Data-Driven In-Situ Sonic-Log Synthesis in Shale Reservoirs for Geomechanical Characterization

Jiabo He, University of Melbourne, and **Hao Li** and **Siddharth Misra*,** University of Oklahoma

Summary

Compressional-travel-time (DTC) and shear-travel-time (DTS) logs acquired using sonic-logging tools are crucial for subsurface geomechanical characterization. In this study, 13 "easy-to-acquire" conventional logs were processed using six shallow-regression-type supervised-learning models—namely, ordinary least squares (OLS), partial least squares (PLS), ElasticNet (EN), least absolute shrinkage and selection operator (LASSO), multivariate adaptive regression splines (MARS), and artificial-neural network (ANN)—to successfully synthesize DTC and DTS logs. Among the six models, ANN outperforms other models with R^2 of 0.87 and 0.85 for the syntheses of DTC and DTS logs, respectively. The six shallow-learning models are trained and tested with 8,481 data points acquired from a 4,240-ft-depth interval of a shale reservoir in Well 1, and the trained models are deployed in Well 2 for purposes of blind testing against 2,920 data points from 1,460-ft-depth interval. After that, five clustering algorithms are applied on subsamples of 13 "easy-to-acquire" logs to identify clusters and compare them with the log-synthesis performance of the shallow-learning models. A dimensionality-reduction algorithm, t-distributed stochastic neighbor embedding (t-SNE), is used to visualize the petrophysical/statistical characteristics of the clustering algorithm. Hierarchical-clustering, density-based spatial clustering of application with noise (DBSCAN), and self-organizing-map (SOM) algorithms are sensitive to outliers and did not effectively differentiate the input data into consistent clusters. A Gaussian-mixture model can differentiate the various formations, but the clusters do not have a strong correlation with the performance of the log-synthesis models. Clusters identified using the *K*-means method have a strong correlation with the performance of the shallow-learning models. By combining the shallow-learning models for log synthesis with the *K*-means clustering, we propose a reliable workflow that can synthesize the DTC and DTS logs, as well as generate a reliability indicator for the synthesis logs to help the user better understand the performance of the shallow-learning models during deployment in new wells.

Introduction

Sonic-logging tools transmit compressional and shear waves through the formation to obtain formation-matrix/fluid information. Compressional waves travel through both the rock matrix and fluid; in contrast, shear waves travel only through the matrix. The wave-travel time depends on the elastic properties and moduli of the rock as well as the composition and microstructure of the formation. DTC and DTS logs can be computed from the waveforms recorded at the receiver. Sonic logs contain critical geomechanical information for unconventional-reservoir characterization. The difference and variation in the DTC and DTS contain information about the formation porosity, rock brittleness, and Young's modulus, to name only a few properties. However, sonic logs might not always be available, because of financial or operational constraints. This study aims to develop a workflow to synthesize both DTC and DTS logs from "easy-to-acquire" conventional well logs and simultaneously generate a reliability indicator for the synthesized logs. Prediction performance refers to the prediction accuracy of the log-synthesis models on the test data set, whereas prediction reliability indicates the reliability of the log-synthesis model on the new data set. For new data sets, we cannot generate the prediction accuracy and we will need to rely on the prediction reliability generated using clustering methods. The primary goal of this study is to generate synthetic sonic logs when they are not available. In this paper, we provide a way to compute the reliability of the synthesized logs. The reliability indicator will facilitate the engineers/scientists to wisely use and trust the synthesized logs. A performance-reliability indicator is needed for all data-driven predictive-modeling tasks.

Well logging is essential for the oil and gas industry to understand the in-situ subsurface petrophysical and geomechanical properties (Alexeyev et al. 2017; Wang et al. 2019). Certain well logs, such as gamma ray (GR), resistivity, density, and neutron, are considered as "easy-to-acquire" conventional well logs and are deployed in most wells. Other well logs, such as nuclear magnetic resonance, dielectric dispersion, elemental spectroscopy, and sonic, are deployed in a limited number of wells. Easy-to-acquire well logs can be processed using statistical- and machine-learning methods to synthesize the well logs that are not frequently acquired in each well. Researchers have explored the possibility of synthesizing certain "hard-to-acquire" well logs under data constraint (Tariq et al. 2016; Li and Misra 2017a, 2017b).

Machine-learning algorithms, such as ANN and fuzzy logic, are widely applied in prediction tasks in the oil and gas industry. The performance of these models can be easily adjusted according to the complexity of the problem. Several studies have tried to implement machine-learning techniques to determine the sonic log from other well logs. ANNs, adaptive neuro-fuzzy inference systems, and support-vector machines were used to predict both compressional- and shear-sonic times from the GR, bulk density, and neutron porosity (Elkatatny et al. 2016). The study achieved a correlation coefficient of 0.99 when tested with field data. In another study, shear-wave velocity (reciprocal of DTS) is predicted using an intelligent system, which combined the ANN, fuzzy-logic, and neuro-fuzzy algorithms. The mean-squared error during the testing stage was approximately 0.05 (Rezaee et al. 2007). The models show stable performance, but the model can predict only shear wave. A similar study applied a committee machine with intelligent systems to predict the sonic-travel time from conventional well logs (Asoodeh and Bagheripour 2012). Apart from machine-learning algorithms, other studies predicted DTS or DTC using empirical equations (Iverson and Walker 1988; Greenberg and Castagna 1992), empirical correlations (Maleki et al. 2014), or a self-developed model (Keys and Xu 2002). Other researchers tried to predict DTS and DTC in thin beds using petrophysical properties instead of raw conventional logs (Baines et al. 2008).

* now with Texas A&M University

This paper (SPE 191400) was accepted for presentation at the SPE Annual Technical Conference and Exhibition, Dallas, Texas, USA, 24–26 September 2018, and revised for publication. Original manuscript received for review 12 July 2018. Revised manuscript received for review 8 May 2019. Paper peer approved 13 May 2019.

Existing studies on log synthesis focus on the implementation of various supervised-learning models and on the evaluation of the predictive performance of these regression-type models (Al-Bulushi et al. 2007; Akinnikawe et al. 2018). There are limited studies on quantifying the predictive performance of log-synthesis models trained in one well and being deployed in different wells (Ao et al. 2018). There are few techniques to determine the reliability of the predictions of machine-learning models on a new well. In this study, we relate the performance of a log-synthesis model (using supervised learning) to formation rock types/clusters (using unsupervised learning). To that end, we developed regression-type models to synthesize sonic logs and an unsupervised-clustering model to identify similar and dissimilar rock types/clusters. After that, the log-synthesis performance is compared with the clusters, and we found that log synthesis using various supervised-learning models exhibits good correlation with the rock clusters identified using the *K*-means-clustering method. The workflow is shown in **Fig. 1.** In doing so, the trained log-synthesis models can be better applied to a new well, such that we can use the unsupervised-clustering method to assess the reliability of the log-synthesis performance in new wells.

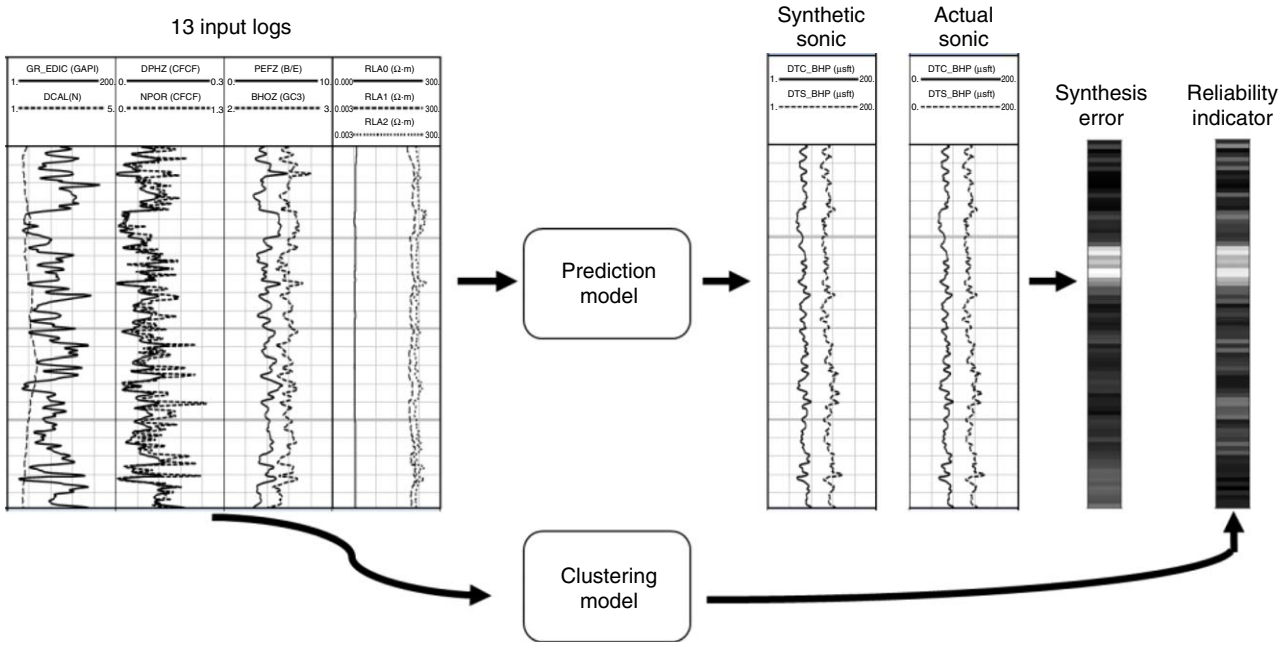

Fig. 1—Sonic-log-synthesis and evaluation workflow.

For the purpose of synthesizing DTC and DTS logs from conventional logs, our study aims to test the performances of shallow-learning models. A complex predictive model with many parameters is not necessary for the regression problem we are investigating in this study. The state-of-the-art deep-learning models, such as the convolutional neural network and the generative adversarial network, have shown great success in processing a high-dimensional complex data set. In this study, the inputs and outputs of the problem are relatively simple and in low dimensions. Moreover, shallow-learning models need fewer data and tend not to overfit. Shallow-learning models have two advantages: They need less computational resources and hyperparameter-tuning time, and they are easy to deploy and interpret. The 13 "easy-to-acquire" conventional logs together with DTC and DTS logs acquired in two wells are used for this study. The 13 input logs are GR, caliper log (DCAL), density–porosity log (DPHZ), neutron-porosity log (NPOR), photoelectric-factor log (PEFZ), bulk-density log (RHOZ), lithology type, and laterolog-resistivity logs at six depths of investigation (RLA0, RLA1, RLA2, RLA3, RLA4, RLA5). The six shallow-learning models implemented in this study are OLS, PLS, LASSO, EN, MARS, and ANN. We selected these because they are simple, well-known, and widely used. The first four models (OLS, PLS, LASSO, and EN) are considered linear models and the last two (MARS and ANN) are nonlinear models. The regression capacity of the last model, ANN, can be adjusted depending on the complexity of the problem. By comparing the performance of the six models with increasing regression capacity, we can identify whether shallow models are adequate for the sonic-log synthesis.

