Seismic-guided estimation of log properties

Part 2: Using artificial neural networks for nonlinear attribute calibration

By SHUKI RONEN, PHILIP S. SCHULTZ, MASAMI HATTORI, CHIP CORBETT
Schlumberger (GeoQuest and Geco-Prakla)

We saw in Part 1 that when we have 3-D data together with a number of logged wells, we can look at possible relationships between some attributes of the seismic data and various properties measured on the logs. At multiple well locations, where we have both seismic and log data, we can look for trends in these two data types on crossplots. If we see a trend, we can quantify it with a derived or specified functional relationship. This functional relationship can be used to convert the attribute values to log properties, and when followed by a residual correction, provide a means to estimate the distribution of these properties away from the wells.

Figure 9 (Figure 6 in Part 1) shows an apparently nonlinear relationship between the instantaneous frequency attribute and the volume fraction of clay log property. In this crossplot, data from 15 wells (both quantities) have been averaged over the vertical extent of the Bravo layer. Although any rock physics relationship between these measurables is obscure at best, the significance value for this crossplot is a compelling 7 1.2 percent.

We can postulate various possible mechanisms and argue their credibility. It is possible to imagine geologic settings where instantaneous frequency will be influenced by the shaliness, or volume, of clay, if the clay exists in stacked thin layers. However, let us assume that there is indeed some underlying physical mechanism for this relationship but that we don't know what it is. Rather than trying to derive a functional relationship from approximations to theory, we will estimate the function from the data themselves.

As evident in Figure 9, we often have to search for a nonlinear function when using seismic attributes to estimate distributions of log properties. Artificial neural networks (ANNS) can help determine nonlinear functions that best fit those relationships.

Neural networks are being increasingly used to solve a variety of mathematical problems concerned with unknown and varied functional relationships among measured variables. In this article, we briefly explain the basics of neural networks and then discuss further the principles of one type of neural network, the radial basis function feed-forward network (RBFN), as it has been applied to finding nonlinear calibration functions for attribute/property relationships.

This discussion of ANNS is limited to just one of the steps in seismic guided log property estimation-the calibration step where we find a smooth curve fit to the scatter of attribute/property data points in the crossplot. The estimation methodology discussed in Part 1 includes several other steps. A determination of which attribute/property pairs are significant precedes the calibration, while residual correction and confidence estimation steps follow to complete the estimation procedure.

A primer on ANNS. Assume that variables x and y are linearly related, $y = ax + b$, and are subject to additive noise. In our case, x normally represents the seismic attribute and y is related to the log property to be estimated away from the wells. We are given several observations of the paired data values (x,y) as follows:

$$(2.1, 5.0), (0.6, 1.8), (9.4, 20.2), (6.7, 13.9)$$

and we want to make a prediction for y when x = 4. The best least-squares fit of these known data points to a straight line is

$$y(x) = 2.06x + 0.54$$

which gives y(4) = 8.78 as our estimate. In this example, had we used a neural network to find the best straight line fit, the four given data points would be called the *learning set.* The *basisfunctions* would be the two polynomials, $f_1(x) = x^0$ and $f_2(x) = x^1$; $w_1 = 0.54$ and $w_2 = 2.06$ would be the *weights* derived from the learning set. Figure 10 shows a neural

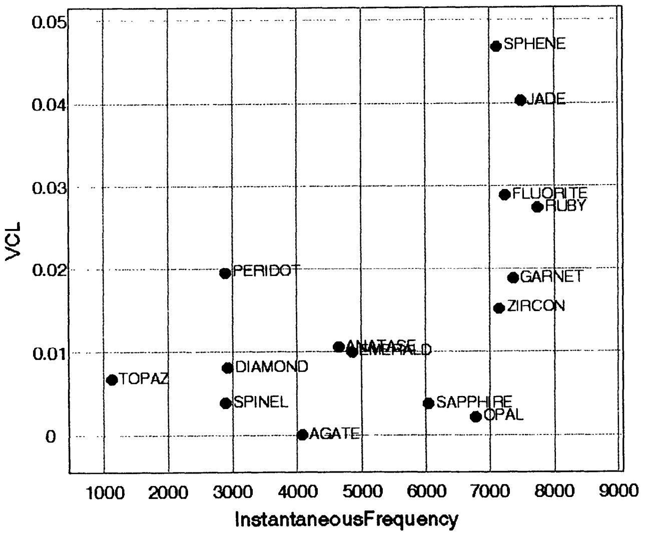

Figure 9. An apparently nonlinear relationship where the instantaneous frequency attribute is plotted against the volume fraction of clay log property. In this crossplot, both quantities have been averaged over the vertical extent of the Bravo layer. Although any rock physics relationships between these measurables is obscure at best, the crossplot does show a compelling significance value of 71.2 percent. We will use an ANN to learn a nonlinear function from these points.

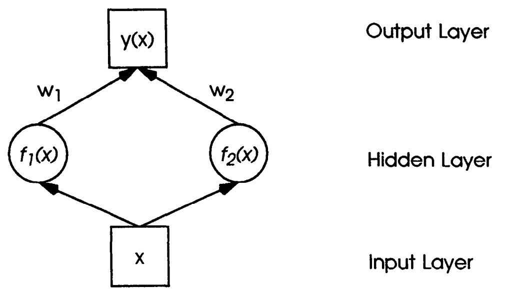

Figure 10. A neural network flow diagram for the case of a single input function, two basis functions, and one hidden layer, following the example in the text. The positions of the basis functions in the hidden layer are called hidden nodes. The algebraic equivalent of this figure is $y(x) = w_1 f_1(x) + w_2 f_2(x)$ where x is input and y is output. Functions f_1 and f_2 are prespecified, and weights are determined from the learning set.

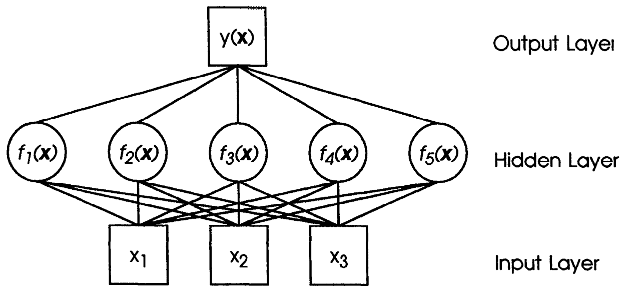

Figure 11. A single hidden layer neural network with three inputs and five hidden nodes, generating a single output function. In a similar manner to Figure 10, weights are applied to the hidden layer output prior to their summation into the output layer. In the case of multiple input variables, weights can be determined for the input layer also, but we alternatively can choose to normalize the variables prior to input.

network diagram of this curve fitting. The step at which the basis functions are evaluated with the input *x* is called the *hidden layer.*

The output, y, can be a function of more than one input variable (i.e., attribute) x = $(x_1, x_2, \ldots x_m)$. We may, for example, want to predict the value of porosity at some location in a geologic layer using, simultaneously, the instantaneous frequency attribute for that layer (which sometimes is related to the volume fraction of clay), the amplitude envelope attribute, and the seismic acoustic impedance attribute (both of which are sensitive to local velocity changes) to make the estimate. These three attributes are the x_1, x_2, and x_3 input variables, each representing known values from the seismic data.

Figure 11 shows a case of three input variables (perhaps the three attributes just mentioned) and five basis functions leading to a single output function. Each basis function is, in general, a function of all three input variables. If we normalize the variables prior to their input to the neural network, there is no need for input weights and there remain five weights to determine. There then must be more than five known data quadruplets (y, x_1, x_2, x_3) for the calculation of the weights $w_1 \ldots w_5$ to be overdetermined. This normally means more than five wells with log values measuring the property in that layer, where we know the desired output, y, and also all three input values.

In these examples, we described x and y as scalars. They can equally well be a two-dimensional grid of values (the usual case) where $x_1(u,v)$ might represent the seismic acoustic impedance values mapped over a geologic surface. In these cases, the processing acts as a serial operation, where the first grid points for each attribute are processed together in the hidden layer to generate the first output point (in the manner of the scalar example in Figure 11) before the second group of input points is read. Then, the second points from each attribute grid are processed, generating the second output point, and so on.

It is possible to introduce lag operators into the basis functions to process the *i + kth* input point(s) during the *ith* processing step. This would be appropriate for applying neural networks to time series prediction or spatial filtering, for example. However, let's assume that all temporal and spatial filtering has already been done. For the remainder of this article, we will concern ourselves with point operations, where the *ith* point in a series of input grids is used only to estimate the *ith* point in the output vector or grid. An input attribute will be written as x, and will normally be assumed to be a two-dimensional grid of values.

Radial basis function feed-forward. If we anticipate a linear relationship between the input and output variables, neural networks are a bit of overkill. We get the answer, but the procedure is unnecessarily cumbersome. We have found that neural networks are useful when searching for nonlinear relationships, particularly if the functional form of that relationship is not known. This relationship will be estimated (learned) from a set of observations and measurements (the learning set), typically from log and seismic data. When the relationship is learned, it will remain constant and can be used to predict or to estimate values of the desired log property away from the wells where we have only seismic data.

We want to choose the simplest, most transparent network architecture with sufficient richness of features to solve our problem adequately. We have found that the best ANN solution to this problem is the single hidden layer, feed forward network, using radially-symmetric basis functions (RBFN) in the hidden layer.

So far, we have been discussing only feed-forward networks, so named because information flows only in the forward direction (toward the output). ANNs can be generalized to include feedback networks, which are useful for applications where the quality of the output estimate can be determined virtually immediately (e.g., in manipulating a robotic arm), and this information can be fed back to the input to refine continually the learning process. Our case is different since the equivalent test of our estimates would involve logging a new well. For our needs, feed-forward networks are well suited.

Choosing a set of basis functions is somewhat arbitrary, and therefore essentially a matter of preference. We choose a set called *radial basis functions,* or RBFs, which are Gaussian functions. The *jth* basis function has the form

$$f_j(\mathbf{x}) = e^{-(x-x_{0j})^2/\sigma_j^2}$$

where σ_j is a shape parameter for the width of the basis functions, and the numerator of the exponent is the negative of the square of Euclidean distance between the input variables, **x**, and the location of the center of the basis functions, $\mathbf{x}_{0j}$. Normalization of the input variables allows the existence of a single value of the width, σ_j , along any axis.

Since we are looking specifically for nonlinear functional relationships, we first estimate and subtract the best-fit linear relationship and use the neural network to estimate only the nonlinear residuals, i.e.,

$$P(\mathbf{x}) = F(\mathbf{x}) + r(\mathbf{x})$$

where *P* is the property to be estimated, *F* is the best-fit linear function through the learning set of points, and *r* contains the nonlinear component plus normal data scatter. The function *F* contains a linear term for each input attribute. For example, for two input attributes, *F* is actually a plane.

So, if *y* is the output of the ANN, we will train it to approximate the nonlinear component, *r*, for the learning set of input points:

$$y\,(\mathbf{x}_{\text{learning}}) \cong r\,(\mathbf{x}_{\text{learning}})\,.$$

$\mathbf{x}_{\text{learning}}$ are the seismic attribute values at the well-layer intersections, where the property, *P*, is known from the logs. So, with the neural network estimating only the nonlinear residual, *r*, the approximation to *P* is

$$P(\mathbf{x}) \cong F(\mathbf{x}) + y(\mathbf{x})\,.$$

This corresponds to the calibration step in the methodology discussed in Part 1, where we find a smooth curve fit to the attribute-property data points in the crossplot. *P(x)* is not yet the final property estimate; the residual correction and confidence estimation steps follow, as discussed in Part 1. Here we focus on the use of ANNS in the calibration step.

The output of the ANN in terms of its *n* basis functions is

$$y(\mathbf{x}) = \sum_{j=1}^{n} w_j \cdot f_j\,(\mathbf{x})$$

We chose these basis functions to be localized. (The σ parameter controls the "size" of each basis function, and $\mathbf{x}_0$ controls the location.) The target function, *r*, can be learned by specifying the *n*, σ_j, $\mathbf{x}_{0j}$, and w_j parameters. The local nature of the fit has some desirable features. One is that it makes the solution generated by the network clearly understandable, and it is a straightforward matter to relate the behavior of the fit to the learning set.

Another advantage of the localized basis functions is that a fast training procedure is available to determine the network parameters.

Training the network. The objective is to fit a nonlinear function to the sample data set. As this method is fully data-adaptive (apart from the choice of ANN), all network parameters must in some way be specified directly from the sample data (the "learning set"). The method about to be described is known as a hybrid scheme.

1) Number: We first need to decide on the number of nodes, *n*, in the hidden layer (i.e., the number of basis functions). This number is chosen empirically and is based on the number of *m* -dimensional sample points in the learning set (for our application, *m* is the number of attributes being used in the estimate, the number of logged wells is the number of sample points in the learning set). There should be fewer nodes than sample points in order for the system to be sufficiently overdetermined that a smooth curve is fit through the points.

A smooth fit is desirable for reasons related more to rock physics than to mathematics. If a relationship exists between input (attributes) and output (property), it will originate from some rock physics basis. So, when we determine a function to relate one data type to another, we want it to be smoothly varying, although possibly nonlinear, to reflect this presumed underlying rock physics relation. Due to normal data scatter, such a smoothly varying function will therefore not typically pass through all the data points in the learning set.

To help ensure that we are seeing the effect of a single lithology type on the seismic response, we prefer to estimate property distributions in a single geologic layer at a time, as opposed to mapping properties into a multilayered data volume in one step. By focusing on individual layers, the variation in lithology is kept to a minimum, helping to justify our search for a single attribute-property function. Searching for a single attribute-property relationship in a larger data volume (say, the volume represented by the entire 3-D seismic data set) would result in a confusion of multiple lithologies. In such a case, it would be inappropriate (and probably impossible) to find a representative single function to predict properties from attributes for the entire volume.

As it is, any geologic or seismic "layer" is composed of finer components, no matter how thin we define it to be. So it probably is not possible to make general rules for when an approach such as we are describing will work. We have to accept that each situation will be unique, and let the data dictate under what conditions and to what extent this approach will give accurate estimates and predictions. The composition of the layer may or may not be such that we can represent the attributes versus property as a functional relationship with any confidence.

2) Location: The next step in the learning procedure is to position each of our localized basis functions in the data space. That is, we need to specify the values of $\mathbf{x}_{0j}$. We use an iterative scheme, called K-means clustering, to locate the nodes.

To begin, the sample points in the learning set are distributed and assigned to nodes; the distribution is arbitrary, but the number assigned to each node should be about equal. The location of each node is determined by the average position, or mean vector, of the sample points assigned to it. An objective function is then computed as the sum of the absolute distances of all sample points to their respective node locations.

The system is then perturbed by reassignment of individual learning set data points to other nodes. If the objective function is lower, the new assignments and new node locations are kept. This perturbation step is repeated as desired until a final assignment or "clustering" of data points to nodes is achieved.

3) Width: The basis functions are now positioned in data space, but we still need to specify the σ parameter for each function to control the width of the nodes. The criterion for selection is that the basis functions adequately cover the range

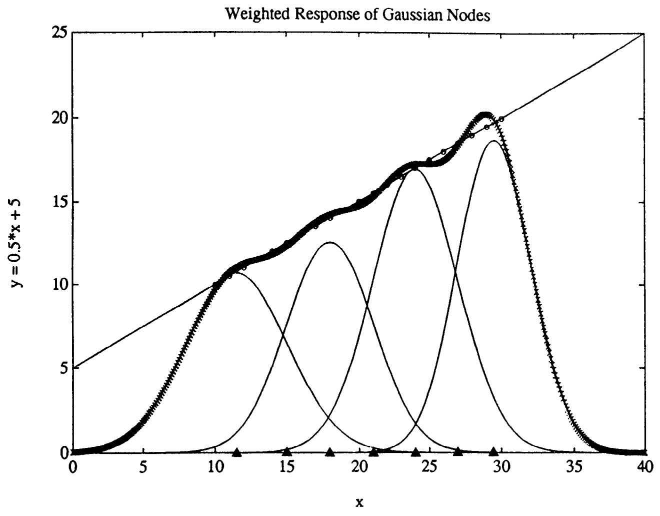

Figure 12. The result of using the RBFN to fit a set of 21 points, all lying in a straight line, to a curve. Bold line shows the ANN-computed fit, with the five basis functions plotted as well. Their placement, respective widths, and heights (weights) were determined using the learning procedure described in the text.

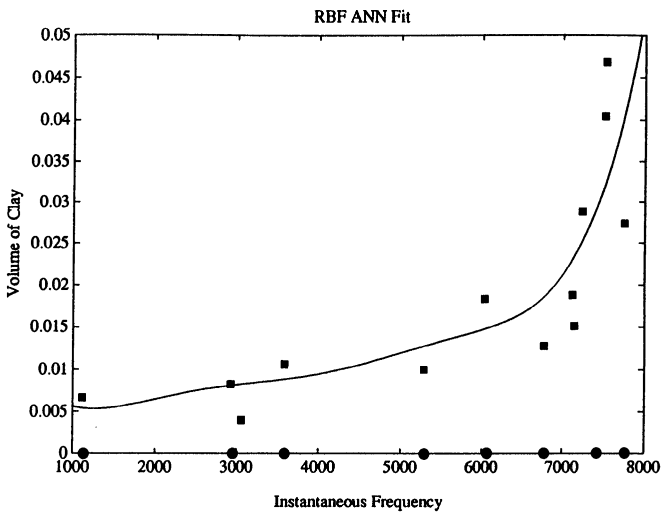

Figure 13. Nonlinear functional relationship determined by RBFN, using the learning technique described. Eight basis functions (x_0 values shown as closed circles on horizontal axis) were used to learn the nonlinear relationship from 13 of the original 15 sample points in the learning set comprised of the data points in Figure 9.

of attributes seen in the learning set.

We choose the σ_j parameters in a two-step process. Initially, each basis function is assigned a value of σ_j equal to the distance to the nearest neighbor function, multiplied by an overlap factor. This initial assignment is kept through the next stage of the learning process, where we estimate the weights. Then the widths are adjusted iteratively to minimize the approximation error.

Recall that we are making a local curve fit to the nonlinear component, *r*. For values of the input attributes well outside the range of the sample set (and therefore a large "distance" from $\mathbf{x}_{0j}$, the centers of the basis functions), the output of the ANN will taper to a zero value for the estimate of the nonlinear component, leaving only the linear component to make the estimate: $P(x) \simeq F(x)$. This behavior is a third beneficial consequence of choosing localized basis functions. At values of x well outside the range of attributes in the learning set, $y(x) \rightarrow \mathbf{0}$, and our curve fit will revert to linear. Consequently, when x is within or near the range of attribute values in the learning set, y(x) will be larger, and the curve fit will be influenced by linear and nonlinear components.

4) Weights: After the σ_j and $\mathbf{x}_{0j}$ parameters have been selected, the basis functions are completely described and all that remains is the final curve fit. The weights are computed by a least squares curve fit through the set of learning points, using singular value decomposition. Except for a brief return to the widths for optimization, the linear and nonlinear components of our estimation function are now complete.

The ANN is now completely specified and is ready to be used in the calibration step to estimate the spatial distribution of properties, according to $P(\mathbf{x}) \simeq F(\mathbf{x}) + y(\mathbf{x})$. Because it is feed forward, no information flows back to readjust the function parameters. Our ANN is static, and during the operation of our network we do nothing more than evaluate an algebraic expression for each set of input values. This expression is the sum of weighted functions that we have just specified.

There is no iterative learning possible, because no new information is coming to us. A new logged well qualifies as new information, but this occurrence is infrequent. When such new data become available, it usually makes more sense to reevaluate the entire ANN from the beginning, rather than to choose an adaptive (feedback) network, which has the disadvantage of modifying the function parameters away from the control of the interpreter.

Examples. Although we said that fitting a linear function was not a good way to use ANNS, we will do it for illustration purposes. In Figure 12, we show a fit to a learning set of 21 points, all of which lie exactly on the straight line y = 0.5x + 5.0 which is plotted passing through the points. The x values range from 10 to 30. The bold curve shows the fit of the ANN, the sum of five radial basis functions, below the fitted curve.

In this example, the location of the basis functions is the center, or midpoint, of each curve and is represented by the x_0 parameters. The width of the curves, controlled by the σs, can be seen in the varying widths of the basis functions. Finally, the weight of each function is represented by its height.

Notice that the approximation is good within the range of the input variables (between 10 and 30), but tapers to zero outside this range. It is outside the range of the learning data set that we want our fit to revert to linear, and this example clearly shows this behavior (recall that in a real situation, the ANN would be fitting the nonlinear component only).

The second example, Figure 13, shows the way that the RBFN has computed a nonlinear curve fit to the real data points of Figure 9. Our ANN has learned a nonlinear functional relationship between the attribute (instantaneous frequency) and the target property (volume fraction of clay), using a learning set comprised of attribute/property pairs from 13 of the original 15 wells, and fitting these data to eight basis. functions.

As a final example, we completed a mapping exercise with this procedure, using the nonlinear function in Figure 13. The volume fraction of clay was mapped, guided by instantaneous frequency. Figure 7 of Part 1 shows the spatial distribution resulting from this estimation.

Conclusions, ANNs are one way of "learning" and evaluating a nonlinear functional relationship between measured variables in the calibration step of seismic-guided log property estimation. They are not the only way to do it, but they are effective and are not particularly expensive to implement if one sticks to the feed forward variety and uses a sensible learning scheme.

Of the two types of ANNs, feed forward and feedback, we feel the more open feed forward architecture is best for the mapping application because it lets the interpreter play an active role in specifying the network parameters. Furthermore, in the particular case of estimating the spatial distribution of log properties from seismic attributes, new data (in the form of newly logged wells) appears infrequently enough that it makes more sense to recompute the entire network rather than letting a feedback scheme refine the network with the new data. We also feel that using localized basis functions is a good choice since it forces the network to exhibit controlled behavior outside the learning set.

In using any statistical scheme to predict the spatial distribution of rock or reservoir properties, there is one important point to remember-a method such as this one for seismic-guided estimation of log property distributions can often work well *but may not always do so* because, even though all apparent relationships have an underlying rock physics basis, these techniques are inherently data-driven. Quality of acquired data, uniformity, consistency, quality of the data processing, and the manner in which the geologic layers respond to the seismic signals are some factors (among many others) that will affect how well a data-driven sceme will work.

Still to come. In Part 3, we will show examples of studies in fields where we estimated the spatial distribution of log properties in layers from seismic attributes. A controlled study will show how we can normally expect a better result from this approach, as compared to mapping properties from log data (where seismic is used only for structural definition). LE