.

♠

Algorytm uczenia liniowej zmodyfikowanej sieci kołmogorowskiej dla badanej funkcji n zmiennych jest następujący:

1. ustal wartości początkowe i_1,i_2,...,i_n, K i wagi $w_{q,p}$
2. wyznacz x_1,x_2,...,x_n (z uwzględnieniem przedziałów $A^q_{K,i}$ wiążącego x_p z i_p)
3. wyznacz z_q ze wzoru $z_q = \sum_{p=1}^{n} w_{q,p}x_p$
4. wyznacz $\varsigma^q_{K,i_1} = \frac{1}{K}[i_1 - 1 - \frac{q}{3n}]$,..., $\varsigma^q_{K,i_1} = \frac{1}{K}[i_n - 1 - \frac{q}{3n}]$
5. wyznacz $g_q(z_q) = \frac{1}{2n+1}h(\varsigma^q_{K,i_1},...,\varsigma^q_{K,i_n})$ z wczytanych wyjść $y = h()$
6. zmieniaj i_1,...,i_n aż wyczerpie się wszystkie wartości i po każdej zmianie przejdź do 2

Zmienne i_1,i_2,...,i_n są zmiennymi pomocniczymi i wewnętrznymi potrzebnymi do wyznaczenia zmiennych ς^q_{K,i_1},...,ς^q_{K,i_n}.

Jak widać algorytm jest nieco skomplikowany i wadą jego jest występowanie "niekorzystnego narastania wag" opisanego w dowodzie twierdzenia 1.

Struktura kołmogorowskiej liniowej sieci neuronowej jest identyczna ze strukturą sieci sigmoidalnej, natomiast inny jest jej sposób trenowania. W sieciach kołmogorowskich wagi są stałe ale wyznaczana jest zewnętrzna funkcja aktywacji. Kołmogorowska sieć liniowa ma prostszy algorytm uczenia o mniejszej złozoności obliczeniowej. W zwiazku z powyższym trenowanie kołmogorowskiej sieci liniowej nie wymaga stosowania złożonych algorytmów gradientowych i propagacji wstecznej.

4.2.2 Sieć potęgowa

W celu eliminacji problemu "narastania wag" (omówionego w podrozdziale 4.2.1) wprowadzono sieci potęgowe. W sieci potęgowej wewnętrzne funkcje aktywacji są funkcjami potęgowymi. Oznacza to że:

$$f_{p,q}(x) = \beta(x)^{\delta} \tag{4.18}$$

gdzie $x = \frac{n-r}{\alpha}$ a δ jest liczbą naturalną, β współczynnikiem rzeczywistym, stała α jest współczynnikiem normalizującym wejście do wartości od 0 do 1 ($\frac{n}{\alpha}$ ma wartość między 0 a 1), a r jest przesunieciem wielkości wejściowych (koniecznym dla spełnienia zmodyfikowanego twierdzenia Kołmogorowa, co zostało uzasadnione poniżej p. punkt 1) i 2), aby $f_{p,q}(x)$ była przekształceniem różnowartościowym), a n liczbą całkowitą odpowiadającą liczbie x.

Kołmogorowskie sieci potęgowe różnią się od sigmoidalnych sieci potęgowych (zwanych wielomianowymi) tym, że funkcjami potęgowymi są wewnętrzne funkcje aktywacji zastępujące wagi a nie zewnętrzne funkcje aktywacji. Zaletą tych sieci jest prostota wewnętrznych funkcji aktywacji (wsród klasy możliwych funkcji) ale postać wewnętrznych funkcji aktywacji jest bardziej skomplikowana od wag w sieci liniowej.

Aby zmodyfikowane twierdzenie Kołmogorowa pozostało prawdziwe, obrazy punktów "siatki przestrzennej', realizowane przez funkcję (4.18), nie mogą się wzajemnie pokryć (co oznacza, że przekształcenie $f_{p,q}(x)$ jest różnowartościowe). Odpowiednikiem poniższego rozważania dla sieci liniowej jest warunek (4.7) nałożony na wagi. Rozważania ogólne są tu bardzo trudne i dlatego ograniczę się do szczególnych przypadków.

1)Aproksymowana funkcja h jest funkcją 2 zmiennych, a funkcje aktywacji w potęgowej sieci neuronowej są $f_{1,q}(n) = \frac{n-r}{\alpha}$, $f_{2,q}(n) = \beta(\frac{n-r}{\alpha})^2$, $q = 1, 2, ..., 2n + 1$. Sprawdźmy kiedy obraz punktów "siatki przestrzennej" odwzorowanych przez przekształcenie (4.18) pokryje się wzajemnie; może to zajść kiedy punkty o współrzędnych (n_1, n_2) i (n_3, n_4) (będące liczbami całkowitymi z przedziału $[0, \alpha]$) i reprzentującymi zmienną wewnętrzną i pomocniczą po przekształceniu pokryją się wzajemnie, czyli gdy

$$\frac{n_1 - r}{\alpha} + \beta(\frac{n_2 - r}{\alpha})^2 = \frac{n_3 - r}{\alpha} + \beta(\frac{n_4 - r}{\alpha})^2 \tag{4.19}$$

a n_1, n_2, n_3, n_4 są pomocniczymi liczbami całkowitymi (reprezentującymi wielkości wejściowe).

Po przekształceniu mamy zależność

$$\alpha(n_1 - r) - \alpha(n_3 - r) = \beta[(n_4 - r)^2 - (n_2 - r)^2] \tag{4.20}$$

po przekształceniu jest

$$\alpha(n_1 - n_3) = \beta(n_4 - n_2)(n_4 + n_2 - 2r) \tag{4.21}$$

czyli

$$\alpha(n_1 - n_3) = \beta(n_4 - n_2)(n_4 + n_2) - \beta(n_4 - n_2)(2r) \tag{4.22}$$

Dla uproszczenia przyjmujemy, że $\beta = 1$; wtedy, gdy $(n_4 - n_2)(2r)$ będzie niecałkowite i równość (4.22) nie będzie spełniona to zmodyfikowane twierdzenie Kołmogorowa, realizowane przez przekształcenie (4.18)(funkcje wewnętrzne) w sieci potęgowej będzie prawdziwe dla wszystkich wielkości wejściowych.

2)Aproksymowana funkcja h jest funkcją 3 zmiennych i funkcje aktywacji w potęgowej sieci neuronowej są $f_{1,q}(n) = \frac{n-r}{\alpha}$, $f_{2,q}(n) = (\frac{n-r}{\alpha})^2$, $f_{3,q}(n) = (\frac{n-r}{\alpha})^3$, $q = 1, 2, ..., 2n+1$. Sprawdźmy kiedy obraz punktów "siatki przestrzennej" odwzorowanych przez przekształcenie (4.18) pokryje się wzajemnie, czyli kiedy zajdzie poniższe równanie

$$\frac{n_1 - r}{\alpha} + (\frac{n_2 - r}{\alpha})^2 + (\frac{n_3 - r}{\alpha})^3 = \frac{n_4 - r}{\alpha} + (\frac{n_5 - r}{\alpha})^2 + (\frac{n_6 - r}{\alpha}) \tag{4.23}$$

Przekształcając

$$\alpha^2(n_1 - n_4) = \alpha[(n_2 - r)^2 - (n_5 - r)^2] + [(n_3 - r)^3 - (n_6 - r)^3] \tag{4.24}$$

otrzymuje się

$$\begin{aligned} \alpha^2(n_1 - n_4) &= \alpha[(n_2 - n_5)(n_2 + n_5 + 2r) + \\ &\quad (n_3 - n_6)[\{n_3 - r)^2 + (n_3 - r)(n_6 - r) + (n_6 - r)^2] \end{aligned} \tag{4.25}$$

czyli

$$\begin{aligned} \alpha^2(n_1 - n_4) &= \alpha(n_2 - n_5)(n_2 + n_5) + 2\alpha(n_2 - n_5)r + \\ &\quad (n_3 - n_6)[n_3^2 - 2rn_3 + r^2 + n_6^2 - 2rn_6 + \\ &\quad r^2 + n_3n_6 + r^2 - r(n_3 + n_6)] \end{aligned} \tag{4.26}$$

Stąd

$$\begin{aligned} \alpha^2(n_1 - n_4) - \alpha(n_2 - n_5)(n_2 + n_5) &= \\ &\quad 2\alpha(n_2 - n_5)r + \\ &\quad (n_3 - n_6)[n_3^2 + n_6^2 - r(3n_3 + 3n_6) + 3r^2] \end{aligned} \tag{4.27}$$

czyli

$$\begin{aligned} &\alpha^2(n_1 - n_4) - \alpha(n_2 - n_5)(n_2 + n_5) - \\ (n_3 - n_6)(n_3^2 + n_6^2) &= 2\alpha(n_2 - n_5)r - \\ &\quad 3(n_3 - n_6)(n_3 + n_6)r + 3(n_3 - n_6)r^2 \end{aligned} \tag{4.28}$$

stąd

$$\begin{aligned} &\alpha^2(n_1 - n_4) - \alpha(n_2 - n_5)(n_2 + n_5) - \\ (n_3 - n_6)(n_3^2 + n_6^2) &= \\ &\quad r[2\alpha(n_2 - n_5) - 3(n_3 - n_6)(n_3 + n_6) + 3(n_3 - n_6)r] \end{aligned} \tag{4.29}$$

Kiedy wartość wyrażenia:

$$2\alpha(n_2 - n_5) - 3(n_3 - n_6)(n_3 + n_6) + 3(n_3 - n_6)r \tag{4.30}$$

będzie niecałkowita, wtedy spełnione będzie zmodyfikowane twierdzenie Kołmogorowa dla punktów "siatki przestrzennej" w sieci potęgowej realizowane przez przekształcenie (4.18).

Algorytm uczenia sieci potęgowej dla przypadku 1) jest następujący:

1. ustal wartości początkowe n_1,n_2,K,α,r
2. wyznacz $\varsigma^q_{K,n_1} = \frac{1}{K}(n_1 - \frac{q}{6})$,$\varsigma^q_{K,n_2} = \frac{1}{K}(n_2 - \frac{q}{6})$
3. wyznacz $z_q = \frac{n_1 - r}{\alpha} + (\frac{n_2 - r}{\alpha})^2$
4. wyznacz $g_q(z_q) = \frac{1}{5}h(\varsigma^q_{K,n_1}, \varsigma^q_{K,n_2})$
5. zmieniaj n_1,n_2 aż wyczerpie się wszystkie wartości i po każdej zmianie przejdź do 2

Algorytm uczenia sieci potęgowej dla przypadku 2) jest następujący:

1. ustal wartości początkowe n_1,n_2,n_3,K,α,r
2. wyznacz $\varsigma^q_{K,n_1} = \frac{1}{K}(n_1 - \frac{q}{9})$,$\varsigma^q_{K,n_2} = \frac{1}{K}(n_2 - \frac{q}{9})$, $\varsigma^q_{K,n_3} = \frac{1}{K}(n_3 - \frac{q}{9})$
3. wyznacz $z_q = \frac{n_1 - r}{\alpha} + (\frac{n_2 - r}{\alpha})^2 + (\frac{n_3 - r}{\alpha})^3$
4. wyznacz $g_q(z_q) = \frac{1}{5}h(\varsigma^q_{K,n_1}, \varsigma^q_{K,n_2}, \varsigma^q_{K,n_3})$
5. zmieniaj n_1,n_2,n_3 aż wyczerpie się wszystkie wartości i po każdej zmianie przejdź do 2

Wielkości trenujące są wczytane w punkcie 4. Wielkości z punktu 1 są zmiennymi wewnętrznymi i pomocniczymi niezbędnymi do pracy algorytmu.

Przykład zastosowań sieci potęgowej przedstawiono w podrozdziale 4.2.5.

4.2.3 Przykład sieci liniowej dla funkcji liniowej

Pierwszy przykład sieci liniowej przeprowadzono dla funkcji liniowej. Prostota funkcji umożliwia przeprowadzenie dokładnych rachunków i pozwala czytelnikowi na zorientowanie się jak przebiega proces uczenia sieci kołmogorowskiej.

Funkcja aktywacji sigmoidalnych sieci neuronowych jest nieliniowa. W sieciach kołmogorowskich funkcja aktywacji dostosowuje się do badanej funkcji. W badanym przypadku funkcja akywacji będzie liniowa.

Rozpatrzmy jak przebiega proces uczenia sieci liniowej dla funkcji dwóch zmiennych (n=2)

$$h(x_1, x_2) = x_1 + x_2 \tag{4.31}$$

która przedstawia płaszczyznę prostopadłą do wektora $[1, 1, -1]$.

Wybrano do badania funkcję liniową o prostej postaci aby uzyskać maksymalnie prosty przebieg obliczeń i klarowne wyniki procesu uczenia.

Dla $n = 2$ należy wyznaczyć $2n + 1$ to jest 5 zewnętrznych funkcji aktywacji g_1, g_2, g_3, g_4, g_5 zgodnie ze wzorem (4.12) w zmodyfikowanym twierdzeniu Kołmogorowa potrzebnych do wyznaczenia funkcji aproksymującej.

Funkcje te wyznaczmy ze wzoru (4.12) $g_q(z) = \frac{1}{5}h(\varsigma^q_{18,i_1}, \varsigma^q_{18,i_2})$ zgodnie ze zmodyfikowanym twierdzeniem Kołmogorowa.

Przyjmijmy że zmienna K (określającą dokładność aproksymacji) ma wartość 18. (Por. wzór (4.4)). Kołmogorow zaleca aby $K = 9n$, gdzie n to liczba zmiennych funkcji h;

czyli $K = 9 * 2 = 18$. Autor udowodnił zmodyfikowane twierdzenie Kołmogorowa dla K całkowitego i dodatniego.

Przedziały liczb rzeczywistych $A^q_{18,i}$ zgodnie ze wzorem (4.4) wynoszą $A^q_{18,i} = \frac{1}{18}[i-1-\frac{q}{6}, i-\frac{1}{6}-\frac{q}{6}]$

Punkty $\varsigma^q_{18,i} = \frac{1}{18}(i-1-\frac{q}{6})$, są lewymi końcami przedziałów $A^q_{18,i}$ (p. def. przedziałów $A^q_{18,i}$)

Zmienna i przyjmuje wartości od 2 do 19 (przedziały $A^q_{18,i}$ dla danych i mieszczą się w przedziale E, por. wzór (4.4)).

Wagi synaptyczne wyznaczone są z warunku $w_{q,p} \geq 19w_{q,p-1}$(por. zmodyfikowane twierdzenie Kołmogorowa, wzór 4.7). Przyjmijmy, że $w_{q,1} = 1$ a $w_{q,2} = 19$ (nie przyjmujemy wagi równej np. 20 bo skomplikowało by to obliczenia; p. rys. 4.2 dla wag dobieranych według wzoru (4.7) - numeracja jest bez luk to jest po kwadracie o numerze m następuje kwadrat o numerze $m+1$).

Wartości z wyznaczamy ze wzoru (por. wzór 4.8)

$$z = \sum_{p=1}^{2} w_{q,p}x_p = x_1 + 19x_2 \tag{4.32}$$

czyli dla punktów $\varsigma^q_{1,i}$

$$z = \frac{1}{18}(i_1 - 1 - \frac{q}{6}) + \frac{19}{18}(i_2 - 1 - \frac{q}{6}) \tag{4.33}$$

po uporządkowaniu

$$z = \frac{1}{18}i_1 + \frac{19}{18}i_2 - \frac{10}{18}\frac{q}{3} - \frac{20}{18} \tag{4.34}$$

czyli

$$z = \frac{1}{18}i_1 + \frac{19}{18}i_2 - \frac{5}{27}q - \frac{10}{9} \tag{4.35}$$

gdzie i_1 reprezentuje zmienną x_1 a i_2 reprezentuje zmienną x_2.

Celem uproszczenia przyjmujemy $q = 0$ (zmienna q zależy od numeru zewnętrznej funkcji aktywacji. Na przykład dla pierwszej funkcji aktywacji $q = 1$).

Wartości z dla różnych i_1, i_2 przedstawia rys. 4.3. Przedstawienie zmiennej x_p w postaci liczb całkowitych i_p pozwala na lepszą czytelność wykresów i ma sens też w zmodyfikowanym indeksowanym twierdzeniu Kołmogorowa, gdzie wielkości wejściowe są transformowane na liczby całkowite.

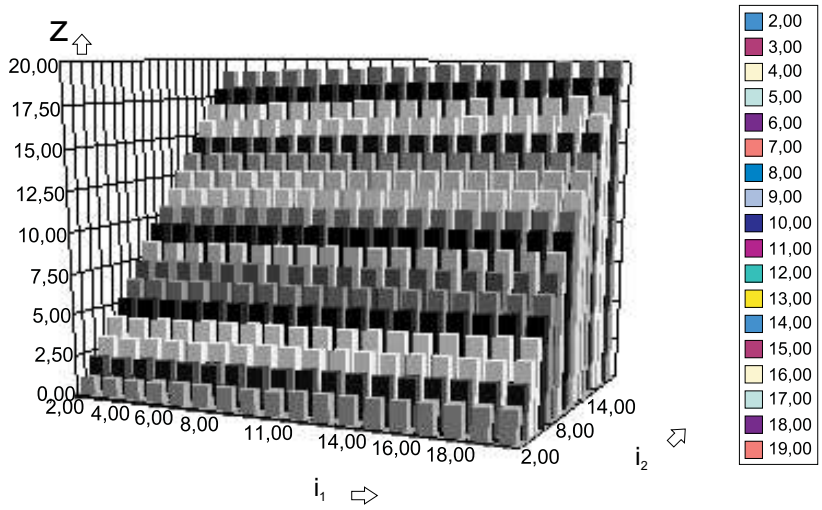
Wartości funkcji aktywacji $g_q(z)$ wyrażone wzorem (4.12) dla przykładu z płaszczyzną wynoszą

$$g_q(i_1, i_2) = \frac{1}{5}(\varsigma^q_{18,i_1} + \varsigma^q_{18,i_2}) \tag{4.36}$$

czyli $g_q(i_1, i_2) = \frac{1}{5}(\frac{1}{18}[i_1 - 1 - \frac{q}{6}] + \frac{1}{18}[i_2 - 1 - \frac{q}{6}])$

Po uproszczeniu otrzymuje się

$$g_q(i_1, i_2) = \frac{1}{90}(i_1 + i_2 - 2 - \frac{q}{3}) \tag{4.37}$$



Rysunek 4.3: Wartość funkcji z w zależności od i_1 i i_2 dla $q = 0$

Wartości pierwszej zewnętrznej funkcji aktywacji g_1 w zależności od i_1 i i_2 dla $q = 0$ przedstawiono na rys. 4.4.

Eliminując i_1 z równania (4.37) i wprowadzając z otrzymamy

$$g_q(z, i_2) = \frac{1}{90}(18z - 18i_2 + 3q + 18) \tag{4.38}$$

lub sporządzając zależność i_2 i i_1 od z z równania (4.35) otrzymamy równania

$$i_2 = 1 + [\frac{18z + \frac{10}{3}q}{19}] \tag{4.39}$$

$$i_1 = 1 + 18z + \frac{10}{3}q - 19[\frac{18z + \frac{10}{3}q}{19}] \tag{4.40}$$

Po wstawieniu równań (4.39) i (4.40) do równania (4.37) otrzymamy

$$g_q(z) = \frac{1}{90}(18z + 3q - 18[\frac{18z + \frac{10}{3}q}{19}] - 2) \tag{4.41}$$

Równanie (4.38) wyjaśnia dlaczego mamy 18 cykli dla $i_2 = 2, 3, ..., 19$.

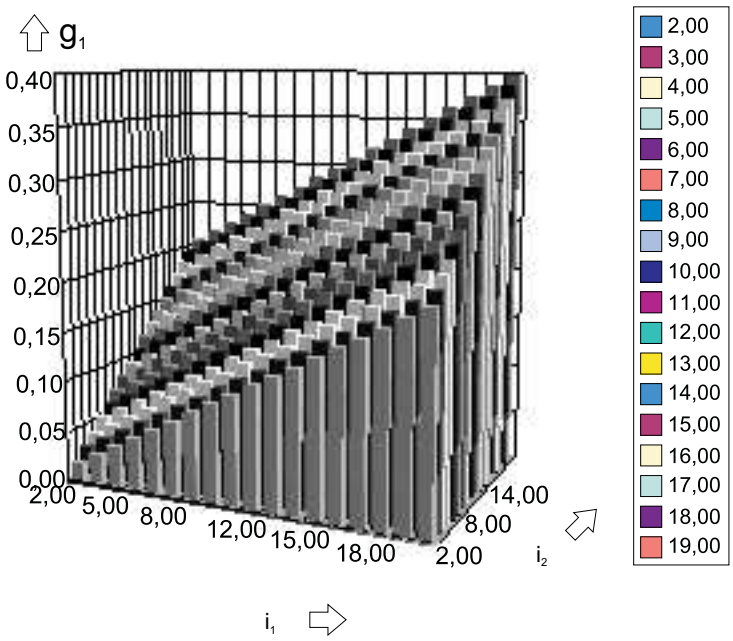
Fragment pierwszej zewnętrznej funkcji aktywacji g_1 w zależności od $z = 1$ do 10 przedstawiono na rysunku 4.5. Przedstawiona funkcja jest funkcją jednej zmiennej z (funkcja aktywacji musi być funkcją jednej zmiennej, tak jak w pracy Kołmogorowa [1] i w sieci neuronowej). Pozostałe funkcje aktywacji różnią się przesunięciem na rysunku (4.6) względem osi z o wartość $\frac{q}{270}$ (por. wzór (4.37)).

Wartość funkcji aproksymującej h_1 wynosi (por. wzór (4.37) i (4.6))

$$h_1 = \frac{1}{90}(5i_1 + 5i_2 - \frac{1}{3} - \frac{2}{3} - 1 - \frac{4}{3} - \frac{5}{3} - 10) \tag{4.42}$$

czyli

$$h_1 = \frac{1}{90}(5i_1 + 5i_2 - 10 - \frac{15}{3}) \tag{4.43}$$



Rysunek 4.4: Zależność pierwszej zewnętrznej funkcji aktywacji g_1 od i_1 i i_2.

a po uproszczeniu

$$h_1 = \frac{1}{18}(i_1 + i_2 - 3) \tag{4.44}$$

Ponieważ wartość dokładna funkcji aproksymowanej $h(x_1, x_2)$ wyrażonej jako funkcja i_1 i i_2 wynosi

$$h = \frac{1}{18}(i_1 + i_2 - 2) \tag{4.45}$$

więc błąd wynosi $\frac{1}{18}$.