The "prediction performance" of the six regression-type models can be easily evaluated on the training/testing data set by using metrics such as the mean-squared error and the correlation coefficient. However, when the trained model is deployed in new wells, it is extremely challenging to evaluate the performance of the model and gauge the adequacy of the model in the new wells. In the absence of metrics for quantifying the performance of the regression-type models in new wells, our study shows that the prediction performance of the regression-type supervised-learning models for the log synthesis in new wells can be explained in terms of a reliability indicator generated using unsupervised-clustering methods, which identify groups in the data set according to the similarity between data points.

Clustering algorithms have been widely applied for the identification of lithofacies (Moghaddas et al. 2017; Shi et al. 2017), clustering wells (Qin et al. 2017), or formation evaluation (Handwerger et al. 2011; Petriello et al. 2013; Jain et al. 2015). In this study, we applied five clustering algorithms to generate the reliability indicators to accompany the synthesized logs. The five clustering algorithms are *K*-means, Gaussian mixture, hierarchical clustering, DBSCAN, and SOM. These five cover most of the popular clustering algorithms. Each of the five clustering methods uses a different approach, with different assumptions, to generate similar and dissimilar groups. Consequently, the clusters generated by these different methods have different statistical distributions and cluster centers. *K*-means clustering performs Voronoi-type partitioning of the data space using an appropriate distance metric (e.g., Euclidian distance), whereas Gaussian-mixture clustering uses expectation maximization that assumes the data are drawn from Gaussian distribution (Tan et al. 2006). Hierarchical clustering uses a greedy divisive or agglomerative strategy with an appropriate distance metric and a linkage criterion, and specifies the dissimilarity of clusters as a function of the pairwise distances of data points in the clusters (Tan et al. 2006). DBSCAN is a density-based clustering algorithm that groups together points that are closely packed together and flags points that lie

alone in low-density regions as outliers (Birant and Kut 2007). SOM-based clustering uses an ANN to produce a low-dimensional, discretized representation of the input space while preserving the topological properties of the input space using a neighborhood function (Kohonen 1990). By comparing the clusters generated by these distinct clustering algorithms, we will find the best model that can be incorporated to assess the reliability of the supervised-learning models. Petrophysical characteristics of clustering results were better visualized using t-SNE (van der Maaten and Hinton 2008), which is a dimensionality-reduction technique.

In this paper, six shallow-learning models and five clustering algorithms process 13 "easy-to-acquire" conventional logs to synthesize the compressional- and shear-travel times along with an indicator of the reliability of the log synthesis. The ANN model has the best prediction accuracy among the six prediction models, with an R^2 of 0.85. The *K*-means-clustering algorithm has a good differentiation ability, and the resultant group numbers are closely related to the prediction accuracy of the ANN. By combining the shallow-learning ANN model and the *K*-means-clustering algorithm, we developed a prediction workflow that can synthesize the compressional- and shear-travel-time logs and simultaneously generate the reliability indicator for the prediction results. This study will enable engineers and geoscientists to obtain improved geomechanical characterization when a sonic-logging tool is not available because of operational or financial constraints. Importantly, the reliability of the prediction results can also be computed to facilitate using the synthesized logs.

Data Preparation and Preprocessing

Data Preparation. Well-logging data used in this study were acquired from two wells in organic-rich shale formations in the Permian Basin. In Well 1, logs were acquired at 8,481 depth points from a 4,240-ft-depth interval. In Well 2, logs were acquired at 2,920 depth points from a 1,460-ft-depth interval. GR, DCAL, DPHZ, NPOR, PEFZ, RHOZ, lithology type, and laterolog-resistivity logs at six depths of investigation (RLA0, RLA1, RLA2, RLA3, RLA4, RLA5) are selected as the 13 "easy-to-acquire" conventional logs (as shown in Tracks 2 through 6 of **Fig. 2**) that are fed into the six predictive models and the five clustering algorithms. The depth shown in Fig. 2 is not used in the training or testing of the models in this study.

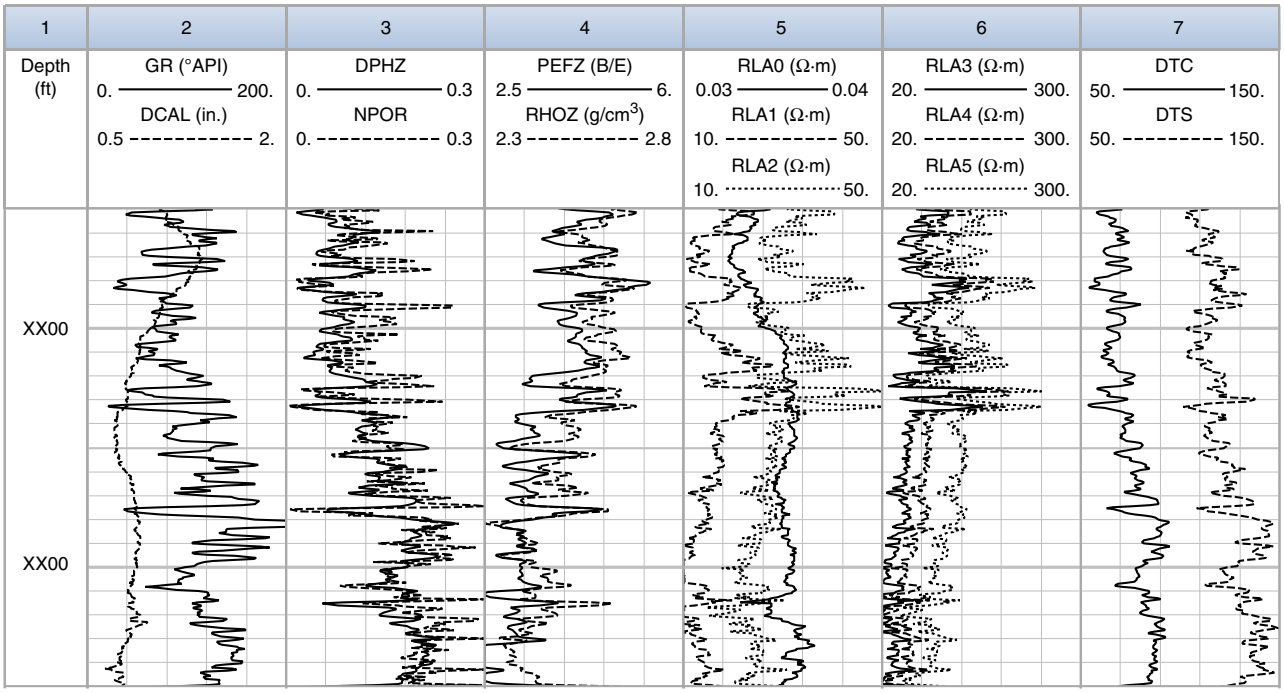

Fig. 2—Track 1 is depth, Track 2 contains GR and DCAL logs, Track 3 contains DPHZ and NPOR logs, Track 4 contains PEFZ and RHOZ logs, Track 5 is laterolog-resistivity logs at shallow depths of investigation (RLA0, RLA1, RLA2), Track 6 is laterolog-resistivity logs at deep depths of investigation (RLA3, RLA4, RLA5), and Track 7 contains DTC and DTS logs for a 200-ft section of the formation.

DCAL is a single-arm caliper, which unlike the multiarm caliper is not strongly correlated with hole breakout. The single-arm caliper indicates the hole size, which can also be attributed to drilling practices and mud parameters (density, composition) and need not be always linked to rock mechanics. For formations susceptible to breakouts, such as shale, variations in DCAL can serve as an indicator of the formation type, which can then be related to the geomechanical properties that govern the DTC and DTS logs. Sonic logs are shallow-sensing logs compared with resistivity and density logs, and consequently DCAL can serve as an indicator of the effects of borehole size on the sonic logs. Machine-learning models are suitable to integrate disparate data types to generate outputs that are beyond simple physics-based models and human perceptions. Consequently, we use the commonly available DCAL as an input in the proposed log synthesis.

DPHZ is derived from density logs (RHOB), with assumptions for matrix density and fluid density. RHOB is a direct measurement, and DPHZ is an estimation. We use both DPHZ and RHOB logs, despite their correlation, because their interplay indicates fluid and matrix properties and their variations along the well length. Having both DPHZ and NPOR assists the shallow-learning models used in our paper despite the information redundancy. For low-dimensional data, like those we have, such dependencies/colinearity does not adversely affect the accuracy of the model. Constructed/derived features, such as DPHZ computed from RHOB, can help the shallow-learning models by establishing a better correlation of features with the target and by reducing the adverse effects of noise. Shallow-learning models need feature engineering, and DPHZ can be considered as an engineered feature to assist the shallow-learning models. In our study, the LASSO model suggests that the RHOZ and NPOR, and not DPHZ, are important for the desired log-synthesis task. The regularization term of the models can successfully eliminate the information redundancy.

Formations with different lithologies exhibit different petrophysical and mechanical properties, which in turn affect the sonic-wave propagation. A geology expert identified 13 broad lithology types in the 4,240-ft-depth interval and assigned specific sections of the depth interval with discrete-valued lithology types ranging from 1 to 13 to indicate the formation lithology. DTC and DTS logs are the outputs of the models (as shown in Fig. 2, Track 7).

Data Preprocessing. Data preprocessing aims to facilitate the training process by appropriately transforming the entire data set to an equivalent range. All logs used as features to develop the data-driven model are scaled using a MinMaxScaler algorithm. Such min-max scaling ensures that the model is robust and helps in the fast convergence of the learning process, especially when using neural networks. Min-max scaling is performed using

$$y'_i = 2\frac{y_i - y_{\min}}{y_{\max} - y_{\min}} - 1, \quad (1)$$

where y_i is the original value of a log response and y'_i is the scaled value of the log response at each depth i. Eq. 1 scales all the logs (features) within a range of $[-1, 1]$; consequently, the models will not be biased toward the logs with large values and variances. Each value is deducted by the minimum value and divided by the maximum range of the data to scale the features within [0, 1]. After this, when multiplied by two and deducted by unity, the features are scaled within $[-1, 1]$. All "easy-to-acquire" logs are scaled using Eq. 1. In this study, we did not reshape the input-log distribution or try other standardization/normalization techniques. Categorical features (such a lithology) are also scaled using Eq. 1. Nominal categorical data (in the absence of inherent order in the categories) usually should be transformed using one-hot encoding and should not be scaled. However, in our study, we decided not to perform one-hot encoding because it would have generated an additional 13 features/inputs. Additional features lead to overfitting and the curse of dimensionality, especially when there are a limited number of samples, as with our study. Instead of one-hot encoding, we applied min-max scaling on the categorical-lithology log.

Measurement of Prediction Performance. The correlation coefficient (R^2) is used to compare the prediction performance of all models, and is formulated as

$$R_j^2 = 1 - RSS_j/TSS_j, \quad (2)$$

where

$$RSS_j = \sum_{i=1}^{n}\left(y_{pi,j} - y_{mi,j}\right)^2, \quad (3)$$

and

$$TSS_j = \sum_{i=1}^{n}\left(y_{pi,j} - \overline{y}_j\right)^2, \quad (4)$$

where n is the number of depths for which prediction needs to be performed; $j = 1$ indicates the DTC log and $j = 2$ indicates the DTS log; i represent the depth; $y_{pi,j}$ is the sonic log j predicted at depth i; $y_{mi,j}$ is the sonic log j measured at depth i; and $\overline{y}_j$ is the mean of the sonic log j measured at all depths for which training or testing is being performed. RSS_j is the sum of squares of the residuals, and TSS_j is the total sum of squares proportional to the variance of the corresponding sonic log j.

