The Delta Rule

The Delta rule (also known as the LMS rule, for least mean squares) rule has many similarities to the Perceptron Learning Rule.

In its simplest form, linear units are used

$$Y = \sum_{i=0}^{n} w_i I_i$$

where I_n is the bias input.

Here Y may have any value (the unit is linear, not a thresholding unit).

The learning rule *looks* identical:

$$\Delta w_i = \eta(D - Y) I_i$$

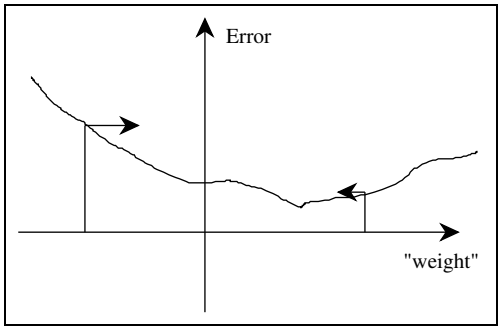
but now, (D-Y) is not restricted to having values of 0, 1 or -1, but may have any value whatever.

As we will see, the value of η (eta) is critical: and this is certainly different from the case of the perceptron rule.

Gradient Descent

Many techniques in neural networks (and elsewhere) rely on trying to minimise some quantity. In this case, we are interested in minimising the error.

If the error is a smooth function of some variable parameters (here, the weights), then one can move around in weight-space so as to reduce the error



1-dimensional representation.

If we move in the direction of decreasing gradient, we move towards a lower value of the function. (but if we move too far, we may move back up again).

So:

1: find direction of decrease of function (error)
2: move weights a small distance in this direction.

Gradient Descent and the Delta Rule

We can use gradient descent iteratively to find the minimum of anycontinuously differentiable function.

We start at a first estimate, x_0.

We produce a new estimate by moving in the direction in which the function is decreasing

$$x_1 = x_0 + \eta(-\frac{df}{dx})$$

for η small enough, one can be sure that

$$f(x_1) \leq f(x_0)$$

with strict inequality if df/dx ≠ 0.

We keep doing this until df/fx = 0. One is then at a minimum of f(x).

Notes:

We have illustrated this for the simplest possible case, namely a 1-dimensional weight space.

In a high dimensional space, there may be very many directions which are all leading "down" in the error space.

In a high dimensional space (and for a general graph) there may be many minima, and they may be close together.

Most of these will be **local** minima, rather than the **global** minimum.

Proving the Delta rule.

To prove the learning algorithm for the Delta rule, we show that it implements gradient descent in a suitable measure of the error, E.

We would like to choose a measure of the error such that

E=0 if and only if all the outputs are precisely correct.
E>0 if any output is not precisely correct

We can then think of the learning rule as an E-minimisation problem.

We use the mean square error

$$E = \frac{1}{2}\sum_j (D_j - Y_j)^2$$

where j indexes the patterns

We can think about the shape of the error surface: that is how the error varies as the weights vary.

$$E = \frac{1}{2}\sum_j (D_j - \sum_i w_i I_i^j)^2$$

that is, the error surface is quadratic in each w_i, with the w_i^2 terms all having positive coefficient.

As a result, the error surface will be a hyper-paraboloid: i.e. bowl-shaped, with only 1 minimum.

Consider the partial derivative of E with respect to w_i:

$$\frac{\partial E}{\partial w_i} = \sum_j (D_j - Y_j)(-\frac{\partial Y}{\partial w_i})$$

so that

$$\frac{\partial E}{\partial w_i} = -\sum_j (D_j - Y_j) I_i$$

This means that our weight change rule

$$\Delta w_i = \eta (D - Y).I_i$$

always moves in the opposite direction from the gradient.

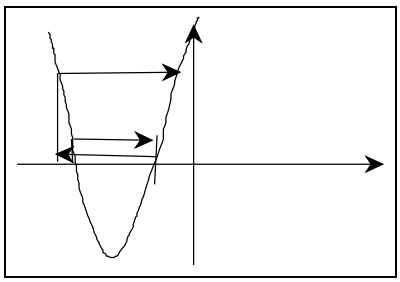
That is, it implements gradient descent.

Actually, it means that we should accumulate the Δw_i's over a complete set of patterns, and then apply them. This is known as **batch** update.

However, it is common to apply the weight changes pattern by pattern as soon as they have been computed: this is known as **on-line** update.

Notes:

i: The size of η is critical:
Too small, and convergence will be very slow.
Too large, and the system may diverge.



ii: The rule can be extended to units which have a nonlinear output function, so long as (i) it is strictly increasing and (ii) it is differentiable.

Say

$$Y = f(\sum w_i I_i)$$

then, with E as before,

$$\frac{\partial E}{\partial w_i} = \sum_j (D_j - Y_j)(-\frac{\partial f}{\partial w_i})$$

,so that

$$\frac{\partial E}{\partial w_i} = -\sum_j (D_j - Y_j) f'(\sum w_i I_i^j) I_i$$

Frequently used output functions are

$$f(x) = \tanh(\beta x)$$

for which the derivative is

$$f'(x) = \beta(1 - f(x)^2)$$

or the **logistic** function,

$$f(x) = \frac{1}{(1 + e^{-\beta x})}$$

which has derivative

$$f'(x) = 2\beta f(x)(1 - f(x))$$

Many others are also usable.

Note that the E-surface remains bowl-shaped so long as f′ (x) exists and is strictly positive.

Note also that E will frequently decrease to some minimum value, but that this may well not be zero.