Wartości dokładne funkcji aproksymowanej h w funkcji $i1$ i i_2 przedstawia rysunek 4.6, a wartości h_1 aproksymujące badaną funkcję h przedstawia rysunek 4.7.

Błąd względny aproksymacji w procentach wynosi:

$$\delta = \frac{h - h_1}{h} = \frac{1}{i_1 + i_2 - 2} \tag{4.46}$$

a wykres błędu przedstawiono na rysunku 4.8.

4.2.4 Przykład sieci liniowej dla funkcji kwadratowej

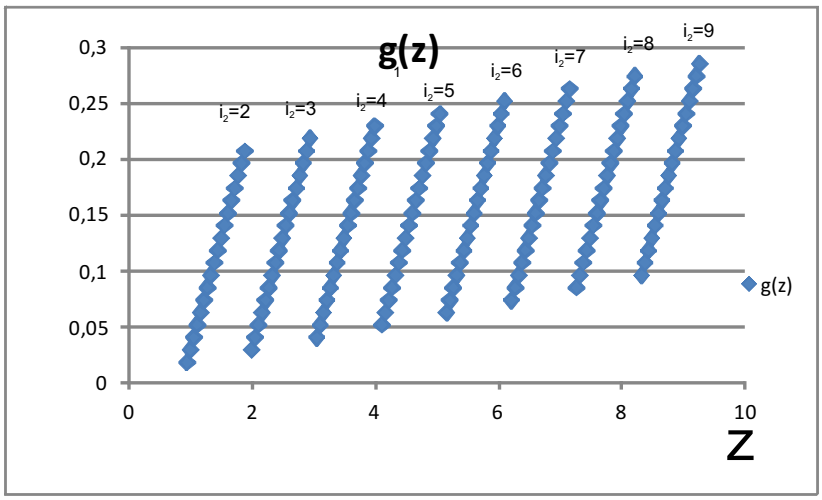
Rozpatrzmy jak przebiega proces uczenia sieci liniowej dla funkcji

$$h(x_1, x_2) = x_1^2 + x_2^2 + 10 \tag{4.47}$$

która przedstawia paraboloidę przechodzącą przez punkt $[0, 0, 10]$.

Należy wyznaczyć 5 zewnętrznych funkcji aktywacji g_1, g_2, g_3, g_4, g_5 .

Funkcje wyznaczmy ze wzoru $g_q(z) = \frac{1}{5}h(\varsigma^q_{18,i_1}, \varsigma^q_{18,i_2})$ zgodnie ze zmodyfikowanym twierdzeniem Kołmogorowa.



Rysunek 4.5: Fragment zewnętrznej funkcji aktywacji g_1 dla z=1 do z=10.

Przedziały liczb rzeczywistych $A^q_{18,i}$, zgodnie ze wzorem (4.4), wynoszą $A^q_{18,i} = \frac{1}{18}[i - 1 - \frac{q}{6}, i - \frac{1}{6} - \frac{q}{6}]$.

Punkty $\varsigma^q_{18,i}$ należą do $A^q_{18,i}$, i są ich lewymi końcami.

Zmienna i przyjmuje wartości od 2 do 19 (są to punkty przedziału E od 0 do 1 por. wzór (4.4)), a zmienna K (określająca dokładność aproksymacji) niech przyjmie wartość 18 (por. wzór (4.4)).

Wagi synaptyczne muszą spełniać warunek: $w_{q,p} \geq 19w_{q,p-1}$(por. zmodyfikowane twierdzenie Kołmogorowa) stąd dla $w_{q,1} = 1$ otrzymuje się $w_{q,2} = 19$.

Aby wyznaczyć wartości z (por. wzór 4.8) korzystamy ze wzoru

$$z = \sum_{p=1}^{2} w_{q,p}x_p = x_1 + 19x_2 \tag{4.48}$$

Wartości funkcji aktywacji dla bieżącego przykładu z paraboloidą wynoszą:

$$g_q(i_1, i_2) = \frac{1}{5}((\varsigma^q_{18,i_1})^2 + (\varsigma^q_{18,i_2})^2 + 10) \tag{4.49}$$

czyli mamy

$$g_q(i_1, i_2) = \frac{1}{5}((\varsigma^q_{18,i_1})^2 + (\varsigma^q_{18,i_2})^2) + 2 \tag{4.50}$$

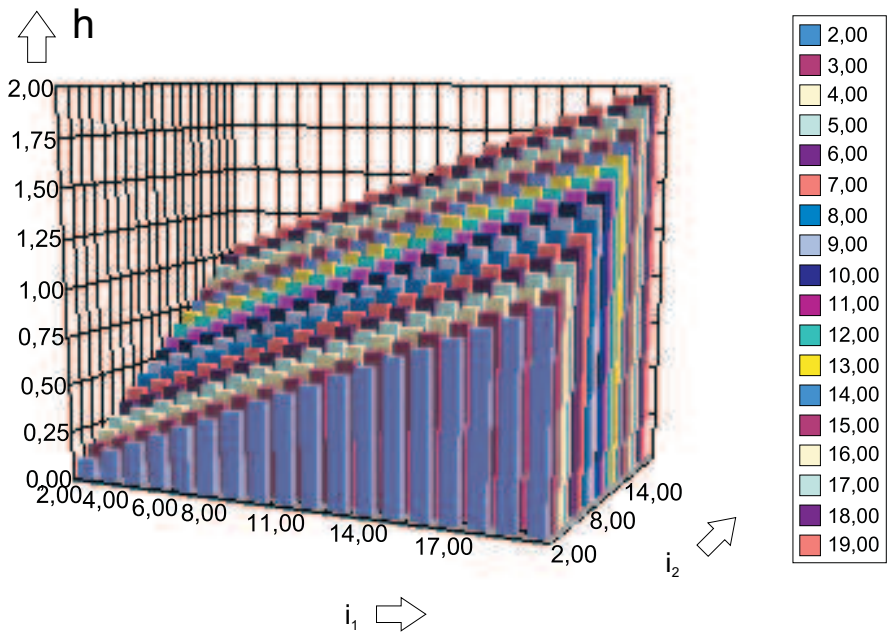
po uwzględnieniu wzoru (4.4) mamy

$$g_q(i_1, i_2) = \frac{1}{5}(\frac{1}{324}[i_1 - 1 - \frac{q}{6}]^2 + \frac{1}{324}[i_2 - 1 - \frac{q}{6}]^2) + 2 \tag{4.51}$$

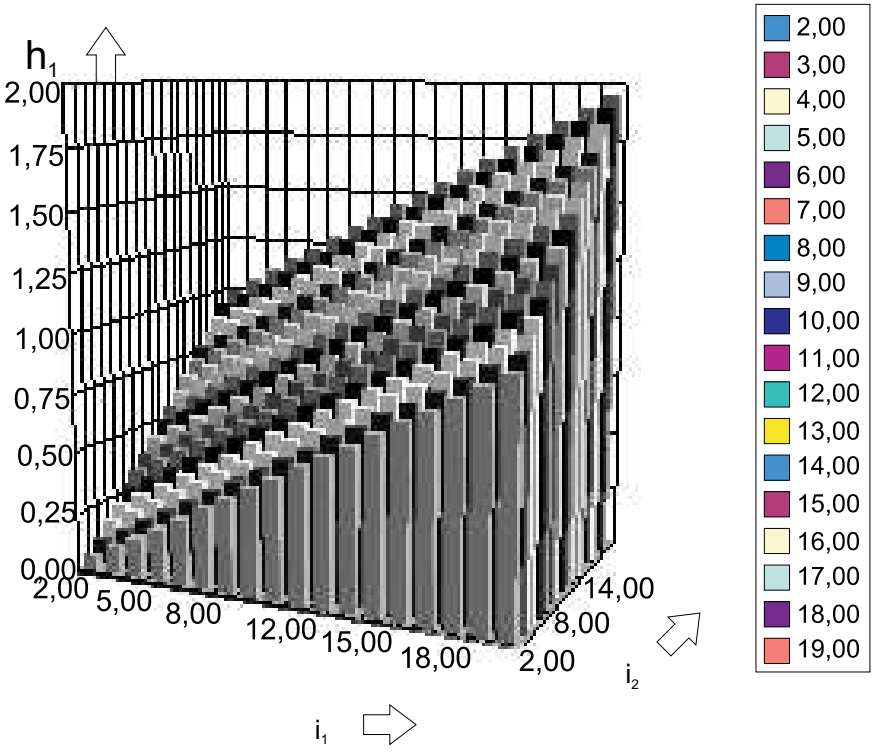
i po uproszczeniu

$$g_q(i_1, i_2) = \frac{1}{1620}[(i_1 - 1 - \frac{q}{6})^2 + (i_2 - 1 - \frac{q}{6})^2] + 2 \tag{4.52}$$

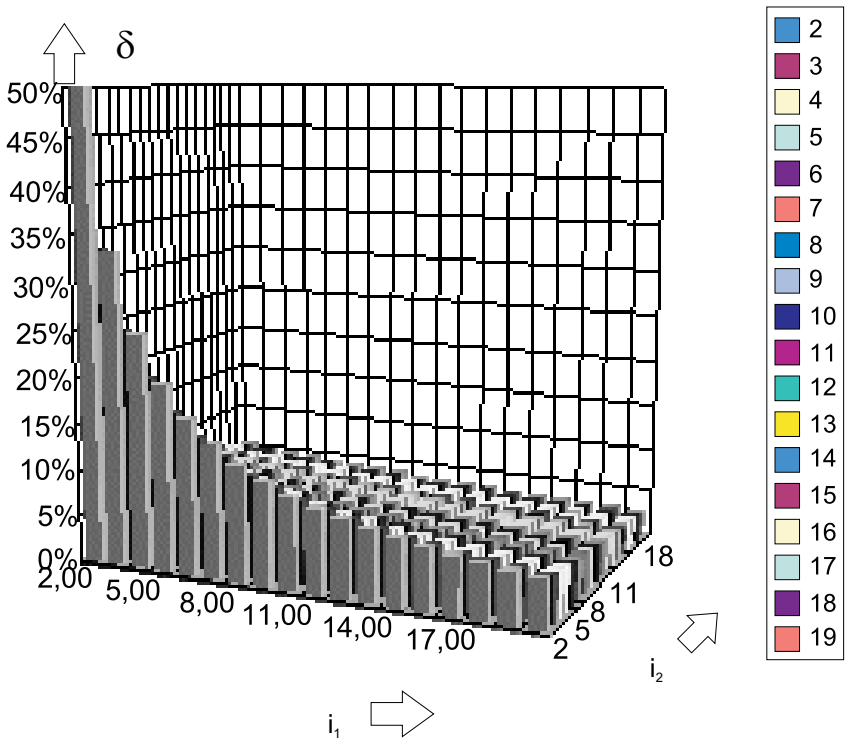
Wartości funkcji aktywacji g_1 (pierwszej z pięciu) przedstawiono na rysunku 4.9. Pozostałe funkcje aktywacje są przesunięte względem osi x o stałą wartość dla danych i_1 i i_2 (por. wzór (4.52)).



Rysunek 4.6: Wartości dokładne funkcji aproksymowanej



Rysunek 4.7: Wartości funkcji aproksymujacej h_1 w zależności od i_1 i i_2.



Rysunek 4.8: Błąd aproksymacji δ w funkcji i_1 i i_2

Wartość funkcji h_1 aproksymujacej paraboloidę wynosi

$$\begin{aligned} h_1 &= \frac{1}{1620}[(i_1-\frac{7}{6})^2+(i_1-\frac{8}{6})^2+(i_1-\frac{9}{6})^2+(i_1-\frac{10}{6})^2+(i_1-\frac{11}{6})^2 \\ &+(i_2-\frac{7}{6})^2+(i_2-\frac{8}{6})^2+(i_2-\frac{9}{6})^2+(i_2-\frac{10}{6})^2+(i_2-\frac{11}{6})^2]+10 \end{aligned} \tag{4.53}$$

Po przekształceniu i uporządkowaniu

$$\begin{aligned} h_1 &= \frac{1}{1620}[5i_1^2-i_1(\frac{7}{3}+\frac{8}{3}+\frac{9}{3}+\frac{10}{3}+\frac{11}{3})+\frac{49+64+81+100+121}{36} \\ &+5i_2^2-i_2(\frac{7}{3}+\frac{8}{3}+\frac{9}{3}+\frac{10}{3}+\frac{11}{3})+\frac{49+64+81+100+121}{36}]+10 \end{aligned} \tag{4.54}$$

co daje

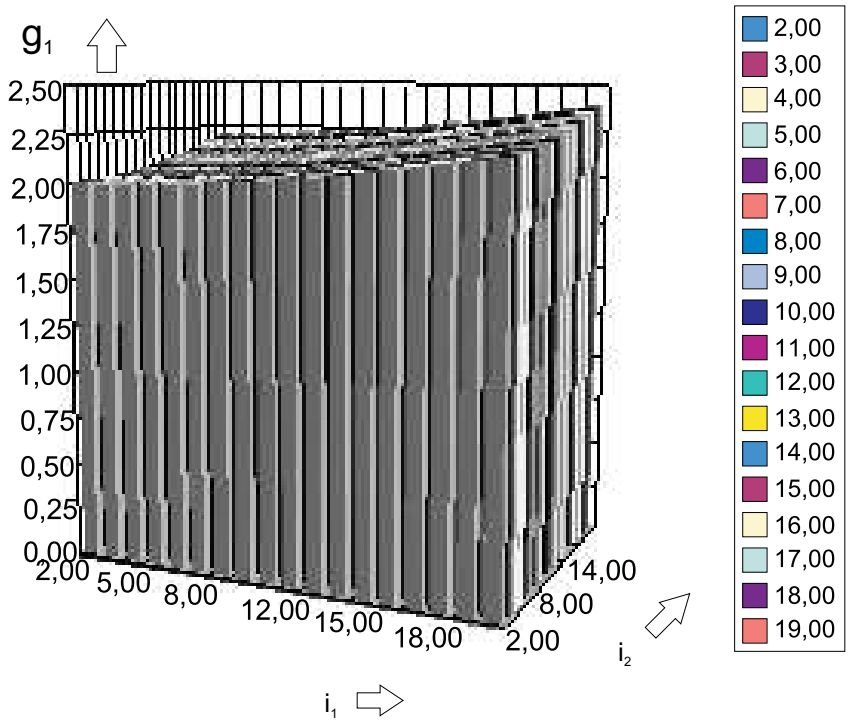
$$h_1=\frac{1}{1620}[5i_1^2+5i_2^2-15i_1-15i_2+\frac{415}{18}]+10 \tag{4.55}$$

a po skróceniu

$$h_1=\frac{1}{324}[i_1^2+i_2^2-3i_1-3i_2+\frac{83}{18}]+10 \tag{4.56}$$

Ponieważ wartość dokładna paraboloidy wynosi

$$h=\frac{1}{324}([i_1^2+i_2^2-2i_1-2i_2+2]+10 \tag{4.57}$$



Rysunek 4.9: Funkcja aktywacji g_1 w zależności od i_1 i i_2.

więc błąd wynosi

$$\delta = \frac{h - h_1}{h} = \frac{\frac{1}{324}(i_1 + i_2 - \frac{47}{18})}{\frac{1}{324}([i_1^2 + i_2^2 - 2i_1 - 2i_2 + 2] + 10} \tag{4.58}$$

Wartości aproksymujące paraboloidę, będące wynikiem uczenia przedstawia 4.10.

Wartości dokładne, czyli wykres paraboloidy w funkcji i_1 i i_2, przedstawia rys. 4.11.

Błąd aproksymacji w procentach przedstawia 4.12.

4.2.5 Przykład sieci potęgowej dla funkcji kwadratowej

Rozpatrzmy jak przebiega proces uczenia sieci potęgowej dla funkcji $h(x_1, x_2) = x_1^2 + x_2^2 + 10$ która przedstawia paraboloidę przechodzącą przez punkt $[0, 0, 10]$.

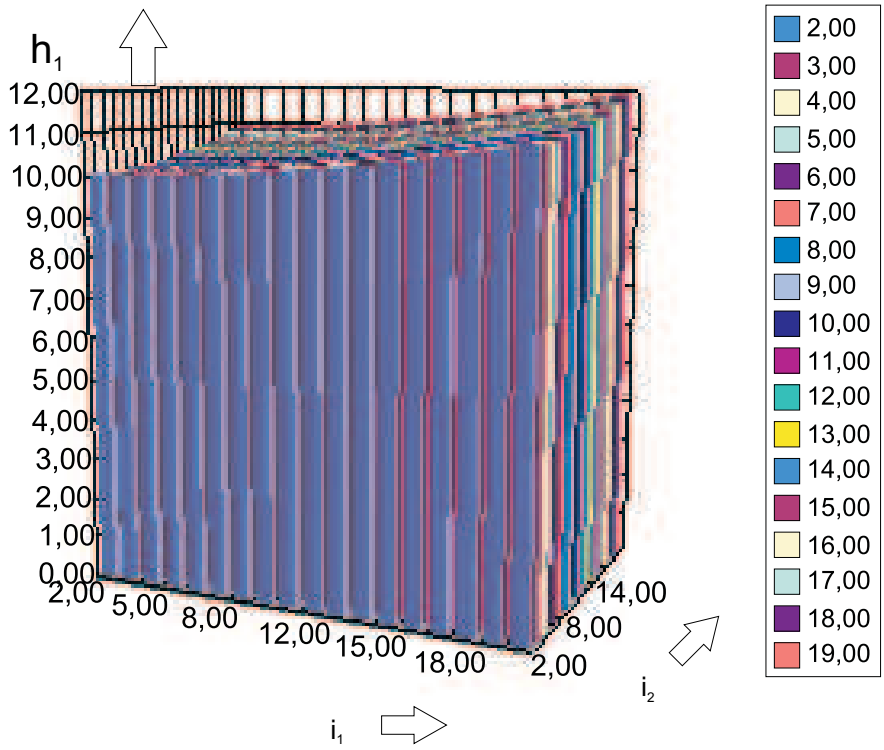
Należy wyznaczyć 5 zewnętrznych funkcji aktywacji g_1, g_2, g_3, g_4, g_5 .

Funkcje wyznaczmy ze wzoru $g_q(z) = \frac{1}{5}h(\varsigma^q_{18,i_1}, \varsigma^q_{18,i_2})$ zgodnie ze zmodyfikowanym twierdzeniem Kołmogorowa.

Przedziały liczb rzeczywistych $A^q_{18,i}$, zgodnie ze wzorem (4.4), wynoszą $A^q_{18,i} = \frac{1}{18}[i - 1 - \frac{q}{6}, i - \frac{1}{6} - \frac{q}{6}]$.

Punkty $\varsigma^q_{18,i}$ należą do $A^q_{18,i}$, i są ich lewymi końcami.

Zmienna i przyjmuje wartości od 2 do 19 (są to punkty przedziału E od 0 do 1 por. wzór (4.4)), a zmienna K (określająca dokładność aproksymacji) niech przyjmie wartość 18 (por. wzór (4.4)).



Rysunek 4.10: Wartości aproksymujące h_1 paraboloidę.

Aby wyznaczyć wartości z korzystamy z lewej strony wzoru (4.19) $z = f_{1,q}(x)+f_{2,q}(x) = x + x^2 = \frac{i_1-r}{\alpha} + (\frac{i_2-r}{\alpha})^2$

Przyjmujemy $r = \frac{1}{40}$ (p. komentarz pod wzorem (4.22)) i $\alpha = 20$ aby ułamek $\frac{i-r}{\alpha}$ był większy od 0 a mniejszy od 1.

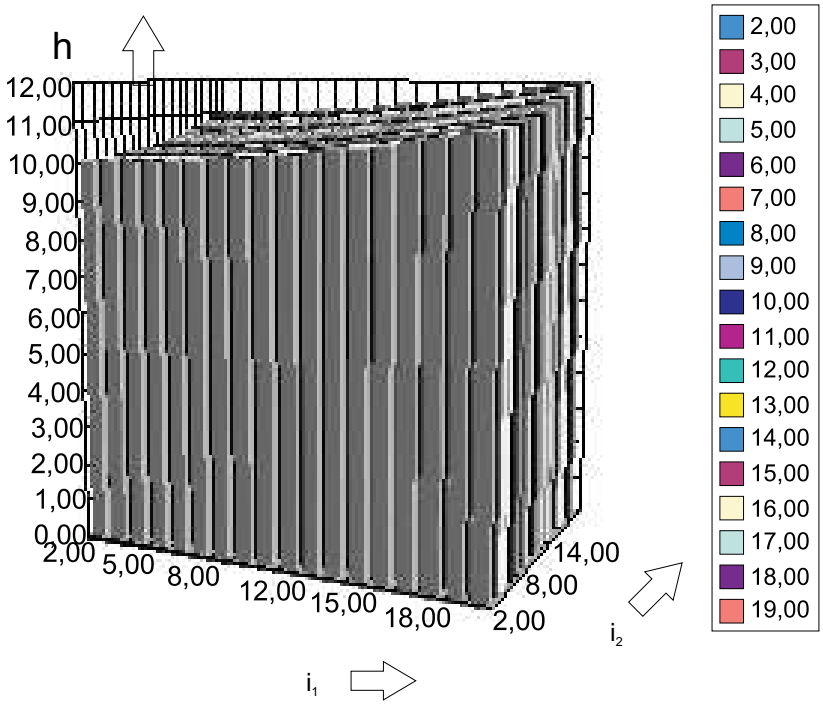
Ze względu na skomplikowane rachunki korzystamy z arkusza kalkulacyjnego aby wykreślić funkcje $g_q(z)$.

Pierwszą funkcje aktywacji przedstawiono na rys. 4.13 dla parametrów $i_1 = 2$ i $i_1 = 3$.