Methodology

Six simple prediction models are used to synthesize DTS and DTC logs by processing 13 "easy-to-acquire" logs. We assume that the six shallow-learning models can capture the hidden relationships between the 13 input logs and the two output sonic logs.

OLS Model. The OLS model is one of the simplest statistical regression models that fits the data by reducing the sum of squared errors (SSE) between the modeled and measured data (Chumney and Simpson 2006). The OLS model assumes the output y_i is a linear combination of input values x_{ip} and error ε_i, formulated as

$$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad (5)$$

where i represents a specific depth from the total-depth samples available for model training and p represents the number of input logs available for the training. The model fits the data set by reducing the SSE, expressed as

$$SSE = \sum_{i=1}^{n}(\hat{y}_i - y_i)^2, \quad (6)$$

where $\hat{y}_i$ is the predicted output of the model. In this study, the input data x_{ip} are the 13 raw logs at depth i and the output data y_i are sonic logs. OLS predictions are adversely affected by outliers, noise in data, and correlations among the inputs.

PLS Model. The OLS model linearly relates features (input) to targets (output). On the other hand, PLS finds a latent space that explains the maximum variance in the target; after that, PLS finds the directions in the feature space that explain the trends in the latent space derived from the target space. The PLS model aims to find the correlations between the input and output data by constructing latent structures. Input and output logs are decomposed into their latent structures (Lingjaerde and Christophersen 2000; Wold 2004; Abdi 2010). The latent structure corresponding to the most variation in output is extracted and then explained using a latent structure in the input space. Finally, the new latent structures are combined with the original variable to form components. The number of components m (no more than inputs) is chosen to maximally summarize the covariance with outputs. The determination of m builds the PLS

model with fewer inputs than the OLS model, which is more suited when there are high correlations among inputs. The removal of redundant inputs can save computational time for training the model while simplifying the structure of the model. In applying the PLS model, the most important parameter is the number of components to be generated. In our study, the smallest m with the best prediction performance is used to build the model. In the PLS model, sonic logs are generated with constructed components, which are combinations of the latent structures of original inputs. We select the number of components by testing a range of values and monitoring the change of the model performance. The best performance of the model occurs when the number of components equals 13, which indicates the absence of correlated inputs.

LASSO Model. LASSO (Tibshirani 1996) is a linear-regression model that combines the SSE and constrains the sparsity of the coefficient matrix with the L_1 norm. The objective function of LASSO is expressed as

$$\min_{w} \frac{1}{2n} \boldsymbol{X}\boldsymbol{w} - \boldsymbol{Y}_2^2 + \lambda \boldsymbol{w}_1, \quad (7)$$

where $\boldsymbol{w}$ is the coefficient vector, $\boldsymbol{X}$ is the measured input vector, $\boldsymbol{Y}$ is the measured output vector, n is the number of depth samples in the training data set, and λ is the penalty parameter that balances the importance between the SSE term and the regularization term, which is the L_1 norm of the coefficient vector. The SSE is represented as the squared L_2 norm. If λ increases, the regularization term punishes the coefficient matrix to be sparser. The λ term is optimized by testing a range of the values. Different values of λ are tested, and the prediction performance of the model is compared. We select $\lambda = 4.83$, for which the prediction accuracy of the model is the best. R^2 for the DTC and DTS predictions is 0.79 and 0.75, respectively. When $\lambda = 4.83$ is used, the values of the coefficient for each input log are listed in **Table 1.** The details of the optimization process are shown in **Fig. 3.** When $\lambda > 4.83$, the coefficient stabilized. Coefficients of six out of the 13 input logs are zero in the LASSO model. Those logs with a coefficient of zero are less important to the desired log synthesis compared with other logs with nonzero coefficients. These logs might contain redundant and unrepresentative information not essential for the LASSO-model predictions.

Lithology	GR	DCAL	DPHZ	NPOR	PEFZ	RHOZ	RLA0	RLA1	RLA2	RLA3	RLA4	RLA5
0.31	0.06	–1.29	0.00	41.63	0.60	–55.51	0.00	0.00	–1.01	–1.05	0.00	0.00

Table 1—Estimates of coefficients for each input log in the LASSO model for $\lambda = 4.83$.

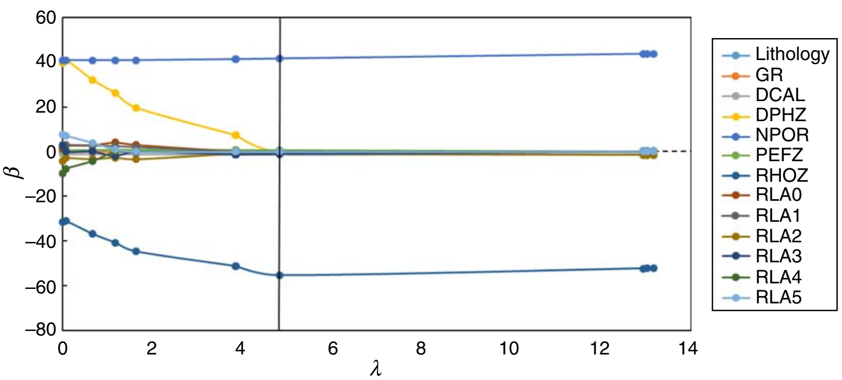

Fig. 3—Optimization plot for λ.

In the proposed LASSO implementation, four of the six resistivity logs (RLA0, RLA1, RLA4, and RLA5) were assigned a weight of zero (Table 1) because the resistivity logs at various depths of investigation usually exhibit strong correlation, and the sonic logs, with medium depth of investigation, tend to have lower correlations with shallow- and deep-resistivity logs, which were assigned weights of zero. By weighting the contributions of certain features by zero, the LASSO model eliminates redundant information at the cost of lower prediction accuracy. The weights of GR and DPHZ are close to zero, which indicates that these logs are correlated with other logs that were assigned high-valued weights by the LASSO model. Notably, RHOZ and NPOR have the highest weights because they measure bulk density, porosity, shaliness, and structural, bound, and free water that strongly influence the sonic-shear- and compressional-travel times (DTS and DTC logs). In summary, the LASSO model achieves a limited predictive performance while neglecting the correlated features, which reduces the curse of dimensionality.

EN Model. The EN model is a linear-regression model suitable for high-dimensional data that combines both the L_1 and L_2 norms as the penalty term (Zou and Hastie 2005; Kim et al. 2007). Unlike the LASSO model, the EN model preserves certain groups of correlated input logs and does not ignore highly correlated inputs. The objective function of the EN model is defined as

$$\min_{w} \frac{1}{2n} \boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}_2^2 + \alpha_1 \boldsymbol{w}_1 + \alpha_2 \boldsymbol{w}_2^2. \quad (8)$$

Compared with Eq. 7 relevant to LASSO, Eq. 8 uses the L_2 norm. EN is a combination of ridge and LASSO regression. Ridge regression uses the L_2 norm, which generates a less-sparse solution compared with LASSO regression. EN balances the L_1 and L_2 norms. Two penalty parameters, α_1 and α_2, are determined through grid search to be 4.8 and 0.1, respectively. Grid search is a technique for finding the best combination of hyperparameters that result in the best generalization performance of the data-driven model; in other words, grid search is a technique for hyperparameter optimization. Hyperparameters, such as α_1 and α_2, control the learning process, and model parameters, such as the elements of vector $\boldsymbol{w}$, are the consequence of learning. In grid search, data-driven models are developed

for different combinations of hyperparameters, as specified by the model developer. For instance, when performing grid search for two hyperparameters a and b to find the best combination of hyperparameters out of the values of $a = [1, 2, 3, 4]$ and $b = [0.1, 1, 10, 100]$, a data-driven model will be developed for all possible combinations of a and b (in this case, 16 combinations) and the one combination of hyperparameters that generates the best testing score is chosen as the optimal hyperparameters.

MARS Model. The MARS model uses multiple linear-regression models in the input space (Friedman 1991). By fitting the data using multiple linear regressions, the model can capture the nonlinearity of the data set. The model is a weighted sum of basis functions. The MARS model is formulated as

$$\hat{y}_i = \sum_{q=1}^{p} \alpha_q B_q(x_{qi}), \qquad (9)$$

where x_{qi} is the value of the qth input log x_q at the ith depth point, $B_q(x_{qi})$ is a basis function, and α_q is the coefficient of B_q. The basis function can have many different forms, but in general it is a hinge function. A hinge function is a linear function within a range expressed as

$$\max(0, x_{qi} - C_q) \text{ or } \max(0, C_q - x_{qi}). \qquad (10)$$

The shape of a hinge function can be visualized in **Fig. 4.** The hinge function partitions the input into different sections by using different C_q values. The MARS algorithm approximates a nonlinear relationship with multiple hinge functions. As expressed in Eq. 10, each hinge function is zero for some portion of the input range depending on the selection of C_q. By combining hinge functions with different C_q values, the MARS algorithm splits the input range into small sections and approximates each section separately. In our study, we use 21 terms to partition the input logs. The model generates DTC and DTS with R^2 values of 0.85 and 0.83, respectively. The number of terms is selected by testing a range of different values.

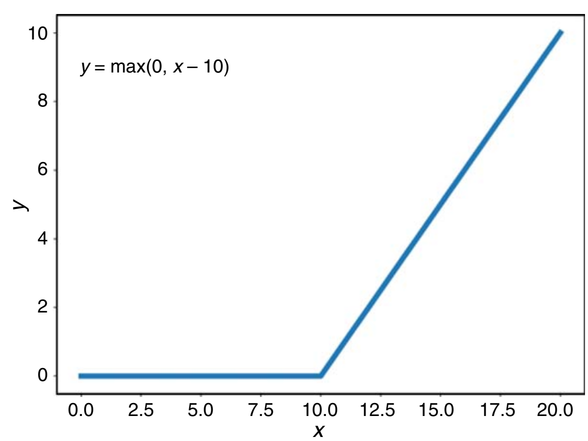

Fig. 4—Shape of a hinge function with $C_q = 10$.

ANN Model. The ANN model is a widely used machine-learning model suitable for both linear and nonlinear regression (Yegnanarayana 2009). A neural network comprises an input layer, an output layer, and several hidden layers. The capacity of the neural-network model to fit data can be adjusted by adding or decreasing the number of hidden layers and the number of neurons in each hidden layer. Each hidden layer contains neurons made of parameters (weights and biases) to perform matrix computations on signals computed in the previous layer. The activation function in each layer adds nonlinearity to the computation. In our case, there are 13 input logs and two output logs to be synthesized, which are the DTS and DTC logs. The dimensions of the input and output layers are 13 and 2, respectively. We use two hidden layers in the ANN model with nine and five neurons in the first and second hidden layer, respectively. The architecture of the neural network can be seen in **Fig. 5** (Beale et al. 2012). The activation function of the hidden layer uses the tanh function, and the output layer uses a linear-activation function. The shape of the tanh activation function is elaborated in **Fig. 6.** The neural network implemented in our study uses scaled conjugate gradient back propagation to update the parameters of the neurons (Møller 1993).

