International Journal of Automation and Computing

16(4), August 2019, 534-542
DOI: 10.1007/s11633-016-0996-0

Convergence Analysis of a New MaxMin-SOMO Algorithm

Atlas Khan[1, 3] Yan-Peng Qu[2] Zheng-Xue Li[1]

[1]Department of Applied Mathematics Dalian University of Technology, Dalian 116024, China

[2]School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China

[3]Department of Computing and Mathematics FFCLRP, University of Sao Paulo, Ribeirao Preto, Brazil

Abstract: The convergence analysis of MaxMin-SOMO algorithm is presented. The SOM-based optimization (SOMO) is an optimization algorithm based on the self-organizing map (SOM) in order to find a winner in the network. Generally, through a competitive learning process, the SOMO algorithm searches for the minimum of an objective function. The MaxMin-SOMO algorithm is the generalization of SOMO with two winners for simultaneously finding two winning neurons i.e., first winner stands for minimum and second one for maximum of the objective function. In this paper, the convergence analysis of the MaxMin-SOMO is presented. More specifically, we prove that the distance between neurons decreases at each iteration and finally converge to zero. The work is verified with the experimental results.

Keywords: Optimization, self organizing map (SOM), SOM-based optimization (SOMO) algorithm, particle swarm optimization (PSO), genetic algorithms (GAs).

1 Introduction

The self-organizing feature map (SOFM)[1−5] is a biologically-inspired method for constructing a structured representation of data from an input space by prototypes, called weight vector in topological order fashion. The weight vector is associated with selected elements, the neurons, of an image space where metric relationships are defined between the elements. For any given data-set, the SOFM algorithm selects weight vectors and assigns them to neurons in the network. The weight vectors as a function of neuron coordinates are called the feature map. The applications of self-organizing map range widely from simulations used for the purpose of understanding and modeling of computational maps in the brain to subsystems for engineering applications such as speech recognition, vector quantization and cluster analysis[6−9].

Earlier studies have developed optimization algorithms for solving optimization problems. For solving continuous and discrete problems, the most common optimization algorithms are genetic algorithms (GAs)[10, 11], evolutionary programming[12], and evolutionary strategies[13]. GAs use operators, reproduction crossover, and mutation to yield the optimal solutions; still, the evolutionary programming does not equip crossover function. Comparing with the other two, the evolutionary approaches use a different view of choosing operators. The particle swarm optimization (PSO) introduced in 1995 is similar to other evolutionary algorithms that initializes with a population of random solutions, however it is based on social behaviour rather the natural selection[14−16]. Recently, in PSO several modifications are made and applied in different research fields[17, 18]. Applications of self organizing map (SOM) are wide-spread and range from vector quantization, adaptive equalization, and cluster analysis[19]. SOM related studies demonstrate feasibility to solve the optimization problem of travelling-salesman problem (TSP)[20, 21].

Su et al.[22−26] proposed a new optimization algorithm named SOM-based optimization (SOMO) to deal with continuous optimization issues Recently Wu and Khan[27] generalized the SOMO algorithm to find m winning neurons in a single learning process. Wu and Khan[28] proposed a MaxMin-SOMO algorithm with multiple winners, for simultaneously finding two winners, i.e., first winner stands for minimum and second winner for maximum of the objective function. Khan et al.[29] proposed the convergence of SOMO with distance measure.

In this paper, the convergence issue of the MaxMin-SOMO algorithm with multiple winners is addressed. Particularly, we prove that with a specific distance measure, the distance between the winners and the other neurons in the network decreases with every iteration. And such distance tends to zero in the learning progress. Numerical examples are presented to support the theoretical findings in the paper.

The rest of the paper is organized as follows, in Section 2, a brief introduction of the SOMO with multiple winners is given. The convergence of SOMO with multiple winners

Research Article
Manuscript received June 3, 2014; accepted July 3, 2015; published online February 21, 2017
Recommended by Associate Editor Hong Qiao

is presented in Section 3. In Section 4, we make some numerical experiments to verify our theoretical results. Conclusions are given in Section 5.

2 SOMO with multiple winners

In this section, we discuss the MaxMin-SOMO algorithm for finding one minimum and one maximum of a function simultaneously. It is an easy matter to generalize the algorithm for finding two or more minima and maxima respectively. The MaxMin-SOMO algorithm has the same training steps as those of the original SOMO algorithm, except the step of weights updating process with m winners. The MaxMin-SOMO algorithm is as follows:

Step 1. The initialization of the MaxMin-SOMO is the same as that of SOMO i.e.

Step 1.1. The weight vectors of the four neurons on the corners are initialized as follows:

$$\begin{aligned}
\underline{W}_{1,j} &= \frac{\underline{W}_{1,N}-\underline{W}_{1,1}}{N-1}(j-1)+\underline{W}_{1,1} = \\
&\frac{j-1}{N-1}\underline{W}_{1,N}+\frac{N-j}{N-1}\underline{W}_{1,1} \\
\underline{W}_{M,j} &= \frac{\underline{W}_{M,N}-\underline{W}_{M,1}}{N-1}(j-1)+\underline{W}_{M,1} = \\
&\frac{j-1}{N-1}\underline{W}_{M,N}+\frac{N-j}{N-1}\underline{W}_{M,1} \\
\underline{W}_{i,1} &= \frac{\underline{W}_{M,1}-\underline{W}_{1,1}}{M-1}(i-1)+\underline{W}_{1,1} = \\
&\frac{i-1}{M-1}\underline{W}_{M,1}+\frac{M-i}{M-1}\underline{W}_{1,1} \\
\underline{W}_{i,N} &= \frac{\underline{W}_{M,N}-\underline{W}_{1,N}}{M-1}(i-1)+\underline{W}_{1,N} = \\
&\frac{i-1}{M-1}\underline{W}_{M,N}+\frac{M-i}{M-1}\underline{W}_{1,N} \qquad (1)
\end{aligned}$$

where $i = 2,\cdots,M-1$, $j = 2,\cdots,N-1$ and M and N are some positive integers.

Step 1.2. Initialization of the remaining neurons

$$\begin{aligned}
\underline{W}_{i,j} &= \frac{\underline{W}_{i,N}-\underline{W}_{i,1}}{N-1}(j-1)+\underline{W}_{i,1} \\
&= \frac{j-1}{N-1}\underline{W}_{i,N}+\frac{N-j}{N-1}\underline{W}_{1,N} \qquad (2)
\end{aligned}$$

where $i = 2,\cdots,M-1$, $j = 2,\cdots,N-1$.

