DE GRUYTER OPEN

JAISCR, 2015, Vol. 5, No. 4, pp. 271– 282
10.1515/jaiscr-2015-0035

PERFORMANCE COMPARISON OF HYBRID ELECTROMAGNETISM-LIKE MECHANISM ALGORITHMS WITH DESCENT METHOD

Hirofumi Miyajima, Noritaka Shigei, Hiromi Miyajima

Kagoshima University, 1-21-40 Korimoto
Kagoshima, Japan

Abstract

Electromagnetism-like Mechanism (EM) method is known as one of metaheuristics. The basic idea is one that a set of parameters is regarded as charged particles and the strength of particles is corresponding to the value of the objective function for the optimization problem. Starting from any set of initial assignment of parameters, the parameters converge to a value including the optimal or semi-optimal parameter based on EM method. One of its drawbacks is that it takes too much time to the convergence of the parameters like other meta-heuristics. In this paper, we introduce hybrid methods combining EM and the descent method such as BP, *k*-means and FIS and show the performance comparison among some hybrid methods. As a result, it is shown that the hybrid EM method is superior in learning speed and accuracy to the conventional methods.

1 Introduction

Metaheuristics are generally applied to problems for which there are few effective problem-oriented algorithms to solve them. They are widely used to solve complex problems in science, engineering and so on [1]. Well-known metaheuristics are random search (RS) by Matyas (1965), the simulated annealing (SA) by Kirkpatrick (1982), genetic algorithm (GA) by Goldberg (1989), bee colonies (BC) by Walker (1993), particle swarm optimization (PSO) by Kennedy (1995), electromagnetism-like mechanism (EM) (2005) and so on [1, 2, 3, 5]. Any of those methods is not always universal, so problem-oriented algorithm must be selected for each problem. The difficult points are to search effectively vast space to find the optimal or semi-optimal solution of the objective function [1]. The criterion to select any method of metaheuristics is how effective the search of the solution is performed by using global and local searches. It is well known that local search takes too much time [1, 4]. In order to construct the effective algorithm, a fast local search with high accuracy is required. Therefore, though many hybrid methods combining metaheuristics and BP or FIS method are introduced, the satisfactory method is not always obtained [7, 8, 9, 15, 16, 17, 18]. RS, PSO and EM methods are well known as metaheuristics and their convergence to a solution for algorithms is guaranteed [2, 5, 7]. EM method is more complicate than RS, but is fewer in the number of parameters than PSO. Further, it is known that PSO and EM methods have the same capability in accuracy [6] and they are superior in accuracy to RS method. On the other hand, a lot of algorithms based on the descend method are known such as BP, vector quantization and FIS and some hybrid methods have been proposed [15, 16]. However, the general hybrid methods combining EM and the descent method are not introduced.

In this paper, we introduce hybrid methods combining EM method and steepest descent methods including BP and show the performance comparison among some hybrid methods. EM is known as one of random search algorithms [5, 6, 10, 11]. The basic idea is that a set of parameters is regarded as position of charged particles and the charge of the particles is corresponding to the value of the objective function for the optimization problem. First, a plural of charged particles are distributed in the domain and the update of global and local searches based on the strength of the charge is repeated. The problem is one that it takes too much time to convergence of the parameters like other random search methods. In order to improve them, hybrid EM methods combining EM with the descend methods such as BP, *k*-means and FIS as local search are introduced and are compared with the conventional methods. Though we assume to use BP, *k*-means and FIS methods, we can also consider hybrid EM method based on other steepest descent methods [12]. In the section 2, the conventional method of metaheuristics as EM and RS, BP, *k*-means and FIS are introduced. In the section 3, some hybrid EM methods are introduced. In the section 4, the results of some numerical simulations are shown and the effectiveness of hybrid EM methods with descent method is demonstrated.

2 Preliminaries

Let $\boldsymbol{R}$ be the set of all real numbers. Let $Z_i = \{1, \cdots, i\}$ for positive integer i. Let $J = [0, 1]$ be the set of all real numbers between 0 and 1. The problem is finding an optimal or semi-optimal solution of a non-linear optimization problem with boundary variables in the form;

$$\min_{\boldsymbol{L} \leq \boldsymbol{x} \leq \boldsymbol{U}} f(\boldsymbol{x}) \tag{1}$$

where $f(\boldsymbol{x})$ is the evaluation function to be minimized, $\boldsymbol{x} \in \boldsymbol{R}^n$ is the variable vector (parameter), and $\boldsymbol{L} = (l_1, \cdots, l_n)$ and $\boldsymbol{U} = (u_1, \cdots, u_n)$ are the lower and upper bounds of $\boldsymbol{x}$, respectively.

For example, let us consider a neural network. Given learning data, let us determine the weights of neural network identifying the learning data. In the case, the mean square error between learning data and output of the network corresponds to evaluation function and the weights of the network corresponds to parameters.

In general, metaheuristics provide how to find the optimum or semi-optimum solution effectively. They consist of global and local searches. In the following, some methods used for local and global searches are introduced.

2.1 EM method

Algorithm EM(m, MAX, LS, δ)

```
{$\boldsymbol{x}^i | i \in Z_m$}: the set of parameters
$MAX$: maximum number of iterations
$LS$: maximum number of iterations for local search
$\delta$: local search parameter, $\delta \in J$
1: Initialize()
2: $iteration \leftarrow 1$
3: while $iteration < MAX$ do
4:     Local($LS$, $\delta$)
5:     $F \leftarrow$ Calc F()
6:     Move($F$)
7:     $iteration \leftarrow iteration + 1$
8: end while
```

Figure 1. Outline of EM method

EM algorithm simulates the interaction, attraction and repulsion, caused by electromagnetic force between electrically charged particles. The general scheme of EM method is shown as Fig.1 [5]. It consists of four phases. In the step 1, a set of particles is initialized. In the step 4, local search to find local optimum is performed. In the step 5, the force worked on each particle is calculated. In the step 6, each particle moves in the direction of the compounded force.

In the Initialization step, the sample points, MAX, LS, and δ are given. Fig. 2 shows the algorithm of local search of the step 4 in Fig.1 [5].

