https://doi.org/10.20965/jaciii.2008.p0085

Paper:

A Global Optimization Method RasID-GA for Neural Network Training

Dongkyu Sohn, Shingo Mabu, Kaoru Shimada, Kotaro Hirasawa, and Jinglu Hu

Graduate School of Information, Production and Systems, Waseda University,
2-7 Hibikino, Wakamatus-ku, Kitakyushu-shi, Fukuoka, 808-0135, JAPAN
E-mail:sohn@suou.waseda.jp, mabu@asagi.waseda.jp, k.shimada@aoni.waseda.jp, {hirasawa, jinglu}@waseda.jp
[Received May 1, 2007; accepted October 2, 2007]

This paper applies an Adaptive Random search with Intensification and Diversification combined with Genetic Algorithm (RasID-GA) to neural network training. In the previous work, we proposed RasID-GA which combines the best properties of RasID and Genetic Algorithm for optimization. Neural networks are widely used in pattern recognition, system modeling, prediction and other areas. Although most neural network training uses gradient based schemes such as well-known back-propagation (BP), but sometimes BP is easily dropped into local minima. In this paper, we train newly developed multi-branch neural networks using RasID-GA with constraint coefficient *C* by which the feasible solution space is controlled. In addition, we use Mackey-Glass time prediction to test a generalization ability of the proposed method.

Keywords: optimization, RasID, GA, switching

1. Introduction

Neural networks are widely used in pattern recognition, system modeling, prediction and other areas. In general, the back-propagation algorithm (BP) based on gradient schemes is used for the neural network training. But, sometimes the BP falls into a local minima easily causing poor performances. In addition, if the objective function is not differentiable, the BP algorithm cannot be applied to neural network training.

To solve the local minima problems, the random search scheme, genetic algorithm, back-propagation combined with random search scheme [1, 2] etc. are proposed by many researchers. But the random search scheme and genetic algorithm are not efficient for the neural network training in terms of the convergence speed. While, the back-propagation combined with random search scheme needs the setting of the conditions for the transition between both algorithms.

This paper presents the novel training algorithm, RasID-GA [3, 4], for training multi-branch neural networks recently developed instead of well-known simple neural networks. The multi-branch neural network is a neural network, where there are a good number of branches between nodes in order to enhance its representation ability. In the previous work, we have introduced RasID-GA, which combines the best properties of RasID [5–8] and Genetic Algorithm [9] for optimization problems. The proposed algorithm can be viewed as a kind of Memetic Algorithms [10, 11]. The basic algorithm of RasID-GA coincides with the concept of Memetic Algorithms where GA is combined with other local search techniques. Generally, in Memetic algorithms, some steps of the local search are executed after one generation of GA search. Unlike Memetic algorithms, after some individuals are trapped in local minima using parallel RasIDs, then, a number of GA search is carried out until they escape from local minima in RasID-GA.

In this research, RasID-GA finds a set of weights that minimizes the criterion function E of multi-branch neural networks using adaptively-adjusted probability density function (PDF). It is updated using the information on the success or failure in the past searching. As a result, we expect to enhance the effectiveness and efficiency of the searching of multi-branch neural networks. In addition, we introduce the constraints for the weights of multi-branch neural networks using constraint coefficient C in order to enhance the generalization ability. This could be done easily using RasID-GA.

This paper is organized as follows. Section 2 describes the structure of RasID-GA and a sophisticated PDF. Section 3 presents multi-branch neural networks. In Section 4, we show the simulation results. Finally, a brief summary of this paper and conclusions are presented in Section 5.

2. Structure of RasID-GA

In the previous work, we proposed RasID-GA [3, 4] which combines the best properties of RasID and Genetic Algorithm for optimization problems. Genetic Algorithm (GA) can find an optimal solution using the mechanism of biological heredity and selection. Although GA is highly evaluated for searching near optimal solutions, it is generally recognized that it cannot search for an optimum solution precisely. While RasID is good at finding a precise local optimum solution, although its diversified searching

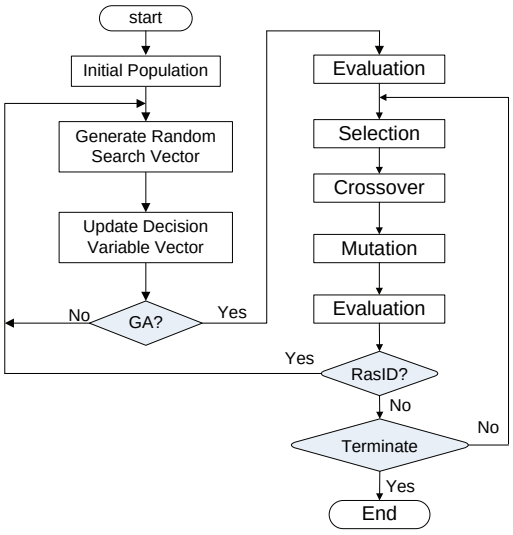


Fig. 1. Flow of RasID-GA searching.

is not so superior when the problem has a great number of local optima. That is why RasID-GA is proposed. RasID and GA are combined in order to improve the searching ability in terms of reducing the time spent on the searching and obtaining a solution as near as the global optimum. **Fig. 1** shows the structure of RasID-GA.