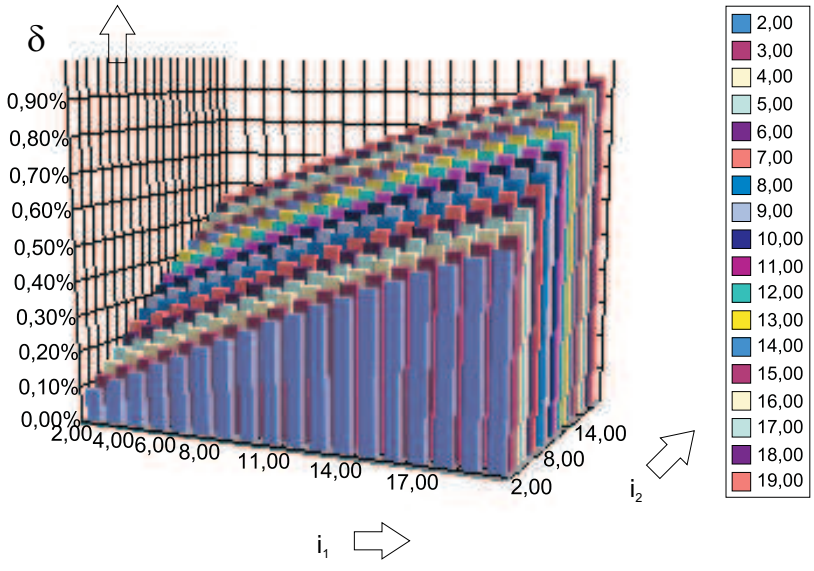
Porównując sieć potęgową i liniową należy stwierdzić mniejszy rozrzut wartości z w sieci potęgowej (od 0 do 1) w porównaniu z siecią liniową (0 do 20). Jest to skutkiem "efektu narastania wag" opisanego w rozdziale 4.2.1. Wyznaczane składniki z_q w sieci potęgowej będą miały zbliżoną dokładność w przeciwieństwie do sieci liniowej gdzie elementy z_q o mniejszej wartości będzie wyznaczana z mniejszą procentową dokładnością. Na przykład przy błędzie 0,1 błąd procentowy będzie zblizony dla z od 0 do 1 a przy z od 0 do 20 błąd procentowy będzie się bardziej różnił dla różnych z.

4.3 Nowe koncepcje zmodyfikowanych kołmogorowskich sieci neuronowych

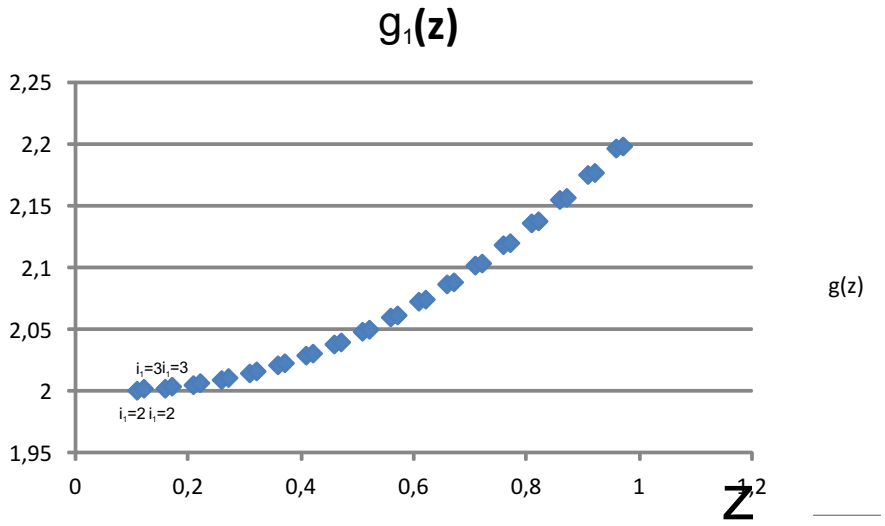
W oparciu o nowe koncepcje zmodyfikowanych kołmogorowskich sieci neuronowych zaproponowano sieć indeksowaną.



Rysunek 4.11: Paraboloida w funkcji i_1 i i_2



Rysunek 4.12: Wykres błędu aproksymacji δ



Rysunek 4.13: Wartość zewnętrznej funkcji aktywacji w zależności od z dla $q = 1$

4.3.1 Sieć indeksowana

W celu eliminacji problemu "narastania wag" (omówionego w podrozdziale 4.2.1) i maksymalnego uproszczenia obliczeń (najprostszy algorytm uczenia porównaj inne algorytmy) wprowadzono sieć indeksowaną. Proponowana przez autora zmodyfikowana indeksowana sieć kołmogorowska przedstawiona jest na rysunku 4.14 . Sieć składa się z bloku transformacji, węzłów funkcji zewnętrznych g_i i końcowego sumatora. Blok tranformacji zamienia wielkości wejściowe x_p na indeksy $i^q_{p,r}$ (stąd nazwa), a następnie oblicza indeksy całkowite j^r_q, które są argumentami zewnętrznych funkcji aktywacji g_q. Blok transformacji zastępuje wagi. Transformacja pozwala na redukcje wymiaru wielkości wejściowych z n na 1. W trakcie procesu uczenia sieci wyznacza się zewnętrzne funkcje aktywacji g_q podobnie jak w sieci liniowej.

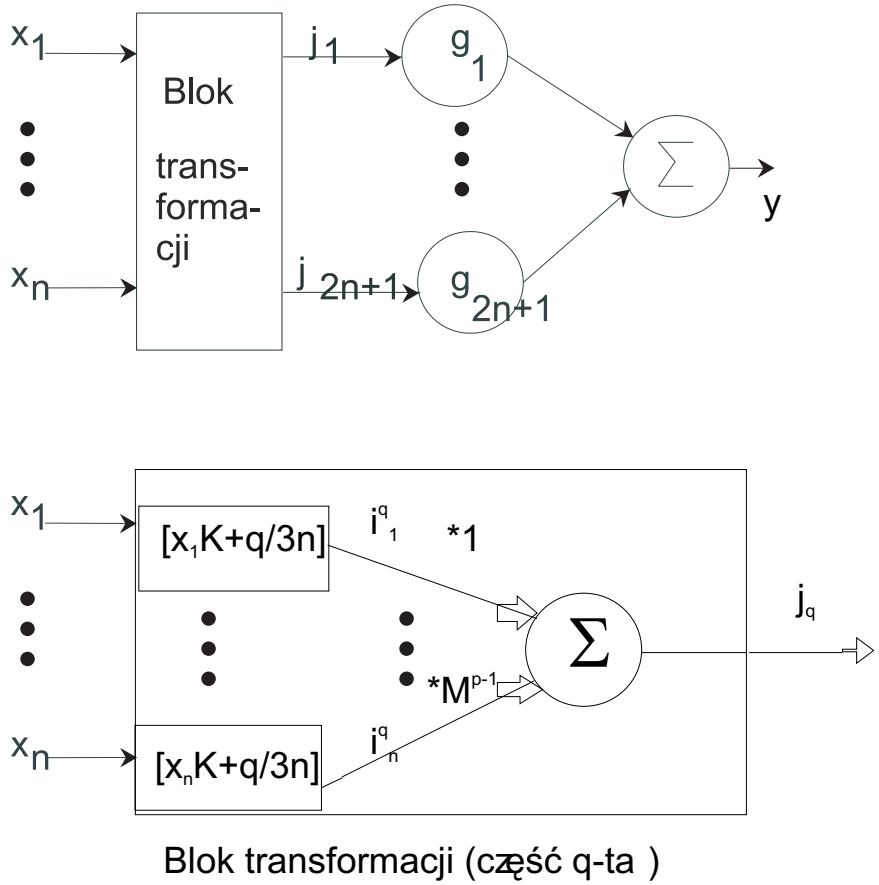
Idea transformacji opiera się na zamianie x_p na $i^q_{p,r}$ zgodnie z nierównością $\frac{1}{K}(i^q_{p,r} - \frac{q}{3n}) \leq x_p < \frac{1}{K}(i^q_{p,r} + 1 - \frac{q}{3n})$. Zapisując równoważnie mamy $x_pK + \frac{q}{3n} - 1 < i^q_{p,r} \leq x_pK + \frac{q}{3n}$. Zapisując równoważnie równaniem mamy

$$i^q_{p,r} = [x_pK + \frac{q}{3n}] \tag{4.59}$$

a następnie obliczeniu indeksu całkowitego (który jest argumentem zewnętrznej funkcji aktywacji) zgodnie ze wzorem

$$j^r_q = \sum_{p=1}^{n} i^q_{p,r} M^{p-1} \tag{4.60}$$

Symbol [x] oznacza cechę liczby x (czyli część całkowitą liczby x). Stała M wynosi $M = K + 1$. Postać równości na indeks cząstkowy wynika z podziału przedziału $[0, 1]$ na K części (liczbę z przdziału [0,1] należy transformować na zbiór liczb całkowitych od 0 do K czyli należy pomnożyć przez K). Na przykład przy podziale na 10 części punkt $\frac{1}{10}$ przejdzie w liczbę 1, punkt $\frac{2}{10}$ przejdzie w liczbę 2, itd. Składnik $\frac{q}{3n}$ wprowadzono aby dla zmiennego q kolejne zewnętrzne funkcje aktywacji g^r_q miały argumenty przesunięte o wartość tego składnika i przybliżały wartość aproksymowaną na różnych przedziałach (p. nierówność zamiany wielkości wejsciowej na indeks cząstkowy) i przez co zwiększały



Rysunek 4.14: Zmodyfikowana indeksowana kołmogorowska sieć neuronowa.

dokładność aproksymacji. Wartość mianownika wynosi $3n$ bo licznik q zmienia się od 1 do $2n+1$, a ułamek nie może przekroczyć wartości 1. Indeksy tych samych punktów a różnych zewnętrznych funkcji aktywacji muszą być zblizone do siebie. Na przykład dla K=7, gdy $q = 1$ ułamek wyniesie $\frac{q}{3n} = \frac{1}{6}$ i punkt $x_p = 0,4$ zostaje zamieniony na indeks cząstkowy $i = 2$. Punkt $x_p = 0,4$ dla $q = 2$ zostaje zamieniony na indeks $i = 3$. Ponieważ miejsca aproksymacji ζ zależą od i zewnętrzne funkcje aktywacji g są określane według wartości funkcji h w innych punktach. Wzór na indeks całkowity zapewnia jednoznaczne "ponumerowanie" dziedziny zewnętrznych funkcji aktywacji (p. rys 4.15 przy M=5 i M=7). Wzór na indeks całkowity jest wzorem zamiany p liczb w systemie o podstawie M na liczbę dziesiętną, przy czym odpowiednikami cyfr są indeksy cząstkowe a liczba zapisana w tym systemie jest odpowiednikeim indeksu całkowitego. W ten sposób w jednej liczbie będącej indeksem całkowitym zakodowane są dane o wszystkich liczbach będących indeksami cząstkowymi np liczba 1242 przy M=10 powstała z indeksów czastkowych 1,2,4,2.

Przykład wyznaczania indeksu cząstkowego i indeksu całkowitego.

Na przykład punkt $(0,5; 0,5)$, przy $K = 18$, $n = 2$ i $r = 1$ ma indeks cząstkowy 9 dla pierwszej zmiennej $p = 1$ i $q = 1$ bo równość zamiany wielkości wejściowej na indeks ma postać $i^1_{p,1} = [18 * 0,5 + \frac{1}{6}] = [9,17] = 9$. Dla drugiej zmiennej $p = 2$ i $q = 1$ indeks cząstkowy wynosi 9 (identyczna równość zamiany wielkości wejściowej na indeks). Stała $M = K + 1$ wynosi 19 bo indeksy cząstkowe i zmieniają się od 0 (dla $x_p = 0$) do 18 (dla $x_p = 1$). Stąd indeks całkowity (p. wzór (4.60)) $j = 9 * 19 + 9 = 180$.

Obecnie przedstawione jest zmodyfikowane wielokrokowe indeksowane twierdzenie Kołmogorowa dotyczące wielokrokowego iteracyjnego wyznaczania różnych i identycznych

zewnętrznych funkcji aktywacji g. Identyczne funkcje aktywacji wyznacza się gdy najważniejsza jest mała złożoność obliczeniowa algorytmu (wyznaczana jest tylko jedna funkcja zamiast kilku czyli przy 5 funkcjach aktywacji nakład obliczeń maleje 5 razy), a drugorzędna jest dokładność aproksymacji. Różne funkcje aktywacji wyznacza się przy wymaganiu dużej dokładności aproksymacji kosztem złożoności algorytmu (wyznaczanych jest kilka funkcji aktywacji).

Twierdzenie 2 (Zmodyfikowane wielokrokowe indeksowane twierdzenie Kołmogorowa)

Niech E i D_K, D_K^n będą zbiorami liczb rzeczywistych na wejściu sieci zdefiniowanymi w rozdziale 4.2 (E jest zbiorem spójnym a D_K zbiorem punktów). Dowolną funkcję $h(x_1, ..., x_n)$ ciągłą i rzeczywistą n zmiennych na $E \times E \times ... \times E(n - razy)$ można, z dowolnie małym błędem, aproksymować wzorem

$$h_r(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q^r(j_q^r) \tag{4.61}$$

gdzie g_q^r są są zewnętrznymi funkcjami aktywacji sieci neuronowej, wskaźnik r jest numerem iteracji, a indeksy j_q^r wyrażają się wzorem (4.60), gdzie $i_{p,r}^q$ jest indeksem cząstkowym zmiennym od 0 do K.

Postać wzoru (4.60) wzorowana jest na systemie pozycyjnym, gdzie cyfry zmieniają się od 0 do 9 a podstwa systemu pozycyjnego wynosi M=10 i dowolną liczbę można zapisać wzorem (4.60)).

Stała M wynosi

$$M = K + 1 \tag{4.62}$$

D'_K jest zbiorem liczb rzeczywistych takim, że: $D'_K = \left\{ \frac{1}{K}(i_{p,r}^q + 1 - \frac{q}{3n}) : i_{p,r}^q = 0, 1, ..., K \right\}$.

Zbiór D'_K różni się od D_K składnikiem $\frac{q}{3n}$, tak aby kolejne zewnętrzne funkcje aktywacji aproksymowały w innych punktach, gdyż wtedy zmniejsza się błąd aproksymacji.

Indeks cząstkowy $i_{p,r}^q$ wyraża się wzorem (4.59).

Pomysłem autora jest zamiana wielkości x_p na indeksy cząstkowe $i_{p,r}^q$ i dalej na indeksy całkowite j_q^r. Argumentem zewnętrznych funkcji aktywacji jest wtedy liczba całkowita.

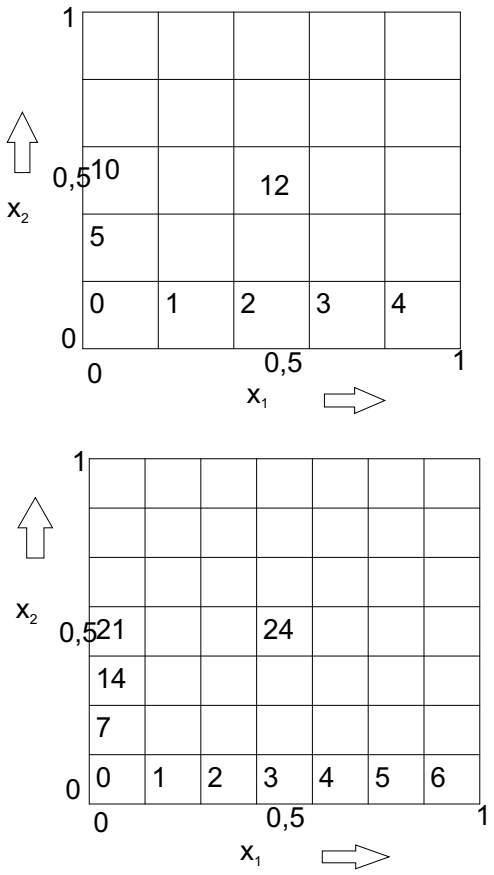
Wartości p i q zmieniają się w zakresie $1 \leq p \leq n$, $1 \leq q \leq 2n + 1$.

Dowód:

Punktowi $(x_1, ..., x_n)$, (gdzie x_p oznacza p tą zmienną) w kolejnych iteracjach odpowiadają indeksy j_q^{r-1} i j_q^r. Wynika to z tego, że dziedzina funkcji jest podzielona na różną liczbę części p. rys. 4.15. Zwiększamy liczbę części kwadratu dziedziny od 1 do ∞, aby zmniejszyć błąd aproksymacji w kolejnych iteracjach. Wynika to z tego, że zmienność ciągłej funkcji aproksymowanej zmniejsza się wraz z zawężaniem dziedziny.

Dla przykładu: wielkości wejściowej x_1, x_2 (0.5, 0.5) przy podziale dziedziny (to jest kwadratu o wierzchołkach (0,0),(0,1),(1,0),(1,1)) na 25 części (por. rysunek 4.15) odpowiada indeks j_q^{r-1} równy 12 (indeks cząstkowy $i_{1,r-1}^1$ równy 2 według wzoru (4.64) i indeks cząstkowy $i_{2,r-1}^1$ równy 2 według wzoru (4.64); stąd indeks całkowity równy jest 2+5*2 według wzoru (4.62)). Przy podziale na 49 części indeks całkowity j_q^r wyniesie 24 (indeks cząstkowy $i_{1,r}^1$ równy 2 i indeks cząstkowy $i_{2,r}^1$ równy 2, stąd indeks całkowity równy 2+5*2).

Podział dziedziny funkcji h na różną ilość części przedstawia rys. 4.15. Górny rysunek przedstawia strukturę dziedziny, gdy $M = 5$. Indeks cząstkowy przyjmuje wartości od



Rysunek 4.15: Podział dziedziny funkcji na 25 i 49 części.

0 do 4. Indeks całkowity obliczany jest ze wzoru (4.68) ze stałą $M = 5$. Dolny rysunek przedstawia strukturę dziedziny, gdy $M = 7$. Indeks cząstkowy przyjmuje wartości od 0 do 6. Indeks całkowity obliczany jest ze wzoru (4.60) ze stałą $M = 7$. Postać numeracji dolnego i górnego rysunku jest konsekwencją decyzji o przyjęciu wartości M.

Funkcja aproksymująca h_r jest równa sumie zewnętrznych funkcji aktywacji:

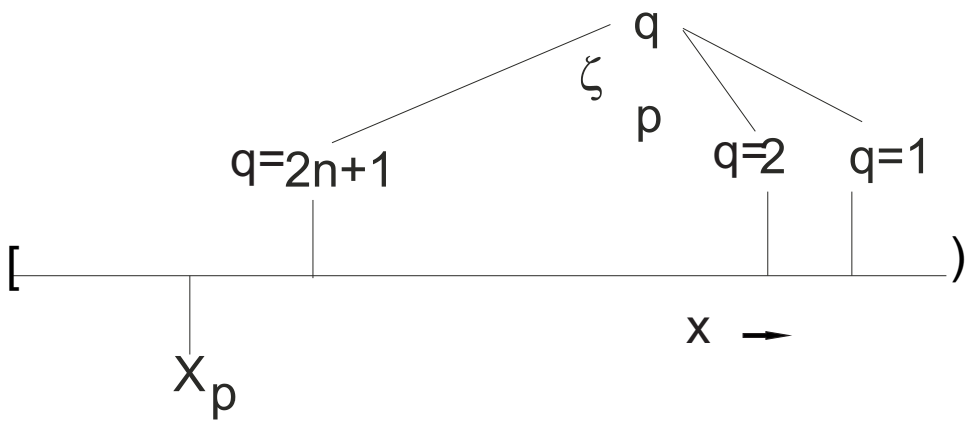
$$h_r(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q^r(j_q^r) = \sum_{q=1}^{2n+1} g_q^r(\sum_{p=1}^{n} i_{p,r}^q M^{p-1}) \tag{4.63}$$

Punkty $\varsigma_{p,K}^q$ (miejsca aproksymacji) należą do zbioru D'_K. Punkty $\varsigma_{p,K}^q$ zmieniają się, gdy zmienia się indeks cząstkowy $i_{p,r}^q$ dla wszystkich zmiennych wejściowych i tworzą "szkielet" przestrzeni E^n, na którym wyznaczana jest wartość aproksymowanej funkcji h. Punkty $\varsigma_{p,K}^q$ wyrażają się wzorem

$$\varsigma_{p,K}^q = \frac{1}{K}(i_{p,r}^q + 1 - \frac{q}{3n}) \tag{4.64}$$

Autor proponuje (taka formuła zapewnia zbieżność zewnętrznej funkcji aktywacji $g_q^r(j_q^r)$ do g_q por. [1]), aby w kolejnych krokach postępowania indukcyjnego przeliczać zewnętrzną funkcję aktywacji według wzoru

$$g_q^r(j_q^r) = g_q^{r-1}(j_q^{r-1}) + \frac{1}{2n+1}(h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q) - h_{r-1}(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q)) \tag{4.65}$$



Rysunek 4.16: Miejsca aproksymacji zewnętrznych funkcji aktywacji.

gdzie $j = 1, 2, ..., M^{n-1}$

Wzór (4.65) jest zmodyfikowanym wzorem z dowodu twierdzenia Kołmogorowa [1]. Intencją Kołmogorowa [1] było, aby w kolejnych krokach przybliżać zewnętrzne funkcje aktywacji sumą poprzednich zewnętrznych funkcji aktywacji i błędem aproksymacji funkcji $h - h_{r-1}$.

Przykładowe położenia punktów $\varsigma^q_{p,k}$ i x_p na przedziale $(\frac{1}{K}i^q_{p,r}, \frac{1}{K}i^q_{p,r} + 1]$ przy zmiennym q przedstawia rysunek 4.16 (por. wzór 4.74). W widocznym na rysunku przedziale (lewostronnie otwartym i prawostronnie domkniętym) wartość x_p jest zamieniana na wartość indeksu cząstkowego $i^q_{p,r}$ zgodnie ze wzorem (4.67), natomiast wartość funkcji h i h_r jest wyznaczana w punktach $\varsigma^q_{p,K}$ odpowiadających temu indeksowi.