Algorithm Local(LS, δ)

```
$u_k$: the $k$-th element of upper bound for $k \in Z_n$
$l_k$: the $k$-th element of lower bound for $k \in Z_n$
U(0,1): standard uniform distribution in $[0,1]$
1: $counter \leftarrow 1$
2: $Length \leftarrow \delta\ (\max_k\{u_k - l_k\})$
3: for $i$ = 1 to $m$ do
4:  for $k$ = 1 to $n$ do
5:     $\lambda_1 \leftarrow$ U (0, 1)
6:     while $counter < LS$ do
```

```
7:        $y \leftarrow x^i$
8:        $\lambda_2 \leftarrow$ U (0, 1)
9:        if $\lambda_1 > 0.5$ then
10:           $y_k \leftarrow y_k + \lambda_2(Length)$
11:        else
12:           $y_k \leftarrow y_k - \lambda_2(Length)$
13:        end if
14:        if $f(y) < f(x^i)$ then
15:           $x^i \leftarrow y$
16:           $counter \leftarrow LS$ - 1 17:        end if
18:        $counter \leftarrow counter$ + 1
19:     end while
20:   end for
21: end for
22: $x^{best} \leftarrow \arg\min_{x^i}\{f(x^i)\}$
```

Figure 2. Local search of EM algorithm

In the algorithm Local(LS, δ), a neighbor point $\boldsymbol{y}$ for the parameter $\boldsymbol{x}$ is given and $f(\boldsymbol{x})$ is compared with $f(\boldsymbol{y})$. In the case of $f(\boldsymbol{y}) < f(\boldsymbol{x})$, $\boldsymbol{x}$ is set to $\boldsymbol{y}$. In other case, the same process for other neighbor $\boldsymbol{y'}$ is repeated. After local searches for all parameters are performed, $\boldsymbol{x}^{best}$ is updated.

Figs.3 and 4 show the algorithm of global search that the new position of each particle in the electromagnetic theory is computed [5].

Algorithm Calc F()

$||\boldsymbol{a}-\boldsymbol{b}||$: the distance between $\boldsymbol{a}$ and $\boldsymbol{b}$, where $||\boldsymbol{a}|| = \sqrt{\sum_{i=1}^{m} a_i^2}$ for $\boldsymbol{a} = (a_1, \cdots, a_n)$

```
1: for $i$=1 to $m$
2: $q_i \leftarrow \exp\left(-n\frac{(f(x^i)-f(x^{best}))}{\sum_{k=1}^{n}(f(x^k)-f(x^{best}))}\right)$
3: $F^i \leftarrow 0$
4: end for
5: for $i$=1 to $m$
6:   for $j$=0 to $m$
7:     if $f(x^i) < f(x^j)$ then
8:       $F^i \leftarrow F^i + (x^i - x^j)\frac{q^i q^j}{||x^i - x^j||^2}$
9:     else
10:      $F^i \leftarrow F^i - (x^i - x^j)\frac{q^i q^j}{||x^i - x^j||^2}$
11:    end if
12:  end for
13: end for
```

Figure 3. The force of each particle in EM

Algorithm Move (F)

```
1: for $i$=1 to $m$
2:   if $i \neq best$ then
3:     $\alpha \leftarrow$ U(0,1)
4:     $F^i \leftarrow \frac{F^i}{||F^i||}$
5:     for $k$=1 to $N$
6:       if $F_k^i > 0$ then
7:         $x_i = x_i + \alpha F_k^i(u_k - x_k^i)$
8:       else
9:         $x_i = x_i + \alpha F_k^i(x_k^i - l_k)$
10:      end if
11:    end for
12:  end if
13: end for
```

Figure 4. The movement of each particle in EM

2.2 Three-layered neural network and BP method

Let us consider the case of n inputs and one output without loss of generality. Let $\boldsymbol{z}^i \in J^n$ for $i \in Z_l$ and $d: J^n \to J$. Giving learning data $\{(\boldsymbol{z}^i, d(\boldsymbol{z}^i)) | i \in Z_l\}$, let us determine the three-layered neural network identifying the learning data by BP method [12]. Let $h = g \circ e : J^n \to J$ be the function defined by a neural network. Let the number of elements in second layer be p. Let $\boldsymbol{w}_j$ for $j \in Z_p$ and $\boldsymbol{v}$ be weights for the second and output layers, respectively. Then g and e are defined as follows (See Fig.5):

$$\begin{aligned} y_j &= e_j(\boldsymbol{z}) = \tau\left(\sum_{i=0}^{n} w_{ij} z_i\right), \\ z_0 &= 1, \\ \tau(u) &= \frac{1}{1+\exp(-u)} \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{z} &= (z_1, \cdots, z_n) \in J^n \\ \boldsymbol{y} &= (y_1, \cdots, y_p) \in J^p \end{aligned}$$

and w_{0j} means the threshold value.

Further,

$$\begin{aligned} g(\boldsymbol{y}) &= \tau\left(\sum_{j=0}^{p} v_j y_j\right), \\ y_0 &= 1, \end{aligned}$$

where v_0 means the threshold value.

Then, the evaluation function is defined as follow:

$$E = \frac{1}{2l}\sum_{i=1}^{l} ||h(\boldsymbol{z}^i) - d(\boldsymbol{z}^i)||^2 \tag{2}$$

The weights $\boldsymbol{w}$ and $\boldsymbol{v}$ are updated based on BP method as follow [12]:

$$\triangle v_i = -\frac{\alpha_1}{l}\sum_{k=1}^{l} \delta_{2k}(\boldsymbol{z}^k)e_i(\boldsymbol{z}^k) \tag{3}$$

$$\triangle w_{ij} = -\frac{\alpha_2}{l}\sum_{k=1}^{l} \delta_{1j}(\boldsymbol{z}^k)z_i^k \tag{4}$$

$$(i = 0, \cdots, p, j = 1, \cdots, n)$$

where α_1 and α_2 are learning coefficients,

$$\delta_{2j}(\boldsymbol{z}) = (h(\boldsymbol{z}) - d(\boldsymbol{z}))h(\boldsymbol{z})(1 - h(\boldsymbol{z}))e_j(\boldsymbol{z}) \tag{5}$$

and

$$\delta_{1j}(\boldsymbol{z}) = \delta_{2j}v_j e_j(\boldsymbol{z})(1 - e_j(\boldsymbol{z})). \tag{6}$$

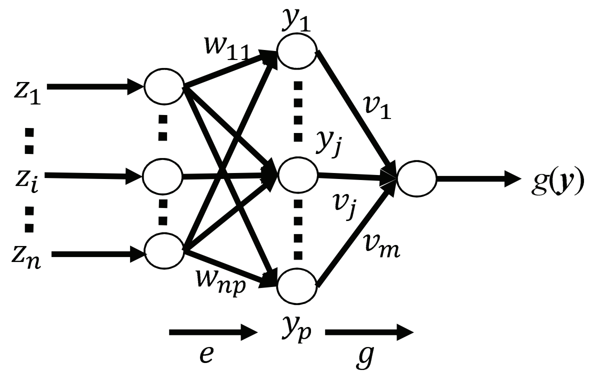


Figure 5. Three-layered neural network

2.3 Clustering by *k*-means method

Vector quantization techniques encode a data space, e.g., a subspace $V \subseteq \boldsymbol{R}^n$, utilizing only a finite set $W = \{\boldsymbol{w}_i | i \in Z_r\}$ of reference vectors (also called cluster centers), which n and r are positive integers. Let us introduce k-means method as one of vector quantization techniques [13].

