An Energy Function-Based Optimization of Matching Parameters and Reference Vectors in SOR Network

Hideaki Misawa and Takeshi Yamakawa
Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology
2-4 Hibikino, Wakamatsu-ku, Kitakyushu 808-0196, Japan
email: misawa-hideaki@edu.brain.kyutech.ac.jp, yamakawa@brain.kyutech.ac.jp

Keywords: self-organizing relationship network, energy function, tuning mode

***Abstract*—** In this paper we propose an energy function-based optimization method in order to improve the approximation ability of the self-organizing relationship (SOR) network. In the execution mode, the SOR network can be used as a fuzzy inference engine. The output of the SOR network is calculated by using the reference vectors and *matching parameters*. The matching parameters, which correspond to the standard deviation of the Gaussian membership function used in fuzzy inference, are only defined in the execution mode. However, the issue of the optimization of the matching parameters has not yet been treated in previous works. To optimize the matching parameters, we introduce an energy function to the SOR network. The energy function can be used not only to tune the matching parameters but also to fine-tune the reference vectors with a gradient descent method. The proposed method is applied to a function approximation problem and the improvement of the approximation ability is confirmed.

1 Introduction

The self-organizing relationship (SOR) network [1][2] was proposed as an extension of the self-organizing map (SOM) [3][4] in order to extract a desirable input-output relationship of a target system using learning vectors with their evaluations. Some models related to the SOM such as the learning vector quantization (LVQ) and supervised SOM [4] are intended for pattern classification and employ supervised learning. On the other hand, the SOR network is designed to implement a given input-output mapping, where the learning is achieved by "learning with evaluation". The input-output data pairs of the target system can be subjectively evaluated by intuition of designers or objectively evaluated by mathematical evaluation functions. Not only desirable data with a good evaluation but also undesirable data with a bad evaluation are utilized as learning data. Therefore, it is especially effective to employ the SOR networks in the cases where it is difficult or costly to acquire the desirable data of the target system, that is, teaching output data. The SOR network has been successfully applied to power system stabilization [1], trailer-truck back-up control [5], image enhancement [6], etc.

The operation of the SOR network is divided into two modes, the learning mode and the execution mode. In the learning mode, the SOR network extracts the desirable input-output relationship of the target system as reference vectors. In the execution mode, the SOR network can be used as a fuzzy inference engine. The execution mode of the SOR network corresponds to a simplified fuzzy inference method [7]. *Matching parameters*, which correspond to the standard deviation of the Gaussian membership function used in fuzzy inference, are only defined in the execution mode in order to calculate similarity measures between an input data vector and reference vectors. Although the output of the SOR network depends on the matching parameters, the optimization of the matching parameters has not yet been discussed in previous papers.

In this paper we propose an energy function-based optimization method in order to improve the approximation ability of the SOR network. In the fuzzy inference scheme, the optimization of parameters of membership functions is carried out by minimizing errors between inference outputs and teaching outputs [8][9]. However, since the SOR network employs learning with evaluation which does not need teaching outputs but evaluations of input-output data pairs, the error potentials typically used in supervised learning cannot be used. In the proposed method, an energy function is introduced to optimize matching parameters. The energy function can be used not only to tune the matching parameters but also to fine-tune the reference vectors with a gradient descent method. To confirm the usefulness of the proposed method, the proposed method is applied to a function approximation problem.

2 Self-organizing Relationship network

The SOR network consists of an input layer, an output layer and a competitive layer as shown in Figure 1. The input and output layers can accept input data vectors of n elements and output data vectors of m elements, respectively. The competitive layer has N units, each of which is characterized by a reference vector $\boldsymbol{v}_j = (\boldsymbol{w}_j, \boldsymbol{u}_j)$ which consists of an input reference vector $\boldsymbol{w}_j$ and an output reference vector

Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007)
Published by the Neuroinformatics Group, Bielefeld University, Germany, ISBN 978-3-00-022473-7
All contributions to WSOM 2007 are available online at: http://biecoll.ub.uni-bielefeld.de

