Correlation Learning Rule

The *correlation learning rule* is derived by starting from the criterion function

$$J(\mathbf{x}) = -\sum_{i=1}^{m} y^i d^i \qquad (2.49)$$

where $y^i = \left(\mathbf{x}^i\right)^{\mathrm{T}} \mathbf{w}$, and performing gradient descent to minimize *J*. Note that minimizing *J*(**w**) is equivalent to maximizing the correlation between the desired target and the corresponding linear unit's output for all $\mathbf{x}^i$, *i* = l, 2, ... , *m*. Now, employing steepest gradient descent to minimize *J*(**w**) leads to the learning rule:

$$\begin{cases} \mathbf{w}^1 = \mathbf{0} \\ \mathbf{w}^{k+1} = \mathbf{w}^k + \rho d^k \mathbf{x}^k \end{cases} \qquad (2.50)$$

By setting ρ to 1 and completing one learning cycle using Equation (2.50), we arrive at the weight vector $\mathbf{w}^*$ given by

$$\mathbf{w}^* = \sum_{i=1}^{m} d^i \mathbf{x}^i = \mathbf{X}\mathbf{d} \qquad (2.51)$$

where **X** and **d** are as defined above. Note that Equation (2.51) leads to the minimum SSE solution in Equation (2.38) if $\mathbf{X}^\dagger = \mathbf{X}$. This is only possible if the training vectors $\mathbf{x}^k$ are encoded such that $\mathbf{X}\mathbf{X}^\mathbf{T}$ is the identity matrix (i.e., the $\mathbf{x}^k$ vectors are orthonormal).

Another version of this type of learning is the *covariance learning rule.* This rule is obtained by steepest gradient descent on the criterion function

$$J(\mathbf{w}) = -\left|\sum_{i=1}^{m}\left(y^i - \langle y \rangle\right)\left(d^i - \langle d \rangle\right)\right|.$$

Here, $\langle y \rangle$ and $\langle d \rangle$ are computed averages, over all training pairs, for the unit's output and the desired target, respectively. Covariance learning provides the basis of the cascade-correlation net.

The Delta Rule

The following rule is similar to the μ-LMS rule except that it allows for units with a differentiable nonlinear activation function *f*. Figure 2-7 illustrates a unit with a sigmoidal activation function. Here, the unit's output is $y = f(net)$, with *net* defined as the vector inner product $\mathbf{x}^T\mathbf{w}$.

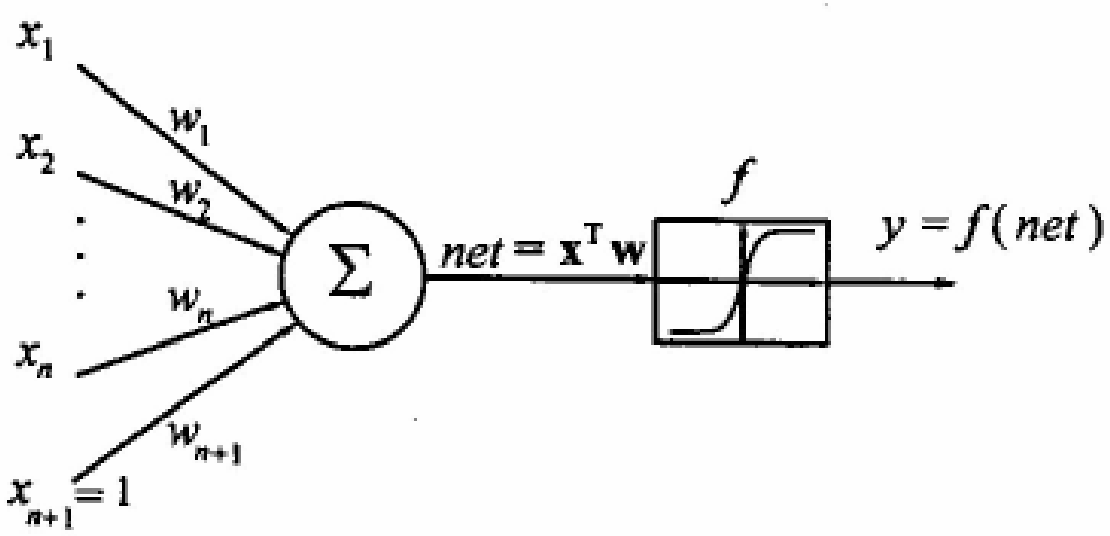

Figure 2-7 A perceptron with a differentiable sigmoidal activation function.

Again, consider the training pairs $\{\mathbf{x}^i, d^i\}$, i= l, 2, ... , m, with $\mathbf{x}^i \in R^{n+1}$ ($x^i_{n+1} = 1$ for all i) and $d^i \in [-1, +1]$. Performing gradient descent on the instantaneous SSE criterion function $J(\mathbf{w}) = \frac{1}{2}(d - y)^2$, whose gradient is given by

$$\nabla J(\mathbf{w}) = -(d - y)\, f'(net)\mathbf{x} \qquad (2.52)$$

leads to the delta rule:

$$\begin{cases} \mathbf{w}^1 \text{ arbitrary} \\ \mathbf{w}^{k+1} = \mathbf{w}^k + \rho\left[d^k - f\left(net^k\right)\right] f'\left(net^k\right)\mathbf{x}^k = \mathbf{w}^k + \rho\delta^k\mathbf{x}^k \end{cases} \qquad (2.53)$$

where $net^k = \left(\mathbf{x}^k\right)^{\mathrm{T}} \mathbf{w}^k$ and $f' = \frac{df}{d\,net}$. If f is defined by $f(net) = \tanh(\beta\, net)$, then its derivative is given by $f'(net) = \beta\left[1 - f^2(net)\right]$. For the "logistic" function, $f(net) = 1/\left(1 + e^{-\beta\, net}\right)$, the

derivative is $f'(net) = \beta\, f(net)\left[1 - f(net)\right]$. Figure 2-8 plots f and f' for the hyperbolic tangent activation function with $\beta = 1$. Note how f asymptotically approaches +1 and –1 in the limit as *net* approaches $+\infty$ and $-\infty$, respectively.