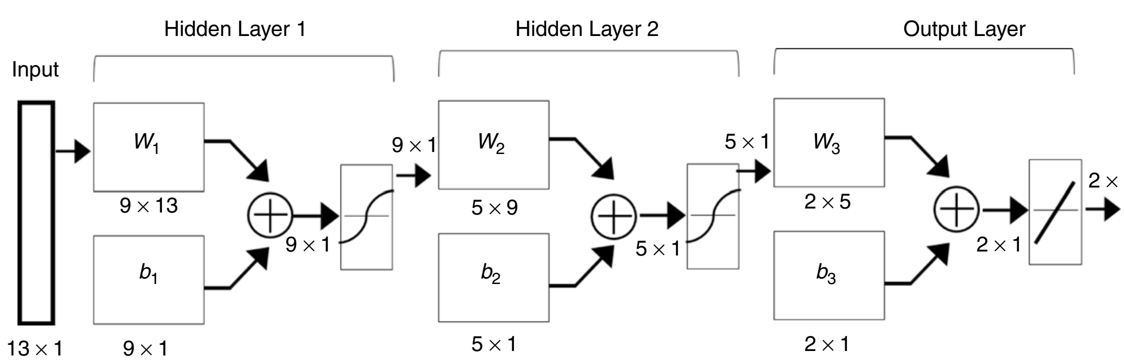

Fig. 5—Neural-network architecture.

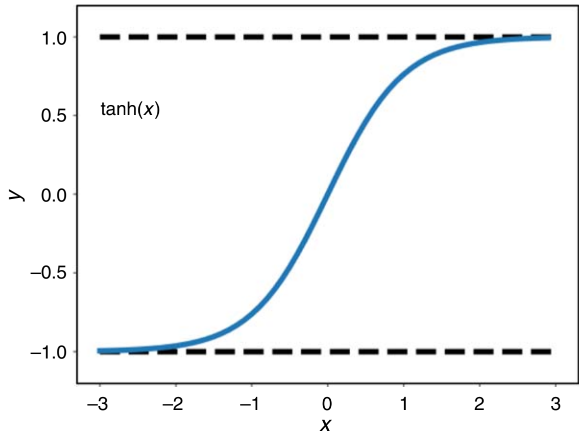

Fig. 6—Shape of tanh activation function.

Case Study

Prediction Results of Six Models. The previously discussed six shallow-learning models are trained and tested in a 4,240-ft-depth interval in Well 1 with 8,481 data points and deployed for blind testing in a 1,460-ft-depth interval in Well 2 with 2,920 data points. During the training and testing phases, the models were trained with 80% of randomly selected data from Well 1 and tested with 20% of the data. Tenfold cross validation was performed to ensure the robustness of the model predictions. The tenfold cross-validation method split the training data set into 10 equal parts. The model is trained 10 times, and each time the model is trained with ninefold of data and validated with the remaining fold of data. This cross-validation method ensures that the whole model is trained to generalize over the entire available data set.

As discussed previously, the "prediction performance" of the six regression-type models can be easily evaluated on the training/testing data set by using metrics such as the mean-squared error and the correlation coefficient. However, when the trained model is deployed in new wells, it is extremely challenging to evaluate the performance of the model and gauge the adequacy of the model in the new wells. In the absence of metrics for quantifying the performance of the regression-type models in new wells, our study shows that the prediction performance of the regression-type supervised-learning models for the log synthesis in new wells can be explained in terms of a reliability indicator generated using unsupervised-clustering methods, which identify groups in the data set depending on the similarity between data points. The idea of applying clustering algorithms is to group the formation layers, such that the layers in each group share similar petrophysical/statistical properties. If a clustering method exhibits a strong correlation with the log-synthesis performance of a shallow-supervised-learning model, the clustering methods can then be used to evaluate log synthesis using the shallow model in new wells.

The performance of log synthesis is evaluated in terms of R^2. The log-synthesis results for Wells 1 and 2 are shown in **Table 2.** OLS and PLS exhibit similar performances during training and testing but not during the deployment (blind testing). LASSO and EN have similar performances during training, testing, and blind testing. Among the six models, ANN performs the best, with R^2 of 0.85 during training and testing and 0.84 during the blind testing, whereas LASSO and EN perform worst with an R^2 of 0.76 during the blind testing. As shown in Table 2, when the trained models are deployed in Well 2, all models exhibit a slight decrease in prediction accuracy. ANN has the best performance during deployment. The accuracies of the DTC and DTS logs synthesized using the ANN model are shown in **Fig. 7,** where the measured and synthesized sonic logs are compared across 300 randomly selected depth samples from Well 2.

Accuracy		OLS	PLS	LASSO	EN	MARS	ANN
Well 1	DTC	0.83	0.83	0.791	0.791	0.847	0.87
	DTS	0.803	0.803	0.756	0.753	0.831	0.848
Well 2	DTC	0.804	0.79	0.778	0.774	0.816	0.85
	DTS	0.794	0.769	0.763	0.755	0.806	0.84

Table 2—Prediction performances in terms of R^2 for the six models trained and tested in Well 1 and deployed in Well 2.

Comparison of Prediction Performances of Six Models in Well 1. In this subsection, the relative error (RE) is used to evaluate the prediction performance of the machine-learning models for log synthesis. RE for a log synthesis is formulated as

$$RE = \frac{|P - M|}{M} \quad \text{(11)}$$

where P is the predicted value and M is the measured value of either the DTS or DTC log at a depth i. RE values are first individually calculated for DTC and DTS logs, and then the two RE values are averaged at each depth to represent the mean prediction performance of a shallow-learning model. The RE values are averaged because the REs for DTC and DTS correlate very well, as shown in Fig. 7. Moreover, by calculating the average of the two sonic logs, we obtain a single value to evaluate the overall log-synthesis performance at each depth. The averaged RE for the two logs at each depth is further averaged over 50-ft depth intervals to reduce the effects of noise, borehole rugosity, and thin layers. The proposed methods still work for the reservoir with thin layers. The averaged RE over the 50-ft interval improves the visualization of the general trend of the log-prediction error. With the RE averaged over the 50-ft-depth interval, we can keep the original trend and visualize the trend as shown in **Fig. 8.**

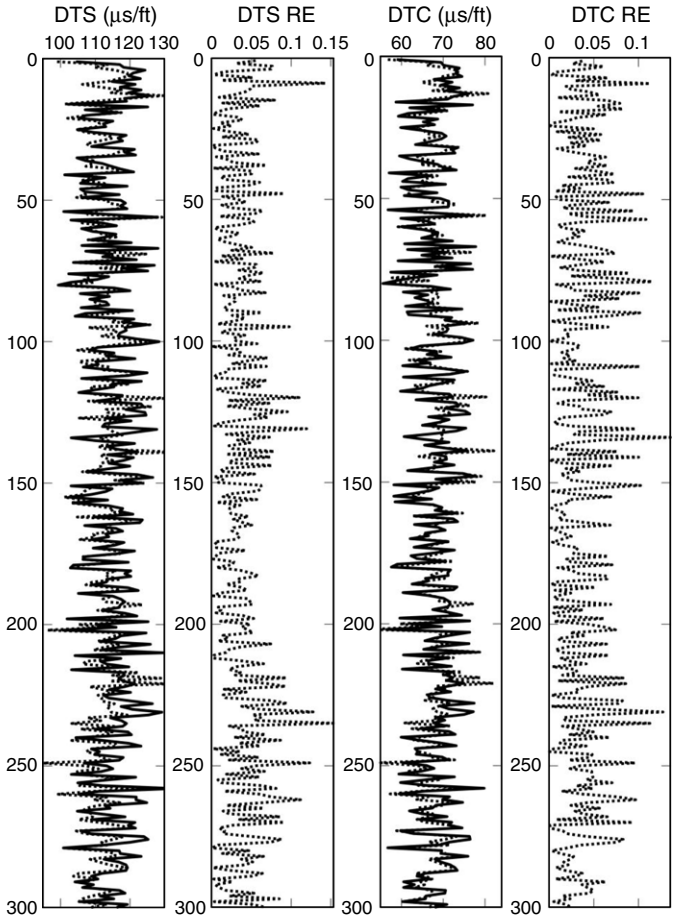

Fig. 7—Comparison of measured (dashed) and synthesized (solid) DTC and DTS logs in Well 2, when the ANN model is trained and tested in Well 1 and deployed in Well 2 to synthesize the DTC and DTS logs.

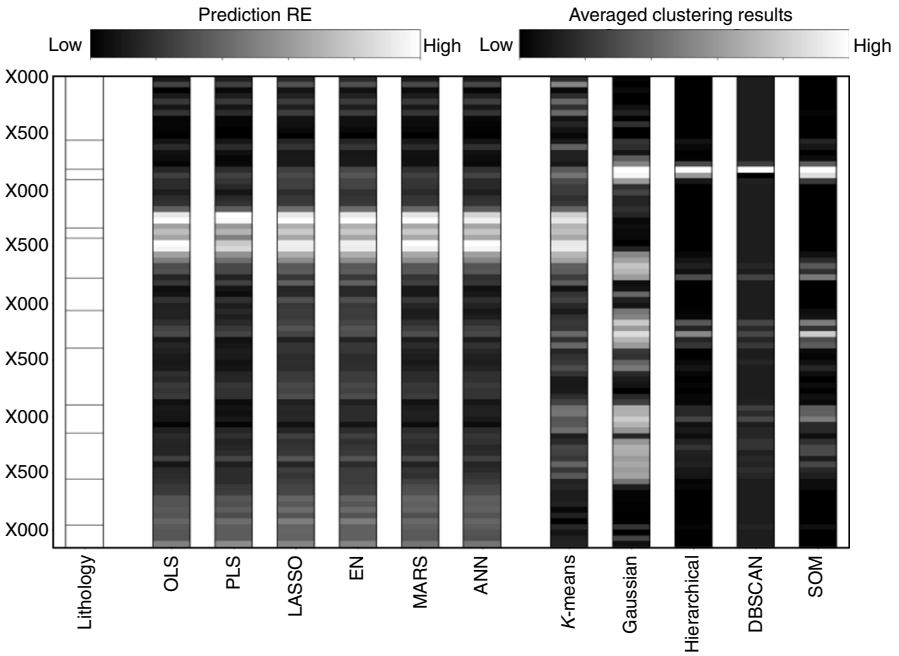

Fig. 8—The first column represents lithology types; each block represents a lithology type. The next six columns are the comparisons of REs of the synthesized DTS and DTC logs generated using the six models for the 4,240-ft-depth interval in Well 1, where paler-colored intervals represent depths that represent zones in which the learning models exhibit bad prediction performances. Paler colors represent higher RE, and darker colors represent lower RE. The last five columns demonstrate the clustering results using different clustering algorithms. The clustering results are averaged over a 50-ft-depth interval.

