Support-Vector Regression for Permeability Prediction in a Heterogeneous Reservoir: A Comparative Study

A. Al-Anazi and **I.D. Gates,** University of Calgary

Summary

Permeability is a key parameter in reservoir-engineering computation, and the relationship between rock petrophysical properties and permeability is often complex and difficult to understand by using conventional statistical methods. Neural-network-based methods can be employed to develop more-accurate permeability correlations, but the correlations from these methods have limited generalizability and the global correlations are usually less accurate compared to local correlations. In this research, the objective is to build a permeability model with promising generalization performance. Recently, support-vector machines (SVMs) based on statistical-learning theory have been proposed as a new intelligence technique for both prediction and classification tasks. The formulation of SVMs embodies the structural-risk-minimization (SRM) principle, which has been shown to be superior to the traditional empirical-risk-minimization (ERM) principle employed by conventional neural networks. This new formulation deals with kernel functions, allows projection to higher planes, and solves more-complex nonlinear problems. SRM minimizes an upper bound on the expected risk, as opposed to ERM, which minimizes the error on the training data. It is this difference that equips SVMs with a greater ability to generalize, which is the goal in reservoir-characterization statistical learning. This novel support-vector-regression (SVR) algorithm was first introduced in well-logs intelligent analysis. Here, a permeability-prediction model using SVR from well logs in a heterogeneous sandstone reservoir is developed. Also, an attempt has been made to review the basic ideas underlying support-vector machines for function estimation. To demonstrate the potential of the proposed SVM's regression technique in prediction permeability, a study was performed to compare its performance with multilayer perceptron neural network, generalized neural network, and radial-basis-function neural networks. Accuracy and robustness were investigated, and statistical-error analysis reveals that the SVM approach is superior to the other methods for generalizing previously unseen permeability data.

Introduction

It is of great importance for geoscientists to develop a permeability regression model that exhibits high accuracy, robustness, computational efficiency, and transparency. Geoscience data, sourced from log data for example, is complex, nonlinear, and contains a great deal of uncertainty. Furthermore, the data are polluted with noise caused by measurement errors.

Reservoirs with high degrees of heterogeneity reflect a nonlinear system-identification problem, which is a crucial but complex problem. There have been several published studies on modeling permeability on the basis of empirical correlations, multilinear regression (MLR), multilayer perception (MLP), and fuzzy neural networks (FNNs) (Huang et al. 1996; Huang et al. 2001; Cuddy 2000; Taghavi 2005). One of the main shortcomings of empirical models is that they are constructed by using all available data, and prediction accuracy is good within the input data set. Therefore, these models have poor generalization capability. Although MLR models perform better on unseen data (data not included in the training part of the process), they tend to overestimate low values and underestimate high values, showing their consistent characteristic of averaging the entire data to yield reasonable values for statistical indicators (Al-Anazi et al. 2009). These drawbacks can lead to significant errors when modeling permeability in heterogeneous reservoirs. Artificial neural networks (ANNs) have been used in permeability prediction and have shown higher generalization performance. However, ANNs are a black-box approach in which little can be understood about the model because an exact function is not generated; there are variable numbers and sizes of hidden layers and several types of transfer functions in each layer and learning algorithm parameters, such as initial weights and learning rate, that are variable. Also, an overfitting problem during data training frequently arises, and, as a consequence, the result is potentially a poor generalization model. The learning process of neural networks is based on least-squares mean-error minimization, which can also get trapped in local minima, which can also lead to large prediction errors.

SVMs are learning machines that implement the SRM inductive principle to obtain good generalization on a limited number of learning patterns. The SVMs based on the structure-risk-minimum principle have better generalizability than other approximation methods of nonlinear function. SRM aims to minimize the error bound so as to achieve improved performance through simultaneous attempts to minimize empirical risk and a regularization term that controls the complexity of the hypothesis space. The SVM methodology is a new learning machine proposed by Vapnik et al. (1995), which has been applied successfully to pattern recognition, regression estimation, signal processing, system identification, and other areas. SVMs are known to produce results as good as those of neural networks, if not better, while being computationally less costly and producing an actual mathematical function. In this research, SVR, a particular implementation of SVMs, is investigated as an alternative technique to predict permeability in a heterogeneous reservoir. SVR has strong nonlinear approximation capabilities and good generalization performance. It can avoid the overfitting problem and the local-minima problem suffered by neural networks (Cristianini and Shawe-Taylor 2000).

Here, the prediction performance of SVMs, including different kernel functions to predict permeability along well trajectories, is compared to the results obtained from an MLP neural network (MLPNN), a generalized neural network (GRNN), and a radial-basis-function neural network (RBFNN). Kernel-function-parameter optimization was performed by using grid-search and pattern-search schemes. Several different error measures, including correlation coefficient *r*, between actual and predicted values—root-mean-square error (RMSE), absolute-average error (AAE), and maximum absolute error (MAE)—were used to compare the methods. Both accuracy and robustness were investigated through several different well-training data sets.

Existing Regression Techniques

This section briefly presents an overview of the three regression techniques against which SVR is compared: MLPNN, GRNN, and RBFNN.

These techniques were chosen on the basis of their widespread use in modeling engineering data.

Original SPE manuscript received for review 12 May 2009. Revised manuscript received for review 23 September 2009. Paper (SPE 126339) peer approved 1 October 2009.

MLPNN Model. ANNs usually refer to MLPNNs. ANN methods learn by using nonlinear optimization techniques. These methods do not require *a priori* selection of the mathematical model. Instead, the underlying relationships that exist between input and output variables are determined automatically, revealing that ANNs represent an ideal choice for problems whose exact functional form among variables is unknown. ANNs can be used to find a mathematical model that relates multidimensional input variables to output data (i.e., it might be regarded as a multivariate regression tool). The basic idea is to provide training patterns that will direct adjustments to the ANN weight matrix parameters (Haykin 1999).

In the most popular ANN learning algorithm, called back propagation (Rumelhart et al. 1986), the inputs are fed forward from the input layer through the hidden layers, and, finally, the network provides its output result. The training process consists of estimating the weights that minimize the square of the deviation between the output result and the actual data. The calculated error is then propagated backward through the network, and the weights are automatically adjusted. The ANN approach has several advantages over conventional statistical and deterministic approaches. The most important one is its ability to mimic complex nonlinear models without *a priori* knowledge of the underlying model; i.e., that it is free from the constraints of a specification of structural relationships between inputs and outputs. In contrast to linear-regression models, it does not force predicted values to lie near the mean values, and, thus, it preserves the actual variability in the data (Rogers et al. 1995).

In this research, the ANN model consisted of three network layers. The Nguyen-Widrow (Sherrod 2009) algorithm was used to select the initial range of weight values, and the conjugate gradient algorithm was used to optimize the weights. To avoid overfitting the data, cross validation was adopted to select the best model.

GRNN Model. A GRNN provides regression estimates of continuous variables and converges to the underlying (linear or nonlinear) regression surface. The estimate $f(\mathbf{x})$, where $\mathbf{x}$ is an input vector, can be visualized as a weighted average of all of the observed values, y, where each observed value is weighted by a Gaussian radial-basis function. The kernel function is weighted according to its Euclidean distance from x. GRNNs have exactly four layers: input layer, hidden layer, pattern layer, and decision layer. The input layer has an equal number of nodes as input variables and standardizes the range of input values and then feeds these values to each of the neurons in the hidden layer. The hidden layer has one neuron for each pattern in the training data set and stores the values of the predictor variables for the pattern along with the target value. It computes the Euclidean distance of the test pattern from the neuron's center point and then applies the radial-basis function (RBF) kernel function by using the variances σ. The Euclidean distance is passed to the pattern layer, where there exist only two neurons, the denominator summation and numerator summation. The denominator summation operation adds up the weight values, whereas the numerator summation operation adds up the weight values multiplied by the actual target value for each hidden neuron. The decision layer computes the ratio of the numerator and denominator, and the output gives the predicted target value. Training is performed by using the conjugate gradient algorithm to select the optimal variances to control the spread of the RBF functions. GRNNs have some advantages over MLPNNs in terms of training speed, accuracy, and insensitivity to outliers (Sherrod 2009). On the other hand, they are slower than MLPNNs at classifying new patterns and require more memory space to store the model, although there are methods to overcome such shortcomings (Specht 1991).

