# Supervised Training of Neural Networks via Ellipsoid Algorithms

**Man-Fung Cheung**
**Kevin M. Passino**
**Stephen Yurkovich**
*Department of Electrical Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210 USA*

**In this paper we show that two ellipsoid algorithms can be used to train single-layer neural networks with general staircase nonlinearities. The ellipsoid algorithms have several advantages over other conventional training approaches including (1) explicit convergence results and automatic determination of linear separability, (2) an elimination of problems with picking initial values for the weights, (3) guarantees that the trained weights are in some "acceptable region," (4) certain "robustness" characteristics, and (5) a training approach for neural networks with a wider variety of activation functions. We illustrate the training approach by training the *MAJ* function and then by showing how to train a controller for a reaction chamber temperature control problem.**

## 1 Introduction

In this paper we will introduce two ellipsoid algorithms that have been used in system identification and parameter estimation to the training of artificial neural networks (ANN) (Widrow and Lehr 1990; Barto 1989; Lippmann 1987; Beale and Jackson 1990; Antognetti and Milutinovic 1991) with general staircase nonlinearities. The utility of the ellipsoid algorithm [some of the earliest uses for parameter set estimation appear in Fogel and Huang (1982)] is motivated by its ease in use and implementation, where its recursive nature makes the training process very attractive computationally. The staircase nonlinearity is a generalization of the hard limiter and such a generalization can be useful in classifying patterns into several linearly separable regions (such as parallel strips in two dimensions). The two ellipsoid algorithms that we propose to use for training are the "Optimal Volume Ellipsoid" (OVE) (Cheung 1991; Cheung *et al.* 1993) algorithm and the "Underbounding Ellipsoid" (UBE) algorithm introduced here. The OVE algorithm results in an ellipsoidal set overbounding the feasible set of weights while the UBE algorithm results in an underbounding ellipsoid inscribed inside the feasible set of

*Neural Computation* **6, 748–760** (1994) 

weights consistent with the training data set. It is guaranteed that the center of the ellipsoid using the OVE algorithm is a feasible solution after the algorithm converges (that is, convergence in the weight estimates if there exists a solution). In fact, all weights inside the final ellipsoid from the UBE algorithm are feasible if the UBE ellipsoids' final volume is nonzero. Several applications are studied and the paper closes with a discussion on the advantages/disadvantages of OVE/UBE.

## 2 The OVE/UBE Algorithms

The objective in parameter set estimation is to identify a feasible set of parameters that is consistent with the measurement data and the model structure used. One can interpret the set estimate as some nominal parameter estimate accompanied by a quantification of the uncertainty parametrically around the nominal model. An important feature in parameter set estimation is the guaranteed inclusion of the true mapping that is not exactly known.

In this section we will provide a brief introduction to the OVE algorithm introduced in Cheung (1991) and Cheung *et al.* (1993) for parameter set estimation in system identification and a discussion on finding an underbounding ellipsoid for the feasible parameter set consistent with the training data set. The underbounding ellipsoid (UBE) algorithm is similar to the overbounding ellipsoid algorithm (OVE), except that the underbounding ellipsoid has the feature that all points in the ellipsoid are feasible parameters. In the next section we will show how the algorithms can be used to train neural networks.

Consider the following $k$th pair of parallel linear constraints

$$|y_k - W^T X_k| \leq \gamma$$

where $y_k \in \Re$, $\gamma \in \Re$, $X_k \in \Re^r$ are known, and $W \in \Re^r$ is the unknown parameter (weight) vector. Let $\mathcal{F}^k \subset \Re^r$ be the set of feasible parameters given $k$ constraints. That is,

$$\mathcal{F}^k = \{W : |y_i - W^T X_i| \leq \gamma, i = 1, \ldots, k\} \tag{2.1}$$

Define also

$$\mathcal{F}_k = \{W : |y_k - W^T X_k| \leq \gamma\} \tag{2.2}$$

and

$$E_k = \{W : (W - W_k)^T P_k^{-1} (W - W_k)\} \tag{2.3}$$

where $W_k$, the $k$th estimate of the unknown parameters, is the center of an ellipsoid $E_k$ and $P_k^{-1}$ is a positive definite matrix that characterizes the size and shape of the ellipsoid.