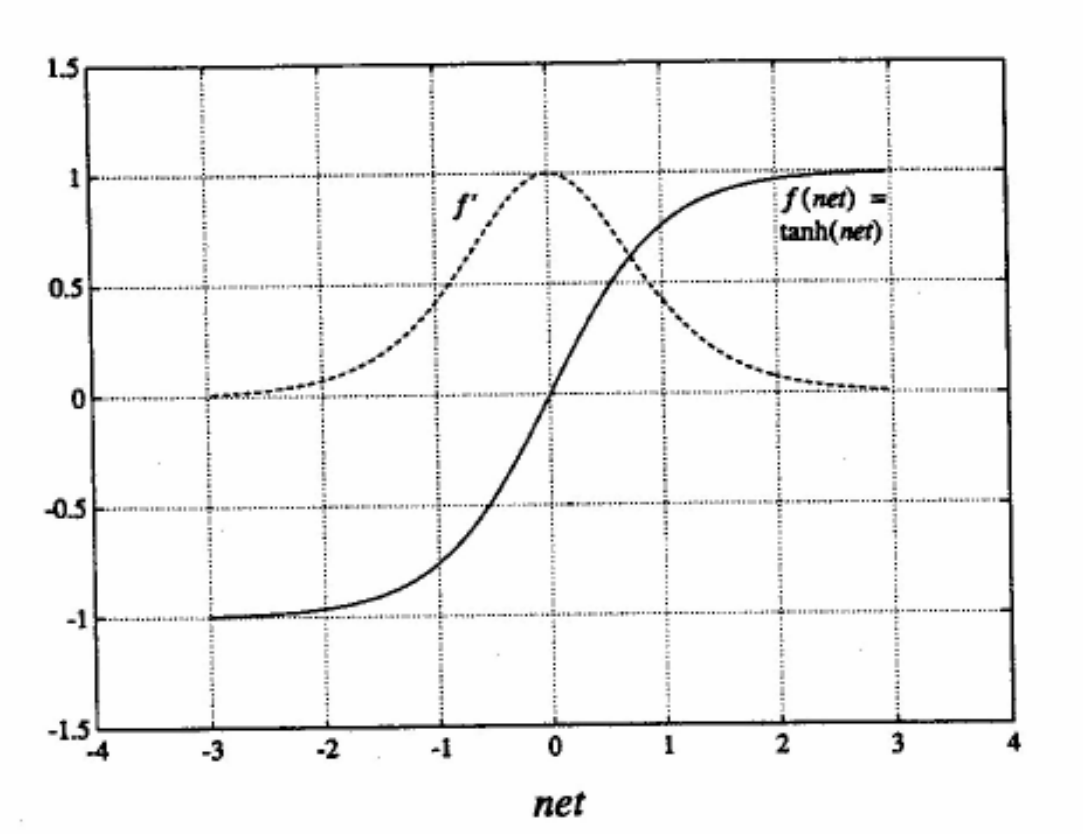

Figure 2-8 Hyperbolic tangent activation function *f* and its derivative f', plotted for $-3 \leq net \leq +3$.

One disadvantage of the delta learning rule is immediately apparent upon inspection of the graph of $f'(net)$ in Figure 2-8. In particular, notice how $f'(net) \approx 0$ when *net* has large magnitude (i.e., $|net| > 3$); these regions are called *flat spots* of f'. In these flat spots, we expect the delta learning rule to progress very slowly (i.e., very small weight changes even when the error ($d - y$) is large), because the magnitude of the weight change in Equation (2.53) directly depends on the magnitude of $f'(net)$. Since slow convergence results in excessive computation time, it would be advantageous to try to eliminate the flat spot phenomenon when using the delta learning rule. One common flat spot elimination technique involves replacing f' by f' plus a small positive bias ε. In this case, the weight update equation reads as

$$\mathbf{w}^{k+1} = \mathbf{w}^k + \rho\left[d^k - f\left(net^k\right)\right]\left[f'\left(net^k\right) + \varepsilon\right]\mathbf{x}^k \qquad (2.54)$$

One of the primary advantages of the delta rule is that it has a natural extension that may be used to train multilayered neural nets. This extension, known as *error back propagation*, will be discussed in Chapter 3.

Adaptive Ho-Kashyap (AHK) Learning Rules

Hassoun and Song (1992) proposed a set of adaptive learning rules for classification problems as enhanced alternatives to the LMS and perceptron learning rules. In the following, three learning rules, AHK I, AHK II, and AHK III, are derived based on gradient-descent strategies on an appropriate criterion function. Two of the proposed learning rules, AHK I and AHK II, are well suited for generating robust decision surfaces for linearly separable problems. The third training rule, AHK III, extends these capabilities to find "good" approximate solutions for nonlinearly separable problems. The three AHK learning rules preserve the simple incremental nature found in the LMS and perceptron learning rules. The AHK rules also possess additional processing capabilities, such

as the ability to automatically identify critical cluster boundaries and place a linear decision surface in such a way that it leads to enhanced classification robustness.

