Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neurocomputing 57 (2004) 389–409
NEUROCOMPUTING
www.elsevier.com/locate/neucom

# Mini-max initialization for function approximation

Xi Min Zhang[a], Yan Qiu Chen[b], Nirwan Ansari[a,*], Yun Q. Shi[a]

[a] *Department of Electrical & Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA*

[b] *Department of Computer Science and Engineering, Intelligent Information Processing Laboratory, Fudan University, People's Republic of China*

Received 14 March 2003; accepted 31 October 2003

**Abstract**

Neural networks have been successfully applied to various pattern recognition and function approximation problems. However, the training process remains a time-consuming procedure that often gets stuck in a local minimum. The optimum network size and topology are usually unknown. In this paper, we formulate the concept of extrema equivalence for estimating the complexity of a function. Based on this formulation, the optimal network size and topology can be selected according to the number of extrema. Mini-max initialization method is then proposed to select the initial values of the weights for the network that is proven to greatly speed up training. The superior performance of our method in terms of convergence and generalization has been substantiated by experimental results.


*Keywords:* Extrema equivalence; Random initialization; Mini-max initialization; Promising area; Chessboard initialization

## 1. Introduction

Since the seminal work of Rumelhart et al. [17], feed-forward neural networks have been widely applied for pattern recognition, and function approximation. However, the training of such networks remains a time-consuming procedure. The learning speed depends on the initial values of the weights and biases, the learning rate, and the network topology. The optimal values for these parameters are usually unknown a

* Corresponding author. Tel.: +1-973-596-3670; fax: +1-973-596-5680.
*E-mail address:* nirwan.ansari@njit.edu (N. Ansari).

0925-2312/$ - see front matter 
doi:10.1016/j.neucom.2003.10.014

priori because they depend on the training data. It has been known that error surfaces of the BackPropagation algorithm have extensive flat areas with very little slope and many local minima, the prime reason for the convergence problem. Thus, initialization is the first critical step.

Several weight initialization methods for feed-forward neural networks have been proposed. Among which, random weight initialization, proposed by Rumelhart et al. to break the symmetry [17], is commonly used. The efficiency of this method depends on the initial weight distribution. It has been found that pure random initialization often produces dormant neurons (small weight changes cause only negligible changes to the output) owing to the characteristics of the sigmoidal function. Lee and Kim [13] proved that the probability of prematurely saturated neurons in multilayer feed-forward networks increase with the maximum value of weights. Wessels and Barnard [19] proposed two weight initialization methods to avoid false local minima, but they did not compare the training speeds. Drago and Ridella [6] made the weights uniformly distributed over the interval $[-a, a]$, with $a = 1.3/(1 + E[x^2])^{-1/2}$ for the input layer and $a = 1.3/(1 + 0.3d_{\text{in}})^{-1/2}$ ($d_{\text{in}}$ is the fan-in number of a neuron) for the output layer, and restricted the number of neurons with absolute activations greater than 0.9. Kim and Ra [12] derived a lower bound $(\eta/d_{\text{in}})^{-1/2}$ for the initial length of the weight vector of a neuron, where $\eta$ is the learning rate. According to [7], this weight initialization scheme is suitable for the BP training. Nguyen and Widrow [15] proposed to initialize a multilayer network with linear activation functions, without substantialing it analytically. Besides these random weight initialization methods, Weymaere and Martens [20] proposed an initialization method which incorporates a clustering procedure, a network construction algorithm and network optimization stage. Denoeux and Lengelle [5] initialized a one-hidden-layer network with reference patterns. This method relies on a transformation of the input patterns to unit-length vectors and cluster analysis. Chen and Nutter [3] combined a random weight initialization scheme with pseudo-inverse method to improve the training speed, but the computational cost for matrix inversions is high. Thimm and Fiesler [18] have reviewed the weight initialization methods and evaluated their performance. They concluded that a fixed weight variance of 0.2, which corresponds to a weight range of $[-0.77, 0.77]$, gave the best mean performance for multilayer perceptrons with one hidden layer. In [11], training data are analyzed and the notion of the critical point is introduced for determining the initial weights. The concept of stepwise regression is used in [14] for cascade–correlation learning of feedforward neural networks.

The approximation capabilities of multilayer neural networks have been investigated by many investigators [4,8–10,16]. It has been proved that three-layered neural networks with sigmoidal activation functions can approximate any continuous function which is defined in $R^N$ and for which $\lim_{|x|\to\infty} f(x)$ exists to any desired degree of accuracy [2]. Almost all of these are in the form of denseness results, essentially saying that if you take enough nodes, you can make an arbitrarily good approximation. Whether these representations of continuous functions can be learned by the gradient method is unclear. Using the random weight initialization methods for function approximation, the training is very likely to get stuck in some local minima and cannot escape from it. Moreover, there has been no effective method to determine the size of

the network for a given function. Atiya and Ji [1] found that, for a uniform random initialization in the interval $[-a, a]$, there exists an optimal range $a$ leading to the best generalization.

In this paper, we first formulate the concept of extrema equivalence for estimating the complexity of the function. We then prove that a three-layer neural network with sigmoidal neurons in the hidden layer can realize any one-dimensional function, and that a network with localized sigmoidal neurons in the hidden layer can realize any high dimensional function within the same extrema equivalence class of the objective function. The optimal network size is thus obtained according to the number of the extrema. MMI method, an effective initialization method, is then proposed. In this method, two approaches, accurate extrema location initialization and random extrema location initialization, are introduced. Results using the proposed methods on different function approximation problems are remarkably better than those using traditional initialization methods.

## 2. Construction of networks using extrema equivalence

It is known that the optimal network size is dependent on the complexity of the model to the problem being solved. Too small a network cannot learn the function well, but too large a size will lead to over-fitting and poor generalization performance. Furthermore, excessive computing may be needed. Methods that can determine an appropriate measure of the complexity of the model are thus highly desirable.

**Definition 2.1.** A real-valued function $f(\mathbf{x})$ is said to have an extremum at position $\mathbf{m} \in \mathbf{R}^n$, if $f(\mathbf{x}) > f(\mathbf{m})\ \forall \mathbf{x} \in \mathbf{N}(\mathbf{m})$ or $f(\mathbf{x}) < f(\mathbf{m})\ \forall \mathbf{x} \in \mathbf{N}(\mathbf{m})$, where $\mathbf{N}(\mathbf{m}) \subset \mathbf{R}^n$ is an open neighborhood of $\mathbf{m}$.

For a continuous function $f(\mathbf{x})$, it is reasonable to assume that the extrema of $f(\mathbf{x})$ contain valuable information. The set of extrema $E = \{(\mathbf{x_1}, f(\mathbf{x_1})), \ldots, (\mathbf{x}_n, f(\mathbf{x}_n))\}$ is a strong constraint for continuous functions. For a given set $E$, the class of functions having the same $E$ bear strong similarity with the function $f(x)$, leading to the following definition of extrema equivalence.

**Definition 2.2.** Two continuous functions $f_1 : \mathbf{R}^n \to \mathbf{R}$ and $f_2 : \mathbf{R}^n \to \mathbf{R}$ are said to be extrema equivalent on a compact set $\mathbf{C} \in \mathbf{R}^n$, denoted as $f_1 \bowtie f_2$, if they have the same set of extrema $E$ in $\mathbf{C}$.