Suppose the ellipsoid $E_k$ is an overbounding ellipsoid for $\mathcal{F}^k$. The OVE algorithm finds the smallest volume ellipsoid $E_{k+1}$ containing the

intersection of $\mathcal{F}_{k+1}$ and $E_k$. The intersection is essentially the portion of $E_k$ cut out by the two parallel hyperplanes defined in $\mathcal{F}_k$.

**Theorem 1.** The OVE algorithm is comprised of the following recursive equations:

$$W_{k+1} = W_k + \frac{\tau_k P_k X_{k+1}}{(X_{k+1}^T P_k X_{k+1})^{1/2}}, \qquad P_{k+1} = \delta_k \left( P_k - \sigma_k \frac{P_k X_{k+1} X_{k+1}^T P_k}{X_{k+1}^T P_k X_{k+1}} \right)$$

where if

(a) $\alpha_k \neq \beta_k$, then

$$\delta_k = \frac{(\tau_k + 1)^2(\beta_k - \alpha_k) - \tau_k(1 + \alpha_k)(2\beta_k - \alpha_k - 1)}{\tau_k + \beta_k - \alpha_k}, \qquad \sigma_k = \frac{-\tau_k}{\beta_k - \alpha_k}$$

and $\tau_k$ is the real solution of

$$(r+1)\tau_k^2 + \left\{ \frac{(1+\alpha_k)(\alpha_k - 2\beta_k + 1)}{\beta_k - \alpha_k} + 2[r(\beta_k - \alpha_k) + 1] \right\} \tau_k + r\alpha_k(\alpha_k - 2\beta_k) + 1 = 0$$

such that $\alpha_k - 2\beta_k < \tau_k < \alpha_k$;

(b) $\alpha_k = \beta_k$, then

$$\delta_k = \frac{r}{r-1}(1 - \beta_k^2), \qquad \sigma_k = \frac{1 - r\beta_k^2}{1 - \beta_k^2}, \qquad \tau_k = 0,$$

where $\alpha_k$ and $\beta_k$ are defined as

$$\beta_k = \frac{\gamma}{\sqrt{X_{k+1}^T P_k X_{k+1}}} \qquad \alpha_k = \frac{y_{k+1} + \gamma - X_{k+1}^T W_k}{\sqrt{X_{k+1}^T P_k X_{k+1}}}$$

**Proof.** The proof appears in Cheung (1991) and Cheung *et al.* (1993) and is available from the authors on request.

Suppose now that the ellipsoid $E_k$ is an *underbounding* ellipsoid for $\mathcal{F}^k$. The underbounding ellipsoid algorithm finds the largest volume ellipsoid $E_{k+1}$ underbounded in the intersection of $\mathcal{F}_{k+1}$ and $E_k$ such that (1) the center of the new ellipsoid is located midway between the parallel hyperplanes defined in $\mathcal{F}_{k+1}$ and (2) the new ellipsoid touches both hyperplanes.

**Theorem 2.** The UBE algorithm is comprised of the following recursive equations:

$$W_{k+1} = W_k + \frac{\tau_k P_k X_{k+1}}{(X_{k+1}^T P_k X_{k+1})^{1/2}} P_{k+1} = \delta_k \left( P_k - \sigma_k \frac{P_k X_{k+1} X_{k+1}^T P_k}{X_{k+1}^T P_k X_{k+1}} \right),$$

where

$$\tau_k = \alpha_k - \beta_k \sigma_k = 1 - \frac{\beta_k^2}{\delta_k}$$

and $\delta_k$ is the solution of

$$\delta_k^2 + (\tau_k^2 - \beta_k^2 - 1)\delta_k + \beta_k^2 = 0$$

such that $\delta_k \tau_k / (\delta_k - \beta_k^2) \leq 1$; $\alpha_k$ and $\beta_k$ are defined as in Theorem 1.