The first column in Fig. 8 is the lithology type. Each block represents a formation with different lithology. The next six columns in Fig. 8 show the averaged RE of the six models in Well 1. The six models exhibit very similar patterns of RE over the 4,250-ft-depth interval, as shown in the six columns. The six models perform badly in the upper-middle part of the selected formation (approximately 1,250 to 1,800 ft below the top of the formation depth under investigation). It is possible that the zone of poor performance has certain physical properties that are very different from the rest or those where the logs have very distinct statistical features. In the following sections, clustering algorithms process the "easy-to-acquire" logs to identify clusters that exhibit high correlation with the REs in the synthesized DTC and DTS logs using the six machine-learning models, with a focus on the ANN model. The primary goal of combining the clustering models and the shallow-log-synthesis models is to use clustering results as a reliability indicator for the synthesized logs.

Clustering Analysis of the Performance

In this section, we describe the implementations of five clustering methods that process the "easy-to-acquire" input logs to differentiate the various formations in a well into distinct groups (depth intervals) depending on certain similarities of the "easy-to-acquire" input logs for various groups. The goal is to generate clusters/groups and assess their correlations with the accuracy of the machine-learning models for log synthesis as applied to the training and testing data sets. In doing so, the clustering-algorithm-generated group numbers exhibit a strong correlation with the RE of synthesized logs that can be used as a reliability indicator for the prediction performance of machine-learning models for log synthesis.

Clustering algorithms aim to find relationships in the data set in an unsupervised manner. The goal of clustering is to group data into a number of clusters such that the data points in the same cluster share the most similarity. A clustering technique provides a direct way to explore and visualize a data set. In this study, we implement five clustering methods: centroid-based *K*-means, distribution-based Gaussian mixture, hierarchical clustering, DBSCAN, and SOM. As mentioned previously, each of the five clustering methods uses a different approach with different assumptions to identify the clusters. Consequently, the clusters generated by these different methods have different statistical distributions and cluster centers. *K*-means clustering performs Voronoi-type partitioning of the data space using an appropriate distance metric (e.g., Euclidian distance), whereas Gaussian-mixture clustering uses expectation maximization that assumes the data are drawn from Gaussian distribution. Hierarchical clustering uses a greedy divisive or agglomerative strategy that uses an appropriate distance metric and a linkage criterion, which specifies the dissimilarity of clusters as a function of the pairwise distances of data points in the clusters (Tan et al. 2006). DBSCAN is a density-based clustering algorithm that groups together points that are closely packed together and flags points that lie alone in low-density regions as outliers (Birant and Kut 2007). SOM-based clustering uses an ANN to produce a low-dimensional, discretized representation of the input space while preserving the topological properties of the input space using a neighborhood function (Kohonen 1990).

The *K*-means-clustering algorithm starts by randomly setting the center of each group. Data points are clustered around the nearest centers, and new centers are iteratively recalculated using data points in various clusters. The Gaussian-mixture-clustering model assumes the data set is generated using Gaussian distribution. The hierarchical-clustering model clusters a data set by merging or splitting data hierarchically according to certain similarities. DBSCAN clusters the data set using the density of the data points. The algorithm will group points with many neighbors into one group and recognize points with fewer neighbors as outliers. DBSCAN identifies different clusters by counting the number of neighbors within a certain distance. The distance is selected manually according to the characteristic of the data set. The last algorithm, SOM, is an algorithm that uses neural networks to perform unsupervised learning on the data set. The algorithm is an unsupervised dimensionality-reduction algorithm that projects high-dimensional data into two dimensions while keeping its original similarity (Pedregosa et al. 2011). Here, we first apply SOM projection, then use *K*-means to cluster the dimensionality-reduced data into groups. The procedure is inspired by the fact that the low-dimensional data abstracted by SOM can be used in clustering of data, and the lower representation of the data improves the efficiency of the clustering algorithm (Vesanto and Alhoniemi 2000).

Correlation Analysis. We first applied the clustering algorithms on all the input logs. The clustering results were unreliable. From our experiments, we found that the clustering results show no correlation with the errors of shallow-model predictions. High dimensionality and high nonlinearity when using all the input logs resulted in complex relationships that are challenging for the clustering algorithms to resolve into consistent groups. We ran several experiments to find the best subsample of injection-well logs that generate reliable clusters. Clustering with three logs (DPHZ, NPOR, and RHOZ) exhibited strong correlation with the errors of shallow-model predictions in Well 1 **(Fig. 9).** The Pearson-correlation coefficients calculated for the three logs DPHZ, NPOR, and RHOZ and the ANN-prediction RE are 0.66, 0.70, and –0.66, respectively.

***K*-Means Clustering.** For the *K*-means clustering, we use

$$p(\vec{x}) = \sum_{i=1}^{n} \min_{\mu_j \in C} x_i - \mu_j^2. \quad (12)$$

K-means clustering requires us to manually set the number of clusters. Formations are clustered into three groups such that the groups correspond to good, intermediate, and bad prediction performances. **Fig. 10** shows the inertia (Pedregosa et al. 2011), or the sum of squared criterion (Eq. 11), with respect to the number of clusters. Ten random experiments are conducted with different random seeds (zero to nine seeds) to test the clustering result of the *K*-means. There is a negligible difference between the "elbow" plots (Fig. 10) for each experiment. Fig. 10 shows 10 "elbow" curves with different seeds. The 10 curves overlap with each other. The turning point of the plot is approximately when the number of clusters is three, such that on addition of more clusters, the rate of decrease in inertia reduces. To compare the cluster numbers with good, intermediate, and bad performances in log synthesis, the number of clusters generated by the five clustering methods is set to three. There are no strict rules to select the number of clusters to be generated. More than three clusters will make it difficult to compare the clusters with the performance of log synthesis categorized into good, intermediate, and bad. Therefore, we generate only three clusters using the five different clustering methods.

K-means clustering was applied to certain "easy-to-acquire" logs acquired along a 4,240-ft-depth interval of Well 1. *K*-means clustering assigns a group number to each depth along the well length. Similar to the calculations of the averaged RE for the synthesized logs, the *K*-means-based group numbers are averaged over each 50-ft-depth interval in the well to eliminate the effects of noise and outliers. The averaged group number is compared with the averaged RE in **Fig. 11,** which shows an approximately linear relationship between the averaged *K*-means-based group numbers and the averaged RE of the ANN-based log synthesis. Several data points in Fig. 11 lie inside the 95% confidence band, indicating that when an averaged *K*-means-based group number is known for any 50-ft-depth

interval, we can then determine the averaged RE for the ANN-based log synthesis in that 50-ft-depth interval with close to 95% confidence. In other words, the group numbers generated by the *K*-means clustering can be used to determine the reliability of the sonic logs synthesized using the shallow-learning models. The *K*-means clustering results are also shown in Column 7 in Fig. 8, which compares the correlation between the group pattern of *K*-means-clustering results with the RE of the log synthesis. Fig. 11 shows a strong correlation between the averaged group number and the prediction accuracy, with a Pearson-correlation-coefficient value of 0.76. The correlation indicates the relationship between different groups obtained from the clustering and the prediction accuracy of the ANN-based log-prediction model. Zones with similar properties are clustered into one group, and each group exhibits similar log-synthesis performance. Fig. 11 shows that if a zone is clustered into Group 2, the prediction performance is very likely to be lower than that of other groups. The group number of the clustering procedure is a categorical type of datum. Permutation of the clustering-group number is performed to generate the best-correlated averaged group number and RE plot.

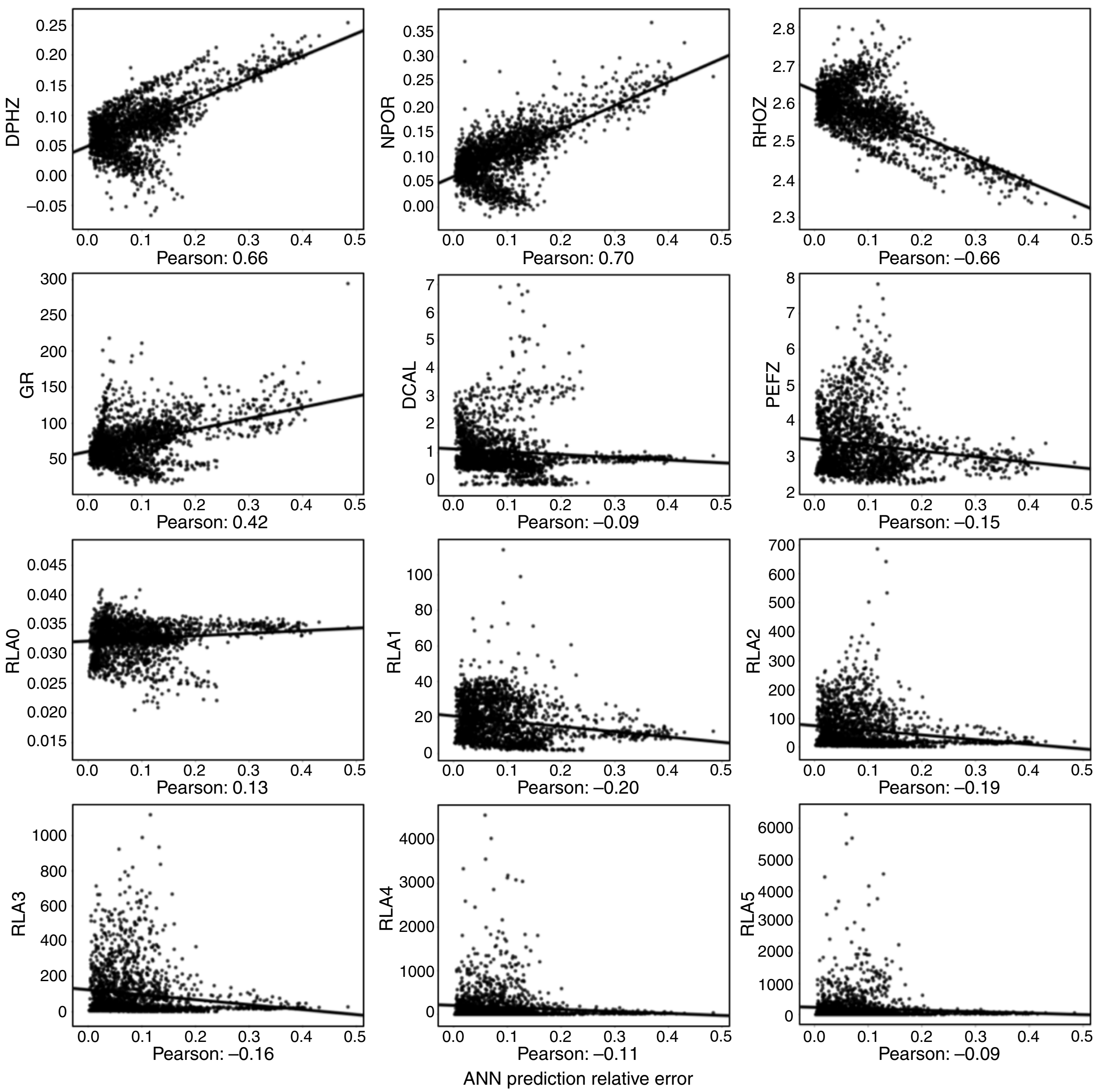

Fig. 9—Correlation plots between an RE in log synthesis using ANN and (a) DPHZ, (b) NPOR, and (c) RHOZ.