Extrema equivalence is transitive, that is, $(f_1 \bowtie f_2) \wedge (f_2 \bowtie f_3) \Rightarrow f_1 \bowtie f_3$. A class of *extrema equivalent* functions is very possible to have the similar level of complexity. Based on this conjecture of complexity for continuous functions, we will show that the optimal size of the network can be estimated (optimal in the sense that the network with the corresponding size can realize an extrema equivalent function as the objective function. In the next section, we propose to utilize more information such as inflection points to accurately estimate the size of a network). For feed-forward

layered networks using sigmoidal function

$$\sigma(x) = \frac{1}{1+\exp(-x)}, \tag{1}$$

we can obtain the following result for one-dimensional function:

**Theorem 2.1.** *If $f(x)$ is a continuous function defined on the closed interval $[a,b]$, and $f(x)$ has $M$ extrema (maxima and minima), then for any $\varepsilon > 0$, there exists a three-layered neural network with $M+1$ sigmoidal neurons in the hidden layer and a linear output neuron, such that*

$$\left| f(x) - \left( \sum_{i=1}^{M+1} c_i \sigma(w_i x + \theta_{\mathbf{i}}) + c_0 \right) \right| < \varepsilon \quad \forall x \in E_{\mathrm{L}} \cup [a,b], \tag{2}$$

*where $E_{\mathrm{L}}$ is the set of extrema locations for both $f(x)$ and $\sum_{i=1}^{M+1} c_i \sigma(w_i x + \theta_{\mathbf{i}}) + c_0$.*

**Proof.** The proof is constructive. We first consider the case for $M=1$ and the extremum being at $x=m$. If $f(a) \geqslant f(b) > f(m)$, since $\sigma(.)$ is monotonically increasing from 0 to 1, for any $\varepsilon > 0$, we can find $w_1$, $w_2$, $\theta_1$, $\theta_2$, such that

$$\begin{aligned}
&\sigma(w_i a + \theta_i) < \frac{\varepsilon}{2|f(m)-f(a)|}, \quad i = 1,2,\\
&1 - \sigma(w_1 m + \theta_1) < \frac{\varepsilon}{2|f(m)-f(a)|},\\
&\sigma(w_2 m + \theta_2) < \frac{\varepsilon}{2|f(m)-f(a)|},\\
&1 - \sigma(w_i b + \theta_i) < \frac{\varepsilon}{2|f(m)-f(a)|}, \quad i = 1,2.
\end{aligned} \tag{3}$$

Let $c_0 = f(a)$, $c_1 = f(m) - f(a)$, $c_2 = f(b) - f(m)$ and

$$g(x) = \sum_{i=1}^{M+1} c_i \sigma(w_i x + \theta_i) + c_0. \tag{4}$$

Then

$$\begin{aligned}
g(a) &= \sum_{i=1}^{2} c_i \sigma(w_i a + \theta_i) + c_0\\
&= \sigma(w_1 a + \theta_1)(f(m) - f(a)) + \sigma(w_2 a + \theta_2)(f(b) - f(m)) + f(a)\\
&= \varepsilon_{a_1} + \varepsilon_{a_2} + f(a).
\end{aligned} \tag{5}$$

Thus we obtain

$$|f(a) - g(a)| = |\varepsilon_{a_1} + \varepsilon_{a_2}| < \varepsilon. \tag{6}$$

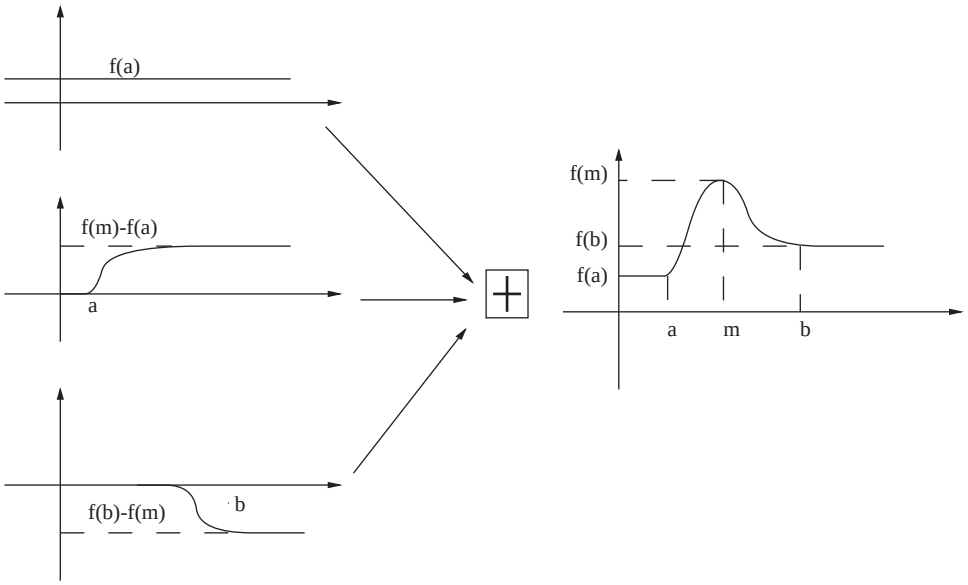


Fig. 1. Using two sigmoidal neurons to approximate a function with one extremum in the interval $[a, b]$.

Repeating the procedure given in Eq. (5) for $x = m$ and $x = b$, we obtain

$$|f(m) - g(m)| = |\varepsilon_{m_1} + \varepsilon_{m_2}| < \varepsilon,$$
$$|f(b) - g(b)| = |\varepsilon_{b_1} + \varepsilon_{b_2}| < \varepsilon. \tag{7}$$

If $f(m) > f(b) \geqslant f(a)$, for any $\varepsilon > 0$, we can select $w_1$, $w_2$, $\theta_1$, $\theta_2$, such that the condition given in Eq. (3) is satisfied. Let $c_0 = f(a)$, $c_1 = f(m) - f(a)$, $c_2 = f(b) - f(m)$, the same result as given by Eqs. (6) and (7) can be obtained (see Fig. 1).

For the case that $M > 1$, let $A$ denote the global maximum and $B$ denote the global minimum of $f(x)$ on interval $[a, b]$. Then for any $\varepsilon > 0$, we can find $w_1, w_2, \ldots, w_{M+1}$ and $\theta_1, \theta_2, \ldots, \theta_{M+1}$, such that

$$\sigma(w_i a + \theta_i) < \frac{\varepsilon}{(M+1)(A-B)}, \quad i = 1, 2, \ldots, M+1,$$
$$1 - \sigma(w_1 m_1 + \theta_1) < \frac{\varepsilon}{(M+1)(A-B)},$$
$$\sigma(w_i m_1 + \theta_i) < \frac{\varepsilon}{(M+1)(A-B)}, \quad i \neq 1,$$
$$1 - \sigma(w_2 m_2 + \theta_2) < \frac{\varepsilon}{(M+1)(A-B)},$$

$$\sigma(w_i m_2 + \theta_i) < \frac{\varepsilon}{(M+1)(A-B)}, \quad i \neq 2,$$

$$\vdots$$

$$1 - \sigma(w_M m_M + \theta_M) < \frac{\varepsilon}{(M+1)(A-B)},$$

$$\sigma(w_i m_M + \theta_i) < \frac{\varepsilon}{(M+1)(A-B)}, \quad i \neq M,$$

$$1 - \sigma(w_i b + \theta_i) < \frac{\varepsilon}{(M+1)(A-B)}, \quad i = 1, 2, \ldots, M+1. \tag{8}$$

Let $c_0 = f(a), c_1 = f(m_1) - f(a), c_2 = f(m_2) - f(m_1), \ldots, c_M = f(m_M) - f(m_{M-1}), c_{M+1} = f(b) - f(m_M)$, then

$$\begin{aligned} g(a) &= \sum_{i=1}^{M+1} c_i \sigma(w_i a + \theta_i) + c_0 \\ &= f(a) + \sigma(w_1 a + \theta_1)(f(m_1) - f(a)) + \sigma(w_2 a + \theta_2)(f(m_2) - f(m_1)) \\ &\quad + \cdots + \sigma(w_M a + \theta_M)(f(m_M) - f(m_{M-1})) \\ &\quad + \sigma(w_{M+1} a + \theta_{M+1})(f(b) - f(m_M)) \\ &= f(a) + \varepsilon_{a_1} + \varepsilon_{a_2} + \cdots + \varepsilon_{a_{M+1}}. \end{aligned} \tag{9}$$

Thus we obtain

$$|f(a) - g(a)| = |\varepsilon_{a_1} + \varepsilon_{a_2} + \cdots + \varepsilon_{a_{M+1}}| < \varepsilon. \tag{10}$$

Repeating the procedure in Eq. (9) to all the local minima and maxima, the same results can be obtained.