2.1. Probability Density Function

In conventional random search algorithms [12–14], Gaussian PDFs are used to generate random search vector θ. To make it more tunable, we introduce a sophisticated PDF $g(\theta_m)$ for generating the random search variable θ_m, defined by

$$g(\theta_m) = \begin{cases} (1-q_m)\beta_m e^{\beta_m \theta_m} & \text{If} \quad \theta_m \leq 0 \\ q_m \beta_m e^{-\beta_m \theta_m} & \text{If} \quad \theta_m > 0 \end{cases} \quad . \quad (1)$$

where, $q_m \in [0, 1]$ and β_m are two kinds of parameters used to perform intensification-diversification search. By integrating Eq. (1), we have the probability distribution function given by

$$G(\theta_m) = \begin{cases} (1-q_m)e^{\beta_m \theta_m} & \text{If} \quad \theta_m \leq 0 \\ 1 - q_m e^{-\beta_m \theta_m} & \text{If} \quad \theta_m > 0. \end{cases} \quad . \quad . \quad (2)$$

Therefore, it follows that a random search variable θ_m is generated by

$$\theta_m = \begin{cases} \frac{1}{\beta_m} \ln(\frac{z_m}{1-q_m}) & \text{If} \quad 0 < z_m \leq 1 - q_m \\ -\frac{1}{\beta_m} \ln(\frac{1-z_m}{q_m}) & \text{If} \quad 1 - q_m < z_m < 1.0, \end{cases} \quad (3)$$

where, z_m's are random values uniformly distributed between 0 and 1.

As shown in **Fig. 2**, the parameter β_m can be used to control the local search range (the variance of search variable θ_m). The larger the β_m is, the smaller the local search range will be. On the other hand, the parameter q_m can be used to control the search probability in positive or negative direction. The larger the q_m is, the higher the search probability in positive direction is. Typically, when $q_m = 0.5$, there is the same search probability in positive direction and in negative direction. In this way, we are able to realize an efficient searching in training by determining the parameters β_m and q_m based on the past success-failure information and local information.

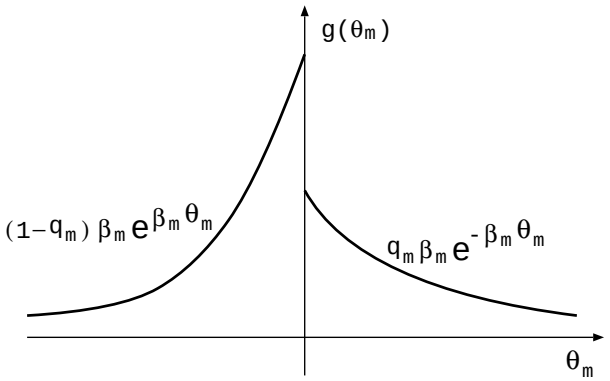


Fig. 2. Probability density function for generating random variable θ_m.

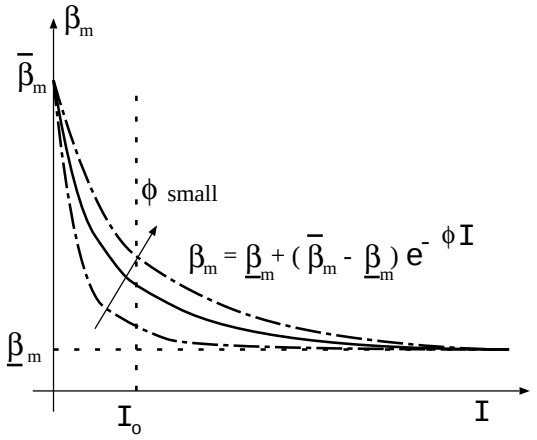


Fig. 3. Relation between β_m and $\bar{\beta}_m$, $\underline{\beta}_m$, ϕ, I.

As can be seen from Eq. (1), when β_m is small, the variance of random search variable is large. We will show that the intensified and diversified search can be realized by adjusting the parameter β_m effectively. For this purpose, we consider the following rules for adjusting β_m.

- Adjust β_m between a lower bound and a upper bound. Let $\underline{\beta}_m$ be the lower bound, and $\bar{\beta}_m$ be the upper bound.
- Adjust β_m based on two indexes, say ϕ and I. The index ϕ is designed to realize an intensified search and the index I for a diversified search.

Based on the above considerations, a typical rule used to determine β_m is introduced by

$$\beta_m = \underline{\beta}_m + (\overline{\beta}_m - \underline{\beta}_m)e^{-\phi I} \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad (4)$$

where $\overline{\beta}_m$, ϕ and I are adjustable indexes with initial values of $\bar{\beta}_{m0}$, ϕ_0 and I_0, respectively. If no useful prior knowledge is available for determining $\overline{\beta}_{m0}$, then $\overline{\beta}_{m0} = \overline{\beta}_0$ is used, where $\overline{\beta}_0$ is an appropriate initial value. **Fig.3** shows the relation between β_m and $\overline{\beta}_m$, $\underline{\beta}_m$, ϕ, I.

2.2. Flow of RasID-GA

Let $x(n)$ be decision variable vector

$$x = (x_1, x_2, ..., x_m, ..., x_M) \in W$$

at nth search, and $\theta(n)$ be the random search vector

$$\theta = (\theta_1, \theta_2, ..., \theta_m, ..., \theta_M) \in W$$

at nth search.

The details of RasID-GA are summarized as follows [4].

Step 1 *Generation of initial population.*
Initial population is randomly produced.

$$x^l(0) \in W,\ n = 0,\ l \in L \quad . \quad . \quad . \quad . \quad . \quad . \quad (5)$$

where,
n, l and L denote the searching step, the suffix of individuals and the set of suffixes of individuals.

Step 2 *RasID searching*
Generate a population of random search vector $\theta^l(n)$. RasID searching of RasID-GA follows GA searching, so the initial population of RasID is the final results of GA searching. The following procedure is carried out for each individual l.
If $(x^l(n) + \theta^l(n)) \notin W$, then let $x^l(n+1) = x^l(n)$ else

1. *Calculate $f(x^l(n) + \theta^l(n))$,*
   If $f(x^l(n) + \theta^l(n)) < f(x^l(n))$,
   then the current search is success and
   $x^l(n+1) = x^l(n) + \theta^l(n)$, *go to* Step *3,*
2. *Calculate $f(x^l(n) - \theta^l(n))$,*
   If $f(x^l(n) - \theta^l(n)) < f(x^l(n))$,
   then the current search is success and
   $x^l(n+1) = x^l(n) - \theta^l(n)$, *go to* Step *3,*
3. *The current search is failure and*

$$x^l(n+1) = \begin{cases} x^l(n) & \text{If } k_{er}^{+l} > k_{er} \text{ and } k_{er}^{-l} > k_{er}, \\ x^l(n) + \theta^l(n) & \text{If } k_{er}^{+l} < k_{er}^{-l}, \\ x^l(n) - \theta^l(n) & \text{If } k_{er}^{+l} \geq k_{er}^{-l}, \end{cases}$$

where,

$$k_{er}^{+l} = f(x^l(n) + \theta^l(n))/f(x^l(n)),$$
$$k_{er}^{-l} = f(x^l(n) - \theta^l(n))/f(x^l(n)).$$

Step 3 $n = n+1$, *then transition to* Step *4 (GA searching), if the criterion of transition is satisfied; Otherwise, go to* Step *2.*

Step 4 *GA searching*
GA searching of RasID-GA follows RasID searching, so initial population of GA is the final results of RasID searching.

