This observation brings out one of the differences between theoretical and practical attitudes. From a theoretical viewpoint, it is interesting that we can obtain a solution in a finite number of steps for any finite set of separable samples, for any initial weight vector $\mathbf{a}(1)$, for any nonnegative margin b, and for any scale factor $\eta(k)$ satisfying Eqs. 28–30. From a practical viewpoint, we want to make wise choices for these quantities. Consider the margin b, for example. If b is much smaller than $\eta(k)\|\mathbf{y}^k\|^2$, the amount by which a correction increases $\mathbf{a}^t(k)\mathbf{y}^k$, it is clear that it will have little effect at all. If it is much larger than $\eta(k)\|\mathbf{y}^k\|^2$, many corrections will be needed to satisfy the conditions $\mathbf{a}^t(k)\mathbf{y}^k > b$. A value close to $\eta(k)\|\mathbf{y}^k\|^2$ is often a useful compromise. In addition to these choices for $\eta(k)$ and b, the scaling of the components of $\mathbf{y}^k$ can also have a great effect on the results. The possession of a convergence theorem does not remove the need for thought in applying these techniques.

A close descendant of the Perceptron algorithm is the Winnow algorithm, which has applicability to separable training data. The key difference is that while the weight vector returned by the Perceptron algorithm has components a_i $(i = 0, ...d)$, in Winnow they are scaled according to 2sinh$[a_i]$. In one version, the balanced Winnow algorithm, there are separate "positive" and "negative" weight vectors, $\mathbf{a}^+$ and $\mathbf{a}^-$, each associated with one of the two categories to be learned. Corrections on the positive weight are made if and only if a training pattern in ω_1 is misclassified; conversely, corrections on the negative weight are made if and only if a training pattern in ω_2 is misclassified.

Winnow algorithm

Algorithm 7 (Balanced Winnow)

```
begin initialize $\mathbf{a}^+, \mathbf{a}^-, \eta(\cdot), k \leftarrow 0, \alpha > 1$
          if sign$[\mathbf{a}^{+t}\mathbf{y}_k - \mathbf{a}^{-t}\mathbf{y}_k] \neq z_k$ (pattern misclassified)
          then if $z_k = +1$ then $a_i^+ \leftarrow \alpha^{+y_i} a_i^+$; $a_i^- \leftarrow \alpha^{-y_i} a_i^-$ for all $i$
               if $z_k = -1$ then $a_i^+ \leftarrow \alpha^{-y_i} a_i^+$; $a_i^- \leftarrow \alpha^{+y_i} a_i^-$ for all $i$
    return $\mathbf{a}^+, \mathbf{a}^-$
end
```

There are two main benefits of such a version of the Winnow algorithm. The first is that during training each of the two consituent weight vectors moves in a uniform direction and this means the "gap," determined by these two vectors, can never increase in size for separable data. This leads to a convergence proof that, while somewhat more complicated, is nevertheless more general than the Perceptron convergence theorem (cf. Bibliography). The second benefit is that convergence is generally faster than in a Perceptron, since for proper setting of learning rate, each constituent weight does not overshoot its final value. This benefit is especially pronounced whenever a large number of irrelevant or redundant features are present (Computer exercise 6).

5.6 Relaxation Procedures

5.6.1 The Descent Algorithm

The criterion function J_p is by no means the only function we can construct that is minimized when $\mathbf{a}$ is a solution vector. A close but distinct relative is

$$J_q(\mathbf{a}) = \sum_{\mathbf{y} \in \mathcal{Y}} (\mathbf{a}^t\mathbf{y})^2, \qquad (32)$$

where $\mathcal{Y}(\mathbf{a})$ again denotes the set of training samples misclassified by $\mathbf{a}$. Like J_p, J_q focuses attention on the misclassified samples. Its chief difference is that its gradient is continuous, whereas the gradient of J_p is not. Thus, J_q presents a smoother surface to search (Fig. 5.11). Unfortunately, J_q is so smooth near the boundary of the solution region that the sequence of weight vectors can converge to a point on the boundary. It is particularly embarrassing to spend some time following the gradient merely to reach the boundary point $\mathbf{a} = \mathbf{0}$. Another problem with J_q is that its value can be dominated by the longest sample vectors. Both of these problems are avoided by the criterion function

$$J_r(\mathbf{a}) = \frac{1}{2} \sum_{\mathbf{y} \in \mathcal{Y}} \frac{(\mathbf{a}^t\mathbf{y} - b)^2}{\|\mathbf{y}\|^2}, \tag{33}$$