Dalej przyjmujemy $g^0_q \equiv 0$ (bo najczęściej przyjmuje się zerowe warunki początkowe).

Podobnie jak w oryginalnym twierdzeniu Kołmogorowa [1] różnicę między funkcją h i jej aproksymatą można wyrazić z użyciem zewnętrznych funkcji aktywcji

$$\begin{aligned} h(x_1, ..., x_n) - h_r(x_1, ..., x_n) &= h(x_1, .., x_n) - h_{r-1}(x_1, ..., x_n) - \\ -(h_r(x_1, ..., x_n) - h_{r-1}(x_1, ..., x_n)) &= h(x_1, .., x_n) - h_{r-1}(x_1, ..., x_n) - \\ &\quad - \sum_{q=1}^{2n+1} (g^r_q(j^r_q) - g^{r-1}_q(j^{r-1}_q)) \end{aligned} \tag{4.66}$$

Z kolei dla składników sumy z wyrażenia (4.66) zachodzi nierówność otrzymana z równości (4.65)

$$g^r_q(j^r_q) - g^{r-1}_q(j^{r-1}_q) \leq \frac{1}{2n+1}(h(x_1, .., x_n) - h_{r-1}(x_1, ..., x_n)) + \frac{1}{2n+1}\frac{1}{n+1}\sup_{E^n}|h - h_{r-1}| \tag{4.67}$$

bo tak można dobrać K ($K > 1$) we wzorze (4.64) aby zbiór D'_K był tak gęsty i różnica różnicy funkcji i jej aproksymaty w punktach dowolnych zbioru E i zbioru D'_K była mniejsza od prawej strony nierówności (4.68) czyli aby

$$\begin{aligned} &\left|h(x_1, .., x_n) - h_{r-1}(x_1, ..., x_n) - (h(\varsigma^q_{1,K}, ..., \varsigma^q_{n,K}) - h_{r-1}(\varsigma^q_{1,K}, ..., \varsigma^q_{n,K}))\right| \\ = &\left|h(x_1, .., x_n) - h(\varsigma^q_{1,K}, ..., \varsigma^q_{n,K})\right| \leq \frac{1}{n+1}\sup_{E^n}|h - h_{r-1}| \end{aligned} \tag{4.68}$$

bowiem funkcja h_r jest stała na elementach "siatki przestrzennej" $D_k'^n$ (patrz rys. 4.15 z objaśnieniem); indeks j_r^q jest stały, bo wielkość rzeczywista jest zamieniana na wielkość całkowitą (stałą na elemencie "siatki przestrzennej"(p. równość (4.59) i (4.60)).

Stąd po przekształceniach wzoru (4.66)

$$\sup_{E^n} |h - h_r| \leq \frac{1}{n+1} \sup_{E^n} |h - h_{r-1}| \tag{4.69}$$

Oznacza to, że różnica między funkcją h i jej aproksymatą h_r w kolejnych krokach r i $r-1$ wynosi

$$N_r = \sup_{E^n} |h - h_r| \leq \frac{1}{n+1} N_{r-1} \tag{4.70}$$

Stąd wyprowadzając nierówność dla N_r i N_0 otrzymamy

$$N_r \leq \left(\frac{1}{n+1}\right)^r N_0 \tag{4.71}$$

Różnica miedzy funkcją h i jej aproksymatą h_r dąży do 0

$$N_r \to 0 \tag{4.72}$$

gdy $r \Rightarrow \infty$

czyli

$$h_r \to h \tag{4.73}$$

Pozostaje udowodnić, że ciąg g_q^r jest zbieżny. Zauważmy, że szereg

$$g_q^r - g_q^0 = \sum_r \frac{1}{2n+1} \left(\frac{1}{n+1}\right)^r N_0 \tag{4.74}$$

jest bezwzględnie zbieżny.

Z kryterium d'Alamberta wynika, że

$$\lim_{r\to\infty} \left|\frac{a_{r+1}}{a_r}\right| = \lim_{r\to\infty} \left|\frac{1}{n+1}\right| \leq 1 \tag{4.75}$$

Stąd wniosek, że funkcja aktywacji g_q^r dana wzorem (4.65) jest ciągiem Cauch'ego (w R).

Zatem zewnętrzne funkcje aktywacji g_q^r są zbieżne do granic

$$g_q^r \to g_q \tag{4.76}$$

co kończy dowód zmodyfikowanego indeksowanego twierdzenia Kołmogorowa.

♠

Algorytm uczenia sieci indeksowanej (wersja wielokrokowa) jest następujący:

1. ustal $r = 1$, $g_q^0 = 0$, $h_0 = 0$ oraz K
2. ustal $i_{1,r}^q$,...,$i_{n,r}^q$ i oblicz $M = K + 1$ $j_q^r = \sum_{p=1}^n i_{p,r}^q M^{p-1}$

3. wyznacz $\varsigma^q_{K,i_1} = \frac{1}{K}(i^q_{1,r} + 1 - \frac{q}{3n})$,..., $\varsigma^q_{K,i_n} = \frac{1}{K}(i^q_{n,r} + 1 - \frac{q}{3n})$

4. wyznacz $g^r_q(j^r_q) = g^{r-1}_q(j^{r-1}_q) + \frac{1}{2n+1}(h(\varsigma^q_{1,k}, ..., \varsigma^q_{n,k}) - h_{r-1}(\varsigma^q_{1,k}, ..., \varsigma^q_{n,k}))$ i $h_r = \sum_{q=1}^{2n+1} g^r_q$

5. zmieniaj $i^q_{1,r}$,...,$i^q_{n,r}$ aby wyczerpać wszystkie wartości i oblicz $j^r_q = \sum_{p=1}^{n} i^q_{p,r} M^{p-1}$, a po każdej zmianie przejdź do 3

6. zwiększ r o 1 i jeśli $r = r_{koniec}$ to koniec a jesli nie to ustal nowe K i przejdź do punktu 2

Zmienne $x_1, ...x_n$ nie są wczytywane przez algorytm wielkości wyjściowe są wczytywane dla zmiennych $\varsigma^q_{K,i_1} = \frac{1}{K}(i^q_{1,r} + 1 - \frac{q}{3n})$,..., $\varsigma^q_{K,i_n} = \frac{1}{K}(i^q_{n,r} + 1 - \frac{q}{3n})$.

Zmienne $i^q_{1,r}$,...,$i^q_{n,r}$ są zmiennymi pomocniczymi.

Pozostaje pytanie czy liczba neuronów musi być równa $2n+1$. Aproksymację z użyciem l neuronów obrazuje twierdzenie 3. Twierdzenie 3 ma postać algorytmu wyrażonego wzorem (z natychmiastowym wyznaczeniem zewnętrznych różnych lub identycznych funkcji aktywacji) w przeciwieństwie do twierdzenia 2, które ma postać wielokrokową (h_r zastąpiono przez h_a).

Twierdzenie 3

Dowolną funkcję $h(x_1, ..., x_n)$ ciągłą i rzeczywistą na $E \times E \times ... \times E(n - razy)$ można aproksymować wzorem

$$h_a(x_1, ..., x_n) = \sum_{q=1}^{l} g_q(j_q) \tag{4.77}$$

z dowolnie małym błędem,

gdzie g_q są rzeczywistymi funkcjami aktywacji wyrażonymi wzorem (4.83), a l jest liczbą neuronów w warstwie wyjsciowej,

indeks całkowity wynosi

$$j_q = \sum_{p=1}^{n} i^q_p M^{p-1} \tag{4.78}$$

a stała M jest równa

$$M = K + 1 \tag{4.79}$$

Indeks czastkowy i^q_p wyznacza się z równania (4.80)

$$i^q_p = [K x_p + \frac{q}{3n}] \tag{4.80}$$

dla $1 \leq p \leq n$, $1 \leq q \leq l$.

Uzasadnienie równań (4.78) i (4.80) podano we wprowadzeniu do twierdzenia 3.

Dowód:

Funkcja aproksymująca h_a jest równa

$$h_a(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} i^q_p M^{p-1}) \tag{4.81}$$

Punkty $\varsigma_{p,K}^q$ należą do zbioru D'_K (miejsca aproksymacji) to znaczy

$$\varsigma_{p,K}^q = \frac{1}{K}(i_{p,r}^q + 1 - \frac{q}{3n}) \tag{4.82}$$

Zewnętrzna funkcja aktywacji g_q jest równa (podobnie jak według wzoru (4.65))

$$g_q(j_q) = \frac{1}{l}h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q) \tag{4.83}$$

gdzie $j_q = 1, 2, ..., M^{n-1}$

Jak można zauważyć podobnie jak w oryginalnym twierdzeniu Kołmogorowa [1]

$$h(x_1, ..., x_n) - h_a(x_1, ..., x_n) = h(x_1, .., x_n) - \sum_{q=1}^{l} g_q(j_q) \tag{4.84}$$

Z kolei dla składników sumy zależności (4.84) zachodzi

$$g_q(j_q) = \frac{1}{l}h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q) \leq \frac{1}{l}h(x_1, .., x_n) + \delta \tag{4.85}$$

gdzie stałą δ można ograniczyć

$$|\delta| \leq N_0 \tag{4.86}$$

gdzie N_0 jest dowolnie małą dodatnią liczbą, bo tak można dobrać k aby zbiór D'_K był tak gęsty aby

$$\left|h(x_1, .., x_n) - h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q)\right| \leq N_0 \tag{4.87}$$

Stąd po przekształceniach

$$|h(x_1, ..., x_n) - h_a(x_1, ..., x_n)| \leq N_0 \tag{4.88}$$

czyli funkcja aproksymująca jest zbieżna do funkcji h

$$h_a \to h \tag{4.89}$$

co kończy dowód zmodyfikowanego indeksowanego twierdzenia Kołmogorowa w wersji jednokrokowej. Twierdzenie 3 mówi, że liczba neuronów w sieci może być między 1 a $2n + 1$.

♠

W szczególnym przypadku gdy $l = 1$ (sieć z 1 neuronem) funkcja aktywacji $g_q(j)$ jest równa $h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q)$

$$g(j) = h(\varsigma_{1,K}^q, ..., \varsigma_{n,K}^q) \tag{4.90}$$

i wtedy w punktach zbioru D'_K aproksymacja staje się interpolacją. Warto zwrócić uwagę, że w równaniu (4.90) następuje redukcja wymiaru z n na 1.

Czyli liczbę neuronów możemy dobierać od 1 do $2n + 1$. Większa liczba neuronów zapewnia mniejszy błąd aproksymacji zależny od zmienności aproksymowanej funkcji. I tak na przykład dla aproksymowanej funkcji wykładniczej dwóch zmiennych w postaci $y = exp(x_1 + x_2)$ określonej na przedziale $[0, 1]^2$ przy $K = 10$ błąd aproksymacji (p.

równanie 4.85) dla $l = 5$ wynosi $0,29$, dla $l = 4$ wynosi $0,36$, dla $l = 3$ wynosi $0,47$, dla $l = 2$ wynosi $0,70$, dla $l = 1$ wynosi $1,34$. Powyższe obliczenia wykonano na arkuszu kalkulacyjnym i ze względu na prostotę zostały pominięte.

Podany indeksowany algorytm wielokrokowy jest skomplikowany, ponieważ wymaga przeliczenia zewnętrznej funkcji aktywacji. W kolejnych krokach siatka punktów aproksymacyjnych zagęszcza się stopniowo tak, że w każdym kroku błąd aproksymacji maleje $n + 1$ razy (por. wzór (4.70)), co jest niekorzystne dla algorytmu.

Wad tych nie posiada indeksowany algorytm jednokrokowy, w którym siatka punktów aproksymacyjnych D'_K zagęszcza się tak, aby błąd aproksymacyjny był dowolnie mały (por. wzór (4.70)). Zaletą algorytmu jednokrokowego jest to, że znajduje minimum globalne w jednym kroku, a nie jak w sieciach sigmoidalnych, gdzie algorytm w wielu krokach "gubi" się w minimach lokalnych. Algorytmy aproksymacji i interpolacji znane w metodach numerycznych to najczęściej algorytmy jednokrokowe w których parametry i funkcje aproksymujące określa się wzorem [46].

Algorytm uczenia sieci indeksowanej w wersji jednokrokowej (funkcja g_q wyznaczana jest jednokrotnie) jest nastepujący:

1. ustal i_1^q,...,i_n^q, K i oblicz $j_q = \sum_{p=1}^{n} i_p^q M^{p-1}$
2. wyznacz $\varsigma_{K,i_1}^q = \frac{1}{K}(i_1^q + 1 - \frac{q}{3n})$,..., $\varsigma_{K,i_n}^q = \frac{1}{K}(i_n^q + 1 - \frac{q}{3n})$
3. wyznacz $g_q(j_q) = \frac{1}{2n+1} h(\varsigma_{1,k}^q, ..., \varsigma_{n,k}^q)$
4. zmieniaj i_1^q,...,i_n^q aby wyczerpać wszystkie wartości i oblicz $j_q = \sum_{p=1}^{n} i_p^q M^{p-1}$, a po każdej zmianie przejdź do 2

Zmienne $x_1, ...x_n$ nie są wczytywane przez algorytm wielkości wyjściowe są wczytywane dla zmiennych $\varsigma_{K,i_1}^q = \frac{1}{K}(i_1^q + 1 - \frac{q}{3n})$,..., $\varsigma_{K,i_n}^q = \frac{1}{K}(i_n^q + 1 - \frac{q}{3n})$.

Zmienne $i_{1,r}^q$,...,$i_{n,r}^q$ są wielkościami pomocniczymi.

Powyższy algorytm jest najprostszy z wszystkich prezentowanych. Przykład obliczeń powyższego algorytmu (dostosowanego do przykładu obliczeniowego) podano w rozdziale 5.2.2.

4.4 Błąd aproksymacji funkcji ciągłej

Błąd aproksymacji jest różnicą między funkcją aproksymowaną $h(x_1, ..., x_n)$ przez sieć neuronową a funkcją $h_a(x_1, ..., x_n)$ realizowaną przez sieć neuronową. Błąd aproksymacji w sieci liniowej i potęgowej zmodyfikowanego twierdzenia Kołmogorowa jest inaczej oceniany niż w indeksowanym zmodyfikowanym twierdzeniu Kołmogorowa. Błąd ten jest wyrażony identycznymi wzorami dla sieci liniowej, potęgowej i zależy liniowo od zmienności funkcji w rozważanej dziedzinie. Przez zmienność funkcji [28] rozumie się różnicę miedzy wartością największą a wartością najmniejszą funkcji na zadanej dziedzinie.

Błąd aproksymacji w sieciach liniowych i potęgowych opartych na zmodyfikowanym twierdzeniu Kołmogorowa ma dwa rodzaje składników:

1. składniki spowodowane przesunięciem względem siebie punktów "siatki przestrzennej" ς_{K,i_p}^q (p. wzór (4.4)) na których są określone zewnętrzne funkcje aktywacji g_q (p. wzór 4.21) przy $q = 1, 2, ..., 2N + 1$;

2. składniki wynikające z aproksymacji tylko w wybranych punktach i zmienności aproksymowanej funkcji względem tych punktów.

Wartości składników błędu pierwszego rodzaju wynoszą

$$|h(x_1, ..., x_n) - h_a(x_1, ..., x_n)| \leq \frac{1}{l} N_0 \tag{4.91}$$

gdzie

$$N_0 = \sup_{E^n} |h| \tag{4.92}$$

a $l > 1$ wskazuje ile razy zmniejszyła się zmienność funkcji h przez zawężenie dziedziny A^q_{K,i_p} i jest określone nierównością

$$\left|h(x_1, ..., x_n) - h(\varsigma^q_{K,i_1}, ..., \varsigma^q_{K,i_n})\right| \leq \frac{1}{l} N_0 \tag{4.93}$$

gdzie

$$\varsigma^q_{K,i_p} \in A^q_{K,i_p} \tag{4.94}$$

i są lewymi końcami A^q_{K,i_p}.

Dokładne uzasadnienie nierówności (4.93) jest podane w rozdziale 4.2.1.

Zwiększając K (p. definicja przedziałów A^q_{K,i_p} wzór 4.4) zmniejszamy długość przedziału A^q_{K,i_p} i tym samym otrzymamy mniejsze zmiany wartości funkcji (p. wzór (4.93)) stąd większe wartości l w nierówności (4.93).

Aby rozważyć błędy drugiego rodzaju zdefiniujmy przedziały liczb rzeczywistych $B_{K,i}$ bliskie węzłom siatki punktów aproksymacji ς^q_{K,i_p}. Rozważmy przedziały $B_{K,i}$ pokrywające przedział liczb rzeczywistych $[0, 1]$ przy zmiennym i od 0 do K(wtedy prawy koniec ostatniego przedziału będzie równy 1, p. wzór (4.95)).

Przedział $B_{K,i}$ jest określonym równaniem (4.95)

$$B_{k,i} = \frac{1}{K}(i - 1, i] \tag{4.95}$$

Składniki błędu drugiego rodzaju, wynikające ze zmienności aproksymowanej funkcji wynoszą

$$\left|h(x_1^{'}, ..., x_n^{'}) - h(x_1, ..., x_n)\right| \leq \frac{1}{l'} N_0 \tag{4.96}$$

gdzie $x_p^{'}, x_p \in B_{K,i}$

Punkt $x_p^{'}$ jest dowolnym punktem dziedziny E, a x_p jest elementem zbioru D_K . Ponieważ przedziały $B_{K,i}$ pokrywają całą dziedzinę, bo dla zmiennej wartości i prawy koniec przedziału (dla i) łączy się z lewym końcem kolejnego przedziału (dla $i + 1$)(p. wzór (4.95)), to zawsze można wybrać takie i , że $x_p^{'}, x_p \in B_{K,i}$. Zmienność funkcji h na przedziale $B_{K,i}$ będzie mniejsza niż zmienność funkcji h na całej dziedzinie. Wskaźnik $l^{'} > 1$ wskazuje o ile wahanie funkcji, czyli różnica między wartością największą a wartością najmniejszą, zmniejszyła się wskutek zawężenia dziedziny, bo wahanie funkcji maleje wraz z zawężaniem dziedziny.

Stąd całkowity błąd aproksymacji dla zmodyfikowanego nieindeksowanego twierdzenia Kołmogorowa jest sumą obu składników błędu i wynosi

$$\left|h(x_1^{'},...,x_n^{'}) - h_a(x_1,...,x_n)\right| \leq (\frac{1}{l} + \frac{1}{l'})N_0 \tag{4.97}$$

Błąd aproksymacji dla indeksowanego zmodyfikowanego twierdzenia Kołmogorowa wynosi (p. wzór 4.71)

$$N_r \leq \left(\frac{1}{n+1}\right)^r N_0 \tag{4.98}$$

gdzie r jest numerem iteracji, N_r błędem aproksymacji w r tej iteracji (p. rozdział 4.3.1).

Błąd aproksymacji ocenia się w iteracyjnej wersji indeksowanego twierdzenia Kołmogorowa (p. twierdzenie 3 w rozdziale 4.3.1)

W wersji opartej na twierdzeniu 4 z punktu 4.3.1 to jest w wersji 1 iteracyjnej zmodyfikowanego, indeksowanego twierdzenia Kołmogorowa obowiązuje wzór (określający błąd aproksymacji) podobny jak w zmodyfikowanym twierdzeniu Kołmogorowa w wersji r iteracyjnej. (p. rozdział 4.3.1)

$$N_1 \leq \frac{1}{l}N_0 \tag{4.99}$$

gdzie l jest liczbą określającą ile razy zmaleje wahanie aproksymowanej funkcji wskutek zawężenia dziedziny. Błąd aproksymacji jest równy $\frac{N_0}{l}$.

Zgodnie ze wzorem (4.98), im większe n, to jest liczba wymiarów funkcji aproksymowanej h, tym "lepsza" jest zbieżność.

4.5 Zestawienie właściwości kołmogorowskich sieci neuronowych

Podane wyżej zmodyfikowane twierdzenie Kołmogorowa umożliwia konstrukcję kołmogorowskich sieci neuronowych w sposób szybszy i prostszy niż stosowane dotychczas metody. Kołmogorowskie sieci neuronowe mają strukturę zdeterminowaną przez twierdzenie o teorii ciągłych funkcji matematycznych. Charakteryzują się one prostym algorytmem uczenia bez metod gradientowych i propagacji wstecznej. Zaletą podanych algorytmów jest prosta budowa i niski nakład obliczeń. W większości kołmogorowskich sieci neuronowych miejsce wag zstępują wewnętrzne funkcje aktywacji. Zasadniczą rolę pełnią w nich neurony z realizowaną zewnętrzną funkcją aktywacji. Budowę takiej sieci zaczyna się od całości struktury przechodząc do pojedynczych neuronów. W sieci sigmoidalnej proces ten przebiega odwrotnie.

W niniejszej pracy zaproponowano struktury trzech kołmogorowskich sieci neuronowych:

- sieć liniową;
- sieć potęgową;

- sieć indeksowaną;

Zalety i wady proponowanych sieci przedstawia tabela 4.1.

Nazwa sieci	Zalety	Wady
Liniowa	Prosta budowa i podobna do sieci sigmoidalnych	Niekorzystne "narastanie wag" przy dużej dokładności
Potęgowa	Funkcja potęgowa zastępują wagi co wyklucza ich narastanie	Metoda uczenia bardziej skomplikowana niż w sieci liniowej
Indeksowana	Prosty algorytm uczenia, z blokiem transformacji, który zastępuje wagi	Sieć o nietypowej architekturze

Tablica 4.1: Wady i zalety sieci kołmogorowskich

Z podanego zestawienia w tabeli 4.1 wynika, że kołmogorowskie sieci neuronowe mają zróżnicowaną strukturę oraz odmienne algorytmy uczenia.

Kołmogorowska sieć liniowa ma budowę identyczną z siecią sigmoidalną ale odmienny sposób uczenia. Metoda uczenia polega na wyznaczaniu zewnętrznej funkcji aktywacji przy stałych wagach. Wadą jest niekorzystne narastanie wag przy dużej dokładności obliczeń, co zostało opisane w rozdziale 4.2.1.