RBFNN Model. RBFNNs have an input layer, a hidden layer, and an output layer. The input layer has one neuron for each predictor variable. The hidden layer has a number of neurons that are optimized by the training process because these neurons sample the domain, and the hidden neurons perform nonlinear transformation of the input space where Gaussian transfer functions are used. During testing, a hidden neuron computes the Euclidean distance of the test pattern from the neuron's center point and then applies the RBF kernel function to this distance using the spread values. The result is passed to the summation layer. Each value is then multiplied by a weight associated with the neuron, and the sum presents the output of the network. An evolutionary training algorithm is used to fit an RBF function by determining the optimal center points and variance of the RBF function for each neuron. In contrast to GRNN, which has one neuron for each point in the training record, RBFNNs have a variable number of neurons that is usually much less than the number of training points (Chen et al. 1991; Sherrod 2009).

In the research documented here, a cross-validation method was adopted to determine the number of neurons to eliminate overfitting. During this process, the data are partitioned into several different splits where the model performance is assessed iteratively and the model with the minimum average error is selected. Similarly, ridge regression was used to determine the optimum weight connecting the hidden and the output layer.

SVR

Statistical Learning. Statistical-learning theory provides a solid mathematical framework to minimize risk in regression and classification models. It makes the assumption that the data to be represented are generated by sampling from an unknown underlying distribution. Also, statistical learning typically assumes that training examples are generated identically and independently according to the distribution and that a finite set of input values together with corresponding output values are given. Indeed, these pairs of samples are used to learn the unknown functional relationship between input and output values by an algorithm that extracts a prediction rule for output values for previously unseen input data. This is a fundamental difference from parametric models where the relationship between the inputs and outputs are assumed to follow some unknown function from a known set of functions (Steinwart and Christman 2008).

Statistical learning constructs a function that estimates a response with the smallest possible risk with an unknown data distribution. Additional information in the form of input/output data pairs is then required where all data are assumed to be generated from the same distribution. Statistical-learning theory has provided a very effective framework for both pattern-recognition and regression-estimation problems. On the basis of this statistical framework, an SVM model is derived by solving a constrained quadratic problem where the convex objective function to be minimized is given by the combination of a loss function with a regularization term that is related to the dimension of the hypothesis space (Basak et al. 2007).

SVM Methodology. The SVM algorithm is a nonlinear generalization of the generalized portrait algorithm developed in Russia in the early 1960s (Vapnik and Lerner 1963; Vapnik and Chervonenkis 1964). Vapnik–Chervonenkis (VC) theory, developed by Vapnik and Chervonenkis (1974) and Vapnik (1982; 1995), characterizes properties of learning machines that enable them to generalize well to previously unseen examples. SVM was further developed at AT&T Bell Laboratories by Vapnik and coworkers (Boser et al. 1992; Guyon et al. 1993; Cortes and Vapnik 1995; Schölkopf et al. 1995, 1996; Vapnik et al. 1997).

There are two main categories for SVMs—support-vector classification and support-vector regression. SVM is a learning system that uses a high-dimensional feature space to construct prediction functions that span a subset of support vectors. Originally, SVM was developed to solve pattern-recognition problems. However, with the introduction of Vapnik's ε-insensitive loss function, SVM has been extended to solve nonlinear regression estimation problems; these methods are referred to as support-vector regression. A regression-estimation problem is concerned with estimating real-valued functions, such as approximating the set of the following data, expressed by $D = [(x^1, y^1), \ldots, (x^l, y^l)]$, $x \in R^n, y \in R$, by a linear function given by

$$f(\mathbf{x}) = \langle \boldsymbol{\omega}, \mathbf{x} \rangle + b,\ \boldsymbol{\omega} \in X, b \in R, \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

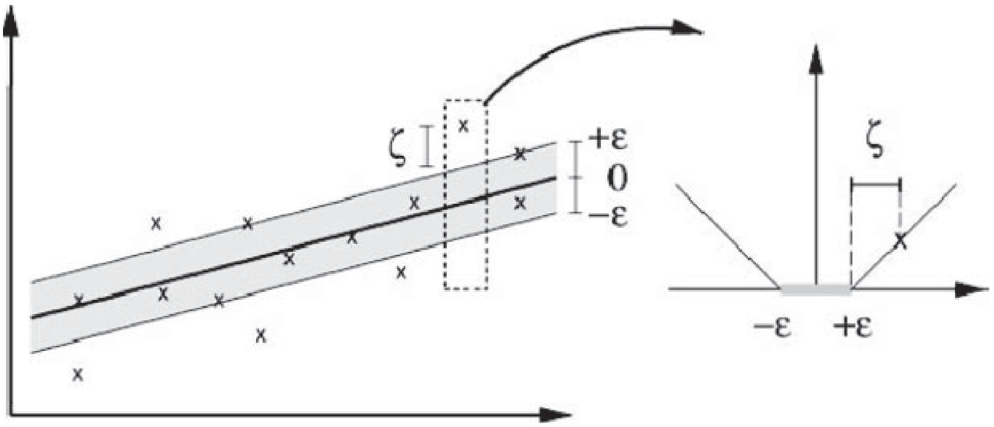

Fig. 1—Soft margin loss setting for a linear SVR (Schölkopf and Smola 2002).

where $\langle \boldsymbol{\omega}, \mathbf{x} \rangle$ denotes the dot product between the two vectors $\boldsymbol{\omega}$ and $\mathbf{x}$. In ε-SVR, the goal is to find a function $f(\mathbf{x})$ that first estimates output values with a deviation less than ε from the actual targets of all the training data and, second, is as flat as possible. Function flatness is related to how small the vector $\boldsymbol{\omega}$ is, and it is achieved by minimizing the Euclidean norm of $\boldsymbol{\omega}$ (i.e., $\|\boldsymbol{\omega}\|^2$). The optimal regression function is determined by solving the following optimization problem to estimate $\boldsymbol{\omega}$ and b:

$$\text{minimize} \quad \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{l}\left(\xi_i + \xi_i^*\right),$$

$$\text{subject to} \quad \begin{cases} y_i - \langle \boldsymbol{\omega}, \mathbf{x}_i \rangle - b \le \varepsilon + \xi_i \\ \langle \boldsymbol{\omega}, \mathbf{x}_i \rangle + b - y_i \le \varepsilon + \xi_i^*, \\ \xi_i, \xi_i^* \ge 0 \end{cases} \quad \text{.................. (2)}$$

where ξ_i and ξ_i^* are slack variables introduced to penalize complex fitting functions. Therefore, SVR fits a function to the given data not only by minimizing the training error. The first term of the objective function is the VC confidence interval term, whereas the second term represents the empirical risk. Both terms encompass the principle of SRM whose objective is to limit an upper bound of the generalization error rather than limit the training error. On the basis of this principle, SVM achieves an optimum structure by striking the balance between empirical error and VC-confidence interval. This leads to better generalization performance than other neural-network models (Peng et al. 2004). The constant C allows penalizing the error by determining the tradeoff between training-error minimization and model complexity, and this penalty is acceptable only if the fitting error is greater than 0, chosen *a priori*. Therefore, a regression-estimation model with a small risk is characterized by controlling both training error and model complexity by using a simple model to explain the given data. The ε-insensitivity loss function $|\xi|_\varepsilon$ is defined as

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \le \varepsilon \\ |\xi| - \varepsilon & \text{otherwise} \end{cases}. \quad \text{.......................... (3)}$$