where now $\mathcal{Y}(\mathbf{a})$ is the set of samples for which $\mathbf{a}^t\mathbf{y} \leq b$. (If $\mathcal{Y}(\mathbf{a})$ is empty, we define J_r to be zero.) Thus, $J_r(\mathbf{a})$ is never negative, and is zero if and only if $\mathbf{a}^t\mathbf{y} \geq b$ for all of the training samples. The gradient of J_r is given by

$$\boldsymbol{\nabla} J_r = \sum_{\mathbf{y} \in \mathcal{Y}} \frac{\mathbf{a}^t\mathbf{y} - b}{\|\mathbf{y}\|^2}\mathbf{y},$$

and the update rule

$$\left.\begin{array}{ll} \mathbf{a}(1) & \text{arbitrary} \\ \mathbf{a}(k+1) = \mathbf{a}(k) + \eta(k) \sum\limits_{\mathbf{y} \in \mathcal{Y}} \frac{b - \mathbf{a}^t\mathbf{y}}{\|\mathbf{y}\|^2}\mathbf{y}. \end{array}\right\} \tag{34}$$

Thus the relaxation algorithm becomes

Algorithm 8 (Batch relaxation with margin)

```
begin initialize $\mathbf{a}, \eta(\cdot), k = 0$
    do $k \leftarrow k + 1$
        $\mathcal{Y}_k = \{\}$
        $j = 0$
        do $j \leftarrow j + 1$
            if $\mathbf{y}_j$ is misclassified then Append $\mathbf{y}_j$ to $\mathcal{Y}_k$
        until $j = n$
        $\mathbf{a} \leftarrow \mathbf{a} + \eta(k) \sum_{\mathbf{y} \in \mathcal{Y}} \frac{b - \mathbf{a}^t\mathbf{y}}{\|\mathbf{y}\|^2}\mathbf{y}$
    until $\mathcal{Y}_k = \{\}$
return $\mathbf{a}$
end
```

As before, we find it easier to prove convergence when the samples are considered one at a time rather than jointly, i.e., single-sample rather than batch. We also limit our attention to the fixed-increment case, $\eta(k) = \eta$. Thus, we are again led to consider a sequence $\mathbf{y}^1, \mathbf{y}^2, ...$ formed from those samples that call for the weight vector to be corrected. The single-sample correction rule analogous to Eq. 33 is

$$\left.\begin{array}{ll} \mathbf{a}(1) & \text{arbitrary} \\ \mathbf{a}(k+1) = \mathbf{a}(k) + \eta \frac{b - \mathbf{a}^t(k)\mathbf{y}^k}{\|\mathbf{y}^k\|^2}\mathbf{y}^k, \end{array}\right\} \tag{35}$$

where $\mathbf{a}^t(k)\mathbf{y}^k \leq b$ for all k. The algorithm is:

Algorithm 9 (Single-sample relaxation with margin)

```
begin initialize $\mathbf{a}, \eta(\cdot), k = 0$
          do $k \leftarrow k + 1$
              if $\mathbf{y}_k$ is misclassified then $\mathbf{a} \leftarrow \mathbf{a} + \eta(k)\frac{b-\mathbf{a}^t\mathbf{y}}{\|\mathbf{y}_k\|^2}\mathbf{y}_k$
                 until all patterns properly classified
   return $\mathbf{a}$
end
```

This algorithm is known as the *single-sample relaxation rule with margin*, and it has a simple geometrical interpretation. The quantity

$$r(k) = \frac{b - \mathbf{a}^t(k)\mathbf{y}^k}{\|\mathbf{y}^k\|} \tag{36}$$

is the distance from $\mathbf{a}(k)$ to the hyperplane $\mathbf{a}^t\mathbf{y}^k = b$. Since $\mathbf{y}^k/\|\mathbf{y}^k\|$ is the unit normal vector for the hyperplane, Eq. 35 calls for $\mathbf{a}(k)$ to be moved a certain fraction η of the distance from $\mathbf{a}(k)$ to the hyperplane. If $\eta = 1$, $\mathbf{a}(k)$ is moved exactly to the hyperplane, so that the "tension" created by the inequality $\mathbf{a}^t(k)\mathbf{y}^k \leq b$ is "relaxed" (Fig. 5.14). From Eq. 35, after a correction,

$$\mathbf{a}^t(k+1)\mathbf{y}^k - b = (1-\eta)(\mathbf{a}^t(k)\mathbf{y}^k - b). \tag{37}$$

If $\eta < 1$, then $\mathbf{a}^t(k+1)\mathbf{y}^k$ is still less than b, while if $\eta > 1$, then $\mathbf{a}^t(k+1)\mathbf{y}^k$ is greater than b. These conditions are referred to as *underrelaxation* and *overrelaxation*, respectively. In general, we shall restrict η to the range $0 < \eta < 2$ (Figs. 5.14 & 5.15).