Let the winner vector $\boldsymbol{w}_{i(\boldsymbol{v})}$ be defined for any vector $\boldsymbol{v} \in V$ as follows:

$$i(\boldsymbol{v}) = \arg\min_{i \in Z_r} ||\boldsymbol{v} - \boldsymbol{w}_i|| \tag{7}$$

From the finite set W, V is partitioned as follows:

$$V_i = \{\boldsymbol{v} \in V | ||\boldsymbol{v} - \boldsymbol{w}_i|| \le ||\boldsymbol{v} - \boldsymbol{w}_j|| \ for \ j \in Z_r\} \tag{8}$$

The evaluation function for the partition is defined as follows:

$$E = \sum_{i=1}^{r}\sum_{\boldsymbol{v} \in V_i} ||\boldsymbol{v} - \boldsymbol{w}_{i(\boldsymbol{v})}||^2 \tag{9}$$

Each parameter $\boldsymbol{w}$ is updated based on the steepest descent method as follow [13]:

$$\triangle \boldsymbol{w}_i = \varepsilon \delta_{ij(\boldsymbol{v}(t))}(\boldsymbol{v}(t) - \boldsymbol{w}_i) \tag{10}$$

where t is the step, ε is learning constant and δ_{ij} is Kronecker Delta. $\boldsymbol{v}(t)$ means data selected from V randomly at step t. The k-means method is denoted by KM.

2.4 Fuzzy inference system and learning algorithm

The conventional fuzzy inference system (FIS) using delta rule is described[12, 15, 19]. Let $\{(\boldsymbol{z}^i, d(\boldsymbol{z}^i))\}$ be the set of learning data. The rule of simplified FIS is expressed as

$$R_j \text{ if } z_1 \text{ is } M_{1j} \text{ and } \cdots z_n \text{ is } M_{nj} \text{ then } y \text{ is } w_j \tag{11}$$

where $j \in Z_c$ is a rule number, $i \in Z_n$ is a variable number, M_{ij} is a membership function of the antecedent part, and w_j is the weight of the consequent part.

A membership value of the antecedent part μ_i for input $\boldsymbol{z}$ is expressed as

$$\mu_j = \prod_{i=1}^{n} M_{ij}(z_i) \tag{12}$$

where M_{ij} is the membership function of the antecedent part. Let c_{ij} and b_{ij} denote the center and the wide values of M_{ij}, respectively.
If Gaussian membership function is used, then M_{ij} is expressed as follow:

$$M_{ij} = \exp\left(-\frac{1}{2}\left(\frac{z_j - c_{ij}}{b_{ij}}\right)^2\right). \tag{13}$$

The output y^* of fuzzy inference is calculated by the following equation.

$$y^* = \frac{\sum_{i=1}^{c}\mu_i \cdot w_i}{\sum_{i=1}^{c}\mu_i} \tag{14}$$

The evaluation function E is defined to evaluate the inference error between the desirable output $y^r\ (= d(\boldsymbol{z}))$ and the inference output y^*.

$$E = \frac{1}{2}\,(y^* - y^r)^2 \tag{15}$$

In order to minimize the objective function E, each parameter $\alpha \in \{c_{ij}, b_{ij}, w_j\}$ is updated based on descent method.

$$\alpha(t+1) = \alpha(t) - K_\alpha \frac{\partial E}{\partial \alpha} \tag{16}$$

where t is iteration time and K_α is a constant.

$$\frac{\partial E}{\partial c_{ij}} = (y^* - y^r)(w_i - y^*)\frac{\mu_i}{\sum_{i=1}^{c}\mu_i}\frac{(z_j - c_{ij})}{b_{ij}^2} \tag{17}$$

$$\frac{\partial E}{\partial b_{ij}} = (y^* - y^r)(w_i - y^*)\frac{\mu_i}{\sum_{i=1}^{c}\mu_i}\frac{(z_j - c_{ij})^2}{b_{ij}^3} \tag{18}$$

$$\frac{\partial E}{\partial w_i} = (y^* - y^r)\frac{\mu_i}{\sum_{i=1}^{c}\mu_i} \tag{19}$$

3 Hybrid EM algorithms

EM method [1] is an effective method of metaheuristics to solve the optimization problem. On the other hand, it takes too much time to converge to an optimum or semi-optimum solution. Therefore, we propose hybrid algorithms combining EM method with the steepest decent methods such as k-means and BP methods.