Owing to the characteristics of the sigmoidal function (Eq. (1)), its derivative approaches 0 as it approaches 1. Considering Eq. (8), if $\varepsilon$ is small enough, the derivative of $\sigma(w_1 x + \theta_1)$ can be thought almost equal to 0 at $x = m_1$ and monotonically decreased with the increase of $x$ from then on ($x > m_1$). Thus, it will not influence the generation of extrema when $x > m_1$. Based on the same reasoning, $\sigma(w_2 x + \theta_2)$ will not influence the generation of extrema when $x > m_2$. Continuing this procedure, only $M$ extrema are generated within the interval and each of them share the same location and value as the extrema of $f(x)$. Hence, Theorem 2.1 is proved. □

The localized sigmoidal function is used for higher dimensional cases:

$$\tilde{\sigma}(\mathbf{x}) = \frac{1}{1 + \exp(\|\mathbf{x}\|^2)}. \tag{11}$$

**Theorem 2.2.** *If $f(\mathbf{x})$ is a continuous function defined on a compact set $\mathbf{C} \in \mathbf{R}^n$, and $f(\mathbf{x})$ has $M$ extrema, then for any $\varepsilon > 0$, there exists a three-layered neural network*

*with M localized sigmoidal neurons in the hidden layer and a linear output neuron, such that*

$$\left| f(\mathbf{x}) - \left( \sum_{i=1}^{M} c_i \tilde{\sigma}(\mathbf{w_i x} + \theta_\mathbf{i}) + c_0 \right) \right| < \varepsilon \quad \forall \mathbf{x} \in E_\mathrm{L}, \tag{12}$$

*where* $E_\mathrm{L} \subset \mathbf{C}$ *is the set of extrema locations for both* $f(\mathbf{x})$ *and* $\sum_{i=1}^{M} c_i \tilde{\sigma}(\mathbf{w_i x} + \theta_\mathbf{i}) + c_0)$, $\mathbf{w_i}$, $\theta_\mathbf{i} \in \mathbf{R}^n$ *and* $\mathbf{w_i x} = (w_{i_1} x_1, w_{i_2} x_2, \ldots, w_{i_n} x_n)^\mathrm{T}$.

**Proof.** Consider the case where the extrema are at $\mathbf{m_1}, \mathbf{m_2}, \ldots, \mathbf{m_M}$. Let $A$ denote the global maximum and $B$ denote the global minimum of $f(\mathbf{x})$ on the compact set $\mathbf{C}$. Then for any $\varepsilon > 0$, we can find $\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_M}$ and $\theta_\mathbf{1}, \theta_\mathbf{2}, \ldots, \theta_\mathbf{M}$, such that

$$\begin{aligned}
&\tfrac{1}{2} - \tilde{\sigma}(\mathbf{w_1 m_1} + \theta_\mathbf{1}) < \frac{\varepsilon}{M(A-B)}, \\
&\tilde{\sigma}(\mathbf{w_i m_1} + \theta_\mathbf{i}) < \frac{\varepsilon}{M(A-B)}, \quad i \neq 1, \\
&\tfrac{1}{2} - \tilde{\sigma}(\mathbf{w_2 m_2} + \theta_\mathbf{2}) < \frac{\varepsilon}{M(A-B)}, \\
&\tilde{\sigma}(\mathbf{w_i m_2} + \theta_\mathbf{i}) < \frac{\varepsilon}{M(A-B)}, \quad i \neq 2, \\
&\vdots \\
&\tfrac{1}{2} - \tilde{\sigma}(\mathbf{w_M m_M} + \theta_\mathbf{M}) < \frac{\varepsilon}{M(A-B)}, \\
&\tilde{\sigma}(\mathbf{w_i m_M} + \theta_\mathbf{i}) < \frac{\varepsilon}{M(A-B)}, \quad i \neq M.
\end{aligned} \tag{13}$$

Let $c_0 = 0, c_1 = 2f(\mathbf{m_1}), c_2 = 2f(\mathbf{m_2}), \ldots, c_M = 2f(\mathbf{m_M})$. Then

$$\begin{aligned}
g(\mathbf{m_1}) &= \sum_{i=1}^{M} c_i \tilde{\sigma}(\mathbf{w_i m_1} + \theta_\mathbf{i}) + c_0 \\
&= 0 + \tilde{\sigma}(\mathbf{w_1 m_1} + \theta_\mathbf{1}) * 2f(\mathbf{m_1}) + \tilde{\sigma}(\mathbf{w_2 m_1} + \theta_\mathbf{2}) * 2f(\mathbf{m_2}) \\
&\quad + \cdots + \tilde{\sigma}(\mathbf{w_M m_1} + \theta_\mathbf{M}) * 2f(\mathbf{m_M}) \\
&= f(\mathbf{m_1}) + \varepsilon_{\mathbf{m}_1} + \varepsilon_{\mathbf{m}_2} + \cdots + \varepsilon_{\mathbf{m}_{M+1}}.
\end{aligned} \tag{14}$$

Thus we obtain

$$|f(\mathbf{m_1}) - g(\mathbf{m_1})| = |\varepsilon_{\mathbf{m}_1} + \varepsilon_{\mathbf{m}_2} + \cdots + \varepsilon_{\mathbf{m}_{M+1}}| < \varepsilon. \tag{15}$$

Repeating the procedure in Eq. (14) to all the extrema and based on the same reasoning as the proof of Theorem 2.1, Theorem 2.2 is proved. □

It is observed that if the constructive method described in Theorems 2.1 and 2.2 is used to build the network, the output of the network will be very smooth and no inflection points exist. Some real world functions, however, may have some inflection points between the extrema. For this situation, the three-layered neural network with $M+1$ sigmoidal neurons cannot approximate the function accurately. To overcome this deficiency, we consider each inflection point as a new extrema. So for the case when the function has $M$ extrema and $N$ inflection points, the number of the hidden neurons is selected to be $M+N+1$.

## 3. Mini-max weight initialization

As discussed in Section 1, though considerable amount of research work has been done on weight initialization, the results are still far away from being satisfactory. We have experimented with several random initialization methods mentioned in Section 1 for function approximation. It is found that when the number of extrema of the function increases, the training time escalates exponentially. For functions with more than 10 extrema, it becomes almost impossible to converge to a satisfactory result using random initialization and the BP algorithm, even though the number of the hidden neurons are increased. Consider a neural network with a given $\varepsilon > 0$, the promising area of the network is the set of initial weights such that the error (the sum of squares of the differences between the neural network outputs and the desired outputs) is less than $\varepsilon$ after learning.

