Chapter III

Weight Vector Relocating Algorithm

After all the boundary vectors are found, the next step is to relocate the weight vector by using the boundary vectors as reference points. The new location of weight vector must be satisfied the condition stated in the previous chapter. The problem of weight vector relocation in this case can be formulated as an optimization problem with the following objective cost functions:

- **objective 1:** $E_t(W) = \frac{1}{2}\sum_{\mu=1}^{P}\sum_{i=1}^{P}(t_i^{\mu} - o_i^{\mu})^2 = 0,$
- **objective 2:** $E_l(W) = \sum_{l=1}^{L}\sum_{k=1}^{K_l}(f_{A,k} + f_{B,k})^2 = 0.$

We would like to find W_i' of each neuron i to make both $E_t(W)$ and $E_l(W)$ zeros. To achieve the objectives with respect to E_t which is less than maximum target error is allowed, first we must know the value of each element of W_i and then relocate W_i to obtain W_i'. Thus the following two consequent procedures are needed.

1. Train the network until it converges to a specified limit E_{t_0} and obtain the weight vector W_i and boundary vector pairs for each neuron i. This step is to satisfy the first objective.

2. Find W_i' satisfying the objective 2 with respect to the objective 1. (Lursinsap and Tanprasert [11] did not consider objective 1 in this step.)

For the first step, the network can be trained by using any existing learning rule such as error backpropagation algorithm [12, 15], but in this thesis we use a hybrid algorithm for finding the global minimum of error function of neural

networks that was reported in [22]. For the second step, the network will be trained by the algorithm in section 3.2. Before we present the novel method of weights relocation we will discuss two important concepts that will be realized later.

3.1 Pareto's Optimality Concept and Random Optimization Algorithm

3.1.1 Compromising $E_t(W)$ and $E_l(W)$

Since we have two objectives to optimize, target error $E_t(W)$ and location error $E_l(W)$, if $E_l(W) = 0$ it does not implies that $E_t(W)$ will be zero. Thus we will compromise these two objectives by using concepts of Pareto Optimal Set [23]. The notations of Pareto's optimality is based on the concept of dominance. Let $f(x) = (f_1(x), \cdots, f_n(x))$ represent a vector valued objective function and u and v represent two solutions. u dominates v, written $u << v$, if and only if for all i $f_i(u) \leq f_i(v)$ and there exists i, $f_i(u) < f_i(v)$. Solutions included in the Pareto Optimal Set are those that cannot be improved along any dimension without simultaneously being deteriorated along other dimensions.

3.1.2 Random Optimization Algorithm

Again $E(W)$ is the objective function, and $W(k) \in \chi$ the region to be searched.

A. Random optimization Method (Matyas [19])

1. Select the initial weights $W(0)$, and let M be the total number of steps.

2. Generate Gaussian random vector $\xi(k)$.
   If $W(k) + \xi(k) \in \chi$, go to step 3. Otherwise go to step 4.

3. If $E(W(k)+\xi(k)) < E(W(k))$, let $W(k+1) = W(k)+\xi(k)$.
   If $E(W(k)+\xi(k)) \geq E(W(k))$, let $W(k+1) = W(k)$.

4. If $k = M$, stop. If $k < M$, let $k = k+1$, and go to step 2.

B. Modified Random Optimization Method (Solis and Wets [20])

Solis and Wets modified the Matyas's random optimization method to find the global minimum of the objective function in a small number of steps and it has been reported that the random optimization method of Solis and Wets exhibits a very fast convergence. Moreover, this method ensures convergence to the global minimum of the objective function with probability 1.

The modified part is in step 3 that the search direction is reversed if it fails to improve the current value of the objection, and it exhibits faster convergence than the original scheme. Except for step 3, the rest steps are same.

- If $E(W(k)+\xi(k)) < E(W(k))$,
  let $W(k+1) = W(k)+\xi(k)$ and $\mathbf{b}(k+1) = 0.4\xi(k) + 0.2\mathbf{b}(k)$.

- If $E(W(k)+\xi(k)) \geq E(W(k))$ and $E(W(k)-\xi(k)) < E(W(k))$,
  let $W(k+1) = W(k)-\xi(k)$ and $\mathbf{b}(k+1) = -0.4\xi(k) + \mathbf{b}(k)$.

