mathematics

MDPI

Article

A Neural Network Approximation Based on a Parametric Sigmoidal Function

Beong In Yun

Department of Mathematics, Kunsan National University, Gunsan 54150, Korea; paulllyun@gmail.com

Received: 12 February 2019; Accepted: 11 March 2019; Published: 14 March 2019

Abstract: It is well known that feed-forward neural networks can be used for approximation to functions based on an appropriate activation function. In this paper, employing a new sigmoidal function with a parameter for an activation function, we consider a constructive feed-forward neural network approximation on a closed interval. The developed approximation method takes a simple form of a superposition of the parametric sigmoidal function. It is shown that the proposed method is very effective in approximation of discontinuous functions as well as continuous ones. For some examples, the availability of the presented method is demonstrated by comparing its numerical results with those of an existing neural network approximation method. Furthermore, the efficiency of the method in extended application to the multivariate function is also illustrated.

Keywords: feed-forward neural network; activation function; parametric sigmoidal function; quasi-interpolation

MSC: 65D15; 92B20; 41A20

1. Introduction

Cybenko [1] and Funahashi [2] proved that any continuous function can be uniformly approximated on a compact set $I \subset \mathbf{R}^n$ by the feed-forward neural networks (FNN) in the form of

$$F_N(x) = \sum_{k=0}^{N} \alpha_k \, \sigma\left(\omega_k \cdot x + \theta_k\right), \qquad x \in I, \tag{1}$$

where σ is called an activation function, $\omega_k \in \mathbf{R}^n$ are weights, $\theta_k \in \mathbf{R}$ are thresholds, and $\alpha_k \in \mathbf{R}$ are coefficients. It is called the universal approximation theorem. Moreover, Hornik et al. [3] showed that any measurable function can be approximated on a compact set by the form of the FNN. Some constructive approximation methods by the FNN were developed in the literature [4–7]. Other examples of the function approximation by the FNN can be found in the works of Cao et al. [8], Chui and Li [9], Ferrari and Stengel [10], and Suzuki [11]. Particularly, the activation function σ is a basic architecture of the neural networks because it imports non-linear properties into the networks. This allows the artificial neural networks to learn from complicated non-linear mappings between inputs in general.

In this paper, aiming efficient approximation to the data obtained from continuous or discontinuous functions on a closed interval, we develop a feed-forward neural network approximation method based on a sigmoidal activation function. First, in the following section, we propose a parametric sigmoidal function $\sigma^{[m]}$ of the form (6) for an activation function. In Section 3 we construct an approximation formula $S_N^{[m]} f(x)$ in (19) based on the proposed sigmoidal function $\sigma^{[m]}$. It is shown that $S_N^{[m]} f(x)$ approximates every given data with error $O\left(\delta^m\right)$, $0 < \delta < 1$, for the parameter m large enough. This implies the so-called quasi-interpolation property of the presented FNN approximation.

Furthermore, in order to better the interpolation errors near the end-points of the given interval, a correction formula (27) is introduced in Section 4. The efficiency of the presented FNN approximation is demonstrated by the numerical results for the data sets extracted from continuous and discontinuous functions. The aforementioned efficiency means that the proposed method requires less neurons to reach similar or lower error levels than the compared FNN approximation method using the conventional logistic function.

In addition, an extended FNN approximation formula for two variable functions is proposed in Section 5 with some numerical examples showing the superiority of the presented FNN approximation method.

2. A Parametric Sigmoidal Function

The role of the activation function in the artificial neural networks is to introduce non-linearity of the input data into the output of the neural network. One of the useful activation functions commonly used in practice is the sigmoidal function σ having the property below.

$$\sigma(t) \to \begin{cases} 0 & \text{as} \quad t \to -\infty \\ 1 & \text{as} \quad t \to \infty \end{cases} \tag{2}$$

For example, two traditional sigmoidal functions are

(i) Heaviside function:

$$\sigma_H(t) \;=\; \begin{cases} 0\,, & t < 0 \\ 1\,, & t > 0 \end{cases} \tag{3}$$

(ii) Logistic function:

$$\sigma_L(t) \;=\; \frac{1}{1+e^{-t}} \;=\; \frac{1}{2}\left\{1+\tanh(t/2)\right\}, \qquad -\infty < t < \infty \tag{4}$$

We recall the following approximation theorem shown in the literature [6].

Theorem 1. (Costarelli and Spigler [6]) *For a bounded sigmoidal function σ and a function $f \in C[a,b]$ let $G_N f$ be a neural network approximation to f of the form*

$$G_N f(x) = f(x_0)\sigma\left(\omega(x-x_{-1})\right) \;+\; \sum_{k=1}^{N}\left\{f(x_k)-f(x_{k-1})\right\}\sigma\left(\omega(x-x_k)\right) \tag{5}$$

for $x \in [a,b]$, $h=(b-a)/N$, and $x_k = a+kh$ $(k=-1,0,1,\cdots,N)$. Then for every $\epsilon > 0$ there exists an integer $N > 0$ and a real number $\omega > 0$ such that

$$\|G_N f - f\|_\infty < \epsilon.$$

Sigmoidal functions have been used in various applications including the artificial neural networks (See the literature [12–17]). In this work we employ an algebraic type sigmoidal function, containing a parameter $m > 0$, as follows.

$$\sigma^{[m]}(t) \;=\; \begin{cases} 0, & t < -L \\ \dfrac{(L+t)^m}{(L+t)^m + (L-t)^m}, & |t| \le L \\ 1, & t > L \end{cases} \tag{6}$$

for a fixed $L > 0$. This function has the following properties.