For most problems, the promising area is only a fraction of the weight space, especially for higher dimensional weight spaces. If we select the starting point randomly in the weight space, the probability of this point lied in the promising area will be very small. Fig. 2 shows the initial network output with different number of hidden neurons using random weight initialization. It is obvious that these can hardly capture the shape (feature) of a function with many extrema. On the other hand, clustering methods are based on the probability distribution of patterns in the feature space. The clusters only reflect the distribution of sample data points that may not resemble the function to be approximated. For example, consider a function with data points being distributed uniformly on the interval $[a,b]$; the clusters may also be uniformly distributed. Therefore, an effective initialization method must be found to initialize weights closer to the promising area.

According to the discussion in Section 2, the extrema equivalent functions have the same set of extrema. Since the extrema are valuable information of the corresponding functions, it is conjectured the promising area of these functions have a common part. To obtain the initial weights, the construction method in the proof in Section 2 is used. Any set of weights that satisfy the condition in the proof can be chosen as the initial weights.

### 3.1. Matched extrema location initialization (MELI)

We have developed a constructive algorithm based on Theorems 2.1 and 2.2 to calculate the initial weights from the input to the hidden layer and from the hidden

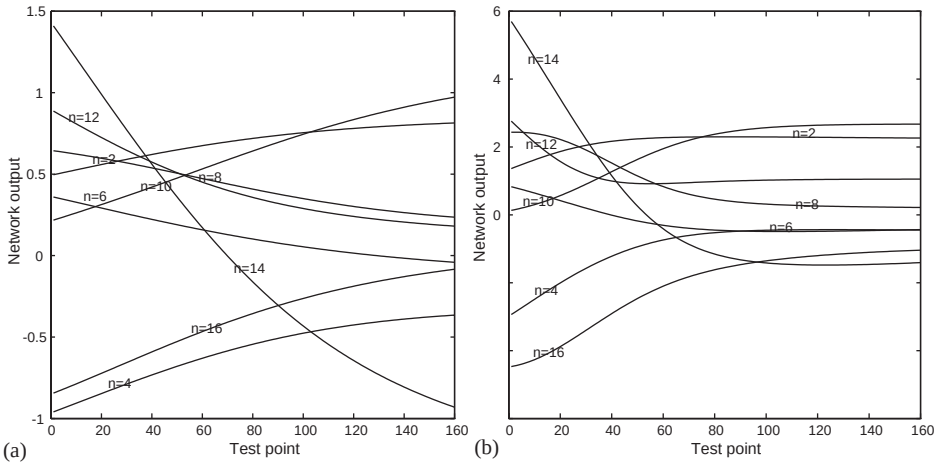


Fig. 2. Network output corresponding to the random initialization: (a) $w \in [-1, 1]$, (b) $w \in [-3, 3]$.

layer to the output layer as follows:

*Step* 1: Find the extrema and inflection points of the function. For the given sample data points $(\mathbf{x_1}, f(\mathbf{x_1}))$, $(\mathbf{x_2}, f(\mathbf{x_2})), \ldots, (\mathbf{x}_n, f(\mathbf{x}_n))$, the following heuristics are adopted to locate the extrema for one-dimensional function:

- Data point $x_i$ is considered to be an extremum if

$$f(x_i) > \max(f(x_{i-1}), f(x_{i+1})) + \eta \tag{16}$$

or

$$f(x_i) < \min(f(x_{i-1}), f(x_{i+1})) - \eta, \tag{17}$$

where $\eta$ is a small positive number.

- If

$$\left|\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}\right| < \min\left(\left|\frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}\right|, \left|\frac{f(x_{i+2}) - f(x_{i+1})}{x_{i+2} - x_{i+1}}\right|\right) - \eta, \tag{18}$$

then the point

$$x^* = \tfrac{1}{2}(x_i + x_{i+1}) \quad \text{and} \quad f(x^*) = f(\tfrac{1}{2}(x_i + x_{i+1})) \tag{19}$$

is considered as a new extrema for weight initialization, but is *not* used as a training point.

For higher dimensional functions, we use the following heuristics to locate the extrema:

- Data point $\mathbf{x_i}$ is considered to be an extremum if

$$f(\mathbf{x_i}) > \max\{f(\mathbf{x}_{k_1}), f(\mathbf{x}_{k_2}), \ldots, f(\mathbf{x}_{k_m})\} + \eta \tag{20}$$

or

$$f(\mathbf{x_i}) < \min\{f(\mathbf{x}_{k_1}), f(\mathbf{x}_{k_2}), \ldots, f(\mathbf{x}_{k_m})\} - \eta \tag{21}$$

where $\eta$ is a small positive number, $\{\mathbf{x}_{k_1}, \ldots, \mathbf{x}_{k_m}\}$ is the set of points that can form a polyhedron encircling $\mathbf{x_i}$, such that $\mathbf{x_i}$ is the only point within this polyhedron.

• If along every axis, the criteria in Eq. (18) are met. Then the point

$$\mathbf{x}^* = \tfrac{1}{2}(\mathbf{x_i} + x_{\mathbf{i+1}}) \quad \text{and} \quad f(\mathbf{x}^*) = f(\tfrac{1}{2}(\mathbf{x_i} + \mathbf{x_{i+1}})) \tag{22}$$

is considered as a new extremum for weight initialization, but is *not* used as a training point.

*Step* 2: Find the upper bound $A$ and lower bound $B$ of $f(\mathbf{x})$ on a compact set $\mathbf{C}$, then use

$$\frac{w_i}{2}(x_{m_i} + x_{m_{i-1}}) + \theta_{\mathbf{i}} = 0 \tag{23}$$

and Eq. (8) to initialize the connections $w_1, w_2, \ldots, w_{M+1}$ and $\theta_1, \theta_2, \ldots, \theta_{M+1}$ between the input and hidden layer for networks with sigmoidal neurons, where $m_0 = a, m_{M+N+1} = b$. Use Eq. (13) to initialize the connections for networks with localized sigmoidal neurons.

*Step* 3: Apply $c_0 = f(a), c_1 = f(m_1) - f(a), c_2 = f(m_2) - f(m_1), \ldots, c_M = f(m_M) - f(m_{M-1}), c_{M+1} = f(b) - f(m_M)$ to initialize the connections between hidden layer and output for networks with sigmoidal neurons. Use $c_0 = 0, c_1 = 2f(\mathbf{m_1}), c_2 = 2f(\mathbf{m_2}), \ldots, c_M = f(\mathbf{m_M})$ to initialize the connections between the hidden layer and output layer for networks with localized sigmoidal neurons.

In Eqs. (16)–(18) and (20) and (21), a small positive number $\eta$ is introduced. It is a threshold to reduce the influence of noise and the inaccuracy of measurement. If $\eta$ is big, the initialization is robust against the noise. However, some real extrema are possibly neglected. On the other hand, a small $\eta$ is possible to lead to over-fitting and worse generalization capability. In our approach, the sequence is firstly normalized. Then $\eta$ is determined as follows: (a) If $f(x_{i-1}) > f(x_{i-2})$ and $f(x_{i+1}) > f(x_{i+2})$, $\eta = 0.05$ in Eq. (16); else $\eta = 0.2$ in Eq. (16); (b) If $f(x_{i-1}) < f(x_{i-2})$ and $f(x_{i+1}) < f(x_{i+2})$, $\eta = 0.05$ in Eq. (17); else $\eta = 0.2$ in Eq. (17); (c) $\eta = 0.1$ in Eqs. (18) and (20) and (21).

