Chapter 5:
Linear Discriminant Functions
(Sections 5.5-5.6)

q Criterion Functions

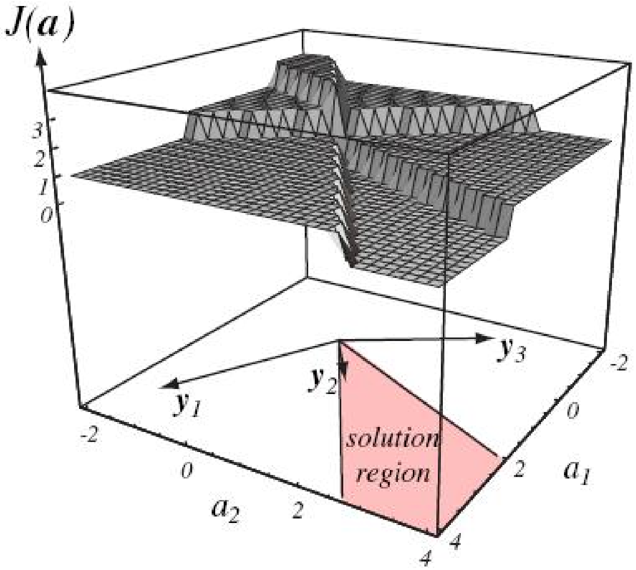
q Relaxation Procedures

Obvious choice

q The number of samples misclassified by **a**

$$J(a; y_1, \mathbf{L}\ y_n)$$

q This function is piecewise constant so not good for gradient descent



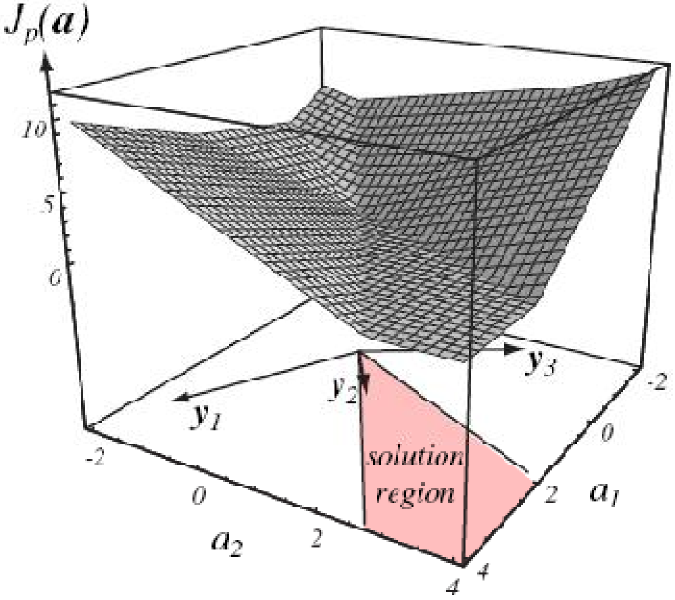
Perceptron Criterion Function

q The sum of misclassified sample functions

$$J_p(a) = \sum_{y \in Y} (-a^t y)$$

q This function is never negative ($a^t y \leq 0$)

q It is proportional to the sum of the distances from the misclassified samples to the decision boundary $r = \frac{g(x)}{\|a\|}$



Batch Perceptron Algorithm

- Begin initialize a, θ, k=0,β(0)
- Do k=k+1
- Find y`s belong to Y ($a^t y \leq 0$)
- $a = a + b(k)\sum_{y \in Y} y$
- Until Y={} or $\left| b(k)\sum_{y \in Y} y \right| < q$
- Return a
- end

Batch refers to the fact that a group of samples is used when computing each weight update