The ε-insensitivity loss function stabilizes estimation and can be visualized as a tube-sized equivalent to the approximation accuracy in training data, as shown in **Fig. 1.** The parameters C and ε in the objective function are user-defined parameters and can be selected automatically by using search algorithms. It turns out, in most cases, that the optimization problem given by Eq. 2 can be solved more easily in its dual form; that is, a Lagrangian function is constructed from the objective function and constraints by introducing a dual set of variables:

$$\begin{aligned} L = \frac{1}{2}\|\boldsymbol{\omega}\|^2 &+ C\sum_{i=1}^{l}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{l}\left(\eta_i \xi_i + \eta_i^* \xi_i^*\right) \\ &- \sum_{i=1}^{l}\alpha_i\left(\varepsilon + \xi_i - y_i + \langle \boldsymbol{\omega}, \mathbf{x}_i \rangle + b\right) \\ &- \sum_{i=1}^{l}\alpha_i^*\left(\varepsilon + \xi_i^* - y_i - \langle \boldsymbol{\omega}, \mathbf{x}_i \rangle - b\right), \quad \text{........... (4)} \end{aligned}$$

where L is the Lagrangian and η_i, η_i^*, α_i, and α_i^* are Lagrange multipliers. The partial derivatives of L with respect to the primal variables $\left(\boldsymbol{\omega}, b, \xi_i, \xi_i^*\right)$ vanish at the optimal solution. That is,

$$\partial_b L = \sum_{i=1}^{l}\left(\alpha_i^* - \alpha_i\right) = 0, \quad \text{............................ (5)}$$

$$\boldsymbol{\omega} - \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)\mathbf{x}_i = 0, \quad \text{............................ (6)}$$

$$\partial_{\xi_i^{(*)}} L = C - \alpha_i^{(*)} - \eta_i^{(*)} = 0. \quad \text{.......................... (7)}$$

Substituting Eqs. 5, 6, and 7 into Eq. 4 generates the following dual optimization problem:

$$\begin{aligned} \text{maximize} \quad & \frac{1}{2}\sum_{i,j=1}^{l}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)\langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ & -\varepsilon\sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right) + \sum_{i=1}^{l} y_i\left(\alpha_i - \alpha_i^*\right) \end{aligned}$$

$$\text{subject to} \quad \begin{cases} \sum_{i=1}^{l}\left(\alpha_i^* - \alpha_i\right) \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases}. \quad \text{........................ (8)}$$

Eq. 6 can be reformulated as

$$\boldsymbol{\omega} = \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)\mathbf{x}_i. \quad \text{............................... (9)}$$

The regression estimation model becomes

$$f(\mathbf{x}) = \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)\langle \mathbf{x}_i, \mathbf{x} \rangle + b. \quad \text{...................... (10)}$$

This is the so-called support-vector expansion (i.e., the function can be described as a linear combination of the training patterns,

TABLE 1—COMMON KERNEL FUNCTION AND CORRESPONDING MATHEMATICAL EXPRESSION

Kernel Function	Mathematical Expression
Linear	$k(\mathbf{x}_i,\mathbf{x})=\langle\mathbf{x}_i,\mathbf{x}\rangle$
Gaussian radial basis function	$k(\mathbf{x}_i,\mathbf{x})=e^{-\frac{\|\mathbf{x}_i-\mathbf{x}\|^2}{2\ ^2}}$
Polynomial	$k(\mathbf{x}_i,\mathbf{x})=\langle\mathbf{x}_i,\mathbf{x}\rangle^d$
Sigmoid	$k(\mathbf{x}_i,\mathbf{x})=\tanh\ [\kappa(\mathbf{x}_i,\mathbf{x})+\vartheta]$

$\mathbf{x}_i$). Even for evaluating $f(\mathbf{x})$, the vector $\boldsymbol{\omega}$ is not needed explicitly. Computation of b is performed by exploiting the Karush-Kuhn-Tucker (KKT) conditions (Karush 1939; Kuhn and Tucker 1951; Schölkopf and Smola 2002), which state that, at the optimal solution, the product of dual variables and constraints has to vanish. In the support-vector case, this means

$$\begin{cases}\alpha_i\left(\varepsilon+\xi_i-y_i+\langle\boldsymbol{\omega},\mathbf{x}_i\rangle+b\right)=0\\ \alpha_i^*\left(\varepsilon+\xi_i^*-y_i+\langle\boldsymbol{\omega},\mathbf{x}_i\rangle+b\right)=0\end{cases}\quad\text{. . . . .(11)}$$

and

$$\begin{cases}\left(C-\alpha_i\right)\xi_i=0\\ \left(C-\alpha_i^*\right)\xi_i^*=0.\end{cases}\quad\text{. . . . .(12)}$$

On the basis of Eqs. 11 and 12, it can be concluded that only samples (x_i, y_i) with corresponding $\alpha_i^{(*)}=C$ lie outside the ε-insensitive tube. Also, $\alpha_i\alpha_i^*=0$ indicates that there can never be a set of dual variables α_i,α_i^* that are both simultaneously nonzero. Finally, for $\alpha_i^{(*)}\in(0,C)$, $\xi_i^{(*)}=0$, and the second term in Eq. 10 must vanish. Hence, b can be computed as follows:

$$\begin{cases}b=y_i-\langle\boldsymbol{\omega},\mathbf{x}_i\rangle-\varepsilon & \text{for } \alpha_{\mathrm{i}}\in(0,C)\\ b=y_i-\langle\boldsymbol{\omega},\mathbf{x}_i\rangle+\varepsilon & \text{for } \alpha_{\mathrm{i}}^*\in(0,C)\end{cases}.\quad\text{. . . . .(13)}$$

On the basis of Eq. 11, it can be shown that the Lagrange multipliers may be nonzero if $\left|f(\mathbf{x})-y_i\right|\geq\varepsilon$, indicating that all Lagrangian multipliers inside the ε-tube vanish. On the other hand, the second term in Eq. 11 is nonzero for $\left|y_i-f(\mathbf{x})\right|<\varepsilon$, indicating that the multipliers have to vanish to satisfy the KKT condition. This indicates the sparsity of this representation of SVR [i.e., only a fraction of the training data is needed to encode the original data and approximate the target function to achieve the capability that would be realized if all the data were used (Schölkopf and Smola 2002; Basak et al. 2007)].

The support-vector algorithm can solve nonlinear problems by preprocessing training patterns $\mathbf{x}_i\in X$ into a high-dimensional feature space H by a map $\boldsymbol{\Phi}$ to linearly estimate the regression. This, therefore, leads to the following optimization problem corresponding to finding the flattest function in the feature space:

$$\text{maximize } -\frac{1}{2}\sum_{i,j=1}^{l}\left(\alpha_i-\alpha_i^*\right)\left(\alpha_j-\alpha_j^*\right)k\left(\mathbf{x}_i,\mathbf{x}_j\right)$$
$$-\varepsilon\sum_{i=1}^{l}\left(\alpha_i+\alpha_i^*\right)+\sum_{i=1}^{l}y_i\left(\alpha_i-\alpha_i^*\right)$$

$$\text{subject to }\begin{cases}\sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)=0\\ \alpha_i,\alpha_i^*\in[0,C]\end{cases}\quad\text{. . . . .(14)}$$

Likewise, the expansion can be written as follows:

$$\boldsymbol{\omega}=\sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)\boldsymbol{\Phi}(\mathbf{x}_i),\quad\text{. . . . .(15)}$$

with

$$f(\mathbf{x})=\left(\alpha_i-\alpha_i^*\right)k\left(\mathbf{x}_i,\mathbf{x}\right)+b\cdot\quad\text{. . . . .(16)}$$

Here, $\boldsymbol{\omega}$ is implicitly given in the estimation function and Eq. 16 gives the standard ε-SVR estimation function.