If $\alpha_k > 1$, reset $\beta_k$ to $\beta_k - [(\alpha_k - 1)/2]$ and then reset $\alpha_k$ to one; on the other hand, if $2\beta - \alpha_k > 1$, reset $\beta_k$ to $(1 + \alpha_k)/2$.

**Proof.** In this proof, the same approach in deriving the OVE algorithm is used here and therefore one has the same form of updating equations for $W_k$ and $P_k$. Using the affine transformation $W = W_k + J\hat{W}$ where $P_k = JJ^T$, the ellipsoid $E_k$ is transformed to a unit radius hypersphere denoted as

$$\hat{E}_k = \{\hat{W} : \hat{W}^T \hat{W} \leq 1\}$$

and the new ellipsoid may be parameterized as

$$\hat{E}_{k+1} = \left\{ \hat{W} : \frac{(\hat{w}_1 - \tau_k)^2}{\beta_k^2} + \frac{\hat{w}_2^2}{\delta_k} + \cdots + \frac{\hat{w}_r^2}{\delta_k} \leq 1 \right\}$$

such that the $\hat{w}_1$ is perpendicular to the transformed parallel hyperplanes. Since $\tau_k$ [given by $(\alpha_k - \beta_k)$] is the midpoint between the hyperplane on $\hat{w}_1$ axis and since the ellipsoid touches both hyperplanes, the semi-axial length of the new ellipsoid along $\hat{w}_1$ axis is therefore $\beta_k$ as the hyperplanes are $2\beta_k$ apart in the transformed coordinate. Finding the largest volume $\hat{E}_{k+1}$ is equivalent to finding the largest $\delta_k$ such that $\hat{E}_{k+1}$ is still an underbounding ellipsoid. The largest allowable $\delta_k$ is achieved when the surfaces of the two ellipsoids $\hat{E}_k$ and $\hat{E}_{k+1}$ touch each other. The surfaces are

$$\hat{w}_1^2 + \hat{w}_2^2 + \cdots + \hat{w}_r^2 = 1$$

and

$$\frac{(\hat{w}_1 - \tau_k)^2}{\beta_k^2} + \frac{\hat{w}_2^2}{\delta_k} + \cdots + \frac{\hat{w}_r^2}{\delta_k} = 1$$

After some manipulation, one gets

$$(\delta_k - \beta_k^2)\hat{w}_1^2 - 2\delta_k \tau_k \hat{w}_1 + \delta_k \tau_k^2 + \beta_k^2 - \beta_k^2 \delta_k = 0 \tag{2.4}$$

Since the surfaces touch each other, then the discriminant in 2.4 must vanish and as a result, one gets

$$\delta_k^2 + (\tau_k^2 - \beta_k^2 - 1)\delta_k + \beta_k^2 = 0$$

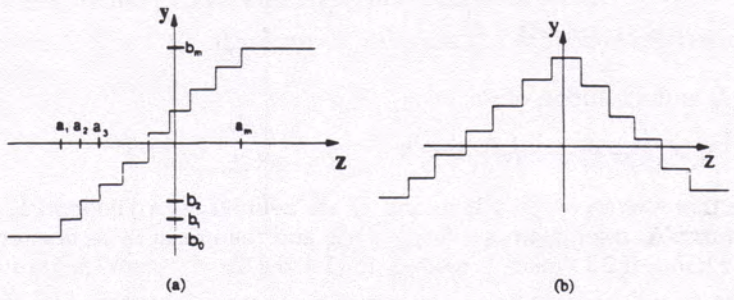


Figure 1: Examples of staircase nonlinearities.