The conditions given in Eqs. (8) and (13) are rather tight. If the value of $\varepsilon$ is small enough, the network with the initial weights can approximate the extrema of a function very accurately. This may, however, require negatively large $w_i$ and $\theta_{\mathbf{i}}$, implying that the hidden neurons can only be activated within a small region during the training. In other words, the training based on gradient descent will be slow since in most cases the derivatives of the activation functions are near zero. Moreover, the global characteristics of the sigmoidal function are not fully utilized. To overcome this deficiency, the condition in Eqs. (8) and (13) is relaxed. Through numerous simulations, the following condition is empirically found to be effective:

$$0.08 \leqslant \frac{\varepsilon}{(M+1)(A-B)} \leqslant 0.2.$$

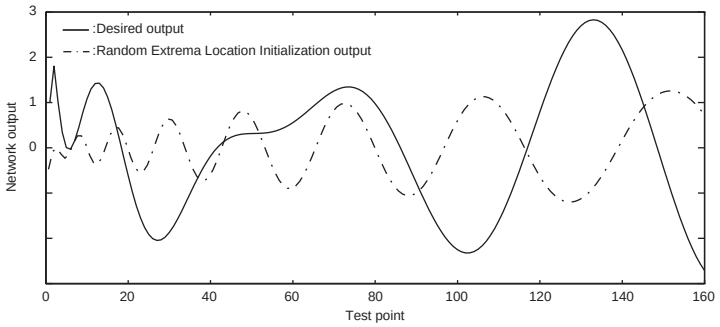


Fig. 3. Network output corresponding to random extrema location initialization.

### 3.2. Random extrema location initialization (RELI)

Computational complexity and dependence on the given function are the drawbacks of the MELI method. Two of the most important reasons that pure random initialization methods are still widely used are their independence on the given data set and their low computational complexity. A method that can exploit the advantages of both of the above initialization methods will be very desirable.

The MELI method proposed in the last section is used to initialize the network with the smallest architecture. If we increase the size of the architecture, the promising area may be enlarged correspondingly, thus providing more flexibility. Based on these heuristics, we propose a random extrema location initialization method. Instead of initializing the network with the same extrema locations and values as the function to be approximated, the network is initialized with extrema distributed randomly within the approximation interval. The only requirement is that the extrema density corresponding to the initialization should be higher than the extrema density of the function to be approximated. This method is illustrated in Fig. 3. Since the extrema are distributed randomly, we do not need to approximate the value of the extrema, and thus the computational complexity of this method is almost the same as the pure random initialization method. In addition, locations of the extrema provide a strong guide to the training, resulting in faster and more accurate convergence.

One interesting question is whether this weight initialization method for function approximation can be used for classification problems. In fact, a classification problem can be considered as a special case of function approximation. In this case, the target function value is 1 for sample points belonging to one class, and 0 for sample points belonging to the other. Then, the same methods for function approximation can be used to solve the classification problem. Based on the RELI method, we have developed a *chessboard initialization method* (CIM). In this method, the weights are initialized such that the extrema are distributed uniformly over the input area (Fig. 6). Experiments in Section 4 show remarkable improvements in convergence time over the random initialization method.

## 4. Experimental results

To evaluate the effectiveness of the proposed initialization methods, several experiments of function approximation have been carried out. The performance of our approach is compared to the previous state-of-the-art method.

**Experiment 1.** The proposed initialization methods are compared to the random initialization method on approximating the Hermite polynomial:

$$f(x) = 1.1(1 - x + 2x^2)\exp(-\tfrac{1}{2}x^2). \tag{24}$$

A random sampling on the interval $[-4, 4]$ is used to obtain 30 sample data for the training set. Note that three extrema (two maxima and one minimum) exist within this interval. Thus, four hidden neurons are needed to approximate the Hermite polynomial within the interval $[-4, 4]$. We first use the sample points to identify approximate positions of the extrema, then the initialization weights are selected according to the procedure described in Section 3.1.

As shown in Fig. 4, the proposed MMI method converges much faster than the random initialization method. This Hermite polynomial problem has been studied in [21] to train the radial basis function neural network. The best results they could

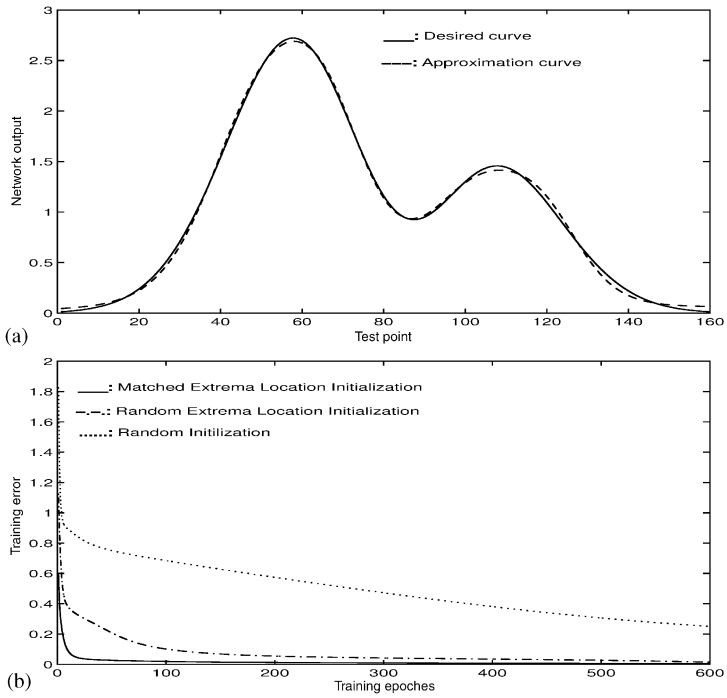


Fig. 4. Approximating the Hermite polynomial: (a) approximation results using the mini-max initialization, (b) training speeds of MMI and random initialization.

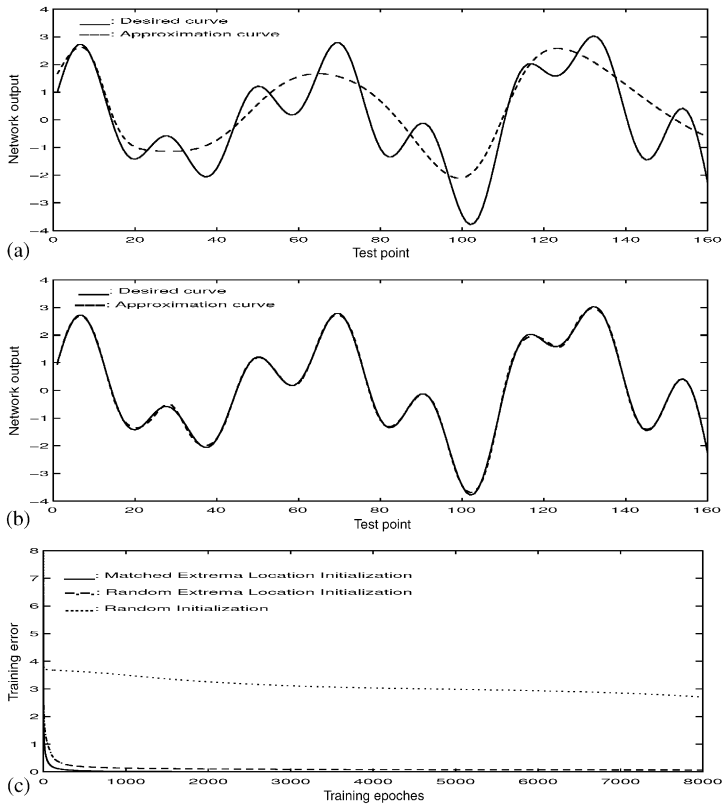


Fig. 5. Approximating $f(x)=(1-0.5x)\sin(2\pi x)+(1+0.4x)\cos(\pi x)+(1+0.1x)\sin(3\pi x)$: (a) approximation result using random initialization, (b) approximation results using MMI, (c) training speeds of MMI and random initialization.

obtain were to use seven hidden units. Note that our method can construct the network with the smaller architecture (only four hidden neurons).