Kernel Functions. The limited computational power in the original input space prevents linear-learning machines from solving complex real-world applications because the output cannot be expressed as a simple linear combination of the given input variables. Therefore, more-abstract features of the data are required to be exploited over more-expressive hypothesis spaces. Kernel representations project the data into a higher-dimensional feature space to increase the computational power of the linear-learning machine to solve nonlinear problems. Thus, the nonlinear regression function in an input space is constructed by considering a linear-regression hyperplane in the feature space. Therefore, to create a nonlinear regression function, the input vectors x are mapped into vectors $\boldsymbol{\Phi}(\mathbf{x})$ of a higher-dimensional feature space, and then a linear-regression problem is solved in this feature space. The mapping to the vectors $\boldsymbol{\Phi}(\mathbf{x})$ is required to be chosen *a priori* to solve a quadratic optimization problem with inequality constraints in the feature space (Cristianini and Shawe-Taylor 2000; Kecman 2005). The advantage of the dual representation is that the number of tunable parameters does not depend on the number of attributes being used. Thus, one can implicitly perform nonlinear mapping to a high-dimensional feature space by replacing the dot product in feature space directly as a function of the original input points with a suitable kernel function without increasing the number of tunable parameters. This implicitly maps the data into a feature space, potentially side-stepping the computational problems inherent in evaluating the feature map (Cristianini and Shawe-Taylor 2000).

The kernel function $k(\mathbf{x}_i, \mathbf{x})$ can be preprocessed, with results stored in the kernel matrix **K.** The kernel matrix must be positive definite to guarantee a unique optimal solution to the quadratic optimization problem. **Table 1** lists several kernel functions that yield positive definite matrices (Cristianini and Shawe-Taylor 2000). Thus, by using the kernel function and corresponding kernel matrix, nonlinear function approximations can be achieved with SVR while maintaining the simplicity and computational efficiency of linear SVR approximations.

Cost Functions. Learning methods such as SVM need a loss function, $L[x, y, f(\mathbf{x})]$, that describes the cost of discrepancy between prediction $f(\mathbf{x})$ and the actual observation y at Point x. L represents the cost of giving an approximation $f(\mathbf{x})$ in place of true value y. A risk is then associated to the loss by the average future-loss function. For a sequence of training data points, $D=\left[(x_1,y_1),...,(x_l,y_l)\right]\in(X\times Y)^n$, there exists

$$D=\frac{1}{l}\sum_{i=1}^{l}\delta_{(x_i,y_i)},\quad\text{. . . . .(17)}$$

where $\delta_{(x_i,y_i)}$ denotes the Dirac delta function. The risk of an approximation function f with respect to this measure is called the empirical L-risk and is defined as

$$R_{L,D}[f]=\frac{1}{l}\sum_{i=1}^{l}L\left[x_i,y_i,f(x_i)\right].\quad\text{. . . . .(18)}$$

This risk measure can be interpreted as an approximation to the average loss on training points, leading to find a minimizer f whose risk is as small as possible for some function class H (Steinwart and

Christman 2008). However, the capacity of H has to be controlled to prevent overfitting and, hence, poor generalization properties. This can be achieved by introducing a regularization term, $\|\boldsymbol{\omega}\|^2$, which results in a modified risk measure with an SRM component (Tikhonov and Arsenin 1977; Vapnik 1982; Morozov et al. 1984; Schölkopf and Smola 2002):

$$R_{\text{reg}}[f] = R_{L,D}[f] + \frac{\lambda}{2}\|\boldsymbol{\omega}\|^2, \quad \text{(19)}$$

where λ is a positive regularization constant. For regression estimation, the VC bound is defined as

$$R(h) \le R_{L,D}(h)\left[1-\sqrt{p-p\ln(p)+\frac{\ln(n)}{2n}}\right]^{-1}, \quad \text{(20)}$$

where $p = h/n$ and h is the VC dimension that characterizes model complexity. It can be shown that, by minimizing $\|\boldsymbol{\omega}\|$, the SVM training process minimizes the VC dimension and, consequently, the generalization error (Kecman 2005; Basak et al. 2007). A suitable loss function should be insensitive to outliers and not be a complex function so that optimization is straightforward. To ensure the existence and uniqueness of a minimum of optimization, the loss function should also be convex (Fletcher 1987).

The prediction performance of an SVM depends significantly on the choice of the loss function (Müller et al. 1997; Smola et al. 1998). The ε-insensitivity loss function defined in Eq. 3 is used in classical SVR. Loss functions of the form $|y - f(\mathbf{x})|_\varepsilon^p$ with $p > 1$ may not be as robust (Huber 1981) because the derivative of the cost function grows without bound. The loss function, on the other hand, becomes nonconvex with $p < 1$. It is worth noting that, with $p = 2$, the least-squares loss penalizes the discrepancy between y and $f(\mathbf{x})$ quadratically (Schölkopf and Smola 2002). The most important difference of the SVR methodology used here compared to classic regression is that a Vapnik-type ε-insensitivity loss function is used. This results in zero estimation cost loss if the difference between the predicted $f(\mathbf{x}_i, \boldsymbol{\omega})$ and the measured value y_i is less than ε. That is, the prediction points are within the ε-tube. For points outside the tube, the cost equals the magnitude of the difference between the predicted value and the radius ε of the tube.

Choosing Parameters for SVR

The performance of the SVR model strongly depends on the selection of model- and kernel-function parameters. Both a grid search and a pattern search are employed to find optimal parameter values. The grid search uses values from a specified range by using geometric steps. On the other hand, the pattern search starts at the center of the given range and takes trial steps in each direction for each parameter. If the parameters at the new point improve the fit, the search center moves to the new point and the process is repeated. Otherwise, the step size is reduced and the search is resumed. The pattern search stops once the step size is reduced to a predefined tolerance. Grid search is computationally expensive because the model must be evaluated at many points within the grid for each parameter. This approach may be computationally infeasible if cross validation is also used for each model evaluation. A pattern search, on the other hand, generally requires far fewer evaluations of the model than a grid search. However, the drawback of pattern search is that it may converge to a local instead of a global optimum. Therefore, to enhance searching, both search methods should be employed in a systematic way. Optimization starts out with the grid search, hoping to find a region near the global optimum point. Subsequently, a pattern search is performed over a narrow search range surrounding the best point found by the grid search, resulting in finding the global optimum.

Permeability Prediction Methodology

Reservoir Description. The reservoir used in this research is located in Middle East and was deposited in a fluvial-dominated deltaic environment of Middle Cretaceous Albian age. It consists of a thick sequence of sandstone, siltstone, and shale with thin intervals of limestone, coal, and varying amounts of ironstone. The reservoir can be divided into three main lithologic units. The first unit consists of mainly shale and sands. They are thinly interbedded with small amount of siltstones, green mud, and small siderite nodules. This unit represents a shallow marine and prodelta environment. The second unit is the main sand, which is thick homogeneous clean sandstone, deposited as distributary mouth bars and distributary and fluvial channels. This sand is the thickest and best part of the reservoir. It has good vertical and lateral continuity and is occasionally interbedded with thin shale. The third unit consists of thin shale and sand layers and some thin beds of shallow marine limestones. Three wells where log and core data were available to model permeability were used in this study. **Fig. 2** shows a semilog plot of core-permeability vs. core-porosity measurements from the three wells. The scatter of these data shows the high degree of heterogeneity in this reservoir.

Training and Testing Procedure. The procedure used to train and test each regression technique is summarized as follows.

Step 1: Generate Training and Testing Data. The three wells were divided into training and testing wells to evaluate accuracy and robustness of SVR relative to the three regression techniques. To ensure thorough analysis through large training and testing performance space, the wells were subdivided, as listed in **Table 2,** where individual wells and a combination of wells were used in the classification procedure. Two types of data were used, raw data and normalized data, to test the predictability of the regression techniques. Each training pattern consisted of log data, including gamma ray, neutron porosity, sonic porosity (DT), bulk density, and formation resistivity, as entries in the input vector and core-based permeability, log(k) as a scalar output.

The following steps were adopted to normalize log and core data.