It is obvious that the solution for $\hat{w}_1$ when the surfaces touch must be less than (or equal to) one, and therefore only the $\delta_k$ that results in $\hat{w}_1 = \delta_k \tau_k / (\delta_k - \beta_k^2) \leq 1$ is valid. This completes the proof of Theorem 2. □

The OVE and UBE algorithms can be initialized with a sufficiently large $E_0$ containing the feasible parameter set, where $W_0 = 0$ and $P_0 = \frac{1}{\epsilon} I$ (with $0 < \epsilon \ll 1$) are typical starting values. Note that for the OVE algorithm, if $E_0$ contains the parameter set, so does all other $E_k$ and it is therefore important to make sure that $E_0$ is large enough so that the new $E_k$ computed are meaningful. For the UBE algorithm, after iterating through the entire data set once, the resulting ellipsoid will be an underbounding ellipsoid for the data set in which every point in the ellipsoid is a feasible parameter vector consistent with the training set. However, the underbounding ellipsoid may vanish even though the feasible parameter set is nonempty.

## 3 Supervised Training of a Single Perceptron via OVE/UBE

Consider a single-layer perceptron with one neuron that has $n$ inputs, $x_1, \ldots, x_n$, and one output $y$; $w_0$ is the weight on the fixed bias input $x_0 = 1$; $w_1, \ldots, w_n$ are the weights on the inputs $x_1, \ldots, x_n$, respectively; $z$ is the output of the summer and the input to $f(\cdot)$ where $f(\cdot)$ is a fixed "staircase nonlinearity function" that has $m + 1$ distinct quantized steps such that $f(a_i) = b_i$ as shown in Figure 1a. As a matter of fact, the staircase nonlinearity shown in Figure 1 can have arbitrary shape as long as the steps have distinct levels, and it can also be used to approximate a sigmoid function and other nonlinearities (for example, Fig. 1b is another possible function).

Without loss of generality, assume the nonlinear function $f(\cdot)$ is completely known, and that $a_i$ and $b_i$ are all given. Let $N$ be the number of training data pairs in the data set

$$S = \{(X_k, y_k) : k \in [1, N]\}$$

and let

$$X = [x_0 \quad x_1 \quad \cdots \quad x_n]^T$$

be the input vector. (Note: the parameter $r$ in Section 2 is equal to $n+1$ here.)

The generic problem of training a single perceptron is to find a weight vector

$$W = [w_0 \quad w_1 \quad \cdots \quad w_n]^T$$

such that the perceptron can regenerate the input–output patterns in the training set, and *generalize* so as to capture the complete input–output mapping of the underlying process. In this paper, our goals are (1) to find an ellipsoidal set using the OVE algorithm to overbound the feasible set of weights consistent with the training data set $S$ (the motivation for using the OVE algorithm is that intrinsic in the algorithm, the center of the resulting ellipsoid is guaranteed to be a feasible solution, after convergence, for designing a perceptron to implement the mapping defined by the available input–output patterns), and (2) to find an ellipsoidal set that underbounds the feasible set of weights consistent with the training data set $S$ using the UBE algorithm. The UBE ellipsoid characterizes a feasible set of weights that shows what variations in the weights can occur for which we are still guaranteed to implement the proper mapping. Consequently, the final UBE ellipsoid provides a characterization of "robustness" of the neural network mapping with respect to variations in the weights (in case of uncertainties in the implementation process or in case they vary after implementation). The motivation for using the UBE algorithm is that the algorithm must be executed only once for the training set to give a feasible set for designing a perceptron to implement the mapping in interest (the center of the final UBE ellipsoid can be used for the weights).

Suppose a data pair $(X_{k+1}, y_{k+1})$ is given and that $y_{k+1} = b_i$; then the interval in which $z_k$ [given by $(X_{k+1}^T \theta)$] lies is known to be $[a_i, a_{i+1})$. That is

$$a_i \leq X_{k+1}^T W < a_{i+1} \tag{3.1}$$

where $a_i$ and $a_{i+1}$ are known. However, the inequalities in 3.1 define a set that is convex but not closed. In order to utilize OVE and UBE that

work on convex and closed sets, it is desired to make the set closed by relaxing the right-hand inequality in 3.1, adding very little conservatism, as

$$a_i \leq X_{k+1}^T W \leq a_{i+1} \tag{3.2}$$

which is a superset of that in 3.1. As a result, the feasible set of weights can be succinctly defined by the following $2N$ inequality constraints: $a_i \leq X_k^T W \leq a_{i+1}$ for $k = 1, \ldots, N$, and $i \in [0, m]$ is such $y_k = b_i$.