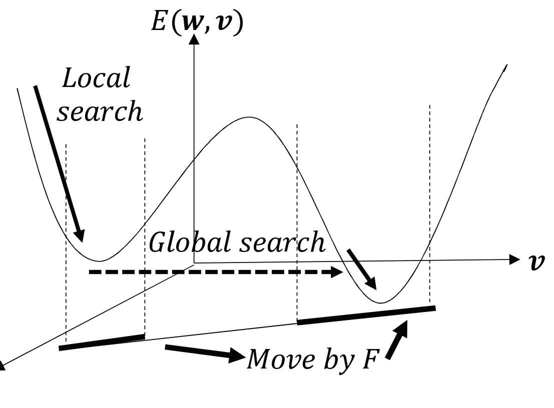


Figure 6. The figure to explain hybrid EM algorithm

Why is the hybrid algorithm needed? Let us explain the reason using EM and BP methods. As shown in Fig.6, the evaluation value $E(\boldsymbol{w}, \boldsymbol{v})$ is determined based on the parameters $\boldsymbol{w}$ and $\boldsymbol{v}$. The initial vectors are selected randomly and the vectors are updated by local search. Since BP method is based on the steepest descent method, $E(\boldsymbol{w}, \boldsymbol{v})$ always decreases. On the other hand, local search in the conventional EM and other methods needs to find a new vector which decreases $E(\boldsymbol{w}, \boldsymbol{v})$ and it takes too much time. Further, if $E(\boldsymbol{w}, \boldsymbol{v})$ does not decrease so much, then the vector moves to a new position by using the force F. Then, the evaluation value $E(\boldsymbol{w}^{new}, \boldsymbol{v}^{new})$ for new parameters, $\boldsymbol{w}^{new}$ and $\boldsymbol{v}^{new}$, is lower than or equal to the evaluation value $E(\boldsymbol{w}^{old}, \boldsymbol{v}^{old})$ for old parameters, $\boldsymbol{w}^{old}$ and $\boldsymbol{v}^{old}$. Continuing the processes, the vector is updated so as to improve the function $E(\boldsymbol{w}, \boldsymbol{v})$. In the case of BP algorithm, it is possible only to search the vectors (parameters) locally and is difficult to search them globally. Hence, global search such as moving by F in the proposed method is needed. Likewise, the other hybrid methods using the steepest descent methods are also interpreted.

3.1 Hybrid EM method with BP

A hybrid EM method with BP (HEM-BP) employs BP method [12] which is one of the steepest descent ones. In HEM-BP, the parameter $\boldsymbol{x}$ of EM method represents a set of the weights $\boldsymbol{w}$ and $\boldsymbol{v}$ for three layered neural network. For the algorithm of HEM-BP, Local(LS, δ) in Fig.1 is replaced with BP procedure BP(LS, ε) whose flowchart is shown in Fig.7, where LS and ε are the number of learning and the threshold for local search, respectively. The BP procedure receives the set of the parameters $X = \{\boldsymbol{x}^i | i \in Z_m\}$ and the best parameters $\boldsymbol{x}^{best}$ in X. According to the procedure, they are updated based on the steepest descent method with the learning data $\{(\boldsymbol{z}_i, d(\boldsymbol{z}_i)) | i \in Z_l\}$ and given to the calling side. HEM-BP involves additional parameters MAX, LS and ε, and it will be denoted as HEM-BP(MAX, LS, ε).

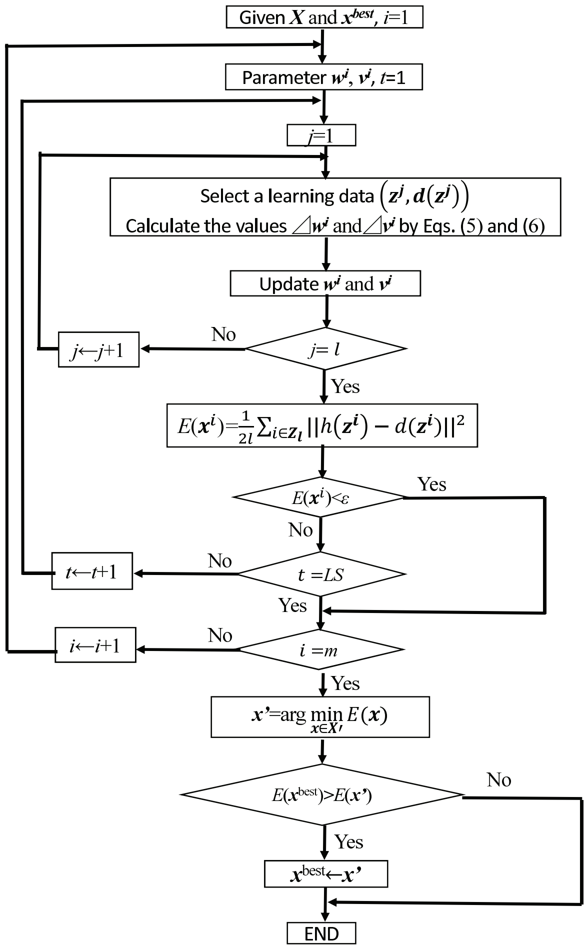


Figure 7. The flowchart of BP(LS, ε)

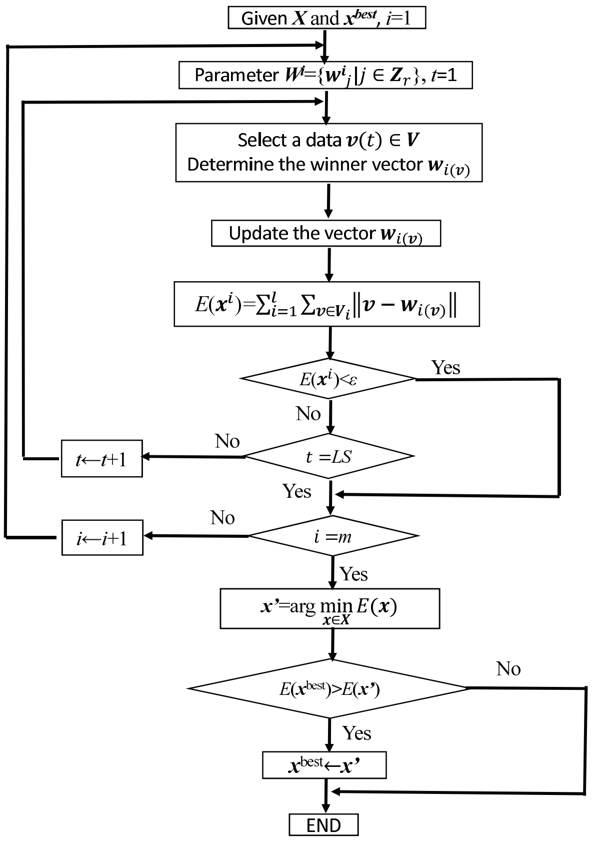


Figure 8. The flowchart of k-means(LS, ε)

3.2 Hybrid EM method for clustering

A hybrid EM method with k-means (HEM-KM) employs k-means method for clustering of data. In HEM-KM, the parameter $\boldsymbol{x}$ of HEM-KM represents the reference vectors $W = \{\boldsymbol{w}_i | i \in Z_r\}$, where r is the number of reference vectors. For the algorithm of HEM-KM, Local(LS, δ) in Fig. 1 is replaced with a k-means procedure KM(LS, ε) whose flowchart is shown in Fig. 8, where LS and ε are the number of learning and the threshold for local search, respectively. The k-means procedure receives the set of parameters $X = \{\boldsymbol{x}^i | i \in Z_m\}$ and the best parameters $\boldsymbol{x}^{best}$ in X from the calling side. According to the procedure, they are updated based on the steepest descent method with the learning data $\{(\boldsymbol{v}(t) \in V \subseteq \boldsymbol{R}^n | t = 1, 2, \cdots, LS\}$ and given to the calling side. HEM-KM involves additional parameters MAX, LS and ε, and it will be denoted as HEM-KM(MAX, LS, ε).