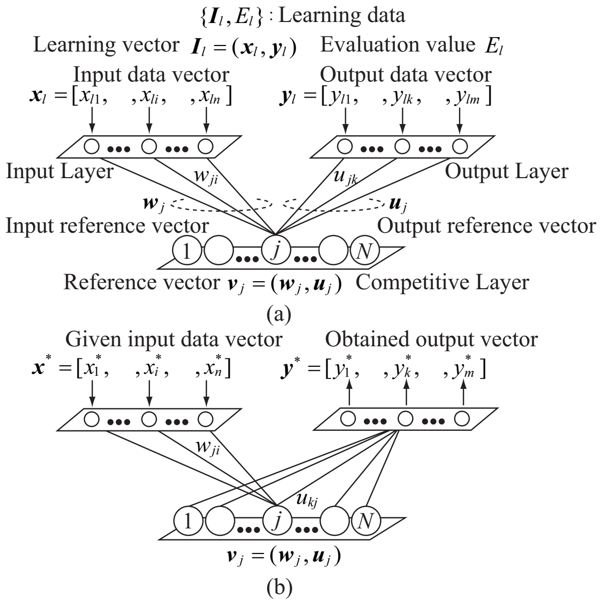

$\{\boldsymbol{I}_l, E_l\}$: Learning data
Learning vector $\boldsymbol{I}_l = (\boldsymbol{x}_l, \boldsymbol{y}_l)$
Evaluation value E_l
Input data vector
$\boldsymbol{x}_l = [x_{l1}, \ , x_{li}, \ , x_{ln}]$
Output data vector
$\boldsymbol{y}_l = [y_{l1}, \ , y_{lk}, \ , y_{lm}]$
Input Layer
w_{ji}
u_{jk}
Output Layer
$\boldsymbol{w}_j$
$\boldsymbol{u}_j$
Input reference vector
Output reference vector
1
j
N
Reference vector $\boldsymbol{v}_j = (\boldsymbol{w}_j, \boldsymbol{u}_j)$
Competitive Layer
(a)
Given input data vector
Obtained output vector
$\boldsymbol{x}^* = [x_1^*, \ , x_i^*, \ , x_n^*]$
$\boldsymbol{y}^* = [y_1^*, \ , y_k^*, \ , y_m^*]$
w_{ji}
u_{kj}
1
j
N
$\boldsymbol{v}_j = (\boldsymbol{w}_j, \boldsymbol{u}_j)$
(b)

Figure 1: Architecture of SOR network. (a) Learning mode. (b) Execution mode.

$\boldsymbol{u}_j$. The network can be established by learning in order to approximate a desired function $\boldsymbol{y} = f(\boldsymbol{x})$, where $\boldsymbol{x}$ is an input vector and $\boldsymbol{y}$ is an output vector.

2.1 Learning mode of the SOR network

In the learning mode, learning vectors $\boldsymbol{I}_l = (\boldsymbol{x}_l, \boldsymbol{y}_l)$ with their respective evaluation value E_l are applied to the input and output layers as shown in Figure 1(a). The evaluation value E_l is assigned by the network designer, given by intuition of the user or obtained by examining the system under test. The value of E_l ranges from -1 to 1. Positive and negative evaluation values represent good and bad evaluations, respectively. The reference vectors are attracted to the learning vectors with a positive evaluation value, i.e. ”attractive learning”, whereas the reference vectors are repulsed from the learning vectors with a negative evaluation value, i.e. ”repulsive learning”. In this paper, the batch learning algorithm proposed in [5] was used. The learning algorithm of the SOR network is summarized as follows.

Step 0. All reference vectors $\boldsymbol{v}_j = (\boldsymbol{w}_j, \boldsymbol{u}_j)$ $(j = 1, 2, \ldots, N)$ are initialized by random numbers.