UNDER-RELAXATION

OVER-RELAXATION

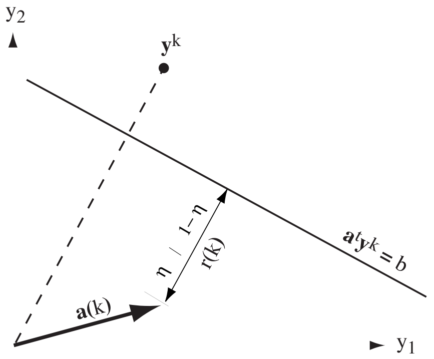

Figure 5.14: In each step of a basic relaxation algorithm, the weight vector is moved a proportion η of the way towards the hyperplane defined by $\mathbf{a}^t\mathbf{y}^k = b$.

5.6.2 Convergence Proof

When the relaxation rule is applied to a set of linearly separable samples, the number of corrections may or may not be finite. If it is finite, then of course we have obtained a solution vector. If it is not finite, we shall see that $\mathbf{a}(k)$ converges to a limit vector on the boundary of the solution region. Since the region in which $\mathbf{a}^t\mathbf{y} \geq b$ is contained in a larger region where $\mathbf{a}^t\mathbf{y} > 0$ if $b > 0$, this implies that $\mathbf{a}(k)$ will enter this larger region at least once, eventually remaining there for all k greater than some finite k_0.

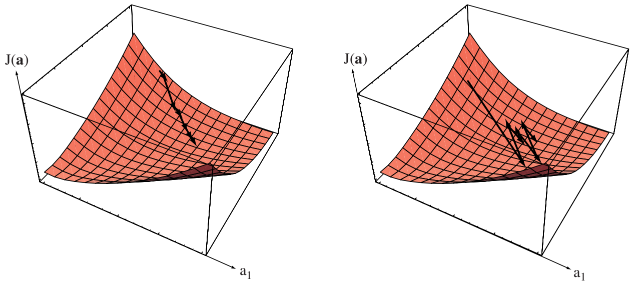

Figure 5.15: At the left, underrelaxation ($\eta < 1$) leads to needlessly slow descent, or even failure to converge. Overrelaxation ($1 < \eta < 2$, shown in the middle) describes overshooting; nevertheless convergence will ultimately be achieved.

The proof depends upon the fact that if $\hat{\mathbf{a}}$ is *any* vector in the solution region — i.e., any vector satisfying $\hat{\mathbf{a}}^t\mathbf{y}_i > b$ for all i — then at each step $\mathbf{a}(k)$ gets closer to $\hat{\mathbf{a}}$. This fact follows at once from Eq. 35, since

$$\begin{aligned}\|\mathbf{a}(k+1)-\hat{\mathbf{a}}\|^2 &= \|\mathbf{a}(k)-\hat{\mathbf{a}}\|^2 - 2\eta\frac{(b-\mathbf{a}^t(k)\mathbf{y}^k)}{\|\mathbf{y}^k\|^2}(\hat{\mathbf{a}}-\mathbf{a}(k))^t\mathbf{y}^k \\ &\quad +\eta^2\frac{(b-\mathbf{a}^t(k)\mathbf{y}^k)^2}{\|\mathbf{y}^k\|^2} \qquad (38)\end{aligned}$$

and

$$(\hat{\mathbf{a}}-\mathbf{a}(k))^t\mathbf{y}^k > b-\mathbf{a}^t(k)\mathbf{y}^k \geq 0, \qquad (39)$$

so that

$$\|\mathbf{a}(k+1)-\hat{\mathbf{a}}\|^2 \leq \|\mathbf{a}(k)-\hat{\mathbf{a}}\|^2 - \eta(2-\eta)\frac{(b-\mathbf{a}^t(k)\mathbf{y}^k)^2}{\|\mathbf{y}^k\|^2}. \qquad (40)$$

Since we restrict η to the range $0 < \eta < 2$, it follows that $\|\mathbf{a}(k+1)-\hat{\mathbf{a}}\| \leq \|\mathbf{a}(k)-\hat{\mathbf{a}}\|$. Thus, the vectors in the sequence $\mathbf{a}(1), \mathbf{a}(2), ...$ get closer and closer to $\hat{\mathbf{a}}$, and in the limit as k goes to infinity the distance $\|\mathbf{a}(k)-\hat{\mathbf{a}}\|$ approaches some limiting distance $r(\hat{\mathbf{a}})$. This means that as k goes to infinity $\mathbf{a}(k)$ is confined to the surface of a hypersphere with center $\hat{\mathbf{a}}$ and radius $r(\hat{\mathbf{a}})$. Since this is true for any $\hat{\mathbf{a}}$ in the solution region, the limiting $\mathbf{a}(k)$ is confined to the intersection of the hyperspheres centered about all of the possible solution vectors.