Step 1.3. Random noise

A small amount of random noise is added to each weight so as to keep the weight vectors from being linearly dependent

$$\underline{W}_{i,j} = \underline{W}_{i,j} + \underline{t} \qquad (3)$$

for $1 \le i \le M$ and $1 \le j \le N$, where $\underline{t}$ denotes a small random noise.

Step 2. This step aims to find two different winners, denoted by (i_1^*, j_1^*) and (i_2^*, j_2^*) with the best objective function values among the neurons. For each neuron (i,j), its corresponding weight $\underline{W}_{i,j}$ is a vector in $\mathbf{R}^n$, where $1 \le i \le M$ and $1 \le j \le N$. For a special input $\underline{x} = (\underline{x}_1,\cdots\underline{x}_n)^{\mathrm{T}} = (1,1,\cdots,1)^{\mathrm{T}} \in \mathbf{R}^n$, the first winner out of all the neurons is defined as

$$\begin{aligned}
(i_1^*, j_1^*) &= \underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}\times\underline{x}_1,\cdots,\underline{W}_{i,j}\times\underline{x}_n) = \\
&\underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}\times 1,\cdots,\underline{W}_{i,j}\times 1) = \\
&\underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}) \qquad (4)
\end{aligned}$$

and similarly the second winner

$$\begin{aligned}
(i_2^*, j_2^*) &= \underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}\times\underline{x}_1,\cdots,\underline{W}_{i,j}\times\underline{x}_n) = \\
&\underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}\times 1,\cdots,\underline{W}_{i,j}\times 1) = \\
&\underset{1\le i\le M, 1\le j\le N}{\arg} \min f(\underline{W}_{i,j}). \qquad (5)
\end{aligned}$$

The idea of MaxMin-SOMO training is applied to the network such that the weight vector $\underline{W}_{i_1^*,j_1^*}$ of the first winner will get closer and closer to the minimum point and similarly the weight vector $\underline{W}_{i_2^*,j_2^*}$ will get closer to the maximum point during the iterative training process.

Step 3. Weights updating rule

The weights updating rule of the winners and its neighbors is as follows:

For the neurons (i,j) in the neighborhood of the first winner (i_1^*, j_1^*) satisfying $p_1 \le i \le p_2$, $q_1 \le j \le q_2$, where

$$p_1 = \max(i_1^* - R_1, 1) \qquad (6)$$

$$p_2 = \min(i_1^* + R_1, M) \qquad (7)$$

$$q_1 = \max(j_1^* - R_1, 1) \qquad (8)$$

$$q_2 = \min(j_1^* + R_1, N) \qquad (9)$$

and R_1 is sizes of the neighbourhoods, the weights updating rule is as follows:

$$\begin{aligned}
\underline{W}_{i,j}(t+1) &= \underline{W}_{i,j}(t) + \eta_1\beta_1(i_1^*, j_1^*, i, j)[\underline{W}_{i_1^*,j_1^*}(t) - \\
&\underline{W}_{i,j}(t)] + \lambda_1(1-\beta_1(i_1^*, j_1^*, i, j))\underline{p} \qquad (10)
\end{aligned}$$

where

$$\beta_1(i_1^*, j_1^*, i, j) = 1 - \frac{d(i_1^*, j_1^*, i, j)}{\sqrt{M^2+N^2}}. \qquad (11)$$

For the rest of (i,j) neurons

$$\begin{aligned}
\underline{W}_{i,j}(t+1) &= \underline{W}_{i,j}(t) + \eta_2\beta_2(i_2^*, j_2^*, i, j)[\underline{W}_{i_2^*,j_2^*}(t) - \\
&\underline{W}_{i,j}(t)] + \lambda_2(1-\beta_2(i_2^*, j_2^*, i, j))\underline{p} \qquad (12)
\end{aligned}$$

where

$$\beta_2(i_2^*, j_2^*, i, j) = 1 - \frac{d(i_2^*, j_2^*, i, j)}{\sqrt{M^2+N^2}}. \qquad (13)$$

The learning rates, η_1, η_2, λ_1 and λ_2, are real-valued parameters which could be either constants pre-defined by the user or time-varying parameters. The vector $\underline{p}$ is called the perturbation vector.

Step 4. Go to Step 2 until a pre-specified number of generations is achieved or the termination criteria are satisfied.

3 Convergence of SOMO with multiple winners

In this section, the convergence analysis of SOMO with multiple winners is presented. For convergence, we define the maximum distances between any two neurons, i.e.

$$d_k = \max_{i,j,i',j'} ||\underline{W}_{i,j}(k) - \underline{W}_{i',j'}(k)||$$

and

$$\overline{d_k} = \max_{\overline{i},\overline{j},\overline{i'},\overline{j'}} ||\underline{W}_{\overline{i},\overline{j}}(k) - \underline{W}_{\overline{i'},\overline{j'}}(k)||$$

at k-th iteration respectively, where d_k and $\overline{d_k}$ are the maximum distances between any two neurons in the neighborhood of first and second winner respectively. Then we prove that both d_k and $\overline{d_k}$ decrease after each iteration simultaneously and finally converge zero, i.e., $d_k \to 0$, $\overline{d_k} \to 0$ as $k \to \infty$. And we also prove that the function values of the winners in the network decrease after every iteration. The following two theorems are the main idea of the paper.

Theorem 1. Let d_k be the maximum distance of any two neurons in the neighborhood of first winner (i_1^*, j_1^*) and $\overline{d_k}$ be the maximum distance of any two neurons in the neighborhood of second winner (i_2^*, j_2^*) at k-th iteration. Assume that $\frac{d(i_1^*, j_1^*, i', j')}{\sqrt{M^2+N^2}} < 1$ and $\frac{d(i_2^*, j_2^*, \overline{i'}, \overline{j'})}{\sqrt{M^2+N^2}} < 1$ for all neurons i', j', $\overline{i}$, $\overline{j}$, all iterations and $\lambda_{1k}||\underline{p}_k|| \le \frac{1}{4}(1 - C_0)d_k$, $\lambda_{2k}||\underline{p}_k|| \le \frac{1}{4}(1 - C_0)\overline{d_k}$. Then $d_k \to 0$ and $\overline{d_k} \to 0$, as $k \to \infty$.