Consider a two-class $\{c_1, c_2\}$ classification problem with *m* labeled feature vectors (training vectors) $\{\mathbf{x}^i, d^i\}$, *i* = 1,2,..., *m*. Assume that $\mathbf{x}^i$ belongs to R^{n+1} (with the last component of $\mathbf{x}^i$ being a constant bias of value 1) and that $d^i = +1(-1)$ if $\mathbf{x}^i \in c_1(c_2)$. Then, a single perceptron can be trained to correctly classify the preceding training pairs if an $(n+1)$-dimensional weight vector $\mathbf{w}$ is computed that satisfies the following set of *m* inequalities (the sgn function is assumed to be the perceptron's activation function):

$$(\mathbf{x}^i)^{\mathrm{T}}\mathbf{w}\begin{cases} > 0 & \text{if } d^i = +1 \\ < 0 & \text{if } d^i = -1 \end{cases} \qquad \text{for } i = 1,2,...,m \qquad (2.55)$$

Next, if we define a set of *m* new vectors $\mathbf{z}^i$ according to

$$\mathbf{z}^i = \begin{cases} +\mathbf{x}^i & \text{if} \quad d^i = +1 \\ -\mathbf{x}^i & \text{if} \quad d^i = -1 \end{cases} \qquad \text{for } i = 1, 2, ..., m \tag{2.56}$$

and we let

$$\mathbf{Z} = \left[\mathbf{z}^1 \;\; \mathbf{z}^2 \;\; ... \;\; \mathbf{z}^m \right] \tag{2.57}$$

then Equation (2.55) may be rewritten as the single matrix equation

$$\mathbf{Z}^{\mathrm{T}}\mathbf{w} > \mathbf{0} \tag{2.58}$$

Now, defining an m-dimensional positive-valued margin vector $\mathbf{b}$ ($\mathbf{b} > \mathbf{0}$) and using it in Equation (2.58), we arrive at the following equivalent form of Equation (2.55):

$$\mathbf{Z}^{\mathrm{T}}\mathbf{w} = \mathbf{b} \tag{2.59}$$

Thus the training of the perceptron is now equivalent to solving Equation (2.59) for **w**, subject to the constraint **b** > **0**. Ho and Kashyap (1965) proposed an iterative algorithm for solving Equation (2.59). In the Ho-Kashyap algorithm, the components of the margin vector are first initialized to small positive values, and the pseu-doinverse is used to generate a solution for **w** (based on the initial guess of **b**) that minimizes the SSE criterion function $J(\mathbf{w},\mathbf{b}) = \frac{1}{2}\left\|\mathbf{Z}^{\mathrm{T}}\mathbf{w} - \mathbf{b}\right\|^2$:

$$\mathbf{w} = \mathbf{Z}^{\dagger}\mathbf{b} \qquad (2.60)$$

where $\mathbf{Z}^{\dagger} = \left(\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\right)^{-1}\mathbf{Z}$, for $m > n+1$. Next, a new estimate for the margin vector is computed by performing the constrained (**b** > **0**) gradient descent

$$\mathbf{b}^{k+1} = \mathbf{b}^{k} + \frac{1}{2}\left[\varepsilon + |\varepsilon|\right] \qquad \text{with } \varepsilon^{k} = \mathbf{Z}^{\mathrm{T}}\mathbf{w}^{k} - \mathbf{b}^{k} \qquad (2.61)$$

where $|\bullet|$ denotes the absolute value of the components of the argument vector, and $\mathbf{b}^k$ is the "current" margin vector. A new estimate of $\mathbf{w}$ can now be computed using Equation (2.60) and employing the updated margin vector from Equation (2.61). This process continues until all the components of ε are zero (or are sufficiently small and positive), which is an indication of linear separability of the training set, or until $\varepsilon < 0$, which is an indication of nonlinear separability of the training set (no solution is found). It can be shown (Ho and Kashyap, 1965) that the Ho-Kashyap procedure converges in a finite number of steps if the training set is linearly separable. For simulations comparing the preceding training algorithm with the LMS and perceptron training procedures, the reader is referred to Hassoun and Clark (1988). This algorithm will be referred to here as the *direct Ho-Kashyap (DHK) algorithm.*

The direct synthesis of the $\mathbf{w}$ estimate in Equation (2.60) involves a one-time computation of the pseudoinverse of $\mathbf{Z}$. However, such computation can be computationally expensive and requires special treatment when $\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$ is ill-conditioned (i.e., the determinant $\left|\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\right|$ close to zero). An

alternative algorithm that is based on gradient-descent principles and which does not require the direct computation of $\mathbf{Z}^{\dagger}$ can be derived. This derivation is presented next.

Starting with the criterion function $J(\mathbf{w},\mathbf{b}) = \frac{1}{2}\left\|\mathbf{Z}^{\mathrm{T}}\mathbf{w} - \mathbf{b}\right\|^2$, gradient descent may be performed with respect to **b** and **w** so that J is minimized subject to the constraint $\mathbf{b} > \mathbf{0}$. The gradient of J with respect to **w** and **b** is given by

$$\nabla_{\mathbf{b}} J(\mathbf{w},\mathbf{b})\big|_{\mathbf{w}^k,\mathbf{b}^k} = -\left(\mathbf{Z}^{\mathrm{T}}\mathbf{w}^k - \mathbf{b}^k\right) \qquad (2.62a)$$

$$\nabla_{\mathbf{w}} J(\mathbf{w},\mathbf{b})\big|_{\mathbf{w}^k,\mathbf{b}^{k+1}} = -\mathbf{Z}\left(\mathbf{Z}^{\mathrm{T}}\mathbf{w}^k - \mathbf{b}^{k+1}\right) \qquad (2.62b)$$

where the superscripts k and $k + 1$ represent current and updated values, respectively. One analytic method for imposing the constraint $\mathbf{b} > \mathbf{0}$ is to replace the gradient in Equation (2.62a) by $-0.5\left(\varepsilon + |\varepsilon|\right)$, with ε as defined in Equation (2.61). This leads to the following gradient-descent formulation of the Ho-Kashyap procedure:

$$\mathbf{b}^{k+1} = \mathbf{b}^k + \frac{\rho_1}{2}\left(\left|\varepsilon^k\right| + \varepsilon\right) \quad \text{with } \varepsilon^k = \mathbf{Z}^{\mathrm{T}}\mathbf{w}^k - \mathbf{b}^k \tag{2.63a}$$