We now show that the common intersection of these hyperspheres is a single point on the boundary of the solution region. Suppose first that there are at least two points $\mathbf{a}'$ and $\mathbf{a}''$ on the common intersection. Then $\|\mathbf{a}'-\hat{\mathbf{a}}\| = \|\mathbf{a}''-\hat{\mathbf{a}}\|$ for every $\hat{\mathbf{a}}$ in the solution region. But this implies that the solution region is contained in the $(\hat{d}-1)$-dimensional hyperplane of points equidistant from $\mathbf{a}'$ to $\mathbf{a}''$, whereas we know that the solution region is $\hat{d}$-dimensional. (Stated formally, if $\hat{\mathbf{a}}^t\mathbf{y}_i > 0$ for $i = 1, ..., n$, then for any $\hat{d}$-dimensional vector $\mathbf{v}$, we have $(\hat{\mathbf{a}}+\epsilon\mathbf{v})^t\mathbf{y} > 0$ for $i = 1, ..., n$ if ϵ is sufficiently small.) Thus, $\mathbf{a}(k)$ converges to a single point $\mathbf{a}$. This point is certainly not inside the solution region, for then the sequence would be finite. It is not outside either, since each correction causes the weight vector to move η times its distance from the boundary plane, thereby preventing the vector from being bounded away from the boundary forever. Hence the limit point must be on the boundary.

5.7 Nonseparable Behavior

The Perceptron and relaxation procedures give us a number of simple methods for finding a separating vector when the samples are linearly separable. All of these methods are called *error-correcting procedures*, because they call for a modification of the weight vector when and only when an error is encountered. Their success on separable problems is largely due to this relentless search for an error-free solution. In practice, one would only consider the use of these methods if there was reason to believe that the error rate for the optimal linear discriminant function is low.

ERROR-CORRECTING PROCEDURE

Of course, even if a separating vector is found for the training samples, it does not follow that the resulting classifier will perform well on independent test data. A moment's reflection will show that *any* set of fewer than $2\hat{d}$ samples is likely to be linearly separable — a matter we shall return to in Chap. **??**. Thus, one should use several times that many design samples to overdetermine the classifier, thereby ensuring that the performance on training and test data will be similar. Unfortunately, sufficiently large design sets are almost certainly *not* linearly separable. This makes it important to know how the error-correction procedures will behave when the samples are nonseparable.

Since no weight vector can correctly classify every sample in a nonseparable set (by definition), it is clear that the corrections in an error-correction procedure can never cease. Each algorithm produces an infinite sequence of weight vectors, any member of which may or may not yield a useful "solution." The exact nonseparable behavior of these rules has been studied thoroughly in a few special cases. It is known, for example, that the length of the weight vectors produced by the fixed-increment rule are bounded. Empirical rules for terminating the correction procedure are often based on this tendency for the length of the weight vector to fluctuate near some limiting value. From a theoretical viewpoint, if the components of the samples are integer-valued, the fixed-increment procedure yields a finite-state process. If the correction process is terminated at some arbitrary point, the weight vector may or may not be in a good state. By averaging the weight vectors produced by the correction rule, one can reduce the risk of obtaining a bad solution by accidentally choosing an unfortunate termination time.

A number of similar heuristic modifications to the error-correction rules have been suggested and studied empirically. The goal of these modifications is to obtain acceptable performance on nonseparable problems while preserving the ability to find a separating vector on separable problems. A common suggestion is the use of a variable increment $\eta(k)$, with $\eta(k)$ approaching zero as k approaches infinity. The rate at which $\eta(k)$ approaches zero is quite important. If it is too slow, the results will still be sensitive to those training samples that render the set nonseparable. If it is too fast, the weight vector may converge prematurely with less than optimal results. One way to choose $\eta(k)$ is to make it a function of recent performance, decreasing it as performance improves. Another way is to program $\eta(k)$ by a choice such as $\eta(k) = \eta(1)/k$. When we examine stochastic approximation techniques, we shall see that this latter choice is the theoretical solution to an analogous problem. Before we