For $\mathcal{F}_{k+1}$ which corresponds to the feasible region for the data pair $(X_{k+1}, y_{k+1})$, the constraints are

$$y_{k+1} - \gamma \leq X_{k+1}^T W \leq y_{k+1} + \gamma \tag{3.3}$$

**Theorem 3.** Given that $y_{k+1} = b_i$ or a feasible set of weights defined by equation 3.2 and a previous bounding ellipsoid defined in equation 2.3, the OVE/UBE algorithm can be used to find a new overbounding/underbounding ellipsoid with the following definitions of $\alpha_k$ and $\beta_k$ :
(a) $i \neq 0$ or $i \neq m$

$$\alpha_k = \frac{a_{i+1} - X_{k+1}^T W_k}{G_k}, \qquad \beta_k = \frac{1}{2}\left(\alpha_k - \frac{a_i - X_{k+1}^T W_k}{G_k}\right),$$

where $G_k = \sqrt{X_{k+1}^T P_k X_{k+1}}$, $W_k$ and $P_k$ are the parameters associated with the previous ellipsoid that is obtained through training the data up to the $k$th pair of data.

(b) $i = 0$ or $i = m$

That is when $y_{k+1} = b_0$ (or $b_m$), the first (or the last) quantized output, then

$$z_k < a_1 \quad (\text{or } > a_m)$$

Incorporating a small degree of conservatism as

$$z_k \leq a_1 \quad (\text{or } \geq a_m)$$

results in

$$\alpha_k = \frac{a_1 - X_{k+1}^T W_k}{G_k} \quad (\text{or } \alpha_k = 1), \qquad \beta_k = \frac{\alpha_k + 1}{2}$$

$$\left[\text{or } \beta_k = \frac{1}{2}\left(1 - \frac{a_m - X_{k+1}^T W_k}{G_k}\right)\right]$$

**Sketch of the Proof.** The result in (a) is obtained by comparing the left-hand sides and the right-hand sides of 3.2 and 3.3, respectively, and then determining the new parametrizations for $\alpha_k$ and $\beta_k$. In case (b), essentially only one hyperplane cuts the previous ellipsoid. Therefore, one can move the noncutting hyperplane to touch the ellipsoid corresponding to either one of the following conditions:

$$\alpha_k = 1 \qquad \text{or} \qquad \alpha_k - 2\beta_k = -1$$

depending on which hyperplane intersects. For details, see Cheung (1991) and Cheung *et al.* (1993).

## 4 Applications

The OVE algorithm has been shown to be convergent in Cheung (1991) and Cheung *et al.* (1993), in that the volume at each iteration is nonincreasing and the center of the elliposid will converge as the size of the data set tends to infinity. In our application to ANN where the number of data are finite, the convergence property must be maintained. By repeatedly applying the OVE algorithm over the entire finite training set, it is easy to show that OVE retains its convergence properties. This is the case because the OVE algorithm views the repetition of the finite training set as a long (possibly infinite) data sequence. Therefore the convergence properties are guaranteed and the center of the ellipsoid, after convergence, must be inside the feasible set and be a feasible solution to the mapping of the finite data set. If this is not true, then a contradiction results since, as shown in Cheung (1991) and Cheung *et al.* (1993), if the center of the ellipsoid does not satisfy one of the parallel hyperplane constraints, one is guaranteed that a smaller volume ellipsoid can be found implying that the ellipsoids have not yet converged.

The notion of convergence is difficult to quantitatively guarantee for finite sets. For the examples studied below, the following stopping criterion is used along with the OVE algorithm. The OVE algorithm ceases if

$$\frac{V(l) - V(l+1)}{V(l)} \leq \epsilon$$

where $V(l)$ denotes the volume of an ellipsoid after sweeping through the entire finite data set $l$ times, and $\epsilon$ is a small positive number. Note that although the choice of $\epsilon$ is heuristic, the algorithm is guaranteed to meet this condition (for some finite $l$ for some $\epsilon$). However, this does not guarantee that the center of the last ellipsoid will be a feasible solution; a feasible solution test must be conducted for validation and a smaller $\epsilon$ may be necessary if the ellipsoid center turns out to be a nonfeasible solution. Often $V(l) - V(l+1) = 0$ for some $l$ implying that the algorithm has actually converged with respect to the available data and that repeatedly applying OVE over the data set will give no new information about the bounding ellipsoid. In other words, the center of the final ellipsoid must be a feasible solution.

In the following two examples, the initial ellipsoid is set to be a sphere centered at the origin with radius 10 units for each case. For the OVE implementation, $\epsilon = 0.001$ is used and the feasibility test is passed for all examples. For the UBE implementation, the algorithm is iterated through the training data set once to give an ellipsoidal set of feasible weights.

**4.1 Perceptron with Hard-Limiter as Nonlinearity.** It is known (Widrow and Lehr 1990; Lippmann 1987) that a single perception can be used to linearly classify input patterns into two different groups. Essentially,

Table 1: I/O Table for *MAJ* Logic

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $x_1$ | 1 | 1 | 1 | 1 | −1 | −1 | −1 | −1 |
| $x_2$ | 1 | 1 | −1 | −1 | 1 | 1 | −1 | −1 |
| $x_3$ | 1 | −1 | 1 | −1 | 1 | −1 | 1 | −1 |
| $y$ | 1 | 1 | 1 | −1 | 1 | −1 | −1 | −1 |

the perceptron with a hard-limiter as the nonlinearity divides the input space into two regions separated by a hyperplane (a line in two-dimensional space). Here, the ellipsoid algorithms with the parametrizations given in Theorem 1 and Theorem 2 are used to train a perceptron to realize the linearly separable logic functions. The example to be studied is the *MAJ* logic function (*OR*, *XOR*, and *AND* logic were also implemented, but not included here). The functional mapping table is given in Table 1 where the inputs are denoted as $x_i$ and the output as $y$.

For training with the OVE algorithm, the entire data set is swept through five times before satisfying the stopping criterion. The following result was obtained:

$$W_{40}(MAJ) = \begin{bmatrix} -0.0671 \\ 3.7428 \\ 3.7548 \\ 3.7799 \end{bmatrix}$$

For training with the UBE algorithm, the following results were obtained:

$$W_8(MAJ) = \begin{bmatrix} 1.6296 \\ 3.6854 \\ 3.7950 \\ 3.8127 \end{bmatrix}$$

and the singular values of the associated $P_k$ matrix which correspond to the square of the semiaxial lengths are (1.0607, 0.8389, 0.6511, 0.3884). Without knowing the orientation of the final ellipsoid, the minimum amount of variation allowed in each weight around the center $W_k$ is given by $\sqrt{\sigma(P_k)/r}$ where $\sigma(P_k)$ is the smallest singular value of $P_k$. Hence the UBE algorithm successfully trained the perceptron and the amount of variation allowed in each weight is at least 0.3116. This provides a range (consistent with the training data) that the weights may vary in the implementation of an ANN when the center estimate is used to implement the weights (that is, it provides a characterization of the robustness of the ANN map).