Gaussian Mixture. The Gaussian-mixture-clustering algorithm assumes that the data in a specific cluster are generated using a specific Gaussian distribution. Gaussian distribution is a widely used distribution to model various engineering data sets. For a data set with multiple dimensions, the Gaussian-mixture model fits the data set by parameterizing the weight of each cluster ϕ_i and the mean and covariance of the cluster $\sum_i$, where i is the cluster number. If there are *K*-clusters in the data set, the Gaussian-mixture model fits the data set by optimizing the sum of Gaussian distributions,

$$p(\mathbf{x}) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\mathbf{x}|\vec{\mu}_i, \Sigma_i), \quad (13)$$

where **x** is the data-point vector, K is the number of clusters, $\vec{\mu}_i$ is the mean of a cluster, $\sum_i$ is the covariance matrix, ϕ_i is the component weight, and $\mathcal{N}$ is the Gaussian distribution. The sum of the weights of all clusters equals unity. After fitting the data with multiple Gaussian distributions, the results can be used to cluster each data point into a cluster.

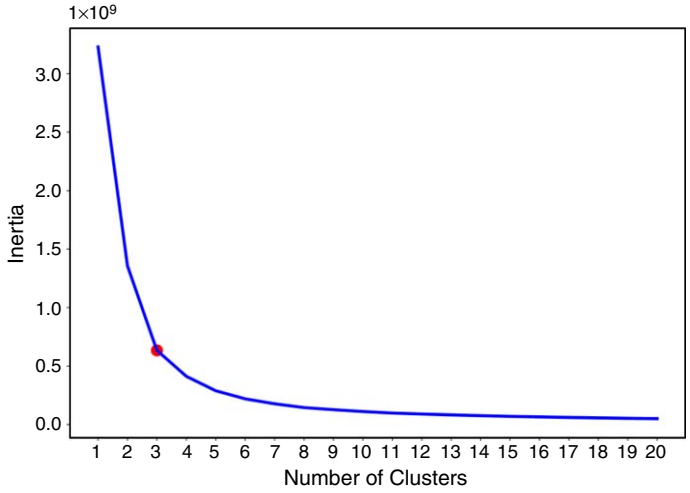

Fig. 10—Inertia with respect to the number of clusters in *K*-means clustering to identify the optimal number of clusters. Ten curves with different seeds are plotted. The curves overlap with each other.

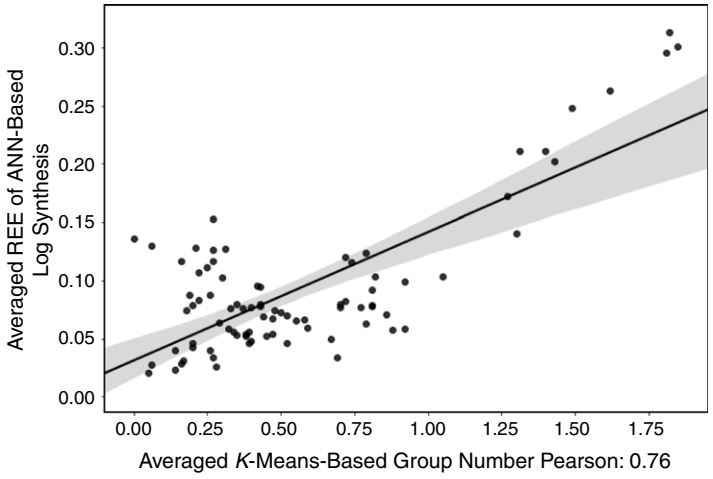

Fig. 11—Comparison between averaged *K*-means-based cluster number (*x*-axis) and averaged RE of ANN-based log synthesis (*y*-axis) with linear regression of 95% confidence. Each point is generated by averaging the *K*-means-based group numbers and the REs of ANN-based log synthesis along a 50-ft-depth interval. The confidence band indicates that we are 95% confident that the linear-regression relationship in log synthesis is in the band.

Using multiple Gaussian distributions to fit the data set is reasonable for clean data with no noise. Well logs used in this study contain noise and uncertainties, which might result in a high variance of each cluster. The clusters might overlap with each other, making it difficult to differentiate into distinct clusters. From **Fig. 12a,** we can see that the average group number correlates loosely with the RE with a Pearson-correlation coefficient of –0.22. The points scatter all over the plot and do not show obvious correlation. In Fig. 8, we can see that the Gaussian-mixture model identified several layers. However, the pattern learned by the model is completely different from the patterns of REs of shallow-learning models used for log synthesis.

Hierarchical Clustering. Our application of the hierarchical-clustering algorithm starts with every data point as a cluster, and then the clusters are repeatedly merged together according to their similarity until the target number of clusters is reached. The similarity of clusters is evaluated depending on the sum of squared difference. The merging process forms a hierarchical tree of clusters. The hierarchical-clustering algorithm does not differentiate the input data set enough. In Fig. 12b, we can see that most of the averaged group numbers are located near Group 0, which means the method clusters most of the formation data points into one group. The cluster number negatively correlated with the RE. The results shown in Figs. 8 and 12b demonstrate that the hierarchical-cluster algorithm does not differentiate different formations, as expected. In **Fig. 13,** the Gaussian-mixture-clustering results and the hierarchical-clustering results have a loose correlation with each other. Clustering algorithms applied in this study are sensitive to outlier values, except for *K*-means. Other than *K*-means, other clustering algorithms exhibit correlation with each other and cluster most of the samples in the data set into a single group.

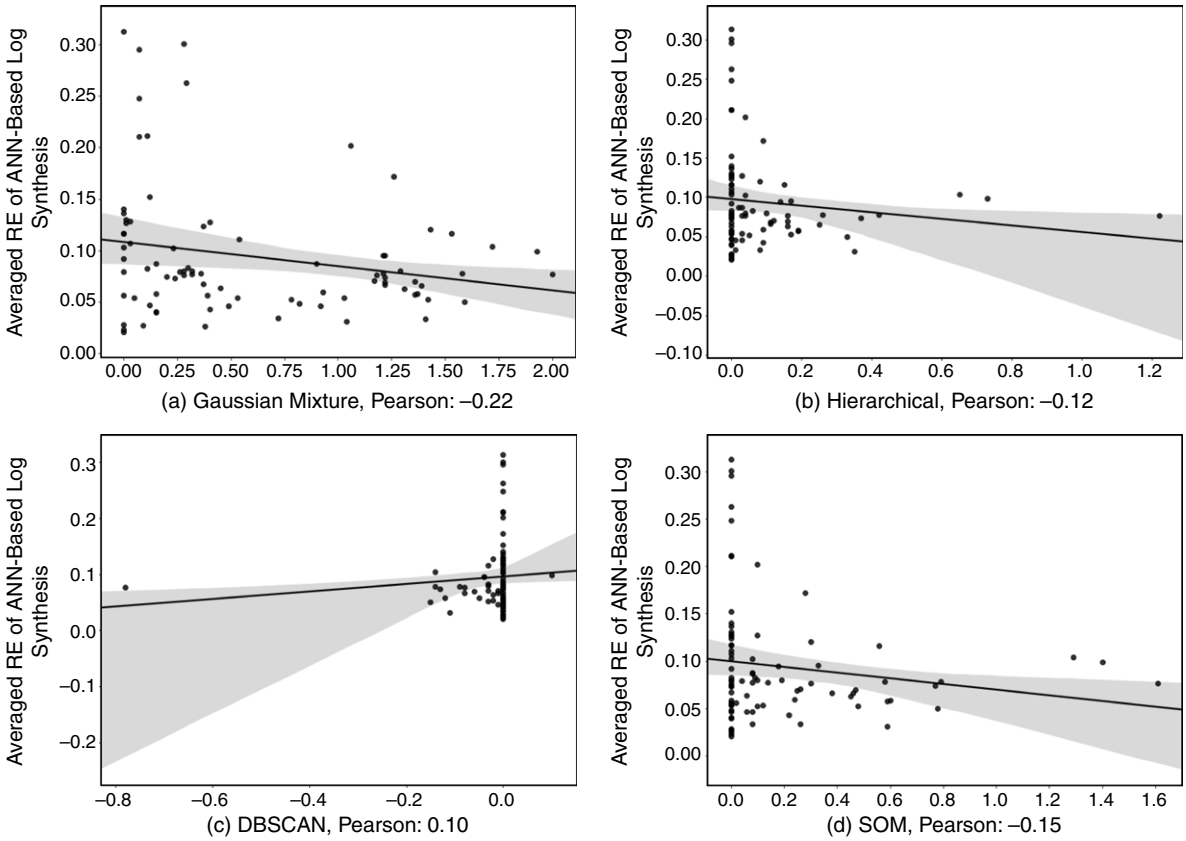

Fig. 12—Visualization of averaged group number and averaged prediction RE with linear regression of 95% confidence.

DBSCAN Clustering. DBSCAN is a density-based-clustering method. Unlike the *K*-means clustering, the DBSCAN methods do not need the user to manually define the number of clusters. Instead, they require the minimum number of neighbors and the range of distance to identify neighbors. Within a certain range, the algorithm will count the number of neighbors, and if the number of neighbors exceeds the minimum values, DBSCAN will identify this group of data points as a cluster. We tried a different combination of the parameters to obtain three clusters. We set our minimum number of neighbors as 100 and the range of distance as 10. The results show that the DBSCAN method identifies many data points as outliers, which are clustered into Group –1. By averaging the result similar to that implemented in *K*-means, we found that the DBSCAN algorithm does not successfully identify the patterns in the input logs. Most of the formations are clustered into Group –1, which means outliers, or are clustered into Group 0.

SOM Clustering. SOM is not strictly a clustering algorithm. It is a dimensionality-reduction algorithm mostly used in high-dimensional-data visualization. On the basis of its characteristics of reducing the dimensionality while retaining the high-dimensional relationship, the SOM algorithm is developed by first applying SOM dimensionality reduction, and then performing *K*-means clustering. Vesanto and Alhoniemi (2000) showed that clustering the SOM projections performs well compared with clustering the original data. In our study, the SOM has a dimension of 50×50. There are no strict guidelines for the selection of SOM dimension. The dimension of SOM is generally selected such that there is a balance between the quality/quantity of clusters and the computational time required for the clustering. This is performed by testing different dimensions. We observed that when the dimension of SOM is too small (e.g., 5×5), the *K*-means clustering, after the SOM, groups most of the data together into one cluster. We also observed that the computation time dramatically increases for SOM dimension greater than 50×50. The SOM was initialized by random weight vectors. During training, the weight vectors are updated according to the similarity between the weight vectors and input vectors. The similarity during the update is evaluated depending on the Euclidean distance. The clustering method is basically a *K*-means clustering performed on the mapping result of SOM. The mapping might result in the alteration of information in the original data set. The result of SOM clustering does not have a strong correlation with the prediction RE of the six models, which can be easily seen in Figs. 8 and 12d.