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \rho_2\mathbf{Z}\left(\mathbf{Z}^{\mathrm{T}}\mathbf{w}^k - \mathbf{b}^{k+1}\right)$$

and

$$= \mathbf{w}^k + \frac{\rho_1\rho_2}{2}\mathbf{Z}\left[\left|\varepsilon^k\right| + \varepsilon^k\left(1 - \frac{2}{\rho_1}\right)\right] \tag{2.63b}$$

where ρ_1 and ρ_2 are strictly positive constant learning rates. Because of the requirement that all training vectors $\mathbf{z}^k$ (or $\mathbf{x}^k$) be present and included in $\mathbf{Z}$, this procedure is called the *batch-mode adaptive Ho-Kashyap (AHK) procedure.* It can be easily shown that if $\rho_1 = 0$ and $\mathbf{b}^1 = \mathbf{1}$, Equation (2.63) reduces to the μ-LMS learning rule. Furthermore, convergence can be guaranteed (Duda and Hart, 1973) if $0 < \rho_1 < 2$ and $0 < \rho_2 < 2/\lambda_{\max}$ where $\lambda_{\max}$ is the largest eigenvalue of the positive definite matrix $\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$.

A completely adaptive Ho-Kashyap procedure for solving Equation (2.59) is arrived at by starting from the instantaneous criterion function

$$J(\mathbf{w},\mathbf{b}) = \frac{1}{2}\left[\left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w} - \mathbf{b}^i\right]$$

which leads to the following incremental update rules:

$$b_i^{k+1} = b_i^k + \frac{\rho_1}{2}\left(\left|\varepsilon_i^k\right| + \varepsilon_i^k\right) \quad \text{with } \varepsilon_i^k = \left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w}^k - b_i^k \qquad (2.64a)$$

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \rho_2\mathbf{z}^i\left[\left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w}^k - b_i^{k+1}\right]$$

and

$$= \mathbf{w}^k + \frac{\rho_1\rho_2}{2}\left[\left|\varepsilon_i^k\right| + \varepsilon_i^k\left(1 - \frac{2}{\rho_1}\right)\right]\mathbf{z}^i \qquad (2.64b)$$

Here, b_i, represents a scalar margin associated with the $\mathbf{x}^i$ input. In all the preceding Ho-Kashyap learning procedures, the margin values are initialized to small positive values, and the perceptron

weights are initialized to zero (or small random) values. If full margin error correction is assumed in Equation (2.64a), i.e., $\rho_1 = 1$, the incremental learning procedure in Equation (2.64) reduces to the heuristically derived procedure reported in Hassoun and Clark (1988). An alternative way of writing Equation (2.64) is

$$\Delta b_i = \rho_1 \varepsilon_i^k \quad \text{and} \quad \Delta \mathbf{w} = \rho_2 \left(\rho_1 - 1 \right) \varepsilon_i^k \mathbf{z}^i \quad \text{if } \varepsilon_i^k > 0 \tag{2.65a}$$

$$\Delta b_i = 0 \quad \text{and} \quad \Delta \mathbf{w} = -\rho_2 \varepsilon_i^k \mathbf{z}^i \quad \text{if } \varepsilon_i^k \leq 0 \tag{2.65b}$$

where Δb and $\Delta \mathbf{w}$ signify the difference between the updated and current values of b and $\mathbf{w}$, respectively. This procedure is called the *AHK I learning rule.* For comparison purposes, it may be noted that the μ-LMS rule in Equation (2.35) can be written as $\Delta \mathbf{w} = -\mu \varepsilon_i^k \mathbf{z}^i$, with b_i, held fixed at +1.

The implied constraint $b_i > 0$ in Equations (2.64) and (2.65) was realized by starting with a positive initial margin and restricting the change Δb to positive real values. An alternative, more flexible way to realize this constraint is to allow both positive and negative changes in Δb, except for the cases where a decrease in *b*, results in a negative margin. This modification results in the following alternative AHK II learning rule:

$$\Delta b_i = \rho_1 \varepsilon_i^k \quad \text{and} \quad \Delta \mathbf{w} = \rho_2 \left(\rho_1 - 1 \right) \varepsilon_i^k \mathbf{z}^i \quad \text{if } b_i^k + \rho_1 \varepsilon_i^k > 0 \tag{2.66a}$$

$$\Delta b_i = 0 \quad \text{and} \quad \Delta \mathbf{w} = -\rho_2 \varepsilon_i^k \mathbf{z}^i \quad \text{if } b_i^k + \rho_1 \varepsilon_i^k \leq 0 \tag{2.66b}$$

In the general case of an adaptive margin, as in Equation (2.66), Hassoun and Song (1992) showed that a sufficient condition for the convergence of the AHK rules is given by

$$0 < \rho_2 < \frac{2}{\max\limits_i \left\| \mathbf{z}^i \right\|^2} \tag{2.67a}$$

$$0 < \rho_1 < 2 \qquad (2.67b)$$

Another variation results in the AHK III rule, which is appropriate for both linearly separable and nonlinearly separable problems. Here, $\Delta\mathbf{w}$ is set to 0 in Equation (2.66b). The advantages of the AHK III rule are that

(1) it is capable of adaptively identifying difficult-to-separate class boundaries and

(2) it uses such information to discard nonseparable training vectors and speed up convergence (Hassoun and Song, 1992).

Example 2.2 In this example the perceptron, LMS, and AHK learning rules are compared in terms of the quality of the solutions they generate. Consider the simple two-class linearly separable problem shown earlier in Figure 2-4. The μ-*LMS* rule of Equation (2.35) is used to obtain the solution shown as a dashed line in Figure 2-9. Here, the initial weight vector was set to 0 and a learning rate μ = 0.005 is used. This solution is not one of the linearly separable solutions for this

problem. Four examples of linearly separable solutions are shown as solid lines in the figure. These solutions are generated using the perceptron learning rule of Equation (2.2), with varying order of input vector presentations and with a learning rate of $\rho = 0.1$. Here, it should be noted that the most robust solution, in the sense of tolerance to noisy input, is given by $x_2 = x_1 + \frac{1}{2}$ which is shown as a dotted line in Figure 2-9. This robust solution was in fact automatically generated by the AHK I learning rule of Equation (2.65).