Step 1. The best matching unit l^* for each learning vector $\boldsymbol{I}_l = (\boldsymbol{x}_l, \boldsymbol{y}_l)$ is chosen as a unit with the smallest Euclidean distance.

$$l^* = \arg \min_j \|\boldsymbol{I}_l - \boldsymbol{v}_j\|. \tag{1}$$

Step 2. A Gaussian neighborhood function is calculated as follows:

$$h(l^*, j) = \exp\left(-\frac{\|\boldsymbol{r}_j - \boldsymbol{r}_{l^*}\|^2}{2\sigma^2(t)}\right) \tag{2}$$

where $\boldsymbol{r}_j$ and $\boldsymbol{r}_{l^*}$ are the positions of the unit j and the best matching unit l^* on the competitive layer and $\sigma(t)$ is the width of the neighborhood function and decreases with the learning step t.

Step 3. Updating value of each reference vector is calculated as follows.

For attractive learning ($E_l \geq 0$):

$$\Delta \boldsymbol{v}_{j,l}^p = \alpha(t) \cdot h(l^*, j) \cdot E_l^p \cdot (\boldsymbol{I}_l - \boldsymbol{v}_j), \tag{3}$$

For repulsive learning ($E_l < 0$):

$$\begin{aligned}\Delta \boldsymbol{v}_{j,l}^n &= \beta(t) \cdot h(l^*, j) \cdot E_l^n \cdot \exp\left(-\frac{\|\boldsymbol{I}_l - \boldsymbol{v}_j\|}{\sigma_r}\right) \\ &\quad \times \frac{(\boldsymbol{I}_l - \boldsymbol{v}_j)}{\|\boldsymbol{I}_l - \boldsymbol{v}_j\|},\end{aligned} \tag{4}$$

where $\Delta \boldsymbol{v}_{j,l}^p$ and $\Delta \boldsymbol{v}_{j,l}^n$ are the updating values for the l-th learning vector with a positive evaluation value E_l^p and a negative evaluation value E_l^n, respectively, $\alpha(t)$ and $\beta(t)$ are the learning rates for the attractive learning and for the repulsive learning, respectively and σ_r is the parameter that decides the extent of the repulsive effect.

Step 4. Each reference vector is updated according to

$$\begin{aligned}\boldsymbol{v}_j(t+1) &= \boldsymbol{v}_j(t) \\ &+ \frac{\sum_{l=1}^{L_p} \Delta \boldsymbol{v}_{j,l}^p + \sum_{l=1}^{L_n} \Delta \boldsymbol{v}_{j,l}^n}{\sum_{l=1}^{L_p} \alpha(t) \cdot h(l^*, j) \cdot E_l^p + \sum_{l=1}^{L_n} \beta(t) \cdot h(l^*, j) \cdot |E_l^n|}\end{aligned} \tag{5}$$

where L_p, L_n are the numbers of the learning vectors with positive and negative evaluation values, respectively. All updating values given by Eq. (3) and (4) are accumulated and normalized in this process.

Step 5. Steps 1 to 4 are repeated with decreasing $\boldsymbol{\theta}(t) = (\alpha(t), \beta(t), \sigma(t))$ monotonically. Usually, these values are calculated at each learning step t by the following equation.

$$\boldsymbol{\theta}(t) = (\boldsymbol{\theta}(0) - \boldsymbol{\theta}(t_{\max})) \cdot \exp(-\frac{t}{\boldsymbol{\tau}}) + \boldsymbol{\theta}(t_{\max}) \tag{6}$$

where $t_{\max}$ is the number of learning iterations and $\boldsymbol{\tau} = (\tau_\alpha, \tau_\beta, \tau_\sigma)$ is decay rate. When $\sigma(t_{\max})$ is very small, the input-output relationship obtained from the learning vectors is finely represented as the reference vectors, whereas when $\sigma(t_{\max})$ is relatively large, the reference vectors roughly approximate the input-output relationship.