Comparison of Clustering Results. To better visualize the clustering results and relate them to petrophysical properties, we use the t-SNE dimensionality-reduction algorithm (van der Maaten and Hinton 2008). The t-SNE method was implemented to project the input logs (features) to two dimensions. Principal-component analysis and t-SNE are two commonly used dimensionality-reduction methods. Principal-component analysis performs dimensionality reduction by iteratively finding orthogonal directions of maximum variance in the data and then eliminating the orthogonal directions with low variances, whereas t-SNE performs nonlinear dimensionality reduction where the original data are projected to a lower-dimensional space while maintaining the original topology of data. The t-SNE algorithm is a recently popular nonlinear dimensionality-reduction algorithm for visualizing high-dimensional data. Dimension reduction enables us to plot the input logs as points while preserving their topological relationships in a high-dimensional space when transformed to a low dimension.

The basic principle is to compare the similarity between two data points and construct probability using similarity. When projecting the data points into a lower space, the t-SNE algorithm constructs a similar probability distribution when minimizing the difference between the original and low-dimensional distributions. If a data point is similar to another in a high-dimensional space, then it is very likely to be picked as a neighbor in a lower space. The algorithm has been applied to visualize data sets such as figures (van der Maaten and Hinton 2008), text (Heuer 2016), and music (Huang and Wu 2016) according to similarity.

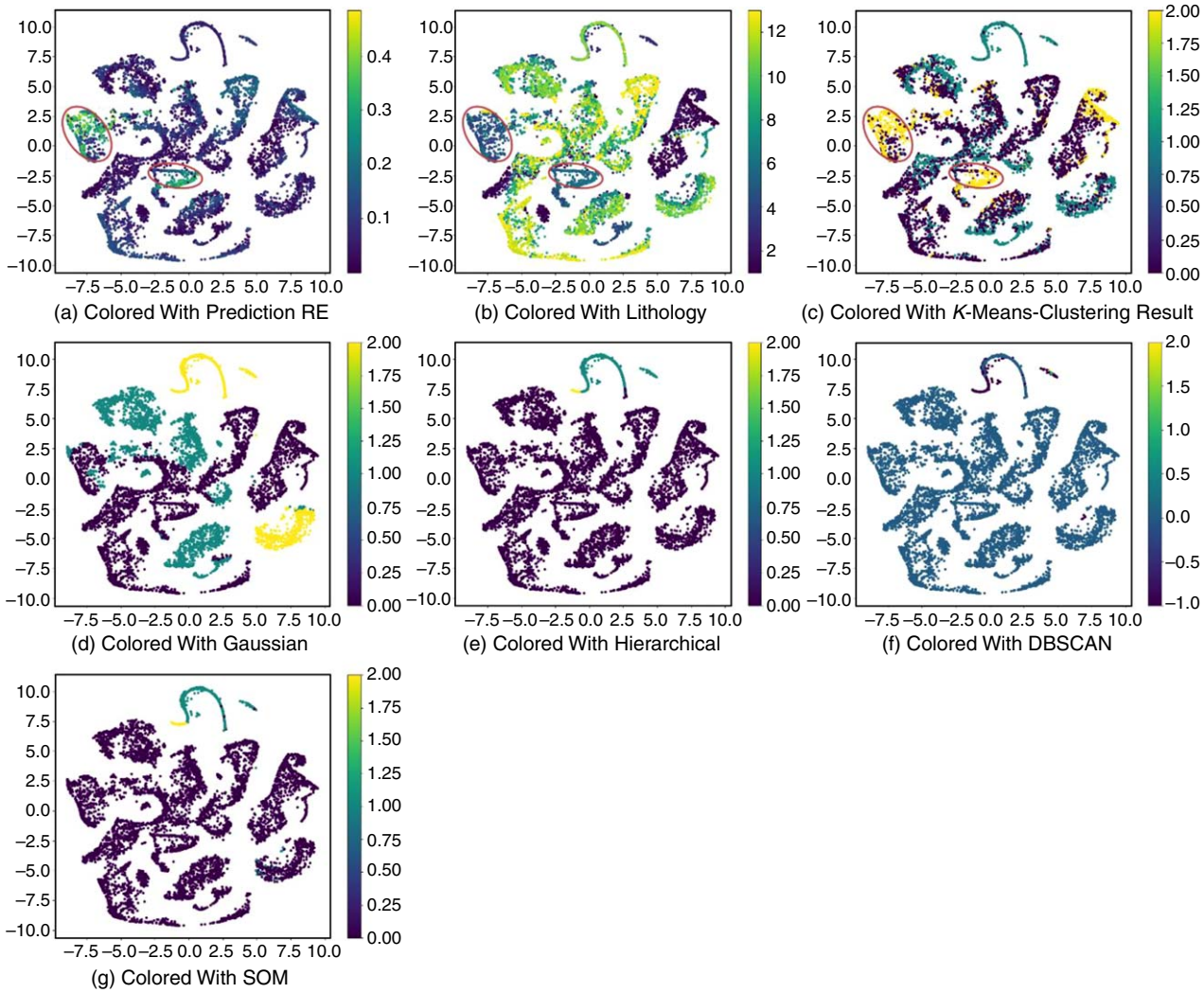

Fig. 13—Visualizing input logs using the t-SNE dimensionality-reduction algorithm colored with clustering results. Each point represents a layer, and points near each other share more similarity. Red circles denote formations with a high RE (mostly greater than 0.3) when applying six predictive models.

Perplexity and the number of training steps are two critical parameters for the t-SNE algorithm. Perplexity defines the number of neighbors, and it usually ranges from 5 to 50 and needs to be larger for a large data set (Wattenberg et al. 2016). There are no strict rules to select values of perplexity and training steps. On the basis of experience, the author of the algorithm (van der Maaten and Hinton 2008) provides a common range of the perplexity (5 to 50). In our study, we tested a range of values and selected the perplexity and the training steps to be 100 and 5,000. The results are shown in Fig. 13.

Fig. 13 plots the results of the t-SNE dimensionality reduction together with prediction RE or clustering results. Fig. 13 has seven subplots, and each plot uses the same manifold from t-SNE but is colored with different information, such as prediction RE, lithology, and cluster numbers obtained from various clustering methods. Each point on the plots represents a formation layer. The t-SNE algorithm projects input logs as a point on the plots. Input logs that are similar to each other will be projected as neighbors. On the plots, points are divided into several blocks. Data points in the same block share the most similarity. The shapes of the blocks are random and might change when the algorithm is applied using different parameters. The shape and position of blocks of the t-SNE dimensionality-reduction results might change, but the data points contained in each block will be the same. The algorithm will still project data points with similar properties to the same block when we run the algorithm again. The x-axis and y-axis of t-SNE figures are locations of points in the low dimension (two dimensions in our case) nonlinearly transformed from the high dimension. In consequence, the x-axis and y-axis of t-SNE figures do not have specific meanings and the t-SNE algorithm is a tool for better visualization of high-dimensional data.

Fig. 13a is colored with the RE of log synthesis performed using the ANN prediction model. In Fig. 13a, formations with higher REs are concentrated in the two blocks highlighted inside red circles. Compared with Fig. 13b, the data points with low prediction accuracy are mostly from Formations 5 and 8. For other layers, the prediction REs are mostly lower than 0.1. Comparing Figs. 13a and 13b, the prediction RE has a similar pattern with the lithology. Figs. 13c through 13g are colored with the group numbers computed using different clustering algorithms. By comparing the clustering results from Figs. 13c through 13g, we can obtain the characteristics of each clustering algorithm. Figs. 13e through 13g are very similar to each other. Hierarchical clustering, DBSCAN, and SOM generate very similar clustering results. Most of the data are clustered into one group, and the rest of the data are clustered into two other groups. Data points in the two smaller clusters are in the upper-middle position in the plots. Interestingly, the DBSCAN algorithm identifies data points in the position as outliers. Hierarchical clustering, DBSCAN, and SOM are sensitive to the outliers. These algorithms can identify outliers as one cluster and the normal data as another cluster.

K-means-clustering results and Gaussian-mixture-clustering results are shown in Figs. 13c and 13d, respectively. Both algorithms show good differentiation ability on the input logs. The Gaussian-mixture-clustering results are closely related to the t-SNE dimensionality-reduction results. Different clusters of the Gaussian-mixture model are assigned to different blocks in the t-SNE dimensionality-reduction results because t-SNE keeps the high-dimensional relationship of the data set when conducting dimensionality reduction. On average, different clusters in the Gaussian-mixture model coincide with different blocks on the plots. This might be because both algorithms rely on probability distribution to model the data. However, the results from the Gaussian-mixture model do not share a similar pattern with the lithology and RE. The *K*-means-clustering algorithm has the best correlation with the prediction RE. For the data points that have a high RE when applying the prediction models (indicated by the red circle), the *K*-means-clustering algorithm clusters most of them into Group 2. The *K*-means-clustering results are not 100% accurate but show the general trend. The *K*-means-clustering results have a pattern very close to that of the lithology plot in Fig. 13b.

Conclusions

This study applied six regression-type shallow-learning models to synthesize the compressional- and shear-travel-time logs in a shale reservoir. These shallow models were trained to process 13 conventional "easy-to-acquire" logs for synthesizing the compressional- and shear-travel-time logs. Five unsupervised-clustering methods were then applied to facilitate the assessment of the log-synthesis performance of the shallow models in new wells. The shallow-supervised-learning models exhibit good log-synthesis performance, and the ANN is the best-performing model with R^2 of 0.85. When ANN models are deployed in new formations, the *K*-means-clustering method can generate a reliability indicator to assess the log-synthesis performances of the shallow-learning models. Unlike *K*-means clustering, other clustering methods such as hierarchical clustering, DBSCAN, and SOM clustering cannot be consistently used to indicate the reliability of the log synthesis. The Gaussian-mixture model for clustering can differentiate the formations into robust clusters, but the clusters do not correlate with the REs in log synthesis. To apply the method described in this study, the users can train an ANN model with the same input and output logs. The accuracy of the ANN model can be calculated and used to build a *K*-means-clustering model to indicate the reliability of the ANN prediction. The ANN and the *K*-means-clustering model together will help engineers generate sonic logs and indicate whether the generated sonic logs are reliable. In summary, supervised-learning models, such as ANNs, can improve geomechanical characterization under data constraint by synthesizing the compressional- and shear-travel-time logs, and the *K*-means-clustering model can assess the reliability of the ANN-based log synthesis in new wells.

Nomenclature

B_q = basis function
R^2 = correlation coefficient
$\boldsymbol{w}$ = coefficient vector
x = input variable
$\boldsymbol{X}$ = input vector
y = actual output variable
$\bar{y}$ = averaged output variable
$\boldsymbol{Y}$ = output vector
α_q = coefficient of B_q
α_1 = penalty parameter
α_2 = penalty parameter
ε_i = error
λ = penalty parameter
ϕ_i = component weight
$\mathcal{N}$ = Gaussian distribution

References

Abdi, H. 2010. Partial Least Squares Regression and Projection on Latent Structure Regression (PLS Regression). *Wiley Interdiscip Rev Comput Stat* **2** (1): 97–106. https://doi.org/10.1002/wics.51.