Other Criterion Functions

The SSE criterion function in Equation (2.32) is not the only possible choice. We have already seen other alternative functions such as the ones in Equations (2.20), (2.24), and (2.25). In general, any differentiable function that is minimized upon setting $y^i = d^i$, for i = 1,2,..., m, could be used. One possible generalization of SSE is the Minkowski-r criterion function (Hanson and Burr, 1988) given by

$$J(\mathbf{w}) = \frac{1}{r}\sum_{i=1}^{m}\left|d^i - y^i\right|^r \qquad (2.68)$$

or its instantaneous version

$$J(\mathbf{w}) = \frac{1}{r}\left|d^i - y^i\right|^r \qquad (2.69)$$

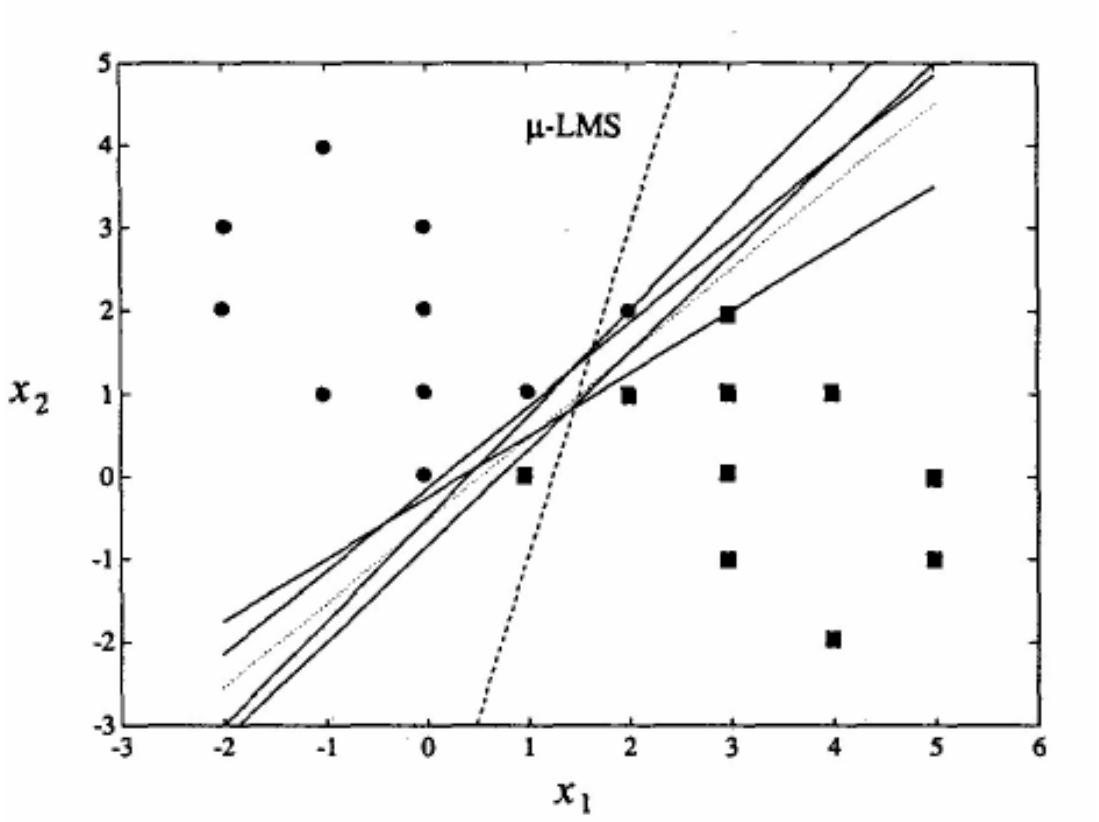

Figure 2-9 LMS-generated decision boundary (dashed line) for a two-class linearly separable problem. For comparison, four solutions generated using the perceptron learning rule are shown (solid lines). The dotted line is the solution generated by the AHK I rule.

Figure 2-10 shows a plot of $\left|d^i - y^i\right|^r$ for r = 1, 1.5, 2, and 20. The general form of the gradient of this criterion function is given by

$$\nabla J(\mathbf{w}) = -\text{sgn}(d - y)\left|d - y\right|^{r-1} f'(net)\mathbf{x} \qquad (2.70)$$

Note that for r = 2 this reduces to the gradient of the SSE criterion function given by Equation (2.52). If r = 1, then $J(\mathbf{w}) = \left|d^i - y^i\right|$ with the gradient [note that the gradient of $J(\mathbf{w})$ does not exist at the solution points $d = y$]

$$\nabla J(\mathbf{w}) = -\text{sgn}(d - y) f'(net)\mathbf{x} \qquad (2.71)$$

In this case, the criterion function in Equation (2.68) is known as the *Manhattan norm.* For $r \to \infty$, a supremum error measure is approached.

A small r gives less weight for large deviations and tends to reduce the influence of the outer-most points in the input space during learning. It can be shown, for a linear unit with normally distributed

inputs, that $r = 2$ is an appropriate choice in the sense of both minimum SSE and minimum probability of prediction error (maximum likelihood). The proof is as follows.