**Experiment 2.** Approximate the continuous function

$$f(x) = (1 - 0.5x)\sin(2\pi x) + (1 + 0.4x)\cos(\pi x) + (1 + 0.1x)\sin(3\pi x) \tag{25}$$

in the interval $[0, 5]$. Sixty uniformly sampled data are used as the training points in this experiment. From the sampled data, it is found that 15 extrema exist in this interval, and thus 16 hidden neurons are needed to approximate this function within interval $[0, 5]$.

As shown in Fig. 5, the network with weights randomly initialized cannot approximate this complicated function accurately. The training is stuck in a local minimum and cannot escape from it, even with more than 10 00 000 epoches. On the other hand, the network is able to approximate the function accurately in only several thousands

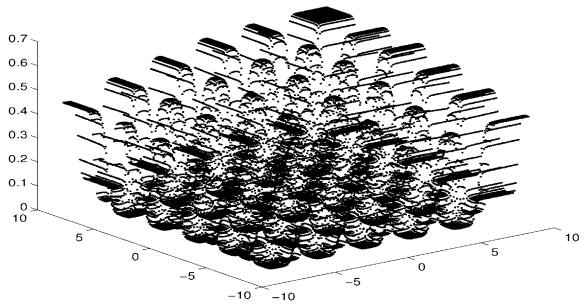


Fig. 6. Chess board initialization for classification.

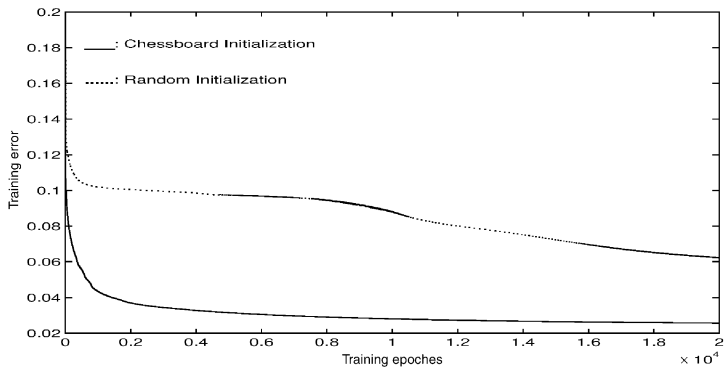


Fig. 7. Performance of the chess board initialization and random initialization on the two-spiral problem.

epoches using the proposed MMI method. The local minima are obviously avoided by the proposed method.

**Experiment 3.** The two-spiral problem is known to be a difficult classification benchmark that requires discrimination between two sets of points lying on two distinct spirals in the plane. Using the radial distance between two spirals as the distance between the extrema, we then obtain the extrema density in this case. According to Theorem 2.1, 20 sigmoidal hidden neurons are selected. Among the hidden neurons, 10 of them are mainly used to initialize the weights corresponding to input 1 ($X$ input) and the others are mainly used to initialize the weights corresponding to input 2 ($Y$ input), resulting in the initialization shown in Fig. 6. As expected, CIM converges much faster than the random initialization method, as shown in Fig. 7.

**Experiment 4.** In order to test the effectiveness of random extrema location method on functions with biased extrema locations, the following function is used:

$$f(x) = \sin(4\pi x^{1/2}) + (1 + 0.4x)\cos(\pi x). \tag{26}$$

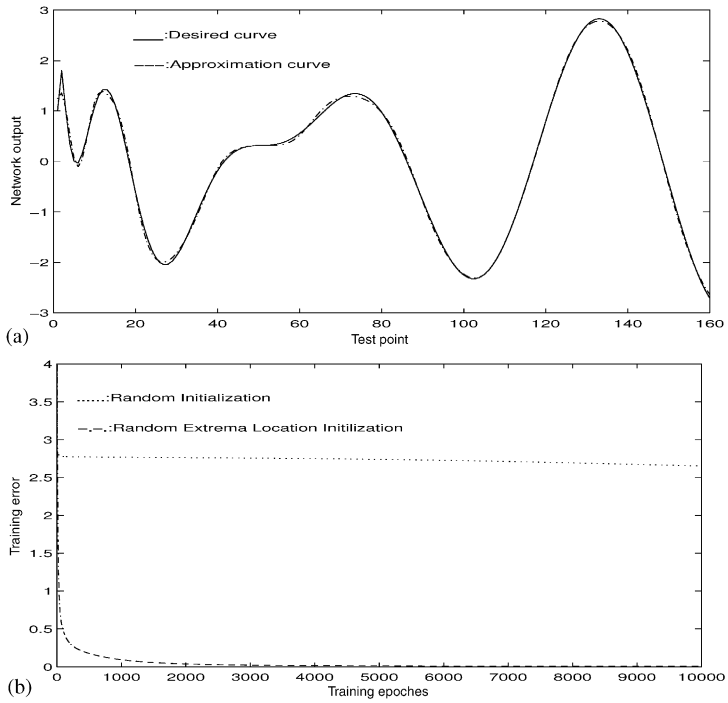


Fig. 8. Performance of the random extrema location initialization and random initialization on the function with biased extrema locations: (a) random extrema location initialization, (b) training speeds of RELI and random initialization.

One hundred uniformly sampled data are used as the training points within the interval $[0,5]$ in this experiment. Note that 9 extrema exist in this interval and locations of the extrema are biased. According to our discussion in Section 3.2, the extrema density of the initialization output should be greater than that of the function to be approximated. Hence, a network with 16 sigmoidal neurons is initialized. Fig. 8 shows that our proposed method converges significantly faster than the random initialization method. That is, our method is effective even in approximating functions with biased extrema locations.

**Experiment 5.** The network is trained to approximate a two-dimensional continuous function:

$$f(x_1,x_2) = \cos(2\pi x_1)\cos(2\pi x_2)\exp -(x_1^2 + x_2^2) \tag{27}$$

on the square $\{(x_1,x_2) : -1 \leqslant x_1 \leqslant 1 \text{ and } -1 \leqslant x_2 \leqslant 1\}$. Four hundred uniformly sampled data are used as training points in this experiment. The localized sigmoidal function is used as the activation function of the hidden neurons. It is found that 25 extrema exist within this interval. Thus, 25 hidden neurons are needed. Using the criteria in Eq. (13) to initialize the weights, the network is constructed to approximate

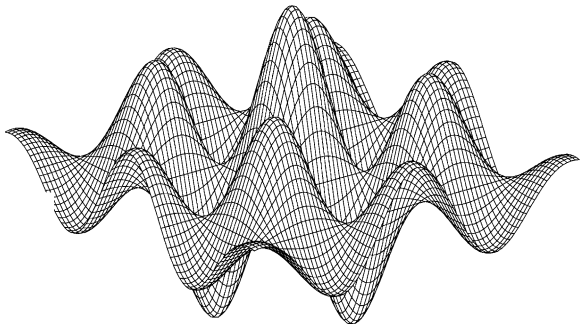

Fig. 9. Surface for the function of $f(x_1,x_2)=\cos(2\pi x_1)\cos(2\pi x_2)\,e^{-(x_1^2+x_2^2)}$.

