University of Alberta

Data-Driven Proxy Modeling during SAGD Operations in Heterogeneous Reservoirs

by

Ehsan Amirian

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Petroleum Engineering

Department of Civil and Environmental Engineering

Spring 2014
Edmonton, Alberta

Dedicated to my parents, brother and sisters, for their love, endless support and encouragement.

Abstract

Evaluation of steam-assisted gravity drainage (SAGD) performance that involves detailed compositional simulations is usually deterministic, cumbersome, expensive (manpower and time consuming), and not quite suitable for practical decision making and forecasting, particularly when dealing with high-dimensional data space consisting of large number of operational and geological parameters. Data-driven modeling techniques, which entail comprehensive data analysis and implementation of machine learning methods for system forecast, provide an attractive alternative.

In this thesis, Artificial Neural Network (ANN) is employed as a data-driven modeling alternative to predict SAGD production in heterogeneous reservoirs. Numerical flow simulations are performed to construct a training data set consists of various attributes describing characteristics associated with reservoir heterogeneities and other relevant operating parameters. Finally, several case studies are studied to demonstrate the improvements in robustness and accuracy of the prediction when cluster analysis techniques are performed to identify internal data structures and groupings prior to ANN modeling.

Acknowledgments

I would like to express my sincerest love and gratitude to my family back home, my parents, my brother Amin and my sisters Shole and Shohreh, for their unflagging support and endless love, throughout my studies.

I am very thankful to my supervisor, Dr. Juliana Leung, for having faith in me, for giving me an opportunity to work on this project, for all of her encouragement and support, and for granting me freedom to explore my ideas.

I would like to thank Dr. Aminah Robinson Fayek and Dr. Huazhou Li, my examining committee members.

I gratefully acknowledge the financial support from Natural Sciences and Engineering Council of Canada (NSERC) for this research through grants held by Dr. Juliana Leung. My thanks are also extended to Computer Modelling Group Ltd. (CMG) for providing the CMG software package for the simulation study.

I am thankful to Siavash Nejadi for his valuable guidance and suggestions that have always been very helpful in improving my reservoir engineering skills. I would like to extend my thanks to my dearest friends Sajjad, Ebrahim, Seyyed Hossein, Alireza, Behrooz, Ehsan, Hossein, Amitt, Jindong and Vikrant for their friendly encouragements.

Finally, I would like to thank all those who helped and inspired with me throughout my research.

Table of Contents

List of Tables

List of Figures

Chapter 1: General Introduction

1.1 Introduction

Steam-Assisted Gravity Drainage (SAGD) is an Enhanced Oil Recovery (EOR) method for extracting heavy crude oil and bitumen, in which a pair of horizontal wells called injector and producer are drilled into the formation with an inter-well spacing of 5 meters. High quality steam is injected through the injector well (upper well) into the reservoir. Steam propagates as a steam chamber vertically and laterally in the formation, causing the heated bitumen or heavy oil to drain into the lower wellbore, where it is produced.

The Steam Assisted Gravity Drainage (SAGD) provides more efficient recovery of unconventional oil resources, such as heavy oil and bitumen, as compared to the other thermal recovery methods. As a supply for future fuel and energy demand, the 95% of the bitumen deposits in North America which are located in Alberta oil sands are expected to become a major source. Due to these issues, the demand for SAGD technique is significantly increasing.

Numerical modeling of SAGD recovery performance can be carried out with traditional compositional thermal simulators. The current flow simulators require a huge number of input parameters such as initial saturations and pressure distributions, porosity, permeability, multi-phase flow functions, and well parameters. Inference of these input parameters is time-consuming, while accurate measurements are often not readily available. Furthermore, many assumptions associated with the process physics are often invoked for the numerical solution. Given the extremely non-linear relationships between input and output variables (e.g. oil production profile), the computational time is also extremely high. Therefore, there has been an increased drive and interest to integrate data-driven approaches for modeling the recovery response of SAGD process.

Data-driven modeling provides a viable alternative for quantitative ranking of different operating areas and assessment of uncertainty due to reservoir heterogeneities, which are crucial elements in optimization of production and development strategies in oil sands operations. High-dimensional data including large number of operational and geological parameters can be processed for efficient decision-making.

Data-driven modeling is based on analysis of all data characterizing the system of interest and focuses on using the machine learning methods to build models that describe the behavior of the corresponding physical processes. Examples of some popular methods used in data-driven modeling are statistical methods, artificial neural networks, and fuzzy logic. The methods used nowadays have advanced significantly beyond the ones used in conventional empirical regression. They are used for solving numerical prediction problems, reconstructing highly non-linear relationships, performing data classification, and building rule-based expert systems.

The general subject of data-driven modeling has been developed with contributions from many overlapping disciplines including virtual intelligence, data mining, knowledge discovery in databases, computational intelligence, machine learning, statistical data analysis, soft computing, and pattern recognition.

Artificial neural network (ANN) is a virtual intelligence technique useful for identifying or approximating a complex non-linear relationship between input and target variables. It utilizes a series of neurons (nodes) in the hidden layers that apply nonlinear activation functions to the weighted sum of input variables. The network is trained using a data set consisting of both input (or predicting) and target variables, and the unknown parameters of the network, typically the weights and biases that connect between all the neurons, are estimated in an inverse problem where the mismatch between the network output and the known values of the target variables is minimized. Figure 1.1 presents a typical flow chart for network training. Since variables can be both categorical and continuous, common applications of ANN include proxy for function evaluation and pattern classification. History and improvements for ANN technology including learning rules, network architecture, convergence behavior, and hybrid techniques can be found in Mehrotra et al. (1997), Suh (2012), and Poulton (2001).

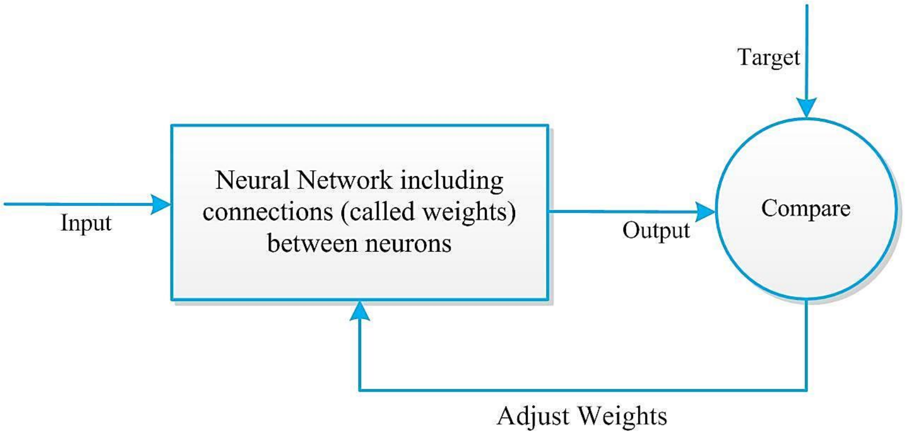

Target
Input
Neural Network including connections (called weights) between neurons
Output
Compare
Adjust Weights

Figure 1.1 General flow chart for neural network training (adapted from Demuth & Beale 1998)

1.2 Problem Statement

Ultimate oil recovery factor and production profile are two key factors in evaluating the amount of oil that is recoverable and also the possible monetary benefits that can be generated over the years. Recovery factor (RF) is the ratio of recoverable oil to estimated initial oil in place. It depends on many factors related to rock, fluid, and operation properties. The profile of cumulative oil production as a function of time is referred to as oil production profile.

Although detailed compositional simulators are available for recovery performance evaluation for SAGD, the simulation process is usually deterministic, cumbersome, expensive (manpower and time consuming), and not quite suitable for real-time decision making and forecasting. This motivates a substitute (alternative) approach of oil recovery evaluations using existing reservoir data set that could be used as a proxy-data-driven model to estimate recovery factor and production profile.

1.3 Research Objectives

Data-driven proxy model is an alternative model which stands in the place of currently used models (e.g., detailed compositional flow simulations) and aims to relate the model inputs to the desired outputs only by means of the information coming from the data. The comprehensive data set is used to drive the model and lead us to the desired output. Data-driven proxy models can utilize any artificial intelligence techniques to develop a solution to the defined problem.

Among the SAGD industries, a large amount of data is coming from interpretation and analysis of well logs. Production data is also readily available from the public domain. However, details regarding the operating conditions (e.g., bottom-hole flowing pressure, injected steam quality and rate) remain proprietary. Therefore, the research objective is to develop a data-driven modeling workflow to estimate the relationship between input attributes derived from well log analysis to output attributes including recovery factor and production profile. As the available data do not include linguistic variables and employing the expert knowledge to the well logs is difficult, artificial neural network (ANN) is selected to be the most suitable technique for this study. Due to the extremely non-linear equations used in flow simulators, ANN is an appropriate method to identify and approximate these equations relating model inputs to the desired outputs.

In the past, ANN and other non-linear regression methods have been employed to predict formation characteristics, such as permeability, porosity and reservoir fluids saturation using available seismic and well-logs analysis, but applications that aim to predict SAGD recovery performance by incorporating heterogeneity measurements as input attributes are lacking. Heterogeneity-associated parameters can be directly inferred from well-logs and have the potential to be used as inputs for the data-driven proxy model to assess and predict the recovery performance of the well.

The principal objective of this thesis is to develop a data-driven proxy model using Artificial Neural Network (ANN) as an alternative to predict SAGD recovery performance in heterogeneous reservoirs. This research aims to assess the key pertinent predicting parameters in relation to SAGD recovery prediction in heterogeneous reservoirs. Synthetic data set is used to investigate the feasibility of this approach in recovery prediction of SAGD reservoirs.

Given that the field data contain noises and considering the need for continues updating when incorporating new field data into reservoir models, in this work, new approaches are proposed to identify and reduce extrapolations in predictions. This study also illustrates how various data-driven modeling approaches, both deterministic and fuzzy-based can be integrated for production estimation. Since there is a lack of application of ANN for SAGD recovery prediction for heterogeneous reservoirs in the literature, the work presented here provides a promising tool of using large amount of operating data for robust forecasting and optimization in a time-efficient manner. Five case studies are implemented to highlight the potential of ANN as a data-driven proxy model alternative for SAGD process.

1.4 Thesis Layout

The research methodology consists of numerical reservoir simulation and proxy model development, model testing and validation against flow simulation results. The work in this thesis is divided into six chapters. Chapter 1 (the current chapter) provides the background and the scope of this research including the general methodology adopted to address the problem statement. Chapter 2 contains the introduction of Artificial Neural Network (ANN). A detailed literature review on the application of this method in petroleum engineering, along with ANN’s most recent modifications is also included in this chapter. Chapter 3 presents in details the data-driven modeling methodology. Backpropagation Neural Network (BPNN) algorithm, Principal Component Analysis (PCA) methods, cluster analysis and parameterization of oil production time series are all explained in this chapter. Chapter 4 comprises data-driven modeling approach for recovery performance prediction in SAGD operations. Three case studies are performed to assess ANN predictability for recover factor (RF) during SAGD process. Chapter 5 investigates the integrated application of cluster analysis and ANN for SAGD recovery performance prediction in heterogeneous reservoirs. Two case studies presented in this chapter highlight the improvements in ANN predictability for recovery factor (RF) and production profile, after incorporating of cluster analysis into the data-driven modeling approach. Chapter 6 summarizes the major findings of the conducted research and presents suggestions for future research on this topic.

Chapter 2: A Review and Background of the Methodology

2.1 Introduction

Data-driven modeling involves comprehensive analysis of all available data associated with the system and utilization of machine learning methods for constructing models to forecast system behavior in the presence of new data. The data-driven model employs virtual intelligence techniques including neural networks, fuzzy logic and genetic algorithm for complementing or replacing physically based models. A major challenge in data-driven modeling is continuous integration of massive available information into the systems real-time.

2.2 A Brief Background of Artificial Neural Network

Artificial neural network (ANN) is a virtual intelligence method used to identify or approximate a complex non-linear relationship between input and target variables. In computer science and related fields, artificial neural networks are biologically inspired computational models based on human’s central nervous systems (in particular the brain) that are capable of machine learning and pattern recognition. It employs a bunch of neurons (nodes) in the hidden layers that apply nonlinear activation functions to the weighted sum of input variables. A data set consisting of both input (or predicting) and target variables is used to train the network, and the unknown parameters of the network (weights and biases) are estimated in an inverse problem procedure where the mismatch between the network output and the known values of the target variables is minimized.

Artificial neural network is developed by training the network to represent the intrinsic relationships existing within the data. The idea of neural network alludes back to 1943 when neurophysiologist Warren McCulloch and mathematician Walter Pitts wrote a paper on how neurons might work (McCulloch & Pitts 1943). They modeled a simple neural network using electrical circuits to explain how neurons function in the brain. With advancements in computer science and technology 1950's, it was finally possible to simulate a hypothetical neural network. In previous decades, it has been pointed out combination of many neurons in neural networks can be more promising than single neurons. McClelland and Rumelhart (1986) developed and promoted the learning rules applicable to large neural networks based on gradient descent methods. Recently, researchers have performed huge contributions to neural networks and developed novel techniques such as self-organizing maps associative memories (Mehrotra et al. 1997). Mehrotra et al. (1997), Poulton (2001) and Suh (2012) provide detailed information on history, background and improvements of artificial neural network.