1. *Selection*
   Tournament selection and elite preserving selection are used in RasID-GA. The tournament selection selects two individuals from the population and to leave a better individual to the next generation. Elite preserving selection transfers the best individual having the highest fitness value to the next generation.
2. *Crossover*
   Two individuals in the population are selected in order to execute the crossover with its probability. New two offspring is produced by crossover as follows.
   $x' = (1-a)x + ay$
   $y' = (1-a)y + ax,$ $a \in [0,1]$ *where*
   x, y : parent individual vector
   x', y' : offspring individual vector.
3. *Mutation*
   - *Select an individual in the population.*
   - *A new individual $\hat{x}$ is generated by changing the individual vector randomly with the probability of mutation.*
     $\hat{x} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_m, \cdots, \hat{x}_M)$,
     $\hat{x}_m = random(u, v)$,
     where
     $\hat{x}$: offspring individual vector,
     $random(u,v)$: generates a random number between u and v,
     u : lower boundary,
     v : upper boundary.

Step 5 $n = n+1$, *then transition to* Step *2 (RasID searching), if the criterion of transition is satisfied.*

Step 6 *If the ending condition is satisfied, the procedure stops; Otherwise, go to* Step *4.*

3. Multi-Branch Recurrent Neural Networks (MBRNNs)

In this section, we describe Multi-Branch Recurrent Neural Networks (MBRNNs) [15, 16]. The MBRNNs are the Recurrent Neural networks with Multi-Branch Structure. The basic concept of the multi-branch structure comes from Universal Learning Networks (ULNs) [17, 18]. ULNs provide a generalized framework to all kinds of structures of neural networks with supervised learning, and one of whose characteristics is a multi-branch structure. It connects nodes using some branches in order to enhance the representation ability of neural networks.

Figure 4 shows the multi-branch structure of MBRNNs. It has multi-branches between nodes, where delay time $D_{ij}(p)$ and weight $W_{ij}(p)$ are set on the pth branch between nodes. In addition, a Gaussian function is set on each branch, while each node has its sigmoidal function as usual. Therefore, the input of node

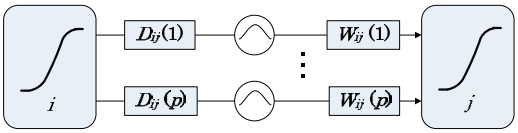


Fig. 4. Multi-branch structure between nodes

j can be described by Eq. (6).

$$\alpha_j(t) = \sum_{i\in JF(j)} \sum_{p\in B(i,j)} w_{ij}(p) g_{ij}(p, h_i(t - D_{ij}(p))) + \theta_j. \quad (6)$$

Where $B(i,j)$ denotes the set of suffixes of branches from node i to node j, $JF(j)$ is the set of suffixes of nodes connected to node j, θ_j is the threshold of node j and $g_{ij}(p, h_i(t - D_{ij}(p)))$ is the modified output value of node i used for pth branch from node i to node j. The function $g_{ij}(p, h_i(t - D_{ij}(p)))$ is to be a certain nonlinear function in order to obtain nonlinear effects from the multi-branch structure and it is defined as follows.

$$g_{ij}(p, h_i(t - D_{ij}(p))) = exp\left(-\frac{(h_i(t - D_{ij}(p)) - c_{ij}(p))^2}{\sigma_{ij}^2(p) + L}\right). \quad (7)$$

where $c_{ij}(p)$ and $\sigma_{ij}(p)$ are the center and breadth of the pth branch from node i to node j of the Gaussian function, respectively. L is a constant to set the lower bound of the breadth of the Gaussian function.

Learning of MBRNNs is realized by the following when the gradient method is used:

$$\lambda_m \leftarrow \lambda_m - \gamma \frac{\partial^+ E}{\partial \lambda_m}, \quad \lambda_m \in \{w_{ij}(p), D_{ij}(p), c_{ij}(p), \sigma_{ij}(p), \theta_j\}, \quad (8)$$

where γ is the learning coefficient assigned a small positive value and λ_m is the learning parameter. Here, the ordered derivative $\frac{\partial^+ E}{\partial \lambda_m}$ is calculated by backpropagation algorithm [17].

4. Simulation

This section describes simulations to demonstrate the effectiveness of the proposed method.

4.1. Test function

We used Mackey-Glass time series prediction to test the ability of the proposed method. The Mackey-Glass equation [19, 20] is a time delayed differential equation and the data is generated by

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1 + x^{10}(t-\tau)} - bx(t) \quad (9)$$

where, $a = 0.2$, $b = 0.1$ and $\tau = 17$ are used in this simulation. The prediction of the future value of x is based on their past values. In this simulation, network inputs are $x(t)$ and we predict the future value of a data at $(t+6)$. The graph of Mackey-Glass function over $t \in [0, 1000]$ is given in **Fig. 5**. Where, the network is trained with the former half 500 point training data, and the latter half 500 point data are used for prediction [21].