1. Determine maximum (MaxValue) and minimum (MinValue) of the variable over all training patterns,

2. Determine midpoint and scale as follows:

$$\text{midpoint} = (\text{MinValue}+\text{MaxValue})/2 \quad \text{(21)}$$

$$\text{scale} = (\text{MaxValue}-\text{MinValue})/2 \quad \text{(22)}$$

3. Determine the scaled values for each training pattern as follows:

$$\text{scaled value} = (\text{value}-\text{midpoint})/\text{scale} \quad \text{(23)}$$

This scales the inputs to the range [−1, 1].

Step 2: Construct Models Using Training Data. Every data set in Table 2 is submitted to each regression algorithm to approximate the underlying function by using input and output data. The DTREG (Sherrod 2009) software package was used to generate permeability-prediction models. For SVR, models with Gaussian RBF, sigmoid, and linear kernel functions were constructed to further compare their accuracy and robustness in permeability prediction. Both SVR-model and kernel-function parameters were selected by using grid and pattern search, as described.

Step 3: Compare the Accuracy and Robustness of Each Technique. To ensure consistency of the comparison of the accuracy of the regression techniques, different combinations of wells were trained and tested as listed in **Table 3.** Three cases are assessed by correlation coefficient r and error statistics including RMSE, AAE, and MAE, as defined in **Table 4.** The RMSE gives an indication of the overall accuracy of the predictive model, whereas the maximum error indicates the presence of a regional area of poor approximation.

To enable comparisons of SVR to the three other regression techniques, the average percentage differences in error values between each technique and the actual data are computed; these are compared then to the SVR error. Therefore, the error in the SVR approximations becomes the benchmark to which all other techniques are compared. Positive percentage values imply that the corresponding approximation had larger errors than SVR, whereas negative values imply that the approximation had smaller errors.

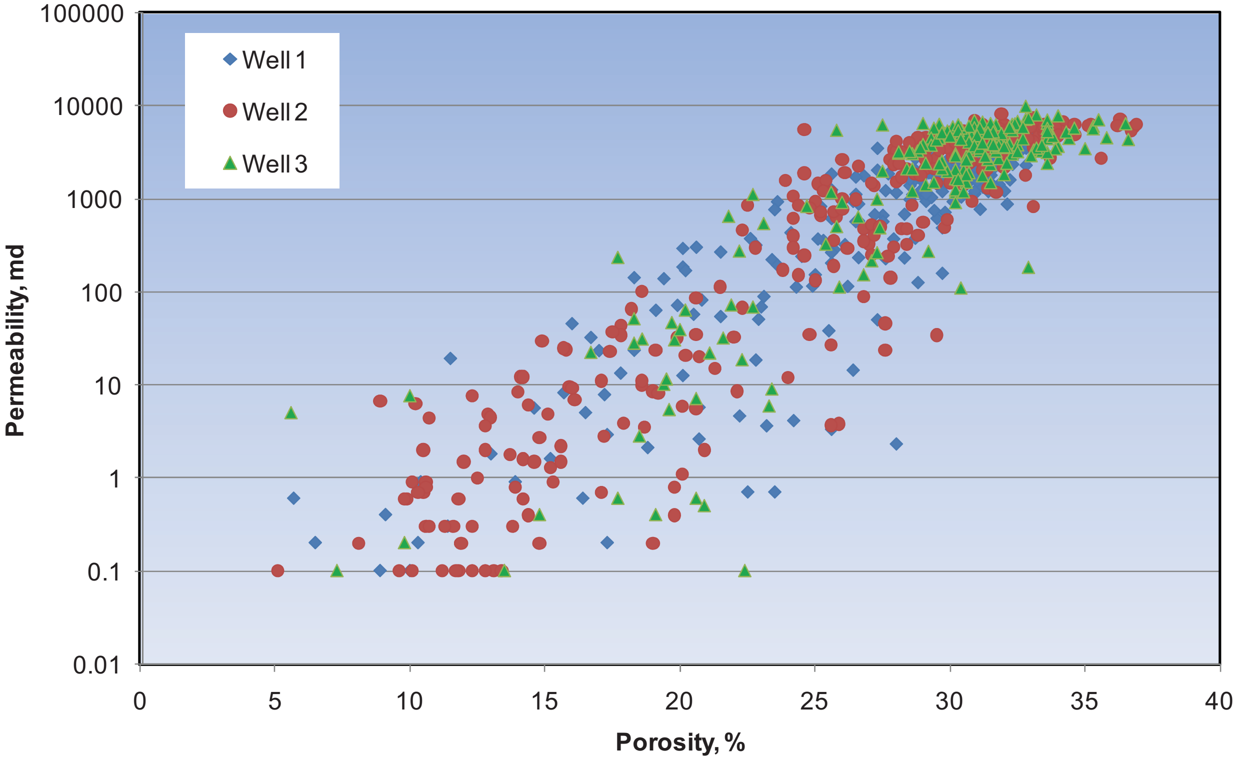

Fig. 2—Crossplot of core permeability vs. core porosity.

To investigate the robustness of each regression technique, the variance between its error values across different testing sample sets as listed in **Table 5** was calculated. The ability of a regression technique to consistently produce high accuracies raises the reliability of the results and the robustness of each technique. SVR is again used as a benchmark to examine the robustness of each approximation technique across the testing set. Robustness was performed by using the standard deviation calculated by the following steps:

1. For each regression technique, find the standard deviation of each error (i.e., RMSE, AAE, and MAE across each of the three cases presented in Table 5). Note that each case includes different training-well data that characterize a different underlying function.

2. Normalize each standard deviation against the standard deviation for SVR for the same case, as can be calculated by the ratio of the standard-deviation difference between each regression and SVR to that of SVR.

The normalized standard deviation reflects the variance of the error for a given approximation technique relative to SVR. A positive value indicates a greater variance than SVR and, hence, less robustness, whereas a negative value indicates a lower variance

TABLE 2—TRAINING AND TESTING WELLS

Well Case Index	Training Wells/Test Well
1	1/2
2	1/3
3	1 and 2/3
4	1 and 3/2
5	2/1
6	2/3
7	2 and 3/1
8	3/1
9	3/2

TABLE 3—TRAINING AND TESTING DATA USED FOR MODEL ACCURACY TEST

Training Wells	Testing Well
2	1
3	1
2 and 3	1
1	2
3	2
1 and 3	2
1	3
2	3
1 and 2	3

TABLE 4—ERROR MEASURES USED FOR ACCURACY ASSESSMENT

Accuracy Measure	Mathematical Expression
Correlation coefficient, r	$\dfrac{\sum_{i=1}^{l}(y_i-\bar{y}_i)(\hat{y}_i-\bar{\hat{y}}_i)}{\sqrt{\sum_{i=1}^{l}(y_i-\bar{y}_i)^2\sum_{i=1}^{NP}(\hat{y}_i-\bar{\hat{y}}_i)^2}}$
RMSE	$\sqrt{\frac{1}{l}\sum_{i=1}^{l}(y_i-\hat{y}_i)^2}$
AAE	$\frac{1}{l}\sum_{i=1}^{l}\lvert y_i-\hat{y}_i\rvert$
MAE	$\max\lvert y_i-\hat{y}_i\rvert$, $i=1,\ldots,l$

TABLE 5—TRAINING AND TESTING DATA USED FOR MODEL ROBUSTNESS TEST

Training Well	Testing Well
1	2
1	3
2	1
2	3
3	1
3	2

than SVR and, hence, more robustness. These results are presented and discussed in the next section.

Analysis of Results

Raw Log Data. Table 6 lists values of the correlation coefficient resulting from the predictions of the SVR method with three different kernel functions, RBF, sigmoid, and linear, compared to the correlation coefficients that result from the MLP, GRNN, and RBFNN regression techniques. The results show that the SVR shows similar or better performance compared to the MLP model over all the tested cases except Well Case 2. Among the regression techniques, GRNN and RBFNN both show worse prediction capability. In some cases, the correlation coefficients of the GRNN and RBFNN are poor. In general, the SVR models yield higher correlation coefficients than the neural-network-based models.