(A1) $\sigma^{[m]}$ is strictly increasing over $[-L, L]$ and $\sigma^{[m]} \in C^{\infty}(-L, L) \cap C^{m-1}(\mathbf{R})$ for an integer $m \geq 1$. In addition, referring to the literature [12], we can see that the Hausdorff distance d between the heaviside function σ_H and the presented sigmoidal function $\sigma^{[m]}$ satisfies

$$\left(\frac{L+d}{L-d}\right)^{m} = \frac{1}{d} - 1, \qquad 0 < d < \min\left\{\frac{1}{2}, L\right\}. \tag{7}$$

That is, $m = O\left(\frac{\ln(1/d)}{\ln(1+d)}\right)$ for d small enough.

(A2) For m large enough $\sigma^{[m]}$ has the asymptotic behavior

$$\sigma^{[m]}(t) = \begin{cases} O\left(\theta(t)^m\right), & -L \leq t < 0 \\ 1 + O\left(\theta(t)^m\right), & 0 < t \leq L \end{cases} \tag{8}$$

where

$$\theta(t) = \left(\frac{L-t}{L+t}\right)^{\mathrm{sgn}(t)}, \tag{9}$$

satisfying $0 \leq \theta(t) < 1$ for all $t \in [-L, L] \setminus \{0\}$. In addition, for any integer $m \geq 2$

$$\frac{d^j}{dx^j}\sigma^{[m]}(\pm L) = 0, \qquad j = 1, 2, \cdots, m-1. \tag{10}$$

(A3) For every $m > 0$

$$\sigma^{[m]}(-t) + \sigma^{[m]}(t) = 1, \qquad t \in \mathbf{R} \tag{11}$$

with $\sigma^{[m]}(0) = \frac{1}{2}$.

3. Constructing a Neural Network Approximation

Suppose for a real valued function $f(x)$, $a \leq x \leq b$, a set of data

$$\{f_k = f(x_k) \mid k = 0, 1, 2, \cdots, N\}$$

is given, where $N \geq 2$ is an integer and x_k are nodes on the interval $[a, b]$. For simplicity, we assume equally spaced nodes as

$$x_k = a + k \cdot h, \qquad h = (b-a)/N. \tag{12}$$

We can observe that, for sufficiently large m, the function $\sigma^{[m]}$ with $L = b - a$ in (6) satisfies

$$\sigma^{[m]}(t) \approx \begin{cases} 0, & t < 0 \\ 1, & t > 0 \end{cases} \tag{13}$$

due to the property (A2).

Moreover, noting that $\sigma^{[m]}$ is an increasing function as mentioned in (A1), we can see that

$$\sigma^{[m]}(t) < \frac{1}{N}, \quad \text{for all} \quad t < \left(\sigma^{[m]}\right)^{-1}\left(\frac{1}{N}\right) = -L \cdot \frac{(N-1)^{1/m} - 1}{(N-1)^{1/m} + 1} \tag{14}$$

and from the property (A3)

$$1 - \sigma^{[m]}(t) < \frac{1}{N}, \quad \text{for all} \quad t > L \cdot \frac{(N-1)^{1/m} - 1}{(N-1)^{1/m} + 1}. \tag{15}$$

To find a lower bound of the parameter m we set $L \cdot \frac{(N-1)^{1/m}-1}{(N-1)^{1/m}+1} = h(= L/N)$. Then we have the lower bound $m = m^*$, satisfying this equation, as

$$m^* = \frac{\log N - 1}{\log \frac{N+1}{N-1}}. \tag{16}$$

That is, for every $m > m^*$ it follows that

$$\sigma^{[m]}(t) < \frac{1}{N}, \quad \text{for all} \quad t < -h \tag{17}$$

and

$$1 - \sigma^{[m]}(t) < \frac{1}{N}, \quad \text{for all} \quad t > h\,. \tag{18}$$

The lower bound m^* given in (16) will be used for a threshold of the parameter m in the numerical implementation of the proposed neural network approximation later.

Referring to the above features of $\sigma^{[m]}$ in (13), (17) and (18), we propose a superposition of $\sigma^{[m]}$ to approximate the given data $\{f_k = f(x_k) \mid k = 0, 1, 2, \cdots, N\}$ as follows.

$$S_N^{[m]} f(x) = f_0 + \sum_{k=1}^{N} (f_k - f_{k-1})\, \sigma^{[m]}(x - \overline{x}_k)\,, \qquad x_0 \le x \le x_N, \tag{19}$$

where $\overline{x}_k = (x_{k-1} + x_k)/2 = x_k - h/2$.

We can see that $S_N^{[m]} f(x)$ interpolates $f(x)$ at $N+1$ nodes, approximately, as implied in the following theorem. Thus we call $S_N^{[m]} f(x)$ a quasi-interpolation of $f(x)$.

Theorem 2. *The FNN $S_N^{[m]} f(x)$ with m large enough as defined in (19) satisfies*

$$S_N^{[m]} f(x_j) = f_j + C_j f''(\xi_j)\, h^2\, \delta^m\,, \qquad 1 \le j \le N-1 \tag{20}$$

for some $\xi_j \in (x_{j-1},\, x_{j+1})$, $0 < \delta < 1$ and a constant C_j. Moreover,

$$S_N^{[m]} f(x_0) = f_0 + C_0 f'(\xi_0)\, h\, \delta^m, \quad S_N^{[m]} f(x_N) = f_N + C_N f'(\xi_N)\, h\, \delta^m,$$

for some $\xi_0 \in (x_0,\, x_1)$, $\xi_N \in (x_{N-1},\, x_N)$ and constants C_0, C_N.