3.3 Hybrid EM method for learning of FIS

A hybrid EM method with FIS (HEM-FIS) employs the learning algorithm based on the descent method for FIS. In HEM-FIS, $X = \{\boldsymbol{x}^i | i \in Z_m\}$ means the set of parameters of FIS, $\boldsymbol{c}$, $\boldsymbol{b}$ and $\boldsymbol{w}$ (or only $\boldsymbol{w}$). For the algorithm of HEM-FIS, Local(LS, ε) in Fig. 1 is replaced with a FIS procedure FIS(LS, ε) whose flowchart is shown in Fig. 9, where LS and ε are the number of learning and the the threshold for local search, respectively. The FIS procedure receives the set of parameters X and the best parameters $\boldsymbol{x}^{best}$ in X from the calling side. According to the procedure, they are updated based on the descent method with the learning data $\{(z_i, d(z_i)) | i \in Z_l\}$ and given to the calling side. HEM-FIS involves additional parameters MAX, LS and ε, and it will be denoted as HEM-FIS(MAX, LS, ε).

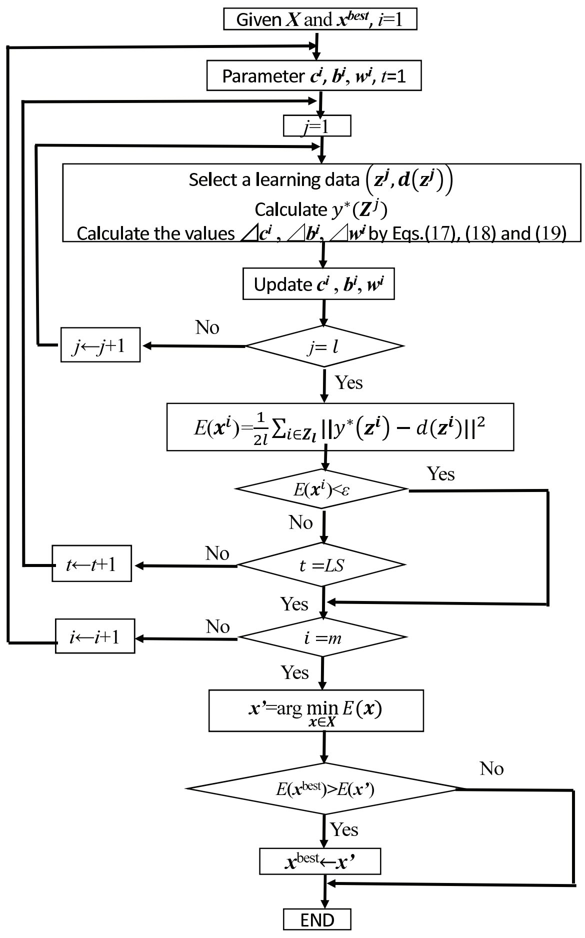

Given X and x^best, i=1
Parameter c^i, b^i, w^i, t=1
j=1
Select a learning data (z^j, d(z^j))
Calculate y*(Z^j)
Calculate the values ⊿c^i, ⊿b^i, ⊿w^i by Eqs.(17), (18) and (19)
Update c^i, b^i, w^i
j←j+1
No
j= l
Yes
E(x^i)=1/2l Σ_{i∈Z_l} ||y*(z^i) − d(z^i)||^2
E(x^i)<ε
Yes
No
t←t+1
No
t =LS
Yes
i←i+1
No
i =m
Yes
x'=arg min_{x∈X} E(x)
E(x^best)>E(x')
No
Yes
x^best←x'
END

Figure 9. The flowchart of FSI(LS, ε)

3.4 Hybrid Random Search Algorithm with BP

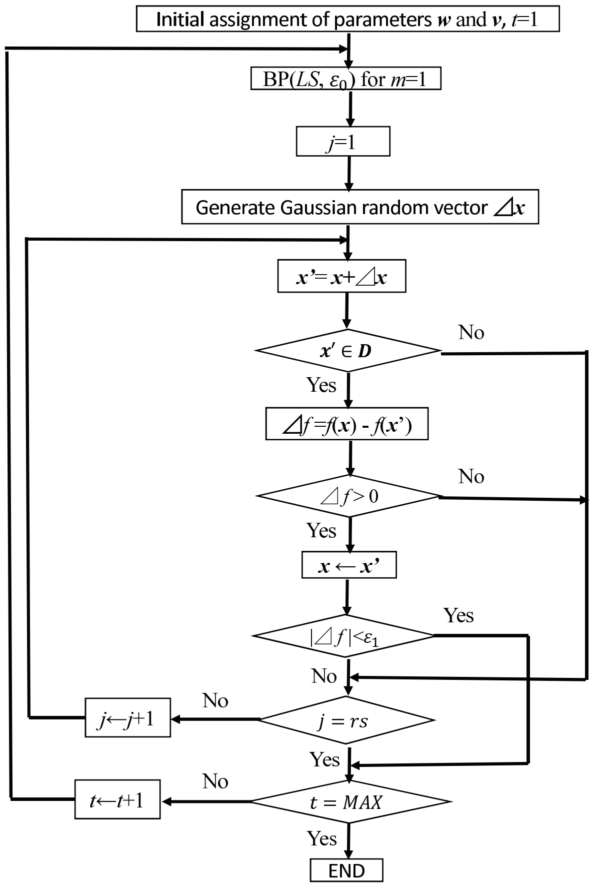


Figure 10. The flowchart of HRS-BP(MAX, LS, rs, ε_0, ε_1)

In order to compare the proposed hybrid EM method with the conventional hybrid one, let us consider the hybrid random search algorithm with BP [4, 8]. The method is well known as one of hybrid methods and is composed of random search algorithm and BP method, which are used for global and local searches, respectively. The method will be denoted as HRS-BP(MAX, LS, RS, ε_0, ε_1), where MAX, LS and RS are the maximum learning times for global, local and random searches, respectively, and ε_0 and ε_1 are the threshold for random search and BP method, respectively. Fig. 10 shows the flowchart of HRS-BP(MAX, LS, rs, ε_0, ε_1), where D is the constrained domain of parameters.