Any neural network is trained using a learning algorithm and training data set. In general there are two types of neural network learning algorithms classification; unsupervised learning and supervised learning. The supervised learning is used to find hidden structure in unlabeled data. The objective is to categorize or discover features or regularities in the training data. Cluster analysis is the most common use of unsupervised learning. In contrast, the supervised learning method requires that target values be provided. A training dataset is needed as the input vector and will generate the rules according to the desired output by adjusting the weights. The weights are then used for processing the inputs of test data set. After providing the desired output to the net, the weights will be adjusted to match the model to the desired goal. The learning process iteration will be continued until the desired goal is reached.

2.3 Artificial Neural Network Architecture

An artificial neural network is composed of many artificial neurons that are linked together according to specific network architecture. The two most common types of network architectures are Single Layer Perceptron (SLP) and Multilayer Perceptron (MLP) which consists of one input and output layer with any number of hidden layers, as illustrated in Figure 2.1. Design of a neural network involves the selection of the number of layers and the number of neurons (nodes) in each layer. There is a tradeoff between accuracy and overfitting of data: mismatch between network predictions and actual values of target variables could be not minimized with insufficient number of neurons, while too many neurons can cause an overfitting of network parameters. The number of free parameters (i.e. number of weights and bias connections) in the hidden layer remains a function of input vector dimension and the total training data set size. The selection of the number of neurons is typically established based on some rules of thumb. The number of independent (input) variables is generally much larger than the number of dependent (target) variables. Ferreira et al. (2012) suggested that the number of neurons should be between the number of input parameters and the number of output parameters; in particular, the number of neurons should be two thirds of the number of input parameters, plus the number of output parameters, but no more than twice the number of input parameters. Haykin (2005) explained that the number of free parameters (i.e. number of weights and bias connections) in the hidden layer should be a function of input vector dimension and the total training data set size. In each of our case study, sensitivity analysis on the network configuration is performed; the optimal architecture is selected by comparing the error/mismatch in network prediction between different configurations.

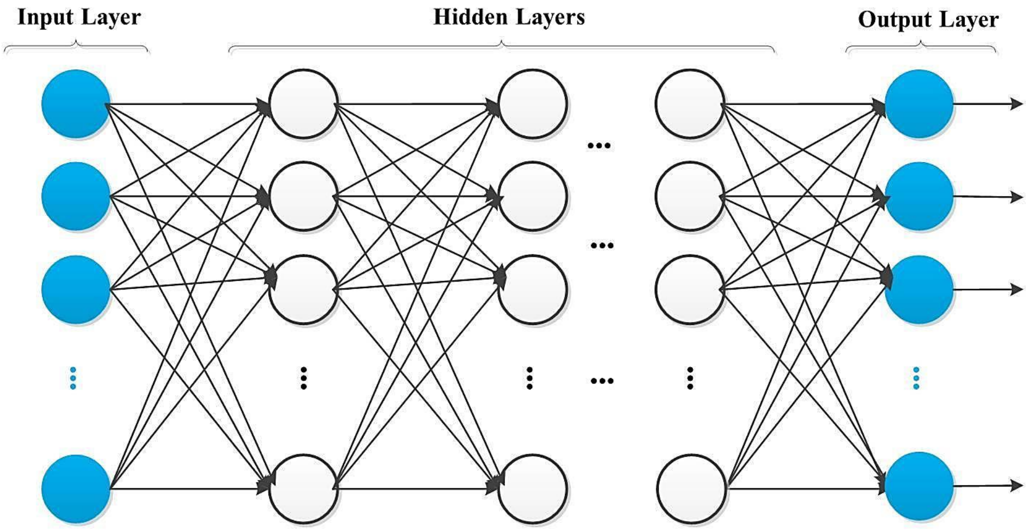


Figure 2.1 Schematic of the interaction of node j with n input signals and a single output

2.4 Application of Artificial Neural Network in Petroleum Engineering

A wide variety of neural network applications can be found in petroleum engineering (Mohaghegh 2002; Bravo et al. 2012; Saputelli et al. 2002; Stundner 2001), particularly in the areas of: classification (Stundner 2001), reservoir characterization or property prediction (Tang et al. 2011; Raeesi et al. 2012; Aminian et al. 2003), proxy for recovery performance prediction (Awoleke & Lane 2011; Lechner & Zangl 2005), history matching (Ramagulam et al. 2007), and design or optimization of production operations and well trajectory (Stoisits et al. 1999; Luis et al. 2007; Artun et al. 2012; Yeten & Durlofsky 2003; Oberwinkler et al. 2004; Malallah & Sami Nashawi 2005; Zangl et al. 2006).

In particular, neural networks have been utilized in recent years as a proxy model to predict heavy oil recoveries (Queipo et al. 2002; Popa et al. 2011; Popa & Patel 2012; Ahmadloo et al. 2010); to perform EOR (enhanced oil recovery) screening (Zerafat et al. 2011; Karambeigi et al. 2011; Parada & Ertekin 2012); to characterize reservoir properties in unconventional plays (Holdaway 2012); and to evaluate performance of CO_2 sequestration process (Mohamadpour et al. 2012).

The number of free parameters (i.e. number of weights and bias connections) in the hidden layer remains a function of input vector dimension and the total training data set size. Several relationships exist in the literature relating the training data set size to some user-defined error parameter (Haykin 2005), with some mechanisms implemented to detect extrapolations (Lohninger 1993). A recent review of a range of design issues related to ANN development in petroleum industry can be found in Al-Bulushi et al. (2012).

Regarding the design of network architecture, most works follow some basic rules of thumb and determine the optimal network architecture via the process of trial-and-error, while others have proposed to generalize training to include learning the appropriate architecture with Bayesian methods, where treatments of probabilistic networks can be found in Poulton (2002) and Khan & Coulibaly (2006).

Recent works in the literature also proposed the use of functional neural networks where activation functions and weight connections associated neurons could vary and be estimated through training (El-Sebakhy et al. 2012). Most works utilize back-propagation or other gradient-based optimization techniques as the learning algorithm, while only a few authors have discussed the use of more general global optimization algorithms such as genetic algorithm (Saemi et al. 2007).

Numerous works also highlight the importance of data pre-processing (the input and target data to be used in the training stage), which includes normalization and outlier detection (Tang et al. 2011). Furthermore, in applications where responses from detailed flow simulations are used to train a network that would serve as a proxy for reservoir performance prediction, experimental design is often performed to reduce redundancy in training data and to minimize computational time (Queipo et al. 2002).

Some authors have also proposed that indices such as permeability ratio of adjacent layers and Dykstra Parsons coefficient should be included in the set of predicting variables to account for heterogeneity and uncertainty in reservoir properties (Ahmadloo et al. 2010).

2.5 Application of Cluster Analysis in Petroleum Engineering

To reduce the dimension of input vector, Principal Component Analysis (PCA), a linear technique for pattern identification in data set (Smith, 2006) can be utilized to (1) reduce the number of variables of a dataset; and (2) identify hidden patterns among data set (Sharma, 2008). PCA is a linear operation that converts the set of correlated variables into a reduced set of linearly uncorrelated (orthogonal) variables. PCA has been widely integrated in a number of pattern recognition applications in the area of petroleum engineering: pattern extraction from seismic amplitude data (Strebelle et al. 2003), lithofacies characterization (Gilbert et al. 2004), and time series analysis of production data (Bhattacharya & Nikolaou 2013).

Data clustering methods should be integrated along with other techniques when facing a large amount of data. Data clustering is a common technique for statistical data analysis that is used in many fields including machine learning, data mining, and pattern recognition (Sharma, 2008). There are several algorithms for data clustering. For example, K-means clustering is a commonly adopted technique because of its robustness and computational efficiency for the small data sets with limited variables (Sharma, 2008). Good predictive results were revealed after using k-means clustering to classify 3D seismic data for permeability prediction (Scheevel & Payrazyan 2001). Chen & Durlofsky (2008) applied k-means clustering prior to two-phase flow functions upscaling, where different statistical procedures are applied to individual groups of grid blocks at the coarse scale with similar fine-scale single-phase flow velocities. K-means is employed to identify clusters of reservoir models exhibiting similar flow response behavior. Reservoir models were mapped on to a lower-dimensional space where clustering is performed (Scheidt & Caers 2009). Awoleke & Lane (2011) applied this clustering technique to identify groups of wells with different water throughput in a shale gas reservoir. Their corresponding physical characteristics are compared in order to mitigate the risk of water production in new wells.

2.6 Application of Fuzzy Logic in Petroleum Engineering

Fuzzy logic is another artificial intelligence modeling framework that has been adopted successfully in a variety of petroleum engineering applications (Mohaghegh 2000), since description of uncertainty due to the random nature of events or imprecision and vagueness of the information associated with the problem can be facilitated using fuzzy sets (Zadeh 1965). These examples include development of rule-based models for screening of enhanced oil recovery (EOR) (Chung et al. 1995), reservoir anisotropy quantification (Zhanggui et al. 1998), lithofacies and permeability forecasting (Hambalek & Gonzalez 2003; Al-Anazi et al. 2009), and well design and placement (Garrouch & Lababidi 2001; Popa 2013).

While neural networks are useful to learn and recognize complex non-linear relationships or patterns in data set, they are not suitable for explaining how these relationships are derived. Fuzzy logic systems, on the other hand, are good at reasoning with vague information, but they generally require subjective expert input in order to formulate the rules for decision-making (Takagi & Hayashi 1991; Yager 1992). Both neural networks and fuzzy logic are powerful design techniques that have their strengths and weaknesses. In recent years, hybrid systems have attempted to merge the two modeling techniques (Lin & Lee 1996).

Hybrid neural network and fuzzy logic systems can be classified into three main categories. The first category is a neural fuzzy system in which the main architecture is a fuzzy rule-based system; however, resultant error between model prediction and the actual response for a particular training data set is back-propagated through the system, as in most neural networks, to adjust fuzzy parameters including membership functions. The second type is fuzzy neural networks in which elements of a neural network are fuzzified; for example, weights are replaced by membership values, while transfer and activation functions of a crisp neuron are substituted with fuzzy operations (Canuto 2001). Finally, the third category is one where the fuzzy logic and neural network methods are executed independently, and their results are assembled and aggregated in a way to reach the assigned goal such as process control or pattern recognition (Lin & Lee 1996).

Applications of these hybrid systems can be found in petroleum engineering (Mohaghegh 2000), particularly in the areas of reservoir lithology identification and property prediction (Zhou et al. 1993; Lim 2005; Aminzadeh & Brouwer 2006), reservoir management (Nikravesh et al. 1998; Alimonti & Falcone 2004; Popa & Cassidy 2012), and optimization of well operations design (Mohaghegh & Reeves 2000; Murillo et al. 2009; Attia et al. 2013).

2.7 Conclusion

Application of different data-mining techniques in petroleum engineering was briefly reviewed in this chapter. The stated problem in this thesis that is to be solved, as mentioned in Chapter 1, is to predict the SAGD recovery performance in heterogeneous reservoirs using data-driven modeling techniques such as ANN.

As presented in this chapter, previous works in this area have considered only the operational parameters as the input parameters of the proxy model, while ignoring the significant role of reservoir heterogeneity in heavy oil recovery evaluation. Including more geological control in the data-driven models should increase the proxy predictability and robustness.

The other issue that has not been widely addressed by these previous works is proposing schemes when facing extrapolations in data-driven proxy modeling to boost the network predictability.

Chapter 3: Data-Driven Modeling Methodology

In this chapter, the methodology section is presented which details the formulations, network architecture, and training algorithm for estimation of the network parameters. Algorithms employed in cluster analysis including k-means clustering and fuzzy c-means (FCM) clustering are explained and parameterization of oil production time series is discussed. As a quantification tool for matching evaluation between data-driven proxy results and results from flow simulation Root Mean Square Error (RMSE) method is provided. The aforementioned algorithms will be explained in detail. The modeling workflow is illustrated in Figure 3.1.

A series of numerical case studies are presented in chapters 4 & 5. This workflow has the potential to integrate other data-mining techniques and can establish and identify the complex nonlinear relation between heterogeneity measurements (inputs) and target outputs.