Example

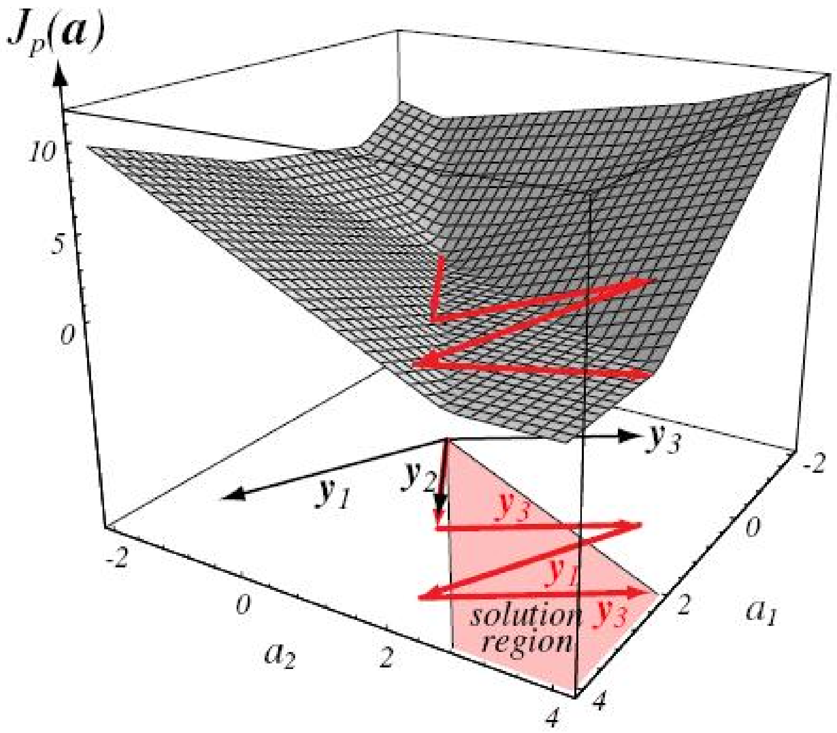
A simple two-dimensional example with a(0)=0 and β(k)=1

k=1: $\mathbf{a} = \mathbf{y}_1 + \mathbf{y}_2 + \mathbf{y}_3$

k=2: $\mathbf{a} = \mathbf{a} + \mathbf{y}_3$

k=3: $\mathbf{a} = \mathbf{a} + \mathbf{y}_1$

k=4: $\mathbf{a} = \mathbf{a} + \mathbf{y}_3$



Note that the misclassified

samples satisfy $a^t y \leq 0$

SINGLE-SAMPLE vs. BATCH

- **Design alternatives:**
  - **Single-sample mode:** weights are updated after each training sample.
  - **Batch mode:** weights updated after seeing all samples, at the end of a complete training pass.
- **Single-sample mode** (also known as **case update**):

  Weights are updated after each training sample:

  $$w \leftarrow w - \beta(p)\,\nabla J(w)$$

  - **Advantage:** faster convergence (at least, at the beginning of training).
  - **Disadvantages:**
    - Sensitive to noise (i.e. isolated out-of-boundary training samples).
    - Tendency to oscillate in the vicinity of the minimum.
- **Batch mode** (also known as **epoch update**):