Proof. Let (i_1, j_1) and (i_2, j_2) be two neurons in the neighborhood of first winner (i_1^*, j_1^*). The weight updating for (i_1, j_1) and (i_2, j_2) are as follows:

$$\begin{aligned}\underline{W}_{i_1,j_1}(k+1) = \underline{W}_{i_1,j_1}(k) + \eta\beta(i_1^*, j_1^*, i_1, j_1)\Big[\underline{W}_{i_1^*,j_1^*}(k) - \\ \underline{W}_{i_1,j_1}(k)\Big] + \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k \end{aligned} \qquad (14)$$

$$\begin{aligned}\underline{W}_{i_2,j_2}(k+1) = \underline{W}_{i_2,j_2}(k) + \eta\beta(i_1^*, j_1^*, i_2, j_2)\Big[\underline{W}_{i_1^*,j_1^*}(k) - \\ \underline{W}_{i_2,j_2}(k)\Big] + \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k. \end{aligned} \qquad (15)$$

Let d_k be the maximum distance between any two neurons in network at k-th iteration and $d_{k+1} = \max\limits_{i_1,j_1,i_2,j_2} ||\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1)||$ at $k+1$st iteration. We have to show that $d_{k+1} < d_k$. Subtracting (14) from (14), we obtain

$$\begin{aligned}&\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1) = \\ &\quad \underline{W}_{i_2,j_2}(k) + \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*,j_1^*}(k) - \underline{W}_{i_2,j_2}(k)] - \\ &\quad \underline{W}_{i_1,j_1}(k) - \eta\beta(i_1^*, j_1^*, i_1, j_1)[\underline{W}_{i_1^*,j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] + \\ &\quad \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k = \\ &\quad \underline{W}_{i_2,j_2}(k) + \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*,j_1^*}(k) - \underline{W}_{i_2,j_2}(k)] - \\ &\quad \underline{W}_{i_1,j_1}(k) - \eta\beta(i_1^*, j_1^*, i_1, j_1)[\underline{W}_{i_1^*,j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] + \\ &\quad \lambda_k((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k). \end{aligned} \qquad (16)$$

We need to take three cases to prove $d_{k+1} < d_k$.

Case 1. If

$$\beta(i_1^*, j_1^*, i_2, j_2) = \beta(i_1^*, j_1^*, i_1, j_1) \qquad (17)$$

then (14) becomes

$$\begin{aligned}&\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1) = \underline{W}_{i_2,j_2}(k) - \\ &\quad \underline{W}_{i_1,j_1}(k) - \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_2,j_2}(k) - \\ &\quad \underline{W}_{i_1,j_1}(k)] + \lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\ &\quad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k) = \\ &\quad (1 - \eta\beta(i_1^*, j_1^*, i_2, j_2))\underline{W}_{i_2,j_2}(k) - \underline{W}_{i_1,j_1}(k) + \\ &\quad \lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\ &\quad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k). \end{aligned} \qquad (18)$$

Taking the norm on both sides of the above equation, we have

$$\begin{aligned}&||\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1)|| \le \\ &\quad (1 - \eta\beta(i_1^*, j_1^*, i_2, j_2))||\underline{W}_{i_2,j_2}(k) - \underline{W}_{i_1,j_1}(k)|| + \\ &\quad ||\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)|| \\ &\quad ||\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1)|| \le \\ &\quad (1 - \eta\beta(i_1^*, j_1^*, i_2, j_2))d_k + ||\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\ &\quad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)||. \end{aligned} \qquad (19)$$

By assumption

$$\frac{d(i_1^*, j_1^*, i_2, j_2)}{\sqrt{M^2 + N^2}} < \alpha < 1 \qquad (20)$$

we have

$$\beta(i_1^*, j_1^*, i_2, j_2) = 1 - \frac{d(i_1^*, j_1^*, i_2, j_2)}{\sqrt{M^2 + N^2}} > \overline{\alpha} > 0 \text{ for all } i_2, j_2.$$

Thus

$$\begin{aligned}&C_0 = (1 - \eta\beta(i_1^*, j_1^*, i_2, j_2)) < 1 - \eta\overline{\alpha} < 1 \\ &\Rightarrow d_{k+1} \le C_0 d_k + \lambda_{1k}||((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\ &\quad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)|| \le \\ &\quad C_0 d_k + ||\lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k|| + \\ &\quad ||\lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k||. \end{aligned} \qquad (21)$$

Again by assumption

$$\lambda_{1k}||p_k|| \le \frac{1}{4}(1 - C_0)d_k \qquad (22)$$

we have

$$\begin{aligned}d_{k+1} \le C_0 d_k + \frac{1}{4}(1 - C_0)d_k + \\ \frac{1}{4}(1 - C_0)d_k = \\ C_0 d_k + \frac{2}{4}(1 - C_0)d_k = \\ C_0 d_k + \frac{1}{2}(1 - C_0)d_k = \\ \frac{1}{2}(1 + C_0)d_k \end{aligned} \qquad (23)$$

$$\Rightarrow d_{k+1} \le \frac{1}{2}(1 + C_0)d_k. \qquad (24)$$

Case 2. If

$$\beta(i_1^*, j_1^*, i_2, j_2) > \beta(i_1^*, j_1^*, i_1, j_1)$$

then add and subtract

$$\eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_1, j_1}(k)]$$

from (16), which follows:

$$\begin{aligned}
&\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1) = \\
&\underline{W}_{i_2,j_2}(k) + \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_2,j_2}(k)] - \\
&\underline{W}_{i_1,j_1}(k) - \eta\beta(i_1^*, j_1^*, i_1, j_1)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] + \\
&\qquad \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] - \\
&\qquad \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] + \\
&\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k) = \\
&\qquad \underline{W}_{i_2,j_2}(k) - \underline{W}_{i_1,j_1}(k) - \\
&\qquad \eta\beta(i_1^*, j_1^*, i_2, j_2)[\underline{W}_{i_2,j_2}(k) - \underline{W}_{i_1,j_1}(k)] + \\
&\eta(\beta(i_1^*, j_1^*, i_2, j_2) - \beta(i_1^*, j_1^*, i_1, j_1))[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_1,j_1}(k)] + \\
&\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k).
\end{aligned}$$

Taking the norm on both sides of the above equation, we have

$$\begin{aligned}
&||\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1)|| \leq \\
&\qquad (1 - \eta(\beta(i_1^*, j_1^*, i_2, j_2))||\underline{W}_{i_2,j_2}(k) - \underline{W}_{i_1,j_1}(k)|| + \\
&\qquad \eta(\beta(i_1^*, j_1^*, i_2, j_2 - \beta(i_1^*, j_1^*, i_1, j_1)))||[\underline{W}_{i_1^*, j_1^*}(k) - \\
&\qquad \underline{W}_{i_1,j_1}(k)]|| + ||\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\
&\qquad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)||. \qquad (25)
\end{aligned}$$

$$\begin{aligned}
&||\underline{W}_{i_2,j_2}(k+1) - \underline{W}_{i_1,j_1}(k+1)|| = \\
&\quad (1 - \eta\beta(i_1^*, j_1^*, i_1, j_1))d_k + ||\lambda_{1k}((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\
&\qquad (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)||. \qquad (26)
\end{aligned}$$