W kołmogorowskiej sieci potęgowej wagi są zastąpione przez wartości funkcji potęgowej, co eliminuje niekorzystny problem z narastaniem wag. Nie obowiązuje w tej sieci warunek (4.7) nałozony na wagi w sieci liniowej (p. rozdział 4.2.2). Uczenie sieci potęgowych jest bardziej skomplikowane niż w sieciach liniowych (por. algorytmy w rozdziałach 4.2.1 i 4.2.2)

Kołmogorowska sieć indeksowana ma nietypową architekturę z blokiem transformacji oraz prosty algorytm uczenia, gdzie wyznacza się zewnętrzne funkcje aktywacji. Wagi w tych sieciach nie występują. Sieć ta ma najlepsze właściwości przy konstrukcji algorytmu uczenia (p. algorytmy w rozdziałach 4.3.1).

Z podanego wyżej wynika, że najlepszą siecią typu kołmogorowskiego jest sieć liniowa i indeksowana z uwagi na prostą metodę uczenia sieci i przejrzystą konstrukcję.

4.6 Podsumowanie

Oryginalne twierdzenie Kołmogorowa dotyczące funkcji wielu zmiennych nie nadaje się do budowy algorytmu uczenia sieci neuronowej. Autor proponuje następujące schematy modyfikacji twierdzenia Kołmogorowa:

1. sposób podstawowy, nieindeksowany;
2. sposób indeksowany.

Sposoby tych modyfikacji są możliwe poprzez:

1. wprowadzenie funkcji liniowej jako wewnętrznej funkcji aktywacji;
2. wprowadzenie funkcji potęgowej jako wewnętrznej funkcji aktywacji;

3. wprowadzenie funkcji całkowitoliczbowej jako wewnętrznej funkcji aktywacji;

Aby algorytm uczenia sieci kołmogorowskiej był zbieżny należy odpowiednio dobrać wewnętrzne i obliczyć zewnętrzne funkcje aktywacji (por. rozdziały 4.2 i 4.3). Zewnętrzne funkcje aktywacji oblicza się w zależności od funkcji aproksymowanej h (p. wzór 4.91). Błąd aproksymacji można sprowadzić do wartości dowolnie małej (p. wzór (4.99)).

Aspekty teoretyczne sieci kołmogorowskiej przedstawiają twierdzenia od 1 do 3 w rozdziale 4 natomiast aspekty praktyczne ilustrują schematy blokowe algorytmów uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej podane w rozdziale 5.

Rozdział 5

Algorytmy uczenia zmodyfikowanej sieci

5.1 Wstęp

Rozdział ten zawiera opis algorytmów uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej oraz analizę zbieżności i dokładności skonstruowanych algorytmów uczenia kołmogorowskich sieci neuronowych.

Analiza dotyczy zmodyfikowanego twierdzenia Kołmogorowa w postaci indeksowanej. Dowód zbieżności algorytmów opartych na zmodyfikowanym twierdzeniu Kołmogorowa jest w rozdziale 4, a rozdział 5 zawiera wnioski i uzupełnienia materiału z rozdziału 4.

Rozdział ten zawiera schematy blokowe algorytmów trenowania sieci neuronowej w oparciu o zmodyfikowane indeksowane twierdzenie Kołmogorowa.

5.2 Algorytmy uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej

5.2.1 Wstęp

Obecnie przedstawiono algorytmy uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej. Są to algorytmy z różnymi i identycznymi funkcjami aktywacji. Algorytm z różnymi funkcjami aktywacji stosuje się przy dużej dokładności obliczeń. Algorytm z identycznymi funkcjami aktywacji stosuję sie gdy najważniejszy jest niski nakład obliczeń. Zaletą podanych algorytmów jest ich prosta budowa, niski nakład obliczeń przy trenowaniu sieci oraz dobra zbieżność o dowolnie małym błędzie aproksymacji. Zastosowanie podanych algorytmów umożliwia identyfikację szerokiej klasy obiektów z nieróżniczkowalną nieliniowością (z ostrzami).

5.2.2 Opis algorytmów

Algorytmy uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej można podzielić na następujące etapy:

1. wczytanie danych uczących w postaci wejść $x_i, i = 1, ..., n$ i wyjścia y (funkcji aproksymowanej $h(x_1, ..., x_n)$);
2. zamiana wielkości wejściowej na indeks cząstkowy i całkowity
3. wyznaczenie zewnętrznych funkcji aktywacji g na podstawie wczytanych danych (por. algorytm jednokrokowy z 4.3.1);
4. wygenerowanie funkcji aproksymującej wczytane dane.

Poniżej podano opis algorytmów uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej. Schematy blokowe algorytmów uczenia są przedstawione na rys. 5.3 - 5.4. Są to algorytmy wyznaczające różne lub identyczne zewnętrzne funkcje aktywacji. Algorytm z różnymi funkcjami aktywacji nadaje się do większości badań eksperymentalnych między innymi do identyfikacji. Algorytm z identycznymi funkcjami aktywacji ma zastosowanie wtedy, gdy mała złożoność algorytmu jest ważniejsza niż duża dokładność aproksymacji. Rys. 5.1 przedstawia część wczytującą dane (wspólny dla dwóch algorytmów). Rys. 5.3 przedstawia część wyznaczającą zewnętrzne funkcje aktywacji (wspólny dla dwóch algorytmów). Ponadto w algorytmie z różnymi funkcjami aktywacji wyznaczana jest funkcja aproksymująca. Na tym kończy się opis pierwszego algorytmu. Rys. 5.4 przedstawia część końcową drugiego algorytmu wyznaczającą funkcje aproksymującą przy identycznych funkcjach aktywacji.

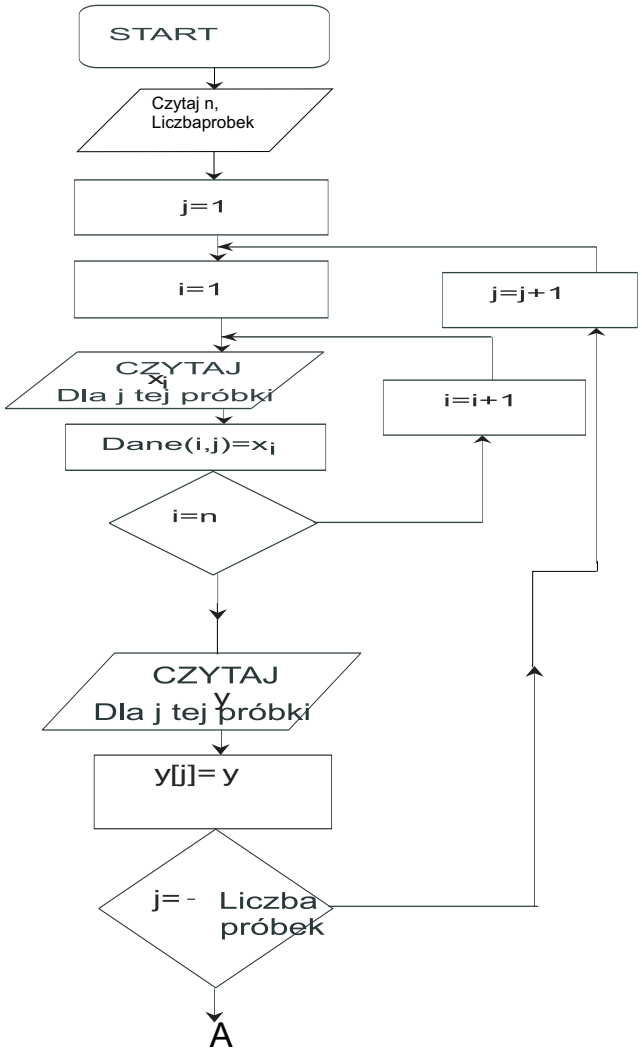
Przykład pracy algorytmu uczenia indeksowanej sieci kołmogorowskiej z różnymi funkcjami aktywacji.

Dana jest funkcja 2 zmiennych $h(x_1, x_2) = x_1 + x_2$. Funkcja h zostaje spróbkowana w 4 punktach ($liczbaprobek = 4$) $x_{1,1} = 0,4$, $x_{2,1} = 0,4$, $y[1] = 0,8$, $x_{1,2} = 0,4$, $x_{2,2} = 0,8$, $y[2] = 1,2$, $x_{1,3} = 0,8$, $x_{2,3} = 0,4$, $y[3] = 1,2$, $x_{1,4} = 0,8$, $x_{2,4} = 0,8$, $y[4] = 1,6$ a dane wpisane do macierzy oznaczonych jako $Dane[i, j]$ i $y[j]$

Macierze $Dane[i, j]$ i $y[j]$ są odpowiednio równe: $Dane[i, j] = \begin{vmatrix} x_{1,1} & x_{2,1} \\ x_{1,2} & x_{2,2} \\ x_{1,3} & x_{2,3} \\ x_{1,4} & x_{2,4} \end{vmatrix} = \begin{vmatrix} 0,4 & 0,4 \\ 0,4 & 0,8 \\ 0,8 & 0,4 \\ 0,8 & 0,8 \end{vmatrix}$

$y[j] = \begin{vmatrix} y[1] \\ y[2] \\ y[3] \\ y[4] \end{vmatrix} = \begin{vmatrix} 0,8 \\ 1,2 \\ 1,2 \\ 1,6 \end{vmatrix}$ Nawiasy || oznaczają symbole macierzy.

Działania algorytmu dla podanego przykładu w części obliczającej zewnętrzną funkcje aktywacji są następujące. Obliczenie zewnętrznej funkcji aktywacji odbywa się w bloku $g[wjj, q] = y[i]/(2n + 1)$ (gdzie wjj jest indeksem całkowitym a $y[i]$ wektorem wielkości wyjściowych aproksymowanej funkcji h) według wzoru (4.91) podanego w rozdziale 4.3.1. Blok z powyższą instrukcją znajduje się wewnątrz dwóch pętli. Pętla wewnętrzna zmienia się wraz z zmianami zmiennej q wskazujacej numer funkcji aktywacji (jest ich 5), pętla zewnętrzna zmienia się wraz z kolejną próbką (jest ich 4). Indeks całkowity wjj wyznaczany jest w bloku $wjj = wjj + (j - 3) * (skok + 1)^{i1-1}$ (por. wzór (4.86) $j_q = \sum_{p=1}^{n} i_p^q M^{p-1}$) - odpowiednikiem indeksu cząstkowego i jest zmienna j, stałej M zmienna $skok + 1$, wykładnika p zmienna $i1$, indeksu całkowitego j zmienna wjj. Funkcja sumy realizowana jest w pętli ze zmienną $i1$. Liczba $i1$ zmienia od 1 do liczby zmiennych $n = 2$. Liczba $skok$ wskazuje na ile części podzielony jest przedział [0,1] (odpowiednik zmiennej K we wzorze (4.67)). Przedział [0,1] jest podzielony na $K = 9n$ części (jak w oryginalnym twierdzeniu



Rysunek 5.1: Algorytm wczytania danych.

Kołmogorowa). Liczba *skok* wynosi $skok = 9n = 9 * 2 = 18$ gdzie $n = 2$ to liczba zmiennych. Najbardziej wewnętrzna pętla służy do wyznaczenia wartości indeksu cząstkowego j (j z algorytmu jest odpowiednikiem $i^q_{p,r}$ we wzorze (4.67)), który wskazuje, w którym miejscu znajduje się próbka (zmienna j jest kolejno zwiększana aż zmienna ww (p. początek rozdziału 4.3.1: $\frac{1}{K}(i^q_{p,r} - \frac{q}{3n}) \leq x_p < \frac{1}{K}(i^q_{p,r} + 1 - \frac{q}{3n})$) przekroczy wartość uu - p. blok w rys. 5.3). W naszym przypadku dla (na przykład) wielkości wejściowej 0,4 (p. macierz *Dane*) mamy j=10 dla q=1,2,3,4 i j=11 dla q=5. Dla przypadku wielkości wejściowej 0,8 (p. macierz *Dane*) mamy j=17 dla q=1,2,3 i j=18 dla q=4,5.

Indeksy całkowite wjj czterech wejściowych próbek (x_1, x_2) przedstawiono na rysunku 5.2 przy zmiennym q. Na przykład dla próbki (0,4;0,4) indeksy cząstkowe wynoszą 10 dla q=1, stąd $wjj = 7 * 19 + 7 = 140$. Dla tych wartości indeksów całkowitych wyznacza się wartości zewnętrznych funkcji aktywacji $g[wjj, q] = y[i]/(2n+1)$ co przedstawiono na rys. 5.2 wraz z wartosciami y.

x_1	x_2	indeks calkowity wjj					y	g[wjj,q]
0,4	0,4	140	140	140	140	160	0,8	0,16
0,8	0,4	147	147	147	148	167	1,2	0,24
0,4	0,8	273	273	273	292	293	1,2	0,24
0,8	0,8	280	280	280	300	300	1,6	0,32
		q= 1	q= 2	q= 3	q= 4	q= 5		

Rysunek 5.2: Indeksy całkowite czterech wejściowych próbek.

Poprzez sumowanie zewnętrznych funkcji aktywacji od q=1 do q=5 według wzoru $hh = hh + g[wjj, q]$ otrzymujemy wartości funkcji aproksymującej hr[1]=0,8, hr[2]=hr[3]=1,2 i hr[4]=1,6.

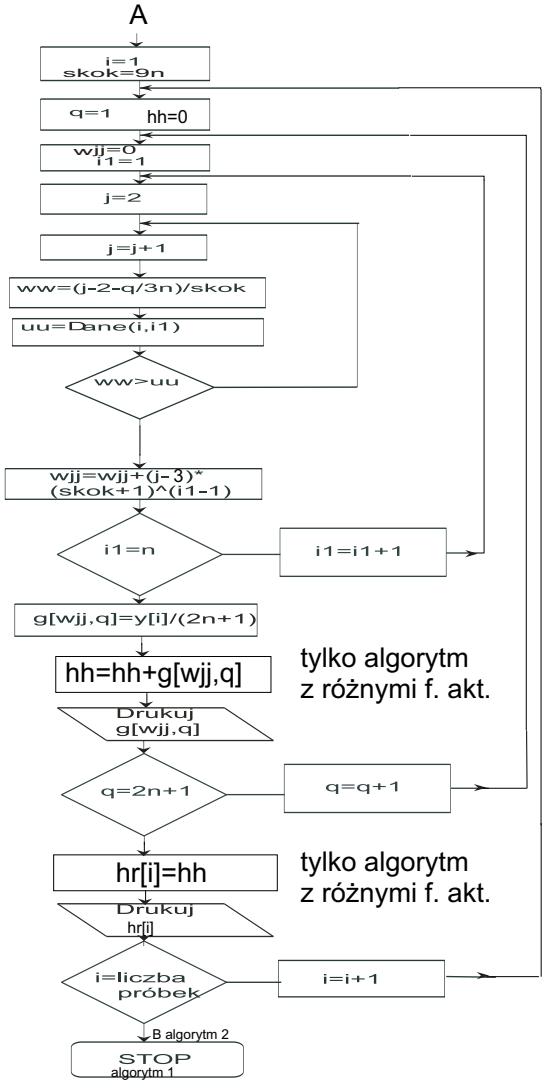
Część algorytmu, która oblicza funkcję aproksymującą dla identycznych zewnętrznych funkcji aktywacji $g_1 = g_2 = ... = g_5$ przedstawia rysunek 5.4 (algorytm z identycznymi funkcjami aktywacji). Algorytm, którego schemat blokowy przedstawiono na rys. 5.3 wyznacza tylko pierwszą zewnętrzną funkcje aktywacji (pozostałe cztery są równe pierwszej) i przekazuje dane to jest zewnętrzną funkcje aktywacji do algorytmu przedstawionego na rys. 5.4.

Funkcję aproksymującą oblicza się na podstawie zewnętrznych funkcji aktywacji, Najważniejsza instrukcja jest w bloku realizującym instrukcję $hh = hh + g[wjj, q]$. Funkcja aproksymująca jest obliczana jako suma pięciu ($2n + 1$ dla $n = 2$) zewnętrznych funkcji aktywacji (wyznaczoną przez algorytm przedstawiony na rys. 5.3) (pętla po q).

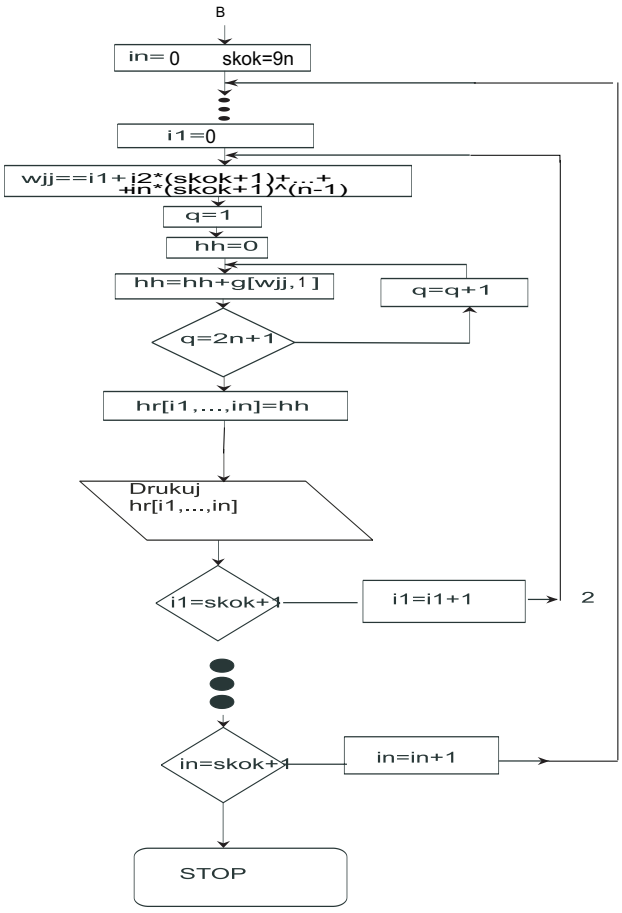
Końcowy wynik identyfikacji przedstawiony jest w tabeli na rys. 5.2.

5.3 Warunki zbieżności i szybkość zbieżności algorytmów uczenia sieci kołmogorowskiej

Zbieżność algorytmów uczenia sieci jest zapewniona zarówno dla zmodyfikowanego twierdzenia Kołmogorowa jak i indeksowanego zmodyfikowanego twierdzenia Kołmogorowa dzięki minimalizacji błędu aproksymacji w trakcie wykonywania algorytmu. Szybkość zbieżności zależy od gęstości 'siatki przestrzennej', na której określona jest aproksymowana funkcja, jak i od zmienności aproksymowanej funkcji. Błąd aproksymacji może być sprowadzony w jednej iteracji algorytmu do wartości dowolnie małej, bo zmienność ciągłej



Rysunek 5.3: Wyznaczanie zewnętrznej funkcji aktywacji w algorytmie z różnymi i jednakowymi funkcjami aktywacji i funkcji aproksymującej w algorytmie z różnymi funkcjami aktywacji.



Rysunek 5.4: Wyznaczanie funkcji aproksymującej w algorytmie z identycznymi funkcjami aktywacji.

aproksymowanej funkcji h na zawężonym przedziale A^q_{K,i_p} (przedział zawężamy zwiekszając K i tym samym przedział jest mniejszy)jest dowolnie małe (co wyjaśniono w rozdziale 4.2.1 i 4.3.1). Jest to fakt ogólnie znany z analizy matematycznej.

Ograniczenia nałożone na wagi synaptyczne podane w równaniu (4.7) (mówiące jakie wartości mogą przyjmować wagi) są warunkiem zbieżności w zmodyfikowanym twierdzeniu Kołmogorowa. Są to warunki wystarczające ale nie konieczne dla zbieżności.

Ciekawą interpretację szybkości zbieżności daje badanie granicy (która jest równa zero)

$$\lim_{K\to\infty} \frac{r}{K(r)} = 0$$

gdzie $K(r)$ jest współczynnikiem we wzorze (4.4) (który rośnie jak maleją przedziały $A^q_{K,i}$)

Mówi ona w jaki sposób rośnie K w kolejnych iteracjach r. Powyższa metoda polega na obliczania stosunku wyrażeń $\frac{r}{K(r)}$ dla dużych wartości K. Im ten stosunek jest większy tym zbieżność jest wolniejsza.

Warunki działania algorytmów uczenia zmodyfikowanej nieindeksowanej sieci neuronowej, (sporządzone na podstawie uwag o wagach z rozdziału 4) dla różnych sieci kołmogorowskich przedstawia tablica 5.1.

lp	sieć	warunek działania algorytmu
1	liniowa n wymiarowa (1)	$w_{q,p} \geq w_{q,p-1}(K+1)$
2	potęgowa 2 wymiarowa	$(n_4 - n_2)(2r)$ niecałkowite p. wzór 4.22
3	potęgowa 3 wymiarowa	$3(n_3 - n_6)r$ niecałkowite p. wzór 4.30

Tablica 5.1: Warunki działania algorytmu dla sieci kołmogorowskiej

Sieci indeksowanych nie uwzględniono w tabeli, ponieważ nie posiadają wag (patrz rozdział 4.3.1). W sieciach indeksowanych warstwę wejściową sieci neuronowej zastępuje blok transformacji. W sieciach indeksowanych problem zbieżności algorytmu nie występuje.

5.4 Podsumowanie

Na podstawie twierdzeń z rozdziału 4 i 5 udowodniono, że algorytmy oparte na zmodyfikowanym twierdzeniu Kołmogorowa (indeksowanym, w wersji wyrażonej wzorem, z różnymi i identycznymi funkcjami aktywacji) mają błąd aproksymacji dowolnie mały.

Algorytmy uczenia sieci kołmogorowskiej mają prostą budowę i nadają się do zastosowania w identyfikacji systemów statycznych i dynamicznych. Jest to dokładnie zilustrowane w rozdziale piątym i szóstym.

Rozdział 6

Eksperymenty ze zmodyfikowaną siecią

6.1 Wstęp

W niniejszym rozdziale rozważa się zastosowanie zmodyfikowanej indeksowanej sieci neuronowej kołmogorowskiej do identyfikacji systemów dynamicznych Hammersteina i Wienera [30]. Powyższe systemy są zbudowne z elementów nieliniowych i dynamicznych. Systemy te służą w technice do modelowania procesów biologicznych i socjologicznych i obejmują szeroką klasę obiektów w przyrodzie, a ich identyfikacja przysparza wiele problemów. Identyfikacja tych systemów sprowadza się do określenia postaci funkcji w elementach nieliniowych i współczynników równania różnicowego w elementach dynamicznych. Literatura podaje wiele metod identyfikacji systemów Wienera i Hammersteina za pomocą sigmoidalnych sieci neuronowych [9].