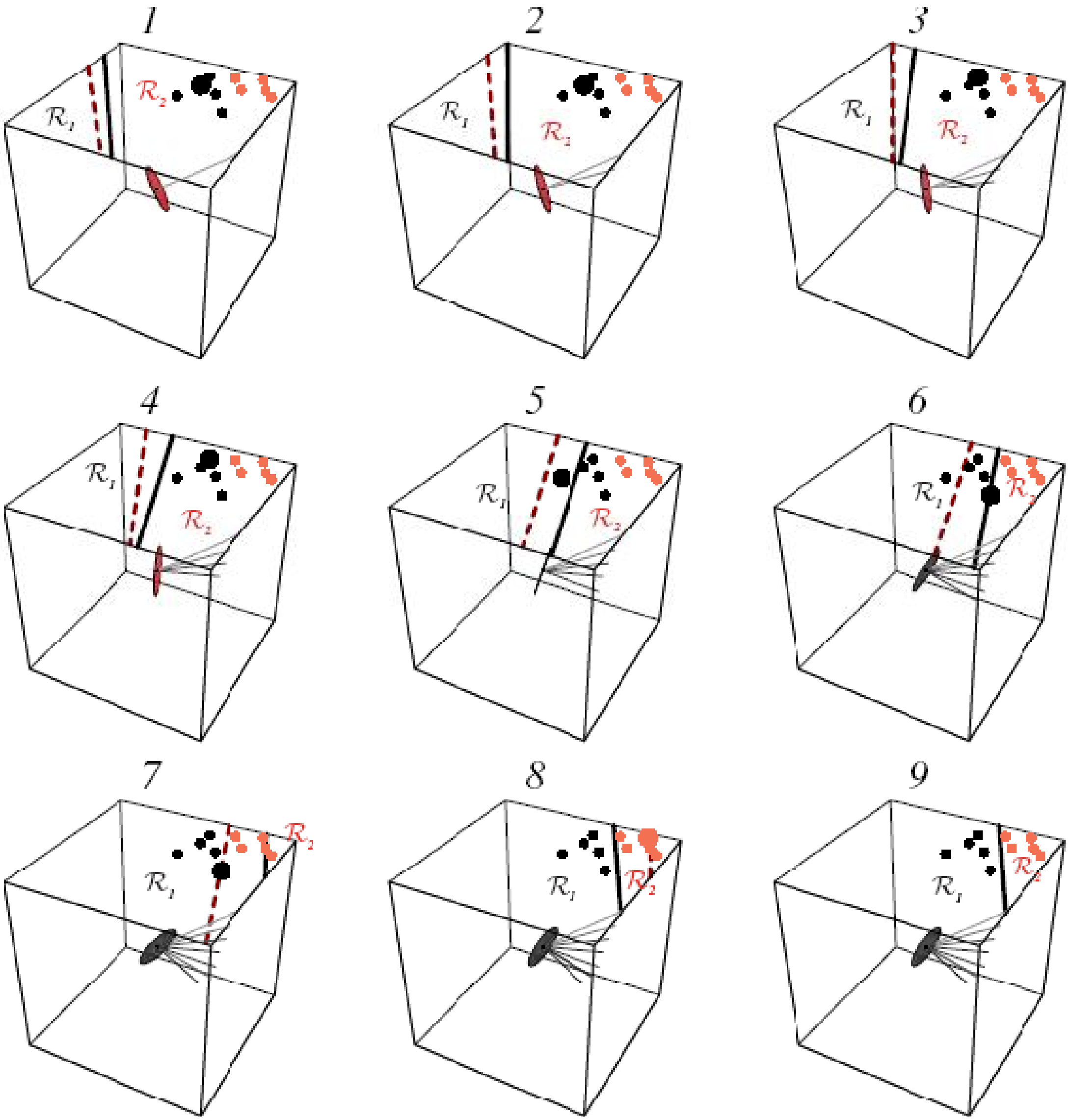
  Throughout a training pass *p*, the errors corresponding to each sample *k*, are accumulated:

  $$\nabla J_p(w) = \Sigma \nabla J_k(w)$$

  At the end of the pass, all the weights are updated at once (based on the cumulative error):

  $$w \leftarrow w - \beta(p)\,\nabla J_p(w)$$

Fixed-Increment Single-Sample Perceptron



a(1): arbitary

$a(k+1)=a(k)+y^k$: $k \geq 1$

Example: two categories

augmented weight

(y=[1,x])

Perceptron Criterion

- If training samples are linearly separable, then the perceptron algorithm (batch or single) achieves the solution (can be shown!)

- It is not as fast as it should be!

- Searching another criterions

Relaxation + Margin

- Perceptron Criterion Function

$$J_p(a) = \sum_{y \in Y} (-a^t y)$$

- Another Criterion:

$$J_q(a) = \sum_{y \in Y} (a^t y)^2$$

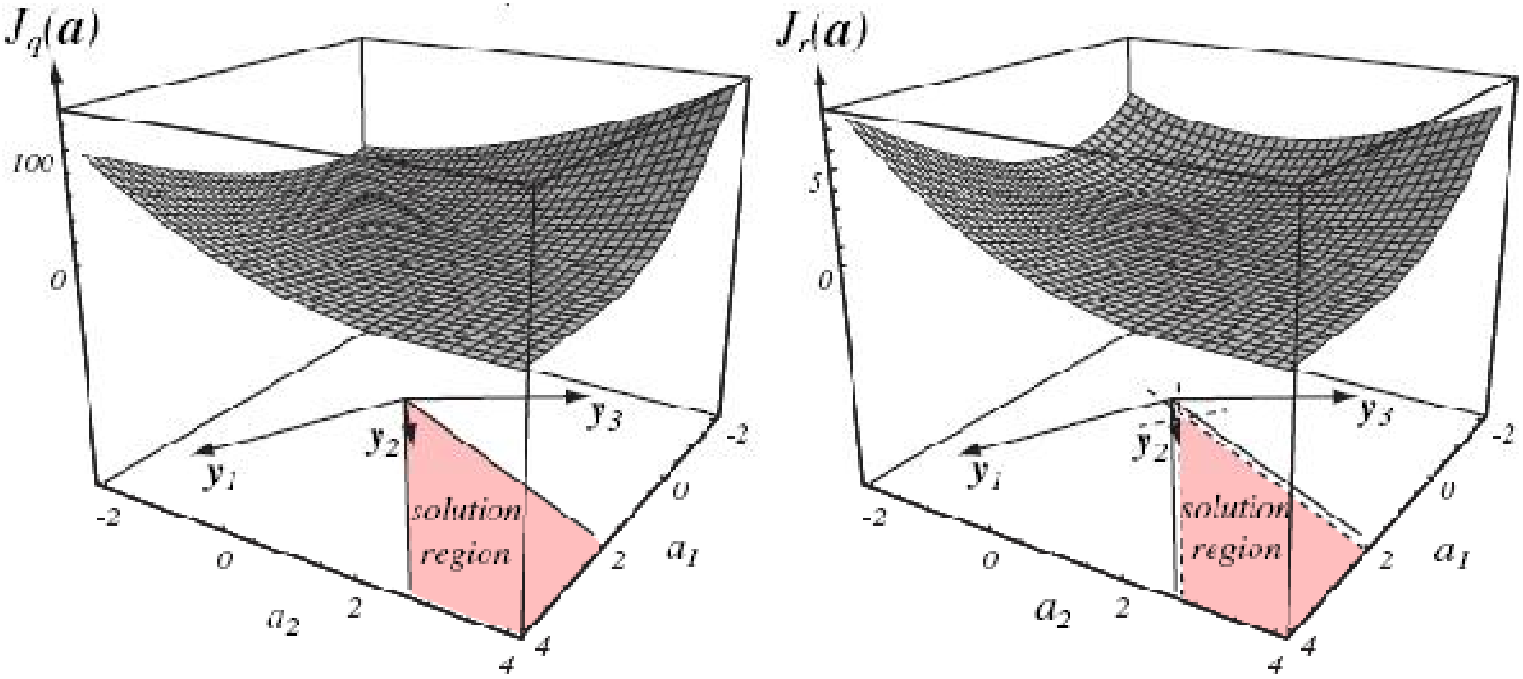
Gradient of J_q is continuous

But! Convergence to the boundary, J_q is dominated by longest sample vectors

- Solution:

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

Y is the set of samples for which $a^t y \leq b$ (misclassified)



$J_q(a)$
100
0
y_1
y_2
y_3
solution region
-2
0
a_2
2
4
4
2
a_1
0
-2
$J_r(a)$
5
0
y_1
y_2
y_3
solution region
-2
0
a_2
2
4
4
2
a_1
0
-2

So $$\nabla J_r = \sum_{y \in Y} \frac{a^t y - b}{\|y\|^2} y$$

Batch relaxation + margin

- Begin initialize a, b, k=0,β(0)
- Do k=k+1
- Find y`s belong to Y ($a^t y \leq b$)
- $$a \longleftarrow a + b(k) \sum_{y \in Y} \frac{b - a^t y}{\|y\|^2} y$$
- Until Y={}
- Return a
- end

Single-Sample relaxation + margin

- Begin initialize a, b, k=0,β(0)
- Do k=k+1
- If $a^t y^k \leq b$ then $a \longleftarrow a + b(k)\frac{b - a^t y^k}{\left\|y^k\right\|^2} y^k$
- Until $a^t y^k > b$ for all y^k
- Return a
- end

Single-Sample relaxation + margin

<u>Geometrical interpretation</u>

a(k) is moved a certain fraction (β) of the distance from a(k) to the hyperplane $a^t y^k = b$.

If β=1, a(k) is moved exactly to the hyperplane (or relaxed).