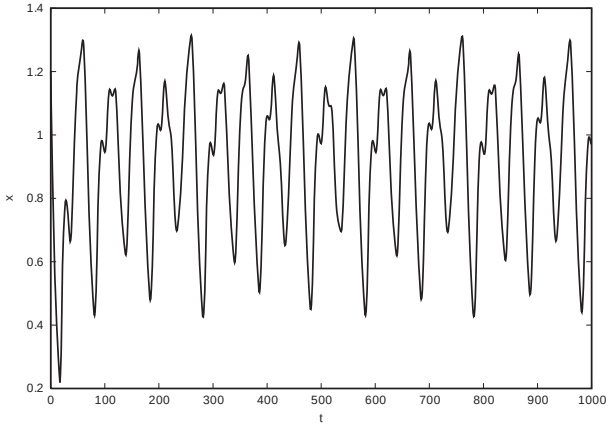


Fig. 5. Mackey Glass time series

4.2. Optimization by RasID-GA

In RasID-GA simulations, the number of iterations is set at 50,000. 30 independent runs are simulated changing the initial decision variables, and the average and standard deviation of the optimal values over 30 runs are calculated. The numbers of individuals are 20. The mutation rate and crossover rate are 0.01 and 0.5 in real-coded GA, respectively. In this simulation, RasID-GA is to find a set of weights with constraints and other parameters like $D_{ij}(p)$, $C_{ij}(p)$, $\sigma_{ij}(p)$ and $\theta_j(p)$ that minimizes the criterion function E. The criterion function E and constraints are defined as follows.

$$\text{minimize} \quad E = \sum_d (h_o - h'_o)^2, \quad \text{subject to} \quad \left|\sum_{w_{ij}(p)} w_{ij}(p)\right| \le CN \quad (10)$$

where h_o is the output value of the network in case that the input vector is d, h'_o is its desired output value, $w_{ij}(p)$ is network weight, C is the constraint coefficient and N is the number of weights. The network weights are constrained by coefficient C.

4.3. Optimization by Backpropagation

The criterion function E is defined as follows.

$$E = \sum_d (h_o - h'_o)^2 + \eta \left(\sum_{w_{ij}(p)} \frac{w_{ij}^2(p)}{\sigma_{ij}^2(p)}\right) \quad (11)$$

where η is the regularization coefficient in weight decay. Here, η was determined adaptively so that the penalty

Table 1. Simulation results of RasID-GA on Mackey Glass time series with various C (MSE, 3 nodes, 114 parameters).

C	0.50	1.00	10.00	15.00	20.00	25.00
Best(training)	5.111×10^{-05}	3.633×10^{-05}	1.185×10^{-05}	4.036×10^{-05}	4.283×10^{-05}	2.798×10^{-05}
Best(testing)	4.189×10^{-05}	3.620×10^{-05}	9.488×10^{-06}	4.279×10^{-05}	4.566×10^{-05}	2.571×10^{-05}
Worst(training)	3.244×10^{-04}	2.345×10^{-04}	1.363×10^{-04}	1.551×10^{-04}	2.950×10^{-04}	2.273×10^{-04}
Worst(testing)	2.694×10^{-04}	2.289×10^{-04}	1.292×10^{-04}	1.673×10^{-04}	2.126×10^{-04}	2.348×10^{-04}
Ave(training)	1.128×10^{-04}	1.090×10^{-04}	5.705×10^{-05}	7.291×10^{-05}	9.388×10^{-05}	9.829×10^{-05}
Ave(testing)	1.021×10^{-04}	1.013×10^{-04}	5.399×10^{-05}	7.263×10^{-05}	9.037×10^{-05}	1.041×10^{-04}
Stdv(training)	9.309×10^{-05}	8.060×10^{-05}	2.342×10^{-05}	3.077×10^{-05}	7.591×10^{-05}	6.562×10^{-05}
Stdv(testing)	7.924×10^{-05}	7.446×10^{-05}	2.190×10^{-05}	3.440×10^{-05}	5.899×10^{-05}	7.162×10^{-05}

Table 2. Simulation results of RasID-GA on Mackey Glass time series with various C (MSE, 5 nodes, 310 parameters).

C	0.10	0.30	0.50	0.70	1.00	10.00
Best(training)	5.800×10^{-04}	2.975×10^{-05}	1.895×10^{-05}	3.411×10^{-05}	2.358×10^{-05}	2.698×10^{-05}
Best(testing)	1.013×10^{-03}	3.096×10^{-05}	1.737×10^{-05}	3.603×10^{-05}	2.446×10^{-05}	2.817×10^{-05}
Worst(training)	5.802×10^{-03}	2.482×10^{-04}	5.620×10^{-05}	1.854×10^{-04}	1.244×10^{-04}	1.536×10^{-04}
Worst(testing)	5.318×10^{-03}	2.369×10^{-04}	5.644×10^{-05}	1.930×10^{-04}	1.334×10^{-04}	1.587×10^{-04}
Ave(training)	2.717×10^{-03}	1.367×10^{-04}	3.908×10^{-05}	8.067×10^{-05}	7.755×10^{-05}	8.129×10^{-05}
Ave(testing)	2.391×10^{-03}	1.447×10^{-04}	3.982×10^{-05}	8.346×10^{-05}	7.968×10^{-05}	8.290×10^{-05}
Stdv(training)	1.689×10^{-03}	7.823×10^{-05}	9.741×10^{-06}	5.114×10^{-05}	3.262×10^{-05}	4.129×10^{-05}
Stdv(testing)	1.407×10^{-03}	8.622×10^{-05}	1.025×10^{-05}	5.328×10^{-05}	3.940×10^{-05}	4.692×10^{-05}

Table 3. Simulation results of RasID-GA on Mackey Glass time series with various C (MSE, 10 nodes, 1220 parameters).