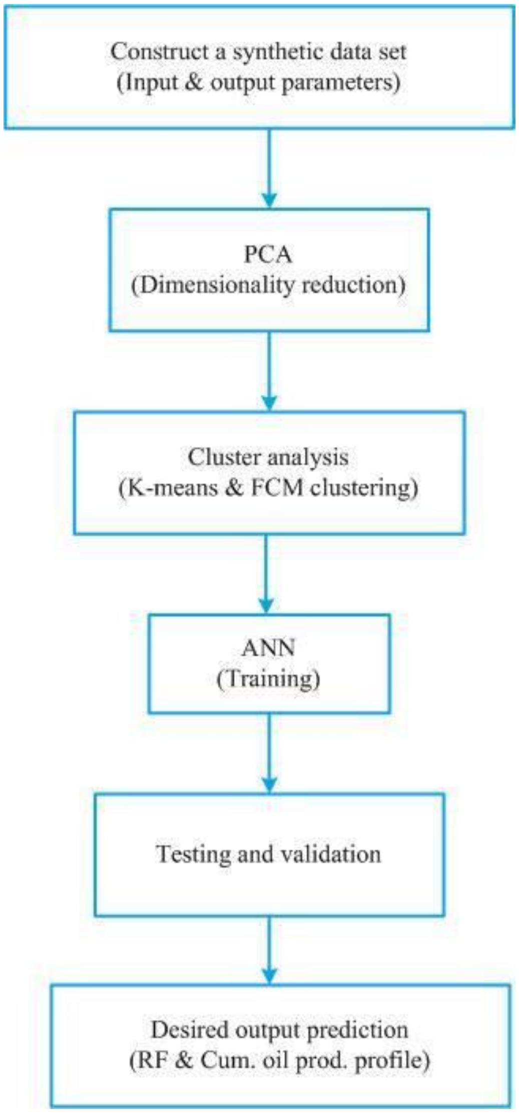

Construct a synthetic data set
(Input & output parameters)
PCA
(Dimensionality reduction)
Cluster analysis
(K-means & FCM clustering)
ANN
(Training)
Testing and validation
Desired output prediction
(RF & Cum. oil prod. profile)

Figure 3.1 General data-driven proxy model methodology flow chart

Special considerations should be exercised in the construction of an efficient ANN model. They include (1) network properties and formulations of its input, hidden, and output layers (Suh 2012); (2) network architecture that includes the number of nodes and hidden layers; and (3) the training algorithm for estimating the unknown network parameters (e.g., weights and biases). In this research, experimental design is implemented in construction of the training data set. The proposed framework also integrates numerous implementations for handling extrapolations and improving network predictability.

3.1 Neural Network Formulation

In a feedforward neural network, signal is passed from an input layer of neurons through a series of hidden layer to an output layer of neurons. The input layer of neurons or input nodes represent the independent variables that are non-linearly related to a set of dependent or target variables, represented by a series of neurons at the output layer. As shown in Figure 2.1, the input and output layers are connected via a series of neurons in the hidden layer(s) or hidden nodes. Weights and biases are assigned to each connection; their values are determined via a supervised learning process using a training data set in which the mismatch between network predictions and known values of the target variables is minimized (Francis 2001).

A schematic of signal transmission from the input layer through a node j to the output layer is shown in Figure 3.2.

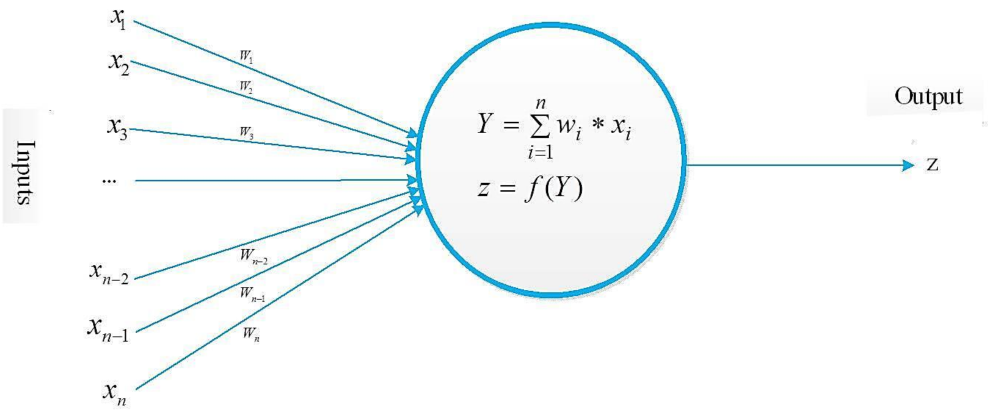


Figure 3.2 Schematic of the interaction of node j with n input signals and a single output

The input signals are multiplied by their corresponding weights to give the value of *Y* as in Eq. (3-1.

$$Y_j = w_0 + \sum_{i=1}^{n} w_{ij} x_i \tag{3-1}$$

Where Y_j is weighted sum of input signals at node *j*; w_0 is threshold (bias) value; w_{ij} is the weight associated with the connection between node *j* and the input node *i*; x_i = value of input node *i*; *n* = number of input nodes. An activation function (e.g., sigmoid function as shown in Eq.(3-2 is applied to the weighted sum.

$$f(Y) = \frac{1}{1 + e^{-Y}} \tag{3-2}$$

Another function often used in neural networks is the hyperbolic tangent function that takes on values between –1 and 1. The value calculated from Eq. (3-2 is the output signal from node *j*, which can be considered as the input signal to the next layer. The signal is transmitted in the same fashion using Eqs. (3-1 & (3-2, until t he final output layer is reached and value for the target variable *z* is calculated.

Due to large disparity in scales of different data sources, normalization is often performed (Francis 2001):

$$x_{new} = \frac{x_{old} - x_{\min}}{x_{\max} - x_{\min}} \tag{3-3}$$

3.2 Estimation of Unknown Network Parameters:

Backpropagation Neural Network (BPNN) Model

The most common algorithm for estimating the unknown network parameters (weights and biases) is the Feedforward Backpropagation Neural Network or Backpropagation Neural Network (BPNN) model. BPNN is a gradient-based minimization technique that utilizes a supervised learning process with feedforward network architecture. Errors are propagated backwards from the output nodes to the input nodes. The algorithm estimates the gradient of error associated with the network's unknown parameters. A gradient-descent algorithm is then applied to estimate parameters that would minimize the error (Werbos 1994). An epoch refers to a cycle (or iteration) in which the entire training data set has been presented to the network. The entire training process typically takes many epochs and the total mismatch/error is expected to decline with increasing number of epochs. A BPNN is easy to implement and suitable to handle complex pattern recognition; hence it is one of the most common techniques for training a MLP neural network (Suh 2012). Figure 3.3 illustrates how signal and error would flow forward and backward, respectively, in a BPNN.

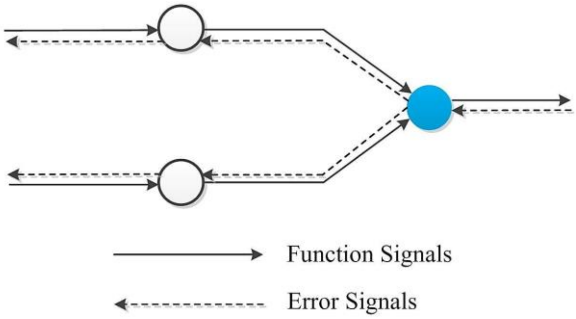


Figure 3.3 Flow of signal and error in a back propagation neural network (BPNN)

BPNN algorithm can be expressed through the following steps (Suh 2012):

1- Set the parameters of the network and the uniform random numbers for Wxh, Why, θh, and θy.

2- Obtain an input training vector *X* and the desired output vector *T*.

3- The output vector *Y* as is calculated as follows:

$$net_h = \sum_i w_{ih} \cdot X_i - \theta_h \tag{3-4}$$

$$net_j = \sum_h w_{hj} \cdot H_h - \theta_j \tag{3-5}$$

$$Y_j = f\left(net_j\right) = \frac{1}{1+e^{-net_j}} \tag{3-6}$$

Where *i* is the number of input nodes, *j* is the number of hidden nodes, W_{xh}, is the weight matrix of node connections between input layer and hidden layer, W_{hy}, is the weight matrix of node connections between hidden layer and output layer, θ_h, is the threshold matrix associated with hidden layer *and* θ_y, is the threshold matrix associated with output layer.

4- The error in each layer is calculated using Eqs. 3-7 & 3-8.

$$\delta_j = Y_j(1 - Y_j) \cdot (T_j - Y_j) \tag{3-7}$$

$$\delta_h = H_h(1 - H_h) \cdot \sum_j w_{hj} \cdot \delta_j \tag{3-8}$$

5- At the output layer the weights and thresholds are adjusted using Eqs. 3-9 & 3-10.

$$\Delta w_{Y_{hj}} = \eta \cdot \delta_j \cdot H_h \tag{3-9}$$

$$\Delta \theta_{Y_j} = \eta \cdot \delta_j \tag{3-10}$$

6- At the hidden layer the weights and thresholds are adjusted using following Eqs.:

$$\Delta w_{X_{ih}} = \eta \cdot \delta_h \cdot X_i \tag{3-11}$$

$$\Delta \theta_{hh} = \eta \cdot \delta_h \tag{3-12}$$

7- Weights and thresholds at the output layer are updated by:

$$w_{Y_{hj}} = w_{Y_{hj}} + \Delta w_{Y_{hj}} \tag{3-13}$$

$$\theta_{Y_j} = \theta_{Y_j} + \Delta \theta_{Y_j} \tag{3-14}$$

8- Weights and thresholds at the hidden layer are updated using:

$$w_{h_{ih}} = w_{X_{ih}} + \Delta w_{X_{ih}} \tag{3-15}$$

$$\theta_{h_h} = \theta_{h_h} + \Delta \theta_{h_h} \tag{3-16}$$

9- Steps 3-7 is repeated until the network converges and the error evolution is stabilized.

3.3 Principal Component Analysis (PCA) Algorithm

PCA is a mathematical procedure that converts a set of correlated variables into a set of orthogonal variables called principal components, which are sequenced in the order of decreasing variance. The mean of each variable is subtracted from the data values of respective variables as shown in Eq.(3-17. Individual element of the data covariance matrix is calculated according to Eq. (3-18.

$$X'_{ij} = X_{ij} - \overline{X}_j \tag{3-17}$$

$$\mathrm{cov}(X'_j, X'_k) = \frac{\sum_{i=1}^{n}(X'_{ij})(X'_{ik})}{(n-1)} \tag{3-18}$$

Where X_{ij} and X_{ik} are the j^{th} and k^{th} variable in data record *i, n* is the total number of data records and $\overline{X}_j$ corresponds to the arithmetic average of variable X_j.

Next, eigen decomposition of the covariance matrix is performed. The eigenvalues are sorted in decreasing order, and those with highest magnitudes and their corresponding eigenvectors (principal components) are retained. Each eigenvalue represents the individual contribution of the corresponding eigenvector in expressing the total variability observed in the original data. This transformation allows us to retain only a reduced basis of principal components that capture majority of variability exhibited in the data. Finally, principal scores (PS), which are considered as inputs to the ANN model, are calculated using Eq.(3-19.

$$\text{Principal Scores} = (\text{Principal Component Matrix}) \times (\text{Data Matrix})^{\mathrm{T}} \tag{3-19}$$

3.4 Cluster Analysis

Cluster analysis or clustering is the classification of data into different subsets or groups, so that the data in each cluster behave similarly or share some common feature according to a particular distance measure (Everitt et al. 2011). Different algorithms including: k-means clustering, fuzzy c-means, and EM clustering exist, and they can be categorized as either hard-clustering (deterministic), in which each object belongs to a single unique cluster, or soft-clustering (fuzzy-based), where each object belongs to each cluster to a certain degree.

3.4.1 K-Means Clustering

This method clusters n objects into k (k<n) partitions by minimizing the total squared error function given by (MacQueen 1967):

$$f = \underset{\{\mu_1,\ldots,\mu_k\}}{Min} \sum_{h=1}^{k} \sum_{x \in x_h} \left\| x - \mu_h \right\|^2 \qquad (3\text{-}20)$$

Where *k* is the number of clusters in the data, μ_h is the center of cluster *h*, and *x* is the data point in the cluster *h*. Each data point is allocated to a cluster distinctly: data points have a belonging degree of 1 to the cluster they are assigned to and belonging degrees of 0 to the neighboring clusters. Silhouette plots are used to illustrate how close each point in one cluster is to points in the neighboring clusters (Kaufman and Rousseeuw 1990). Each cluster is visualized by a silhouette that indicates the inter-cluster separation. The average silhouette value gives an assessment of clustering validity and may be used to choose the optimum number of clusters (Rousseeuw 1987). Different solutions can be attained depending on the initial guess of cluster centers; therefore, the procedure should be repeated multiple times, and the final solution is selected as the one that gives the maximum separation between clusters.

3.4.2 Fuzzy C-Means (FCM) Clustering

As opposed to the deterministic clustering with k-means, membership degrees of each data point corresponding to each cluster center are assigned based on the distance between the cluster center and the data point (Dunn 1973; Bezdek 1981). Membership value is inversely proportional to the distance away from a particular cluster center (Cai et al. 2007). After each iteration, membership and cluster centers are updated according to the Eqs. 3-21 & 3-23.

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{d_{ij}}{d_{ik}})^{(\frac{2}{m-1})}} \tag{3-21}$$

$$\sum_{j=1}^{c} \mu_{ij} = 1 \tag{3-22}$$

$$v_j = \frac{(\sum_{i=1}^{n} (\mu_{ij})^m x_i)}{(\sum_{i=1}^{n} (\mu_{ij})^m)}, \forall j = 1,2,\ldots c \tag{3-23}$$

Where n is the number of data points, m is the fuzziness index that $m \in [1,\infty]$ measures the tolerance of the required clustering, c is the number of cluster centers, μ_{ij} is the membership of i^{th} data belonging to j^{th} cluster, and d_{ij} is the Euclidean distance between i^{th} data and j^{th} cluster center, which is denoted by v_j. The cluster centers are estimated by minimizing the following objective function:

$$J = \sum_{i=1}^{n}\sum_{j=1}^{c}(\mu_{ij})^m \left\| x_i - vj \right\|^2 \tag{3-24}$$

J is the objective function, $\|x_i - v_j\|$ is the Euclidean distance between i^{th} data and j^{th} cluster center (v_j).