Metodami identyfikacji systemów Hammersteina i Wienera według Janczaka są [9]:

1. metody korelacyjne [39];
2. metody regresji parametrycznej [40];
3. metody regresji nieparametrycznej [41];
4. metody wykorzystujące gradientowe techniki optymalizacji [31];
5. metody wykorzystujące modele neuronowe [9].

Wszystkie te metody są skomplikowane i wymagają dużego nakładu obliczeń. Zastosowanie kołmogorowskiej sieci neuronowej z funkcjami liniowymi lub całkowitoliczbowymi jako wewnętrznymi funkcjami aktywacji dają uproszczenie obliczeń. Proponowany przez autora sposób identyfikacji idealnie nadaje się do utworzenia prostego algorytmu komputerowego opisanego w rozdziale 4.3, ponieważ otrzymywana zewnętrzna funkcja aktywacji systemu dynamicznego Hammersteina i Wienera jest poszukiwaną funkcją nieliniowości systemu.

Badanie eksperymentalne uczenia sieci służą do ilustracji twierdzeń udowodnionych przez autora i zamieszczonych w rozdziale 4 pracy. W badaniach rozpatrzono jeden obiekt statyczny oraz cztery dynamiczne. Wszystkie obiekty są modelowane funkcjami dwóch zmiennych. Została tu podwierdzona dokładność i zbieżność podanego przez autora algorytmu uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej.

6.2 Identyfikacja charakterystyk obiektów dynamicznych za pomocą sigmoidalnej sieci neuronowej

6.2.1 Modele identyfikacji obiektów dynamicznych

Celem identyfikacji obiektu jest znalezienie modelu (wzoru) matematycznego, który opisuje relacje pomiędzy wielkością wejściową a wyjściową obiektu. Możemy identyfikować obiekty statyczne lub obiekty dynamiczne. Każdy z tych obiektów może być liniowy lub nieliniowy.

Identyfikację obiektów dynamicznych opisanych przez równania różnicowe rozważają Narendra i Parthasarathy w swojej pracy [16]. Podają oni cztery następujące modele opisane przez równania nieliniowe. Mimo, że modele są opisane równaniami różnicowymi, nieliniowość modelu jest opisana funkcją ciągłą.

Model 1 opisany jest równaniem:

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + g[x(k), x(k-1), ..., x(k-m+1)] \tag{6.1}$$

W modelu 1 zależność wyjścia od wyjść w poprzednich chwilach jest liniowa.

Model 2 opisany jest równaniem:

$$y(k+1) = f[y(k), y(k-1), ..., y(k-n+1)] + \sum_{i=0}^{m-1} \beta_i x(k-i) \tag{6.2}$$

W modelu 2 zależność wyjścia od wejść w poprzednich chwilach jest liniowa.

Model 3 opisany jest równaniem:

$$y(k+1) = f[y(k), y(k-1), ..., y(k-n+1)] + g[x(k), x(k-1), ..., x(k-m+1)] \tag{6.3}$$

W modelu 3 zależność wyjścia od wyjść i wejść w poprzednich chwilach jest nieliniowa.

Model 4 opisany jest równaniem:

$$y(k+1) = f[y(k), y(k-1), ..., y(k-n+1), x(k), x(k-1), ..., x(k-m+1)] \tag{6.4}$$

Model 4 jest najbardziej ogólnym i zależność wyjścia od wejść i wyjść w poprzednich chwilach jest nieliniowa.

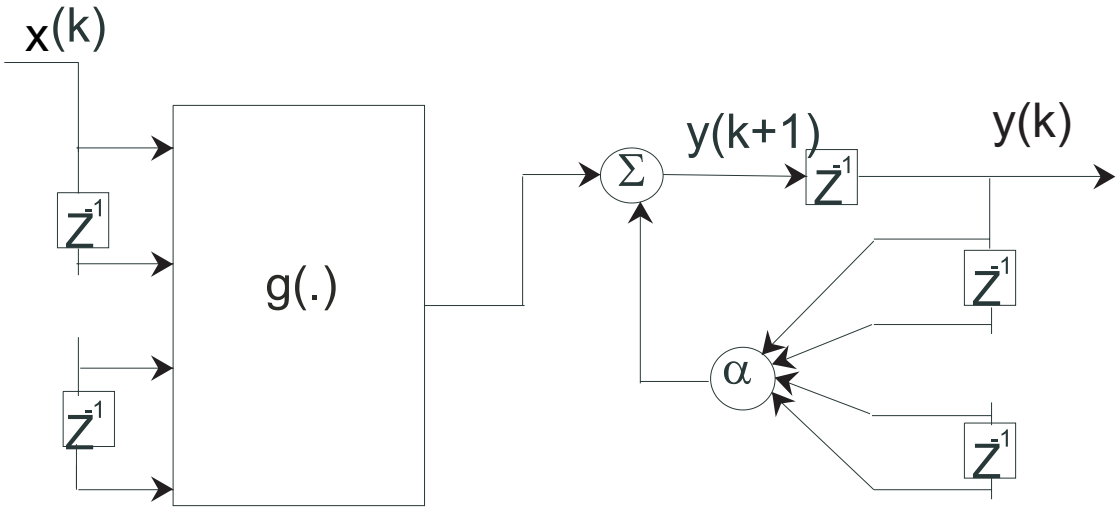
Przez y oznaczono wyjścia obiektu a przez x wejścia obiektu. Rząd układu względem sygnału wyjściowego wynosi n, a względem sygnału wejściowego m.

Reprezentacje modelu 1 przedstawiono na rysunku 6.1.

Identyfikacja obiektu polega na wyznaczeniu współczynników równania różnicowego i funkcji nieliniowej występujących w modelu. Współczynniki równania różnicowego wyznacza się metodą najmniejszych kwadratów, a funkcję nieliniową przy zastosowaniu metod gradientowych i wstecznej propagacji błędu [3],[15],[17].

6.2.2 Zarys identyfikacji systemów dynamicznych Hammersteina i Wienera

Autor ograniczył się do identyfikacji złożonych obiektów dynamicznych takich jak systemy Hammersteina i Wienera, które są opisane w rozdziale 6. System Hammmersteina



Rysunek 6.1: Reprezentacja modelu wyrażonego wzorem (6.1)

jest połączeniem kolejno: elementu nieliniowego i dynamicznego liniowego, a system Wienera jest połączeniem kolejno: elementu liniowego dynamicznego i elementu nieliniowego. Są one często stosowane w technice do modelowania różnych procesów biologicznych i socjologicznych. Identyfikację złożonych obiektów dynamicznych takich jak Hammersteina i Wienera rozważa Janczak w swej pracy [9]. Są możliwe dwa sposoby modelowania obiektu: szeregowo-rownoległe lub równoległe, co objaśniono poniżej.

Połączenie obiektu i modelu przedstawiaja rysunki 6.2 i 6.3.

W modelu szeregowo-równoległym do modelowania stosowane są sygnały wejściowe x i sygnały wyjściowe y z obiektu. W modelu równoległym do modelowania stosowane są sygnały wejściowe x i zamiast sygnałów wyjściowych y z obiektu sygnały wyjściowe y^m z modelu.

Dla modeli szeregowo-równoległych stosowane są równania systemu Hammersteina (por. wzór (6.1) modelu 1)

$$y_k^m = -a_1 y_{k-1} - ... - a_n y_{k-n} + b_1 f(x_{k-1}) + ... + b_n f(k_{i-n}) \tag{6.5}$$

Model szeregowo-równoległy dla systemu Hammersteina przedstawia rysunek 6.4 (por. rysunek 6.2 i wejścia modelu x i y), natomiast system Wienera opisany jest przez (por. wzór (6.4) modelu 4)

$$y_k^m = f(s_k) \tag{6.6}$$

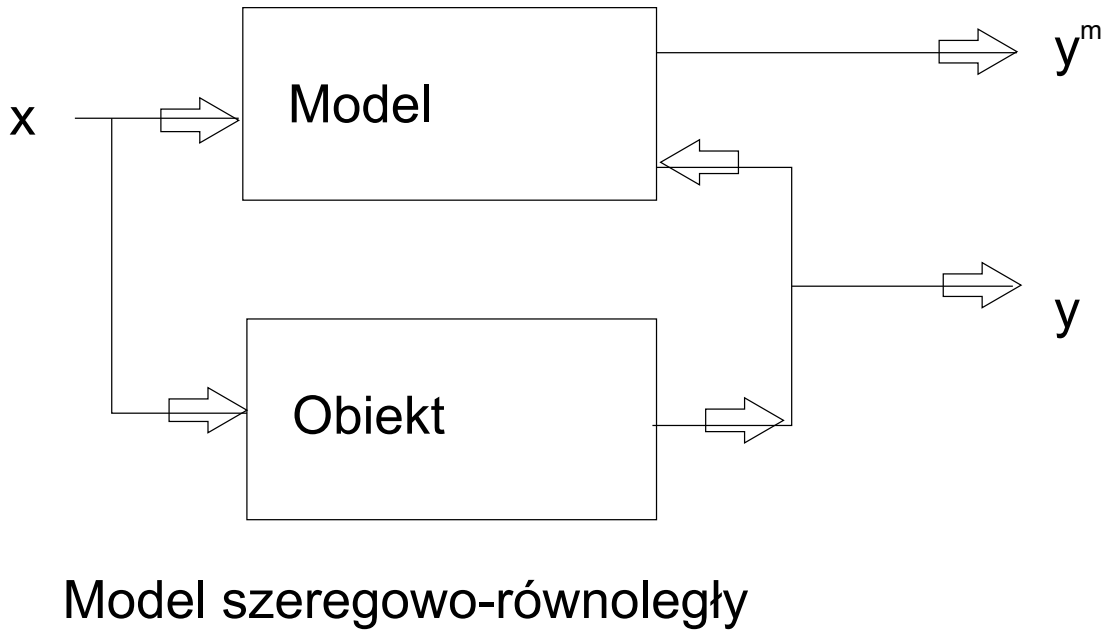
gdzie zmienna pomocnicza s_k jest wyrazona przez:

$$s_k = -a_1 f^{-1}(y_{k-1}) - ... - a_n f^{-1}(y_{k-n}) + b_1 x_{k-1} + ... + b_n x_{k-n} \tag{6.7}$$

gdzie y_i^m jest sygnałem wyjściowym modelu a y_i sygnałem wyjściowym badanego obiektu, x_i sygnałem sterowania a f^{-1} oznacza funkcje odwrotną do f.

Dla modeli równoległych stosowane są równania systemu Hammersteina i Wienera podobne do poprzednich z tym, że zamiast y_i wprowadzane jest y_i^m.

Model równoległy dla systemu Hammersteina przedstawia rysunek 6.5 (por. rysunek 6.3 i wejście modelu x)



Rysunek 6.2: Model szeregowo-równoległy.

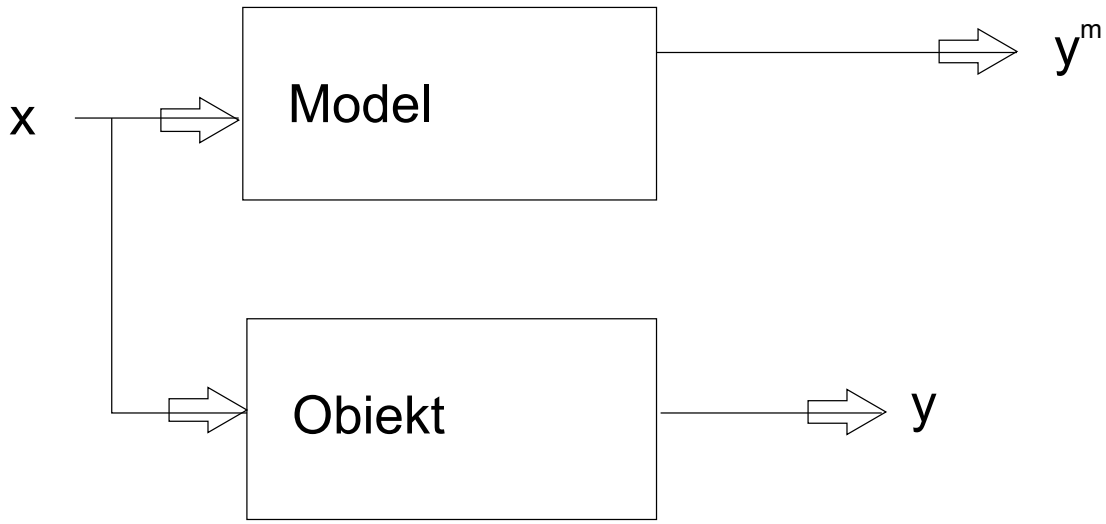
6.3 Identyfikacja charakterystyk nieliniowych obiektów dynamicznych przy zastosowaniu zmodyfikowanej indeksowanej sieci kołmogorowskiej

.

Przedstawiony w rozdziale 5.2 algorytm uczenia zmodyfikowanej sieci neuronowej może być wykorzystany do identyfikacji charakterystyk nieliniowych. Przykłady ilustrujące identyfikacje charakterystyk nieliniowych podano w niniejszym rozdziale.

Zadanie identyfikacji obiektów nieliniowych sprowadza się do określenia zewnętrznych funkcji aktywacji g_q zmodyfikowanej sieci kołmogorowskiej. Aby zidentyfikować obiekt należy:

1. określić wielkości wejściowe sieci neuronowej (ustala użytkownik), które mogą być uzyskane generatorem losowym lub ustalane przez użytkownika;
2. określić wielkość wyjściową y sieci neuronowej (ustala użytkownik);
3. określić liczbę neuronów (im większa liczba neuronów tym mniejszy błąd aproksymacji);
4. zastosować algorytm uczenia zmodyfikowanej indeksowanej sieci kołmogorowskiej (p. rys. 5.3);
5. zastosować wewnętrzne funkcje aktywacji $f_{p,q}$ jako funkcje całkowitoliczbowe (p. sieci indeksowane);
6. określić zewnętrzne funkcje aktywacji g_q, co jest główną częścią zadania identyfikacji (p. rys. 5.3);



Rysunek 6.3: Model równoległy.

Zmodyfikowana indeksowana sieć kołmogorowska idealnie nadaje się do zadania identyfikacji nieliniowych obiektów dynamicznych. Zadanie identyfikacji dotyczy układu dynamicznego z czasem dyskretnym i dowolną funkcją nieliniową.

Przyjęto, że wyjście obiektu dynamicznego jest funkcją wejść i wyjść opóźnionych, co możemy określić następującym wzorem (por. model 4 wyrażony wzorem (6.4))

$$y(k+1) = f[y(k), y(k-1), ..., y(k-n+1), x(k), x(k-1), ..., x(k-m+1)] \qquad (6.8)$$

gdzie liczba n wskazuje liczbę wyjść opóźnionych (względem chwili bieżącej k) a liczba m liczbę wejść opóźnionych.

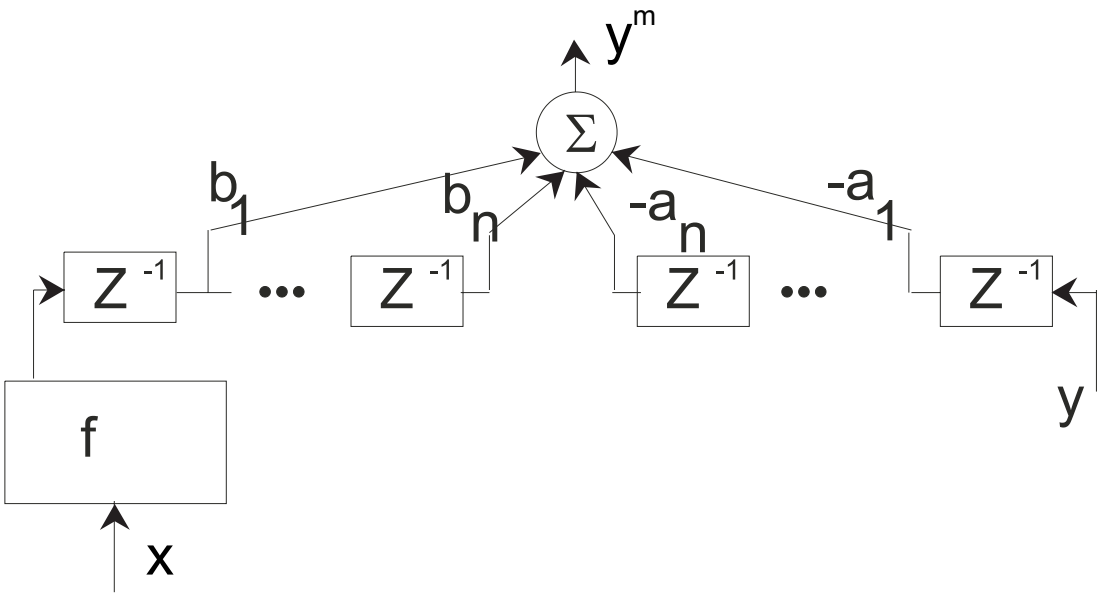
Aby nauczyć sieć musimy wygenerować wszystkie możliwe wartości wejściowe układu którymi są:

- wielkości sterujące x oraz

- wielkości wyjściowe y w poprzednich chwilach czasu.

Wielkości sterujące x są zmiennymi niezależnymi i można je łatwo uzyskać. Większą trudnością jest uzyskanie występujących w poprzednich chwilach czasu wielkości wyjściowych y (dlatego potrzebny jest generator losowy wielkości x, bo gdy x jest losowe to y jest też losowe). Po dostatecznie długim czasie pracy algorytmu potrzebnym do wygenerowania kilkuset próbek, są wygenerowane (losowo) prawie wszystkie możliwe wartości wielkości wyjściowych y i sieć neuronowa staje się prawidłowo nauczona, co potwierdzają wyniki pracy. Wartości y, nie występujące podczas uczenia sieci są odtwarzane z błędem. Błąd ten można zmniejszyć interpolując lub aproksymując wartości nieznane wartościami sąsiednimi (problem ten wykracza poza ramy pracy).

Uczenie sigmoidalnej sieci neuronowej polega na odpowiednim doborze wag minimalizujących funkcje błędu, przy czym w miarę uczenia wielkości wyjściowe stają się coraz bliższe poprawnych. Zasadniczą różnicę w procesie uczenia sieci sigmoidalnej i zmodyfikowanej przez autora indeksowanej sieci kołmogorowskiej polega na zastąpieniu wag przez blok transformacji w sieci kołmogorowskiej.



Rysunek 6.4: Szeregowo-równoległy model Hammersteina.

6.4 Identyfikacja systemu statycznego

Identyfikację systemu statycznego przeprowadzono przy użyciu programu zbudowanego według algorytmu uczenia sieci neuronowej i opartego na indeksowanym zmodyfikowanym twierdzeniu Kołmogorowa, ponieważ nieindeksowane wersje algorytmu są bardziej złożone w zastosowaniu.

Przeprowadzono identyfikację obiektu wyrażonego równaniem

$$y = 10 + \exp(x_1 + x_2) \tag{6.9}$$

Identyfikacja polega na wyznaczeniu zewnętrznych funkcji aktywacji sieci. Następuje przy tym redukcja wymiaru danych wejściowych z n do 1 zwana w literaturze angielskiej PCA (principial component analysis - analiza głównych składowych) [45], bo identyfikowana funkcja h n zmiennych jest przedstawiona w postaci sumy zewnętrznych funkcji aktywacji g_q, które są funkcjami jednej zmiennej j. W wyniku identyfikacji otrzymano pięć zewnętrznych funkcji aktywacji ($2n + 1$ dla $n = 2$), z czego pierwszą przedstawiono na rysunku 6.6. Zewnętrzne funkcje aktywacji wyrażone są wzorem

$$g_q(j_q) = \frac{1}{l} h(\varsigma^q_{1,K}, ..., \varsigma^q_{n,K}) \tag{6.10}$$

gdzie l jest liczbą neuronów równą 5. Dla badanej funkcji otrzymamy

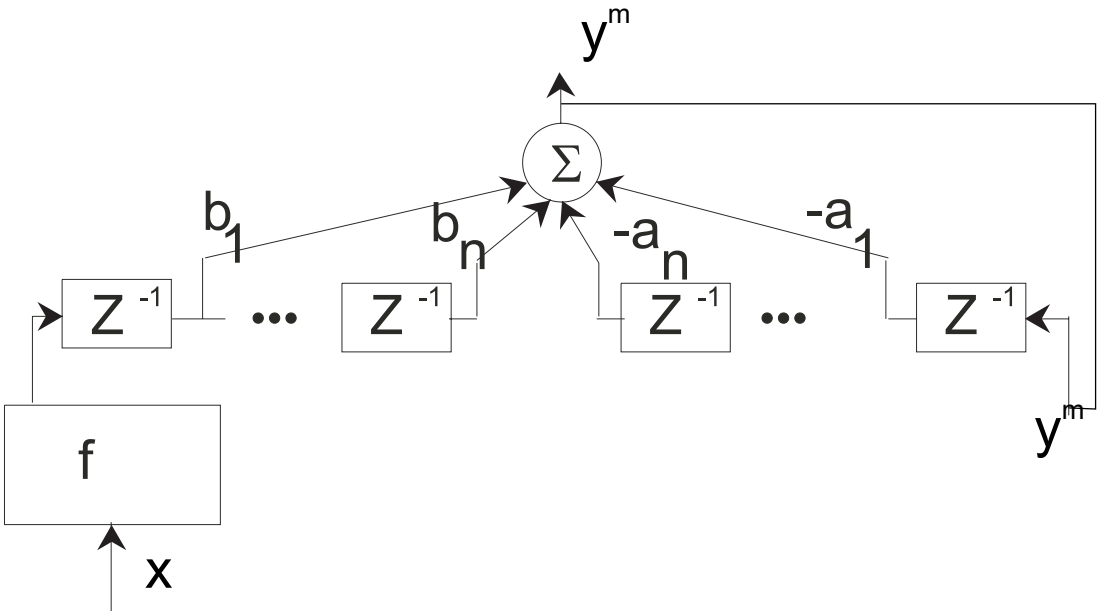
$$g_q(j_q) = \frac{1}{5}(10 + exp(\varsigma^q_{1,K} + \varsigma^q_{2,K})) \tag{6.11}$$

Podstawiając ze wzoru (4.72) $\varsigma^q_{1,K} = \frac{i^q_1}{18} - \frac{q}{108} + \frac{1}{18}$ i $\varsigma^q_{2,K} = \frac{i^q_2}{18} - \frac{q}{108} + \frac{1}{18}$.