By assumption

$$\frac{d(i_1^*, j_1^*, i_1, j_1)}{\sqrt{M^2 + N^2}} < \alpha < 1 \qquad (27)$$

we have

$$\beta(i_1^*, j_1^*, i_1, j_1) = 1 - \frac{d(i_1^*, j_1^*, i_1, j_1)}{\sqrt{M^2 + N^2}} > \overline{\alpha} > 0$$

for all i_1, j_1.

$$\begin{aligned}
\Rightarrow d_{k+1} \leq\ & C_0 d_k + \lambda_{1k}||((1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k - \\
& (1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k)|| \leq \\
& C_0 d_k + ||\lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_2, j_2))\underline{p}_k|| + \\
& ||\lambda_{1k}(1 - \beta(i_1^*, j_1^*, i_1, j_1))\underline{p}_k||. \qquad (28)
\end{aligned}$$

Again by assumption

$$\lambda_{1k}||p_k|| \leq \frac{1}{4}(1 - C_0)d_k \qquad (29)$$

we have

$$\begin{aligned}
d_{k+1} \leq\ & C_0 d_k + \frac{1}{4}(1 - C_0)d_k + \frac{1}{4}(1 - C_0)d_k = \\
& C_0 d_k + \frac{2}{4}(1 - C_0)d_k = \\
& C_0 d_k + \frac{1}{2}(1 - C_0)d_k = \\
& \frac{1}{2}(1 + C_0)d_k \qquad (30)
\end{aligned}$$

$$\Rightarrow d_{k+1} \leq \frac{1}{2}(1 + C_0)d_k. \qquad (31)$$

Case 3. If

$$\beta(i_1^*, j_1^*, i_2, j_2) < \beta(i_1^*, j_1^*, i_1, j_1)$$

then add and subtract $\eta\beta(i_1^*, j_1^*, i_1, j_1)[\underline{W}_{i_1^*, j_1^*}(k) - \underline{W}_{i_2,j_2}(k)]$ from (16) and the proof of Case 3 is similar to Case 2.

$$\Rightarrow d_{k+1} \leq \frac{1}{2}(1 + C_0)d_k \qquad (32)$$

From the above three cases, i.e.

$$\begin{aligned}
d_{k+1} \leq\ & \frac{1}{2}(1 + C_0)d_k \leq (\frac{1}{2}(1 + C_0))^2 d_{k-1}, \cdots, \leq \\
& (\frac{1}{2}(1 + C_0))^k d_1. \qquad (33)
\end{aligned}$$

Hence $d_k \to 0$, as $k \to \infty$.

For the rest of (i, j) neurons. Let $(\overline{i_1}, \overline{j_1})$ and $(\overline{i_2}, \overline{j_2})$ be two neurons in the neighborhood of second winner (i_2^*, j_2^*). The weight updating for $(\overline{i_1}, \overline{j_1})$ and $(\overline{i_2}, \overline{j_2})$ are as follows:

$$\begin{aligned}
&\underline{W}_{\overline{i_1}, \overline{j_1}}(k+1) = \underline{W}_{\overline{i_1}, \overline{j_1}}(k) + \eta\beta(i_2^*, j_2^*, \overline{i_1}, \overline{j_1})[\underline{W}_{i_2^*, j_2^*}(k) - \\
&\qquad \underline{W}_{\overline{i_1}, \overline{j_1}}(k)] + \lambda_{1k}(1 - \beta(i_2^*, j_2^*, \overline{i_1}, \overline{j_1}))\underline{p}_k. \qquad (34)
\end{aligned}$$

$$\begin{aligned}
&\underline{W}_{\overline{i_2}, \overline{j_2}}(k+1) = \underline{W}_{\overline{i_2}, \overline{j_2}}(k) + \eta\beta(i_2^*, j_2^*, \overline{i_2}, \overline{j_2})[\underline{W}_{i_2^*, j_2^*}(k) - \\
&\qquad \underline{W}_{\overline{i_2}, \overline{j_2}}(k)] + \lambda_{1k}(1 - \beta(i_2^*, j_2^*, \overline{i_2}, \overline{j_2}))\underline{p}_k. \qquad (35)
\end{aligned}$$

Let $\overline{d_k}$ be the maximum distance between any two neurons in the neighborhood of second winner at k-th iteration and

$$\overline{d_{k+1}} = \max_{\overline{i_1}, \overline{j_1}, \overline{i_2}, \overline{j_2}} ||\underline{W}_{\overline{i_2}, \overline{j_2}}(k+1) - \underline{W}_{\overline{i_1}, \overline{j_1}}(k+1)||$$

at $k + 1$st iteration. The proof of the distance $\overline{d_k}$ is similar to the d_k.

Hence d_k and $\overline{d_k}$ approach to zero simultaneously as $k \to \infty$. □

Theorem 2. Let $\underline{W}_{i,j}(k)$ be $M \times N$ neurons in the network, for $1 \leq i \leq M$, $1 \leq j \leq N$. Let $\underline{W}_{i_1^*, j_1^*}(k)$, $\underline{W}_{i_2^*, j_2^*}(k)$ be the two winners in the network at k-th iteration. Assume that $\lambda_{1k}||\underline{p}_k|| \leq \frac{1}{4}(1 - C_0)d_k$, $\lambda_{2k}||\underline{p}_k|| \leq \frac{1}{4}(1 - C_0)\overline{d_k}$. Then the function values $f(\underline{W}_{i_1^*, j_1^*}(k+1)) < f(\underline{W}_{i_1^*, j_1^*}(k))$, $f(\underline{W}_{i_2^*, j_2^*}(k+1)) < f(\underline{W}_{i_2^*, j_2^*}(k))$, as $k \to \infty$.

Proof. The weight updating of the neurons in the neighborhood of first winner (i_1^*, j_1^*) is as follows:

$$\begin{aligned}
&\underline{W}_{i,j}(k+1) = \underline{W}_{i,j}(k) + \eta\beta(i_1^*, j_1^*, i, j)[\underline{W}_{i_1^*, j_1^*}(k) - \\
&\qquad \underline{W}_{i,j}(k)] + \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i, j))\underline{p}_k. \qquad (36)
\end{aligned}$$

Let

$$f(\underline{W}_{i,j}(k+1)) = f(\underline{W}_{i,j}(k) - \eta\beta(i_1^*, j_1^*, i, j)[\underline{W}_{i_1^*,j_1^*}(k) - \underline{W}_{i,j}(k)) + \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i, j))\underline{p}) = f((1 - \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i,j}(k) + \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i_1^*,j_1^*}(k) + \lambda_{1k}(1 - \beta(i_1^*, j_1^*, i, j))\underline{p}_k). \tag{37}$$

By assumption

$$\lambda_k||\underline{p}_k|| \le \frac{1}{4}(1 - C_0)d_k. \tag{38}$$

Thus the above equation (37) becomes

$$f(\underline{W}_{i,j}(k+1)) \le f((1 - \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i,j}(k) + \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i_1^*,j_1^*}(k) + \frac{1}{4}(1 - C_0)d_k) \tag{39}$$

$$f(\underline{W}_{i,j}(k+1)) = f((1 - \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i,j}(k) + \eta\beta(i_1^*, j_1^*, i, j)\underline{W}_{i_1^*,j_1^*}(k) \quad (\text{since by Theorem 1 } d_k \to 0) \tag{40}$$

$$f(\underline{W}_{i,j}(k+1)) < (1 - \eta\beta(i_1^*, j_1^*, i, j)f(\underline{W}_{i,j}(k)) - \eta\beta(i_1^*, j_1^*, i, j)f(\underline{W}_{i_1^*,j_1^*}(k)) \quad (\text{By convexity of } f) = (1 - \eta\beta(i_1^*, j_1^*, i, j)f(\underline{W}_{i,j}(k)) + \eta\beta(i_1^*, j_1^*, i, j)f(\underline{W}_{i,j}(k)) = f(\underline{W}_{i,j}(k)) \quad (\text{since } f(\underline{W}_{i_1^*,j_1^*} \le f(\underline{W}_{i,j}(k)). \tag{41}$$

Hence

$$f(\underline{W}_{i,j}(k+1)) < f(\underline{W}_{i,j}(k)). \tag{42}$$

The above equation implies that

$$f(\underline{W}_{i_1^*,j_1^*}(k+1)) < f(\underline{W}_{i_1^*,j_1^*}(k)). \tag{43}$$

For the rest of (i, j), the weight updating is as follows:

$$\underline{W}_{\bar{i},\bar{j}}(k+1) = \underline{W}_{\bar{i},\bar{j}}(k) + \eta\beta(i_2^*, j_2^*, \bar{i}, \bar{j})[\underline{W}_{i_2^*,j_2^*}(k) - \underline{W}_{\bar{i},\bar{j}}(k)] + \lambda_{2k}(1 - \beta(i_2^*, j_2^*, i, j))\underline{p}_k \quad \text{for} \quad 1 \le \bar{i} \le M, 1 \le \bar{j} \le N. \tag{44}$$

The proof as same as above. Hence

$$f(\underline{W}_{i_2^*,j_2^*}(k+1)) < f(\underline{W}_{i_2^*,j_2^*}(k)). \tag{45}$$

Hence the function values of the both winners are decreasing after every iteration simultaneously. □

In summary, we initialized the neurons on four corners, then after every iteration, the neurons get closer and closer around the first and second winners. We proved in the above theorems, that the neurons i.e., (i, j) for $1 \le i \le M$, $1 \le j \le N$ are converging to two different points after every iteration, and the function values of winners are decreasing after every iteration simultaneously.

We remark that the MaxMin-SOMO algorithm has an issue of defining the different neighbourhood for maximum and minimum winners. We intend to investigate this problem in future works.

4 Experimental results

4.1 Objective functions

We have carried out simulations with the following functions, to investigate the rate of convergence of the SOMO with multiple winners.

1) U_1 function:

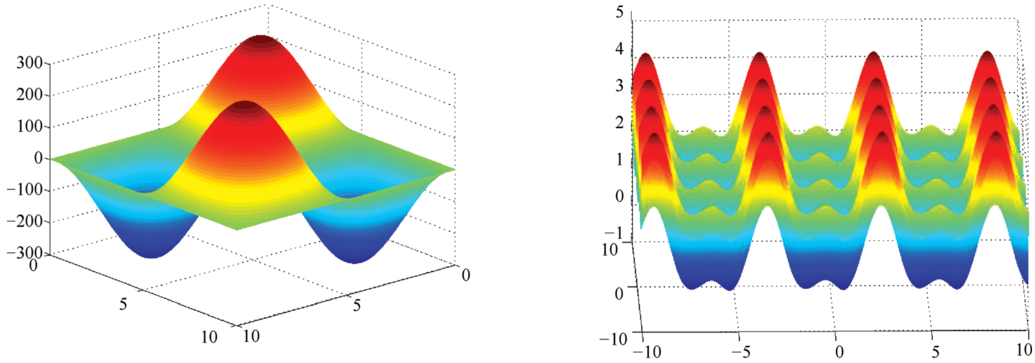
$$f(\underline{x}) = 300\sin(2\pi x)\sin(2\pi y) - \sin(\pi x)\sin(\pi y) \tag{46}$$

2) Giunta function:

$$f(\underline{x}) = \sum_{i=1}^{30}\left(\sin\left(\frac{16}{15}x_i - 1\right) + \sin^2\left(\frac{16}{15}x_i - 1\right) + \frac{1}{50}\sin\left[4\left(\frac{16}{15}x_i - 1\right)\right]\right) + 0.3. \tag{47}$$

4.2 Parameters of simulation

Table 1 tabulated the global minimum, dimensions and the upper bound of the number of iterations. All simulations conducted with the network size 30 × 30 neurons and the learning parameters are $\eta = 0.2$ and λ_{1k}, λ_{2k} are decreasing after each proceeding iteration. Fig. 1 shows the graphs of the two test functions.



(a) U_1 function

(b) Giunta function

Fig. 1 Graphs of the two test functions

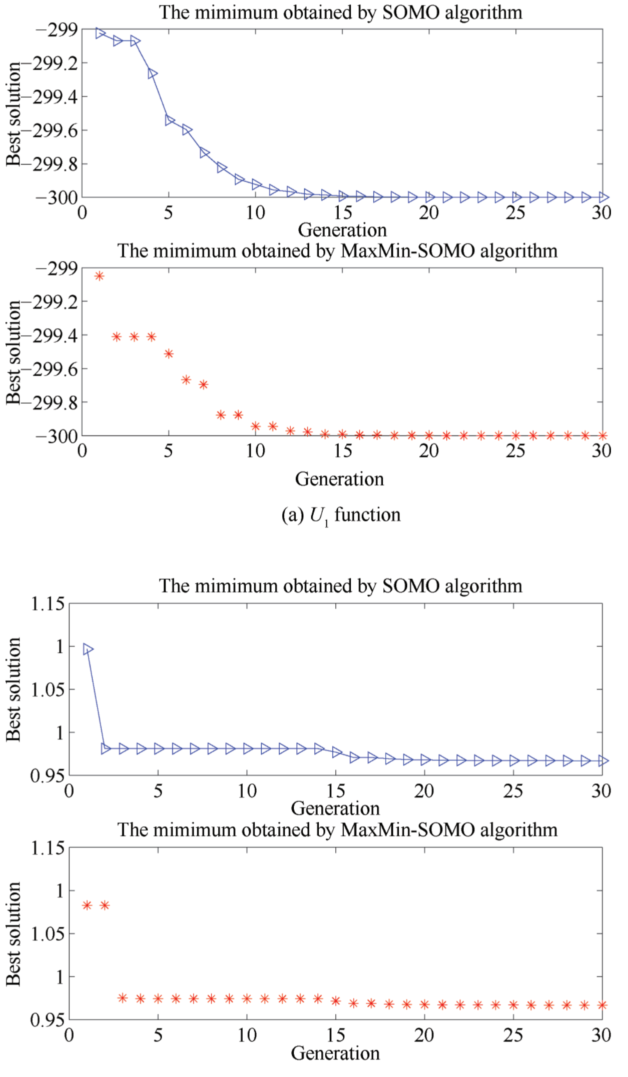
Table 1 Parameters for the test functions

Test function	Dimensions	Initial range	Minimum	Number of iterations
U_1 function	2	$0\leq x_i \leq 10$		100
Giunta	30	$-10 \leq x_i \leq 10$	≈ 0.9	100

4.3 Simulations of the SOMO with multiple winners

For each function, each algorithm conducted 30 runs. The best solutions found for each run after pre-specified number of generations recorded. Table 2 tabulates the comparison of the simulation results. The mean column and the standard deviation column represent the mean and the standard deviation of the best solutions of 30 runs. In Table 2 we find the minimum and maximum of a function by running PSO and SOMO twice. But by SOMO with multiple winners we find the both minimum and maximum of a function simultaneously.

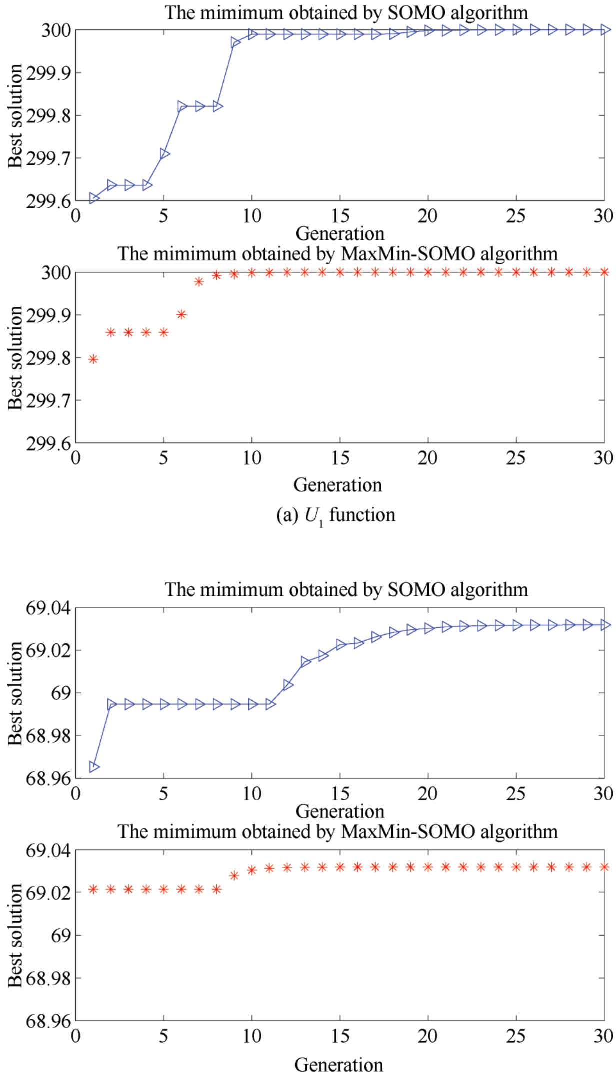
Figs. 2 and 3 show the best performance curves by finding the minimum and maximum of a function respectively. In Fig. 2 we compare the minimum of a function obtained by SOMO and SOMO with multiple winners respectively. In Fig. 3 we compare the maximum of a function obtained by SOMO and SOMO with multiple winners respectively. By SOMO, we find the minimum and maximum of a function separately by running the algorithm twice.



(a) U_1 function

(b) Giunta function

Fig. 2 Best performance curves corresponding to SOMO algorithm and MaxMin-SOMO algorithm for finding a minimum of a function



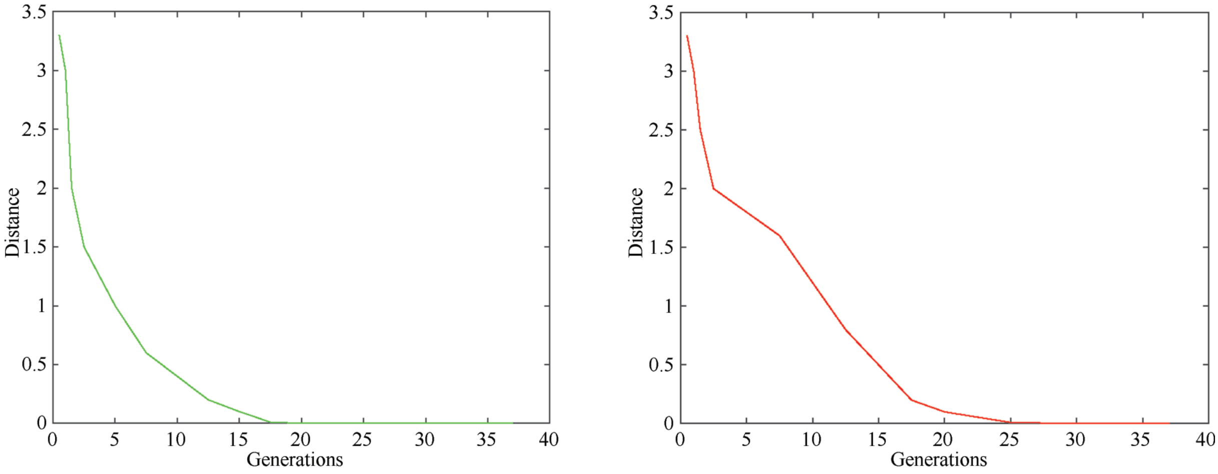
(a) U_1 function

(b) Giunta function

Fig. 3 Best performance curves corresponding to SOMO algorithm and MaxMin-SOMO algorithm for finding a maximum of a function

Table 2 Comparison of MaxMin-SOMO and the original SOMO algorithms for finding minimum and maximum

Test function	Algorithm	Mean	Standard deviation	Mean of time	Standard deviation of time
U_1 Function	PSO				
	Minimum	−1.873 2E+002	4.239 5E−012	2.021 4	2.067 9
	Maximum	2.312 3E+002	1.348 1E−010	2.693 9	2.959 2
	SOMO				
	Minimum	−2.999 9E+002	5.684 3E−014	1.064 2	1.067 2
	Maximum	2.999 9E+002	1.718 4E−012	1.051 8	1.030 7
	MaxMin-SOMO				
	Minimum	−2.999 9E+002	4.224 6E−011	0.553 8	0.566 4
	Maximum	2.999 9E+002	3.821 6E−013		
Giunta	PSO				
	Minimum	4.003 5	1.087 0	1.783 1	0.063 0
	Maximum	72.031 2	1.092 5	1.635 2	0.059 0
	SOMO				
	Minimum	0.967 0	1.490 1E−09	1.011 6	1.004 9
	Maximum	69.031 8	1.480 7E−011	1.151 2	1.142 6
	MaxMin-SOMO				
	Minimum	0.967 0	1.656 5E−010	0.737 2	0.663 6
	Maximum	69.031 8	2.288 0E−011		



(a) Distance between first winner and rest of neurons

(b) Distance between second winner and rest of neurons for U_1 function

Fig. 4 Value of winners for U_1 function

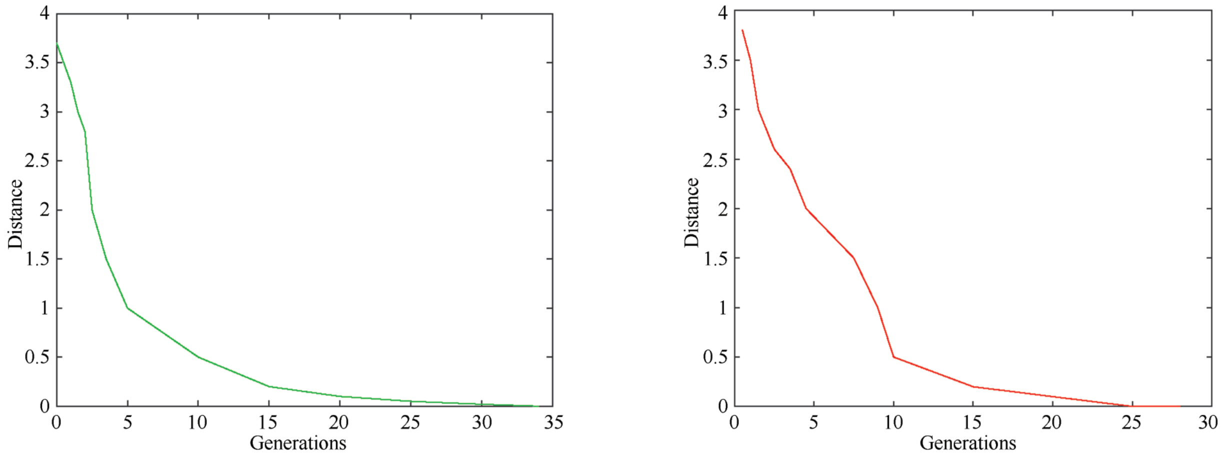
But by SOMO with multiple winners, we find the minimum and maximum of a function simultaneously.

4.4 Simulations for the convergence analysis of SOMO with multiple winners

The experiments show that the neurons i.e., $\underline{W}_{i,j}$ for $1 \leq i \leq M, 1 \leq j \leq N$ get closer and closer around the first $\underline{W}_{i_1^*,j_1^*}(k)$ and second winner $\underline{W}_{i_2^*,j_2^*}(k)$ during iteration process. From Figs. 4 and 5 we observe that the distance between the winners $\left(\text{i.e.}, (i_1^*, j_1^*), (i_2^*, j_2^*)\right)$ and other neurons are decreasing exponentially. It can be concluded that the distance between winners and other neurons, i.e., $\underline{W}_{i,j}$ for $1 \leq i \leq M, 1 \leq j \leq N$ approach to zero as $k \to \infty$.

5 Conclusions

Convergence results are established for the SOMO with multiple winners, with a specific distance measure. We define a specific maximum distance between neurons, we proved that this distance approached to zero, i.e, $d_k \to 0$ and $\overline{d_k} \to 0$ as $k \to \infty$. Also we showed that all neurons converge to a winner after each iteration process. Two numerical examples for the algorithm are provided to support our theoretical findings and demonstrate that the distance between neurons approaches to zero after each iteration.



(a) Distance between first winner and rest of neurons

(b) Distance between second winner and rest neurons for Giunta function

Fig. 5 Value of winners for the Giunta function

Acknowledgements

This work was supported by National Natural Science Foundation of China (Nos. 11171367 and 61502068), the Fundamental Research Funds for the Central Universities of China (No. 3132014094), the China Postdoctoral Science Foundation (Nos. 2013M541213 and 2015T80239) and Fundacao da Amaro a Pesquisa do Estado de Sao Paulo (FAPESP) Brazil (No. 2012/23329-5).

The authors wish to thank the associate editor and the anonymous reviewers for their helpful and interesting comments. We are very grateful to professor Wu Wei for extremely helpful discussion and comments to improve the quality of the paper. Thanks to professor Ricardo Zorzetto Nicoliello Vêncio for his help to improve the text of the paper.

References

[1] T. Kohonen. Analysis of a simple self-organizing process. *Biological Cybernetics*, vol. 44, no. 2, pp. 135–140, 1982.

[2] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[3] T. Kohonen, E. Oja, O. Simula, A. Visa, J. Kangas. Engineering applications of the self-organizing map. *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1358–1384, 1996.

[4] T. Kohonen. Essentials of the self-organizing map. *Neural Networks*, vol. 37, pp. 52–65, 2013.

[5] H. J. Yin. Nonlinear dimensionality reduction and data visualization: A review. *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 294–303, 2007.