C	0.05	0.10	0.30	0.50	1.00	10.00
Best(training)	2.922×10^{-05}	9.216×10^{-05}	2.478×10^{-05}	3.268×10^{-05}	5.718×10^{-05}	8.928×10^{-05}
Best(testing)	3.151×10^{-05}	9.649×10^{-05}	2.636×10^{-05}	4.240×10^{-05}	5.407×10^{-05}	8.132×10^{-05}
Worst(training)	3.437×10^{-04}	3.777×10^{-04}	7.753×10^{-05}	1.568×10^{-04}	1.756×10^{-04}	3.809×10^{-04}
Worst(testing)	3.308×10^{-04}	3.717×10^{-04}	7.925×10^{-05}	1.948×10^{-04}	2.040×10^{-04}	3.804×10^{-04}
Ave(training)	1.405×10^{-04}	2.339×10^{-04}	5.204×10^{-05}	8.466×10^{-05}	1.342×10^{-04}	1.530×10^{-04}
Ave(testing)	1.341×10^{-04}	2.382×10^{-04}	5.262×10^{-05}	1.219×10^{-04}	1.341×10^{-04}	1.484×10^{-04}
Stdv(training)	1.162×10^{-05}	9.414×10^{-05}	2.568×10^{-05}	4.421×10^{-05}	3.266×10^{-05}	1.204×10^{-04}
Stdv(testing)	1.079×10^{-05}	9.465×10^{-05}	2.669×10^{-05}	9.055×10^{-05}	4.326×10^{-05}	1.247×10^{-04}

Table 4. Comparison between RasID-GA and BP algorithms (MSE, 3 nodes/114 parameters, 5 nodes/310 parameters, 10 nodes/1220 parameters).

	3 nodes/114 parameters		5 nodes/310 parameters		10 nodes/1220 parameters	
	RasID-GA	BP	RasID-GA	BP	RasID-GA	BP
Best(training)	1.185×10^{-05}	6.573×10^{-06}	1.895×10^{-05}	1.015×10^{-06}	2.478×10^{-05}	2.520×10^{-06}
Best(testing)	9.488×10^{-06}	5.472×10^{-06}	1.737×10^{-05}	6.940×10^{-07}	2.636×10^{-05}	4.380×10^{-06}
Worst(training)	1.363×10^{-04}	8.748×10^{-04}	5.620×10^{-05}	9.865×10^{-04}	7.753×10^{-05}	7.220×10^{-06}
Worst(testing)	1.292×10^{-04}	2.470×10^{-02}	5.644×10^{-05}	5.842×10^{-04}	7.925×10^{-05}	3.180×10^{-06}
Ave(training)	5.705×10^{-05}	1.756×10^{-04}	3.908×10^{-05}	9.887×10^{-05}	5.204×10^{-05}	1.010×10^{-06}
Ave(testing)	5.399×10^{-05}	1.783×10^{-03}	3.982×10^{-05}	4.916×10^{-05}	5.262×10^{-05}	5.630×10^{-06}
Stdv(training)	2.342×10^{-05}	2.588×10^{-04}	9.741×10^{-06}	2.370×10^{-04}	2.568×10^{-05}	1.560×10^{-06}
Stdv(testing)	2.190×10^{-05}	6.081×10^{-03}	1.025×10^{-05}	1.182×10^{-04}	2.669×10^{-05}	7.130×10^{-06}

term in Eq. (11) becomes 0.2% of the error term and the generalization ability is obtained. The learning coefficient γ initialized at 0.001 changes adaptively: When the value of E decreases, $\gamma \leftarrow 1.05\gamma$, while $\gamma \leftarrow 0.95\ \gamma$ when it increases. In addition, if $E(s)/E(s-1) > 1.05$, then the parameter update is not executed, where $E(s)$ is the criterion value at step s. Sigmoidal function $\frac{1-e^{-\alpha_j(t)}}{1+e^{-\alpha_j(t)}}$ is used as the node function and $|p| = 3$ is used in the simulations.

4.4. Effectiveness of Constraint Coefficient C

In the learning of MBRNNs with RasID-GA, the network weights are constrained by coefficient C. To obtain better generalization results in the learning of MBRNNs, adjusting C is essential. To evaluate the effects of C on the results, we did experiments changing the constraint coefficient C in RasID-GA. **Table 1** shows the simulation results on mean squared errors (MSE) by the MBRNNs (the best value, average value, worst value and standard deviation of the best solutions over 30 runs in training and testing), where 3 nodes and 114 parameters are used under $C = 0.50$, $C = 1.00$, $C = 10.00$, $C = 15.00$, $C = 20.00$ and $C = 25.00$. **Table 2** shows the simulation results of the MBRNNs with 5 nodes and 310 parameters under $C = 0.10$, $C = 0.30$, $C = 0.50$, $C = 0.70$, $C = 1.00$ and $C = 10.00$. **Table 3** shows the simulation results of the MBRNNs with 10 nodes and 1220 parameters under $C = 0.05$, $C = 0.10$, $C = 0.30$, $C = 0.50$, $C = 1.00$ and $C = 10.00$. It is found from **Tables 1**, **2**, and **3** that when

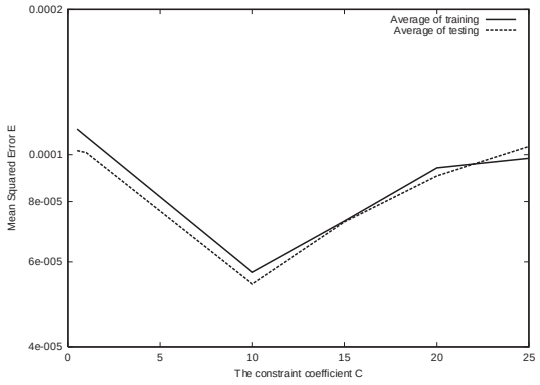


Fig. 6. Simulation result of RasID-GA on Mackey Glass time series with various C (3 nodes, 114 parameters).

