How does the LMS algorithm estimate the gradient?

Paul Cuff

February 5, 2005

The LMS (adaptive filter) algorithm estimates the gradient of the performance curve from a single data sample. A simple proof of the derivation is given in the textbook [1] on page 100. I would like to make a comparison between LSM and a different approach to updating weights that yields an almost identical algorithm. The intention is to provide insight into what LMS is doing when it estimates the gradient.

It is important to realize that the gradient "estimate" of the performance curve is in fact a gradient *measurement* of the performance curve for the specific data point at hand. In other words, the true performance curve, based off of the statistics of the data and input, can be shown to be a paraboloid with respect to the weights, and may have a minimum value greater than zero. The instantaneous performance curve, as I will call it, based off of the current sample of data and input, is a trough with zero squared-error in a hyperplane of L (one less than the number of weights) dimensions. The LMS algorithm is in fact measuring the gradient of the instantaneous performance curve.

So what does this mean, and where will this gradient estimate be pointing? The following explanation has given me a little bit of insight. Suppose we try a different strategy for updating the weights. We will find a set of weights, w^*, that will bring the error to zero for a particular sample of input and data.

$$d - w^{*T}x = d - x^T w^* = 0$$

$$d = x^T w^*$$

Since x is a column vector of presumably more than one element, the system described above is grossly underdetermined. There is an L dimensional set of solutions for w^*. However, since we already have a previous set of weights, w, it would be desirable to find the solution for w^* closest to w. This is done in the following way:

$$w^* = w + \overline{w}$$

$$d - x^T w^* = d - x^T(w + \overline{w}) = 0$$

$$d - x^T w = x^T \overline{w}$$

We wish to find the minimum-norm solution for $\overline{w}$. To simplify matter let's substitute the following:

$$d - x^T w = d - w^T x = d - \hat{d} = e$$

$$e = x^T \overline{w}$$

Minimum-norm:

$$\overline{w} = x(x^T x)^{-1} e = \frac{e}{||x||^2} x$$

Notice the similarity between the correction to the weights as suggested by this method and the one used in the LMS algorithm:

LMS:	$w_{k+1} = w_k + 2\mu e_k x_k$
Least-norm:	$\overline{w} = \frac{e}{||x||^2} x$

By letting $\mu = \frac{1}{2||x||^2}$ the two methods become the same, though that's not quite legal because $||x||^2$ is a random variable. It is not surprising that the corrections to the weights in both these methods point in the same direction because indeed the gradient of a trough always points away from closest minimum point.

If the weights are adjusted according to the minimum-norm solution for zero instantaneous error, the adjustment would be as follows:

$$\begin{aligned} w_{k+1} &= w_k^* = w_k + \overline{w}_k = w_k + x_k(x_k^T x_k)^{-1} e_k \\ &= x_k(x_k^T x_k)^{-1}(d_k - x_k^T w_k) \\ &= (1 - x_k(x_k^T x_k)^{-1} x_k^T) w_k + \frac{d_k}{x_k^T x_k} x_k \end{aligned}$$

The next weights consist of the components of the current weights not in the x direction and a vector in the x direction that depends only on the current inputs and data. Therefore, in the x direction there is no dependence on the previous weights; there is no averaging in that sense.

There are a few interesting things to consider about these two similar approaches to updating the weights. One question I'm tempted to ask is how well the minimum-norm solution method (i.e. μ is itself a random variable that depends on the input as $\frac{1}{2||x||^2}$) will perform in comparison to LMS. This analysis appears complicated to me, but the method seems to have an obvious downside. On the occasion that $||x||$ is small this method will overreact to noise in the data.

On the other hand, consider letting $\mu = \frac{1}{2E[||x||^2]}$. In this case,

$$E[||x||^2] = \sum_i E[x_i^2] = \sum_i R_{ii} = \sum_i \lambda_i$$

$$\mu = \frac{1}{2\sum_i \lambda_i}$$

where λ_i is the i^{th} eigenvalue of R. This is interesting because the result is a reasonable suggestion for the value of μ. Remember that μ must be less than $\frac{1}{\lambda_{max}}$ in order for the filter to converge to the "bottom of the bowl." Notice that,

$$\mu = \frac{1}{2\sum_i \lambda_i} \leq \frac{1}{2\lambda_{max}}.$$

References

[1] Widrow, Bernard and Samuel Stearns. *Adaptive Signal Processing*. New Jersey: Prentice-Hall, Inc., 1985.