Akinnikawe, O., Lyne, S., and Roberts, J. 2018. Synthetic Well Log Generation Using Machine Learning Techniques. Presented at the SPE/AAPG/SEG Unconventional Resources Technology Conference, Houston, 23–25 July. URTEC-2877021-MS. https://doi.org/10.15530/URTEC-2018-2877021.

Al-Bulushi, N., Araujo, M., Kraaijveld, M. et al. 2007. Predicting Water Saturation Using Artificial Neural Networks (ANNs). Presented at the SPWLA Middle East Regional Symposium, Abu Dhabi, 15–19 April. SPWLA-MERS-2007-W.

Alexeyev, A., Ostadhassan, M., Mohammed, R. A. et al. 2017. Well Log Based Geomechanical and Petrophysical Analysis of the Bakken Formation. Presented at the 51st US Rock Mechanics/Geomechanics Symposium, San Francisco, 25–28 June. ARMA-2017-0942.

Ao, Y., Li, H., Zhu, L. et al. 2018. Logging Lithology Discrimination in the Prototype Similarity Space with Random Forest. *IEEE Geosci Remote Sens Lett* **16** (5): 687–691. https://doi.org/10.1109/LGRS.2018.2882123.

Asoodeh, M. and Bagheripour, P. 2012. Prediction of Compressional, Shear, and Stoneley Wave Velocities from Conventional Well Log Data Using a Committee Machine with Intelligent Systems. *Rock Mech Rock Eng* **45** (1): 45–63. https://doi.org/10.1007/s00603-011-0181-2.

Baines, V., Bootle, R., Pritchard, T. et al. 2008. Predicting Shear and Compressional Velocities in Thin Beds. Presented at the SPWLA 49th Annual Logging Symposium, Austin, Texas, 25–28 May. SPWLA-2008-I.

Beale, M. H., Hagan, M. T., and Demuth, H. B. 2012. *Neural Network Toolbox™ 7 User's Guide*. Natick, Massachusetts: The MathWorks, Inc.

Birant, D. and Kut, A. 2007. ST-DBSCAN: An Algorithm for Clustering Spatial–Temporal Data. *Data Knowl Eng* **60** (1): 208–221. https://doi.org/10.1016/j.datak.2006.01.013.

Chumney, E. C. G. and Simpson, K. N. 2006. *Methods and Designs for Outcomes Research*, Bethesda, Maryland: American Society of Health-System Pharmacists.

Elkatatny, S. M., Zeeshan, T., Mahmoud, M. et al. 2016. Application of Artificial Intelligent Techniques to Determine Sonic Time from Well Logs. Presented at the 50th US Rock Mechanics/Geomechanics Symposium, Houston, 26–29 June. ARMA-2016-755.

Friedman, J. H. 1991. Multivariate Adaptive Regression Splines. *Ann. Statist.* **19** (1): 1–67. https://doi.org/10.1214/aos/1176347963.

Greenberg, M. L. and Castagna, J. P. 1992. Shear-Wave Velocity Estimation in Porous Rocks: Theoretical Formulation, Preliminary Verification and Applications. *Geophys Prospect* **40** (2): 195–209. https://doi.org/10.1111/j.1365-2478.1992.tb00371.x.

Handwerger, D. A., Keller, J., and Vaughn, K. 2011. Improved Petrophysical Core Measurements on Tight Shale Reservoirs Using Retort and Crushed Samples. Presented at the SPE Annual Technical Conference and Exhibition, Denver, 30 October–2 November. SPE-147456-MS. https://doi.org/10.2118/147456-MS.

Heuer, H. 2016. Text Comparison Using Word Vector Representations and Dimensionality Reduction. arXiv preprint arXiv:1607.00534.

Huang, A. and Wu, R. 2016. Deep Learning for Music. arXiv preprint arXiv:1606.04930.

Iverson, W. P. and Walker, J. N. 1988. Shear and Compressional Logs Derived from Nuclear Logs. In *SEG Technical Program Expanded Abstracts 1988*, 111–113. Tulsa: Society of Exploration Geophysicists. https://doi.org/10.1190/1.1892237.

Jain, V., Gzara, K., Makarychev, G. et al. 2015. Maximizing Information Through Data Driven Analytics in Petrophysical Evaluation of Well Logs. Presented at the SPE Annual Technical Conference and Exhibition, Houston, 28–30 September. SPE-174735-MS. https://doi.org/10.2118/174735-MS.

Keys, R. G. and Xu, S. 2002. An Approximation for the Xu-White Velocity Model. *Geophysics* **67** (5): 1406–1414. https://doi.org/10.1190/1.1512786.

Kim, S.-J., Koh, K., Lustig, M. et al. 2007. An Interior-Point Method for Large-Scale ℓ_1-Regularized Least Squares. *IEEE J Sel Top Signal Process* **1** (4): 606–617. https://doi.org/10.1109/JSTSP.2007.910971.

Kohonen, T. 1990. The Self-Organizing Map. *Proc. IEEE* **78** (9): 1464–1480. https://doi.org/10.1109/5.58325.

Li, H. and Misra, S. 2017a. Prediction of Subsurface NMR T2 Distribution from Formation-Mineral Composition Using Variational Autoencoder. *SEG Technical Program Expanded Abstracts 2017*: 3350–3354. Tulsa: Society of Exploration Geophysicists. https://doi.org/10.1190/segam2017-17798488.1.

Li, H. and Misra, S. 2017b. Prediction of Subsurface NMR T2 Distributions in a Shale Petroleum System Using Variational Autoencoder-Based Neural Networks. *IEEE Geosci Remote Sens Lett* **14** (12): 2395–2397. https://doi.org/10.1109/LGRS.2017.2766130.

Lingjaerde, O. C. and Christophersen, N. 2000. Shrinkage Structure of Partial Least Squares. *Scand Stat Theory Appl* **27** (3): 459–473. https://doi.org/10.1111/1467-9469.00201.

Maleki, S., Moradzadeh, A., Riabi, R. G. et al. 2014. Prediction of Shear Wave Velocity Using Empirical Correlations and Artificial Intelligence Methods. *NRIAG-JAG* **3** (1): 70–81. https://doi.org/10.1016/j.nrjag.2014.05.001.

Moghaddas, H., Habibnia, B., Ghasemalaskari, M. K. et al. 2017. Lithofacies Classification Based on Multiresolution Graph-Based Clustering Using Image Log in South Pars Gas Field. Presented at the 2017 SEG International Exposition and Annual Meeting, Houston, 24–29 September. SEG-2017-17724469.

Møller, M. F. 1993. A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. *Neural Netw* **6** (4): 525–533. https://doi.org/10.1016/S0893-6080(05)80056-5.

Pedregosa, F., Varoquaux, G., Gramfort, A. et al. 2011. Scikit-Learn: Machine Learning in Python. *J Mach Learn Res* **12** (February): 2825–2830.

Petriello, J., Marino, S., Suarez-Rivera, R. et al. 2013. Integration of Quantitative Rock Classification with Core-Based Geologic Studies: Improved Regional-Scale Modeling and Efficient Exploration of Tight Shale Plays. Presented at the SPE Unconventional Resources Conference and Exhibition-Asia Pacific, Brisbane, Australia, 11–13 November. SPE-167048-MS. https://doi.org/10.2118/167048-MS.

Qin, X., Xu, Y., Yan, H. et al. 2017. Unsupervised Well Clustering: Pattern Recognition in Overpressure Mechanisms. Presented at the 2017 SEG International Exposition and Annual Meeting, Houston, 24–29 September. SEG-2017-17797818.

Rezaee, M. R., Ilkhchi, A. K., and Barabadi, A. 2007. Prediction of Shear Wave Velocity from Petrophysical Data Utilizing Intelligent Systems: An Example from a Sandstone Reservoir of Carnarvon Basin, Australia. *J Pet Sci Eng* **55** (3–4): 201–212. https://doi.org/10.1016/j.petrol.2006.08.008.

Shi, X., Cui, Y., Guo, X. et al. 2017. Logging Facies Classification and Permeability Evaluation: Multi-Resolution Graph Based Clustering. Presented at the SPE Annual Technical Conference and Exhibition, San Antonio, Texas, 9–11 October. SPE-187030-MS. https://doi.org/10.2118/187030-MS.

Tan, P.-N., Steinbach, M., and Kumar, V. 2006. *Introduction to Data Mining*. Boston, Massachusetts: Pearson.

Tariq, Z., Elkatatny, S., Mahmoud, M. et al. 2016. A New Artificial Intelligence Based Empirical Correlation to Predict Sonic Travel Time. Presented at the International Petroleum Technology Conference, Bangkok, 14–16 November. IPTC-19005-MS. https://doi.org/10.2523/IPTC-19005-MS.

Tibshirani, R. 1996. Regression Shrinkage and Selection via the Lasso. *J R Stat Soc Series B Stat Methodol* **58** (1): 267–288. https://www.jstor.org/stable/2346178.

van der Maaten, L. and Hinton, G. 2008. Visualizing Data Using t-SNE. *J Mach Learn Res* **9** (November): 2579–2605.

Vesanto, J. and Alhoniemi, E. 2000. Clustering of the Self-Organizing Map. *IEEE Trans Neural Netw* **11** (3): 586–600. https://doi.org/10.1109/72.846731.

Wang, B., Zhou, F., Zou, Y. et al. 2019. Quantitative Investigation of Fracture Interaction by Evaluating Fracture Curvature During Temporarily Plugging Staged Fracturing. *J Pet Sci Eng* **172** (January): 559–571. https://doi.org/10.1016/j.petrol.2018.08.038.

Wattenberg, M., Viégas, F., and Johnson, I. 2016. How to Use t-SNE Effectively. *Distill* (13 October), https://doi.org/10.23915/distill.00002.

Wold, H. 2004. Partial Least Squares. In *Encyclopedia of Statistical Sciences*, second edition, ed. S. Kotz, C. B. Read, N. Balakrishnan, et al., Vol. 9. Hoboken, New Jersey: Wiley.

Yegnanarayana, B. 2009. *Artificial Neural Networks*. New Delhi, India; Prentice Hall of India Private Limited.

Zou, H. and Hastie, T. 2005. Regularization and Variable Selection via the Elastic Net. *J R Stat Soc Series B Stat Methodol* **67** (2): 301–320. https://www.jstor.org/stable/3647580.

Jiabo He is currently a PhD-degree candidate in computer science at the University of Melbourne, Australia. As a graduate research assistant with coauthor Siddharth Misra at the University of Oklahoma, he worked on shallow-learning methods for sonic-log generation, applying machine-learning models to deploy in wells drilled within the reservoir. He holds a master's degree in petroleum engineering from the University of Oklahoma and a bachelor's degree in petroleum engineering from China University of Petroleum, Beijing.

Hao Li is currently a PhD-degree candidate in petroleum engineering at the University of Oklahoma, researching machine-learning applications in well logging and pressure/rate-transient analysis. He holds master's and bachelor's degrees in petroleum engineering, both from China University of Petroleum, Beijing.

Siddharth Misra is an associate professor with the Harold Vance Department of Petroleum Engineering, Texas A&M University, researching data-driven predictive models, machine learning, geosensors, and front-tracking methods. He holds a PhD degree in petroleum engineering from the University of Texas at Austin and a bachelor's degree in electrical engineering from the Indian Institute of Technology.