Proof. Since $\sigma^{[m]}$ is an increasing function and it satisfies the asymptotic behaviour in (8), for each $1 \le j \le N-1$ with m large enough, we have

$$\begin{aligned} S_N^{[m]} f(x_j) &\sim f_{j-1} + (f_j - f_{j-1})\, \sigma^{[m]}\left(\frac{h}{2}\right) + (f_{j+1} - f_j)\, \sigma^{[m]}\left(-\frac{h}{2}\right) \\ &= f_j \left\{1 - \sigma^{[m]}\left(-\frac{h}{2}\right)\right\} + f_{j-1}\sigma^{[m]}\left(-\frac{h}{2}\right) + (f_{j+1} - f_j)\, \sigma^{[m]}\left(-\frac{h}{2}\right) \\ &= f_j + \{f_{j-1} - 2f_j + f_{j+1}\}\, \sigma^{[m]}\left(-\frac{h}{2}\right). \end{aligned}$$

The second equation above results from the relation $\sigma^{[m]}\left(\frac{h}{2}\right) = \left\{1 - \sigma^{[m]}\left(-\frac{h}{2}\right)\right\}$ based on the property (A3). Denoting by Δ and Δ^2 the first and the second forward difference operators, respectively, and using the function $\theta(t)$ defined in (9), we have

$$S_N^{[m]} f(x_j) = f_j + \Delta^2 f_{j-1}\, O\left(\theta\left(-\frac{h}{2}\right)^m\right).$$

Since $\Delta^2 f_{j-1} = f''(\xi_j) h^2$ for some $\xi_j \in (x_{j-1}, x_{j+1})$, setting $\delta = \theta\left(-\frac{h}{2}\right)$, we have the formula (20).

On the other hand, for $x = x_0$ and m large enough

$$S_N^{[m]} f(x_0) \sim f_0 + (f_1 - f_0)\,\sigma^{[m]}\left(-\frac{h}{2}\right) = f_0 + \Delta f_0\, O\left(\theta\left(-\frac{h}{2}\right)^m\right).$$

Since $\Delta f_0 = f'(\xi_0) h$ for some $\xi_0 \in (x_0, x_1)$, we have

$$S_N^{[m]} f(x_0) = f_0 + C_0 f'(\xi_0)\, h\, \delta^m$$

for a constant C_0. For $x = x_N$ and m large enough

$$\begin{aligned} S_N^{[m]} f(x_N) &\sim f_{N-1} + (f_N - f_{N-1})\,\sigma^{[m]}\left(\frac{h}{2}\right) \\ &= f_{N-1} + (f_N - f_{N-1})\left\{1 - \sigma^{[m]}\left(-\frac{h}{2}\right)\right\} \\ &= f_N - (f_N - f_{N-1})\,\sigma^{[m]}\left(-\frac{h}{2}\right) \\ &= f_N + \Delta f_{N-1}\, O\left(\theta\left(-\frac{h}{2}\right)^m\right). \end{aligned}$$

Since $\Delta f_{N-1} = f'(\xi_N) h$ for some $\xi_N \in (x_{N-1}, x_N)$, we have

$$S_N^{[m]} f(x_N) = f_N + C_N f'(\xi_N)\, h\, \delta^m$$

for a constant C_N. Thus the proof is completed. □

Theorem 2 implies that, for N fixed(i.e., h fixed), approximation errors of $S_N^{[m]} f(x)$ at every nodes can be accelerated by increasing the value of the parameter m.

The sum $S_N^{[m]} f(x)$ in (19) can be written by

$$\begin{aligned} S_N^{[m]} f(x) = f_0 \left\{1 - \sigma^{[m]}(x - \overline{x}_1)\right\} + f_N \sigma^{[m]}(x - \overline{x}_N) \\ + \sum_{k=1}^{N-1} f_k \left\{\sigma^{[m]}(x - \overline{x}_k) - \sigma^{[m]}(x - \overline{x}_{k+1})\right\}. \end{aligned} \tag{21}$$

Using a function $\psi^{[m]}$ defined as

$$\psi^{[m]}(t) = \sigma^{[m]}(t + h/2) - \sigma^{[m]}(t - h/2), \qquad -L \le t \le L, \tag{22}$$

with $L = b - a$, satisfying $0 \le \psi^{[m]}(t) \le 1$ for all t, we may rewrite $S_N^{[m]} f(x)$ by

$$S_N^{[m]} f(x) = f_0 \left\{1 - \sigma^{[m]}(x - \overline{x}_1)\right\} + f_N \sigma^{[m]}(x - \overline{x}_N) + \sum_{k=1}^{N-1} f_k \psi^{[m]}(x - x_k) \tag{23}$$

for $x_0 \le x \le x_N$. In fact, it follows that

$$\psi^{[m]}(x - x_k) = \sigma^{[m]}(x - \overline{x}_k) - \sigma^{[m]}(x - \overline{x}_{k+1}), \qquad 1 \le k \le N - 1. \tag{24}$$

The formula (23) is a form of the feed-forward neural networks based on the activation function $\psi^{[m]}$ with constant weights $w_k = 1$ and thresholds x_k.