Tables 7 through 9 list the RMSE, AAE, and MAE for the SVR and the MLP, GRNN, and RBFNN cases for Wells 1, 2, and 3. The results show that the SVR methods in general perform better than the other methods, with lower errors. However, there are exceptions. For Well 2, the MLP appears to have better performance than the SVR, whereas, for Well 3, the MLP method shows, in general, better performance than SVR.

Tables 10 through 12 list the normalized standard deviations of the errors for each regression technique relative to the SVR methods. For Wells 1 and 2, the results indicate that SVR is the most robust of the

TABLE 6—COMPARISON OF CORRELATION COEFFICIENTS OF SVR AND MLP, GRNN, AND RBFNN METHODS

				SVR Kernel Function		
Well Case	MLP	GRNN	RBFNN	Linear	Sigmoid	RBF
1	0.7642	0.6146	0.6899	0.7348	0.7437	0.7790
2	0.7155	−0.0901	0.1092	0.5278	0.5409	0.4113
3	0.7661	0.1410	−0.0236	0.7839	0.7852	0.7839
4	0.8149	0.7480	0.7961	0.8055	0.7167	0.8101
5	0.6898	0.7012	0.7096	0.6978	0.6997	0.6992
6	0.6898	0.7012	0.7096	0.6978	0.6997	0.6992
7	0.8084	0.2236	0.3041	0.7993	0.8097	0.7585
8	0.6764	0.6481	0.5272	0.7046	0.7087	0.6971
9	0.6851	0.5900	0.5735	0.6552	0.5793	0.6736

TABLE 7—COMPARISON OF RMSE, AAE, AND MAE BETWEEN THE RBF-KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	0.0908	0.103	0.1331	0.1159	0.1606	0.0718	0.2087	0.1564	0.5711
2	−0.0339	0.0215	−0.1466	0.0818	0.1361	0.0183	0.0449	0.0803	−0.0824
3	−0.2129	−0.3234	−0.017	1.1659	1.0557	0.1527	1.558	1.4316	0.9008

TABLE 8—COMPARISON OF RMSE, AAE, AND MAE BETWEEN THE SIGMOID-KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	0.0605	0.1063	−0.1336	0.0934	0.1732	−0.1971	0.1889	0.1616	0.2464
2	−0.0247	−0.0424	−0.2332	0.0924	0.0615	−0.0846	0.0578	0.0126	−0.1627
3	−0.1364	−0.2334	−0.1082	1.3657	1.2882	0.0536	1.9318	1.803	0.777

TABLE 9—COMPARISON OF RMSE, AAE, AND MAE BETWEEN THE LINEAR KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	0.1661	0.1945	0.1571	0.1971	0.2597	0.0893	0.2998	0.2527	0.6128
2	0.0061	−0.0223	−0.0803	0.1258	0.0818	0.1115	0.0935	0.0339	−0.0025
3	−0.1394	−0.1843	−0.1311	1.3561	1.412	0.0265	1.9446	2.0147	0.729

TABLE 10—COMPARISON OF RMSE, AAE, AND MAE FOR THE NORMALIZED STANDARD DEVIATIONS OF THE ERRORS BETWEEN THE RBF-KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	2.2228	5.451	1.9251	8.4158	14.531	0.2859	3.2712	10.111	2.0631
2	0.9023	2.0135	−0.0503	5.5506	4.0713	1.106	4.9925	5.5948	4.370
3	−0.9418	−0.8268	−0.9609	−0.5543	−0.7852	−0.5925	−0.4774	−0.6073	−0.8964

TABLE 11—COMPARISON OF RMSE, AAE, AND MAE FOR THE NORMALIZED STANDARD DEVIATIONS OF THE ERRORS BETWEEN THE SIGMOID-KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	0.6938	19.374	0.7144	3.9488	48.055	−0.2463	1.2449	34.094	0.7953
2	0.1122	0.4211	−0.6571	2.8297	1.3915	−0.2397	2.5034	2.1099	0.9388
3	−0.7612	−0.6279	−0.9852	0.8301	−0.5384	−0.8461	1.1462	−0.1563	−0.9609

TABLE 12—COMPARISON OF RMSE, AAE, AND MAE FOR THE NORMALIZED STANDARD DEVIATIONS OF THE ERRORS BETWEEN THE LINEAR-KERNEL–BASED SVR AND MLP, GRNN, AND RBFNN METHODS

	MLP			GRNN			RBFNN		
Well	RMSE	AAE	MAE	RMSE	AAE	MAE	RMSE	AAE	MAE
1	0.4298	47.2291	0.8609	3.1773	115.1204	−0.1819	0.8949	82.0729	0.9486
2	0.4111	0.7347	0.0626	3.859	1.9193	1.3565	3.445	2.7963	5.0087
3	−0.8933	−0.8068	−0.8597	−0.1822	−0.7604	0.4623	−0.041	−0.5621	−0.6283

four techniques. On the other hand, for Well 3, the results reveal that the SVR method has the least robust model of the four methods.

Fig. 3 compares predictions of the permeability vs. relative depth from the SVM-RBF, MLP, GRNN, and RBFNN methods. The results demonstrate that the SVM method provides improved estimates of the permeability data along the well trajectory. **Fig. 4** compares predictions of the permeability vs. relative depth of the RBF, linear, and sigmoid kernel functions. The plot reveals that the RBF kernel function appears to predict the permeability the best among the three kernel functions tested.

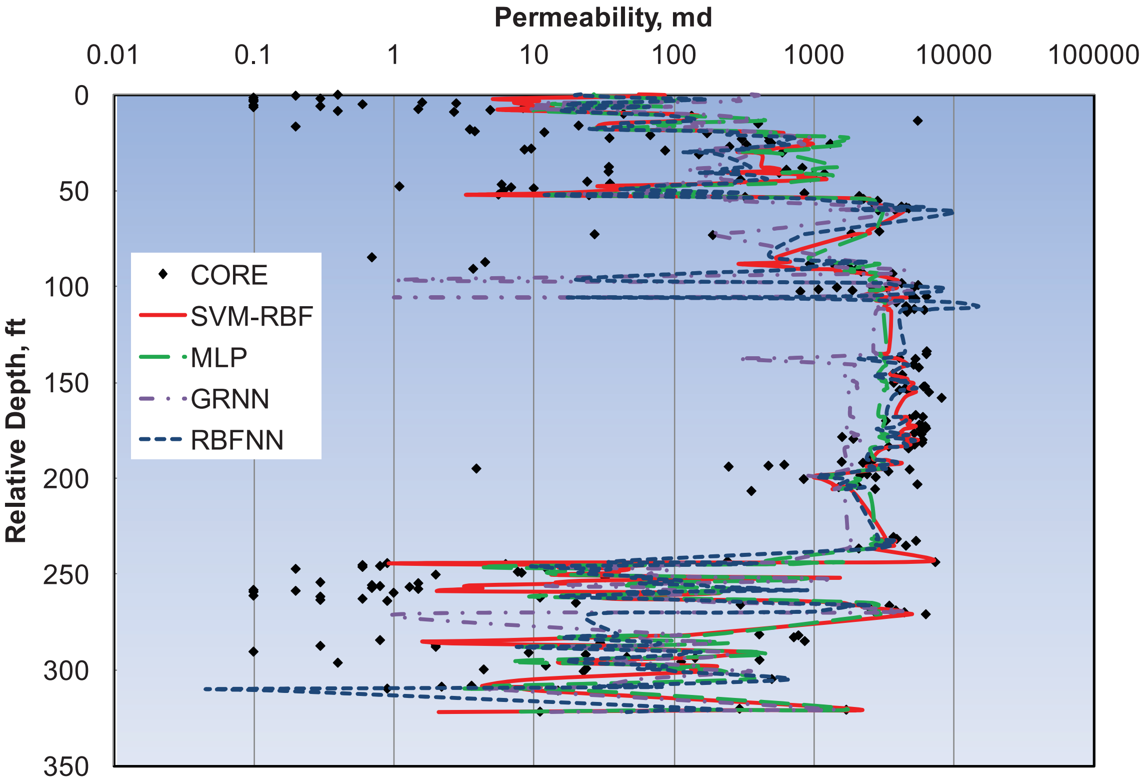

Fig. 3—Comparison of SVM-RBF, MLP, GRNN, and RBFNN predictions for Well Case 1 (Well 1 used to train with predictions of Well 2).