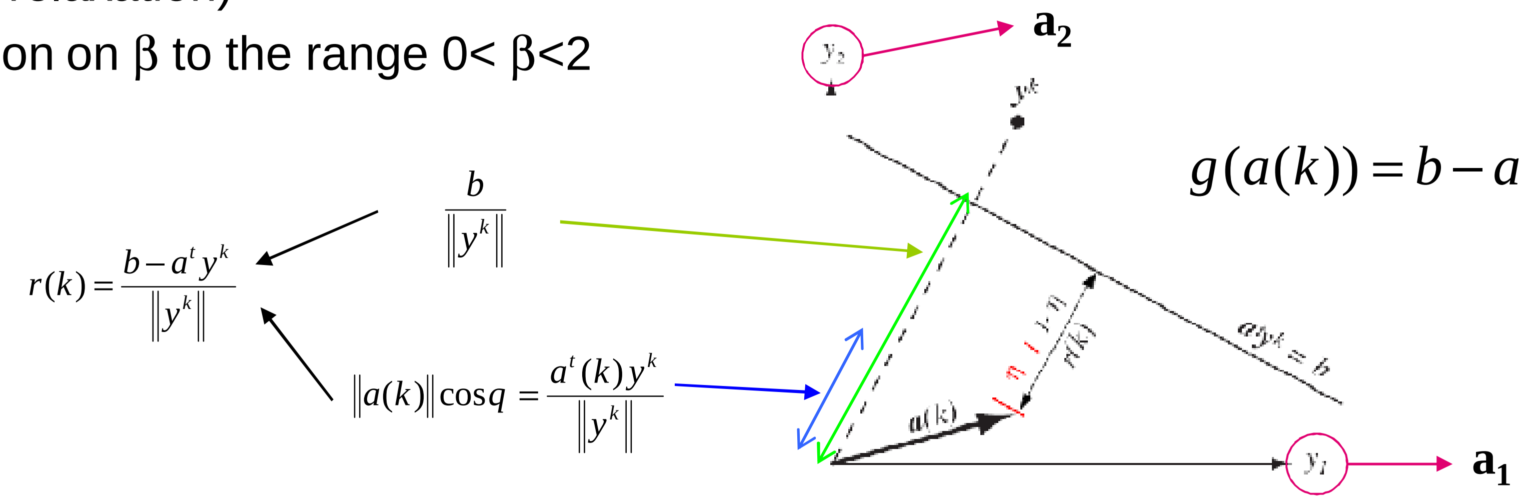
If β<1, $a^t(k+1)y^k$ is still less than b (underrelaxation)

If β>1, $a^t(k+1)y^k$ is greater than b (overrelaxation)

Restriction on β to the range 0< β<2

$$a(k+1) = a(k) + b\,\frac{b - a^t(k)y^k}{\left\|y^k\right\|^2}\,y^k$$

$$a^t(k+1)y^k - b = (1-b)(a^t(k)y^k - b)$$



$$g(a(k)) = b - a^t y^k$$

$$\frac{b}{\left\|y^k\right\|}$$

$$r(k) = \frac{b - a^t y^k}{\left\|y\right\|}$$

$$\left\|a(k)\right\|\cos q = \frac{a^t(k)y^k}{\left\|y^k\right\|}$$

FIGURE 5.14. In each step of a basic relaxation algorithm, the weight vector is moved a proportion η of the way toward the hyperplane defined by $a^t y^k = b$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

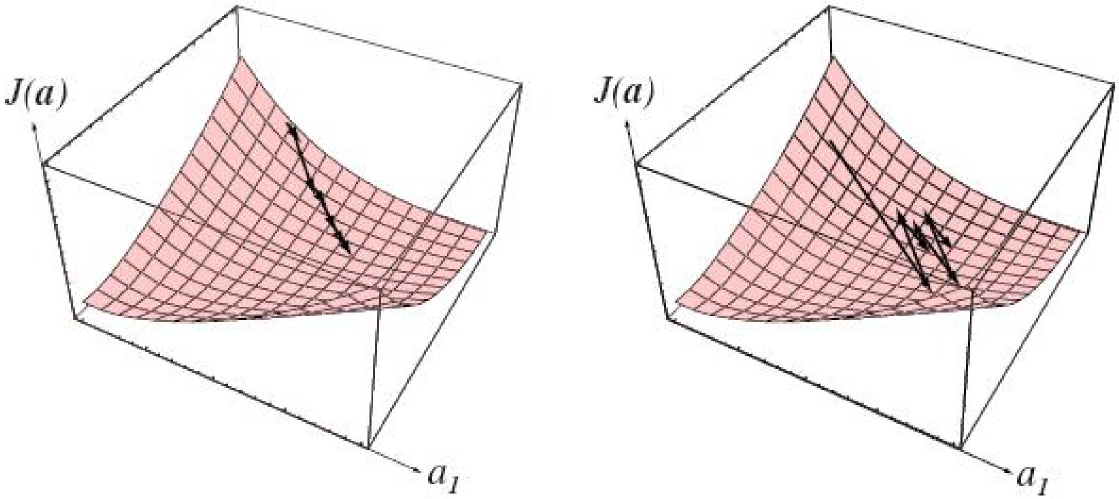


FIGURE 5.15. At the left, underrelaxation ($\eta < 1$) leads to needlessly slow descent, or even failure to converge. Overrelaxation ($1 < \eta < 2$, shown at the right) describes overshooting; nevertheless, convergence will ultimately be achieved. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.