- Otherwise, let $W(k+1) = W(k)$ and $\mathbf{b}(k+1) = 0.5\mathbf{b}(k)$.

Where $\mathbf{b}(0) = 0$ and $\mathbf{b}(k)$ is the bias which becomes the center of the Gaussian random vector $\xi(k)$ at the $k'th$ step.

In this thesis, we propose a hybrid of random optimization algorithm and Pareto's optimality concept approach to solve two-objective optimization.

3.2 Hybrid Algorithm: Pareto's Optimality Concept and Random Optimization

In this thesis, the Pareto's optimality objective function is formed as follows.

Let $E(W) = (E_t(W), E_l(W))$, W be a current solutions and W' be a candidate. We will accept W' as a new solution if and only if $E_t(W') \leq E_t(W)$ and $E_l(W') < E_l(W)$.

The random optimization algorithm will change W to W' iteratively such that the location error $E_l(W)$ is minimized and $E_t(W)$ less than or equal to the maximum allowable data-fitting error, E_{t_0}. The weight change for the $(k+1)^{th}$ epoch can be expressed as follows. Let M be the maximum number of epochs and $E_{lmax}(W)$ the acceptable maximum location error.

1. Set $E_{t_0} = E_t(W(0))$ and $k = 1$.

2. Generate the Gaussian random vector $\xi(k)$ with initial center at $\mathbf{b} = 0$ and standard deviation σ.
   If $W(k) + \xi(k)$ and $W(k) - \xi(k)$ still correctly classifies input vectors then go to step 3 else go to step 4.

3. If $E_l(W(k) + \xi(k)) < E_l(W(k))$ and $E_t(W(k) + \xi(k)) \leq E_{t_0}$
   then let $W'(k+1) = W(k) + \xi(k)$ and $\mathbf{b}(k+1) = 0.4\xi(k) + 0.2\mathbf{b}(k)$
   else If $E_l(W - \xi) < E_l(W)$ and $E_t(W - \xi) \leq E_{t_0}$
   then let $W'(k+1) = W(k) - \xi(k)$ and $\mathbf{b}(k) = \mathbf{b}(k) - 0.4\xi(k)$
   otherwise let $W'(k+1) = W(k)$ and $\mathbf{b}(k+1) = 0.5\mathbf{b}(k)$.

4. If $k = M$ or $E_l(W'(k)) \leq E_{lmax}$ then stop the total calculation else let $k = k + 1$ and go to Step 2.

The basic idea of the weight perturbation by random optimization of Solis and Wets is to transit the system to a state at which the system has lower square

error. Consequently, the system is perturbed to a state away from the local minimum. Since the domain of W_i is closed [22] and the next theorem will show that the random optimization of Solis and Wets converges with probability 1. It is noteworthy that this evolution is not restricted to one node. The technique can be simultaneously done at every neuron to achieve lower total location error.

Theorem 4 *[22] Let W be a compact region in which we have to find an appropriate weight vector w. Further, let $\hat{w}$ be one of the points that give the global minimum of $E(W)$ in W:*

$$E(\hat{w}) = \min_{w \in W} E(w);\ w \in W$$

Let $\hat{W}_\epsilon$ be the region such that $\hat{W}_\epsilon = \{w : |E(w) - E(\hat{w})| < \epsilon, w \in W\}$. Assume the following:

1. *for any positive number δ, the Euclidean measure of $U_\delta(\hat{w}) \cap W$ is positive, where $U_\delta(\hat{w}) = \{w : ||w - w'|| < \delta\}$.*
2. *There is no bound on M. That is, one can calculate $w(k)$ $(k = 1, 2, \ldots)$ as many times as one wants.*
3. *$E(w)$ belongs to class C^1.*
4. *$E(\hat{w}) < \epsilon$.*

Then the algorithm ensures the following:

(a) For any positive number ϵ, $\lim_{k \to \infty} P\{w | w^k \in W_\epsilon\} = 1$.

(b) $E(w^k)$ converges to $E(\hat{W})$ with probability 1.

Proof see [22].

Another random optimization algorithm is of Sun et al. [24] and the modified version are adapted to solve the problem. The detail of algorithm was appeared in Sunat and Lursinsap [27].