CHAPTER 5. TECHNOLOGY TO MANAGE LARGE DIGITAL DATASETS

Geoffrey C. Bohling

Introduction

Development of the Hugoton field model has required automated processing of large data volumes at several steps, including prediction of lithofacies from geophysical well logs in numerous wells based on a neural network trained on log-facies associations observed in cored wells, generation of geologic controlling variables (depositional environment indicator and relative position in cycle) from a tops dataset, and computation of porosities corrected for mineralogical variations between facies and for washouts. In addition, we have developed code for batch processing the predicted facies and corrected porosities at the wells to estimate water saturations and original gas in place using petrophysical transforms and height above free-water level, providing a quickly computed measure of the plausibility of the geomodel. This chapter describes the data management and facies-prediction tools developed for this project, which we have provided in the form of two Excel workbooks and an Excel add-in. Having automation tools allowed us to efficiently handle large volumes of data and the ability to perform the multiple iterations required to test preliminary and intermediate algorithms and solutions.

Facies-prediction Tools

Prediction of facies from wire-line logs in 1600 “node” wells distributed throughout the field was a critical step in developing the Hugoton field model. The primary tool used in this process was a single hidden layer network (Hastie et al., 2001) designed for prediction of a categorical output variable (facies) from a set of continuous input variables (logs), as illustrated in Figure 5.1. We have implemented the neural network in Excel, adding the neural network training and prediction code to a previously existing Excel add-in for nonparametric regression and classification, Kipling.xla (Bohling and Doveton, 2000). The new version of the add-in, Kipling2.xla, includes code for batch prediction of facies (or of any other categorical or continuous variable) based on logs read from a set of LAS files, with prediction results being written out as curves in a corresponding set of output LAS files. The computationally intensive task of determining the optimal neural network parameters based on crossvalidation of the training dataset, from wells with both core and log data, was performed outside Excel, using scripts written in the R statistical language.

Prior to performing facies prediction, we added two geologic constraining variables to the set of log values: a code representing the depositional environment (1 - nonmarine, 2 - marine, or 3 - intertidal) associated with each of the 25 members (half-cycles) in the model, and a curve representing the relative vertical position within each half-cycle, ranging from 0 at the base to 1 at the top. The depositional environment indicator variable, MnM, helps to distinguish between lithofacies with similar petrophysical properties but developed in different broad depositional environments. Including the relative position curve, RelPos, allowed the network to encode information regarding the fairly regular succession of lithofacies succession commonly exhibited within each interval, and thus transfer some of that character to the sequence of predicted lithofacies in each well. The two curves were computed from a database of formation tops using Visual Basic code within an Excel spreadsheet. They were then combined with the wire-line log curves to complete the feature vector used as input to the neural network.

Depositional Environment Code and Relative Position Curve

The depositional environment code (MnM) and relative position (RelPos) curves are generated by Visual Basic code in an Excel workbook named GenMnM.xls. Figure 5.2 shows the layout of an input spreadsheet for the workbook. The fundamental input information is a set of formation names or labels, listed in order of increasing depth in row 5, and a set of top depths for each formation by well, with wells identified by API number (or some other unique identifier) in column A, starting from row 9, and top depths for each well listed to the right of the API number, each in the appropriate column for the formation. The data shown in Figure 5.2 have been extracted from the Kansas Geological Survey's database of Hugoton top depths, imported into Excel from Oracle. However, the code is completely general and will work with any set of tops. The spreadsheet in Figure 5.2 contains additional columns of data to the right of those shown in the figure, with tops continuing down through the Council Grove.

The code will generate a set of LAS files containing the two geologic constraining variables, and cell B2 on the spreadsheet contains the name of the folder to which these files should be written. Cell B3 specifies the depth increment (here 0.5 feet) at which the curves should be written. Row 6 contains the depositional environment code numbers associated with each formation. The numbers shown here are those appropriate for the current study, but any set of code numbers could be employed. Row 7 contains a list of "default" thicknesses for each formation. The default thicknesses come into play when there is a gap in the tops sequence for a well. In this case, information for the last formation above the gap is written out on the basis of the default thickness for that formation, with null values being written out below the depth given by the top depth plus the default thickness (unless the next top depth below the gap is encountered first).

Once the input information is set up on the spreadsheet, clicking on the **Generate MnM Files** button will launch the Visual Basic code, generating a set of output LAS files in the specified output folder. This folder should be created in advance of running the code, using Windows Explorer. Each output file is named MnM*nnnn*.las, where *nnnn* is replaced by the API number listed in the spreadsheet. Figure 5.3 shows the headers and first few data lines of the LAS file generated from the first line of data shown in Figure 5.2.

Neural Network Prediction of Lithofacies

As discussed in Chapter 6, we explored several approaches for predicting lithofacies from logs and ended up choosing to use a neural network. The neural network implemented in Kipling2.xla is a simple single hidden-layer feed-forward network, as illustrated in Figure 5.1 and described in Chapter 11 of Hastie et al. (2001). Each hidden layer node in the network computes a linear combination or weighted sum of the input variables (logs), representing variation along a certain direction in the input variable space. The lines connecting the input and hidden layer nodes represent the weights in this linear combination. This weighted sum is then passed through a sigmoid transfer function, with output values ranging between zero and one, forming an S-shaped basis function along the direction represented by the linear combination. Each output layer node then computes a linear combination of the sigmoid basis function values computed by the hidden layer nodes and the entire set of linear combinations computed by the output nodes is rescaled to represent a set of probabilities of facies membership. Thus, on a single forward pass, the network transforms a set of log values into an associated set of probabilities of facies occurrence.

The network is trained based on a set of training data where both log values and facies are known, in this case data from the 28 cores described in the field. In the training process, the network weights are iteratively adjusted until the set of facies predicted by the network match the set of core facies as closely as possible. However, the training process starts from a random set of initial weights and will generally converge to a different set of final weights for a different set of initial weights, meaning that there can be multiple, equally likely “realizations” of the trained neural network based on a single training dataset.