3.5 Hybrid PSO method with BP

```
Algorithm PSO($T_{max}$)
Initialize():
$\boldsymbol{x}^i \leftarrow U(\boldsymbol{L}, \boldsymbol{U})$ for $i \in Z_m$
$\boldsymbol{p} \leftarrow \boldsymbol{x}^i$ for $i \in Z_m$
$\boldsymbol{x}^{best} \leftarrow \arg\min_i \{f(\boldsymbol{x}^i)\}$
$\boldsymbol{v}^i \leftarrow$ U($-|\boldsymbol{U}-\boldsymbol{L}|$, $|\boldsymbol{U}-\boldsymbol{L}|$)
$w$, $c_1$, $c_2$;parameters
1: $counter \leftarrow 1$
2: while $counter < T_{max}$
3:   for $i$=1 to $m$
4:     $r_1$, $r_2 \leftarrow$ U(0, 1)
5:     for $k$=1 to $n$
6:       $v_k^i \leftarrow w v_k^i + c_1 r_1 (p_k^i - x_k^i) + c_2 r_2 (x_k^{best} - x_k^i)$
7:     end for
8:     $\boldsymbol{x}^i \leftarrow \boldsymbol{x} + \boldsymbol{v}^i$
9:     if $f(\boldsymbol{x}^i) < f(\boldsymbol{p}^i)$
10:      $\boldsymbol{p}^i \leftarrow \boldsymbol{x}^i$
11:    end if
12:  end for
13: end while
14: $\boldsymbol{x}^{best} \leftarrow \arg\min_i \{f(\boldsymbol{p}^i)\}$
```

Figure 11. PSO algorithm

```
Algorithm HPSO-BP($MAX$, $LS$, $T_{max}$, $\delta$)
1: Initialize
2: $Iteration \leftarrow 1$
3: while $iteration < MAX$
4:   BP($LS$, $\delta$)
5:   PSO($T_{max}$)
6:   $iteration \leftarrow iteration$ + 1
7: end while
```

Figure 12. HPSO-BP algorithm

In order to compare the proposed hybrid EM method with the conventional hybrid one, let us introduce hybrid PSO method with BP(HPSO-BP) as the same method as section 3.1. PSO is well known as the popular one of metaheuristic methods [9, 18]. Figs. 11 and 12 show the basic PSO algorithm and the outline of HPSO-BP with the parameters MAX, LS, T_{max} and δ.

4 Numerical simulations

4.1 The learning speed of hybrid methods for EX-OR problem

In order to compare the speed of learning for BP, EM and HEM-BP methods, a simulation of learning for EX-OR problem with two variables is performed as one example. The EX-OR problem is the logical function defined by $y = x_1 \oplus x_2$, where $x_1, x_2, y \in \{0, 1\}$ and $\oplus$ means the Exclusive OR operation [12]. Fig. 13 shows the average of ten trials for learning of three methods. The result shows that HEM-BP algorithm is superior in learning speed to the other ones, where MSE means the mean square error.

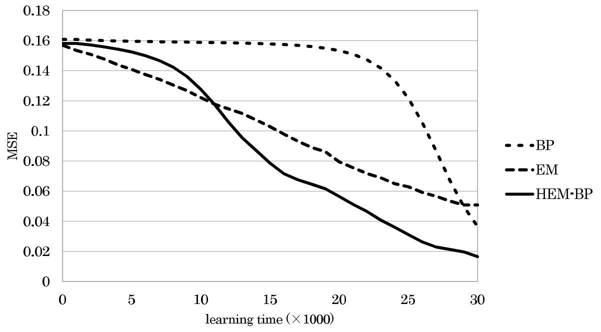


Figure 13. The learning speed of hybrid EM algorithm for EX-OR problem

4.2 The approximation capability of HEM-BP

In order to compare the approximation capability of HEM-BP with the conventional methods, two simulation are performed.

4.2.1 Function approximation

This simulation uses four systems specified by the following functions with $[0, 1]^4$(Eqs.(20) and (21)) and $[-1, 1]^4$(Eqs.(22) and (23)). The numbers of data for learning and test are 225 and 6400, respectively.

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{37.21} \times \frac{(4\sin(\pi x_3)+2\cos(\pi x_4)+6)}{12} \quad (20)$$

$$y = \frac{(\sin(2\pi x_1)\cos(\pi x_2)\sin(\pi x_3)x_4+1.0)}{2.0} \quad (21)$$

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{74.42} + \frac{(3e^{3x_3}+2e^{-4x_4})^{-0.5}-0.077}{4.68} \quad (22)$$

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{74.42} + \frac{(4\sin(\pi x_3)+2\cos(\pi x_4)+6)}{446.52} \quad (23)$$

The following algorithms are compared and three-layered neural network with $p = 12$ is used, where $\alpha_1 = \alpha_2 = 0.1$ for all and $w = 1.0$, $c_1 = c_2 = 0.1$ for HPSO-BP.

1. BP(10^5, 10^{-5})
2. HRS-BP(20, 5000, 10^5, 10^{-5}, 10^{-5})
3. HPSO-BP(20, 5000, 10^6, 10^{-5})
4. HEM-BP(20, 5000, 10^{-5})

Table 1 shows the result for four algorithms (1), (2), (3) and (4), where the upper and lower values in each box show MSE($\times 10^{-4}$) for learning and test data, respectively.

Table 1. Result of function approximation for HEM-BP

	Eq.(20)	Eq.(21)	Eq.(22)	Eq.(23)
(1)	0.68	21.29	1.46	1.28
	1.26	36.32	3.47	1.75
(2)	0.60	14.31	1.09	0.80
	1.09	25.06	3.20	1.35
(3)	0.50	10.38	1.01	0.96
	1.17	20.54	2.74	1.68
(4)	0.32	5.40	0.62	0.62
	0.81	16.33	2.62	1.30

HEM-BP is superior in accuracy to BP and is the same capability as HPSO-BP.

4.2.2 Classification problem

Iris, Wine, Sonar and BCW data from USI database shown in Table 2 are used for numerical simulation [14]. In this simulation, 5-fold cross validation is used. Then, the following algorithms are compared and three-layered neural network with $p = 20$ is used, where $\alpha_1 = \alpha_2 = 0.1$.

