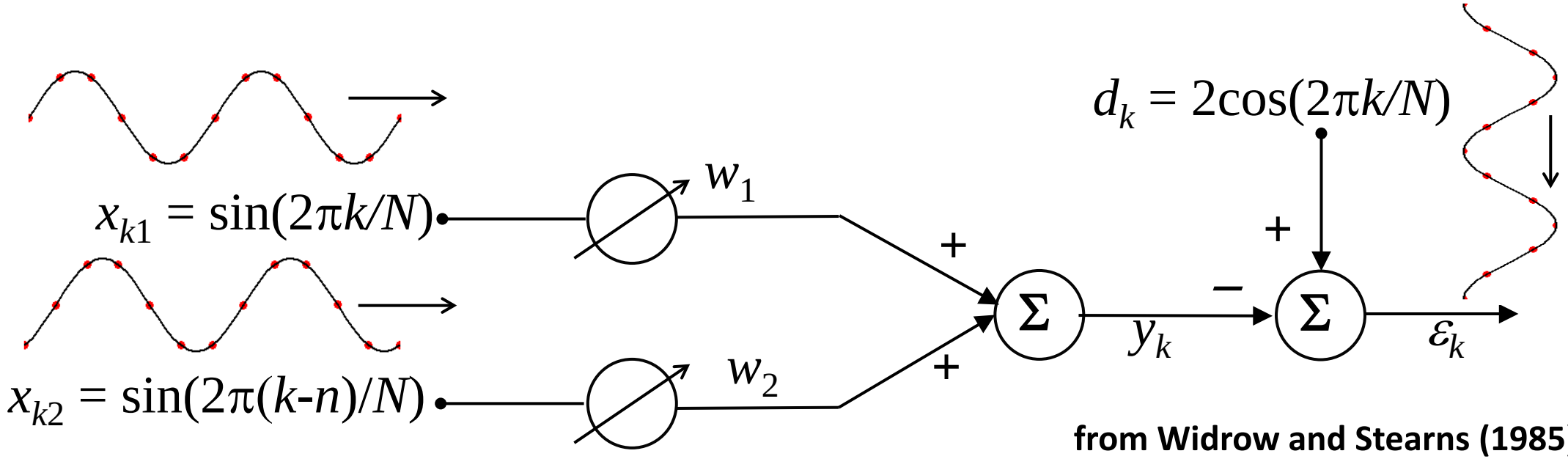
The Least-Mean-Square (LMS) algorithm and its geophysical applications

Brian Russell

Introduction

- The least-mean-square (LMS) algorithm is an adaptive filter developed by Widrow and Hoff (1960) for electrical engineering applications.
- It is used in applications like echo cancellation on long distance calls, blood pressure regulation, and noise-cancelling headphones.
- Along with the perceptron learning rule (Rosenblatt, 1962) the LMS algorithm also lead to the development of both linear and nonlinear neural networks (Rumelhart et al., 1986, Hagan et al., 1996).
- Thus, an understanding of the LMS algorithm is the first step in understanding neural networks and machine learning.
- In this talk, I will use examples from Widrow and Stearns (1985) and geophysics to explain the LMS algorithm, and also compare it to the least-squares, gradient descent and conjugate gradient methods.

The basic problem



from Widrow and Stearns (1985)

- Consider two shifted sinusoids input into separate recording channels, then weighted and summed so that the output produces an error ε_k with a desired sinusoid.
- The objective is to change the weights to reduce the error (ideally to 0), which in this case has the analytical solution:

$$w_1 = 2\cot\left(\frac{2\pi n}{N}\right) \text{ and } w_2 = -2\csc\left(\frac{2\pi n}{N}\right)$$

Recording all the samples

- In geophysical measurements we record all the data first as a complete time series, rather than recording it one sample at a time, giving in general:

$$\varepsilon = d - y = d - XW = \begin{bmatrix} d_0 \\ \vdots \\ d_{N-1} \end{bmatrix} - \begin{bmatrix} x_{01} & x_{02} \\ \vdots & \vdots \\ x_{N-1,1} & x_{N-1,2} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

- Setting $N = 6$ and $n = 1$ in our example gives (with computed weights):

$$y = 1.155 \begin{bmatrix} 0 \\ 0.866 \\ 0.866 \\ 0 \\ -0.866 \\ -0.866 \end{bmatrix} - 2.309 \begin{bmatrix} -866 \\ 0 \\ 0.866 \\ 0.866 \\ 0 \\ -0.866 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ -2 \\ -2 \\ -1 \\ 1 \end{bmatrix}$$

The squared error

- Squaring the error using vector notation, we get:

$$\varepsilon^T\varepsilon = d^T d + W^T X^T XW - 2d^T XW \Rightarrow \sum_{k=1}^{N}\varepsilon_k^2 = \sum_{k=1}^{N} d_k^2 - W^T RW - 2P^T W$$

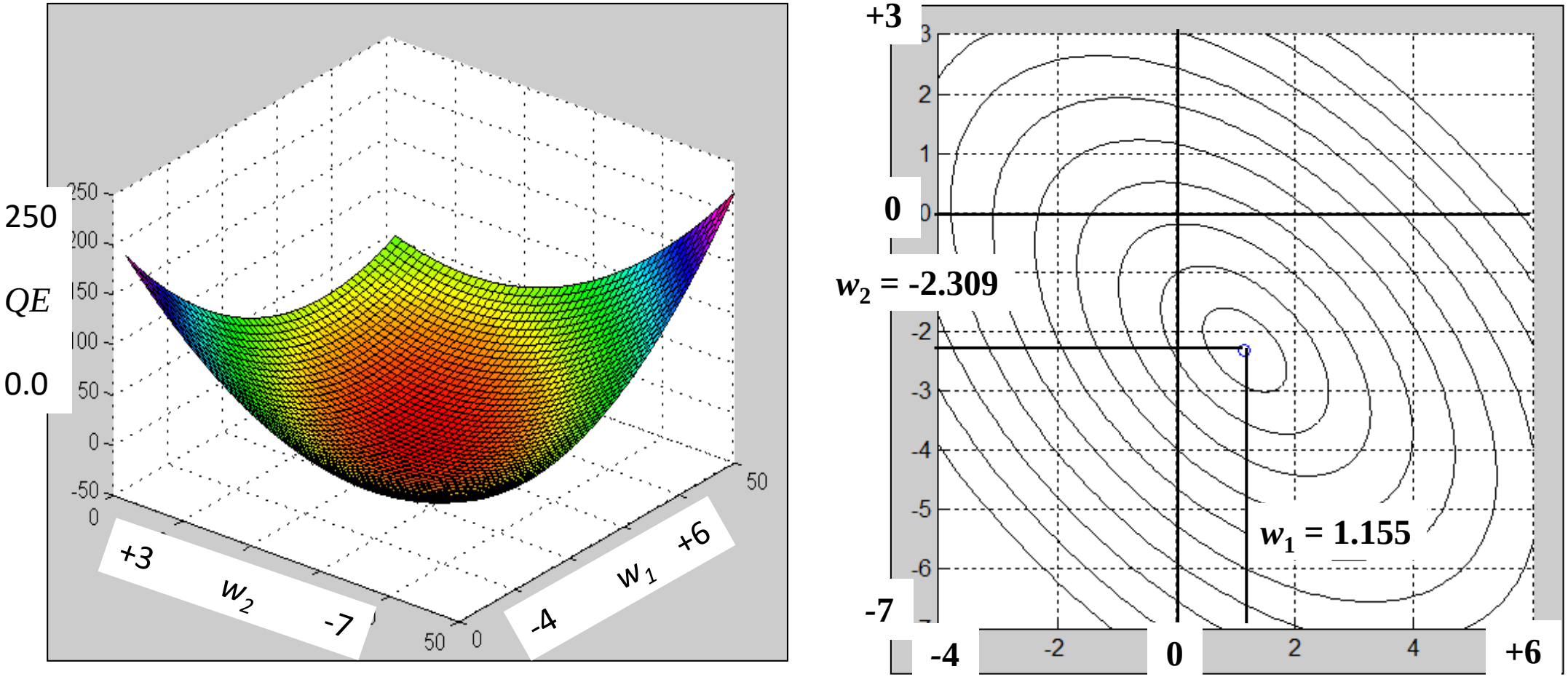
- In this equation, the matrix $R = X^T X$ is the cross-correlation of the input vectors and the vector $P^T = d^T X$ is the cross-correlation of the desired output with the input (note that often these are both divided by N).

- In our example, R and P are: $R = \begin{bmatrix} 3 & 1.5 \\ 1.5 & 3 \end{bmatrix}$ and $P = \begin{bmatrix} 0 \\ -5.196 \end{bmatrix}$

- We can call also define the quadratic error (QE):

$$QE = \sum_{k=1}^{N}\varepsilon_k^2 - \sum_{k=1}^{N} d_k^2 = W^T RW - 2P^T W$$

The quadratic error surface



A 3D view of the quadratic error surface (left) and the contoured surface (right), where the correct weights are found at the minimum error.

The gradient

- The gradient of the mean-square-error can be obtained by differentiating the error with respect to the weights, or:

$$\nabla = \frac{\partial \xi}{\partial W} = RW - P, \text{ where } \xi = \sum_{k=1}^{N} \varepsilon_k^2.$$

- Widrow and Stearns(1985) include a factor of 2 in the gradient, which I have dropped (as do most books on optimization, such as Gill et al., 1981).
- The obvious solution is to set the gradient to zero and invert:

$$\nabla = 0 \Rightarrow W = R^{-1}P \Rightarrow W = \begin{bmatrix} 0.444 & -0.222 \\ -0.222 & 0.444 \end{bmatrix} \begin{bmatrix} 0 \\ -5.196 \end{bmatrix} = \begin{bmatrix} 1.155 \\ -2.309 \end{bmatrix}$$

- But for large geophysical and neural network problems, this is not an option due to the size of the datasets, and we need to find other methods.

Steepest descent

- All search methods for finding the best solution start with an initial guess and then iterate towards the answer, using the gradient in some way:

$$W_{i+1} = W_i + \Delta W_i, \text{ where } i = 0,1,\ldots,M$$

  - One approach is Newton's method, but it requires the full inverse of *R*.
  - A less costly approach is called steepest descent:

$$W_{i+1} = W_i - \alpha_i \nabla_i, \text{ where } \alpha_i = \text{the step size.}$$

  - That is, we approach the solution in a series of steps controlled by the step size α_i, each time updating the gradient and the step size.
  - The most crucial choice in gradient descent is therefore the value of α_i.

Finding the step size

- It can be shown the value of the step size should be somewhere between 0 and 2 divided by the largest eigenvalue of the matrix R, or:

$$0 < \alpha < \frac{2}{\lambda_{\max}}, \text{ where } \lambda_{\max} = \text{largest eigenvalue of } R.$$

- The eigenvalues of R are $\lambda_{max} = 4.5$ and $\lambda_{min} = 1.5$, so $2/\lambda_{max} = 0.444$.
- The optimum value of α_i at each step is found by line minimization to be:

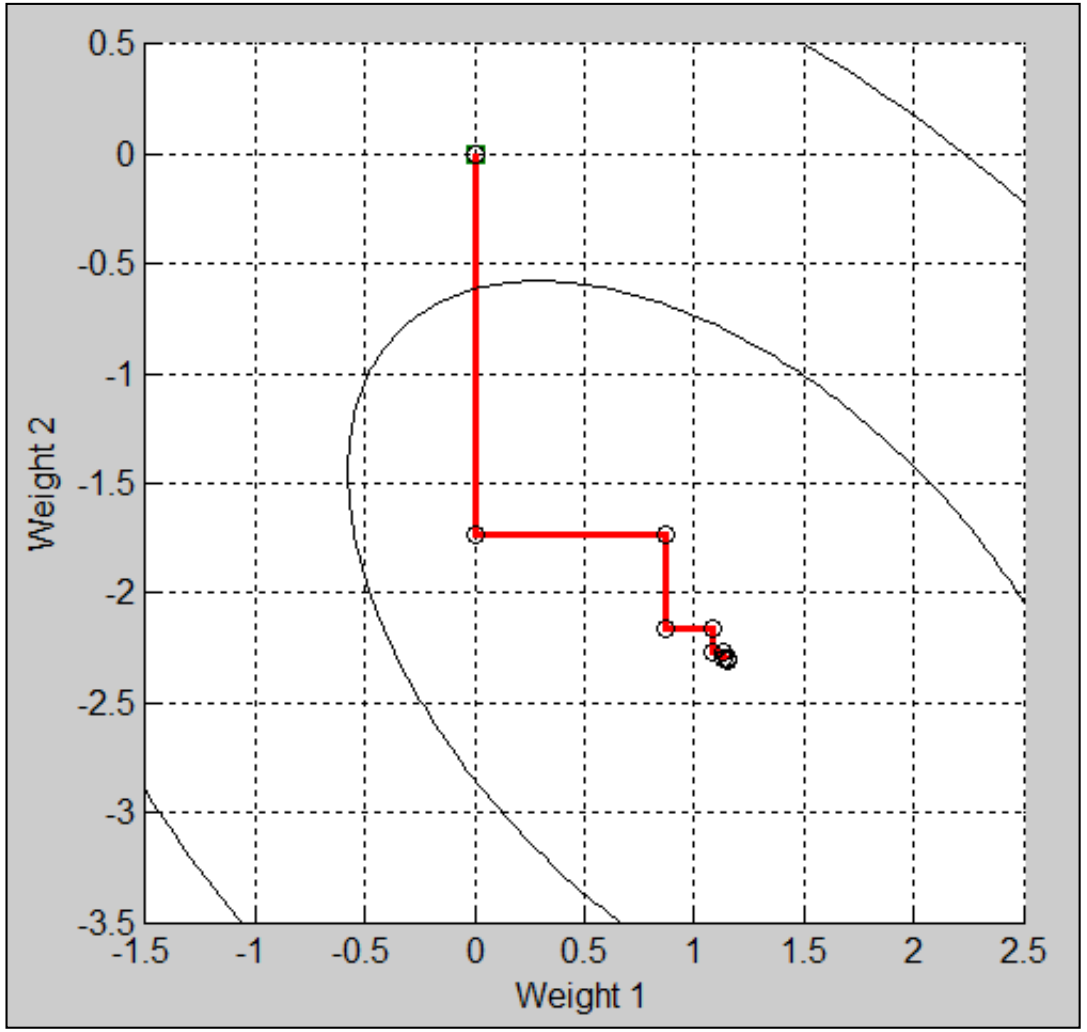
$$\alpha_i = \frac{\nabla_i^T \nabla_i}{\nabla_i^T R \nabla_i} = 0.333 \text{ for the first iteration (1/average of eigenvalues).}$$

- Note that this means that α will vary for each iteration.

The steepest descent path

- This figure shows the path taken by the steepest descent algorithm using the optimum step size given in the last slide, starting at an initial guess of $W_0^T = [0,0]$.
- Note that each step is orthogonal to the previous step.
- The first step is the largest and is computed as:

$$W_1 = W_0 - \alpha_0 \nabla_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \frac{1}{3}\begin{bmatrix} 0 \\ 5.2 \end{bmatrix} = \begin{bmatrix} 0 \\ -1.733 \end{bmatrix}$$



The conjugate gradient (CG) method

- In the conjugate gradient (CG) method (Hestenes and Steifel, 1952), conjugate directions p_i are found in which $p_i^T R p_j = 0$.
- The conjugate gradient algorithm can be written:

$$W_{i+1} = W_i + \alpha_i p_i, \text{ where } p_0 = -\nabla_0,$$

$$\text{and } p_i = -\nabla_i + \beta_i p_{i-1} \text{ for } i > 0, \text{ where } \beta_i = \frac{\nabla_i^T \nabla_i}{\nabla_{i-1}^T \nabla_{i-1}}.$$

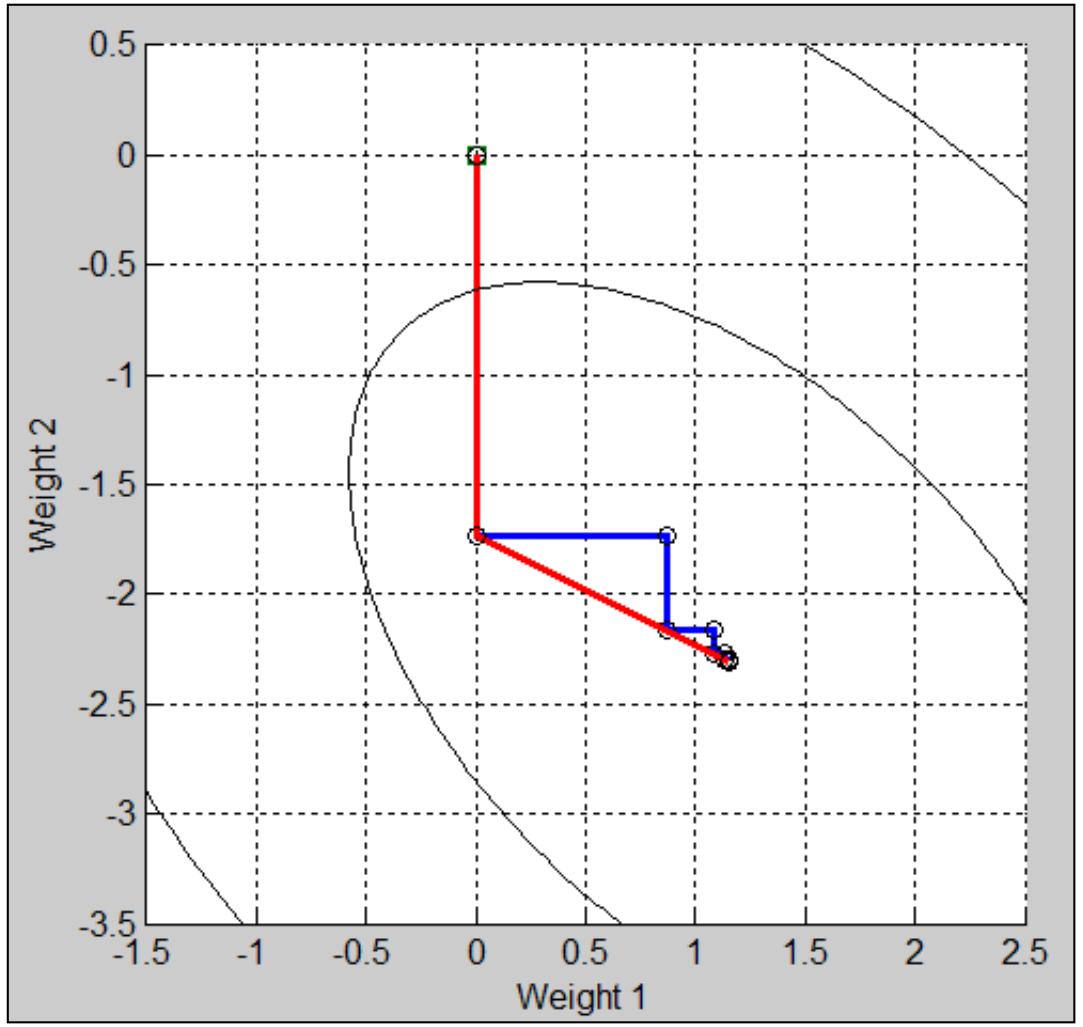
- To show that this works on our dataset, note that the calculations for the first two conjugate gradient directions give:

$$p_0 = -\nabla_0 = \begin{bmatrix} 0 \\ -5.2 \end{bmatrix}, p_1 = \begin{bmatrix} 2.6 \\ -1.3 \end{bmatrix} \Rightarrow p_0^T R p_1 = \begin{bmatrix} 0 & -5.2 \end{bmatrix} \begin{bmatrix} 3 & 1.5 \\ 1.5 & 3 \end{bmatrix} \begin{bmatrix} 2.6 \\ -1.3 \end{bmatrix} = 0$$

The conjugate gradient method

- The conjugate gradient algorithm is shown as the red curve on this plot.
- It always converges to the correct answer in the same number of iterations as the number of unknown weights.
- The steepest descent path is shown in blue on this plot, where the first step is equal to that of CG.
- Note that the second step of CG gives the correct weights:

$$W_2 = \begin{bmatrix} 0 \\ -1.733 \end{bmatrix} + 0.444 \begin{bmatrix} 2.6 \\ -1.3 \end{bmatrix} = \begin{bmatrix} 1.15 \\ -2.31 \end{bmatrix}$$



The Least-Mean-Square Algorithm

- Recall that the error at each input sample is given by:

$$\varepsilon_k = d_k - y_k = d_k - X_k W_k = d_k - \left[x_{k1} w_{k1} + x_{k2} w_{k2}\right]$$

- Also recall the definition of the gradient, and note that for our two weight example we can write:

$$\nabla_k = \begin{bmatrix} \frac{\partial \varepsilon_k^2}{\partial w_1} \\ \frac{\partial \varepsilon_k^2}{\partial w_2} \end{bmatrix} = 2\varepsilon_k \begin{bmatrix} \frac{\partial \varepsilon_k}{\partial w_1} \\ \frac{\partial \varepsilon_k}{\partial w_2} \end{bmatrix} = -2\varepsilon_k \begin{bmatrix} x_{k1} \\ x_{k2} \end{bmatrix} = -2\varepsilon_k X_k^T$$

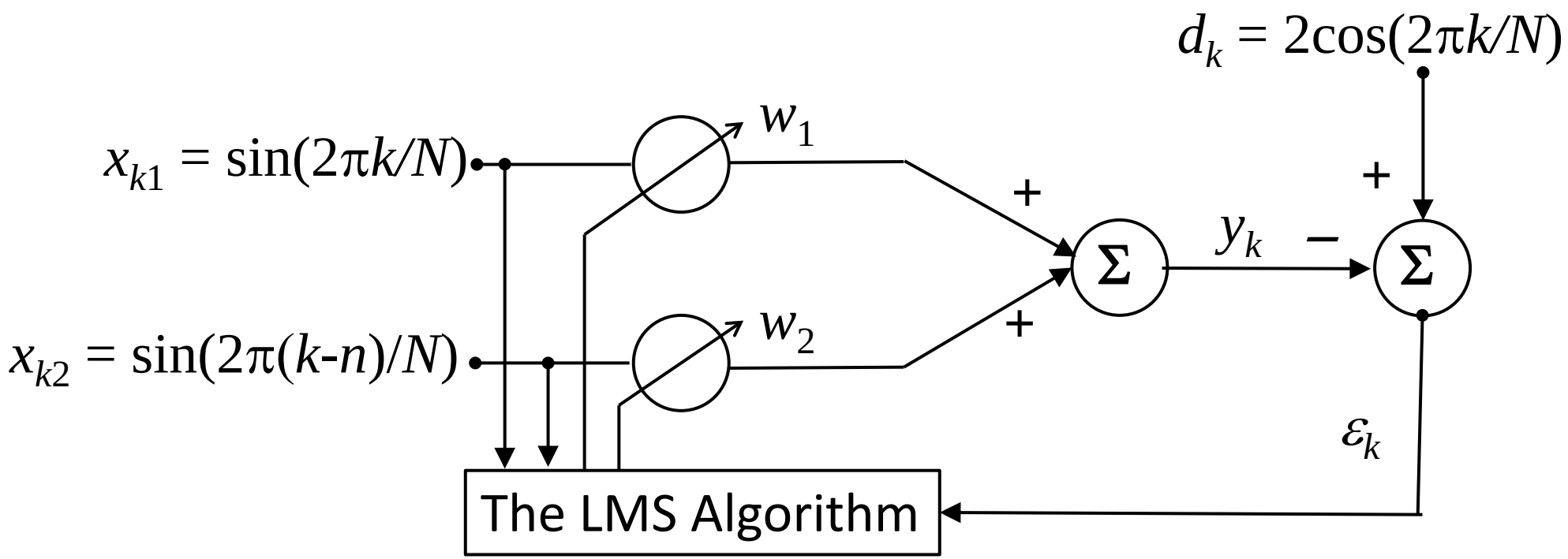
The Least-Mean-Square Algorithm

- This suggests that we update the weights after each sample, as follows:

$$W_{k+1} = W_k - \alpha\nabla_k = W_k + 2\alpha\varepsilon_k X_k^T$$

- This is called the Least-Mean-Square, or LMS, algorithm.
- Since the full correlation matrix is not available, the step size α cannot be calculated for each step, so is set to a reasonable value for the complete set of iterations.
- The block diagram for the LMS solution is shown in the next slide, and the solution to our problem is shown in the following slide with α set to 0.333.

The LMS algrorithm

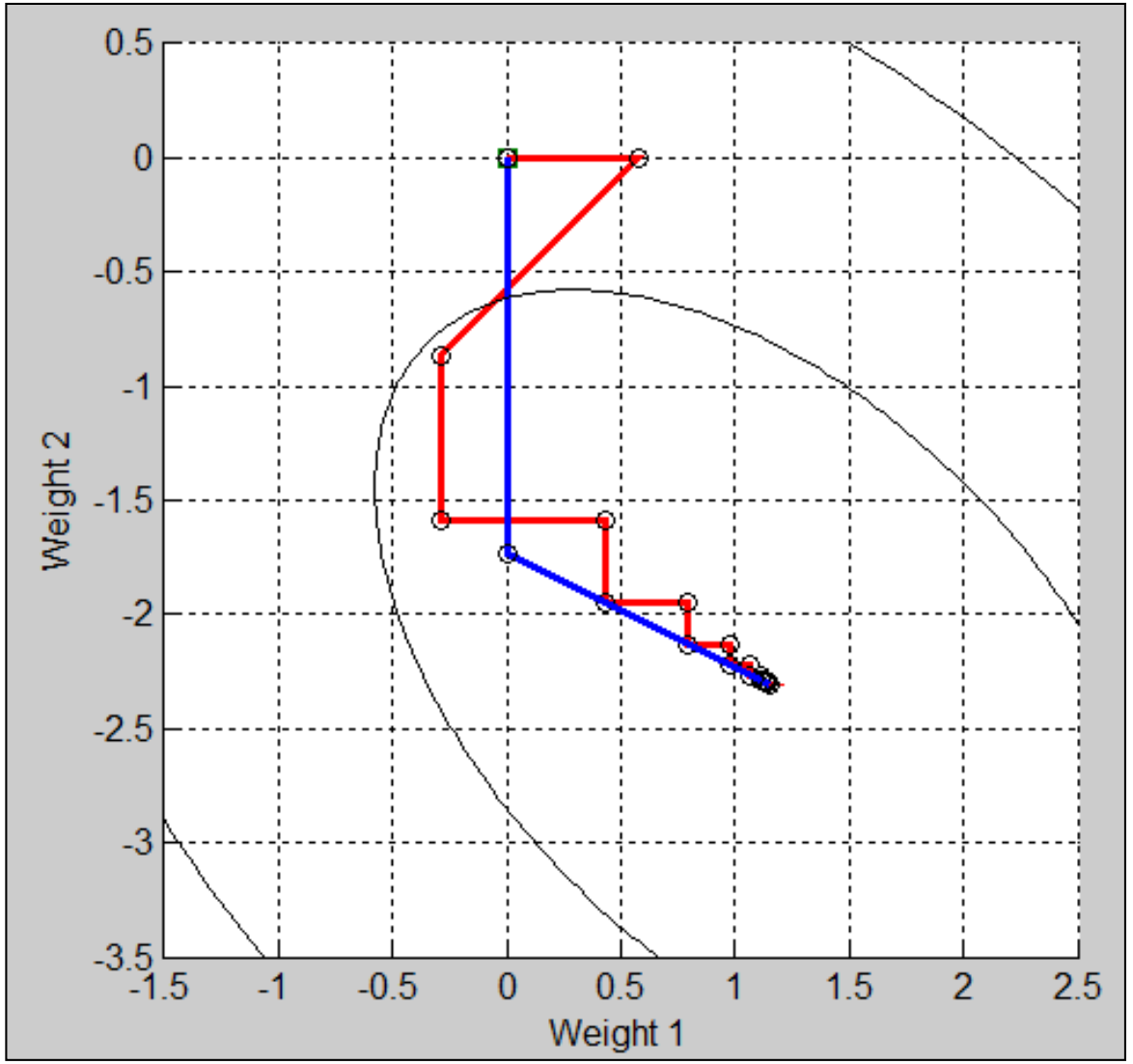


from Widrow and Stearns (1985)

- In the LMS algorithm the error is fed back into the algorithm after each sample is input and the weights are adjusted.

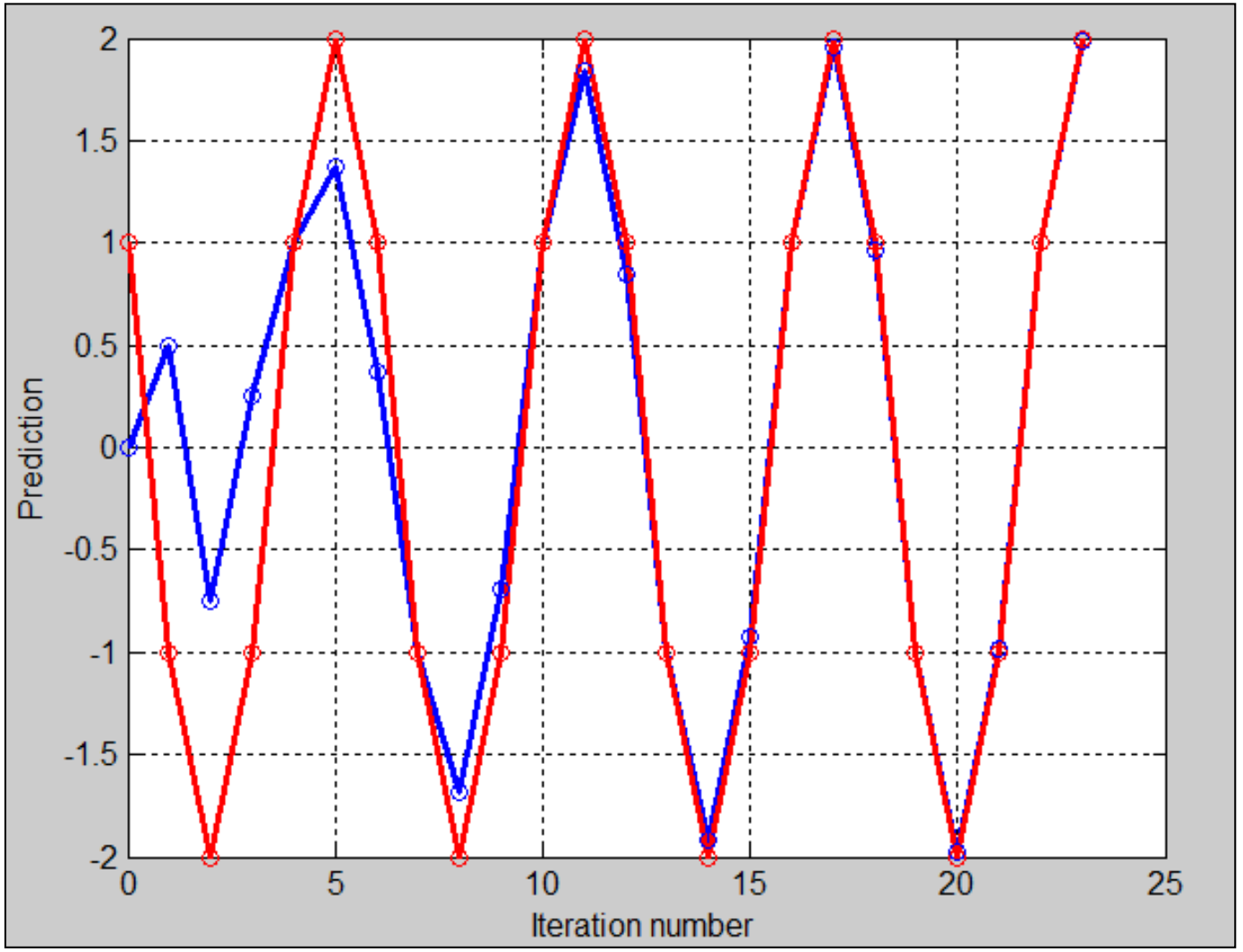
The LMS path

- The path taken by the LMS algorithm is shown in red.
- The conjugate gradient path is shown in blue.
- Note that the LMS path is chaotic at first but soon takes orthogonal steps like steepest descent.



LMS prediction

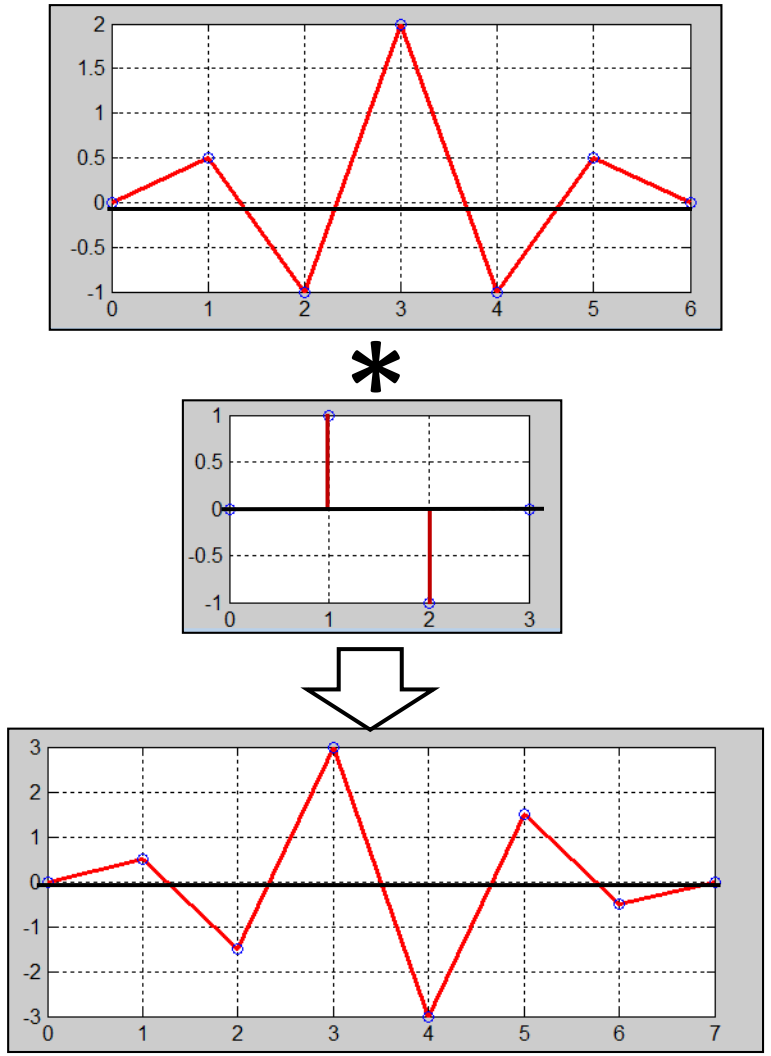
- The red curve shows the desired sinusoid and the blue curve shows the LMS prediction after every iteration.
- Note that it fits almost perfectly after 19 iterations.
- This is because we have a repetitive signal that we can keep feeding into LMS.
- Let us now see how the algorithm would work on geophysical problems.



First seismic example: deconvolution

- Let us now consider deconvolution (Claerbout, 1976, Robinson and Treitel, 2000) using the wavelet, reflectivity and seismic as shown:
- We can write this as follows, where s = the seismic, W = the wavelet matrix and r = the reflectivity:

$$Wr = s \Rightarrow \begin{bmatrix} 0.5 & 0 \\ -1 & 0.5 \\ 2 & -1 \\ -1 & 2 \\ 0.5 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ -1.5 \\ 3 \\ -3 \\ 1.5 \\ -0.5 \end{bmatrix}$$

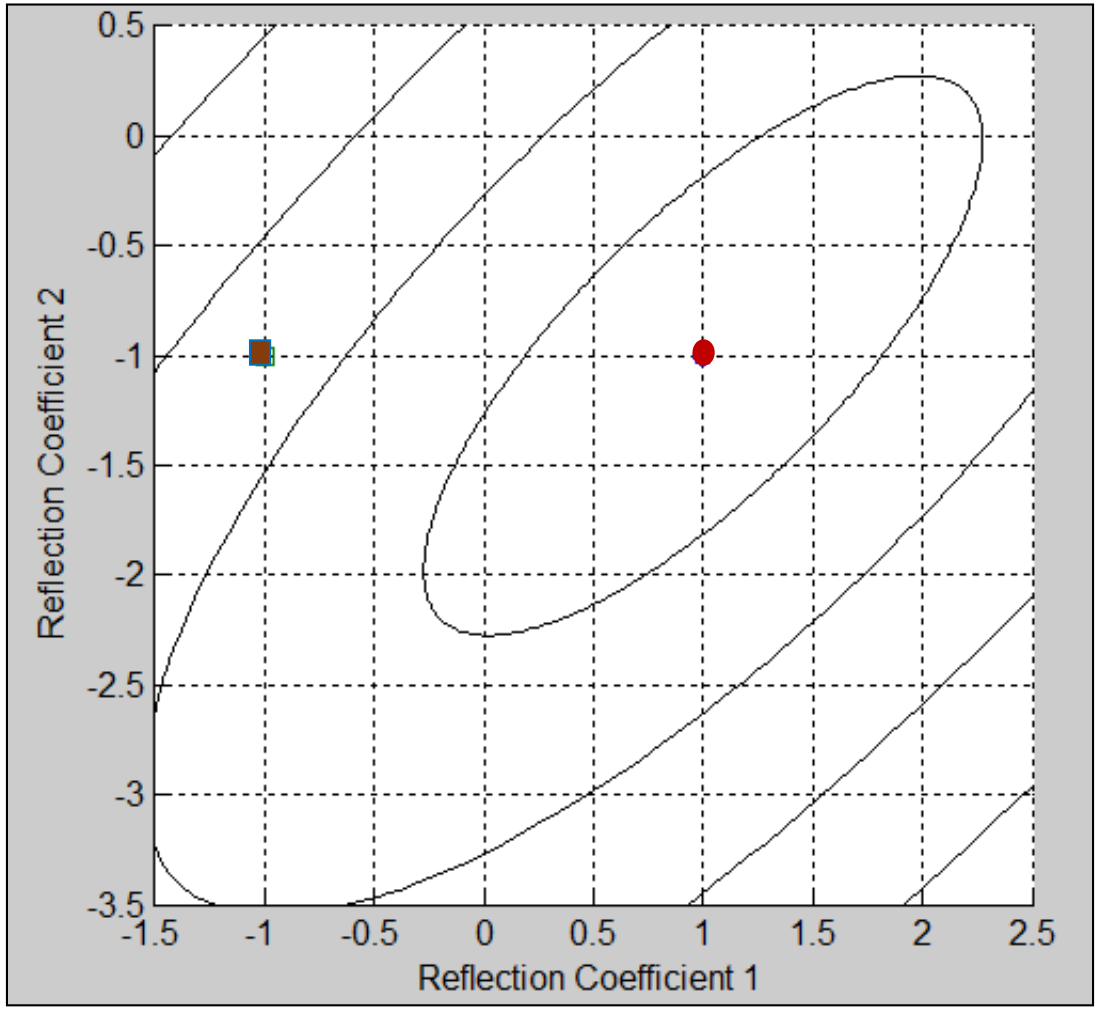


Least-squares solution and quadratic error plot

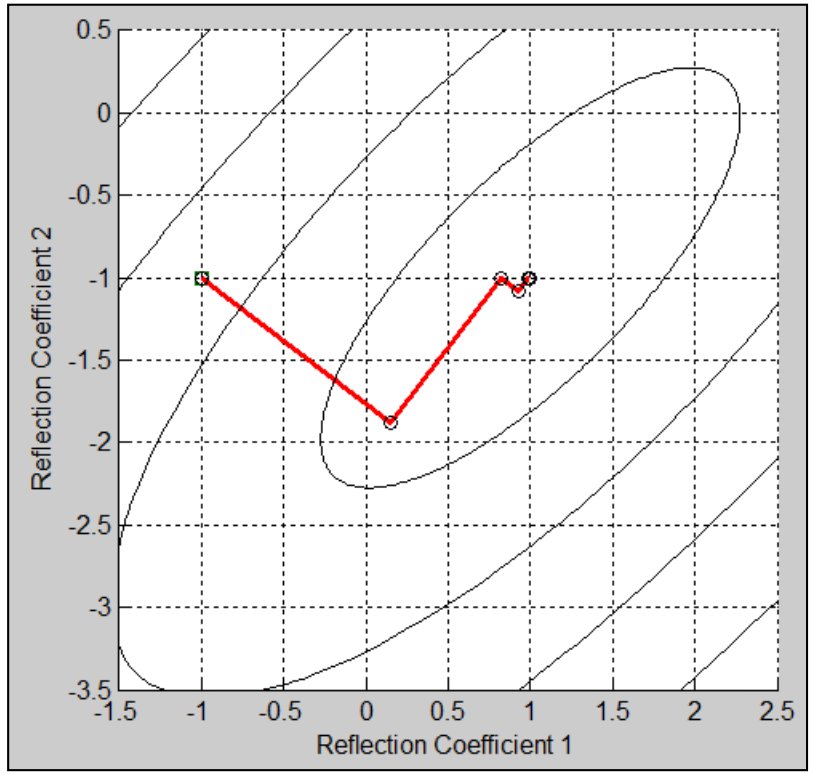
- As expected, the least-squares solution gives us the correct result as shown below, plotted as the red circle on the quadratic error surface on the right:

$$r = (W^T W)^{-1}(W^T s) = \begin{bmatrix} 0.377 & 0.29 \\ 0.29 & 0.377 \end{bmatrix} \begin{bmatrix} 11.5 \\ -11.5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

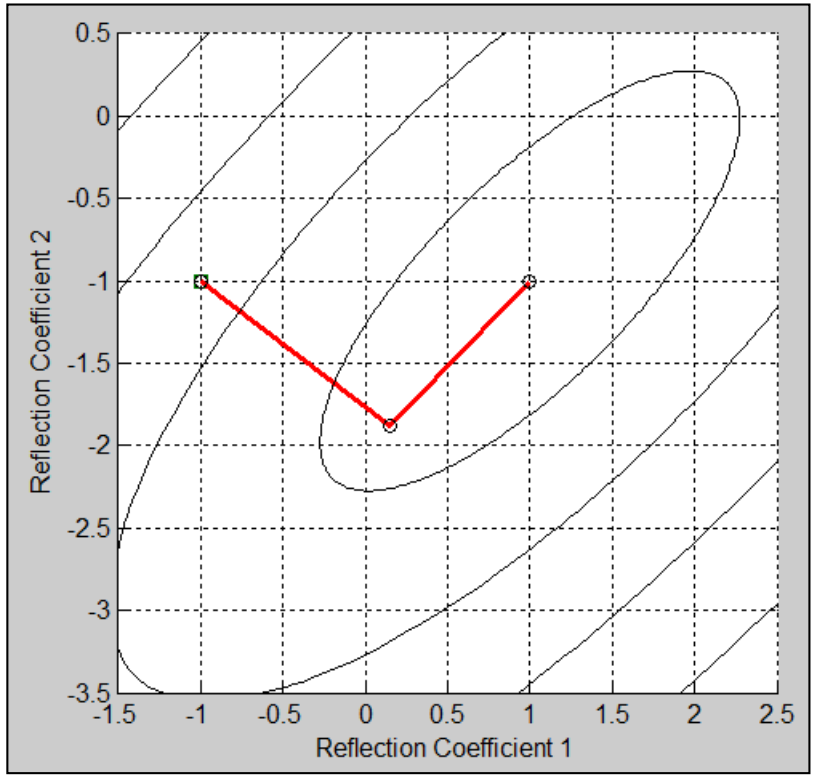
- As a starting guess for the gradient search methods, we will use r^T= [-1,-1], shown as the blue square on the quadratic error plot.



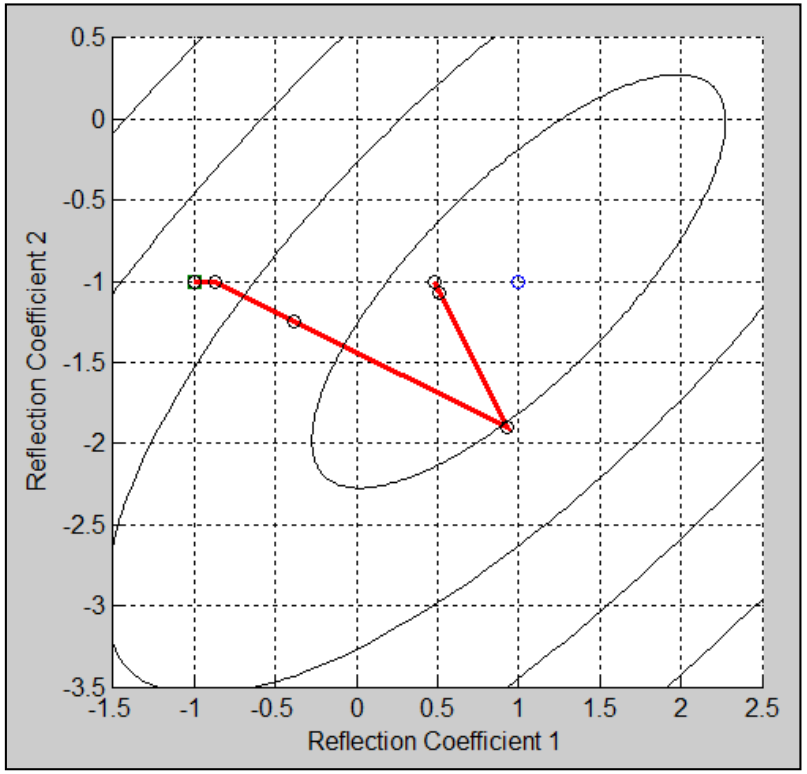
Gradient search solutions



Steepest descent solution.



Conjugate gradient solution.



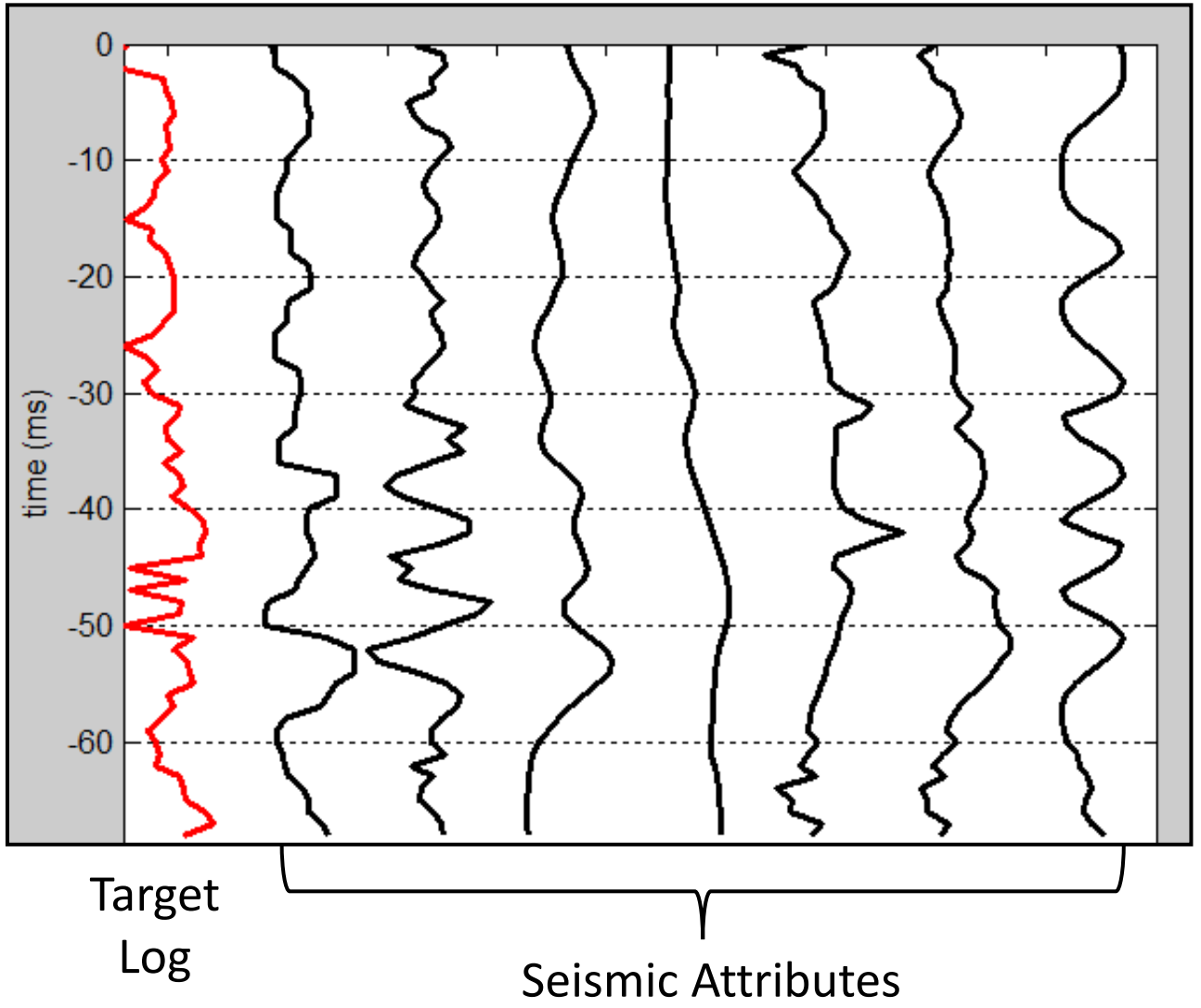
Least-mean-square solution.

The three gradient search solutions are shown above, where both steepest descent and conjugate gradient have converged to the correct answer but LMS has not converged due to the small number of samples in this example.

Second seismic example: attribute prediction

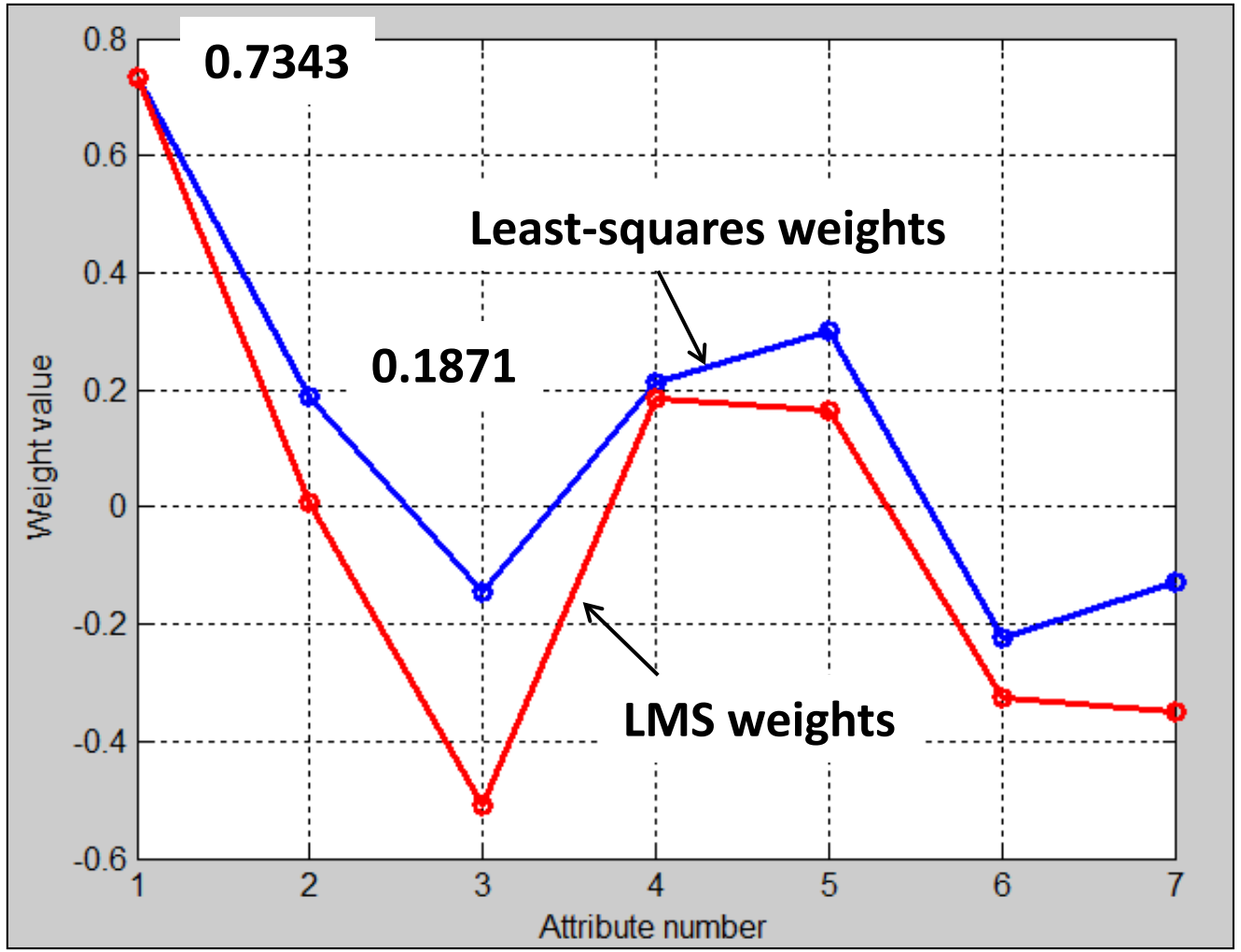
- We now consider a second seismic example in which we predict a target porosity log (shown in red) using a weighted set of seismic attributes (shown as the black traces).
- In this case we have seven normalized attributes (zero mean, unit standard deviation), so the equation is:

$$\phi = \sum_{i=1}^{7} w_i A_i, \text{ where } A_i = i^{th} \text{ Attribute.}$$



Second seismic example: attribute prediction

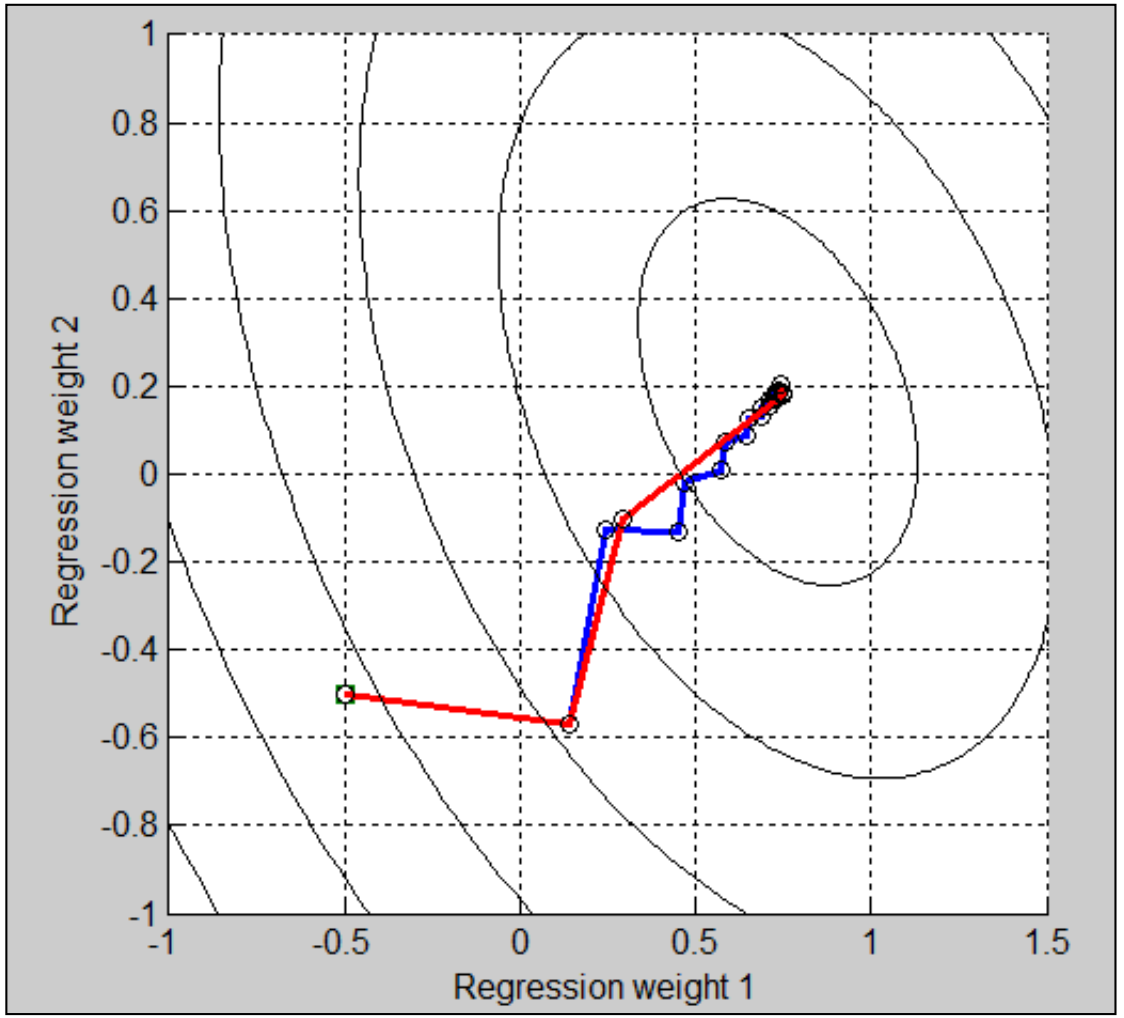
- This figure shows the least-squares weights (blue) and LMS weights (red), where they have been normalized to the first weight (the first two weight values are shown).
- Since LMS was sensitive to the initial guess, the guess was initialized using random numbers between -0.5 and +0.5.
- Also key was the step size α and it was adjusted until the LMS error was optimized.