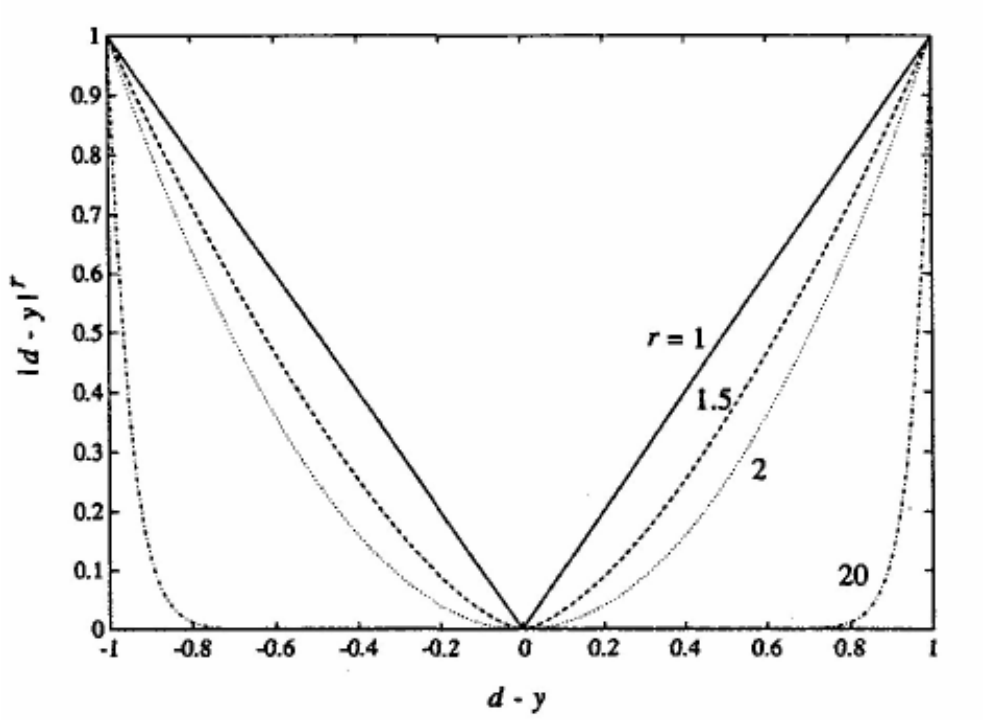

Figure 2-10 A family of instantaneous Minkowski-*r* criterion functions

Another criterion function that can be used (Hopfield, 1987) is the instantaneous relative entropy error measure (Kullback, 1959) defined by

$$J(\mathbf{w}) = \frac{1}{2}\left[(1+d)\ln\left(\frac{1+d}{1+y}\right) + (1-d)\ln\left(\frac{1-d}{1-y}\right)\right] \qquad (2.76)$$

where d belongs to the open interval $(-1,+1)$. As before, $J(\mathbf{w}) \geq 0$, and if $y = d$, then $J(\mathbf{w}) = 0$. If $y = f(net) = \tanh(\beta\, net)$, the gradient of Equation (2.76) is

$$\nabla J(\mathbf{w}) = -\beta(d-y)\mathbf{x} \qquad (2.77)$$

The factor $f'(net)$ in Equations (2.53) and (2.70) is missing from Equation (2.77). This eliminates the flat spot encountered in the delta rule and makes the training here more like μ-LMS [note, however, that y here is given by $y = f(net) \neq net$]. This entropy criterion is "well formed" in the sense that gradient descent over such a function will result in a linearly separable solution, if one

exists (Hertz et al., 1991). On the other hand, gradient descent on the SSE criterion function does not share this property, since it may fail to find a linearly separable solution, as demonstrated in Example 2.2.

In order for gradient-descent search to find a solution $\mathbf{w}^*$ in the desired linearly separable region, we need to use a well-formed criterion function. Consider the following general criterion function:

$$J(\mathbf{w}) = \sum_{i=1}^{m} g\left(\mathbf{z}^{\mathrm{T}}\mathbf{w}\right) \qquad (2.78)$$

where

$$z = \begin{cases} +\mathbf{x} & \text{if } \mathbf{x} \in \text{class } c_1 \\ -\mathbf{x} & \text{if } \mathbf{x} \in \text{class } c_2 \end{cases}$$

Let $s = \mathbf{z}^{\mathrm{T}}\mathbf{w}$. The criterion function $J(\mathbf{w})$ is said to be *well formed* if $g(s)$ is differentiable and satisfies the following conditions (Wittner and Denker, 1988):

1. For all *s*, $-\frac{dg(s)}{ds} \geq 0$; i.e., *g* does not push in the wrong direction.

2. There exists $\varepsilon > 0$ such that $-\frac{dg(s)}{ds} \geq \varepsilon$ for all $s \leq 0$; i.e., *g* keeps pushing if there is a misclassification.

3. *g(s)* is bounded from below.

For a single unit with weight vector **w**, it can be shown (Wittner and Denker, 1988) that if the criterion function is well formed, then gradient descent is guaranteed to enter the region of linearly separable solutions $\mathbf{w}^*$, provided that such a region exists.

Table 2-1 Summary of Basic Learning Rules

Learning Rule	Criterion Function[1]	Learning Vector[2]	Conditions	Activation Function[3]	Remarks
Perceptron rule (supervised)	$J(\mathbf{w}) = -\sum_{\mathbf{z}^{\mathrm{T}}\mathbf{w} \le 0} \mathbf{z}^{\mathrm{T}}\mathbf{w}$	$\begin{cases} \mathbf{z}^k & \text{if } (\mathbf{z}^k)^{\mathrm{T}}\mathbf{w}^k \le 0 \\ 0 & \text{otherwise} \end{cases}$	$\rho > 0$	$f(net) = \mathrm{sgn}(net)$	Finite convergence time if training set is linearly separable. $\|\mathbf{w}\|$ stays bounded for arbitrary training sets.
Perceptron rule with variable learning rate and fixed margin (supervised)	$J(\mathbf{w}) = -\sum_{\mathbf{z}^{\mathrm{T}}\mathbf{w} \le b} (\mathbf{z}^{\mathrm{T}}\mathbf{w} - b)$	$\begin{cases} \mathbf{z}^k & \text{if } (\mathbf{z}^k)^{\mathrm{T}}\mathbf{w}^k \le b \\ 0 & \text{otherwise} \end{cases}$	$b > 0$ ρ^k satisfies: 1. $\rho^k \ge 0$ 2. $\sum_{k=1}^{m} \rho^k = \infty$	$f(net) = \mathrm{sgn}(net)$	Converges to $\mathbf{z}^{\mathrm{T}}\mathbf{w} > b$ if training set is linearly separable. Finite convergence if $\rho^k = \rho$, where ρ is a finite positive constant.