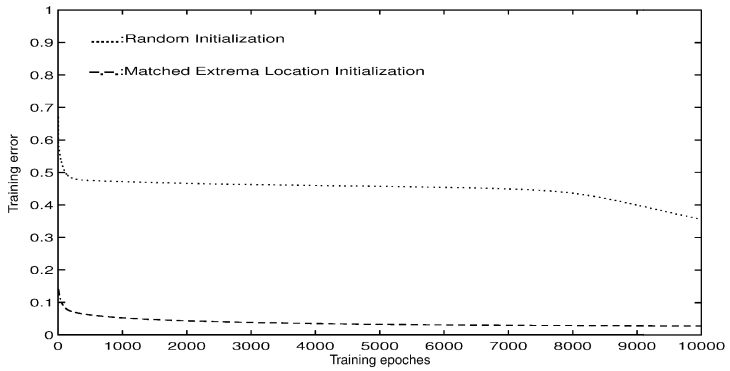


Fig. 10. Approximating $f(x_1,x_2)=\cos(2\pi x_1)\cos(2\pi x_2)\,e^{-(x_1^2+x_2^2)}$.

the function $f(x_1,x_2)$ shown in Fig. 9. Again, as shown in Fig. 10, the function is approximated successfully using our method while random initialization is stuck at a local minimum.

Table 1 summarizes the performance comparison between MMI and the random initialization method in terms of mean square error (MSE) at different epoches. For the five experiments simulated above, it demonstrates that MMI has outperformed the random initialization method in all experiments.

**Experiment 6.** To test the performance of the proposed method on non-analytic problems, two real-life data sets are considered. Both of them are obtained from MPEG-4 compressed video bitstreams; each point of a data set represents the distortion of a frame between the decoded video and the original video sequence. In this experiment, two different groups of frames are selected from the Coastguard sequence to approximate the distortion curve, and 36 hidden neurons are used. The initialization method which has been tested optimal in [18] (variance of 0.2, which corresponds to a weight range of $[-0.77, 0.77]$) is selected to compare with our method. Figs. 11 and 12

Table 1
Comparison of the training results obtained by the MMI and random initialization method for five problems at different epoches

| Experiment | MSE | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | MMI | | | | Random initialization | | | |
| | 500 | 1000 | 3000 | 10 000 | 500 | 1000 | 3000 | 10 000 |
| 1 | 0.031 | 0.012 | 0.009 | 0.008 | 0.310 | 0.127 | 0.045 | 0.011 |
| 2 | 0.167 | 0.103 | 0.049 | 0.024 | 3.681 | 3.469 | 3.107 | 2.652 |
| 3 | 0.056 | 0.042 | 0.034 | 0.026 | 0.107 | 0.103 | 0.100 | 0.091 |
| 4 | 0.175 | 0.102 | 0.021 | 0.008 | 2.753 | 2.696 | 2.691 | 2.615 |
| 5 | 0.065 | 0.050 | 0.029 | 0.020 | 0.479 | 0.473 | 0.465 | 0.351 |

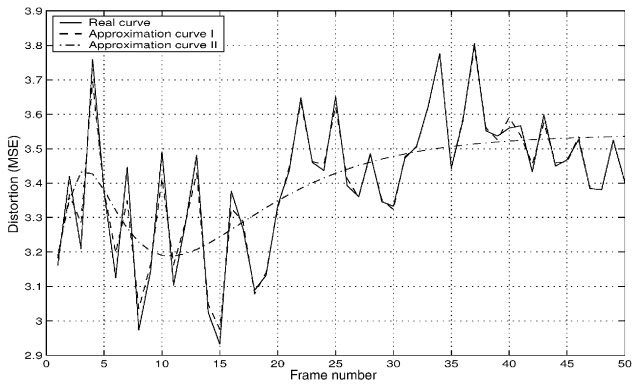


Fig. 11. Approximation results of two initialization methods on the coastguard sequence (first 50 frames): Mini-max initialization (curve I); initialization with variance of 0.2 corresponding to a weight range of $[-0.77, 0.77]$ (curve II).

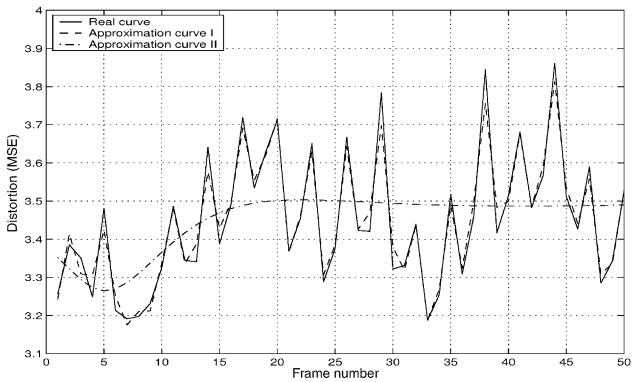


Fig. 12. Approximation results of two initialization methods on the coastguard sequence (second 50 frames): Mini-max initialization (curve I); Initialization with variance of 0.2 corresponding to a weight range of $[-0.77, 0.77]$ (curve II).

show the approximation results using our proposed method and the method in [18] on both data sets, respectively. The results demonstrate the effectiveness of our method in tackling non-analytic problems. On the other hand, the learning by initialization with a variance of 0.2 corresponding to the weight range of $[-0.77, 0.77]$ is stuck at a local minimum.

## 5. Conclusions

A novel initialization method, referred to as the MMI method, has been proposed to efficiently approximate functions. MMI is based on the extrema equivalence concept: we can thus use the number of extrema to estimate the complexity of the function, and construct the network with the most compact size which guarantee the approximation in the extrema points. In order to initialize points located within the promising area, we have proposed the MELI and random extrema location Initialization to start the training. The main advantage of MMI is that it can construct the network with the optimal size with the initial point located very likely within the promising area. Thus, local minima are avoided, and fast convergence can be achieved. Experimental results demonstrate that the proposed MMI method is very effective for function approximation. MMI is also applicable to and effective in solving the classification problem such as the two-spiral problem; some good results have been obtained.

## Acknowledgements

This work was supported in part by the New Jersey Commission on Higher Education via the NJI-TOWER project, and the New Jersey Commission on Science and Technology via the New Jersey Center for Wireless Telecommunications.

## References

[1] A. Atiya, C. Ji, How initial conditions affect generalization performance in large networks, IEEE Trans. Neural Networks 8 (2) (1997) 448–451.

[2] T. Chen, H. Chen, R.W. Liu, Approximation capabilities in $c(r^n)$ by multilayer feedforward networks and related problems, IEEE Trans. Neural Networks 2 (1991) 47–55.

[3] C.L. Chen, R.S. Nutter, Improving the training speed of three-layer feedforward neural nets by optimal estimation of the initial weights, Proceedings of the International Joint Conference on Neural Networks, Vol. 3, Singapore, 1991, pp. 2063–2068.

[4] N. Cotter, The Stone–Weierstrass theorem and its application to neural networks, IEEE Trans. Neural Networks 1 (1990) 290–295.

[5] T. Denoeux, R. Lengelle, Initializing backpropagation networks with prototypes, Neural Networks 6 (1993) 351–363.