2.2 Execution mode of the SOR network

After learning, the SOR network is ready to be used as a fuzzy inference engine as shown in Figure 1(b). An output vector of the SOR network for a given input data vector

$\boldsymbol{x}^* = [x_1^*, \ldots, x_i^*, \ldots, x_n^*]$ is calculated as follows. Soft-matching degree z_j is calculated from the function with respect to the distance between $\boldsymbol{x}^*$ and the input reference vector $\boldsymbol{w}_j$:

$$z_j = \exp\left(-\frac{\|\boldsymbol{x}^* - \boldsymbol{w}_j\|^2}{2\gamma_j^2}\right) \tag{7}$$

where the Gaussian function is used to transform the distance, i.e. dissimilarity measure into similarity measure. We refer to this transform function as *matching function* and to the parameter γ_j as *matching parameter*. A suitable matching function is one that corresponds to the membership function in fuzzy inference, e.g. an exponential function or a triangular function. The matching parameter γ_j represents fuzziness of similarity. In [1], the constant matching parameter was used for all matching functions, whereas in [5], the matching parameters were assigned to the average distance between the reference vector $\boldsymbol{w}_j$ and the nearest other reference vectors as follows:

$$\gamma_j = \frac{1}{H}\sum_{h=1}^{H} \|\boldsymbol{w}_j - \boldsymbol{w}_j^{(h)}\| \tag{8}$$

where $\boldsymbol{w}_j^{(h)}$ is the h-th nearest reference vector according to the distance to the reference vector $\boldsymbol{w}_j$. H is the number of the reference vectors used for average calculation and is assigned empirically in consideration of the distribution of the reference vectors.

The obtained output vector $\boldsymbol{y}^* = [y_1^*, \ldots, y_k^*, \ldots, y_m^*]$ is the weighted average of the output reference vectors by the soft-matching degrees. Thus, the k-th element of $\boldsymbol{y}^*$ is given by

$$y_k^* = \frac{\sum_{j=1}^{N} z_j \cdot u_{kj}}{\sum_{j=1}^{N} z_j}. \tag{9}$$

3 Energy Function-Based Optimization Method

In this section, an energy function-based optimization method is described. Since the output of the SOR network is calculated using Eq. (7) and (9), the approximation ability of the SOR network depends on the matching parameters. An energy function is introduced to the SOR network in order to improve the approximation ability.

3.1 Definition of Energy Function

Let us consider the case that the SOR network has a scalar output. Energy functions for learning data with positive and negative evaluation values are chosen to be represented by

$$J_l^p = \frac{1}{2}E_l^p \cdot (y_l - y_l^*)^2 \quad for\ E_l \geq 0 \tag{10}$$

$$J_l^n = \varepsilon \cdot |E_l^n| \cdot \exp\left(-\frac{|y_l - y_l^*|}{\sigma_r}\right) \quad for\ E_l < 0 \tag{11}$$

where y_l is the output data of the l-th learning vector and y_l^* is an obtained output for the l-th input data vector. ε and σ_r are the parameters that determine the height and width of the energy function. The evaluation values E_l^p, E_l^n represent a degree of contribution to the energy. These energy functions are based on the concepts of attractive learning and repulsive learning. As the obtained output comes close to the output data with a positive evaluation value, the energy becomes small and vice versa. On the other hand, as the obtained output comes close to the output data with a negative evaluation value, the energy becomes large and vice versa. The energy can be interpreted as a degree of undesirability of the obtained output. These energy functions are used to optimize the matching parameters and the reference vectors. The optimization of the parameters is achieved by reducing the energy

$$J = \sum_{l=1}^{L_p} J_l^p + \sum_{l=1}^{L_n} J_l^n. \tag{12}$$

3.2 Modification of the execution mode

In the execution mode, the soft-matching degree z_j is calculated by Eq. (7). To improve the representation ability of the SOR network, the calculation of the soft-matching degree is modified as follows:

$$z_{ji} = \exp\left(-\frac{(x_i^* - w_{ji})^2}{2\gamma_{ji}^2}\right), \tag{13}$$

$$z_j = \prod_{i=1}^{N} z_{ji} \tag{14}$$

where z_{ij} is the soft-matching degree between i-th element of $\boldsymbol{x}^*$ and i-th element of $\boldsymbol{w}_j$. The matching parameter γ_{ji} is determined with the following tuning method.

3.3 Tuning mode of the SOR network

In order to obtain the accurate input-output relationship, the operation of the SOR is extended from two modes to three modes. After the learning mode, the matching parameters and reference vectors are tuned with a gradient descent method using the learning data. We refer to this procedure as *tuning mode*. The main purpose of the tuning mode is to tune the matching parameters according to the energy gradient since the matching parameters are not determined during learning. The initial value of the matching parameter γ_{ji} is assigned to the average distance calculated by Eq. (8). The reference vectors are also fine-tuned in a manner taking into account the output of the SOR network. In the learning of the SOM, reference vectors after learning approximately represent the distribution of the input data vectors. The learning of the SOR network inherits this property. The reference vectors approximate the distribution of the learning vectors with a positive evaluation, avoiding the learning vectors with a negative evaluation. However, the learning of the SOR network does not take into consideration the output calculated by Eq. (9).

The matching parameters and reference vectors are tuned after each learning data is presented, according to the energy gradient as follows:

$$\mathbf{\Psi}(s+1) = \mathbf{\Psi}(s) - \eta \cdot \frac{\partial J_l^p}{\partial \mathbf{\Psi}(s)} \quad for\ E_l \geq 0 \qquad (15)$$

$$\mathbf{\Psi}(s+1) = \mathbf{\Psi}(s) - \eta \cdot \frac{\partial J_l^n}{\partial \mathbf{\Psi}(s)} \quad for\ E_l < 0 \qquad (16)$$

where $\mathbf{\Psi}(s) = (w_{ji}(s), u_j(s), \gamma_{ji}(s))$, s is the tuning step that is increased each time one of the learning data is presented, η is the tuning rate and sgn$(\cdot)$ represents the sign function. The partial derivatives are calculated as follows:

For learning data with a positive evaluation ($E_l \geq 0$):

$$\begin{aligned} \frac{\partial J_l^p}{\partial u_j} &= -E_l^p \cdot (y_l - y_l^*) \cdot \frac{z_j}{\sum_{i=1}^N z_i}, && (17) \\ \frac{\partial J_l^p}{\partial \gamma_{ji}} &= -\frac{\partial J_l^p}{\partial u_j} \cdot (y_l^* - u_j) \cdot \frac{(x_{li} - w_{ji})^2}{\gamma_{ji}^3}, && (18) \\ \frac{\partial J_l^p}{\partial w_{ji}} &= -\frac{\partial J_l^p}{\partial u_j} \cdot (y_l^* - u_j) \cdot \frac{(x_{li} - w_{ji})}{\gamma_{ji}^2}. && (19) \end{aligned}$$

For learning data with a negative evaluation ($E_l < 0$):

$$\begin{aligned} \frac{\partial J_l^n}{\partial u_j} &= \varepsilon \cdot |E_l^n| \cdot \frac{\mathrm{sgn}(y_l - y_l^*)}{\sigma_r} \\ &\quad \cdot \exp(-\frac{|y_l - y_l^*|}{\sigma_r}) \frac{z_j}{\sum_{i=1}^N z_i}, && (20) \\ \frac{\partial J_l^n}{\partial \gamma_{ji}} &= \frac{\partial J_l^n}{\partial u_j} \cdot (u_j - y_l^*) \cdot \frac{(x_{li} - w_{ji})^2}{\gamma_{ji}^3}, && (21) \\ \frac{\partial J_l^n}{\partial w_{ji}} &= \frac{\partial J_l^n}{\partial u_j} \cdot (u_j - y_l^*) \cdot \frac{(x_{li} - w_{ji})}{\gamma_{ji}^2}. && (22) \end{aligned}$$

The tuning process is repeated until a termination criterion is met.