[1] Note: $\mathbf{z}^k = \begin{cases} \mathbf{x}^k & \text{if } d^k = +1 \\ -\mathbf{x}^k & \text{if } d^k = -1 \end{cases}$

[2] The general form of the learning equation is $\mathbf{w}^{k+1} = \mathbf{w}^k + \rho^k \mathbf{s}^k$, where ρ^k is the learning rate and $\mathbf{s}^k$ is the learning vector.

[3] $net = \mathbf{x}^{\mathrm{T}}\mathbf{w}$

			3. $\frac{\sum_{k=1}^{m}\left(\rho^k\right)^2}{\left(\sum_{k=1}^{m}\rho^k\right)^2}=0$		
May`s rule (supervised)	$J(\mathbf{w})=\frac{1}{2}\sum_{\mathbf{z}^{\mathrm{T}}\mathbf{w}\le b}\frac{\left(\mathbf{z}^{\mathrm{T}}\mathbf{w}-b\right)^2}{\left\|\mathbf{z}\right\|^2}$	$\begin{cases}\frac{b-\left(\mathbf{z}^k\right)^{\mathrm{T}}\mathbf{w}^k}{\left\|\mathbf{z}\right\|^2}\mathbf{z}^k & \text{if } \left(\mathbf{z}^k\right)^{\mathrm{T}}\mathbf{w}^k\le b\\ 0 & \text{otherwise}\end{cases}$	$0<\rho<2$ $b>0$	$f(net)=\mathrm{sgn}(net)$	Finite convergence to the solution $\mathbf{z}^{\mathrm{T}}\mathbf{w}\ge b>0$ if the training set is linearly separable.
Butz`s rule (supervised)	$J(\mathbf{w})=-\sum_i\left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w}$	$\begin{cases}\mathbf{z}^k & \text{if } \left(\mathbf{z}^k\right)^{\mathrm{T}}\mathbf{w}^k\le 0\\ \gamma\mathbf{z}^k & \text{otherwise}\end{cases}$	$0\le\gamma<1$ $\rho>0$	$f(net)=\mathrm{sgn}(net)$	Finite convergence if training set is linearly separable. Places **w** in a region that tends to minimize the probability of error for nonlinearly separable cases.
Widrow-Hoff rule (α -LMS) (supervised)	$J(\mathbf{w})=\frac{1}{2}\sum_i\frac{\left[d^i-\left(\mathbf{x}^i\right)^{\mathrm{T}}\mathbf{w}\right]^2}{\left\|\mathbf{x}^i\right\|^2}$	$\left[d^k-\left(\mathbf{x}^k\right)^{\mathrm{T}}\mathbf{w}^k\right]\mathbf{x}^k$	$\rho^k=\frac{\alpha}{\left\|\mathbf{x}^k\right\|^2}$ $0<\alpha<2$	$f(net)=net$	Converges in the mean square to the minimum SSE or LMS solution if $\left\|\mathbf{x}^i\right\|=\left\|\mathbf{x}^j\right\|$ for al *i, j*.

μ-LMS (supervised)	$J(\mathbf{w}) = \frac{1}{2}\sum_i \left[d^i - \left(\mathbf{x}^i\right)^{\mathrm{T}} \mathbf{w} \right]^2$	$\left[d^k - \left(\mathbf{x}^k\right)^{\mathrm{T}} \mathbf{w}^k \right] \mathbf{x}^k$	$0 < \rho < \frac{2}{3\left\langle \left\| \mathbf{x} \right\|^2 \right\rangle}$	$f(net) = net$	Converges in the mean square to the minimum SSE or LMS solution.
Stochastic μ-LMS rule (supervised)	$J(\mathbf{w}) = \frac{1}{2}\left\langle \left[d^i - \left(\mathbf{x}^i\right)^{\mathrm{T}} \mathbf{w} \right]^2 \right\rangle$	$\left[d^k - \left(\mathbf{x}^k\right)^{\mathrm{T}} \mathbf{w}^k \right] \mathbf{x}^k$	ρ^k satisfies: 1. $\rho^k \geq 0$ 2. $\sum_{k=1}^{\infty} \rho^k = +\infty$ 3. $\sum_{k=1}^{\infty} \left(\rho^k\right)^2 < \infty$	$f(net) = net$	$\langle \bullet \rangle \equiv$ mean operator. (At each leaning step the training vector $\mathbf{x}^k$ is drawn at random). Converges in the mean square to the minimum SSE or LMS solution.
Correlation rule (supervised)	$J(\mathbf{w}) = -\sum_i d^i \left(\mathbf{x}^i\right)^{\mathrm{T}} \mathbf{w}$	$d^k \mathbf{x}^k$	$\rho > 0$	$f(net) = net$	Converges to the minimum SSE solution if the vectors $\mathbf{x}^k$ are mutually orthonormal.
Delta rule (supervised)	$J(\mathbf{w}) = \frac{1}{2}\sum_i \left(d^i - y^i\right)^2$ $y^i \triangleq \left(\mathbf{x}^i\right)^{\mathrm{T}} \mathbf{w}$	$\left(d^k - y^k\right) f'\left(net^k\right) \mathbf{x}^k$	$0 < \rho < 1$	$y = f(net)$ where f is a sigmoid function	Extends the μ-LMS rule to cases with differentiable nonlinear activations.

Learning Rule	**Criterion Function**[4]	**Learning Vector**[5]	**Conditions**	**Activation Function**[6]	**Remarks**
Minkowski-*r* delta rule (supervised)	$J(\mathbf{w}) = \frac{1}{r}\sum_i \left\lvert d^i - y^i \right\rvert^r$	$\operatorname{sgn}\left(d^k - y^k\right)\left\lvert d^k - y^k \right\rvert^{r-1} f'\left(net^k\right)\mathbf{x}^k$	$0 < \rho < 1$	$y = f(net)$ where f is a sigmoid function	$0 < r < 2$ for pseudo-Gaussian distribution $p(\mathbf{x})$ with pronounced tails. r = 2 gives delta rule. r = 1 arises when $p(\mathbf{x})$ is a Laplace distribution.