[6] G.P. Drago, S. Ridella, Statistically controlled activation weights initialization, IEEE Trans. Neural Network 3 (1992) 627–631.

[7] M. Fernandez-Redondo, C. Hernandez-Espinosa, A comparison among weight initialization methods for multilayer feedforward networks, Proceedings of the IEEE–INNS–ENNS International Joint Conference on Neural Networks, Vol. 4, Como, Italy, 2000, pp. 543–548.

[8] K. Funahash, On approximate realization of continuous mappings by neural networks, Neural Networks 2 (1989) 183–192.

[9] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks 4 (1991) 251–257.

[10] S.-C. Huang, Y.-F. Huang, Bounds on the number of hidden neurons in multilayer perceptrons, IEEE Trans. Neural Networks 2 (1991) 47–55.

[11] K.N. Kanad Keeni, H. Shimodaira, A training scheme for pattern classification using multi-layer feed-forward neural networks. Third International Conference on Computational Intelligence and Multimedia Applications, New Delhi, India, 1999, pp. 307–311.

[12] Y.K. Kim, J.B. Ra, Weight value initialization for improving training speed in the backpropagation network, Proceedings of the International Joint Conference Neural Networks, Vol. 3, Singapore, 1991, pp. 2396–2401.

[13] Y. Lee, S.-H. Oh, M.W. Kim, An analysis of premature saturation in backpropagation learning, Neural Networks 6 (1993) 719–728.

[14] M. Lehtokangas, Fast initialization for cascade-correlation learning, IEEE Trans. Neural Networks 10 (2) (1999) 410–414.

[15] D. Nguyen, B. Widrow, Improving the learning speed of two-layer neural networks by choosing initial values of the adaptive weights, Proceedings of the International Joint Conference on Neural Networks, Vol. 3, San Diego, CA, 1990, pp. 21–26.

[16] J. Park, I.W. Sandberg, Universal approximation using radial-basis function, Neural Comput. 3 (1991) 246–257.

[17] D. Rumelhart, G. Hinton, R. Williams, Learning internal representations by error propagation, in: D. Rumelhart et al., (Eds.), Parallel Distributed Processing, MIT Press, Cambridge, MA, 1986, pp. 318–362.

[18] G. Thimm, E. Fiesler, High-order and multilayer perceptron initialization, IEEE Trans. Neural Network 8 (2) (1997) 349–359.

[19] L.F.A. Wessels, E. Barnard, Avoiding false local minima by proper initialization of connections, IEEE Trans. Neural Network 3 (1992) 899–905.

[20] N. Weymaere, J.-P. Martens, On the initializing and optimization of multilayer perceptrons, IEEE Trans. Neural Networks 5 (5) (1994) 738–751.

[21] L. Yingwei, N. Sundararajan, P. Saratchandran, A sequential learning scheme for function approximation by using minimal radial basis function neural networks, Neural Comput. 9 (1997) 461–478.

**Xi Min Zhang** received the B.E. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, IN 1991, and received the M.E. degree in electrical engineering from Nanyang Technological University, Singapore in 1999, and the Ph.D. degree from New Jersey Institute of Technology, Newark, NJ, in 2003.

From 1991 to 1997, he was with CAAC, China as a systems engineer. Since 1999, he has been a Research Assistant at New Jersey Institute of Technology. In the summer of 2002, he worked as an intern in Mitsubishi Electric Research Labs, Murray Hill, NJ, where he was a Full-Time Consultant afterwards. Since June 2003, he has been a senior engineer in Media Processing Division, Sony Electronics Inc, San Jose, CA. He received the best student paper award at VCIP 2002 and Hashimoto award of New Jersey Institute of Technology in 2003. His research interests include M-D interleaving for various information security issues, H.264 and MPEG video codec, streaming video through network, error resilient coding, artificial neural network and digital watermarking.

**Yan Qiu Chen** received his BEng and MEng respectively in 1985 and 1988 from Tongji University, Shanghai, China. He received his PhD in 1995 from Southampton University, UK. He was an assistant researcher from 1988 to 1991 with Shanghai Maritime University, Shanghai, China, a postdoctoral researcher in 1995 with Glamorgan University, UK, and was an assistant professor from 1996 to 2001 with Nanyang Technological University, Singapore. Dr Chen joined Fudan University, Shanghai, China in 2001. He is currently a full professor, executive deputy director of Intelligent Information Processing Lab, vice chairman of Department of Computer Science and Engineering of Fudan University, P.R. China. Dr Chen's research interest includes computer vision, artificial neural networks, and artificial life.

**Nirwan Ansari** received the B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively.

He joined the Department of Electrical and Computer Engineering, NJIT, as an assistant professor in 1988, and has been promoted to a full professor since 1997. His current research focuses on various aspects of multimedia communications and high speed networks. He is a technical editor of the IEEE Communications Magazine as well as the Journal of Computing and Information Technology. He was a guest editor for the special issue, "Scalability in IP-Oriented Networks", published in the IEEE Communications Magazine June 2003 issue. He authored with E.S.H. Hou *Computational Intelligence for Optimization* (1997, and translated into Chinese in 2000), and edited with B. Yuhas *Neural Networks in Telecommunications* (1994), both published by Kluwer Academic Publishers. He has frequently been invited to give talks and tutorials. He is a keynote speaker for the International Conference on E-Business and Telecommunication Networks (ICETE2004).

He organized (as the General Chair) the First IEEE International Conference on Information Technology: Research and Education (ITRE2003), was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award and a 2003 Chapter Achievement Award, served as the Chair of the IEEE North Jersey Section and in the IEEE Region 1 Board of Governors during 2001-2002, and currently serves in various IEEE committees. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award.

**Yun Q. Shi** has joined the Department of Electrical and Computer Engineering at the New Jersey Institute of Technology, Newark, NJ since 87, and is currently a professor there. He obtained his B.S. degree and M.S. degree from the Shanghai Jiao Tong University, Shanghai, China; his M.S. and Ph.D. degrees from the University of Pittsburgh, PA. His research interests include visual signal processing and communications, digital multimedia data hiding, digital image processing, computer vision and pattern recognition, theory of multidimensional systems and signal processing. Prior to entering graduate school, he had industrial experience in a radio factory as a principal design and test engineer in numerical control manufacturing and electronic broadcasting devices. Some of his research projects are currently supported by several federal and New Jersey State funding agencies. He is an author/coauthor of more than 140 journal and conference proceedings papers in his research areas and a book on *Image and Video Compression for Multimedia Engineering*. He has been an IEEE senior member, the chairman of Signal Processing Chapter of IEEE North Jersey Section, an editorial board member of *International Journal of Image and Graphics*, a member of IEEE Circuits and Systems Society's Technical Committee of Visual Signal Processing and Communications as well as Technical Committee of Multimedia Systems and Applications, a member of IEEE Signal Processing Society's Technical Committee of Multimedia Signal Processing. He is currently an IEEE CASS Distinguished Lecturer, the guest editor of special issue on Image Data Hiding for *International Journal of Image and Graphics*, and the guest editor of special issue on Multimedia Signal Processing for *Journal of VLSI Signal Processing Systems*. He was a

co-general chair of IEEE 2002 International Workshop on Multimedia Signal Processing, a formal reviewer of the *Mathematical Reviews*, an Associate Editor for *IEEE Transactions on Signal Processing* in the area of Multidimensional Signal Processing, the guest editor of the special issue on Image Sequence Processing for the *International Journal of Imaging Systems and Technology*, one of the contributing authors in the area of Signal and Image Processing to the *Comprehensive Dictionary of Electrical Engineering*.