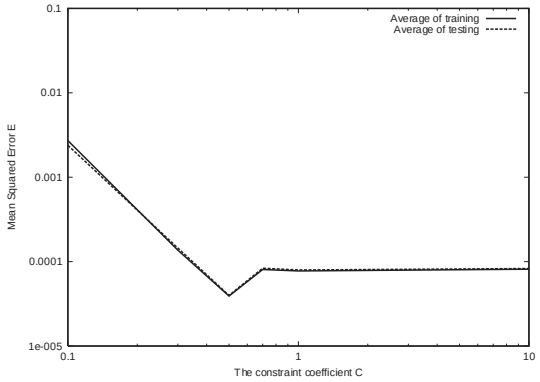


Fig. 7. Simulation result of RasID-GA on Mackey Glass time series with various C (5 nodes, 310 parameters).

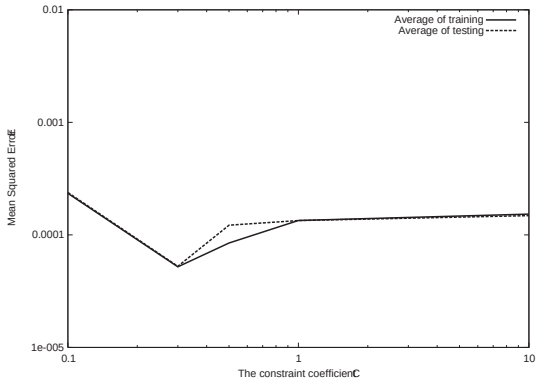
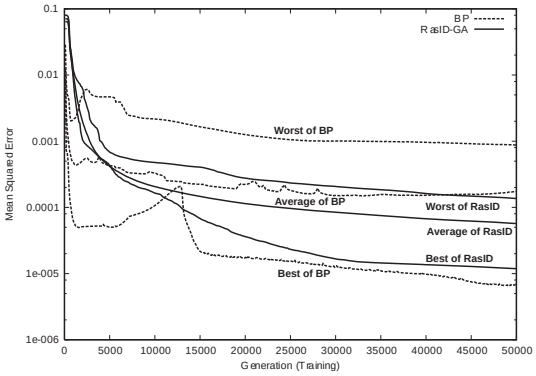
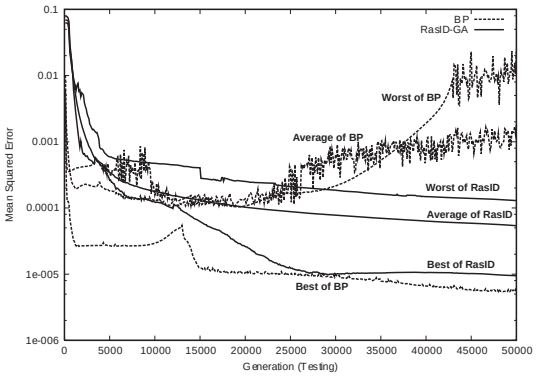


Fig. 8. Simulation result of RasID-GA on Mackey Glass time series with various C (10 nodes, 1220 parameters).

the constraint coefficient C is too small, the quality of the simulation results would be poor. In this case, finding the optimal solution is difficult because the feasible solution space is so small due to small constraint coefficient C. As a result, the generalization ability also does not show good results. On the other hand, when constraint coefficient C is too large, the feasible solution space becomes large, but it becomes also difficult to find the optimal solution because of the large feasible solution space itself. As a result, the representation and generalization ability depend on the magnitude of the feasible solution space which is controlled by constraint coefficient C. Therefore, we can find the suitable constraint coefficient C.



(a) Simulation results of training.



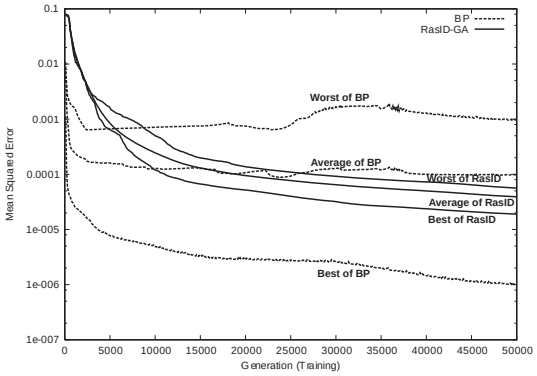
(b) Simulation results of testing.

Fig. 9. Comparison between RasID-GA and BP algorithms (MSE, 3 nodes, 114 parameters).

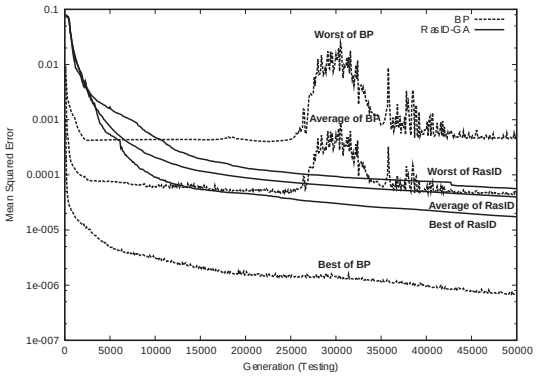
Figs. 6, **7**, and **8** show the average mean squared errors over 30 runs in training and testing, where the suitable constraints, i.e, $C = 10$, $C = 0.5$ and $C = 0.3$ of MBRNNs with 3nodes/114 parameters, 5nodes/310 parameters and 10nodes/1220 parameters are used, respectively.

4.5. Comparison Between RasID-GA and BP

In the simulations, RasID-GA are compared with BP algorithm. 30 independent runs are carried out changing the initial decision variables. MBRNNs have 3 nodes/114 parameters, 5 nodes/310 parameters and 10 nodes/1220 parameters. **Table 4** shows the simulation results of RasID-GA and BP algorithm on mean squared errors. **Table 4** shows the best value, average value, worst value and standard deviation of the best solutions ($C = 10$ for 3 nodes/114 parameters: $C = 0.5$ for 5 hidden nodes/310 parameters: $C = 0.3$ for 10 nodes/1220 parameters) over 30 independent runs in training and testing. The better result between two methods is underlined in the **Table 4**. From **Table 4**, it is found RasID-GA has better performances in terms of the average, worst and standard deviation in the networks with small size, however, BP obtains better performances in terms of the best, average worst and standard deviation in the networks with large size. **Figs. 9**, **10**, and **11** show the comparison of the learning curves between RasID-GA and BP algorithm. They

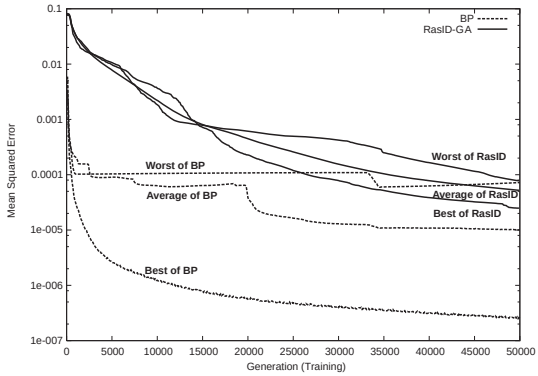


(a) Simulation results of training.

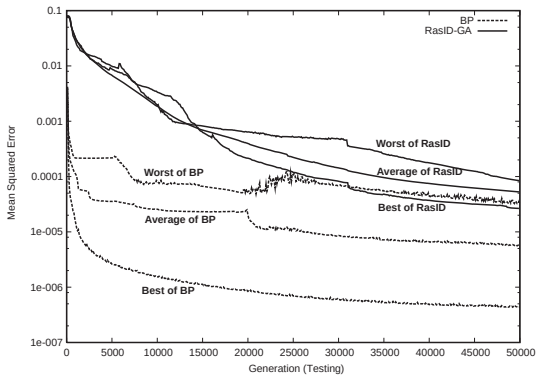


(b) Simulation results of testing.

Fig. 10. Comparison between RasID-GA and BP algorithms (MSE, 5 nodes, 310 parameters).



(a) Simulation results of training.



(b) Simulation results of testing.

Fig. 11. Comparison between RasID-GA and BP algorithms (MSE, 10 nodes, 1220 parameters).

show that BP has faster convergence and obtains better solutions than RasID-GA in terms of the best in training and testing, but RasID-GA performs well in terms of the average, worst and standard deviation for networks with 3 hidden nodes/114 parameters and 5 hidden nodes/310 parameters. In that sense, RasID-GA shows the stable generalization ability in the networks with small size, which means the worst case, that is essential for evaluating the generalization ability, of RasID-GA is far better than BP when the size of networks is small. In the networks with 10 nodes/ 1220 parameters, RasID-GA is not superior to GA algorithm. The reason for it is that RasID-GA is a kind of random search, in other words, RasID-GA should find a better solution than the current solution by searching 2^M points, although it could use a sophisticated probability density function for generating the search points. Therefore, the increase of the number of parameters makes it difficult to search an optimal solution efficiently and effectively in RasID-GA.

5. Conclusions

We have proposed a novel learning algorithm, RasID-GA, for multi-branch neural networks. Then we discussed that the network weights are constrained by coefficient C in order to have good generalization ability. In addition, the proposed method is compared with BP algorithm for Mackey Glass time series prediction. As a result, RasID-GA showed the stable generalization ability in not very large networks, which means the RasID-GA could be used for the learning of most of the neural networks.

References:

[1] N. Baba, "A new approach for finding the global minimum of error function of neural networks," Neural Networks, Vol.2, pp. 367-373, 1989.

[2] N. Baba, "A hybrid algorithm for finding a global minimum," Int. Journal of Control, Vol.37, pp. 929-942, 1983.

[3] D. Sohn, H. Hatakeyama, S. Mabu, K. Hirasawa, and J. Hu, "Adaptive Random Search with Intensification and Diversification Combined with Genetic Algorithm," Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol. 10, No. 6, pp. 954-963, 2006.

[4] D. Sohn, K. Hirasawa, and J. Hu, "Adaptive Random Search with Intensification and Diversification combined with Genetic Algorithm," Congress on Evolutionary Computation 2005 (CEC2005), pp. 1462-1469, 2005.

[5] K. Hirasawa, H. Miyazaki, and J. Hu, "Enhancement of RasID and Its Evaluation," T.SICE, Vol.38, No.9, pp. 775-783, 2002.

[6] K. Hirasawa, K. Togo, J. Hu, M. Ohbayashi, and J. Murata, "A New Adaptive Random Search Method in Neural Networks -RasID-," T.SICE, Vol.34, No.8, pp. 1088-1096, 1998.

[7] J. Hu, K. Hirasawa, and J. Murata, "RasID-Random Search for Neural Network Training," Journal of Advanced Computational Intelligence Informatics, Vol.2, No.4, pp. 134-141, 1998.