[4] Note: $\mathbf{z}^k = \begin{cases} \mathbf{x}^k & \text{if } d^k = +1 \\ -\mathbf{x}^k & \text{if } d^k = -1 \end{cases}$

[5] The general form of the learning equation is $\mathbf{w}^{k+1} = \mathbf{w}^k + \rho^k \mathbf{s}^k$, where ρ^k is the learning rate and $\mathbf{s}^k$ is the learning vector.

[6] $net = \mathbf{x}^{\mathrm{T}}\mathbf{w}$

Relative entropy delta rule (supervised)	$J(\mathbf{w}) = \frac{1}{2}\sum_i \left[\left(1+d^i\right)\ln\left(\frac{1+d^i}{1+y^i}\right)\right] + \left[\left(1-d^i\right)\ln\left(\frac{1-d^i}{1-y^i}\right)\right]$	$\beta\left(d^k - y^k\right)\mathbf{x}^k$	$0 < \rho < 1$	$y = \tanh\left(\beta\, net\right)$	Eliminates the flat spot suffered by the delta rule. Converges to one linearly separable solution if one exists.
AHK I (supervised)	$J(\mathbf{w},\mathbf{b}) = \frac{1}{2}\sum_i \left[\left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w} - \mathbf{b}_i\right]^2$	Margin $\Delta b_i = \begin{cases} \rho_1 \varepsilon_i^k & \text{if } \varepsilon_i^k > 0 \\ 0 & \text{otherwise} \end{cases}$ Weight vector: $\begin{cases} \rho_2\left(\rho_1 - 1\right)\varepsilon_i^k \mathbf{z}^i & \text{if } \varepsilon_i^2 > 0 \\ -\rho_2 \varepsilon_i^k \mathbf{z}^i & \text{if } \varepsilon_i^k \le 0 \end{cases}$ $\varepsilon_i^k = \left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w}^k - b_i^k$	$\mathbf{b}^1 > 0$ $0 < \rho_1 < 2$ $0 < \rho_2 < \frac{2}{\max_i \left\|\mathbf{z}^i\right\|^2}$	$f\left(net\right) = \mathrm{sgn}\left(net\right)$	b_i values can only increase from their initial values. Converges to a robust solution for linearly separable problems.
AHK II (supervised)	$J(\mathbf{w},\mathbf{b}) = \frac{1}{2}\sum_i \left[\left(\mathbf{z}^i\right)^{\mathrm{T}}\mathbf{w} - \mathbf{b}_i\right]^2$ with margin vector $\mathbf{b}>\mathbf{0}$	Margin Δb_i^k	$\mathbf{b}^1 > 0$ $0 < \rho_1 < 2$	$f\left(net\right) = \mathrm{sgn}\left(net\right)$	b_i values can take any positive value. Converges to a robust solution for linearly separable problems.

		$= \begin{cases} \rho_1 \varepsilon_i^k & \text{if } \varepsilon_i^k > \dfrac{-b_i^k}{\rho_1} \\ 0 & \text{otherwise} \end{cases}$ Weight vector: $\begin{cases} \rho_2 (\rho_1 - 1) \varepsilon_i^k \mathbf{z}^i & \text{if } \varepsilon_i^2 > \dfrac{-b_i^k}{\rho_1} \\ -\rho_2 \varepsilon_i^k \mathbf{z}^i & \text{if } \varepsilon_i^k \le \dfrac{-b_i^k}{\rho_1} \end{cases}$ $\varepsilon_i^k = \left(\mathbf{z}^i\right)^{\mathrm{T}} \mathbf{w}^k - b_i^k$	$0 < \rho_2 < \dfrac{2}{\max\limits_i \left\|\mathbf{z}^i\right\|^2}$		
AHK III (supervised)	$J(\mathbf{w}, \mathbf{b}) = \dfrac{1}{2} \sum\limits_i \left[\left(\mathbf{z}^i\right)^{\mathrm{T}} \mathbf{w} - \mathbf{b}_i \right]^2$ with margin vector $\mathbf{b} > \mathbf{0}$	Margin Δb_i^k $= \begin{cases} \rho_1 \varepsilon_i^k & \text{if } \varepsilon_i^k > \dfrac{-b_i^k}{\rho_1} \\ 0 & \text{otherwise} \end{cases}$ Weight vector:	$\mathbf{b}^1 > 0$ $0 < \rho_1 < 2$ $0 < \rho_2 < \dfrac{2}{\max\limits_i \left\|\mathbf{z}^i\right\|^2}$	$f(net) = \operatorname{sgn}(net)$	Converges for linearly separable as well as nonlinearly separable cases. It automatically identifies and discards the critical points affecting the nonlinear separability, and results in a solution which tends to minimize

		$\begin{cases} \rho_2(\rho_1-1)\varepsilon_i^k \mathbf{z}^i & \text{if } \varepsilon_i^2 > \dfrac{-b_i^k}{\rho_1} \\ 0 & \text{otherwise} \end{cases}$ $\varepsilon_i^k = (\mathbf{z}^i)^{\mathrm{T}} \mathbf{w}^k - b_i^k$			misclassifications.
Delta rule for stochastic units (supervised)	$J(\mathbf{w}) = \frac{1}{2}\sum_i \left(d^i - \langle y^i \rangle\right)^2$	$\beta\left[d^k - \tanh(\beta \mathrm{net}^k)\right] \cdot$ $\left[1-\tanh^2(\beta net^k)\right]\mathbf{x}^k$	$0 < \rho < 1$	Stochastic activation: $y = \begin{cases} +1 & \text{with } P(y=+1) \\ -1 & \text{with } 1-P(y=+1) \end{cases}$ $P(y=1) = \frac{1}{1+e^{-2\beta net}}$	Performance in the average is equivalent to the delta rule applied to a unit with deterministic activation: