geoconvention
Virtual Event
September 21-23
2020
CSPG • CSEG • CWLS • GAC • MAC • IAH

Machine learning and geophysical inversion

Brian H. Russell, CGG GeoSoftware

Summary

In this presentation, a numerical example is used to illustrate the difference between geophysical inversion and machine learning inversion. In doing so, an attempt is made to demystify machine learning algorithms and show that, like inverse problems, they have a definite mathematical structure that can be written down and understood. The example used in this study is the extraction of the underlying reflection coefficients from an overlapping wavelet response that was created by convolving a reflection coefficient dipole with a symmetric wavelet. In discussing the solution to this problem the topics of deconvolution, recursive inversion, linear regression and nonlinear regression using a feedforward neural network are all discussed and compared.

The forward and inverse seismic model

The forward model used is shown in Figure 1 (Russell,2019), where the geology in Figure 1a consists of a thin porous wet sand of P-impedance 5500 m/s*g/cc between two shale layers of P-impedance 4500 m/s*g/cc (shown in Figure 1(b)), which results in reflection coefficients of + and –0.1 (Figure 1((c)). The reflectivity is then convolved with a three point symmetric wavelet given by (-1,2,-1) to produce the synthetic seismic trace as shown in Figure 1(d).

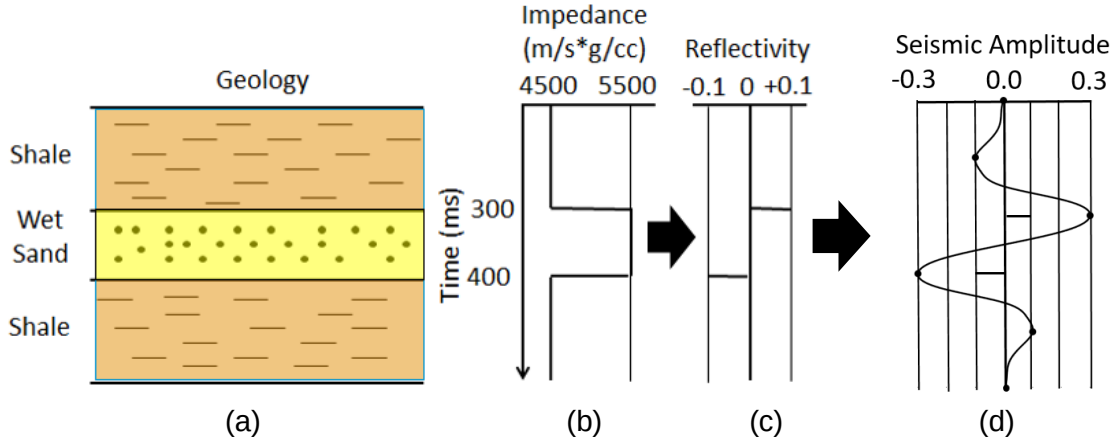


Figure 1. The forward seismic model, where (a) is the geology, (b) is the P-impedance, (c) is the reflection coefficients, and (d) is the synthetic seismic trace.

Mathematically, we can write

$$\boldsymbol{s} = G\boldsymbol{r} = \begin{bmatrix} -1 & 0 \\ 2 & -1 \\ -1 & 2 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} +0.1 \\ -0.1 \end{bmatrix} = \begin{bmatrix} -0.1 \\ +0.3 \\ -0.3 \\ +0.1 \end{bmatrix}, \tag{1}$$

where $\boldsymbol{s}$ is the seismic trace vector, G is the geophysical wavelet matrix and $\boldsymbol{r}$ is the reflectivity. If we know the wavelet matrix we can invert equation (1) using least-squares deconvolution, as follows:

$$\boldsymbol{r} = (G^T G)^{-1} G^T \boldsymbol{s} = \begin{bmatrix} -0.3 & 0.4 & 0.1 & -0.2 \\ -0.2 & 0.1 & 0.4 & -0.3 \end{bmatrix} \begin{bmatrix} -0.1 \\ +0.3 \\ -0.3 \\ +0.1 \end{bmatrix} = \begin{bmatrix} +0.1 \\ -0.1 \end{bmatrix}, \tag{2}$$

which recovers the correct reflection coefficients.

Inversion versus Machine Learning

In inversion using deconvolution we extract the wavelet and apply its inverse. However, in the machine learning approach, we present the input seismic trace and desired reflectivity to the machine learning algorithm and let the algorithm determine the relationship, as shown in Figure 2. That is, machine learning does not understand the physics of the problem, but develops a mathematical transform to convert the seismic trace into reflectivity.

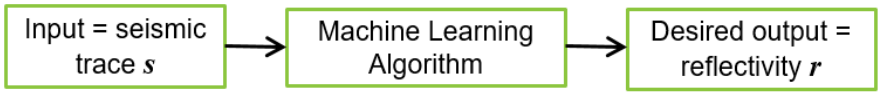


Figure 2. The basic concept behind Machine Learning

Since the most common type of machine learning algorithm, the feedforward neural network, is a type of nonlinear regression, we will start by applying linear regression to the problem.

Linear regression

In linear regression, we need to find the weights for the following relationship between the reflectivity and seismic trace

$$\hat{\boldsymbol{r}} = w_0 + w_1 \boldsymbol{s}\,, \tag{3}$$

where w_0 is the intercept and w_1 is the gradient. This can be done using the same approach as in equation (2), but where the unknowns are now the weights.

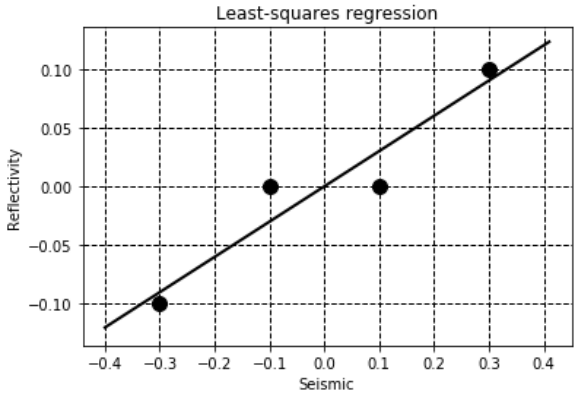


Figure 3. The linear regression solution.

Computing the weights for the seismic deconvolution problem gives w_0=0 and w_1= 0.3. The linear regression result is shown by the black line in Figure 3, where the black circles represent the true values. This illustrates the difference between the regression and deconvolution approaches. In deconvolution we got a perfect fit to the four points because our assumptions about the model were correct. In least-squares regression, the points have been fit in a "best" least-squares sense, which is not exact.

The feedforward neural network

Now, let's see how the feedforward neural network differs from the linear regression approach. Figure 4 is a flowchart that illustrates the method. In the forward process, the seismic trace is input into a first layer of two neurons, applying two weights per neuron, where the subscripts ij indicate the i^{th} weight into the j^{th} neuron and the superscript (1) indicates the first layer. This is similar to a linear regression of the input into each neuron, but differs because each neuron then applies a nonlinear function. In the first layer of neurons the logistic function given by

$$f(y) = \frac{1}{1+\exp(-y)} \tag{4}$$

is used, which has the sigmoidal shape shown in Figure 5. The outputs of the two neurons can be thought of as intermediate estimates of the final reflectivity, which are labelled $\hat{\boldsymbol{r}}_1^{(1)}$ and $\hat{\boldsymbol{r}}_2^{(1)}$ in Figure 4. These estimates are then weighted and fed into a third neuron, where the superscript (2) now indicates the second layer. In neuron 3 a linear function is used rather than a logistic function, so the weighted sum passes through unchanged, giving a final estimate of the reflectivity for each seismic sample t as:

$$\hat{r}_t^{(2)} = w_0^{(2)} + w_1^{(2)}\hat{r}_{1t}^{(1)} + w_2^{(2)}\hat{r}_{2t}^{(1)} = w_0^{(2)} + \frac{w_1^{(2)}}{1+\exp\left(-w_{01}^{(1)} - w_{11}^{(1)}s_t\right)} + \frac{w_2^{(2)}}{1+\exp\left(w_{02}^{(1)} + w_{12}^{(1)}s_t\right)}. \tag{5}$$

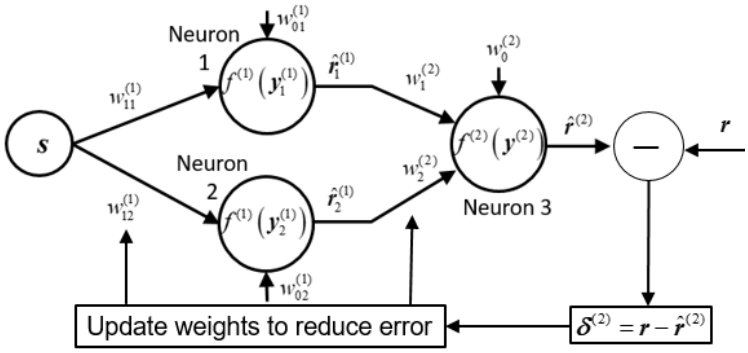


Figure 4. The feedforward network and backpropagation.

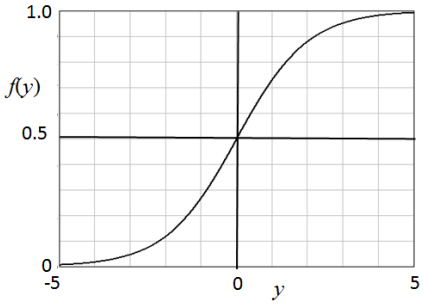
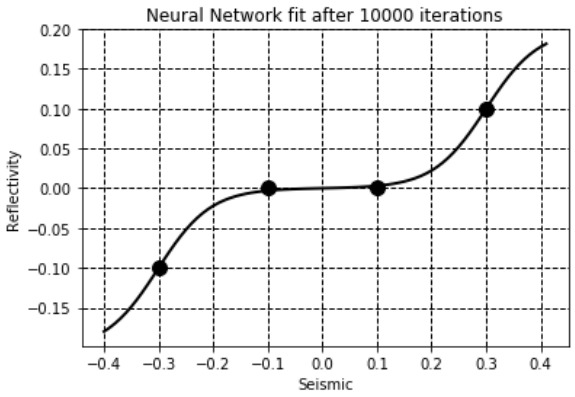


Figure 5. The logistic function.

To determine the weights, the backpropagation technique is used, as illustrated at the bottom part of Figure 4. An initial random set of weights are estimated and then the output reflectivity is subtracted from the true reflectivity to give an error term δ. The weights are then iteratively updated to decrease δ until an acceptable fit is found. In our case, the final weights are given as follows after 10,000 iterations through the network:

$$\hat{r}_i^{(2)} = 1.99 - \frac{2.03}{1+\exp\left(6.24+2.07s_i\right)} - \frac{1.99}{1+\exp\left(-6.23+2.07s_i\right)}, \tag{6}$$



This produces the fit shown in Figure 6, which should be compared with the linear regression fit in Figure 3. The four known points in Figure 6 now show an almost perfect fit, but the curve is nonlinear and shows the "imprint" of the logistic function. For seismic values above 0.3 and below -0.3, the curve starts to flatten off instead of staying straight as in Figure 3. Notice that although we have produced a perfect fit to the reflection coefficients, this fit has not honored the underlying geophysical model.

Figure 6. The output of the neural network.

Conclusions

An obvious conclusion to this study is that applying physics to a problem is better than applying a neural network, since then the solution has a real physical meaning and is not just a mathematical transform. But actual geophysical problems are not that simple. When the geophysics is fully understood and applicable it is always the better option. However, our geophysical solution is usually overly simplistic (for example, in the real earth we have to take into account dispersion of the wavelet, inhomogeneity and anisotropy in the earth layers, etc.). Therefore, a neural network might find nonlinearities in the solution that we were unaware of in our theory. Second, our example consisted of very few points. In real geophysical studies we have large amounts of data, so a neural network might find hidden regularities in the data that we have overlooked. Furthermore, large amounts of data allow us to cross-validate our results by leaving parts of the data out of the initial training and using our trained weights to predict those parts of the data that were unknown to the neural network algorithm. The answer is therefore a judicious combination of both physical theories and machine learning techniques in our attempts to understand the earth.

References

Russell, B.H., 2019, Machine learning and geophysical inversion — A numerical study: THE LEADING EDGE, 38**,** no. 7, 512-519.