[6] X. F. Hu, Q. H. Weng. Estimating impervious surfaces from medium spatial resolution imagery using the self-organizing map and multi-layer perceptron neural networks. *Remote Sensing of Environment*, vol. 113, no. 10, pp. 2089–2102, 2009.

[7] R. Q. Huang, L. F. Xi, X. L. Li, C. R. Liu, H. Qiu, J. Lee. Residual life predictions for ball bearings based on self-organizing map and back propagation neural network methods. *Mechanical Systems and Signal Processing*, vol. 21, no. 1, pp. 193–207, 2007.

[8] D. R. Chen, R. F. Chang, Y. L. Huang. Breast cancer diagnosis using self-organizing map for sonography. *Ultrasound in Medicine and Biology*, vol. 26, no. 3, pp. 405–411, 2000.

[9] A. Skupin, J. R. Biberstine, K. Börner. Visualizing the topical structure of the medical sciences: A self-organizing map approach. *PLoS ONE*, vol. 8, no. 3, pp. e58779, 2013.

[10] J. H. Holland. *Adaptation in Natural and Artificial Systems*, Ann Arbor, USA: University of Michigan Press, 1975.

[11] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Boston, USA: Addison-Wesley, 1989.

[12] L. J. Fogel. Evolutionary programming in perspective: The top-down view. *Computational Intelligence: Imitating Life*, J. M. Zurada, R. J. Marks II, C. J. Robinson, Eds., Piscataway, USA: IEEE Press, pp. 135–146, 1994.

[13] I. Rechenberg. Evolution strategy. *Computational Intelligence: Imitating Life*, J. M. Zurada, R. J. Marks II, C. J. Robinson, Eds., Piscataway, USA: IEEE Press, pp. 147–159, 1994.

[14] J. Kennedy, R. C. Eberhart, Y. H. Shi. *Swarm Intelligence*, New York, USA: Academic Press, 2001.

[15] M. S. Armugam, M. V. C. Rao. On the optimal control of single-stage hybrid manufacturing systems via novel and different variants of particle swarm optimization algorithm. *Discrete Dynamics in Nature and Society*, vol. 2005, no. 3, pp. 257–279, 2005.

[16] R. Eberhart, J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, IEEE, Nagoya, Japan, pp. 39–43, 1995.

[17] S. K. Goudos, J. N. Sahalos. Microwave absorber optimal design using multi-objective particle swarm optimization. *Microwave and Optical Technology Letters*, vol. 48, no. 8, pp. 1553–1558, 2006.

[18] H. Zhang, C. M. Tam, H. Li, J. J. Shi. Particle swarm optimization-supported simulation for construction operations. *Journal of Construction Engineering and Management*, vol. 132, no. 12, pp. 1267–1274, 2006.

[19] T. Kohonen. *Self-organizing Maps*, Berlin, Germany: Springer-Verlag, 1995.

[20] B. Angéniol, G. de La Croix Vaubois, J. Y. Le Texier. Self-organizing feature maps and the travelling salesman problem. *Neural Networks*, vol. 1, no. 4, pp. 289–293, 1988.

[21] H. D. Jin, K. S. Leung, M. L. Wong, Z. B. Xu. An efficient self-organizing map designed by genetic algorithms for the traveling salesman problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 877–888, 2003.

[22] J. H. Chen, L. R. Yang, M. C. Su. Comparison of SOM-based optimization and particle swarm optimization for minimizing the construction time of a secant pile wall. *Automation in Construction*, vol. 18, no. 6, pp. 844–848, 2009.

[23] M. C. Su, T. K. Liu, H. T. Chang. Improving the self-organizing feature map algorithm using an efficient initialization scheme. *Tamkang Journal of Science and Engineering*, vol. 5, no. 1, pp. 35–48, 2002.

[24] J. H. Chen, L. R. Yang, M. C. Su, J. Z. Lin. Optimal construction sequencing for secant pile wall. In *Proceedings of the 2008 IEEE International Conference on Industrial Engineering and Engineering Management*, IEEE, Singapore, pp. 2142–2147, 2008.

[25] M. C. Su, Y. X. Zhao. A variant of the SOM algorithm and its interpretation in the viewpoint of social influence and learning. *Neural Computing & Applications*, vol. 18, no. 8, pp. 1043–1055, 2009.

[26] M. C. Su, Y. X. Zhao, J. Lee. SOM-based optimization. In *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks*, IEEE, Budapest, Hungarg, pp. 781–786, 2004.

[27] W. Wu, A. Khan. SOMO-m optimization algorithm with multiple winners. *Discrete Dynamics in Nature and Society*, Article ID 969104, 2012.

[28] W. Wu, A. Khan. MaxMin-SOMO: An SOM optimization algorithm for simultaneously finding maximum and minimum of a function. In *Proceedings of the 9th International Symposium on Neural Networks, Lecture Notes in Computer Science*, Springer, Shenyang, China, pp. 598–606, 2012.

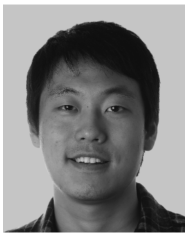
[29] A. Khan, L. Z. Xue, W. Wei, Y. P. Qu, A. Hussain, R. Z. N. Vencio. Convergence analysis of a new self organizing map based optimization (SOMO) algorithm. *Cognitive Computation*, vol. 7, no. 4, pp. 477–486, 2015.

Atlas Khan received the B. Sc. and M. Sc. degrees in mathematics from Gomal University DI Khan Pakistan, in 2005 and 2007, respectively, and M. Phil. degree in mathematics from Quaid-i-Azam University, Pakistan in 2010. He obtained the Ph. D. degree from Department of Applied Mathematics, Dalian University of Technology, China in 2013. Since August 2013, he is doing post-docotor in bioinformatics with Department of Computing and Mathematics, University of Sao Paulo, Brazil. He has published a number of papers in international journals and conferences.

His research interests include bioinformatios, computational biology, neural networks and coding theory.

E-mail: atlas.khan@ffclrp.usp.br (Corresponding author)

ORCID iD: 0000-0002-6651-2725



Yan-Peng Qu received the Ph. D. degree in computational mathematics from Dalian University of Technology, China in 2012. He is a lecturer with the Information Science and Technology College at Dalian Maritime University, China.

His research interests include rough and fuzzy set theory, pattern recognition, neural networks, classification and feature selection.

E-mail: yanpengqu@dlmu.edu.cn

Zheng-Xue Li received the Ph. D. degree in mathematics from Jilin University, Changchun, China in 2001. He is currently an associate professor with Dalian University of Technology, China.

His research interests include nonlinear algorithm analysis and intelligent information processing.

E-mail: lizx@dlut.edu.cn