4 Simulation Results

In order to verify the proposed method, it was applied to the approximation of the following function:

$$\begin{aligned} y &= 0.3\cos(\pi x_1) + 0.3\cos(\pi x_2) \\ &+ 0.8\exp(-30(x_1^2 + x_2^2)) - 0.4 \quad x_1, x_2 \in [-1, 1]. \end{aligned} \qquad (23)$$

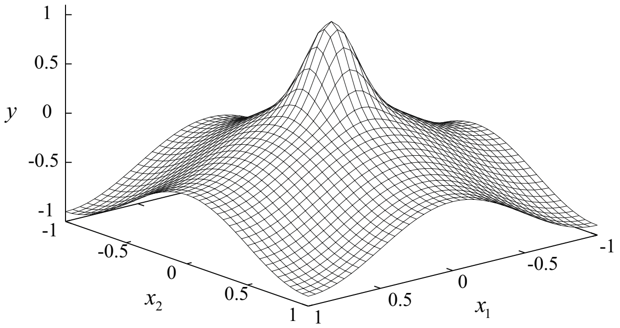


Figure 2: Target function.

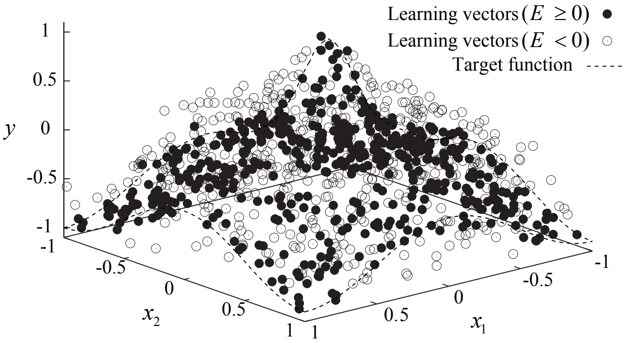
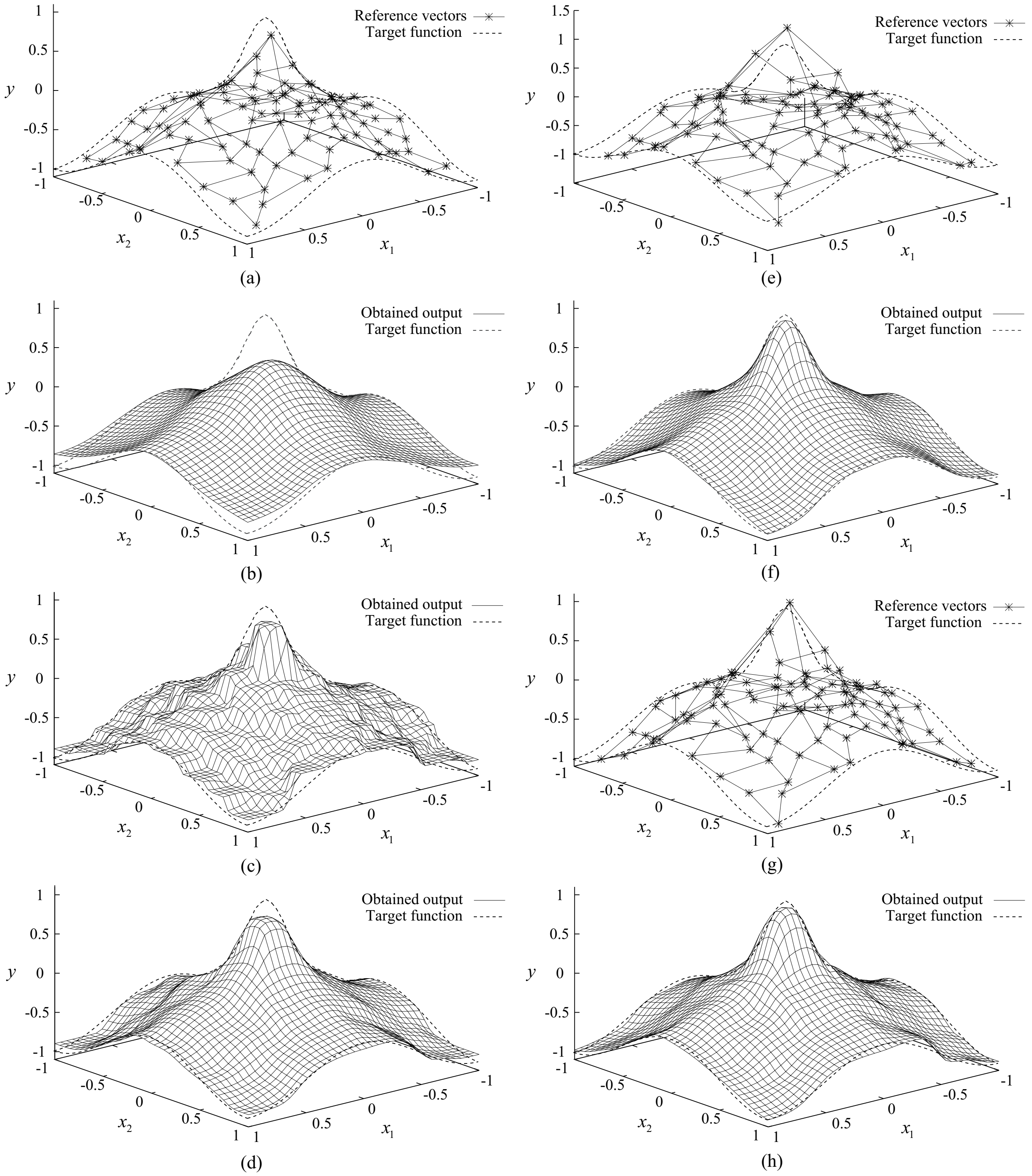