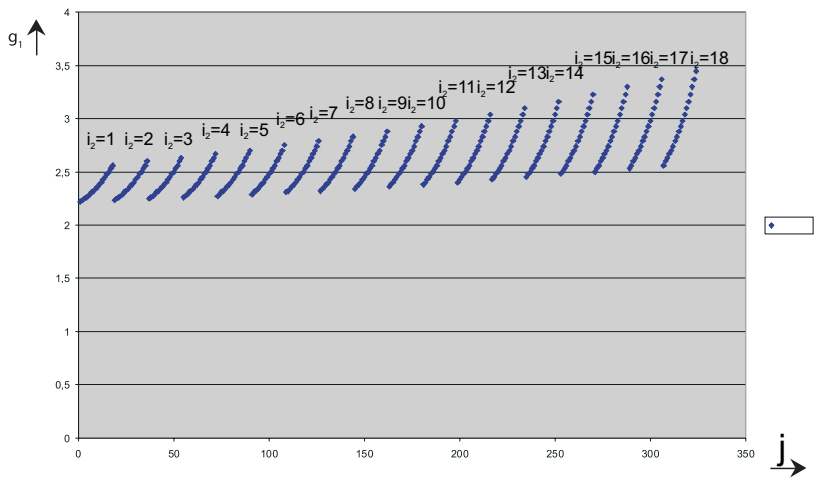
Przyjęto $K = 18$ zgodnie z zaleceniem Kołmogorowa aby $K = 9n$ gdzie n to liczba zmiennych (równa 2).

Stąd

$$g_q(j_q) = 2 + \frac{1}{5} exp(\frac{i^q_1}{18} + \frac{i^q_2}{18} - \frac{q}{54} + \frac{2}{18}) \tag{6.12}$$



Rysunek 6.5: Równoległy model Hammersteina.



Rysunek 6.6: Pierwsza funkcja aktywacji systemu statycznego.

Ponieważ $j_q = i_1^q + 19i_2^q$ lub

$$i_1^q = j_q - 19i_2^q \tag{6.13}$$

to otrzymamy

$$g_q(j_q) = 2 + \frac{1}{5}exp(\frac{j_q}{18} - i_2^q - \frac{q}{54} + \frac{2}{18}) \tag{6.14}$$

Pozostałe cztery zewnętrzne funkcje aktywacji pominięto ze względu na identyczny kształt i przesunięcie wykresu o wektor $\frac{q-6}{54}$ względem osi OY w stosunku do pierwszej funkcji aktywacji (por. wzór (6.14)). Identyfikacja systemu statycznego umożliwia redukcje wymiaru funkcji aproksymowanej (równej 2) do wymiaru zewnętrznej funkcji aktywacji (równej 1). Możliwe staje się przy tym modelowanie funkcji wielu zmiennych przy pomocy sieci kołmogorowskiej.

Na rysunku 6.6 widać wyraźnie eksponencjalny charakter funkcji aktywacji. Widoczna jest regularność powtarzania danych w 18 cyklach, bo przedział [0,1] jest podzielony na $K = 18$ części (dowolna liczba całkowita)), a cykle wynikają ze zmiany parametru i_2

funkcji eksponencjalnej w 18 częściach przedziału [0,1] (wraz ze zmianą parametru i_2 przesuwa się wykres funkcji g p. wzór (6.14)). Na wykresie 6.6 mamy rodzinę funkcji aktywacji g z 18 wartościami parametru i_2. Liczba 18 wynika z zalecenia podanego w twierdzeniu Kołmogoorwa, a które mówi, że przedział wejsciowy dzielimy na $9n$ części, gdzie n to liczba zmiennych wejsciowych ($n = 2$).

6.5 Identyfikacja systemów dynamicznych Hammersteina

Systemy dynamiczne Hammersteina składają się z kaskadowego połączenia elementu nieliniowego (pierwszego elementu) i elementu dynamicznego liniowego (drugiego elementu). Identyfikacja systemu przy użyciu sieci kołmogorowskiej polega na wyznaczeniu zewnętrznej funkcji aktywacji g_q, która przedstawia szukaną nieliniowość systemu.

Pierwszy eksperyment przeprowadzono na obiekcie jak na rysunku 6.7 (system Hammersteina 1), który jest opisany równaniem

$$y(k) = 0,5y(k-1) + 0,5\sqrt{x(k-1)} \tag{6.15}$$

gdzie k jest indeksem określającym kolejne chwile czasu lub inaczej zapisując

$$y(k) = \Phi(y(k-1), x(k-1)) = 0,5y(k-1) + 0,5\varphi(x(k-1)) \tag{6.16}$$

gdzie $\varphi()$ jest nieliniowością w systemie Hammersteina (w naszym przykładzie φ jest funkcją pierwiastkową) a Φ jest funkcją dwu zmiennych. Zewnętrzna funkcja aktywacji zgodnie ze wzorem (4.93) (dla $2n + 1 = 5$ neuronów) jest równa

$$g_q(j) = \frac{1}{5}h(\zeta_1^q, \zeta_2^q) = \frac{1}{5}y(k) = \frac{1}{5}\Phi(y(k-1), x(k-1)) \tag{6.17}$$

gdzie

$$\zeta_1^q = x(k-1), \zeta_2^q = y(k-1) \tag{6.18}$$

a h jest badaną funkcją realizowaną przez system Hammersteina.

Stąd otrzymuje się

$$g_q(j) = 0,1\varphi(x(k-1)) + 0,1y(k-1) \tag{6.19}$$

i dalej

$$g_q(\sum_{p=1}^{2} M^{p-1} i_p) = 0,1\varphi(x(k-1)) + 0,1y(k-1) \tag{6.20}$$

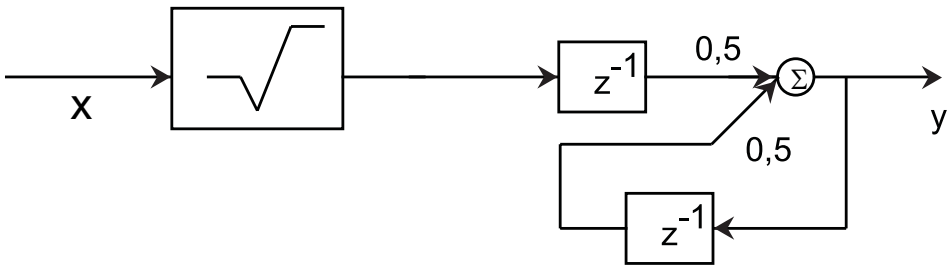
gdzie p wskazuje numer zmiennej, stąd

$$g_q(i_1(x) + j_\alpha) = 0,1\varphi(u(k-1)) + y_0 \tag{6.21}$$

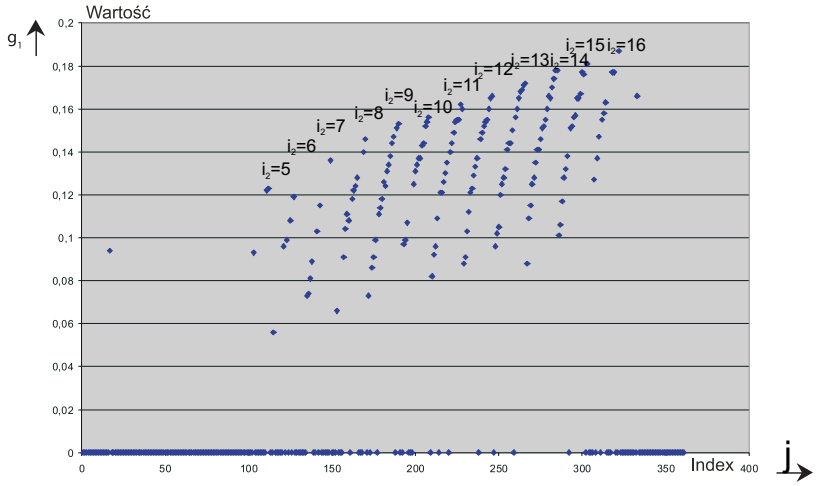
gdzie j_α jest sumą indeksów od i_2 do i_n pomnożonych przez stałą M w potędze p (p. wzór (4.68)). Indeks i_1 jest zależny od x. Zewnętrzna funkcja aktywacji $g_q(x)$ przedstawia szukaną funkcję nieliniowości systemu Hammersteina $\varphi(x)$ i jest wynikiem identyfikacji.

W analizowanym przykładzie

$$g_q(i_1(x)) = y_0 + 0,1\sqrt{x} \tag{6.22}$$



Rysunek 6.7: System Hammersteina 1.



Rysunek 6.8: Pierwsza funkcja aktywacji systemu Hammersteina 1.

Otrzymano pięć funkcji aktywacji (p. rys. 6.8 do 6.12) dla $n = 2$, gdzie n jest liczbą zmiennych badanej nieliniowej funkcji. Na wykresie z rys. 6.8 zaznaczono parametry wykresu i_2. Punkty na wykresach są losowe, bo wejście obiektu jest pobudzane losowo (generator losowy wielkości wejsciowej x z rozkładem prostokatnym z przedziału [0,1]) i na wykresach są widoczne niezerowe punkty wygenerowane przez system Hammersteina. Algorytmy w rozdziale 5 przedstawiają część analizującą wczytane dane z obiektu z pominięciem części generującej dane wejściowe (metoda identyfikacji p. podrozdział 6.3 str. 57 jest bardziej rozbudowana niż algorytm uczenia sieci, który obejmuje punkty 4-6 ze str. 57).

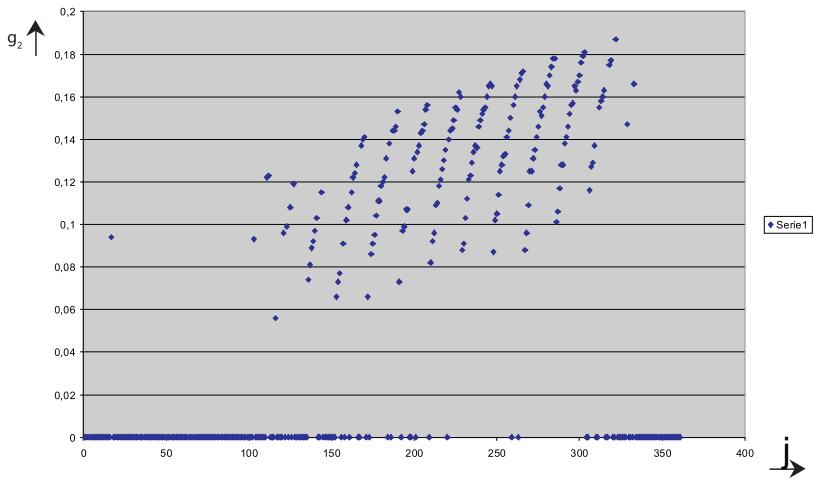
Na wykresach widać wyraźnie pierwiastkowy charakter funkcji aktywacji i regularność jej powtarzania w 18 cyklach (przedział [0,1] podzielono na 18 części). Uzasadnienie cyklów takie jak w przypadku obiektu statycznego. Możliwe jest liczbowe określenie nieliniowości na podstawie wartości zewnętrznej funkcji aktywacji g_q. Funkcja aktywacji g_q reprezentuje szukaną nieliniowość systemu.

Porównano wartości wyjściowe obiektu i sieci neuronowej dla 50 kroków wartości czasowej. Wartość absolutna błędu (będącego różnicą wyjścia obiektu i sieci neuronowej) została przedstawiona na rys 6.13.

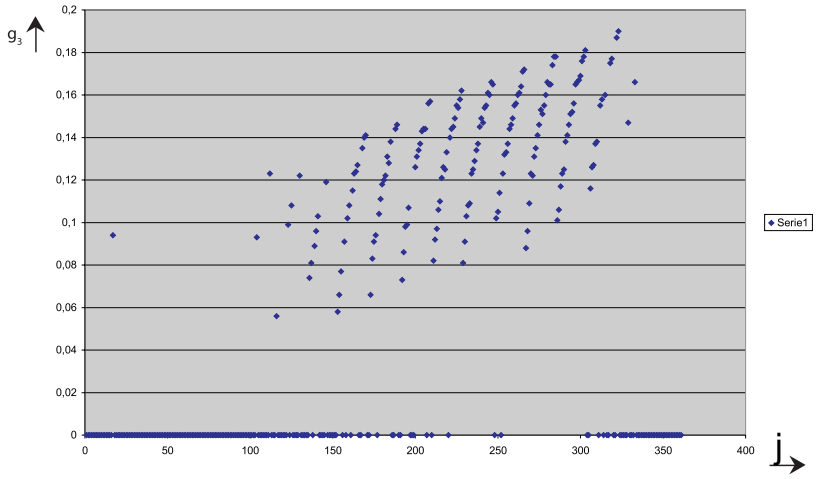
Drugi eksperyment przeprowadzono na obiekcie jak na rysunku 6.14 (system Hammersteina 2) stosując tą samą procedurę jak przy pierwszym badaniu.

Obiekt można opisać równaniem

$$y(k) = 0,5y(k-1) + 0,5(x(k-1))^2 \tag{6.23}$$



Rysunek 6.9: Druga funkcja aktywacji systemu Hammersteina 1.



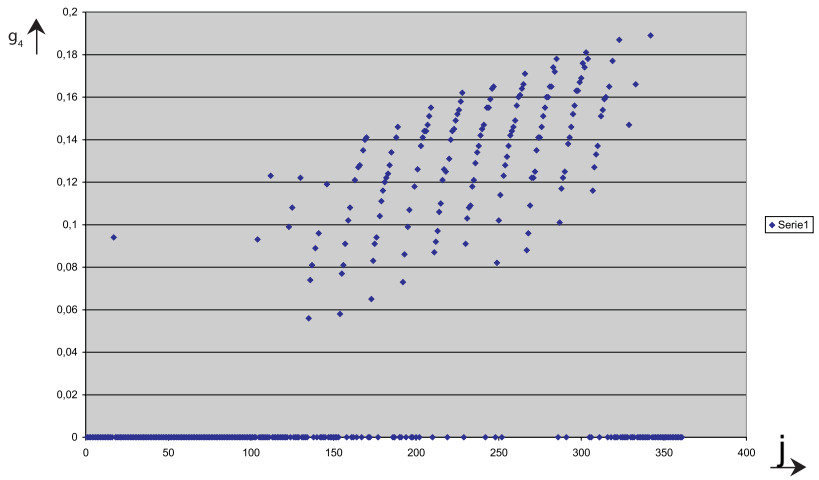
Rysunek 6.10: Trzecia funkcja aktywacji systemu Hammersteina 1.

Otrzymano pięć funkcji aktywacji podobnych do poprzednich z czego jedną przedstawiono na rysunku 6.15.

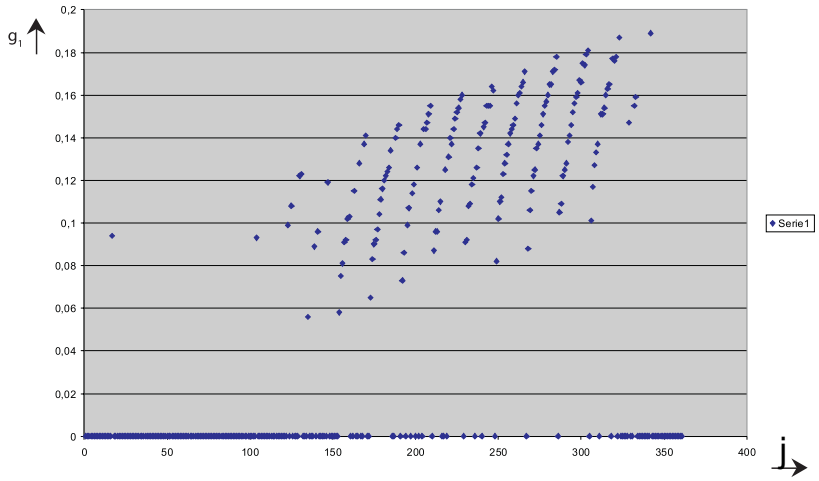
Na wykresach widać wyraźnie kwadratowy charakter funkcji aktywacji i regularność jej powtarzania w 18 cyklach. Uzasadnienie cyklów takie jak w przypadku obiektu statycznego. Możliwe jest liczbowe określenie nieliniowości na podstawie wartości zewnętrznej funkcji aktywacji g_q.

Na podstawie współrzędnych zewnętrznych funkcji aktywacji g_q można oszacować współczynnik równania różnicowego (a=0,5) ze wzoru $(2n+1)K(g_q(j_2) - g_q(j_1) = 5(0,016 - 0,010)18 = 0,54$. Liczby $g_q(j_2) = 0,016$ i $g_q(j_1) = 0,010$ to współrzędne (na wykresie z rys. 6.14) odpowiadające początkom dwóch kolejnych powtarzających się fragmentów wykresu czyli cyklów (indeksy j_2, j_1 to współrzędne początkowych punktów cyklów zaznaczone na rys. 6.15).

System dynamiczny Hammersteina może mieć nieliniowość w postaci funkcji nieróżniczkowalnej (z ostrzami). Identyfikacja sigmoidalnymi sieciami neuronowymi takiej nieliniowości nie jest możliwa, natomiast sieciami kołmogorowskimi jest możliwa.



Rysunek 6.11: Czwarta funkcja aktywacji systemu Hammersteina 1.



Rysunek 6.12: Piąta funkcja aktywacji systemu Hammersteina 1.

6.6 Identyfikacja systemów dynamicznych Wienera

Systemy dynamiczne Wienera składają się z kaskadowego połączenia elementu dynamicznego liniowego (pierwszego elemnetu) i elementu nieliniowego (drugiego elementu).

Przeprowadzono identyfikacje na obiekcie jak na rysunku 6.16 (system Wienera 1).

Obiekt z rysunku 6.16 można opisać równaniami (6.24) i (6.25)

$$y(k) = \sqrt{w(k)} \tag{6.24}$$

gdzie

$$w(k) = 0,5w(k-1) + 0,5x(k-1) \tag{6.25}$$

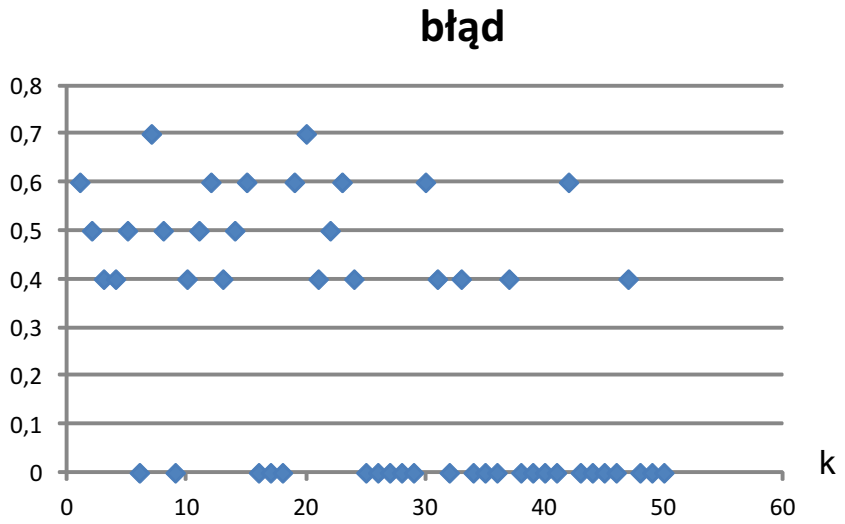
a k jest indeksem określającym kolejne chwile czasu.