3.5 Parameterization of Oil Production Time Series

In order to parameterize the continuous time series of oil production, the empirical Arps decline equations are adopted. Arps (1945) proposed a series of decline models to curve-fit the production rate-time data. Due to the empirical nature, these equations are valid for most reservoir conditions, and they are summarized as follow:

$$Q_p = [\frac{q_i}{(1-b)D_i}][1-(1+bD_i t)^{(b-\frac{1}{b})}] \text{ (Hyperbolic case; } 0<b<1) \tag{3-25}$$

$$Q_p = (\frac{q_i}{D_i})(1-e^{-D_i t}) \text{ (Exponential case; } b=0) \tag{3-26}$$

$$Q_p = (\frac{q_i}{D_i})[\ln(1+D_i t)] \text{ (Harmonic case; } b=1) \tag{3-27}$$

Where *b* is decline exponent, D_i is decline rate, q_i is initial oil rate, Q_p is cumulative oil production, and *t* is elapsed time from start of production. Parameters *b*, D_i, and q_i are used as output variables of the ANN models.

3.6 Root Mean Square Error (RMSE)

RMSE is applied to assess the quality of the model prediction in comparison with the actual data (Nash and Sutcliffe 1970) as shown in Eq. 3-28.

$$RMSE = 1 - \frac{\sum_{t=1}^{N_t} (Q_{p,obs,t} - Q_{p,t})^2}{\sum_{t=1}^{N_t} (Q_{p,obs,t} - \overline{Q_{p,obs}})^2} \tag{3-28}$$

Where $Q_{p,obs,t}$ is observed (actual) cumulative oil production obtained from flow simulation at time step t, while $Q_{p,t}$ is predicted value from ANN at time t. N_t is the total number measurement time steps. The RMSE value derived from this formula can range from -∞ to 1, where a value of 1 (RMSE = 1) resembles to a perfect match.

3.7 Conclusion

In this chapter, a brief description of the algorithms used in our data-driven proxy modeling is explained. These numerical studies aim to illustrate how various techniques can be integrated to achieve the stated research objectives.

Chapter 4: Data-Driven Modeling Approach for Recovery Performance Prediction in SAGD Operations

Experimental design (ED) techniques are used to generate input variables that are evenly distributed in the solution space and reduce the redundancy among samples (Algosayir et al. 2012). A basic factorial design is employed in this chapter by incorporating combinations of all design elements at different prescribed levels. Other ED techniques such as orthogonal arrays can be easily integrated into the proposed workflow if needed.

Three case studies are presented in this chapter to illustrate the efficient application of ANN in recovery performance prediction of SAGD operations. In each case, ED techniques, as described in the previous section, are used to determine the levels of each input variable to be used. Each set of input parameters is subjected to numerical SAGD simulation and the resultant recovery factor is recorded as the output response. These numerical simulation cases create a comprehensive data set for training and testing the ANN models. Selection of input parameters will be discussed in detail. Furthermore, new modifications to the workflow are proposed to handle extrapolations and improve network predictability. These procedures are explained in case study 3.

4.1 Case Study 1

Heterogeneity in hydrocarbon reservoirs creates a great amount of risk during SAGD process. In this case study, ANN is applied to predict SAGD recovery for a series of layered reservoirs with varying porosity and permeability values between layers. Each 2D model (in x-z orientation) consists of 12 layers; each layer has a thickness of 5 m. Porosity (ϕ) is assigned to each layer following a normal distribution N(μ_ϕ, σ_ϕ), where μ_ϕ and σ_ϕ are the corresponding mean and standard deviation. Permeability values (k in mD) are related to porosity as $3330\phi^2$.

Three different normal distributions illustrated in Figure 4.1, are used to generate a total of 60 reservoir models. Each model is subject to SAGD flow modeling with three levels of injection pressure (P_{inj} = 1900, 2200, or 2500 psi) using CMG STARS (Computer Modeling Group, 2009). Therefore, a total of 60 x 3 = 180 simulation cases are used to construct the training data set.

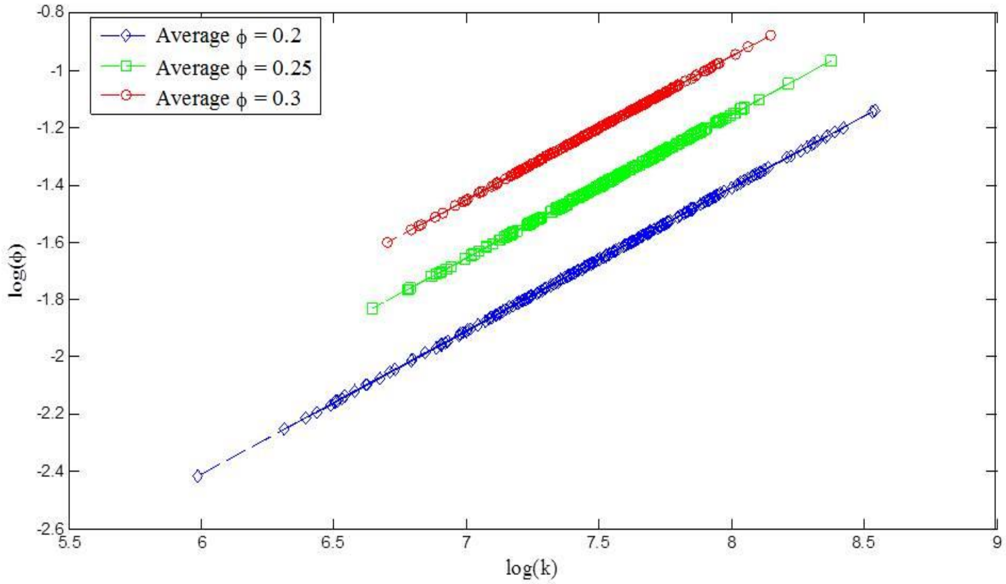


Figure 4.1 Crossplot of log (ϕ) vs. log (k) used in the case study 1

An ANN is set up with only one output/target variable (node): recovery factor or RF and 6 input variables (nodes): mean and variance of porosity values in each reservoir: ϕ_{avg}, $Var(\phi)$; mean and variance of permeability values in each reservoir: k_{avg}, $Var(k)$; Dykstra-Parsons coefficient: V_{DP}; and P_{inj}. Other researchers have also used V_{DP} (Dykstra & Parsons 1950) to describe heterogeneous permeability distribution in layered reservoirs (Charles 2008). A sensitivity analysis of the network configuration is performed where the mismatch between network predictions and actual values of target variables after a fixed number of epochs is compared among different configurations. We conclude that for this application, a network of 2 hidden layers with 8 and 16 nodes in the 1st and 2nd hidden layer, as shown in Figure 4.2, provides the optimal design with the least mismatch between network predictions and actual values of target variables. It should also be noted that 55% of the training data set is used for training while the remaing 45% is used for verification or testing.

There is always a tradeoff between having sufficient data for training and the ability to validate the quality of model prediction. Most of researchers have proposed of using a 75% percent of original data set for training and the rest for testing (Haykin 2005). In this case study, sensitivity analysis was performed for different split percentage of training and testing data set. The results revealed that we can still have a robust model in terms of predictability when using 55% of data cas training. The reason can be that range of predicted range of output attribute (i.e., recovery factor) is approximately between 70 to 80 %. This relatively narrow range enabled us to reduce the percentage of training data from 75 % to the 55 % of the original data set. The backpropagation network is implemented to estimate the unknown network parameters such as weights and thresholds explained in prevoius chapter in Eq. (3-1.

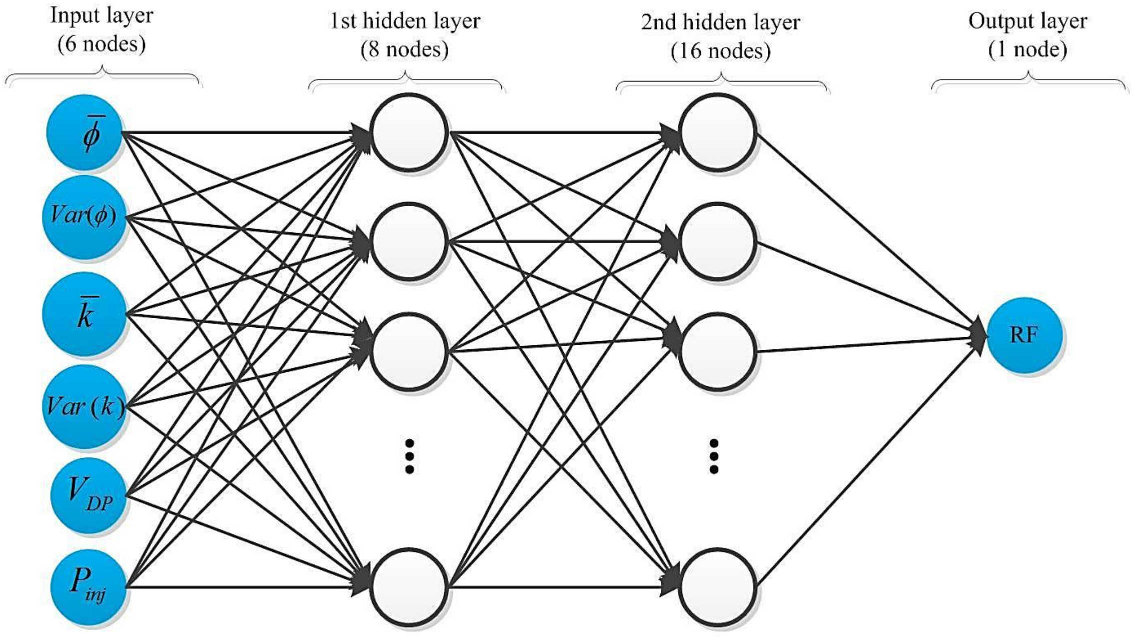


Figure 4.2 Neural network architecture used in the case study 1

Cross-plots between actual target values and network predictions for the training and testing data sets are shown in Figure 4.3. The figure demonstrates that good agreement between the actual values and predictions is obtained. Figure 4.4 shows the decrease in error/mismatch between network predictions and actual target values as a function of number of epochs (training cycles) for the training data set.

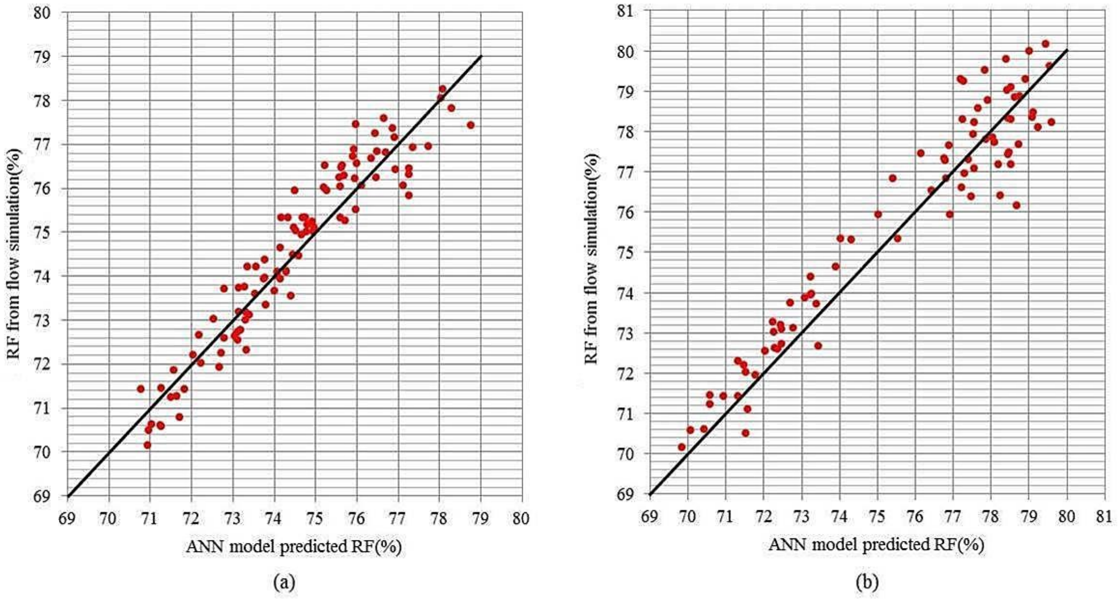


Figure 4.3 Cross plot of actual flow simulation results (target values) against network predictions for case study 1: a) Training data set; b) Testing data set