**4.2 Perceptron with a Staircase Nonlinearity—A Control Application.** Consider a reaction chamber temperature following the control

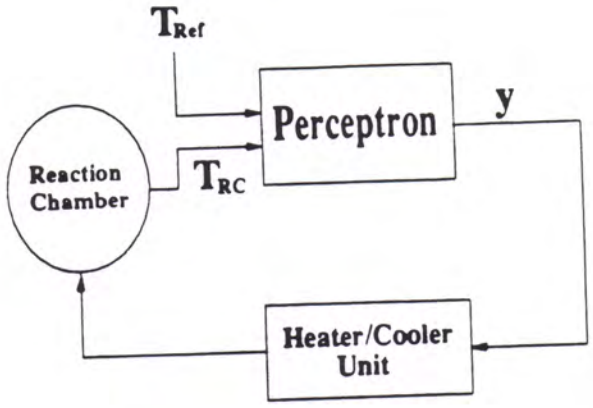


Figure 2: Temperature following control problem.

problem shown in Figure 2. $T_{RC}$ is the temperature of the reaction chamber which is desired to follow the reference temperature $T_{Ref}$. The temperature inside the reaction chamber can be controlled by appropriately switching on the heater/cooler unit. The following rules for activating the heater/cooler unit are given:

1. If $T_{RC} < T_{Ref} - 3$, turn on the heater;
2. If $T_{RC} > T_{Ref} + 3$, turn on the cooler;
3. If $|T_{RC} - T_{Ref}| \leq 3$, neither the heater nor the cooler is on.

Assume that the heater/cooler unit is under a single control $y$, the output of a neural net controller or the perceptron, so that when

$y = 1$, heater is on;
$y = -1$, cooler is on;
$y = 0$, the heater/cooler unit is idle.

A training data set of 12 data pairs is used which is generated according to the rules above, and shown in Table 2. Figure 3 shows the training data pattern on the $T_{RC} - T_{Ref}$ plane. The solid parallel lines separate the plane into three regions: Region A corresponds to case (1) with $y = 1$; Region B corresponds to case (2) with $y = -1$; and Region C corresponds to case (3) with $y = 0$. The problem here is to use a perceptron with a staircase nonlinearity to classify the input pattern into three different classes. The staircase nonlinearity used here has the following parameters: $m = 2$, $a_1 = -3$, $a_2 = 3$, $b_0 = -1$, $b_1 = 0$ and $b_2 = 1$; $x_1$

Table 2: Training Set for the Heater/Cooler Unit Activation

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $T_{RC}$ | 3 | 1 | 0 | 1.5 | 9 | 0 | −3 | 2 | 1 | −3 | 4 | 5 |
| $T_{Ref}$ | 5.5 | 4.5 | 1 | −2 | 14 | 2.4 | −3.9 | 5.4 | −3 | 7 | −2 | 1.5 |
| $y$ | 0 | 1 | 0 | −1 | 1 | 0 | 0 | 1 | −1 | 1 | −1 | −1 |

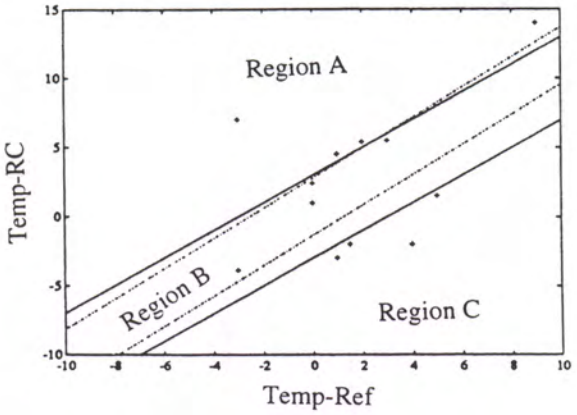


Figure 3: Training data pattern for temperature control problem.

and $x_2$ correspond to $T_{Ref}$ and $T_{RC}$, respectively. The training data set is swept through 48 times until the OVE algorithm stops (convergence is achieved); the following results were obtained:

$$W_{576} = \begin{bmatrix} -1.0296 \\ -1.5859 \\ 1.4492 \end{bmatrix}$$

The parallel separating lines from the perceptron using $W_{576}$ are indicated as dash–dot lines shown in Figure 3. Clearly, the perceptron can categorically separate the training set appropriately. In fact, if a larger set of training instances is used, the classification boundaries from the perceptron will closely match the desired ones (solid parallel lines).