Under the assumption that m is large enough, the proposed quasi-interpolation $S_N^{[m]}f(x)$ in (23) has the following properties:

(B1) Since $1-\sigma^{[m]}(x-\overline{x}_1) \approx \psi^{[m]}(x-x_0)$ and $\sigma^{[m]}(x-\overline{x}_N) \approx \psi^{[m]}(x-x_N)$ over the interval $[a,b]=[x_0,x_N]$, it follows that

$$S_N^{[m]}f(x) \approx \sum_{k=0}^{N} f_k \psi^{[m]}(x-x_k), \qquad x_0 \le x \le x_N. \tag{25}$$

(B2) For each $k=0,1,2,\cdots,N$,

$$\psi^{[m]}(x_j-x_k) \approx \begin{cases} 1, & j=k \\ 0, & j\neq k \end{cases}$$

and

$$\psi^{[m]}(\overline{x}_k-x_k) = \psi^{[m]}(\overline{x}_{k+1}-x_k) = \psi^{[m]}\left(\pm\frac{h}{2}\right) \approx \frac{1}{2}.$$

Graphs of the activation functions, $\sigma_k^{[m]}(x) := \sigma^{[m]}(x-\overline{x}_k)$ and $\psi_k^{[m]}(x) := \psi^{[m]}(x-x_k)$ shown in Figure 1 illustrate the intuition of the construction of the presented quasi-interpolation $S_N^{[m]}f(x)$. In addition, Figure 2 includes the graphs of $\psi_k^{[m]}(x)$ with respect to the values $m=1,2,4,16$, which shows that $\psi_k^{[m]}(x)$ becomes flatter near the node x_k and far from the node as the parameter m goes higher.

It is well known that the interpolants for continuous functions are guaranteed to be good if and only if the *Lebesgue constants* are small [15]. Regarding the formula (25) as an interpolation with equispaced points $\{x_k\}_{k=0}^N$, its Lebesgue function satisfies

$$\lambda_N(x) := \sum_{k=0}^{N} \left|\psi^{[m]}(x-x_k)\right| = \sum_{k=0}^{N} \psi^{[m]}(x-x_k) \approx 1$$

for all x, and thus the corresponding Lebesgue constant becomes $\Lambda_N = \|\lambda_N(x)\|_\infty \approx 1$. Noting that for the polynomial interpolation, the Lebesgue constant grows exponentially such as $\Lambda_N \sim \frac{2^{N+1}}{eN\log N}$ as $N\to\infty$, we may expect that $S_N^{[m]}f(x)$ will be better than the polynomial interpolation in approximation to any continuous function, at least.

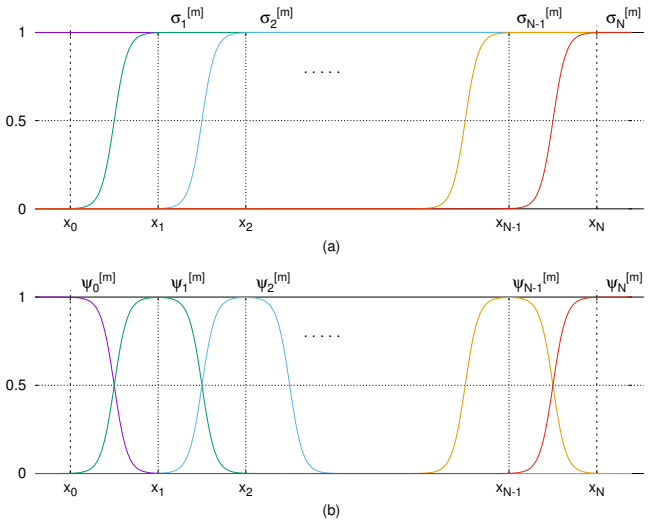


Figure 1. Graphs of the sigmoidal functions $\sigma_k^{[m]}(x)$ in (**a**) and those of $\psi_k^{[m]}(x)$ in (**b**).

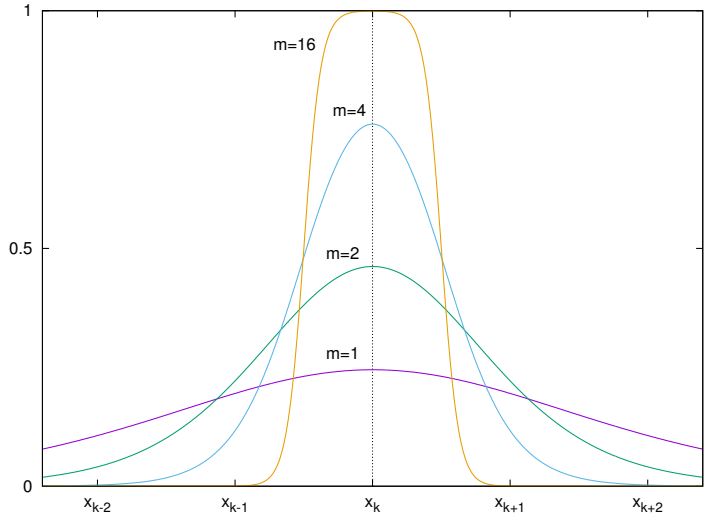


Figure 2. Graphs of $\psi_k^{[m]}(x)$ for each $m = 1, 2, 4$, and 16.

4. Correction Formula

In order to improve the interpolation errors near the end-points of the given interval, that is, to make the formula (20) in Theorem 2 hold for all $0 \le j \le N$, we employ two values at the points $x_{-1} := x_0 - h = a - h$ and $x_{N+1} := x_N + h = b + h$ defined as

$$f_{-1} = 2f_0 - f_1\,, \qquad f_{N+1} = 2f_N - f_{N-1}. \tag{26}$$

Using these additional data, we define a correction formula of (23) as

$$S_N^{[m]} f(x) \;=\; f_{-1}\left\{1 - \sigma^{[m]}\left(x - \overline{x}_0\right)\right\} \;+\; f_{N+1}(x)\sigma^{[m]}\left(x - \overline{x}_{N+1}\right) \;+\; \sum_{k=0}^{N} f_k \psi^{[m]}\left(x - x_k\right). \tag{27}$$

To explore the availability of the proposed approximation method (27), we consider the following examples which were employed in the literature [6].