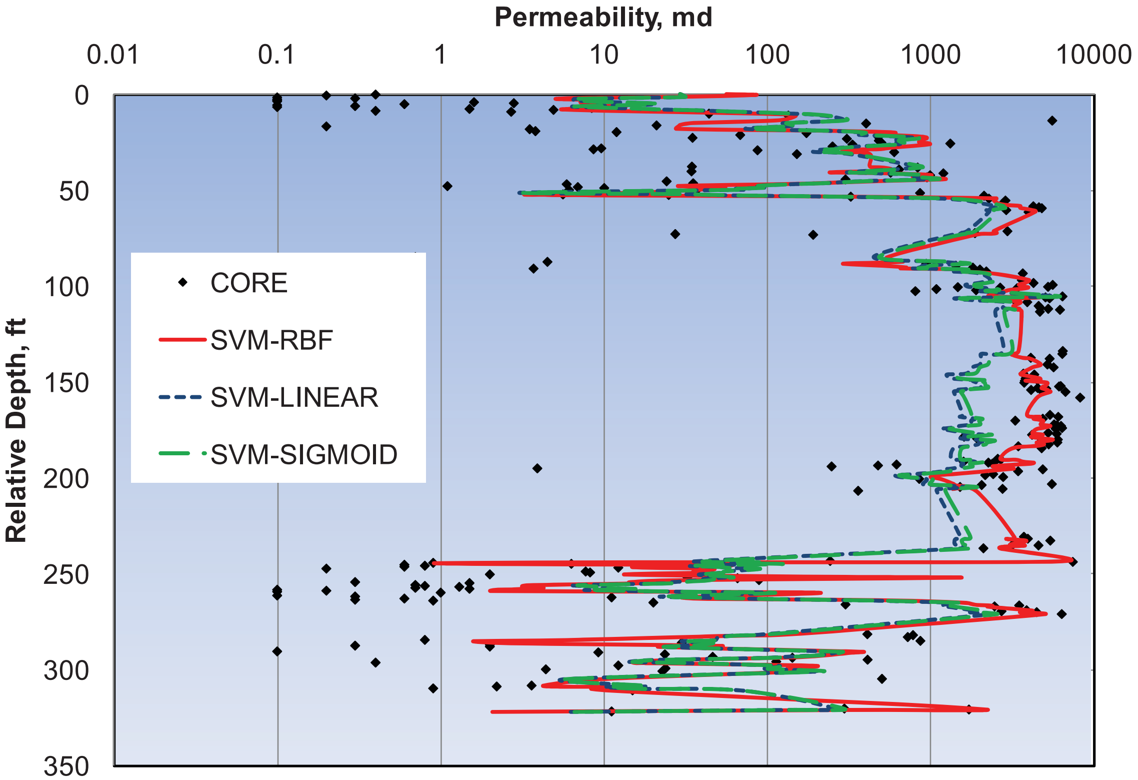

Fig. 4—Comparison of SVM-RBF, SVM-LINEAR, and SVM-SIGMOID predictions for Well Case 1 (Well 1 used to train with predictions of Well 2).

Normalized Log Data. The log data were normalized by scaling them between −1 and +1. **Table 13** presents a comparison of the correlation coefficients of the SVR methods with and without normalization. The results show that the prediction capability of the SVR methods is improved after the log data are normalized. A comparison of the SVR results with normalized log data vs. that of MLP, listed in Table 6, reveals that the SVR method with linear and sigmoid kernel functions outperforms the MLP regression technique for all well cases. However, normalization does not provide consistent improvement over all well cases.

Conclusions

SVR has been used to predict permeability in wells after the method is trained with data from other wells. The results of this study reveal that SVR is accurate and robust. The theory behind the SVR method has been presented, with the result being an actual approximating function. This is different from neural-network methods where the transformation from input data to output results is encapsulated in a black box. The conclusions of this study are as follows:

- The SVR method has utility as a new framework to predict permeability in heterogeneous reservoirs.
- The SVR method, in general, provides better predictions of the permeability than the MPNN, general regression neural networks, and RBFNN methods.
- Normalization of the log data appears to improve the prediction capability of the SVR method.
- In most cases, the sigmoid kernel function appears to predict the permeability best of the SVR methods.

Nomenclature

b = bias constant
C = regularization parameter
DT = sonic porosity log

TABLE 13—COMPARISON OF CORRELATION COEFFICIENTS OF SVR METHODS WITH AND WITHOUT NORMALIZATION

	Raw Log Data SVR Kernel Function			Normalized Log Data SVR Kernel Function		
Well Case	Linear	Sigmoid	RBF	Linear	Sigmoid	RBF
1	0.7348	0.7437	0.7790	0.8076	0.8054	0.7684
2	0.5278	0.5409	0.4113	0.7744	0.7760	0.7807
3	0.7839	0.7852	0.7839	0.7421	0.7748	0.7401
4	0.8055	0.7167	0.8101	0.7067	0.7708	0.7939
5	0.6978	0.6997	0.6992	0.7019	0.7009	0.7118
6	0.7993	0.8097	0.7585	0.743	0.7593	0.7383
7	0.7046	0.7087	0.6971	0.7029	0.7118	0.6489
8	0.6552	0.5793	0.6736	0.6651	0.6859	0.7274
9	0.7348	0.7437	0.7790	0.6662	0.7332	0.7470

f = unknown function

h = Vapnik-Chervonenkis dimension

H = high-dimensional feature space

k = kernel function

K = permeability or kernel matrix

L = Lagrangian equation for a dual programming problem or loss function

r = correlation coefficient

R = empirical risk

R_{reg} = regularized risk

x = input variable

X = input space or set

y = output variable

$\hat{y}$ = estimated output value

Y = output set

α, α^* = Lagrangian multiplier to be determined

δ = Dirac measure

ε = error accuracy

η, η^* = Lagrangian multipliers

κ, ϑ = sigmoid-function parameters

λ = regularization constant

μ = mean value

ξ, ξ^* = slack variables

σ = variance

σ^2 = standard deviation

$\mathbf{\Phi}$ = mapping function from input space into a high-dimensional feature space

$\boldsymbol{\omega}$ = weight vector/regularization term

Subscripts and Superscripts

i, j = indices

l = number of samples

n = input space dimension

References

Al-Anazi, A., Gates, I.D., and Azaiez, J. 2009. Innovative Data-Driven Permeability Prediction in a Heterogeneous Reservoir. Paper SPE 121159 presented at the EUROPEC/EAGE Conference and Exhibition, Amsterdam, 8–11 June. doi: 10.2118/121159-MS.

Basak, D., Pal, S., and Patranabis, D.C. 2007. Support Vector Regression. *Neural Information Processing—Letters and Reviews* **10** (10): 203–224.

Boser, B.E., Guyon, I.M., and Vapnik, V.N. 1992. A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory, ed. D. Haussler, 144–152. New York: ACM Press.

Burges, C.J.C. 1996. Simplified support vector decision rules. In *Proceedings of the International Conference on Machine Learning*, ed. L. Saitta, L., 71–77. San Mateo, CA: Morgan Kaufmann Publishers.

Burges, C.J.C. and Schölkopf, B. 1997. Improving the accuracy and speed of support vector learning machines. In *Advances in Neural Information Processing Systems 9*, ed. M.C. Mozer, M.I. Jordan, and T. Petsche, 375–381. Cambridge, Massachusetts: MIT Press.

Chen, S., Cowan, C.F.N., and Grant, P.M. 1991. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks* **2** (2): 302–309.

Cortes, C. and Vapnik, V. 1995. Support-Vector Networks. *Machine Learning* **20:** 273–297.

Cristianini, C. and Shawe-Taylor, J. 2000. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge, UK: Cambridge University Press.