Figure 3: Learning vectors. Filled circles and open circles indicate the learning vectors with positive and negative evaluation values, respectively.

The target function and the distribution of the learning vectors are presented in Figure 2 and 3, respectively. The learning vectors consist of 512 learning vectors with a positive evaluation value (denoted by filled circles) and 512 learning vectors with a negative evaluation value (denoted by open circles). The evaluation value E_l of the l-th learning vector $\boldsymbol{I}_l = (\boldsymbol{x}_l, \boldsymbol{y}_l)$ is calculated by the following equation:

$$E_l = -1 + 2\exp(-5d_l) \qquad (24)$$

where d_l is the distance between the output of the target function and the output data y_l of the learning vector. In the learning, the number of learning iterations $t_{\max} = 200$, the number of units on the competitive layer $N = 100(10 \times 10)$, $\sigma_r = 0.05$, $\alpha(0) = 1.0$, $\alpha(t_{\max}) = 0.01$, $\beta(0) = 0.2$, $\beta(t_{\max}) = 0.001$, $\sigma(0) = 10$, $\sigma(t_{\max})$=0.01, $\tau_\alpha = 30, \tau_\beta = 30, \tau_\sigma = 30$. In the tuning, the tuning rate $\eta = 0.01$ and the number of tuning epochs was 1000. The distribution of the reference vectors after the learning mode is presented in Figure 4(a). Almost all reference vectors were placed in the regions where the learning vectors with a high positive evaluation value were distributed. However, the sharpness of the central region was not represented by the reference vectors. A few reference vectors were placed around the central region since the number of the learning vector around the peak of the target function was smaller than other regions.



(a) (e)

(b) (f)

(c) (g)

(d) (h)

Figure 4: Simulation Results. (a) Distribution of the reference vectors after the learning mode. (b) Obtained output with the matching parameters assigned by the average distance with $H = 2$. (c) Obtained output with the constant matching parameter $\gamma_{ji} = 0.05$. (d) Obtained output after tuning of the matching parameters. (e) Distribution of the reference vectors after tuning of the reference vectors. Note that the scale of y-axis is different from other graphs. (f) Obtained output after tuning of the reference vectors. (g) Distribution of the reference vectors after tuning of both the reference vectors and matching parameters. (h) Obtained output after tuning of both the reference vectors and matching parameters.

Table 1: Energy and RMSE comparisons.