Example 1. *A smooth function on the interval* $[-5, 5]$.

$$f_1(x) = \left(2 + \cos^2 x\right)\sin x + 2x + \frac{x^2}{8} + 4\,, \qquad -5 \le x \le 5,$$

Example 2. *A function with jump-discontinuities.*

$$f_2(x) = \begin{cases} \frac{4}{x^2-2}\,, & x < -2 \\ -3\,, & -2 \le x < 0 \\ \frac{5}{2}\,, & 0 \le x < 2 \\ \frac{3x+2}{x^3-1}\,, & x \ge 2\,, \end{cases}$$

We compare the results of the presented method with those of the existing neural network approximation method (5) using the activation function $\sigma = \sigma_L$ in (4). In the literature [6], it was proved that Theorem 1 holds if the weight ω is chosen such as

$$\omega > \frac{N}{L}\log(N-1)\,, \qquad L = b - a\,. \tag{28}$$

In practice, we have used $\omega = N^2/L$ in implementation of the existing FNN $G_N f(x)$ in (5) for the examples above. The high level software, *Mathematica*(V.10) has been used as a programming tool throughout the numerical performance for the examples.

For the smooth function f_1 in Example 1, approximations of the proposed FNN $S_N^{[m]} f_1(x)$, with small number of neurons ($N = 10$) are shown in Figure 3 with respect to each parameter $m = 10, 15, 20, 30$. The higher the value of m is, the more clearly $S_N^{[m]} f_1(x)$ reveals the so-called quasi-interpolation property as shown in Theorem 2. Moreover, Figure 4 shows errors of $S_N^{[m]} f_1(x)$ with $m = 2N > m^*$, for m^* the lower bound of m as given in (16), compared with errors of $G_N f_1(x)$ for $N = 10, 20, 30, \cdots, 80$. Therein the errors are defined as $\left\| S_N^{[m]} f_1(x) - f_1 \right\|_\infty / \|f_1\|_\infty$ and $\|G_N f_1(x) - f_1\|_\infty / \|f_1\|_\infty$. The figure illustrates that the presented FNN is superior to the existing FNN $G_N f_1(x)$ for continuous test function f_1.

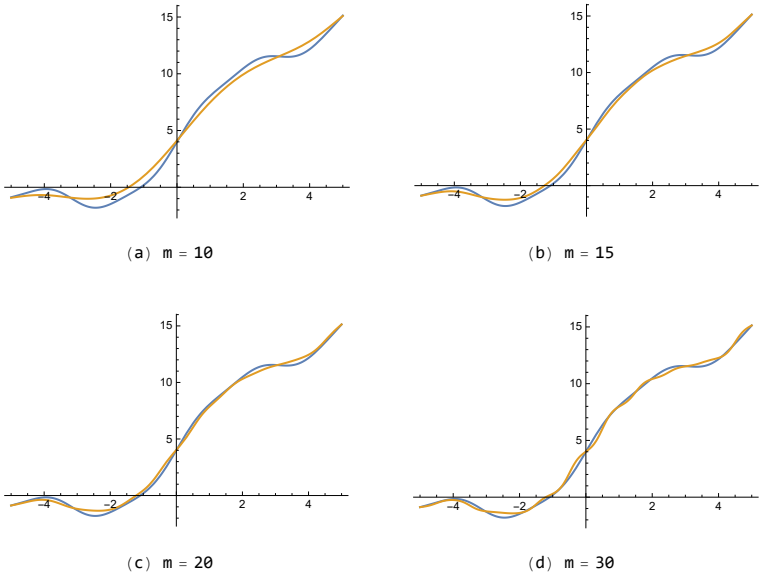


Figure 3. Approximations to $f_1(x)$ by the presented feed-forward neural networks (FNN) $S_N^{[m]} f_1(x)$ with $N = 10$ for each $m = 10, 15, 20, 30$.

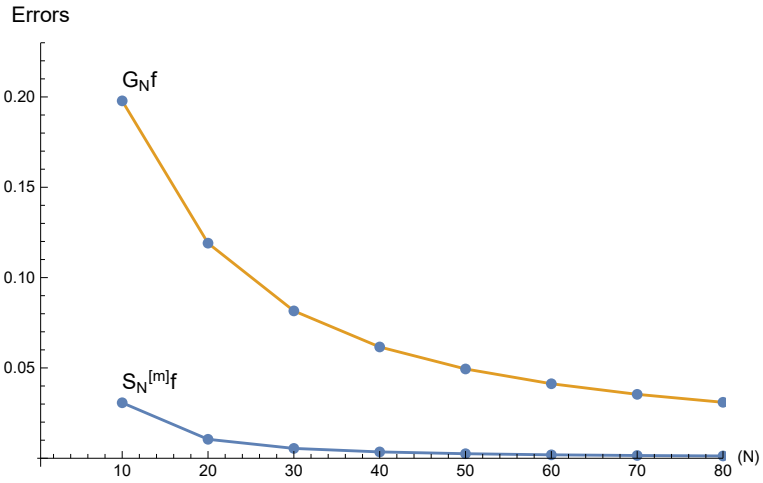


Figure 4. Errors of the presented FNN approximations $S_N^{[m]} f_1(x)$ with $m = 2N$ and the existing FNN approximations $G_N f_1(x)$ for $N = 10, 20, 30, \cdots, 80$.