The primary parameters controlling the behavior of the neural network are the number of hidden layer nodes and a damping or decay parameter. Increasing the number of hidden layer nodes allows the network to more closely reproduce the details of the training dataset, while fewer hidden layer nodes results in more generalized representation of the training data. Increasing the damping parameter forces the network weights to be smaller in magnitude, which results in a smoother or more generalized representation of the training data. Using a very large number of hidden-layer nodes and no or very little damping would allow the network to reproduce the training dataset very accurately, but this would almost certainly result in reduced accuracy in predictions on other datasets, a situation referred to as overtraining. Thus, the number of hidden-layer nodes (network size) and damping parameter should be adjusted to produce the optimal balance between detail and generalization. This optimization is usually achieved through crossvalidation, holding out different portions of the full training dataset from the training process, predicting on the withheld data and comparing predicted and true facies, and repeating the process many times over in order to obtain representative summaries of the prediction behavior for different parameter combinations. This process is much too computationally intensive to perform in Excel, so for this study we have used R language scripts to perform the crossvalidation, using R’s *nnet* function (Venables and Ripley, 1999), which implements the same style of neural network as in Kipling2.xla. The details and results of the crossvalidation process for this study are described in Chapter 6.

For training a neural network in Kipling2.xla, the training data, including logs and known facies values from a set of wells, must be gathered together in one Excel spreadsheet, with logs and facies values, coded as a set of integers, in columns and data points – that is, the set of data for each measurement depth in each well – in rows, with a row of column labels at the top. Figure 5.4 shows the dialog box for selecting the predictor variables (logs) and categorical response variable (the variable L10, representing facies codes) from a training data worksheet. The user has also entered a comment which will be transferred to the output neural network worksheet as a reminder of what the worksheet represents. Figure 5.5 shows the dialog box for specifying the neural network parameters to use in the training process. In addition to the number of hidden-layer nodes (network size) and damping parameter, the user can also specify the number of iterations in the training process. More iterations will allow a closer reproduction of the training data. We have generally left the number of iterations at 100, controlling the network behavior through the network size and damping parameter. The values of these two parameters shown in Figure 5.5, 20 and 1.0, were the optimal values determined by the crossvalidation process described in Chapter 6 (in most cases).

Once the training process starts, the code will generate a new worksheet and starting writing out the value of the objective function, a measure of the discrepancy between predicted and actual facies values over the entire training dataset, versus iteration number. The objective function will decrease with each iteration, more rapidly at first and gradually leveling off, indicating a point of diminishing return in further adjustments of the network weights. After the training process has completed, the user may use standard Excel plotting functions to generate a plot of objective function versus iteration number, as shown in Figure 5.6.

After training is completed, the new worksheet will be populated with a set of numbers representing the weights in the trained neural network, along with summary statistics for the predictor variables (logs) in the training dataset. These numbers are not really intended for human consumption; they just need to be stored somewhere so that they can be used for prediction. The trained network can then be used for predicting facies from log values stored in an Excel worksheet, with results being written to a new worksheet, or predicting facies in batch based on logs in a set of LAS files, with results being written out to a new set of LAS files.

For the batch-prediction process, the user needs to specify the correspondence between the logs used in the training process – the model variables – and the logs included in the LAS files. The batch-prediction code first presents the user with a standard dialog box for opening a file, asking the user to navigate to the folder containing the LAS files and open one of these files. The set of log mnemonics listed in the curve information section in this file is assumed to represent the set of mnemonics for the entire set of files, at least for the logs used in the prediction process. In other words, aliasing of logs to a common set of mnemonics has to have been done in advance. Figure 5.7 shows the dialog box for matching LAS file log mnemonics (Logs in file) to the log names used in the training process and stored in the neural network worksheet (Model variables). Figure 5.8 shows a portion of an output LAS file produced by the batch prediction process, including the curve information section of the headers and a few of the output data lines. The output curves include the predicted facies code at each depth, here labeled PredLL8, the probability of occurrence of each facies at each depth, here labeled ProbPE1 – ProbPE8. The predicted facies code is that associated with the largest occurrence probability.

This batch-prediction process has been critical to the development of the Hugoton field model, allowing rapid production of facies curves from a large number of input LAS files.

Porosity Correction

The porosity correction code performs two tasks, first compensating for variations from the reference mineralogy for the neutron and density porosity logs and then applying an ad-hoc process to correct for washouts. The compensation for mineralogical variations employs the lithofacies-specific regression relationships between core and log porosity measurements described in Section 4.3. The correction for washouts removes porosity spikes identified as washouts on the basis of lithofacies-specific threshold values of the neutron-density crossplot porosity, and also levels out the "shoulders" leading up to these spikes, as illustrated in Figure 5.9. The user specifies both the threshold crossplot porosity value used to identify washouts and a replacement porosity value to use as the corrected porosity over the region identified as washout plus the shoulders leading up to the peak, typically an expected mean porosity value for the lithology. Note that Figure 5.9 is a little deceiving in that it treats the crossplot and corrected porosity curves as if they were one and the same; in reality, the washout-and-shoulder identification is done on the basis of the crossplot porosity, but the substitution is performed on the corrected porosity curve.

The porosity-correction code is contained in a workbook named BatchCurveProc.xls, which is included in the appendix, along with documentation. This workbook also contains the code for batch computation of original gas in place and estimation of free water level, described in the next section. In the input worksheet for porosity correction, illustrated in Figure 5.10, the user specifies the folder containing the input LAS files and the names of the lithology, neutron porosity, density porosity, and crossplot porosity curves in those files, along with the name of an output folder to contain the LAS files with corrected porosity curves. The table to the right contains the set of coefficients for the regressions of core porosity versus neutron and density porosity, along with the crossplot porosity (PhiND) threshold values for identifying washouts in each lithology and the replacement values for the corrected porosity in each lithology. After the user modifies input values on this spreadsheet as appropriate and clicks the **Run!** button, the code will generate a set of LAS files containing corrected porosity curves, in the specified output file.

Water Saturation, OGIP and Free Water Level Estimation

The BatchCurveProc.xls workbook also contains code for computing water and gas saturation at half-foot intervals on a well-by-well basis based on curves of lithology and corrected porosity together with height above the free water level, using the porosity-saturation transform relationships described in Section 4.2. The input worksheet layout is fairly similar to that used for the porosity correction code, with a table of parameters for the porosity-saturation transforms taking the place of the table of regression coefficients. The input LAS curves are the lithology curve and the corrected porosity curve. The free water level elevation must also be specified for each well, for the code to be able to compute height above free water, and this elevation may be specified as a single value in the spreadsheet, or read from the LAS file headers on a well-by-well basis. The output LAS files contain curves of water saturation and incremental OGIP at half-foot intervals.