Method	Energy J	RMSE
Average distance	1.256	0.0824
Constant($\gamma_{ji} = 0.05$)	0.592	0.0612
Tuning: matching parameters	0.561	0.0493
Tuning: reference vectors	0.607	0.0436
Tuning: both	0.376	0.0352

The links between the reference vectors represent the neighboring relationships of the units on the competitive layer. The obtained output with the matching parameters that were assigned to the average distance are shown in Figure 4(b). H was assigned to be 2. In this case, the obtained output was very smooth and did not represent the sharpness around the central region. The obtained output with the constant matching parameter $\gamma_{ji} = 0.05$ is shown in Figure 4(c). The sharpness around the central region was relatively well represented. However, the output was not smooth over the whole regions. To obtain the output with both the smoothness and the sharpness, each matching function needs to have an optimal matching parameter. The obtained output after the tuning of the matching parameters is shown in Figure 4(d). In this case, the reference vectors were fixed and only the matching parameters were tuned with Eq. (15) and (16). The smoothness and the sharpness of the target function were represented except the peak region where the reference vectors did not exist. The distribution of the reference vectors and the obtained output after the tuning of the reference vectors are shown in Figure 4(e) and (f), respectively. The reference vectors around the central region were moved greatly from the initial positions indicated in Figure 4(a) to reduce the energy. The obtained output represented the input-output relationship of the target function well. The distribution of the reference vectors and the obtained output after the tuning of both the reference vectors and matching vectors are shown in Figure 4(g) and (h), respectively. The movements of the reference vectors were smaller than those in Figure 4(e). Nevertheless, the obtained output approximated the input-output relationship of the target function well.

To compare the proposed method with ones previously used, the energy J for the learning data and root-mean-square error (RMSE) for 1600 test input vectors were calculated. The averages of the energy J and RMSEs in ten trials are shown in Table 1. The tuning of both the reference vectors and matching parameters resulted in the smallest energy and the smallest RMSE. The results indicate that the proposed method outperforms the methods previously used.

5 Conclusions

In this paper we proposed an energy function-based optimization method in order to improve the approximation ability of the SOR network. The operation of the SOR network was extended from two modes to three modes. The matching parameters and reference vectors are tuned with a gradient descent using the learning data in the tuning mode. The proposed method was verified through a function approximation problem. The smallest energy and the smallest RMSE were obtained by tuning the reference vectors and matching parameters according to the proposed energy function. In future work, it is necessary to apply the proposed method to practical applications to verify the effectiveness.

Acknowledgements

This work was supported by a COE program (center #J19) granted to Kyushu Institute of Technology by MEXT of Japan. This work is also supported by Grant-in-Aid for Scientific Research (A), 2006, 18200015.

References

[1] T. Yamakawa and K. Horio, “Self-Organizing Relationship (SOR) Network,” *IEICE Transaction on Fundamentals*, Vol. E82-A, No. 8, pp. 1674-1677, 1999.

[2] T. Yamakawa and K. Horio, Japanese Patent, No. 2000-1229991A (Appl. 15 Oct., 1998).

[3] T. Kohonen, “Self-organized formation of topologically correct feature maps,” *Biological Cybernetics*, Vol.43, pp.59- 69, 1982.

[4] T. Kohonen, *Self-Organizing Maps* (3rd ed.), Springer, 2001.

[5] T. Koga, K. Horio and T. Yamakawa, “The Self-Organizing Relationship (SOR) network employing fuzzy inference based heuristic evaluation,” *Neural Networks*, Vol.19, pp.799-811, 2006.

[6] K. Horio, T. Haraguchi and T. Yamakawa, “An intuitive contrast enhancement of an image data employing the self-organizing relationship (SOR),” *Proc. of international joint conference on neural networks*, pp. 10-16, 1999.

[7] M. Maeda, Z. Murakami and S. Murakami, “The Designs of Fuzzy Logic Controller (in Japanese),” *Bulletin of the Kyushu Institute of Technology. Science and technology*, Vol. 49, pp. 43-51, 1984.

[8] H. Nomura, I. Hayashi and N. Wakami, “A self-tuning method of fuzzy control by descent method,” *Proc. of the 4th International Fuzzy Systems Association World Congress*, vol. Eng., pp. 155-158, 1991.

[9] H. Ichihashi, “Iterative fuzzy modeling and a hierarchical network,” *Proc. of the 4th International Fuzzy Systems Association World Congress*, vol. Eng., pp. 49-52, 1991.