For the discontinuous function f_2 in Example 2, approximations of the proposed FNN $S_N^{[m]} f_2(x)$, with small number of neurons ($N = 10$), are shown in Figure 5 with respect to each $m = 10, 20, 40, 80$. In addition, approximations of $S_N^{[m]} f_2(x)$ for various values $N = 10, 20, 40, 80$, with $m = 4N$, are also given in Figure 6. One can see that the results of the presented method $S_N^{[m]} f_2(x)$ are better than those of $G_N f_2(x)$ shown in Figure 7. On the other hand, it is noted that the FNN approximations are free from the so-called Gibbs phenomenon, generating wiggles (i.e., overshoots and undershoots) near the jump-discontinuity, which appears inevitably in partial sum approximations composed of the polynomial or trigonometric base functions in general.

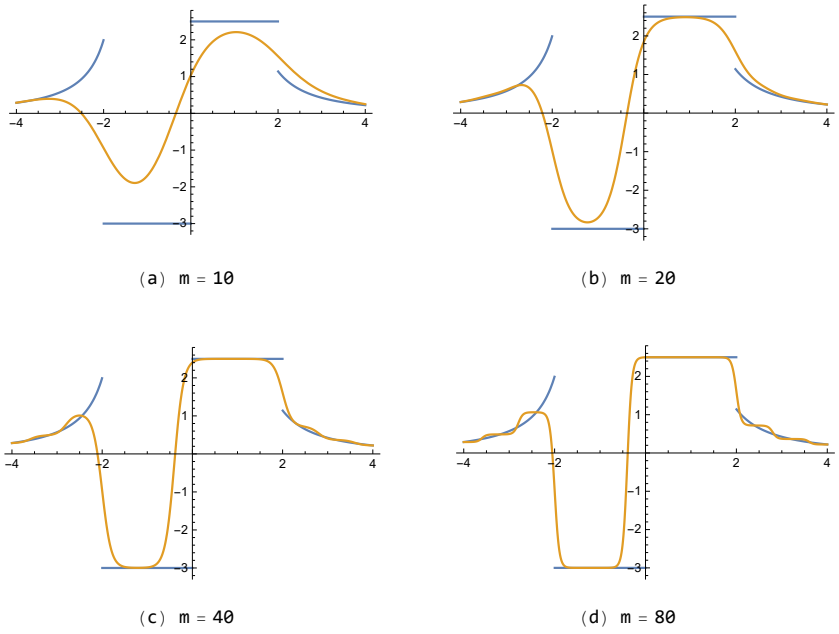


Figure 5. Approximations to $f_2(x)$ by the presented FNN $S_N^{[m]} f_2(x)$ with $N = 10$ for each $m = 10, 20, 40, 80$.

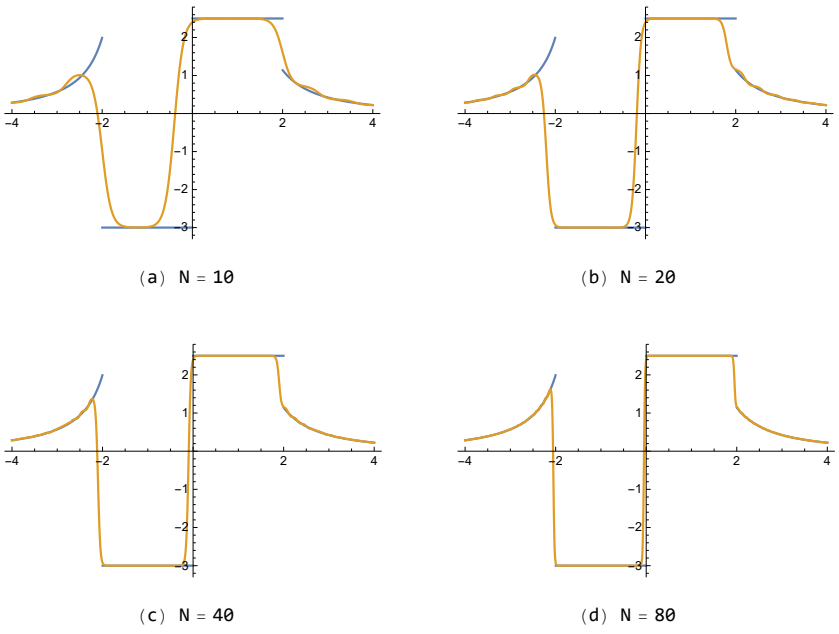


Figure 6. Approximations to $f_2(x)$ by the presented FNN $S_N^{[m]} f_2(x)$ for each $N = 10, 20, 40, 80$ with $m = 4N$.

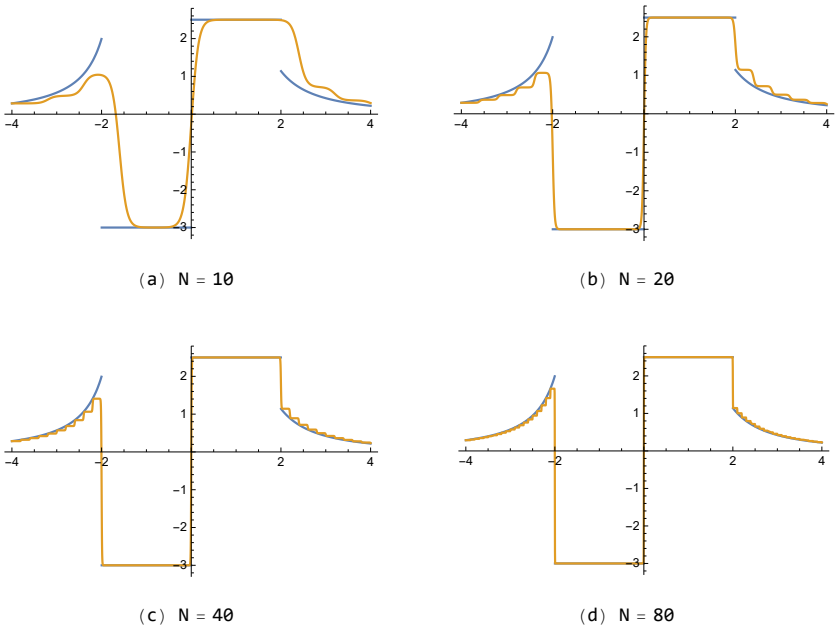


Figure 7. Approximations to $f_2(x)$ by the existing FNN $G_N f_2(x)$ for each $N = 10, 20, 40, 80$.