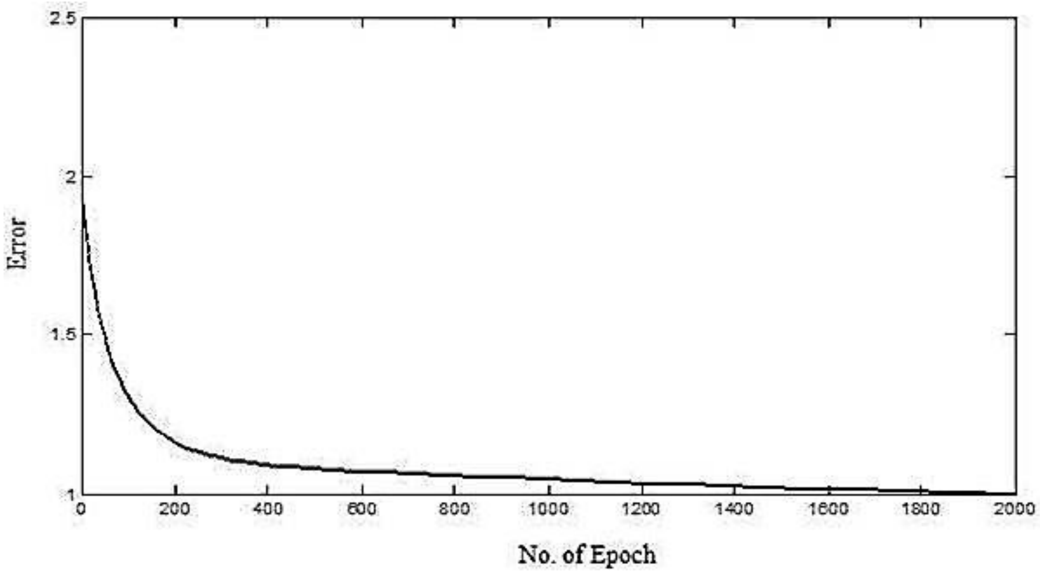
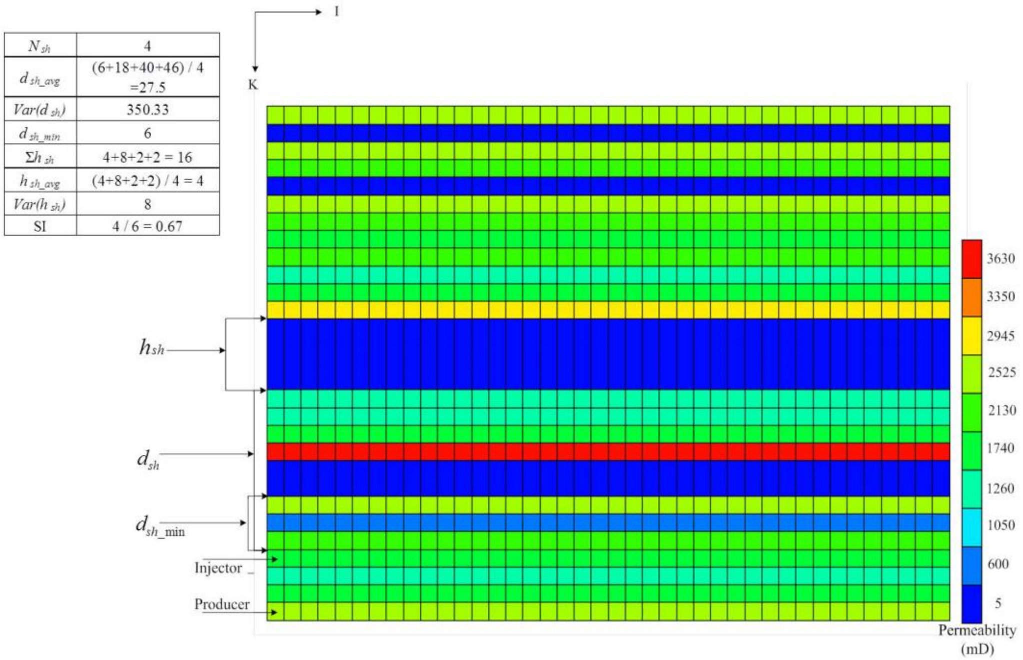


Figure 4.4 Mismatch between network predictions and actual target values in the training data set as a function of number of epochs for case study 1

4.2 Case Study 2

This case study aims to identify additional key predicting variables that would have significant impacts on recovery performance. For example, most practitioners suggest that continuity of sand bodies, influenced by the presence of shale strings should be considered as an important variable. Given that the location and thickness of shale layers are important factors to consider, a series of layered reservoirs with varying amount of low-permeability (shaly) layers are used. Each model is in 2D (x-z) and consists of 30 layers, with each layer being 2m in thickness. Sand porosity and permeability values for each layer are first assigned based on a normal distribution of N(μ_ϕ= 0.25, σ_ϕ = 0.04). Next, low-permeability layers (ϕ = 15% and k = 5mD) with different thicknesses are randomly distributed in each reservoir model. An example of a reservoir model with four low-permeability or shaly layers is shown in Figure 4.5. Also shown in the figure are the two additional parameters defined for each shaly layer: h_{sh} (thickness) and d_{sh} (distance to the injection well).



N_{sh}	4
d_{sh_avg}	(6+18+40+46) / 4 =27.5
$Var(d_{sh})$	350.33
d_{sh_min}	6
Σh_{sh}	4+8+2+2 = 16
h_{sh_avg}	(4+8+2+2) / 4 = 4
$Var(h_{sh})$	8
SI	4 / 6 = 0.67

Figure 4.5- An example of reservoir model with 4 randomly-distributed shaly (low-permeability) layers for case study 2

The injection pressure is kept constant at 2526 kpa. A total of 100 reservoir models with different porosity/permeability and distributions of shaly layers are subjected to flow simulation and the corresponding recovery factor are recorded as part of the training data set. In this case study, 75% of the training data is used in the training stage, while the remaining 25% is used for the verification/testing purpose.

Next, an ANN model is constructed involving one output/target variable (RF) and a total of eight input variables that include N_{sh} (number of shaly layers in each model), d_{sh_avg} (average distance of shaly layers to the injection well), $Var(d_{sh})$ (variance of the distance of shaly layers to the injection well), d_{sh_min} (d_{sh} of the shaly layer that is located at the shortest distance to the injection well), Σh_{sh} (total thickness of shaly layers), h_{sh_avg} (average thickness of shaly layers), $Var(h_{sh})$ (variance of the thickness of shaly layers), shale indicator (SI). The last input variable (SI) is proposed as a new normalized shale continuity indicator defined as the thickness-to-distance ratio of the shaly layer located at the shortest distance to the injector, i.e., h_{sh_min}/d_{sh_min}: a large value would indicate a thick shale barrier that is located close to the injector, impeding the formation of steam chamber. Figure 4.6 illustrates the relationships between d_{sh_min} and SI with RF based on all 100 cases in the training data set. The plot on the left shows that as d_{sh_min} increases, RF also increases since shaly layers could act as barriers for steam chamber development. The plot on the right shows that there is a strong inverse relationship between the proposed shale indicator SI and RF; as a result, it can be considered as an appropriate input variable for the ANN model.

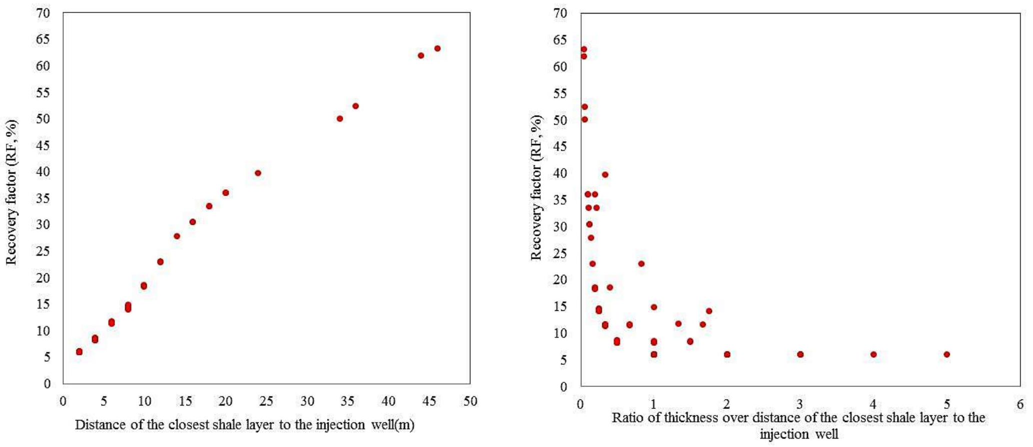


Figure 4.6 Effect of distance (dsh_min) and thickness (hsh_min) of the closest shaly layer to the injection well on the recovery factor (RF)

Cross-plots between actual target values and network predictions for the training and testing data sets are shown in Figure 4.7. Good agreement between the actual values and predictions can be observed. Figure 4.8 presents the decline of error/mismatch as a function of number of epochs; significant error reduction is achieved as the training progresses. The results suggest that the chosen input variables including the new shale indicator can be used successfully to predict recovery performance.

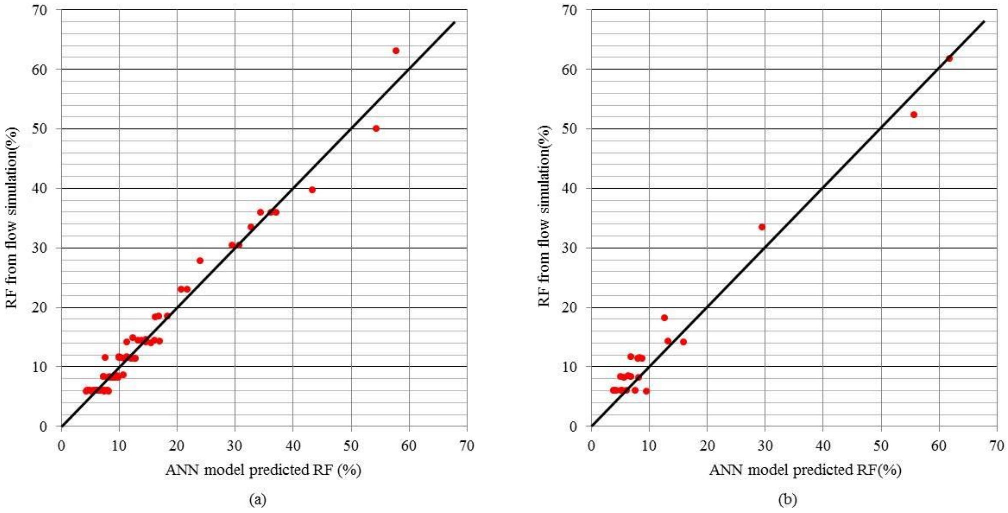


Figure 4.7 Cross plot of actual flow simulation results (target values) against network predictions for case study 2: a) Training data set, b) Testing data set

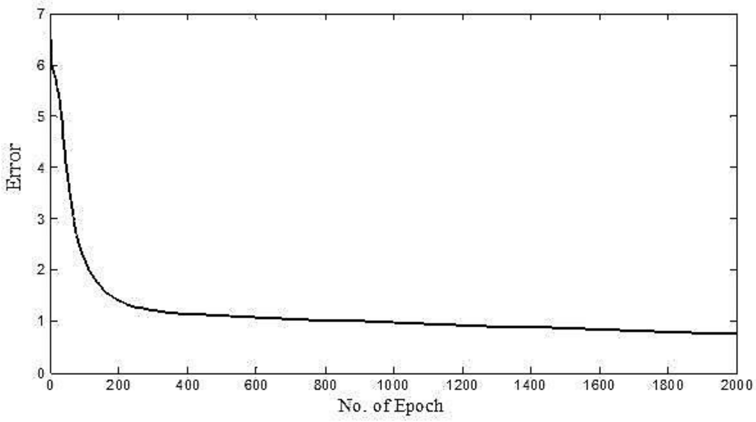


Figure 4.8 Mismatch between network predictions and actual target values in the training data set as a function of number of epochs for case study 2

4.3 Case Study 3

In this section, we would like to investigate the robustness of the ANN model in scenarios where extrapolations may be required. The trained ANN model from the previous section has been used to predict recovery performance for a number of reservoirs larger in size (with 50 layers and a total reservoir thickness of 100m) than those in the training data set (with 30 layers with a total reservoir thickness of 60m). The ANN model trained in the last section is applied to a new set of 100 50-layered reservoir models. These models are also subjected to flow simulations and the “true” RF values are obtained. Values of RF predicted by the network are then compared to the true values as shown in the cross-plot Figure 4.9. It is observed that the flow simulation values and the network predictions do not exhibit an acceptable match, particularly for cases with RF greater than 25%. It appears that the network consistently overestimates in those cases.