[8] J. Hu and K. Hirasawa, "Adaptive random search approach to identification of neural network model," Proc. of the 31st ISCIE Int. Symposium on Stochastic Systems Theory and its Applications, Yokohama, pp. 73-78, Nov. 11-12, 1999.
[9] J. Holland, "Adaptation in Natural and Artificial System," Ann Arbor, MIT University of Michigan Press, 1975.
[10] P. A. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Toward memetic algorithms," Caltech Concurrent Computation program, California Institute of Technology, Pasadena, Tech. Rep. 790, 1989.
[11] D. Molina, F. Herrera, and M. Lozano, "Adaptive Local Search Parameters for Real-Coded Memetic Algorithms," Congress on Evolutionary Computation 2005 (CEC2005), pp. 888-895, 2005.
[12] J. Matyas, "Random optimization," Automation and Remote Control, Vol.26, pp. 244-251, 1965.
[13] F. J. Solis and J. B. Wets, "Minimization by random search techniques," Mathematics of Operations Research, Vol.6, pp. 19-30, 1981.
[14] A. Torn and A. Zilinskas, "Global optimization," in Lecture Notes in Computer Science, 350, Berlin Germany, Springer-Verlag, 1989.
[15] T. Yamashita, K. Hirasawa, J. Hu, and J. Murata, "Multi-Branch Structure of Layered Neural Networks," Int. Conf. on Neural Information Processing, pp. 243-247, Nov., 2002.
[16] T. Yamashita, K. Hirasawa, and J. Hu, "Multi-Branch Structure and its Localized Property in Layered Neural Networks," Proc. of IJCNN, pp. 1039-1044, 2004.
[17] K. Hirasawa, X. Wang, J. Murata, J. Hu, and C. Jin, "Universal learning network and its application to chaos control," Neural Networks, Vol.13, pp. 239-253, 2000.
[18] K. Hirasawa, M. Ohbayashi, H. Fujita, and M. Koga, "Universal Learning Network Theory, Trans. of Institute of Electrical Engineers of Japan, Vol.116-C, No.7, pp. 794-801, 1996.
[19] M. Farzad, H. Tahersima, and H. Khaloozadeh, "Predicting the Mackey Glass Chaotic Time Series Using Genetic Algorithm," SICE-ICASE Int. Joint Conf., pp. 5460-5463, Oct., 2006.
[20] Z. Shi and M. Han, "Support Vector Echo-State Machine for ChaoticTime-Series Prediction," IEEE trans. Neural Networks, Vol.18, No.2, pp. 359-372, Mar., 2007.
[21] T. Kondo and J. Ueno, "Revised GMDH-Type Neural Network Algorithm with a Feedback Loop Identifying Sigmoid Function Neural Network," Int. Journal of Innovative Computing, Information and Control, Vol.2, No.5, pp. 985-996, 2006.

Name:
Dongkyu Sohn

Affiliation:
Ph.D. Student, Graduate School of Information, Production and Systems, Waseda University

Address:
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka 808-0135, Japan

Brief Biographical History:
2005- Received MS degree from Waseda University of Japan

Main Works:
• "Adaptive Random search with Intensification and Diversification combined with Genetic Algorithm," 2005 IEEE Congress on Evolutionary Computation, pp. 1462-1469.

Name:
Shingo Mabu

Affiliation:
Graduate School of Information, Production and Systems, Waseda University

Address:
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka 808-0135, Japan

Brief Biographical History:
2003 Received the M.E. degree from Kyushu University
2006 Received Ph.D. degree from Waseda University
2006- Visiting Lecturer, Waseda University
2007- Assistant Professor, Waseda University

Main Works:
• "A Graph-Based Evolutionary Algorithm: Genetic Network programming (GNP) and Its Extension using Reinforcement Learning," Evolutionary Computation, MIT Press, Vol.15, No.3, pp. 369-398, 2007.
• "Online Learning of Genetic Network Programming and its Application to Prisoner's Dilemma Game," IEEJ Trans. EIS, Vol.123, No.3, pp. 535-543, Mar., 2003.

Membership in Academic Societies:
• Institute of Electrical and Electronics Engineer (IEEE)
• The Society of Instrument and Control Engineers (SICE)
• The Institute of Electrical Engineers of Japan (IEEJ)

Name:
Kaoru Shimada

Affiliation:
Visiting Lecturer, Information, Production, and Systems Research Center, Waseda University

Address:
2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka 808-0135, Japan

Brief Biographical History:
1999 Received M. Med. Sci. degree from University of Tsukuba
2007 Received Ph.D. degree from Waseda University
2007- Visiting Lecturer, Waseda University

Main Works:
• "Genetic Network Programming with Acquisition Mechanism of Association Rules," Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.10, No.1 pp. 102-111, 2006.
• "Alternate Genetic Network Programming with Association Rules Acquisition Mechanisms Between Attribute Families," Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.10, No.6, pp. 954-963, 2006.

Membership in Academic Societies:
• Institute of Electrical and Electronics Engineer (IEEE)
• The Society of Instrument and Control Engineers (SICE)
• The Institute of Electrical Engineers of Japan (IEEJ)
• The Information Processing Society of Japan (IPSJ)

Name:
Kotaro Hirasawa

Affiliation:
Dr. Professor, Graduate School of Information, Production and Systems, Waseda University

Address:
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka 808-0135, Japan

Brief Biographical History:
1966 Received the M.E. degree from Kyushu University
1966 Joined Hitachi Corp.
1992- Professor, Kyushu University
2002- Professor, Waseda University

Main Works:
• "A Study of Evolutionary Multiagent Models Based on Symbiosis," IEEE Trans. on Systems, Man and Cybernetics -Part B-, Vol.35, No.1, pp. 179-193, 2006.
• "Propagation and control of stochastic signals through universal learning networks," Neural Networks, Vol.19, pp. 487-499, 2006.
• "A Function Localized Neural Network with Branch Gates," Neural Networks, Vol.16, pp. 1461-1481, Dec., 2003.
• "Chaos Control on Universal Learning Networks," IEEE Trans. on Systems, Man and Cybernetics, PART C, Vol.30, No.1, pp. 95-104, Jan., 2000.

Membership in Academic Societies:
• Institute of Electrical and Electronics Engineer (IEEE)
• The Association for Computing Machinery (ACM)
• The Society of Instrument and Control Engineers (SICE)
• The Institute of Electrical Engineers of Japan (IEEJ)
• The Information Processing Society of Japan (IPSJ)

Name:
Jinglu Hu

Affiliation:
Dr. Associate Professor, Graduate School of Information, Production and Systems, Waseda University

Address:
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka 808-0135, Japan

Brief Biographical History:
1997 Received Ph.D. degree from Kyushu Institute of Technology
1997- Research Associate, Kyushu University
2003- Associate Professor, Waseda University

Main Works:
• "A Homotopy Approach to Improving PEM Identification of ARMAX Model," Automatica, Vol.37, No.9, pp. 1323-1334, 2001.
• "A Quasi- ARMAX Approach to the Modeling of Nonlinear Systems," Int. Journal of Control, Vol.4, No.18, pp. 1754-1766, 2001.

Membership in Academic Societies:
• The Society of Instrument and Control Engineers (SICE)
• The Institute of Electrical Engineers of Japan (IEEJ)