5. Multivariate Approximation

For simplicity we consider a function of two variables $g(x, y)$ on a region $[a, b] \times [c, d] \subset \mathbf{R}^2$, and assume that a set of data $\{g_{ij} = g(x_i, y_j)\}_{0 \le i,j \le N}$ is given for the nodes

$$x_i = i \cdot h_x, \qquad y_j = j \cdot h_y,$$

where $h_x = (b-a)/N$, $h_y = (d-c)/N$. Set activation functions

$$\psi_{i,j}^{[m]}(x, y) = \sigma^{[m]}\left(\chi_i(x)|x - \overline{x}_i|\right) \cdot \sigma^{[m]}\left(\chi_j(y)|y - \overline{y}_j|\right) \tag{29}$$

for $0 \le i, j \le N$, where $\overline{\xi}_k = (\xi_{k-1} + \xi_k)/2$,

$$\chi_k(\xi) = \begin{cases} 1, & \xi_{k-1} < \xi \le \xi_k \\ -1, & \text{otherwise} \end{cases} \tag{30}$$

and $\sigma^{[m]}$ is the parametric sigmoidal function in (6). Then, referring to the formula (25) under the assumption that m is large enough, we define an extended version of the FNN approximation to g as

$$S_N^{[m]} g(x, y) = \sum_{i=0}^{N} \sum_{j=0}^{N} g_{ij} \psi_{i,j}^{[m]}(x, y). \tag{31}$$

To testify the efficiency of the presented method (31), we choose functions of two variables below. In the numerical implementation for the examples the software, *gnuplot*(V.5) was used as it is rather fast for evaluating and graphing on two dimensional region.

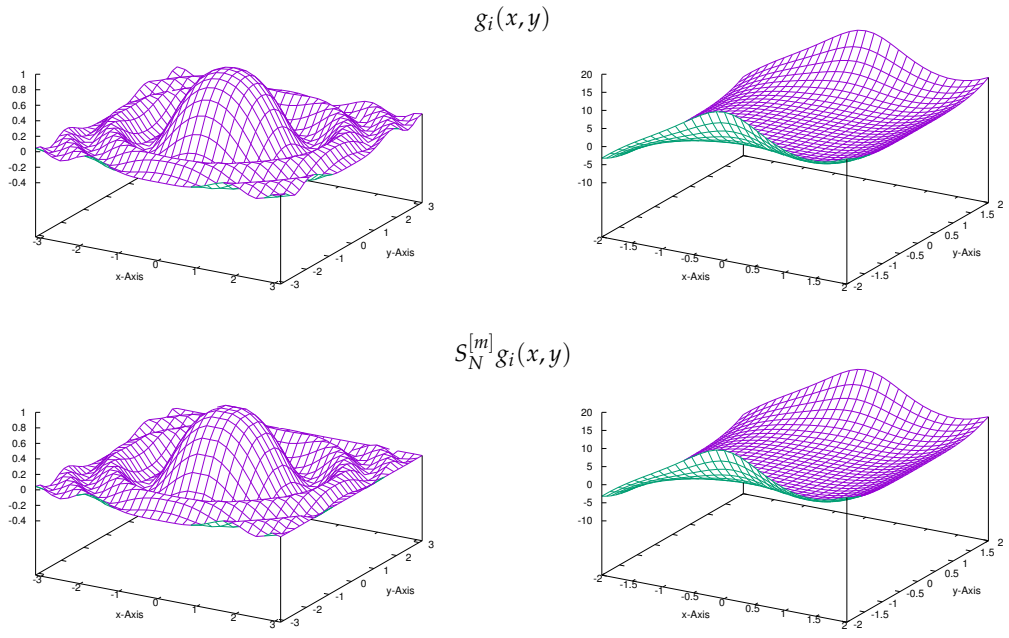
Example 3.

$$g_1(x, y) = \frac{\sin(x^2 + y^2 + 1)}{x^2 + y^2 + 1}, \qquad -\pi \le x, y \le \pi. \tag{32}$$

Example 4.