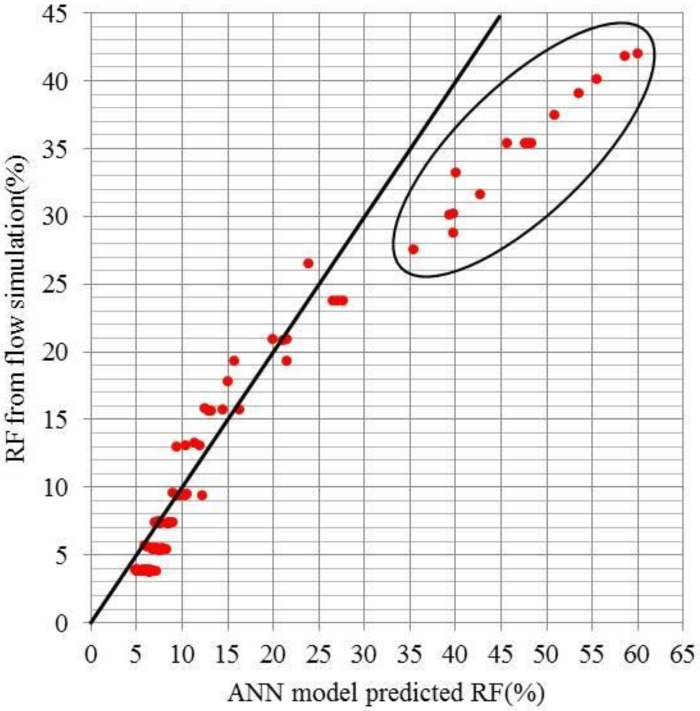


Figure 4.9 Crossplot of actual flow simulation results (target values) against network predictions for the testing data set in case study 3 without network updating

Data driven modeling techniques do not typically provide reliable predictions when there is extrapolation beyond the range of values represented by the training data. Algosayir et al. (2012) has shown significant improvement in the predictability of their response surface model after being updated periodically with additional conditioning (training) data. A new procedure is proposed in which extrapolations are identified and used to generate additional simulation results to be incorporated into the training data set. Using the expanding training data set, the ANN model is updated periodically.

A preliminary analysis reveals that due to the larger domain size (thickness) in this new data set, many cases have input variables that are outside of the ranges of values found in the training data set. In particular, much of the extrapolation is associated with these three input variables: (1) d_{sh_min}, (2) d_{sh_avg}, and (3) SI. Since RF represents the fraction of initial fluid in place to be recovered, it would be influenced by the position of shaly layers in relation to the well locations. In addition to these extrapolations in the input variables, it is noted that network overestimation occurs primarily when predicted RF is greater than 25%. Inspecting the range of RF values in the training data set, it can be readily observed that there is insufficient conditioning data with large RF values. As a result, a procedure is implemented such that for those cases where extrapolation is encountered in any of the input variables (d_{sh_min}, d_{sh_avg} and SI), a flag is activated; furthermore, if the predicted RF is greater than 25%, a portion (e.g., 1/3) of those cases are also flagged. A total of 7 extrapolation cases are identified in this case study based on the above selection criteria.

Once these extrapolation cases are identified, they can be used to generate additional flow simulation data and be added to the training data set. Two schemes have been implemented in which the network is updated or “re-trained” at different stages. Prior to re-training, all network parameters are initialized using the previously-tuned values; hence fewer epochs are required in each of these updating or “re-training” stages. Details of each scheme, as well as their performances, are discussed next.

4.3.1 Implementation #1 – Batch Updating

All 100 cases of the 50-layered models in the testing data set are screened for extrapolations simultaneously in a batch operation. Each case that is flagged is subjected to flow simulation and the results are added to the existing training data set, which is used subsequently to update the network parameters. The updated network model is used at the end to generate predictions for the unflagged cases in the testing set. This workflow is summarized in Figure 4.10. Together with the first training stage involving the original training data set, a total of two training stages are used to obtain the final RF predictions for all test cases. Cross-plots between actual target values and network predictions for the testing data set after ANN model updating are shown in Figure 4.11. The results show significant improvements in network predictability, particularly for RF values that are greater than 25%.

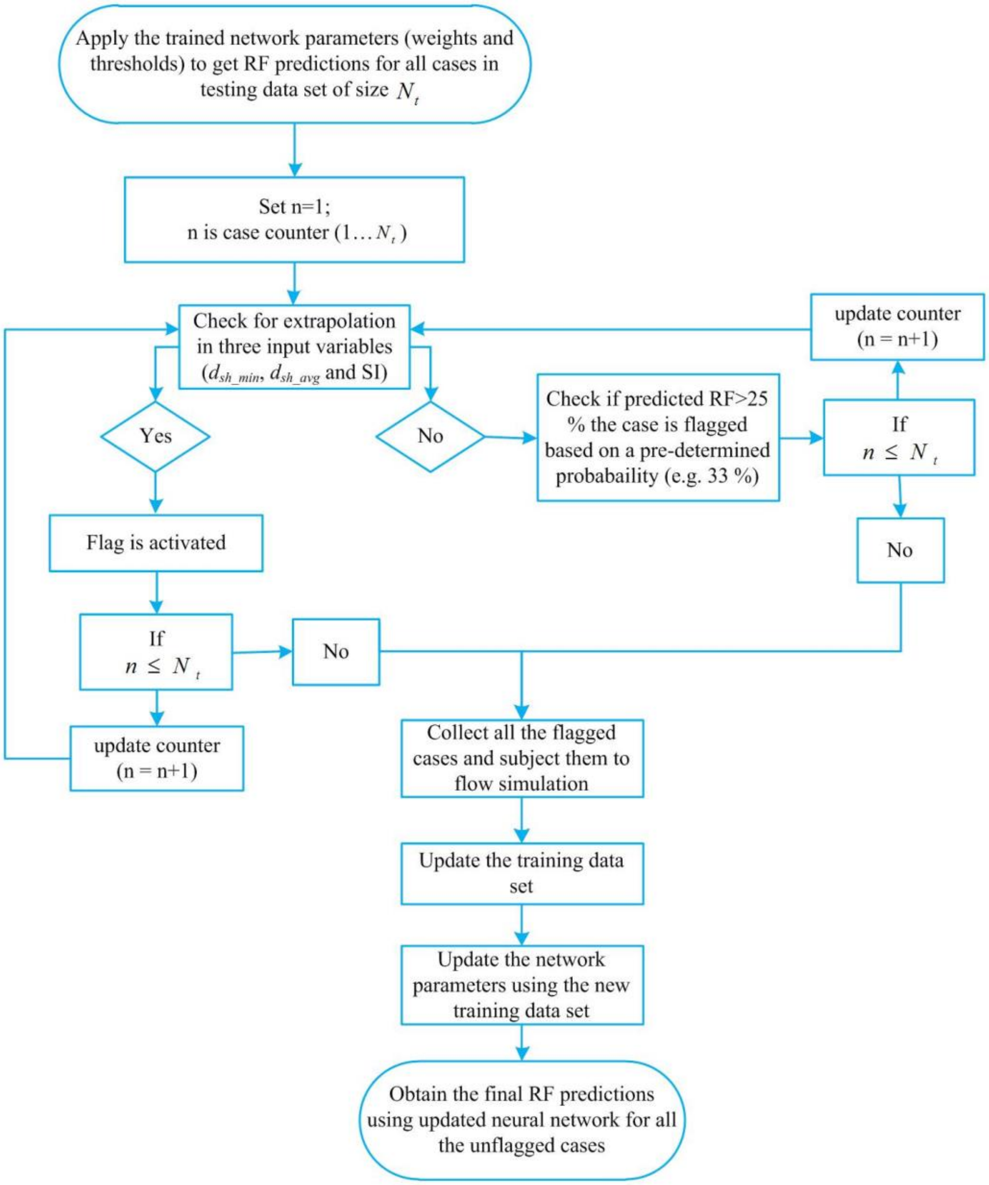

Apply the trained network parameters (weights and thresholds) to get RF predictions for all cases in testing data set of size N_t
Set n=1;
n is case counter (1… N_t)
Check for extrapolation in three input variables (d_{sh_min}, d_{sh_avg} and SI)
Yes
No
Check if predicted RF>25 % the case is flagged based on a pre-determined probabaility (e.g. 33 %)
update counter (n = n+1)
If $n \le N_t$
No
Flag is activated
If $n \le N_t$
No
update counter (n = n+1)
Collect all the flagged cases and subject them to flow simulation
Update the training data set
Update the network parameters using the new training data set
Obtain the final RF predictions using updated neural network for all the unflagged cases

Figure 4.10 Flow chart of implementation #1 in case study 3 to handle extrapolations

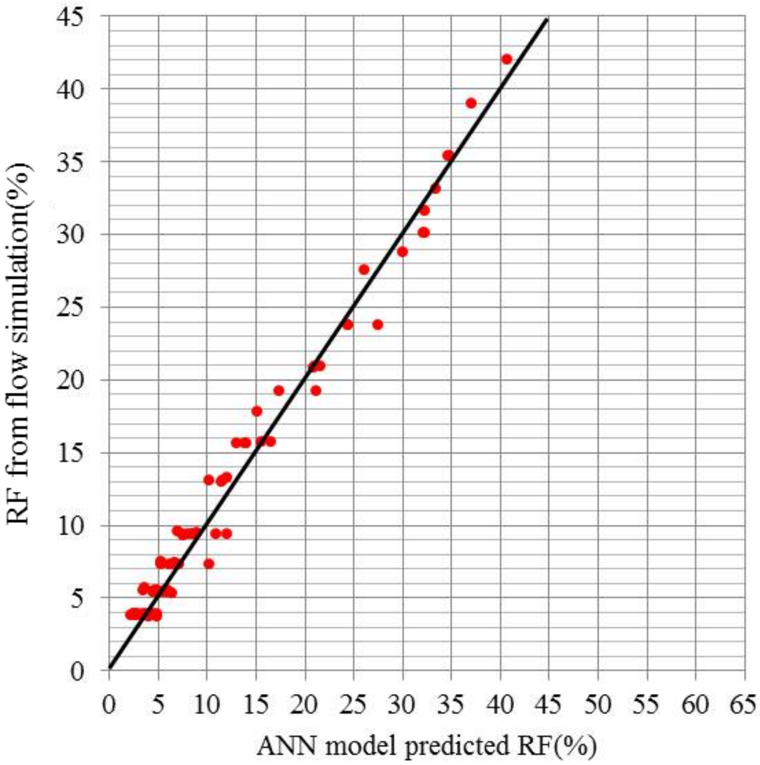


Figure 4.11 Crossplot of actual flow simulation results (target values) against network predictions for the testing data set in case study 3 with ANN model updating using implementation #1

4.3.2 Implementation #2 – Updating After Every Occurrence of Extrapolation

Cases in the testing data set are screened sequentially for extrapolations. Every time a case is flagged, it is subjected to flow simulation. The results are added to the existing training data set, which is used immediately to update the network model. The workflow of this implementation is schematically illustrated in Figure 4.12. Comparison of the ANN model predicted values with the actual target values in Figure 4.13 indicates significant improvement in the network predictability is achieved with this updating scheme.

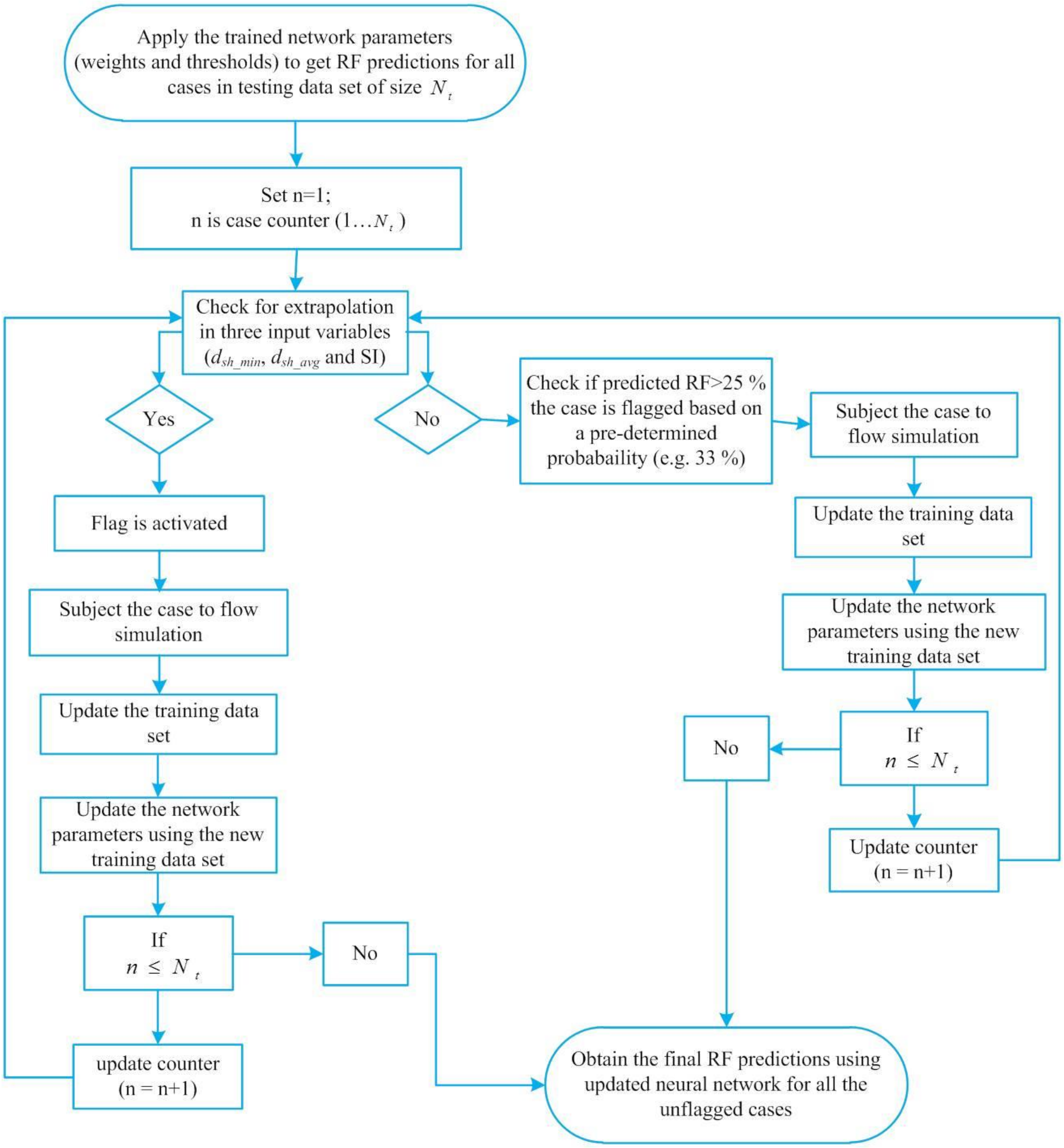

Apply the trained network parameters (weights and thresholds) to get RF predictions for all cases in testing data set of size N_t
Set n=1;
n is case counter (1…N_t)
Check for extrapolation in three input variables (d_{sh_min}, d_{sh_avg} and SI)
Yes
No
Check if predicted RF>25 % the case is flagged based on a pre-determined probabaility (e.g. 33 %)
Subject the case to flow simulation
Update the training data set
Update the network parameters using the new training data set
If $n \le N_t$
No
Update counter (n = n+1)
Flag is activated
Subject the case to flow simulation
Update the training data set
Update the network parameters using the new training data set
If $n \le N_t$
No
update counter (n = n+1)
Obtain the final RF predictions using updated neural network for all the unflagged cases