Stąd

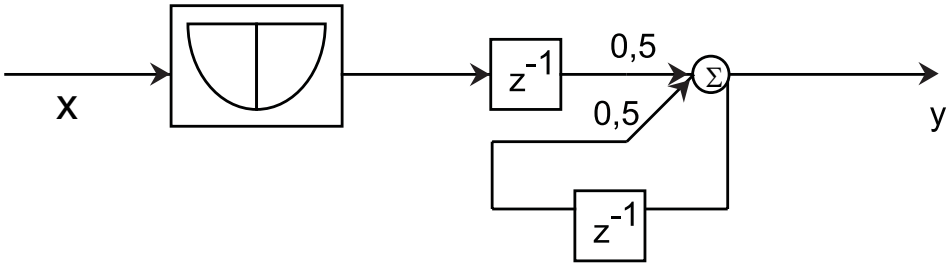
$$y(k) = \sqrt{(0,5(y(k-1))^2 + 0,5x(k-1))} \tag{6.26}$$

lub inaczej

$$y(k) = \phi(y(k-1), x(k-1)) = \varphi(0,5w(k-1) + 0,5x(k-1)) \tag{6.27}$$



Rysunek 6.13: Błąd absolutny różnicy sygnału wyjściowego obiektu i sieci neuronowej



Rysunek 6.14: System Hammersteina 2.

gdzie $\varphi()$ jest nieliniowością w systemie Wienera (w naszym przykładzie φ jest funkcją pierwiastkową)

Zewnętrzna funkcja aktywacji zgodnie ze wzorem (4.93) jest równa

$$g_q(j) = \frac{1}{5}h(\zeta_1^q, \zeta_2^q) = \frac{1}{5}y[k] = \frac{1}{5}\Phi(y(k-1), x(k-1)) \tag{6.28}$$

gdzie

$$\zeta_1^q = x(k-1), \zeta_2^q = y(k-1) \tag{6.29}$$

stąd otrzymuje się

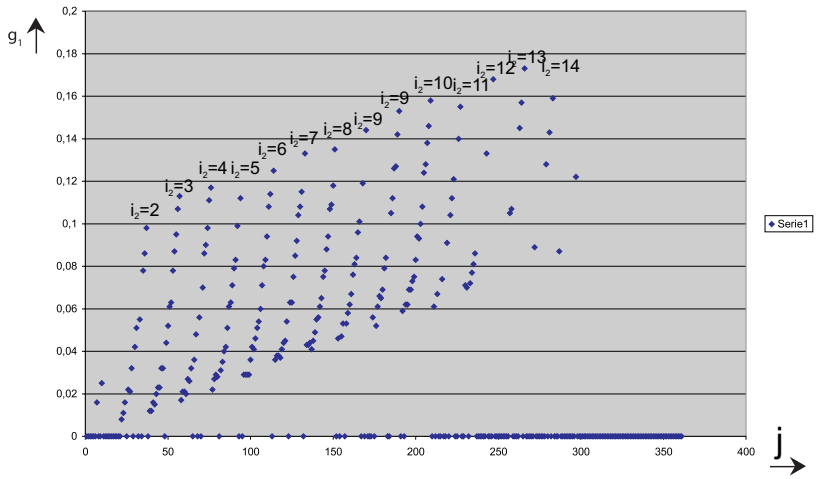
$$g_q(j) = \frac{1}{5}\varphi(0,5x(k-1) + 0,5\varphi^{-1}(y(k-1))) \tag{6.30}$$

Dla funkcji addytywnej φ to znaczy takiej, dla której wartość sumy składników jest równa sumie wartości, otrzymuje się:

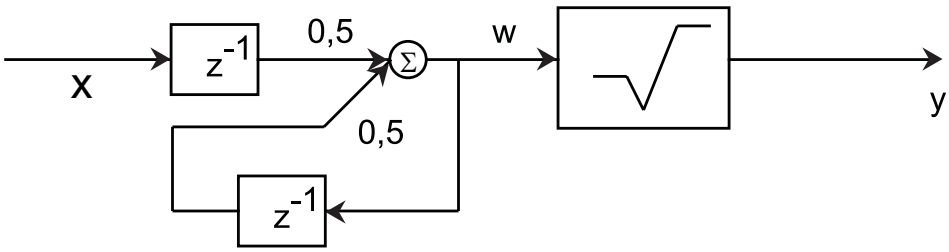
$$g_q(\sum_{p=1}^{2} M^{p-1} i_p) = 0,2\varphi(x(k-1)) + 0,2\varphi^{'}(y(k-1)) \tag{6.31}$$

stąd

$$g_q(i_1(x) + j_\alpha) = 0,2\varphi(0,5x(k-1)) + y_0 \tag{6.32}$$



Rysunek 6.15: Pierwsza funkcja aktywacji systemu Hammersteina 2.



Rysunek 6.16: System Wienera 1.

Zewnętrzna funkcja aktywacji $g_q(x)$ przedstawia szukaną funkcję nieliniowości systemu Wienera $\varphi(x)$.

Otrzymano pięć (podobnie jak dla systemu Hammersteina) funkcji aktywacji, z których pierwszą przedstawiono na rysunku 6.17. Pozostałe rysunki są podobne.

Na wykresach widać wyraźnie pierwiastkowy charakter funkcji aktywacji i regularność jej powtarzania w 18 cyklach. Uzasadnienie cyklów takie jak w przypadku obiektu statycznego.

Drugie badanie przeprowadzono na obiekcie jak na rysunku 6.18 (system Wienera 2) według procedury jak dla pierwszego badania.

Obiekt można opisać równaniami (6.33) i (6.34)

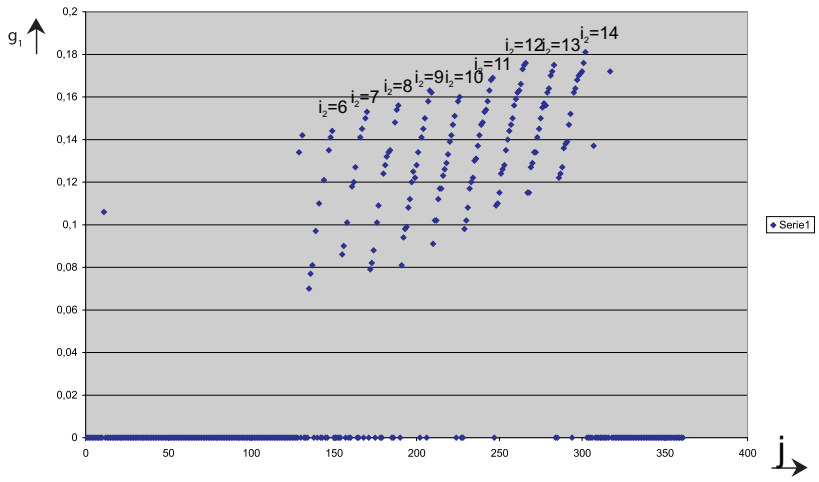
$$y(l) = w(l)^2 \tag{6.33}$$

gdzie

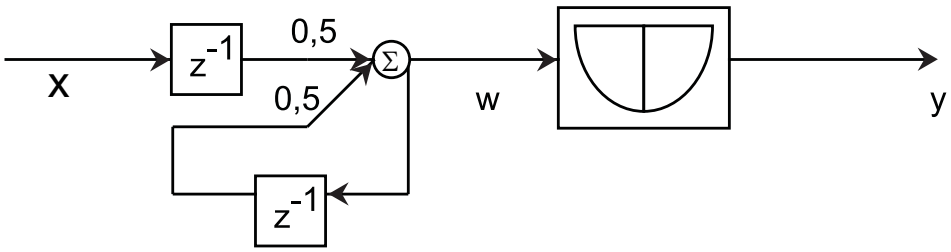
$$w(l) = 0,5w(l-1) + 0,5x(l-1) \tag{6.34}$$

Otrzymano pięć funkcji aktywacji podobnych do poprzednich, z czego jedną przedstawiono na rysunku 6.19.

Na wykresach widać wyraźnie kwadratowy charakter funkcji aktywacji i regularność jej powtarzania w 18 cyklach. Uzasadnienie cyklów takie jak w przypadku obiektu statycznego.



Rysunek 6.17: Pierwsza funkcja aktywacji systemu Wienera 1.



Rysunek 6.18: System Wienera 2.

6.7 Złożoność obliczeniowa algorytmu uczenia sieci kołmogorowskiej i algorytmu Sprechera

Na złożoność obliczeniową algorytmu wpływają dwa czynniki:

1. ilość operacji arytmetycznych,
2. rozmiar pamięci niezbędnej do pracy algorytmu.

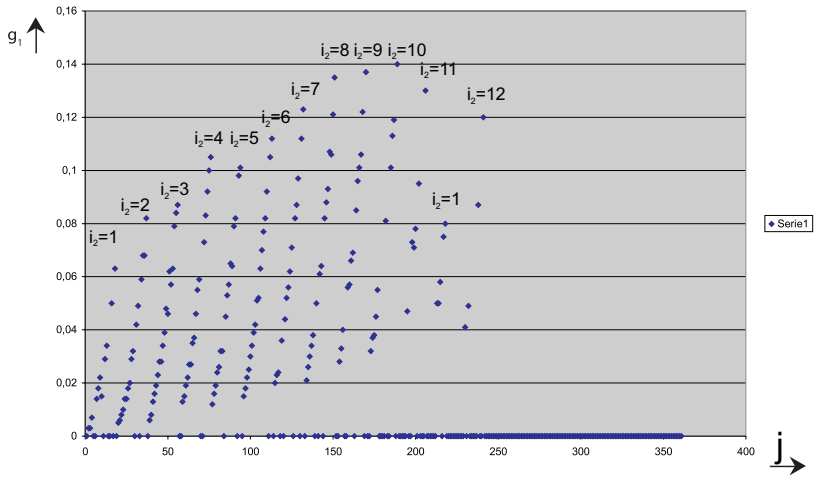
Kluczowym czynnikiem dla algorytmu uczenia indeksowanej sieci kołmogorowskiej jest wymiar pamięci zewnętrznej potrzebnej dla zewnętrznej funkcji aktywacji g. Zależy ona wielomianowo od dokładności aproksymacji, co określono jako problem łatwy oraz zależy wykładniczo od wymiaru funkcji aproksymowanej, co określono jako problem trudny.

Porównanie złożoności algorytmów autora i Sprechera dla przykładu z funkcją aproksymowaną h dwóch zmiennych (p. rozdział 6.4 funkcja expotencjalna) wykazuje, że:

1. algorytm autora wymaga 3 razy 324 mnożeń i 2 razy 324 odejmowań;
2. algorytm Sprechera wymaga: 745 dodawań, 4456 mnożeń, 2 potęgowania.

Złożoność algorytmów dokonano dla "siatki przestrzennej" D'_K (wzór (4.72)) przedziału dziedziny [0,1] podzielonnej na 18 części i funkcji aproksymowanej o dwóch wymiarach.

Przyjmując, że mnożenie jest 10 razy dłuższe od dodawania, a potęgowanie 50 razy dłuższe od dodawania otrzymamy:



Rysunek 6.19: Pierwsza funkcja aktywacji systemu Wienera 2.

1. 10368 jednostek (dodawania) dla algorytmu autora opartego na indeksowanym zmodyfikowanym twierdzeniu Kołmogorowa
2. 45405 jednostek (dodawania) dla algorytmu Sprechera

Tak więc algorytm autora jest 4,5 razy szybszy.

6.8 Porównanie złożoności algorytmu autora i algorytmu A. Janczaka

Porównano algorytm oparty na indeksowanym zmodyfikowanym twierdzeniu Kołmogorowa ze standartowym algorytmem identyfikacji z użyciem sigmoidalnej sieci neuronowej, który jest opisany w rozdziale 6.2.2 (A. Janczak). Porównania dokonano dla systemu Hammersteina z ciągłą nieliniowością (p. rozdział 6.5) i obiektem dynamicznym pierwszego rzędu, z czego wynika że:

1. algorytm autora wymaga 3 krotnie 324 mnożeń i 2 krotnie 324 odejmowań;
2. algorytm A. Janczaka wymaga do obliczania 1 indeksu (odpowiedzialnego za symulacje dynamiki) i 60 wag (odpowiedzialnych za symulacje nieliniowości) 61 operacji obliczania pochodnej, 61 odejmowań i 61 mnożeń w 1 kroku (dla 5 kroków 305) i co najmniej 100 mnożeń, 100 odejmowań, 100 dodawań aby modelować funkcję celu (minimalizującej błąd uczenia sieci).

Przyjmując, że mnożenie jest 10 razy dłuższe od dodawania, a obliczanie pochodnej 200 razy dłuższe od dodawania otrzymamy:

1. 10368 jednostek (dodawania) dla algorytmu autora
2. 35405 jednostek (dodawania) dla algorytmu Janczaka

Niniejsze porównanie wykazuje zalety algorytmu autora, który sprowadza się do mniejszej liczby wykonywanych działań matematycznych, a czas wykonania algorytmu autora jest 3,5 razy krótszy.

6.9 Podsumowanie

Nauczona indeksowana zmodyfikowana kołmogorowska sieć neuronowa umożliwia identyfikacje obiektu statycznego lub dynamicznego oraz potwierdza aproksymacyjne właściwości podanego przez autora zmodyfikowanego twierdzenia Kołmogorowa.

Przeprowadzone badania dotyczą dwóch zmiennych wejściowych. Badania eksperymentalne dla liczby zmiennych wiekszej niż 2 będą podobne, ale ze względu na obszerność wyników zostały pominięte.

Wynikiem identyfikacji obiektów statycznych i dynamicznych jest zestaw funkcji zewnętrznych aktywacji g_q. Zewnętrzne funkcje aktywacji reprezentują nieliniowość systemu, co przedstawino w rozdziale 6.5.

Rozdział 7

Zakończenie

7.1 Oryginalne wyniki naukowe rozprawy

Uzyskano następujące wyniki:

- sformułowano i udowodniono następujące twierdzenia dotyczące sieci neuronowych:
  1. podstawowe zmodyfikowane twierdzenie Kołmogorowa;
  2. indeksowane zmodyfikowane twierdzenie Kołmogorowa;
  3. potęgowe zmodyfikowane twierdzenie Kołmogorowa;
- skonstruowano algorytmy uczenia kołmogorowskich sieci neuronowych wraz z dowodem ich zbieżności dla:
  1. indeksowanej kołmogorowskiej sieci neuronowej z różnymi funkcjami aktywacji;
  2. indeksowanej kołmogorowskiej sieci neuronowej z identycznymi funkcjami aktywacji;
- przeprowadzono badanie eksperymentalne systemów automatyki wraz z wykazaniem zbieżności algorytmów identyfikacji dla:
  1. układu statycznego przedstawionego funkcją eksponencjalną;
  2. układu Hammersteina;
  3. układu Wienera;
- przeanalizowano błąd aproksymacji i warunki zbieżności; algorytmów opartych na twierdzeniach dotyczących sieci kołmogorowskich;
- przeanalizowano i porównano twierdzenia autora (indeksowane i nieindeksowane) dotyczące aproksymacji funkcji wielu zmiennych przez kołmogorowską sieć neuronową; najlepsze do zastosowania jest w tym przypadku zmodyfikowane indeksowane twierdzenie Kołmogorowa;
- przeanalizowano algorytmy Sprechera pod kątem budowy algorytmu uczenia sieci kołmogorowskiej i wykazano mniejszą złożoność algorytmu autora;

- porównano złożoność obliczeniową algorytmu identyfikacji autora i Janczaka;
- dokonano analizy przydatności twierdzenia Kołmogorowa do budowy algorytmu uczenia sieci neuronowych oraz konstrukcji tego algorytmu. Podano prosty i łatwy w obliczeniach algorytm numeryczny uczenia kołmogorowskich sieci neuronowych;
- dokonano analizy aproksymujących właściwości sieci sigmoidalnych i kołmogorowskich. Podano algorytm na aproksymację funkcji ciągłej kołmogorowską siecią neuronową;
- dokonano implementacji algorytmu uczenia sieci kołmogorowskich do identyfikacji systemów statycznych i dynamicznych Hammersteina i Wienera. Wykazano, że sieci kołmogorowskie idealnie nadają się do identyfikacji szerokiej klasy obiektów, w tym systemów z nieliniowością nieróżniczkowalną (z ostrzami).

7.2 Uwagi końcowe

Podana przez autora nowa koncepcja sieci neuronowych typu Kołmogorowa rozszerza wiadomości o teorii sieci neuronowych oraz umożliwia budowę algorytmu numerycznego do m.in. identyfikacji. Twierdzenie Kołmogorowa o reprezentacji funkcji wielu zmiennych zostało zamienione na twierdzenie o aproksymacji przez dokonaną modyfikację twierdzenia. Zaletą prezentowanej zmodyfikowanej sieci kołmogorowskiej jest prostota jej realizacji i niski nakład obliczeń przy uczeniu.

Porównanie sieci kołmogorowskiej z klasyczną siecią sigmoidalną wykazuje następujące cechy:

1. sieci kołmogorowskie mają prosty algorytm uczenia i nie wymagają stosowania metod gradientowych i propagacji wstecznej,
2. sieci kołmogorowskie idealnie nadają się do identyfikacji szerokiej klasy obiektów w systemach Hammersteina i Wienera,
3. w sieciach sigmoidalych uczenie polega na określeniu wag, natomiast w sieciach kołmogorowskich uczenie polega na określaniu zewnętrznych funkcji aktywacji,
4. sieci kołmogorowskie mają sztywną strukturę i składają się z dwóch warstw i nie mogą mieć sprzężeń zwrotnych, natomiast sieci sigmoidalne mają strukturę bardziej elastyczną o dowolnej liczbie warstw.

Rozdział 8

Bibliografia

[1] Kołmogorow A. N. : On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Dokl. Akad. Nauk ZSRR, vol. 114, s. 953-956, 1957.

[2] Michalkiewicz J. i inni :An algorithm for training a neural network based on Kolmogorovs modified theorem. str. 1339-1342 MMAR Międzyzdroje 2004 r.

[3] Osowski S. : Sieci neuronowe w ujęciu algorytmicznym. WNT Warszawa 1996 r..

[4] Hornik K. , Stinchcombe M. , White H. : Multilayer feedforward networks are universal approximators. Neural Networks, vol. 2, pp. 359-366, 1989.

[5] Cybenko G. : Approximation by superposition of a sigmoidal function. Mathematics of Control, Signals, and Systems, vol. 2, pp. 303-314, 1989.

[6] Igielnik B. : Kolmogorovs Spline Network. IEEE Transactions on Neural Networks, vol. 14, No 4, July 2003.

[7] Sprecher D. A. : A numerical implementation of Kolmogorovs superpositions. Neural Networks, vol. 9, pp. 765-772, 1996.

[8] Sprecher D. A. : A numerical implementation of Kolmogorovs superpositions II Neural Networks, vol. 10, pp. 447-457, 1997.

[9] Janczak A. : Sieci neuronowe w identyfikacji systemów Wienera i Hammersteina. Biocybernetyka i inżynieria biomedyczna. Tom 6. Sieci neuronowe. 2000 r.

[10] Kosiński R. A. : Sztuczne sieci neuronowe - dynamika nieliniowa i chaos. WNT Warszawa 2002 r.

[11] Tadeusiewicz R. : Problemy biocybernetyki. PWN Warszawa 1991r.

[12] Skubalska-Rafajlowicz E. : Krzywe wypełniające w rozwiązywaniu wielowymiarowych problemów decyzyjnych. Oficyna wydawnicza Politechniki Wrocławskiej Wrocław 2001.

[13] Sprecher D. A. : Space-filling curves and Kolmogorov superposition-based neural networks. Neural Networks, vol. 15, pp.57-67, 2002.

[14] Devroye, Gyoifi, Lugosi : Probabilistic Theory of Patterrn Recognition. Springer 1996r.

[15] Tadeusiewicz R. :Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa 1993r.

[16] Narendra K. S, Parthasarathy K.: Identification and control of dynamical systems using neural networks IEEE Transaction on Neural Networks Vol. 1 pp 4-27, 1990.

[17] Hertz J., Krogh A., Palmer R. G. : Wstęp do teorii obliczeń neuronowych WNT Warszawa 1995.

[18] Wielka Encyklopedia Powszechna PWN Warszawa.

[19] Poradnik matematyczny PWN Warszawa 1978.

[20] Rudin W. Analiza funkcjonalna. Wydawnictwo Naukowe PWN.

[21] Rudin W. Podstawy analizy matematycznej. Wydawnictwo Naukowe PWN.

[22] Barron A. R. Universal approximation bounds for superpositions of a sigmoidal function. IEEE Transactions on Information Theory, vol. 39, No 3, May 1993.

[23] Lorentz G. G. The 13th problem of Hilbert. Proceeding of Symposia in Pure Mathematics Vol 28 ,1976.

[24] Białasiewicz J. T.: Falki i aproksymacje. WNT Warszawa 2000 r.

[25] Białko M.: Sztuczna inteligencja i elementy hybrydowych systemów ekspertowych. Wydawnictwo Uczelniane Politechniki Koszalińskiej Koszalin 2005 r.

[26] Gyorfi L., Kohler M., Krzyżak A., Walk H. :A distribution-free theory of nonparametric regresion. Plik pdf.

[27] Kurkova V. :Kolmogorov's theorem and multilayer neural networks. Neural Networks Vol 5 str 501-506, 1992.

[28] Fichtenholz G. M.: Rachunek różniczkowy i całkowy tom 3 PWN Warszawa 1980 r.

[29] Engelking R, Sieklucki K :Wstęp do topologii. PWN Warszawa 1986.

[30] Michalkiewicz J (2005) Kolmogorov's neural network in the identification of Hammerstein system. str. 915-918 MMAR 2005 Miedzyzdroje.

[31] Bubnicki Z. : Identyfikacja obiektów sterowania. PWN Warszawa 1974 r.

[32] Tadeusiewicz R. : Wstęp do sieci neuronowych. Biocybernetyka i inżynieria biomedyczna. Tom 6. Sieci neuronowe. 2000 r.

[33] Papadimitriou C.,H. :Złożoność obliczeniowa. WNT Warszawa 2002 r.

[34] Michalkiewicz J.: New concept of Kolmogorov's Neural Networks. str. 77-82. Artificial Intelligence and Soft Computing. Exit Warszawa 2006 r.

[35] Michalkiewicz J.: Polynomial Kolmogorov's Neural Network in the identification of Wiener systems. str. 691-694. MMAR Szczecin 2007 r.

[36] Michalkiewicz J.: Zmodyfikowane twierdzenie Kołmogorowa. Matematyka stosowana vol. 10/51 str. 23-38. Warszawa 2009.

[37] Michalkiewicz J.: Modified Kolmogorov's Neural Network in the identification of Hammerstein and Wiener Systems, IEEE Transactions on Neural Networks and Learnining Systems, vol. 23, no. 4, April 2012, str. 657-662.

[38] Jakóbczak D., Kosiński W.: Hurwitz-Radon operator in monochromatic medical image reconstruction, Journal of Medical Informatics and Technologies, vol. 11/2007, str. 69-78.

[39] Billings S.,A., Fakhouri S.,Y.: Identification of systems containing linear dynamic and static nonlinear elements. Automatica, vol. 18, no. 1, str. 15-26.

[40] Gallman P.,G.: A comparision of two Hammerstein model identification algorithms. IEEE Transaction Automatic Control, vol. 21, no. 1, str. 124-126.

[41] Greblicki W.: Nonparametric approach to Wiener system identification. IEEE Transaction Circuits System I: Fundam. Theory Appl., vol. 44, no. 6, str. 538-545.

[42] Mhaskar H., N., Micchelli C., A.: Degree of approximation by neural and translation network with a single hidden layer. Advances in Applied Mathematics, vol. 16, str. 151-183.

[43] Chui C., Li X.: Approximation by ridge functions and neural networks with one hidden layer. Journal of Approximation Theory, vol. 70, str. 131-141.

[44] Leshno M., Lin V.,Y., Pinkus A., Schocken S.: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Networks, vol. 6, str. 861-867.

[45] Belda I.: Umysł, maszyny i matematyka. RBA 2012 r.

[46] Jankowscy J. i M.: Przegląd metod i algorytmów numerycznych. WNT Warszawa 1988 r.