In addition to the OGIP calculation code, BatchCurveProc.xls also contains code for estimating the free water level in each well from a comparison of the volumetrically computed OGIP (from the porosity-saturation transforms) and an independent estimate of OGIP, most likely a mass balance estimate. In this case, the user specifies a minimum allowed free water level to use for all wells and, at each well, the code uses this minimum allowed level as the starting point for an iterative search for a free water level that produces a volumetric OGIP matching the, mass balance estimate. The code uses a simple bracketing search (Press et al., 1992), illustrated in Figure 5.11. Although this code contributed to the process of estimating free water level variations in this study, the final free water level estimates were based primarily on elevations of the lowest productive perforated interval in each well, as described in Section 7.2.

References

Bohling, G. C., and J. H. Doveton, 2000, Kipling.xla: An Excel Add-in for Nonparametric Regression and Classification: Kansas Geological Survey, http://www.kgs.ku.edu/software/Kipling/Kipling1.html. (accessed June 15, 2006)

Hastie, T., R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2001, Springer-Verlag, New York, 533 p.

Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, 1992, Numerical Recipes in Fortran, The Art of Scientific Computing, Second Edition: Cambridge University Press, 963 p.

Venables, W.N., and B.D. Ripley, 1999, Modern Applied Statistics with S-Plus: Third Edition, Springer-Verlag, New York, 501 p.

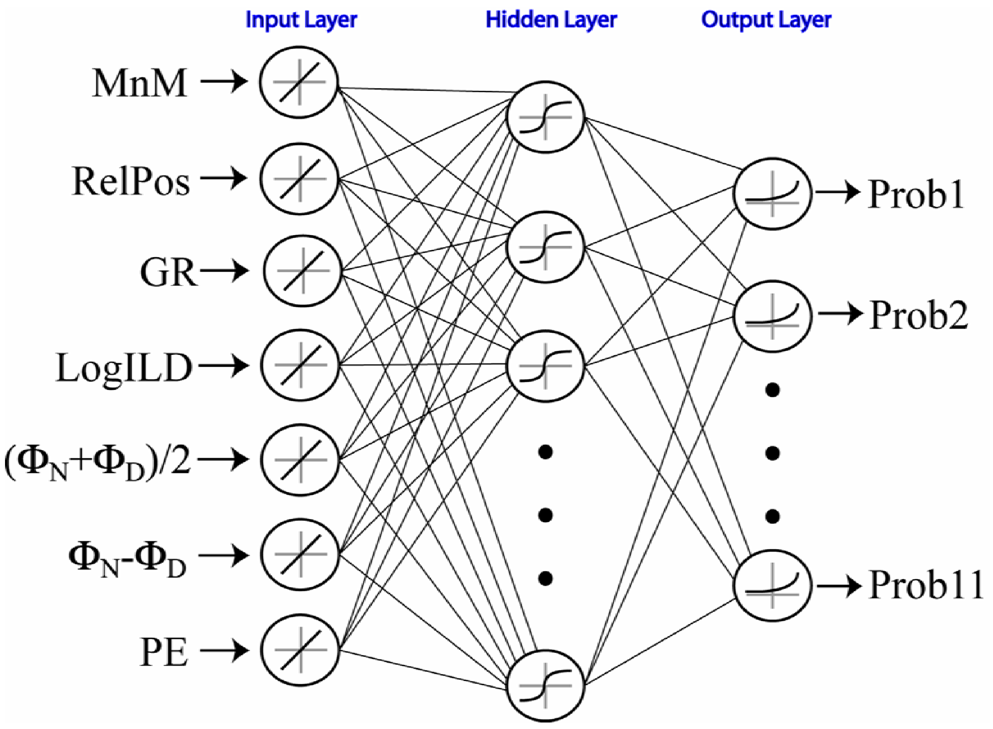
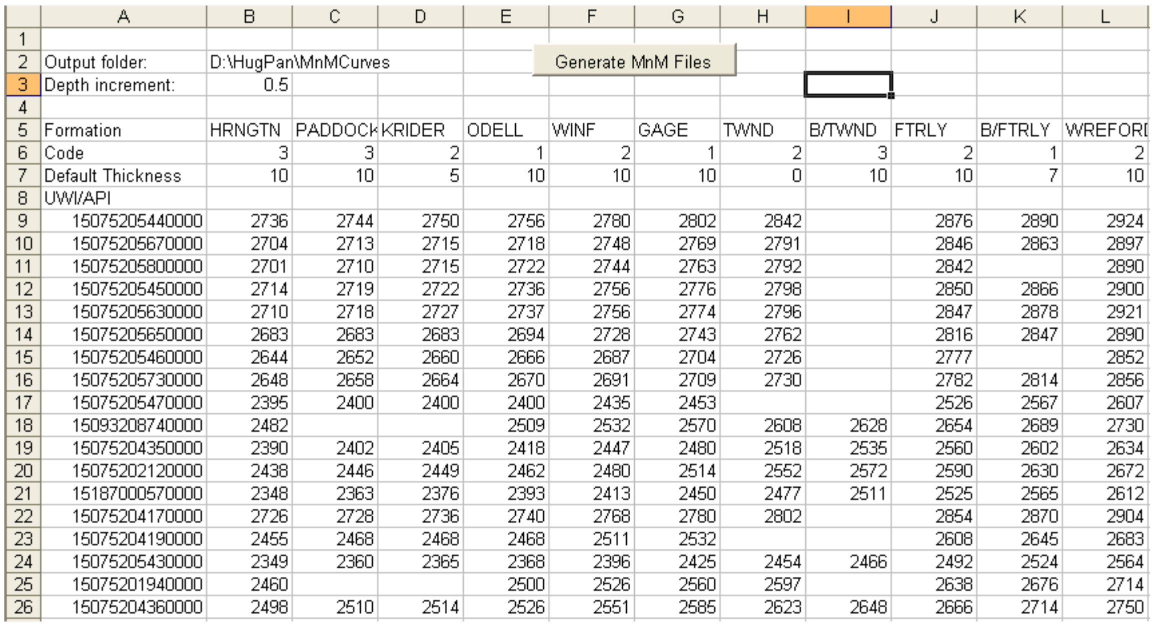
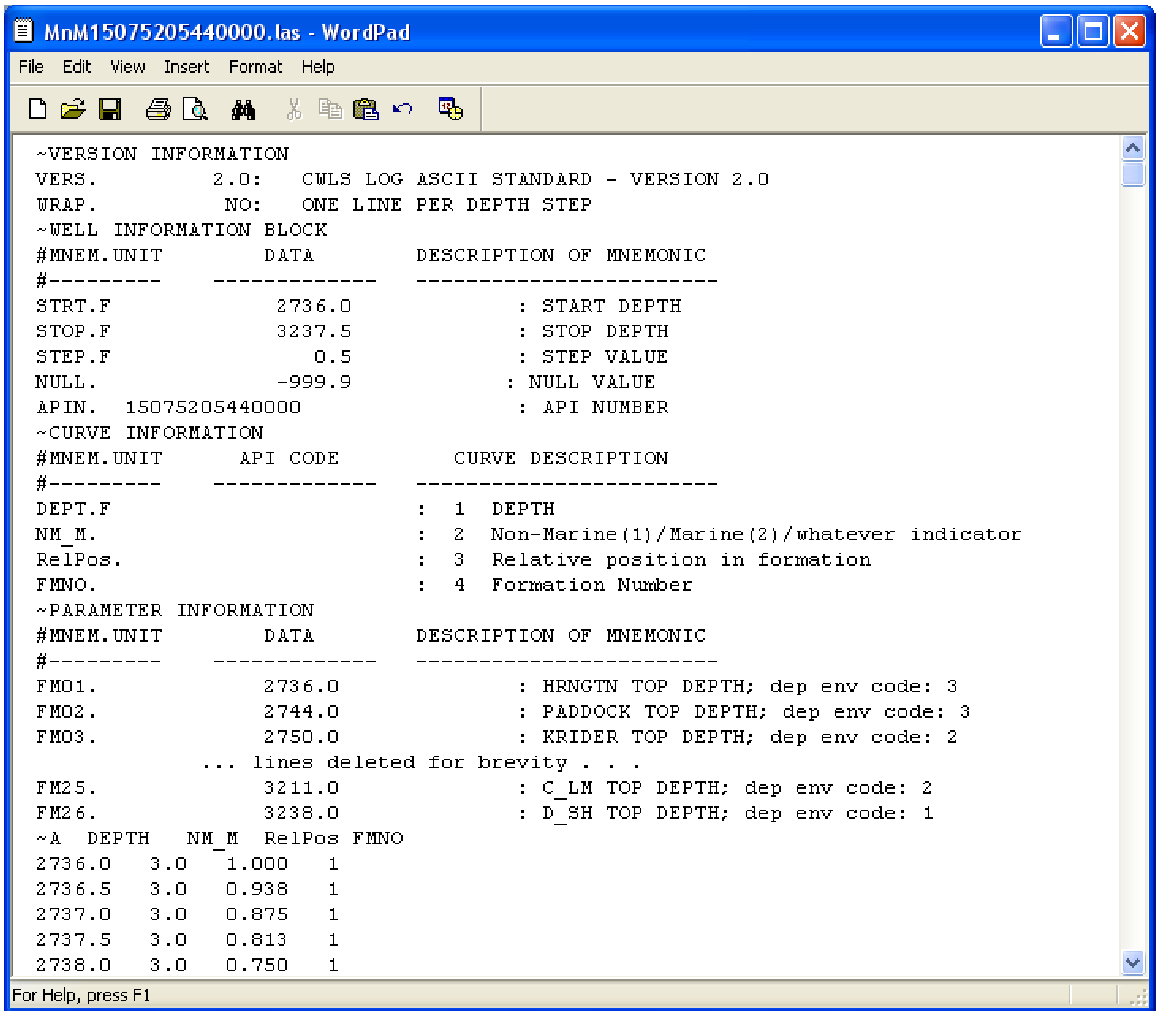