Figure 4.12 Flow chart of implementation #2 in case study 3 to handle extrapolations

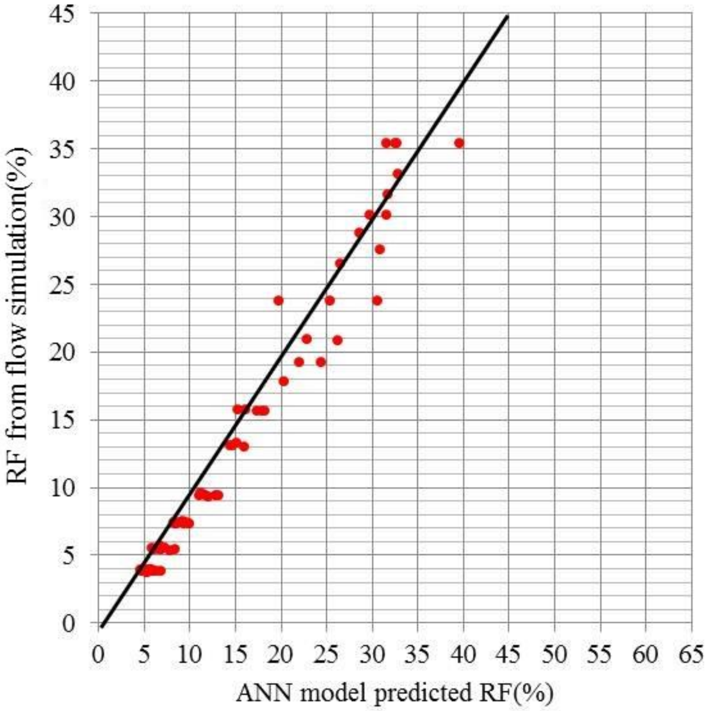


Figure 4.13 Crossplot of actual flow simulation results (target values) against network predictions for the testing data set in case study 3 with ANN model updating using implementation #2

An important consideration is computational efficiency. Additional effort is required to perform flow simulations on extrapolation cases, which are then used to re-train the network. A comparison of computational time for the base case (no network updating) and the two proposed implementations is presented in Table 4-1. Both implementations require additional flow simulations to be performed for the 7 extrapolation cases; however, a single updating/re-training stage is required in implementation #1, as opposed to the 7 additional updating stages for implementation #2. A training stage is complete when data mismatch has been reduced by 85% of its initial value. As compared to the various updating stages in implementation #2, the single updating stage in implementation #1 employs fewer epochs but entails slightly higher computational time. Re-training the network parameters with the entire expanded training data set of all original 100 cases plus 7 extrapolation cases in a batch operation involves higher computational effort per epoch. Nevertheless, less extra cost is incurred in implementation #1 when considering the total number of epochs and computational time for all training stages. Also, since in each updating stage, the network parameters are initialized using values obtained from the last training stage, the number of epochs required in a subsequent updating stage is less than the first training stage. It should be emphasized that in order to incorporate more new information in a training data set, the amount of time needed to achieve the same level of error reduction would also increase.

Table 4-1 Comparison of computational time in case study 3

	No updating due to extrapolations	Implementation 1 – batch updating	Implementation 2 – updating after every occurrence of extrapolation
Size of original training data set	100	100	100
# Extrapolation cases	0	7	7
# Flow simulations	100	107	107
Computational time for each flow simulation(sec)	360	360	360
# Training stages	1	2	8
# Epochs in each stage (training terminates when error reduction exceeds 85% of its initial value)	{1148}	{1148, 768}	{1148, 948, 935, 915, 870, 855, 828, 822}
Computational time for each training cycle	{10}	{10, 11}	{10, 10, 10, 10.5, 10.8, 10.8, 11, 11}

Comparing Figure 4.11 with Figure 4.13, it can be observed that implementation #1 provides a better match between actual target values and network predictions. Updating network parameters after every occurrence of extrapolation leads to overfitting. In many practical applications where training data is comprised of real-time actual measurements that are prone to errors, our results suggest that periodic updating of network models in a batch mode could improve network predictability.

4.4 Conclusion

In this chapter, 3 case studies were presented and can be summarized as following:

- Case study 1:using both operational and geological parameters as input variables, ANN was employed to predict recover factor as a sole output/target value in a heterogeneous reservoir. A neural network of 2 hidden layers with 8 and 16 nodes in the 1st and 2nd hidden layers was used in this study. Presented results revealed good agreement between recovery factor values recorded from flow simulator and predicted results from proxy model.
- Case study 2: additional key predicting variables that would have significant impacts on recovery performance were identified. ANN model was constructed involving one output/target variable (RF) and a total of eight input variables. An input variable, SI, was proposed as a new normalized shale continuity indicator defined as the thickness-to-distance ratio of the shaly layer located at the shortest distance to the injector. Actual target values and network predictions for the training and testing data sets were in a good agreement.

- Case study 3: two implementations were proposed to improve the robustness of network predictions in cases of extrapolations. In batch updating, all cases in the testing data set were screened for extrapolations simultaneously in a batch operation. Each case that was flagged was subjected to flow simulation and the results were added to the existing training data set, which is used subsequently to update the network parameters. In second implementation, updating is performed after every occurrence of extrapolation: every time a case was flagged, it was subjected to flow simulation. The results were added to the existing training data set, which were used immediately to update the network model. Implementation #1 provided a better match between actual target values and network predictions. Updating network parameters after every occurrence of extrapolation leads to overfitting. Our results suggest that periodic updating of network models in a batch mode could improve network predictability, in most practical applications where training data is comprised of real-time actual measurements that are prone to errors.

Chapter 5: An Integrated Application of Cluster Analysis and Artificial Neural Networks for SAGD Recovery Performance Prediction in Heterogeneous Reservoirs

Two case studies are presented in this chapter to demonstrate the application of the proposed modeling framework for SAGD performance recovery. Each set of input parameters representing a particular heterogeneous reservoir is subjected to numerical SAGD flow simulation; the resultant oil recovery profile is recorded as the output response.

These numerical simulation cases are assembled into a comprehensive data set for training and testing the cluster-ANN models. Selection of input parameters will be discussed in detail.

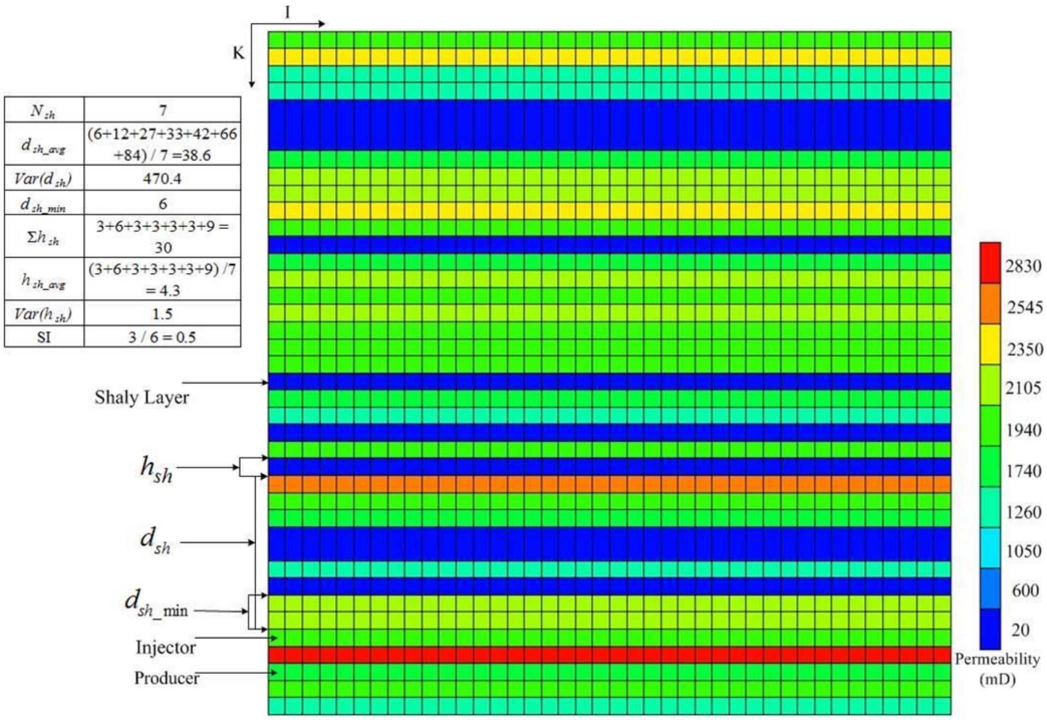
5.2 Construction of Training Data Set

A total of 150 layered reservoir models with varying porosity and permeability values between layers are generated according to three heterogeneous reservoir settings: low-side, expected, and high-side. For each setting, 50 reservoir models are built. Each model is in 2-D (in the x-z plane) with 40 3m-thick layers. Sand porosity and permeability values for each layer are sampled randomly from normal distributions, whose statistics are summarized in Table 5-1.

Table 5-1 Statistics of porosity and permeability distributions corresponding to each heterogeneous reservoir setting

Reservoir setting	Permeability		Porosity	
	mean (μ_k, D)	standard deviation (σ_k)	mean (μ_ϕ)	standard deviation (σ_ϕ)
#1 low-side	0.1	0.04	0.25	0.04
#2 expected	1.5	0.04	0.30	0.04
#3 high-side	4.5	0.04	0.40	0.04

Additional low-permeability or shaly layers (ϕ = 15% and k = 20mD) with different thicknesses are randomly placed within each reservoir. An example of a model constructed based on reservoir setting #2 is shown in Figure 5.1.



N_{sh}	7
d_{sh_avg}	(6+12+27+33+42+66 +84) / 7 =38.6
$Var(d_{sh})$	470.4
d_{sh_min}	6
Σh_{sh}	3+6+3+3+3+3+9 = 30
h_{sh_avg}	(3+6+3+3+3+3+9) /7 = 4.3
$Var(h_{sh})$	1.5
SI	3 / 6 = 0.5

Figure 5.1 An example of expected reservoir setting model with 7 randomly-distributed shaly (low-permeability) layers for case study 1

Two additional parameters are introduced for characterizing each shaly layer: h_{sh} (thickness) and d_{sh} (distance to the injection well). Properties of reservoir simulation model are tabulated in Table 5-2.

Table 5-2 Properties of the dynamic flow simulation model

Property	Value	Unit
Bitumen properties	Viscosity: 41	Pa.s
	API: 10	°API
Initial reservoir temperature	15	°C
Initial reservoir pressure	1000	kPa
Operating bottom-hole pressure	Injector: 2526	kPa
	Producer: 1560	kPa
Rock heat capacity	2.35e6	J/(m^3.°C)
Rock thermal conductivity	1.35	W/(m.°C)

A total of twelve input variables are identified including N_{sh} (number of shaly layers in each model), d_{sh_avg} (average distance of shaly layers to the injection well), $Var(d_{sh})$ (variance of the distance of shaly layers to the injection well),

d_{sh_min} (d_{sh} of the shaly layer that is located at the shortest distance to the injection well), Σh_{sh} (total thickness of shaly layers), h_{sh_avg} (average thickness of shaly layers), $Var(h_{sh})$ (variance of the thickness of shaly layers), ϕ_{avg} (average porosity), $Var(\phi)$ (variance of porosity), k_{avg} (average permeability), $Var(k)$ (variance of permeability) and shale indicator (SI). The variable (SI) is a normalized shale continuity indicator defined as the thickness-to-distance ratio of the shaly layer located at the shortest distance to the injector, i.e., h_{sh_min}/d_{sh_min}: a large value would indicate a thick shale barrier that is located close to the injector, impeding the formation of steam chamber.