Cuddy, S.J. 2000. Litho-Facies and Permeability Prediction From Electrical Logs Using Fuzzy Logic. *SPE Res Eval & Eng* **3** (4): 319–324. SPE-65411-PA. doi: 10.2118/65411-PA.

Fletcher, R. 1987. *Practical Methods of Optimization*. New York: John Wiley & Sons.

Guyon, I., Boser, B., and Vapnik, V. 1993. Automatic capacity tuning of very large VC-dimension classifiers. In *Advances in Neural Information Processing Systems 5*, ed. S.J. Hanson, J.D. Cowan, and C.L. Giles, 147–155. San Mateo, California: Morgan Kaufmann Publishers.

Haykin, S. 1999. *Neural Networks, a Comprehensive Foundation*, second edition. Upper Saddle River, New Jersey: Prentice-Hall Inc.

Huang, Y., Gedeon, T.D., and Wong, P.M. 2001. An integrated neural-fuzzy-genetic-algorithm using hyper-surface membership functions to predict permeability in petroleum reservoirs. *Engineering Applications of Artificial Intelligence* **14** (1):15–21. doi: 10.1016/S0952-1976(00)00048-8.

Huang, Z., Shimeld, J., Williamson, M., and Katsube, J. 1996. Permeability prediction with artificial neural network modelling in the Venture gas field, offshore eastern Canada. *Geophysics* **61** (2): 422–436. doi: 10.1190/1.1443970.

Huber, P.J. 1981. *Robust Statistics*. New York: Wiley Series in Probability and Statistics, Wiley-Interscience.

Karush, W. 1939. Minima of functions of several variables with inequalities as side constraints. MSc thesis, Department of Mathematics, University of Chicago, Chicago, Illinois.

Kecman, V. 2005. Support Vector Machines—An Introduction. In *Support Vector Machines: Theory and Applications*, ed. L. Wang, Chap. 1, 1–47. Heidelberg, Germany: Studies in Fuzziness and Soft Computing, Springer-Verlag.

Kuhn, H.W. and Tucker, A.W. 1951. Nonlinear programming. In *Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, 481–492. Berkeley, California: University of California Press.

Morozov, V.A., Nashed, Z., and Aries, A.B. 1984. *Methods for Solving Incorrectly Posed Problems*. New York: Springer.

Müller, K.-R., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., and Vapnik, V. 1997. Predicting time series with support vector machines. In *Artificial Neural Networks—ICANN'97, Lausanne, Switzerland, October 8–10, 1997*, ed. W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, No. 1327, 999–1004. Berlin, Germany: Lecture Notes in Computer Science, Springer.

Peng, K.-L., Wu, C.-H., and Goo, Y.-J. 2004. The Development of a New Statistical Technique for Relating Financial Information to Stock Market Returns. *International Journal of Management* **21** (4): 492–505.

Rogers, S.J., Chen, H.C., Kopaska-Merkel, D.C., and Fang, J.H. 1995. Predicting permeability from porosity using artificial neural networks. *AAPG Bulletin* **79** (12): 1786–1797.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J. 1986. Learning internal representations by error propagation. In *Parallel Distributed Processing, Vol. 1: Foundations*, ed. D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, 318–362. Cambridge, Massachusetts: MIT Press.

Schölkopf, B. and Smola, A. 2002. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. Cambridge, Massachusetts: Adaptive Computation and Machine Learning, MIT Press.

Schölkopf, B., Burges, C., and Vapnik, V. 1995. Extracting support data for a given task. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, ed. U.M. Fayyad and R. Uthurusamy, 252–257. Menlo Park, California: AAAI Press.

Schölkopf, B., Burges, C., and Vapnik, V. 1996. Incorporating Invariances in Support Vector Learning Machines. In *Artificial Neural Networks: 6th International Conference Proceedings/ICANN'96, Bochum, Germany, July 16–19, 1996*, ed. C. von der Malsburg, W. von Seelen, J.C. Vorbrüggen, and B. Sendhoff, No. 1112, 47–52. Berlin, Germany: Lecture Notes in Computer Science, Springer.

Schölkopf, B., Mika, S., Burges, C., Knirsch, P., Müller K.-R., Rätsch G., and Smola, A. 1999. Input Space vs. Feature Space in Kernel-Based Methods. *IEEE Transactions on Neural Networks* **10** (5): 1000–1017.

Schölkopf, B., Smola, A., Williamson, R.C., and Bartlett, P.L. 2000. New Support Vector Algorithms. *Neural Computation* **12** (2): 1207–1245.

Sherrod, P.H. 2009. *DTREG Predictive Modeling Software* (version 9.1 user's manual), http://www.dtreg.com.

Smola, A., Schölkopf, B., and Müller, K.-R. 1998. The connection between regularization operators and support vector kernels. *Neural Networks* **11** (4):637–649. doi: 10.1016/S0893-6080(98)00032-X.

Specht, D. 1991. A general regression neural network. *IEEE Transactions on Neural Networks* **2** (6): 568–576. doi: 10.1109/72.97934.

Steinwart, I. and Christmann, A. 2008. *Support Vector Machines*. New York: Information Science and Statistics, Springer Science+Business Media.

Taghavi, A.A. 2005. Improved Permeability Estimation through Use of Fuzzy Logic in a Carbonate Reservoir from Southwest, Iran. Paper SPE 93269 presented at the SPE Middle East Oil and Gas Show and Conference, Bahrain, 12–15 March. doi: 10.2118/93269-MS.

Tikhonov, A.N. and Arsenin, V.Y. 1977. *Solution of Ill Posed Problems*. New York: V.H. Winston and Sons.

Vapnik ,V.N. 1995. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag.

Vapnik, V. and Chervonenkis, A. 1964. A note on one class of perceptrons. *Automation and Remote Control* **25** (1).

Vapnik, V. and Chervonenkis, A. 1974. *Theory of Pattern Recognition* (in Russian). Nauka, Moscow. (German translation: Wapnik, W. and Tscherwonenkis, A. 1979. Theorie der Zeichenerkennung. Berlin, Germany: Akademie-Verlag).

Vapnik, V. and Lerner, A. 1963. Pattern recognition using generalized portrait method. *Automation and Remote Control* **24:** 774–780.

Vapnik, V., Golowich, S., and Smola, A. 1997. Support vector method for function approximation, regression estimation, and signal processing. In *Advances in Neural Information Processing Systems 9*, ed. M.C. Mozer, M.I. Jordan, and T. Petsche, 281–287. Cambridge, Massachusetts: MIT Press.

Vapnik, V.N. 1982. *Estimation of Dependences Based on Empirical Data*. Berlin, Germany: Springer Series in Statistics, Springer.

Ammal. F. Al-Anazi is a PhD student in the Department of Chemical and Petroleum Engineering at the University of Calgary. He has been with Saudi Aramco since 1997, where he was working as a production engineer. He holds an MS degree from University of Southern California and a BS degree from King Fahd University of Petroleum and Minerals, all in petroleum engineering. Al-Anazi holds US patent 7,661,480, issued on 16 February 2010. His prime research interest is in the area of artificial-intelligence applications in petrophysical-properties estimation and fractured-reservoir characterization. He is a member of the American Association of Petroleum Geologists, the Canadian Well Logging Society, the International Association for Mathematical Geology, and SPE. **Ian D. Gates** is an associate professor in the Department of Chemical and Petroleum Engineering at the University of Calgary. He worked for 7 years in the industry before joining the university. His primary research interests are in thermal and thermal-solvent methods and optimization of these technologies for in-situ heavy-oil and bitumen recovery, energetics and efficiency, emissions, and water use; application of smartwell technologies for adaptive production of heavy-oil fields; support vector machine learning for reservoir characterization; and in-situ gasification (in-reservoir reaction engineering). Gates holds a BS degree from the University of Calgary, an MS degree from the University of British Columbia, and a PhD degree from the University of Minnesota, all in chemical engineering. He is a registered professional engineer in Alberta and a member of SPE.