For the training with UBE in this example, no feasible solution was obtained. This can happen as each time the UBE algorithm is used, part of the feasible parameter set may be discarded and eventually an intermediate underbounding ellipsoid may not contain any feasible solutions.

## 5 Discussion

In this paper we have examined how two ellipsoid algorithms that are useful in general system identification studies can be used for the training of neural networks. Both ellipsoid algorithms provide their unique features to neural net training and implementation. In particular:

1. The choice of the initial ellipsoid $E_0$ in both the OVE and UBE algorithms can be used as an instrument to confine the consideration of physical realizable weights in an ANN. This may bear some practical significance when hardware implementation is considered. The ellipsoid algorithm approach to training a perceptron has another distinct advantage over other training algorithms in that it solves the problem of choosing initial weights. One merely needs to pick a large enough initial ellipsoid that guarantees the overbounding of the feasible set.

2. The OVE algorithm gives a convergent estimate and can be used as an automatic test for linear separability of input–output mappings. However, this is not necessarily true for the UBE algorithm as indicated in the control example of the last section.

3. The results from the UBE algorithm give a characterization of the feasible weights to indicate the flexibility in implementing a perceptron to realize a certain mapping; it may have a strong bearing on the robustness of a perceptron with respect to disturbances in the inputs as well. The center of the UBE ellipsoidal set could be the best choice of weights as they may vary slightly during implementation.

4. The UBE algorithm needs to be executed the same number of times as the number of data patterns available for training to give a feasible set. However, the UBE algorithm may fail to characterize a feasible ellipsoid even though the feasible set is nonempty (again as indicated in the control application example). Nevertheless, this can be complemented by using the OVE algorithm as it is guaranteed that the center of the overbounding ellipsoid is a feasible solution if the feasible set is nonempty, after the algorithm has converged.

5. For training of multilayer perceptrons, because of the nondifferentiability of the activation function, a heuristic approach has been used (but not reported on here due to space constraints) to train

each perceptron independently. This required assigning input–output patterns for each perceptron appropriately so that the entire ANN works in the manner desired.

6. Finally, we note that the complexity of the OVE and UBE algorithms is discussed in detail in Cheung *et al.* (1993).

In this initial investigation into using ellipsoid algorithms for training ANN we have shown several advantages of OVE/UBE; however, much work remains. For instance, there is the need to extend the results (including the desirable convergence and robustness properties) to the training of general multilayer perceptrons with general staircase nonlinearities.

## Acknowledgments

K. Passino was supported in part by an Engineering Foundation Research Initiation Grant.

## References

Antognetti, P., and Milutinovic, V. eds. *Neural Networks: Concepts, Applications and Implementation*. Prentice Hall, New York.

Barto, A. G. 1989. Connectionist learning for control: An overview. COINS Tech. Rep. 89-89 5, University of Massachusetts, Amherst.

Beale, R., and Jackson, T. 1990. *Neural Computing: An Introduction*. Adam Hilge, New York.

Cheung, M. F. 1991. On optimal algorithms for parameter set estimation. Ph.D. thesis, The Ohio State University, Columbus, OH.

Cheung, M. F., Yurkovich, S., and Passino, K. M. 1993. An optimal volume ellipsoid algorithm for parameter set estimation. *IEEE Conf. Decision Control* (Brighton, UK), pp. 969–974, 1993. An expanded version will appear in *IEEE Transact. Automatic Control*, Vol. 38, No. 8, pp. 1292–1296.

Fogel, E., and Huang, Y. F. 1982. On the value of information in system identification-Bounded noise case. *Automatica* **18**(2), 229–238.

Lippmann, R. P. 1987. An introduction to computing with neural nets. *IEEE ASSP Mag.* 4–22.

Widrow, B., and Lehr, M. A. 1990. 30 years of adaptive neural networks: Perceptron, Madaline, and back propagation. *Proc. IEEE* **78**.

Received April 14, 1992; accepted September 14, 1993.