1. BP (50000, 10^{-5})
2. EM(500, 5)
3. HRS-BP(500, 1000, 10000, 10^{-5}, 10^{-5})
4. HEM-BP(500, 1000, 10^{-5})

Table 2. Data for classification problem

	The number of input	The number of clusters	The number of data
iris	4	3	150
wine	13	3	178
sonar	60	2	208
BCW	9	2	683

Table 3 shows the result of classification for four algorithms (1), (2), (3) and (4), where the values in each box mean the rate of misclassification (%) of MSE in the upper and one of minimum and maximum errors in the lower.

Table 3. Result of classification for HEM-BP

	iris	wine	sonar	BCW
(1)	4.0	13.7	18.5	4.3
	[0.0, 10.0]	[11.4, 20.0]	[14.6, 24.4]	[2.2, 8.1]
(2)	4.0	16.6	19.5	4.3
	[0.0, 6.7]	[8.6, 25.7]	[12.2, 29.3]	[2.2, 6.6]
(3)	4.7	9.7	19.5	4.3
	[0.0, 13.3]	[5.7, 14.3]	[17.1, 24.4]	[2.9, 5.1]
(4)	3.3	4.0	15.1	4.0
	[0.0, 6.7]	[2.2, 5.1]	[9.1, 24.4]	[2.2, 5.1]

Table 3 shows that HEM-BP is most effective in four algorithms.

4.3 HEM-KM for clustering

In order to compare the capability of clustering for four algorithms, data of Table 2 are also used. Then, the following algorithms are used.

1. KM(50000, 10^{-4})
2. EM(500, 1000)
3. HRS-KM(500, 1000, 10000, 10^{-4}, 10^{-4})
4. HEM-KM(500, 1000, 10^{-4})

Table 4 shows the result of classification for the above four algorithms (1), (2), (3) and (4), where each value in each box is the same meaning as one in 4.2. As shown in Table 4, HEM-KM is superior in accuracy to other methods.

Table 4. Result of clustering for HEM-KM

	iris	wine	sonar	BCW
(1)	11.3	19.7	45.6	3.9
	[4.0, 33.3]	[5.6, 33.9]	[43.8, 46.2]	[3.8, 4.0]
(2)	4.5	39.7	46.6	4.4
	[4.0, 6.7]	[36.5, 40.0]	[46.6, 46.6]	[4.0, 5.0]
(3)	4.0	23.1	45.7	3.9
	[4.0, 4.0]	[5.6, 41.6]	[45.7, 45.7]	[3.8, 4.0]
(4)	4.0	6.8	45.0	4.0
	[4.0, 4.0]	[6.2, 7.3]	[44.7, 45.7]	[3.9, 4.0]

4.4 The approximation capability of HEM-FIS

In order to compare the approximation capability of HEM-FIS, two simulations are performed. Further, the following cases for learning are performed.
Case 1: The parameters for learning are $\boldsymbol{c}$, $\boldsymbol{b}$ and $\boldsymbol{w}$.
Case 2: the parameter for learning is $\boldsymbol{w}$.

4.4.1 Function approximation

This simulation uses four systems specified by the functions Eqs.(20), (21), (22) and (23), where $c = 81$ in Case 1 and $c = 256$ in Case 2 and $K_c = K_b = 0.01$ and $K_w = 0.1$ in all cases.

The following algorithms are compared:

1. FIS(20000, 10^{-5})
2. HRS-FIS(20, 1000, 10000, 10^{-5}, 10^{-5})
3. HEM-FIS(20, 1000, 10^{-5})

Tables 5 and 6 show the results for Case 1 and Case 2, respectively. The number of data for learning is 512, and for test is 6400. The learning rate are $K_c = 0.01$, $K_w = 0.01$ and $K_w = 0.1$, respectively. The results show that HEM-FIS is superior to other algorithms in all cases.

Table 5. Result of function approximation for FIS in Case 1.

	Eq.(20)	Eq.(21)	Eq.(22)	Eq.(23)
(1)	0.10	0.10	0.10	0.10
	1.92	3.29	1.62	3.29
(2)	0.08	0.09	0.10	0.09
	1.83	2.56	2.05	2.56
(3)	0.09	0.09	0.09	0.09
	1.92	2.79	1.56	2.79

Table 6. Result of function approximation for FIS in Case 2.

	Eq.(20)	Eq.(21)	Eq.(22)	Eq.(23)
(1)	0.41	0.10	1.83	1.83
	7.74	14.47	23.89	21.16
(2)	1.26	1.87	2.88	2.99
	21.76	23.96	31.63	33.85
(3)	0.38	1.11	2.15	1.95
	3.27	7.69	14.32	13.81

4.4.2 Classification problem

The following classification problem is performed as numerical simulation: In the classification problems, points on $[0,1]^3$ are classified into two classes: class 0 and class 1. The class boundaries are given as spheres centered at (0.5,0.5,0.5). For Sphere, the inside of sphere is associated with class 1 and the outside with class 0. For Double-Sphere, the area between Spheres 1 and 2 is associated with class 1 and the other area with class 0. For triple-Sphere, the inside of Sphere 1 and the area between Sphere 2 and Sphere 3 is associated with class1 and the other area with class 0. Fig. 14 shows the case of Triple-Sphere.

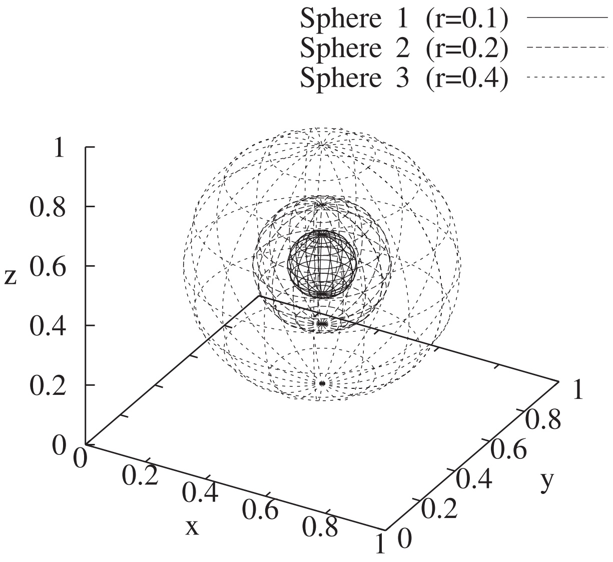


Figure 14. Triple-Sphere for two-category classification problem

Then, three algorithms are performed:

1. FIS(20000, 10^{-4})
2. HRS-FIS(20,1000,10000,10^{-4},10^{-4})
3. HEM-FIS(20, 1000, 10^{-4})

Tables 7 and 8 show the results for $c = 27$ in Case 1 and $c = 125$ in Case 2, respectively. The numbers of data for learning and test are 512 and 6400, respectively. Each value in each box is same as Table 3. The results show that HEM-FIS is superior to other algorithms.