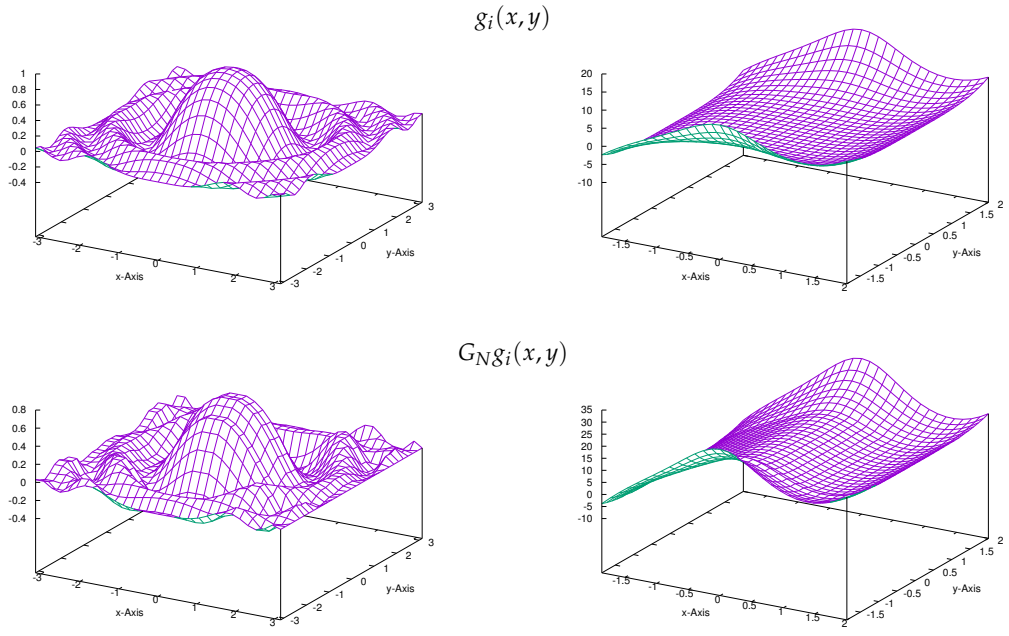
$$g_2(x, y) = \frac{x^5 + y^4}{x^2 + 1}, \qquad -2 \le x, y \le 2. \tag{33}$$

Figure 8 shows the approximations of the presented method $S_N^{[m]}g_i(x,y)$ to the test functions $g_i(x,y)$, $i = 1,2$, for $N = 30$ with $m = 120$. We can see that $S_N^{[m]}g_i(x,y)$ approximates $g_i(x,y)$ properly over the whole region, while the existing method $G_N g_i(x,y)$ given in the literature [6] produces considerable errors as shown in Figure 9.



(**a**) $i = 1$ (**b**) $i = 2$

Figure 8. Test functions $g_i(x,y)$(: upper row), $i = 1,2$, and their approximations by the presented FNN $S_N^{[m]}g_i(x,y)$(: lower row) for $N = 30$ with $m = 120$.



(**a**) $i = 1$ (**b**) $i = 2$

Figure 9. Test functions $g_i(x,y)$(: upper row), $i = 1,2$, and their approximations by the existing FNN approximations $G_N g_i(x,y)$(: lower row) for $N = 30$.

6. Conclusions

In this work we proposed an FNN approximation method based on a new parametric sigmoidal activation function $\sigma^{[m]}$. It has been shown that the presented method with the parameter m large enough has a feature of the quasi-interpolation at the given nodes. As a result, we can note that the presented method is better than the existing FNN approximation method as demonstrated by the numerical results for several examples of univariate continuous and discontinuous functions. Additionally, the availability of the method in extended application to the multivariate function was illustrated.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2017 R1A2B4007682).

Conflicts of Interest: The author declares no conflict of interest.

References

1. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signal.* **1989**, *2*, 303–314. [CrossRef]
2. Funahashi, K.I. On the approximate realization of continuous mappings by neural networks. *Neural Netw.* **1989**, *2*, 183–192. [CrossRef]
3. Hornik, K.; Stinchcombe, M.; White, H. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networs. *Neural Netw.* **1990**, *3*, 551–560. [CrossRef]
4. Barron, A.R. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. Inform. Theory* **1993**, *39*, 930–945. [CrossRef]
5. Chen, Z.; Cao, F. The approximation operators with sigmoidal functions. *Comput. Math. Appl.* **2009**, *58*, 758–765. [CrossRef]
6. Costarelli, D.; Spigler, R. Constructive approximation by superposition of sigmoidal functions. *Anal. Theory Appl.* **2013**, *29*, 169–196.
7. Hahm, N.; Hong, B.I. An approximation by neural networks with a fixed weight. *Compu.t Math. Appl.* **2004**, *47*, 1897–1903. [CrossRef]
8. Cao, F.L.; Xie, T.F.; Xu, Z.B. The estimate for approximation error of neural networks: A constructive approach. *Neurocomputing* **2008**, *71*, 626–630. [CrossRef]
9. Chui, C.K.; Li, X. Approximation by ridge functions and neural networks with one hidden layer. *J. Approx. Theory* **1992**, *70*, 131–141. [CrossRef]
10. Ferrari, S.; Stengel, R.F. Smooth Function Approximation Using Neural Networks. *IEEE Trans. Neural Netw.* **2005**, *16*, 24–38. [CrossRef] [PubMed]
11. Suzuki, S. Constructive functions-approximation by three-layer artificial neural networks. *Neural Netw.* **1998**, *11*, 1049–1058. [CrossRef]
12. Kyurkchiev, N.; Markov, S. Sigmoidal functions: Some computational and modelling aspects. *Biomath Commun.* **2014**, *1*. [CrossRef]
13. Markov, S. Cell growth models using reaction schemes: Batch cultivation. *Biomath* **2013**, *2*. [CrossRef]
14. Prössdorf, S.; Rathsfeld, A. On an integral equation of the first kind arising from a cruciform crack problem. In *Integral Equations and Inverse Problems*; Petkov, V., Lazarov, R., Eds.; Longman: Coventry, UK, 1991; pp. 210–219.
15. Trefethen, L.N. *Approximation Theory and Approximation Practice*; SIAM: Oxford, UK, 2013; pp. 107–115.
16. Yun, B.I. An extended sigmoidal transformation technique for evaluating weakly singular integrals without splitting the integration interval. *SIAM J. Sci. Comput.* **2003**, *25*, 284–301. [CrossRef]
17. Yun, B.I. A smoothening method for the piecewise linear interpolation. *J. Appl. Math.* **2015**, *2015*, 376362. [CrossRef]