Figure 5.1. Neural network employed for prediction of lithofacies. Output values are probabilities of membership in different lithofacies.



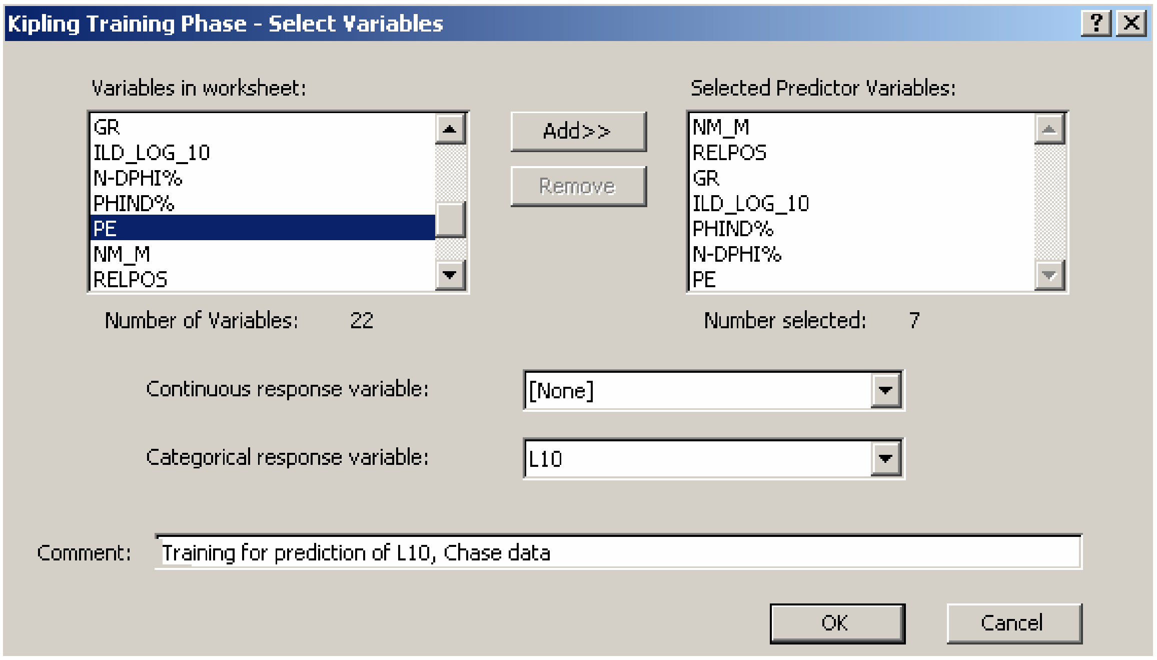
	A	B	C	D	E	F	G	H	I	J	K	L
1												
2	Output folder:	D:\HugPan\MnMCurves				Generate MnM Files						
3	Depth increment:	0.5										
4												
5	Formation	HRNGTN	PADDOC	KRIDER	ODELL	WINF	GAGE	TWND	B/TWND	FTRLY	B/FTRLY	WREFOR
6	Code	3	3	2	1	2	1	2	3	2	1	2
7	Default Thickness	10	10	5	10	10	10	0	10	10	7	10
8	UWI/API											
9	15075205440000	2736	2744	2750	2756	2780	2802	2842		2876	2890	2924
10	15075205670000	2704	2713	2715	2718	2748	2769	2791		2846	2863	2897
11	15075205800000	2701	2710	2715	2722	2744	2763	2792		2842		2890
12	15075205450000	2714	2719	2722	2736	2756	2776	2798		2850	2866	2900
13	15075205630000	2710	2718	2727	2737	2756	2774	2796		2847	2878	2921
14	15075205650000	2683	2683	2683	2694	2728	2743	2762		2816	2847	2890
15	15075205460000	2644	2652	2660	2666	2687	2704	2726		2777		2852
16	15075205730000	2648	2658	2664	2670	2691	2709	2730		2782	2814	2856
17	15075205470000	2395	2400	2400	2400	2435	2453			2526	2567	2607
18	15093208740000	2482			2509	2532	2570	2608	2628	2654	2689	2730
19	15075204350000	2390	2402	2405	2418	2447	2480	2518	2535	2560	2602	2634
20	15075202120000	2438	2446	2449	2462	2480	2514	2552	2572	2590	2630	2672
21	15187000570000	2348	2363	2376	2393	2413	2450	2477	2511	2525	2565	2612
22	15075204170000	2726	2728	2736	2740	2768	2780	2802		2854	2870	2904
23	15075204190000	2455	2468	2468	2468	2511	2532			2608	2645	2683
24	15075205430000	2349	2360	2365	2368	2396	2425	2454	2466	2492	2524	2564
25	15075201940000	2460			2500	2526	2560	2597		2638	2676	2714
26	15075204360000	2498	2510	2514	2526	2551	2585	2623	2648	2666	2714	2750

Figure 5.2. Input tops data worksheet for generation of depositional environment and relative position curves.


MnM15075205440000.las - WordPad
File Edit View Insert Format Help
For Help, press F1

```
~VERSION INFORMATION
VERS.          2.0:   CWLS LOG ASCII STANDARD - VERSION 2.0
WRAP.           NO:   ONE LINE PER DEPTH STEP
~WELL INFORMATION BLOCK
#MNEM.UNIT        DATA        DESCRIPTION OF MNEMONIC
#---------    -------------   ------------------------
STRT.F             2736.0             : START DEPTH
STOP.F             3237.5             : STOP DEPTH
STEP.F                0.5             : STEP VALUE
NULL.              -999.9            : NULL VALUE
APIN.  15075205440000                 : API NUMBER
~CURVE INFORMATION
#MNEM.UNIT      API CODE         CURVE DESCRIPTION
#---------    -------------   ------------------------
DEPT.F                        :  1  DEPTH
NM_M.                         :  2  Non-Marine(1)/Marine(2)/whatever indicator
RelPos.                       :  3  Relative position in formation
FMNO.                         :  4  Formation Number
~PARAMETER INFORMATION
#MNEM.UNIT        DATA        DESCRIPTION OF MNEMONIC
#---------    -------------   ------------------------
FM01.             2736.0              : HRNGTN TOP DEPTH; dep env code: 3
FM02.             2744.0              : PADDOCK TOP DEPTH; dep env code: 3
FM03.             2750.0              : KRIDER TOP DEPTH; dep env code: 2
             ... lines deleted for brevity . . .
FM25.             3211.0              : C_LM TOP DEPTH; dep env code: 2
FM26.             3238.0              : D_SH TOP DEPTH; dep env code: 1
~A  DEPTH   NM_M  RelPos FMNO
2736.0   3.0   1.000   1
2736.5   3.0   0.938   1
2737.0   3.0   0.875   1
2737.5   3.0   0.813   1
2738.0   3.0   0.750   1
```

Figure 5.3. Portion of an output LAS file containing depositional environment code (NM_M) and relative position (RelPos) curves.



Variables in worksheet:

GR
ILD_LOG_10
N-DPHI%
PHIND%
PE
NM_M
RELPOS

Number of Variables: 22

Selected Predictor Variables:

NM_M
RELPOS
GR
ILD_LOG_10
PHIND%
N-DPHI%
PE

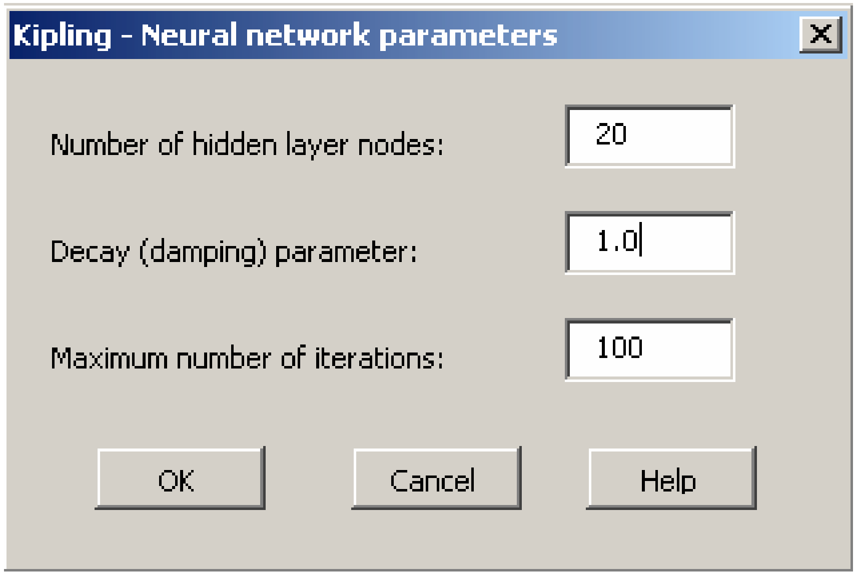
Number selected: 7

Continuous response variable: [None]

Categorical response variable: L10

Comment: Training for prediction of L10, Chase data

Figure 5.4. Training variable selection dialog box in Kipling2.xla.



Number of hidden layer nodes: 20

Decay (damping) parameter: 1.0

Maximum number of iterations: 100

Figure 5.5. Neural network parameter dialog box in Kipling2.xla.

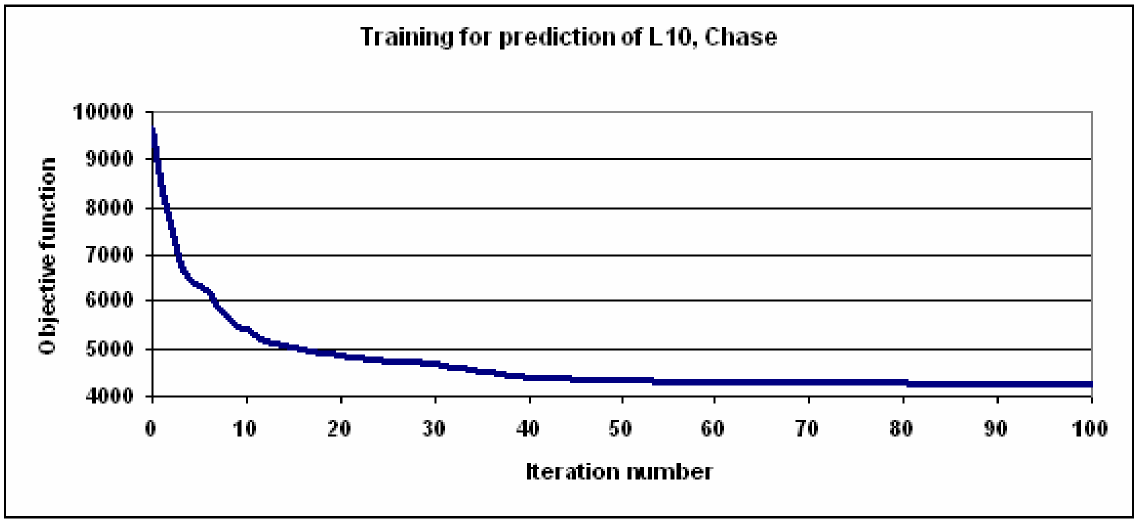
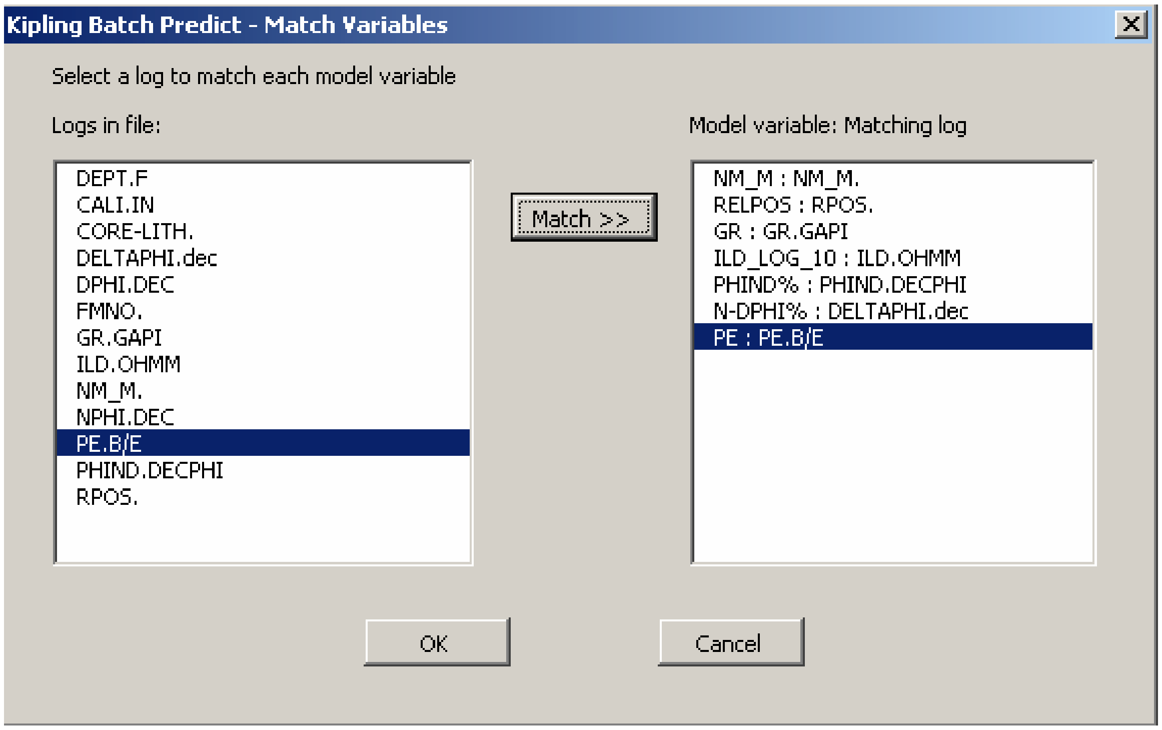


Figure 5.6. Example plot of objective function, measuring mismatch between true and predicted facies, versus iteration of the neural network training process.


Kipling Batch Predict - Match Variables

Select a log to match each model variable

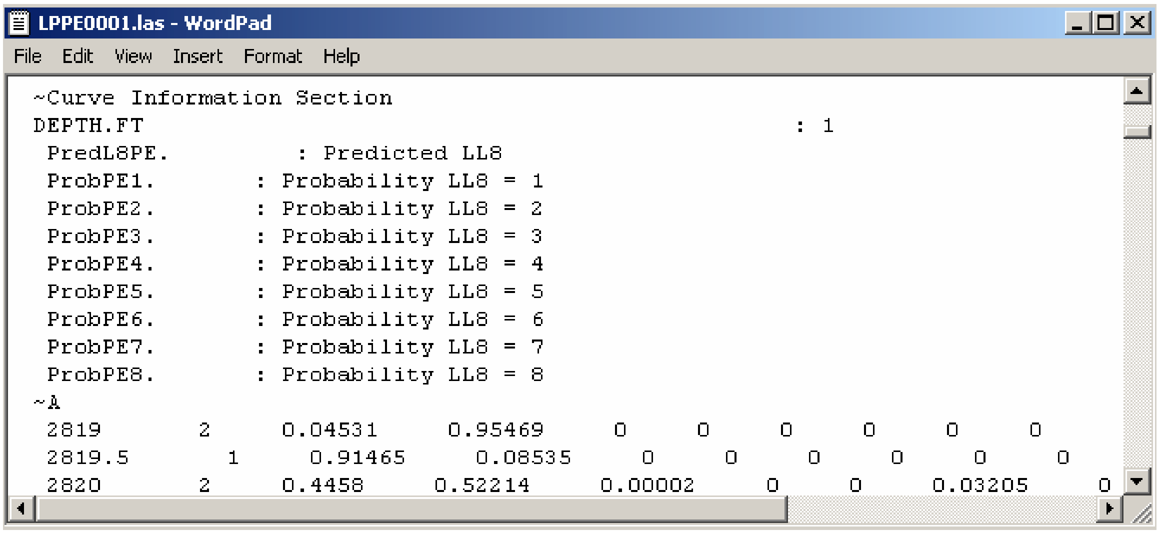
Logs in file:

DEPT.F
CALI.IN
CORE-LITH.
DELTAPHI.dec
DPHI.DEC
FMNO.
GR.GAPI
ILD.OHMM
NM_M.
NPHI.DEC
PE.B/E
PHIND.DECPHI
RPOS.

Model variable: Matching log

NM_M : NM_M.
RELPOS : RPOS.
GR : GR.GAPI
ILD_LOG_10 : ILD.OHMM
PHIND% : PHIND.DECPHI
N-DPHI% : DELTAPHI.dec
PE : PE.B/E

Figure 5.7. Dialog box for matching LAS log mnemonics to log names in trained neural network (model variables) for batch facies prediction process in Kipling2.xla.



```
~Curve Information Section
DEPTH.FT                                                        : 1
 PredL8PE.         : Predicted LL8
 ProbPE1.       : Probability LL8 = 1
 ProbPE2.       : Probability LL8 = 2
 ProbPE3.       : Probability LL8 = 3
 ProbPE4.       : Probability LL8 = 4
 ProbPE5.       : Probability LL8 = 5
 ProbPE6.       : Probability LL8 = 6
 ProbPE7.       : Probability LL8 = 7
 ProbPE8.       : Probability LL8 = 8
~A
 2819        2     0.04531     0.95469     0     0     0     0     0     0
 2819.5        1     0.91465     0.08535     0     0     0     0     0     0
 2820        2     0.4458     0.52214     0.00002     0     0     0.03205     0
```

Figure 5.8. Portion of an output LAS file produced by the batch prediction process, including curves of predicted facies (here labeled PredL8PE) and facies membership probabilities.

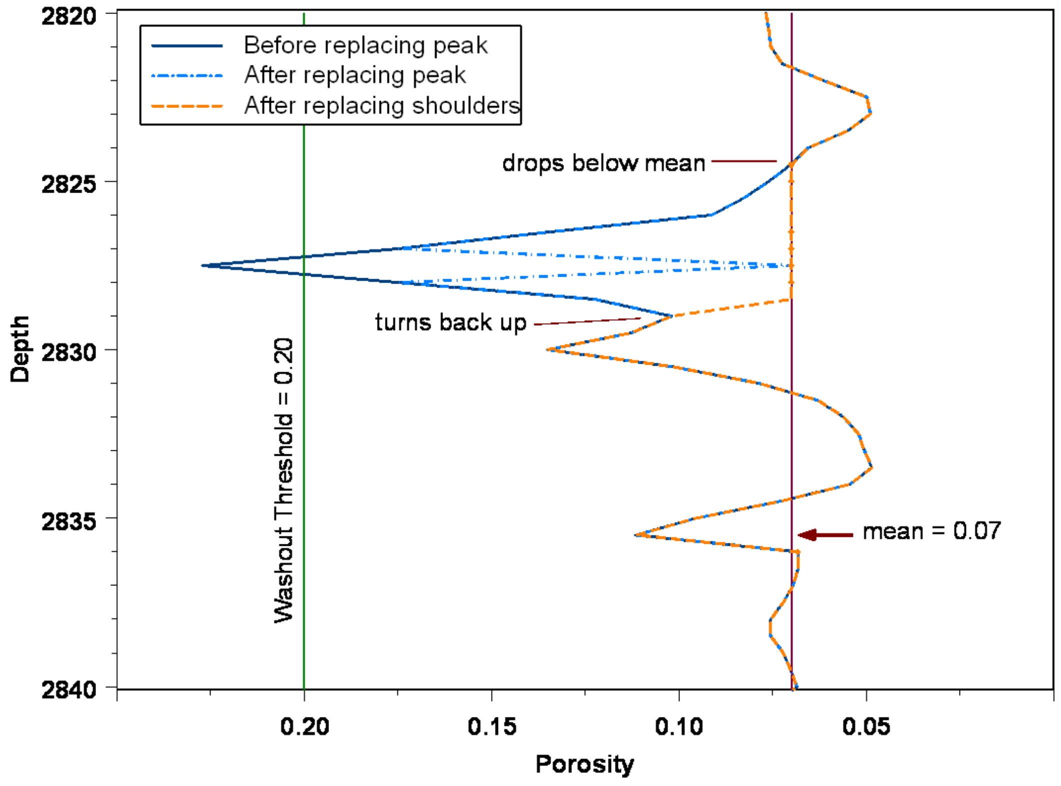
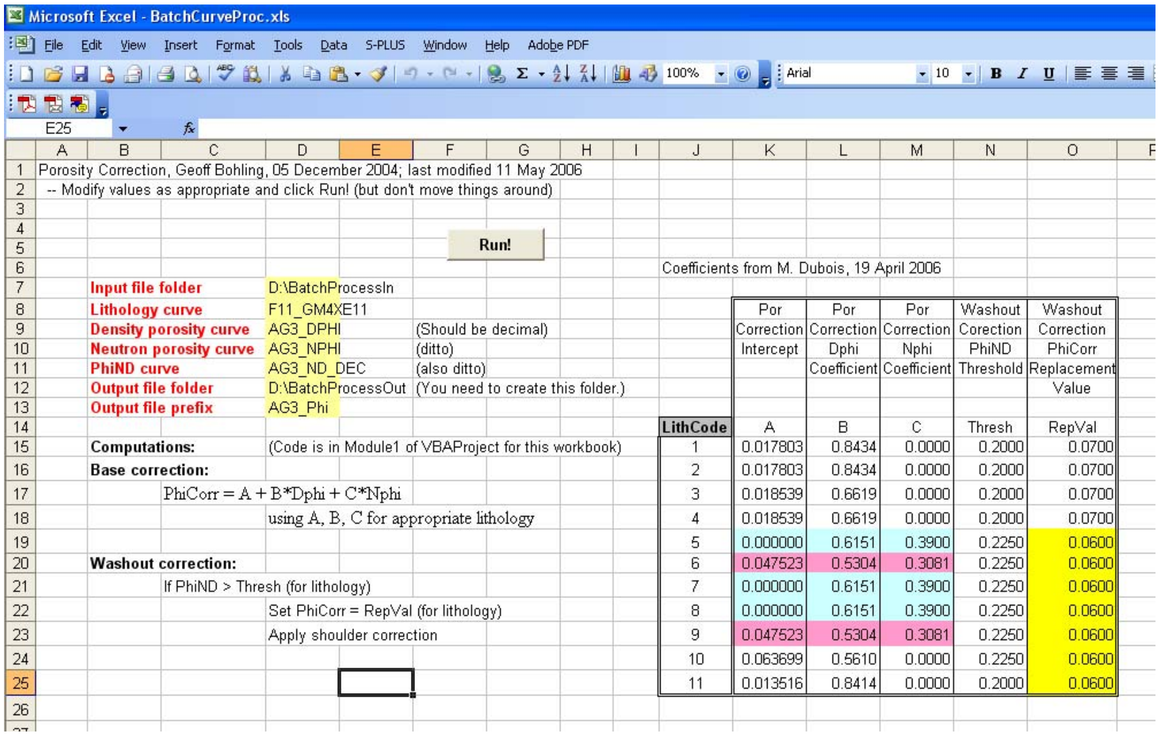


Figure 5.9. Schematic illustration of peak-and-shoulder removal process for washout correction. Depth is in feet and porosity is in decimal values.


Microsoft Excel - BatchCurveProc.xls
File Edit View Insert Format Tools Data S-PLUS Window Help Adobe PDF
Run!

Porosity Correction, Geoff Bohling, 05 December 2004; last modified 11 May 2006
-- Modify values as appropriate and click Run! (but don't move things around)

Input file folder	D:\BatchProcessIn	
Lithology curve	F11_GM4XE11	
Density porosity curve	AG3_DPHI	(Should be decimal)
Neutron porosity curve	AG3_NPHI	(ditto)
PhiND curve	AG3_ND_DEC	(also ditto)
Output file folder	D:\BatchProcessOut	(You need to create this folder.)
Output file prefix	AG3_Phi	

Computations: (Code is in Module1 of VBAProject for this workbook)

Base correction:

PhiCorr = A + B*Dphi + C*Nphi

using A, B, C for appropriate lithology

Washout correction:

If PhiND > Thresh (for lithology)

Set PhiCorr = RepVal (for lithology)

Apply shoulder correction

Coefficients from M. Dubois, 19 April 2006

	Por Correction Intercept	Por Correction Dphi Coefficient	Por Correction Nphi Coefficient	Washout Corection PhiND Threshold	Washout Correction PhiCorr Replacement Value
LithCode	A	B	C	Thresh	RepVal
1	0.017803	0.8434	0.0000	0.2000	0.0700
2	0.017803	0.8434	0.0000	0.2000	0.0700
3	0.018539	0.6619	0.0000	0.2000	0.0700
4	0.018539	0.6619	0.0000	0.2000	0.0700
5	0.000000	0.6151	0.3900	0.2250	0.0600
6	0.047523	0.5304	0.3081	0.2250	0.0600
7	0.000000	0.6151	0.3900	0.2250	0.0600
8	0.000000	0.6151	0.3900	0.2250	0.0600
9	0.047523	0.5304	0.3081	0.2250	0.0600
10	0.063699	0.5610	0.0000	0.2250	0.0600
11	0.013516	0.8414	0.0000	0.2000	0.0600

Figure 5.10. Input spreadsheet for porosity correction code.

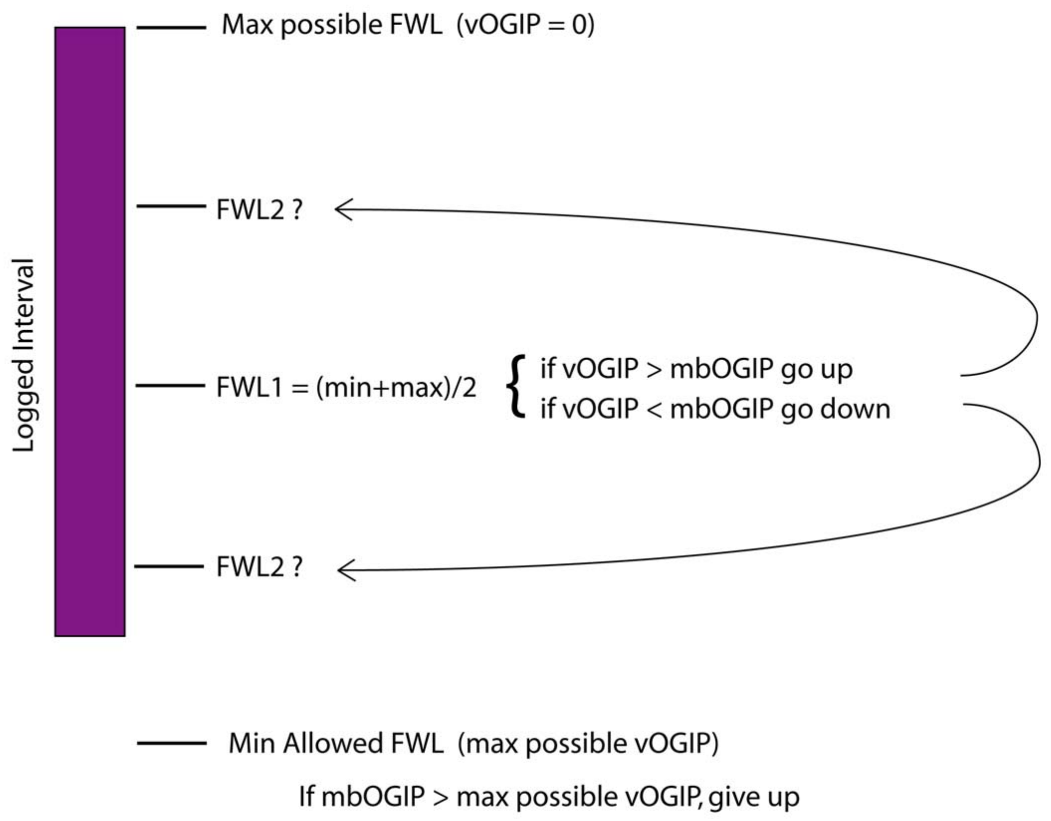

Max possible FWL (vOGIP = 0)
Logged Interval
FWL2 ?
FWL1 = (min+max)/2
if vOGIP > mbOGIP go up
if vOGIP < mbOGIP go down
FWL2 ?
Min Allowed FWL (max possible vOGIP)
If mbOGIP > max possible vOGIP, give up

Figure 5.11. Schematic representation of bracketing search to find free water level elevation to produce volumetric OGIP (vOGIP) matching a specified mass balance OGIP (mbOGIP).