Table 7. Result of classification problem for FIS in Case 1.

	Sphere	Double-Sphere	Triple-Sphere
(1)	2.7 [1.5, 3.8]	7.5 [6.2, 8.6]	8.1 [6.4, 9.4]
(2)	2.6 [2.0, 3.6]	6.6 [5.4, 9.7]	7.8 [6.0, 10.5]
(3)	2.4 [1.6, 2.9]	6.9 [6.1, 7.6]	7.7 [6.4, 9.2]

Table 8. Result of classification problem for FIS in Case 2.

	Sphere	Double-Sphere	Triple-Sphere
(1)	3.3 [2.7, 3.9]	5.1 [4.0, 6.1]	5.6 [4.8, 6.7]
(2)	3.1 [2.4, 3.6]	4.6 [3.3, 5.7]	5.8 [4.5, 6.9]
(3)	3.1 [2.6, 3.8]	4.6 [3.9, 5.5]	5.1 [4.4, 6.5]

5 Conclusion

In this paper, we investigated the performance of hybrid EM methods combining EM method with the steepest descent methods such as *k*-means, BP and FIS. It is shown that they are superior in learning speed and accuracy to the conventional EM, BP, *k*-means, and learning of FIS methods. Further, they showed better performance than the hybrid random search methods with BP, *k*-means and learning of FIS methods. Though we used *k*-means and BP methods as local search techniques, the other methods based on the steepest descent methods can be also applied.

As the future works, we will consider to propose hybrid EM methods with the other steepest descent methods and to prove the convergence of the proposed hybrid EM methods.

References

[1] I. Boussaid, J. Lepagnot, P. Siarry: A Survey on Optimization Metaheuristics, Information & Sciences, 237, pp.82-117, (2013)

[2] J. Matyas: Random Optimization, Automation & Remote Contr., 26, pp.246-253 (1965)

[3] J. Kennedy, R. Eberhart: Particle Swarm Optimization, IEEE Inf. Conf. on Neural Netwarks, 4, 1942-1948, (1995)

[4] N. Baba: A new Approach for Finding the Global Minimum of Error Function of Neural Networks, Neural Networks, 2, pp. 367-373, (1989)

[5] S. I. Birbil, S. C. Fang: An Electromagnetism-like Mechanism for Global Optimization, Journal of Global Optimization, 25, pp.263-282, (2003)

[6] J. Jiang, H. Shang, K. Liu, Q. Su, L. Zhang: A Clustering Method Using Electromagnetism-like

Mechanism Algorithm, Journal of Computational Information Systems, 9, 10, pp.3985-3191, (2013)

[7] M. Clerc, J. Kennedy: The Particle Swarm-Explosion, Stability, and Convergence in a Multi-dimensional Complex Space, IEEE Trans. on Evolutionary Computation, 6, 1, (2002)

[8] N. Baba: A Hybrid Algorithm for Finding the Global Minimum of Error Function of Neural Networks and its applications, Neural Networks, 7, 8, pp.1253-1265, (1994)

[9] H. Yuan, J. Ahi, J. Liu: Application of Particle Swarm Optimization Algorithms based Fuzzy BP Neural Network for Target Damage Accesment, Scientific Research and Essays, 6, 15, pp.3109-3121, (2011)

[10] J. L. Lin, C.H. Wa, H.Y. Chung: Performance Comparison of Electromagnetism-like Algorithms for Global Optimization, Applied Mathematics, 3, pp.1265-1275, (2012)

[11] C.H.Lee, C.T.Li, F.Y.Chang: A Species-based improved Electromagnetism-like Mechanism Algorithm for TSK-type integral-valued Neural Fuzzy System Optimization, Fuzzy Set and Systems, 171, pp. 22-43, (2011)

[12] M. M. Gupta, L. Jin, N. Honma: Static and Dynamic Neural Networks, IEEE Pres, Wiley-Interscience, (2003)

[13] T. M. Martinetz, S. G. Berkovich, K. J. Schulten: Neural Gas Network for Vector Quantization and its Application to Time-series Prediction, IEEE Trans. Neural Network, 4, 4, pp.558-569, (1993)

[14] UCI Repository of Machine Learning Databases and Domain Theories, ftp://ftp.ics.uci.edu/pub/machinelearning-Databases

[15] C.H. Lee, F.Y. Chang, C.T. Lee: A hybrid of electromagnetism-like mechanism and back-propagation algorithms for recurrent neural fuzzy systems design, Journal of Systems Science, 43, 2, pp. 231-247, (2012)

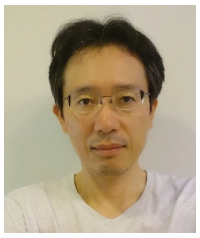
[16] H.H Chang, T.Y. Huang: Mixture Experiment Design Using Artificial Neural Networks and Electromagnetism-like Mechanism Algorithm, in Proc. Second International Conference on Innovative Computing, Information and Control, pp. 397-397, (2007)

[17] X.J. Wang, L. Gao, C.Y. Zhang: Electromagnetism-Like Mechanism Based Algorithm for Neural Network Training, LNAI, 5227, pp. 40-45, (2008)

[18] C.H. Lee, F.K. Chang, Y.C. Lee: Nonlinear systems design by a novel fuzzy neural system via hybridization of electromagnetism-like mechanism and particle swarm optimization algorithms, Information Sciences, 186, 1, pp. 59-72, (2012)

[19] S. Fukumoto, H. Miyajima: Learning Algorithms with Regularization Criteria for Fuzzy Reasoning Model, Journal of Innovative Computing Information and Control, 1,1, pp. 249-163, (2006)

Hirofumimi Miyajima received the B.E. degree in Electronics and Information Engineering from Hokkaido University, Japan, in 2010, and the M.E. degree in Information Science and Technology from Osaka University, Japan, in 2012. He is currently working toward his Dr. Eng. Degree at Kagoshima University. His current research interests include fuzzy modeling, neural networks and learning algorithms for them.



Noritaka Shigei received the B.E., M.E., D.E. degrees from Kagoshima University, Japan, in 1992, 1994, and 1997, respectively. He is currently an Associate Professor in Graduate School of Science and Engineering at Kagoshima University. His current research interests include neural network, wireless sensor network, digital communication system, digital circuit design, and parallel computing system.