Second seismic example: attribute prediction

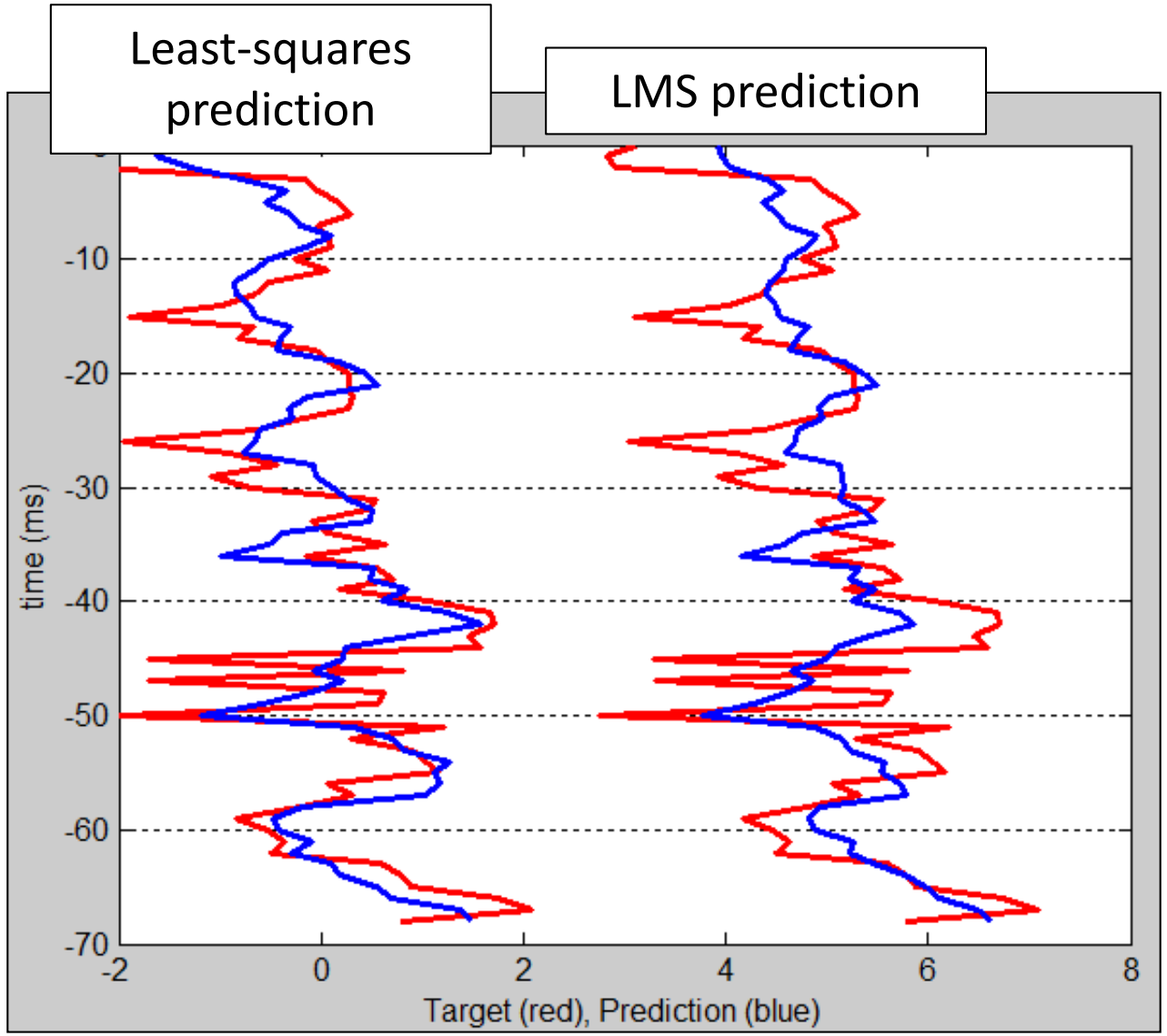
- Here is the convergence of steepest descent (blue) and conjugate gradient (red) for the first two weights, using a starting guess where all weights = -0.5.
- The actual values below show that CG took 7 iterations:

	Weight 1	Weight 2
Iteration 1:	-0.5000	-0.5000
	0.1415	-0.5696
	0.2943	-0.1018
	0.7490	0.1810
	0.7440	0.2067
	0.7483	0.1865
	0.7347	0.1872
Iteration 7:	0.7343	0.1871



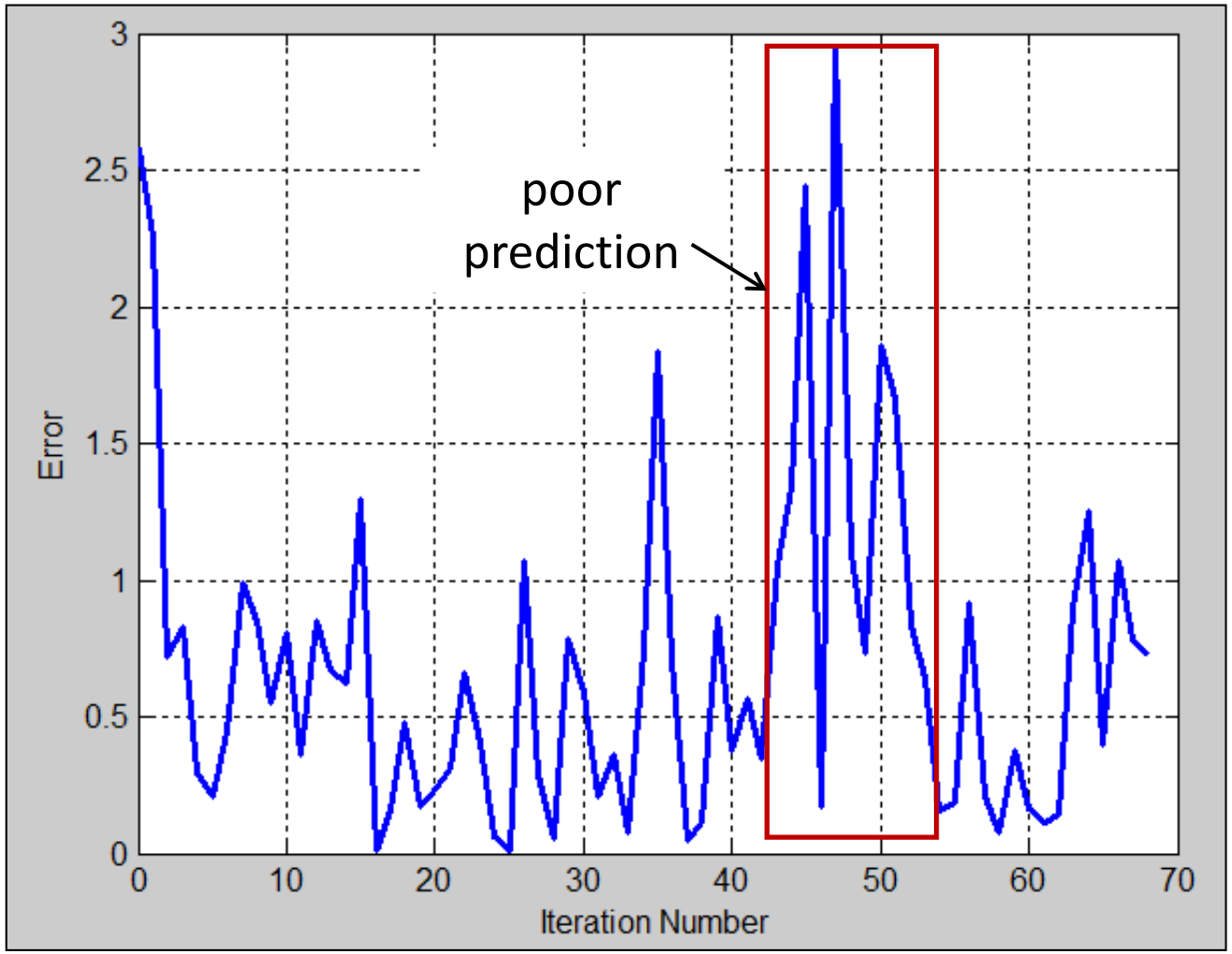
Second seismic example: attribute prediction

- Here is the predicted target log for both algorithms, where red shows the true log and blue shows the predicted log.
- As shown here, the least-squares solution gives a better fit.
- Depending on the initial guess, the LMS answer can get better or worse than this!
- This is probably due to the fact there is no exact answer.



Second seismic example: attribute prediction

- However, a useful diagnostic that we can derive from the LMS algorithm is the relative error after each iteration.
- This gives us an indication of how well each newly derived set of weights predicts the target log.
- As expected, when we hit the very high frequency values between samples 45 and 55, the prediction is poor.



Conclusions

- The least-mean-square (LMS) algorithm is an adaptive filter developed by Widrow and Hoff (1960) for electrical engineering applications.
- In this talk, I used examples from Widrow and Stearns (1985) and geophysics to explain the LMS algorithm, and also compare it to the least-squares, gradient descent and conjugate gradient methods.
- For the example using continuous sinusoids, LMS converged to a solution equivalent to the least-squares and gradient methods.
- But for our geophysical applications, LMS was inferior to least-squares and gradient methods, due to lack of iterations (decon) and noise (regression).
- However, the LMS algorithm is the method used in neural networks and machine learning, where the data arrives sample by sample.
- Thus, an understanding of LMS is crucial to understanding neural networks.

Acknowledgements

- I wish to thank the CREWES sponsors and my colleagues at Hampson-Russell, CGG, and CREWES.

References

Claerbout, J, 1976, Fundamentals of Geophysical Data Processing: McGraw Hill, Inc.

Gill, P.E., Murray, W., and Wright, M.H.,1981, Practical Optimization: Academic Press, New York.

Hagan, M.T., Demuth, H.B., and Beale, M., 1996, Neural Network Design: PWS, Boston.

Hestenes, M.R., and Stiefel, E., 1952, Methods of conjugate gradients for solving linear systems: Journal of Research of the National Bureau of Standards, 29, 409-439.

Robinson, E.A., and Treitel, S., 2000, Geophysical Signal Analysis: SEG, Tulsa.

Rosenblatt, M., 1958, The perceptron: A probabilistic model for information storage and organization in the brain: Psychological Review, 65, 386-408.

Rummelhart, D.E., Hinton, G.E., and Williams, R.J., 1986, Learning representations of back-propagation errors: Nature, 323, 533-536.

Widrow, B., and Hoff, M.E., 1960, Adaptive switching circuits: *IRE WESCON* Conv. Rec., pt. 4, p 96-104.

Widrow, B., and Stearns, S., 1985, Adaptive Signal Processing: Prentice-Hall, New York.