In this chapter, these 150 reservoir models are subjected to the flow simulator. In case study 1, recovery factor (RF) defined as the cumulative production divided by the total in-place oil volume is considered as the output variable for the cluster-ANN model, while Arps decline parameters (b, D_i, and q_i) are utilized to parameterize the entire oil production time series in case study 2.

PCA is applied to the input variables to reduce the dimensionality of the data set. The cumulative variance is plotted to identify how variability within the data set is distributed among the corresponding principal components. As illustrated in Figure 5.2, the first two components capture the majority of variability within the data set. Using only these two principal scores (PSs) as input parameters and 120 cases as training data set for ANN modeling do not provide satisfactory results: even with a large number of epochs, a match could not be achieved between the model predictions and the target values during the training stage.

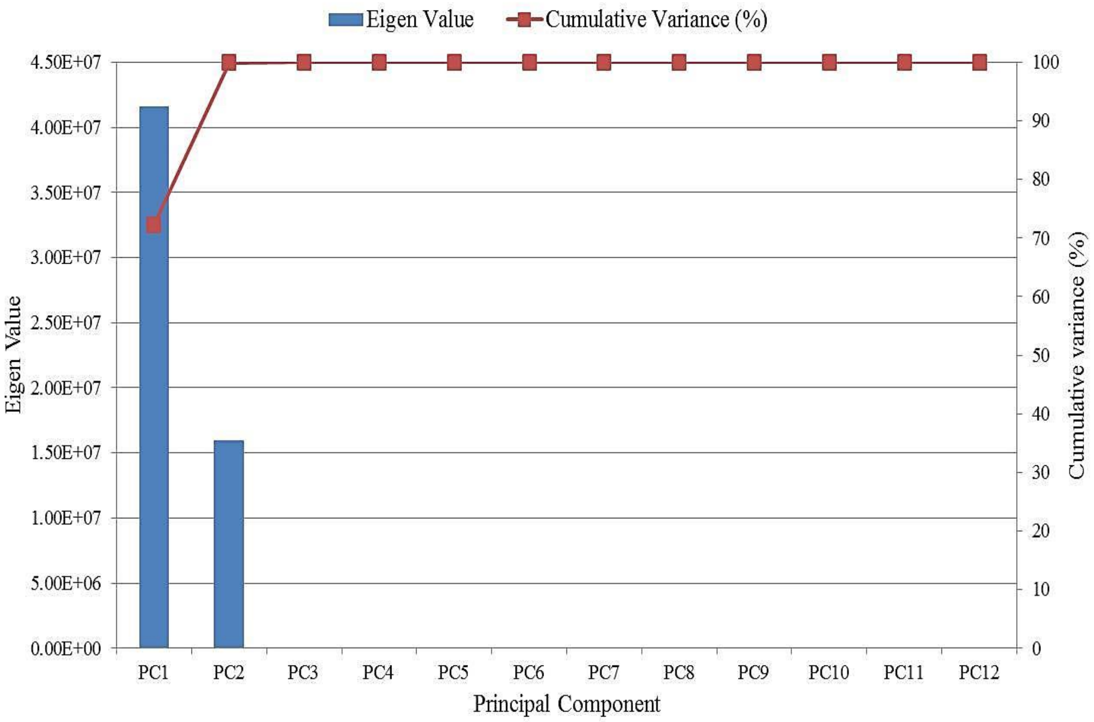
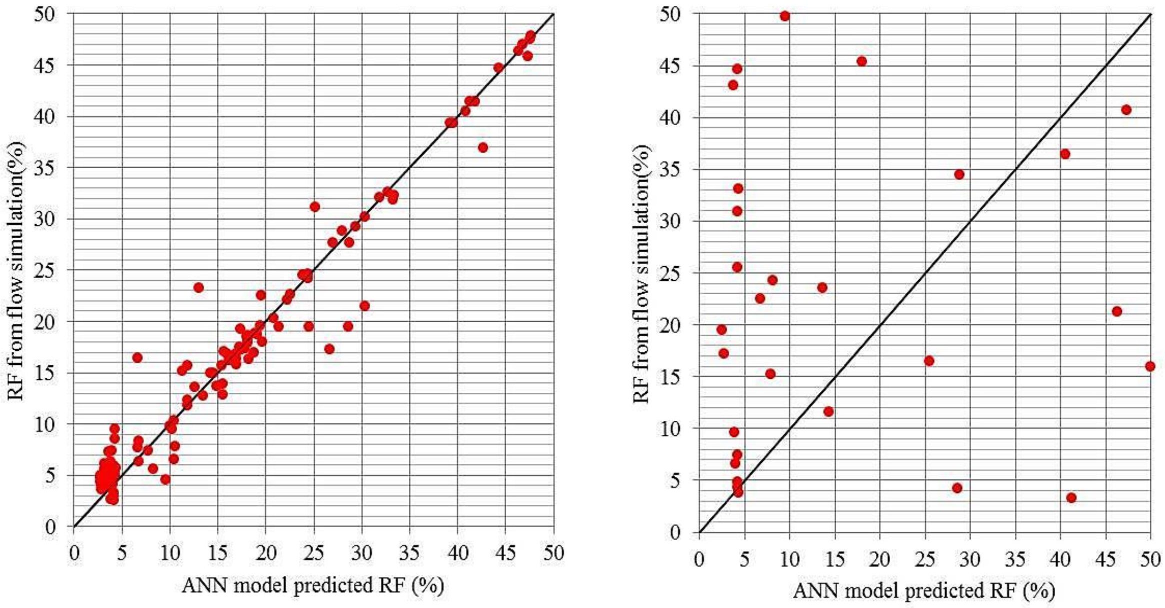


Figure 5.2 Pareto-screeplot of the principal component analysis (PCA)

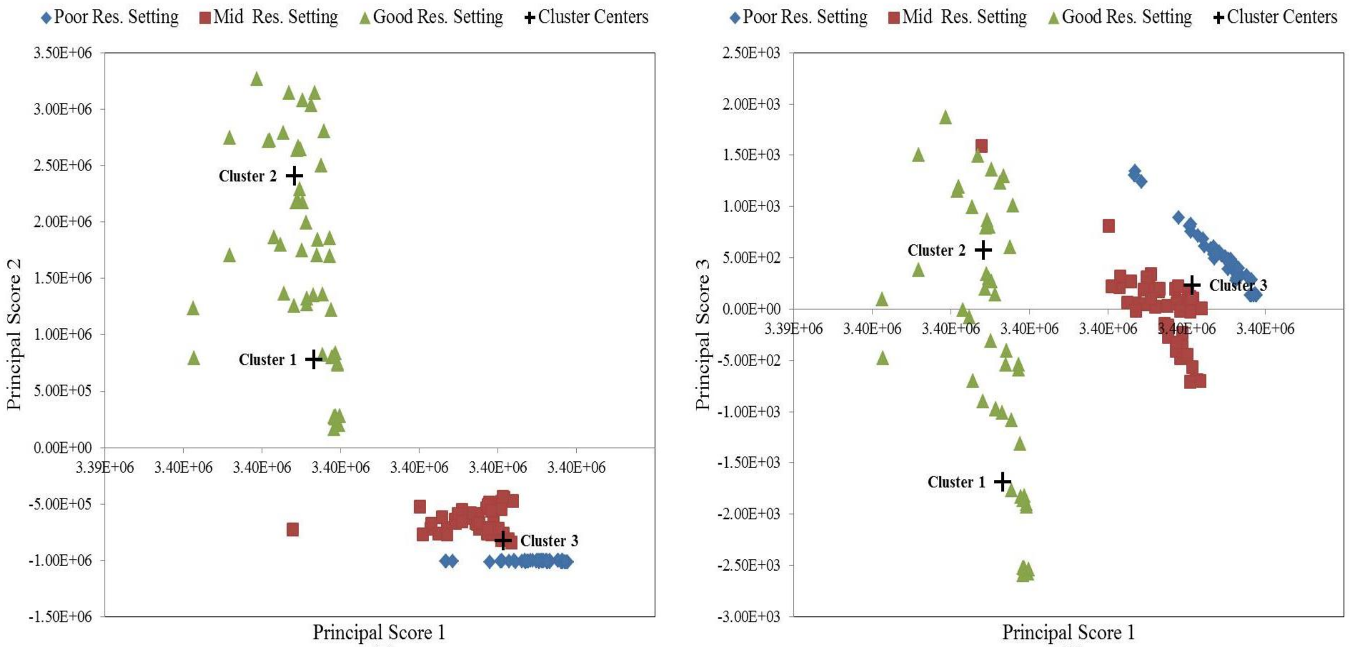
Increasing the number of principal components improves the training results, but results with the testing/validation cases are poor, as shown in Figure 5.3. It is concluded that given that internal patterns exist among the data set, separate ANN model should be developed for each individual grouping. The top three principal scores for all 150 cases are plotted against each other in Figure 5.4, and their corresponding reservoir settings are also labeled on these plots.



(a)

(b)

Figure 5.3 Crossplot of ANN model predicted RF against simulator output RF, using 9 PCs; (a) Training data set, (b) Test data set.



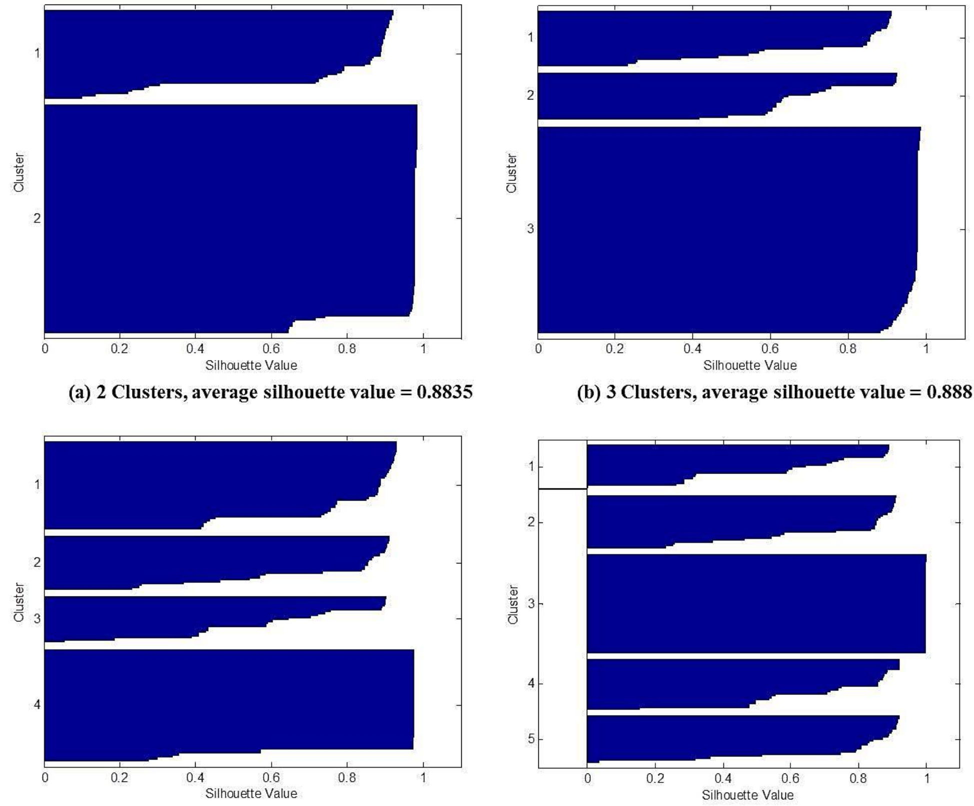
(a)

(b)

Figure 5.4 Scatter-plot between principal scores: (a) Principal scores 1 & 2; (b) Principal scores 1 & 3

5.3 Case Study 1: K-Means Assisted Artificial Neural Networks for Recovery Factor Prediction

A statistical measure called average silhouette value can be used to identify the optimum number of clusters, for which the average silhouette value would be the maximum. Figure 5.5 illustrates cluster numbers against silhouette value for two, three, four and five clusters. As shown in this plot, the optimal number is three with the maximum average silhouette value.



(a) 2 Clusters, average silhouette value = 0.8835

(b) 3 Clusters, average silhouette value = 0.8881

(c) 4 Clusters, average silhouette value = 0.8088

(d) 5 Clusters, average silhouette value =0.8052

Figure 5.5 Silhouette plots and average silhouette values for 2, 3, 4 and 5 clusters for case study 1

The entire data set of 150 cases is classified into 3 clusters. For each cluster, a separate ANN model is trained using 75% of the data and tested using the remaining 25% of the data. This split of data into training and testing has been proposed by previous researchers. Our sensitivity analysis on training and testing percentage for this case study also revealed that 75% of the data can be used in training and the rest as testing data set. Once again, PSs and RF (%) are considered as input and output variables, respectively, in the ANN models. To investigate the sensitivity to data dimensionality reduction, different numbers of PSs are employed. Cross-plots between actual target values and network predictions for the training and testing data sets using 9 PSs are shown in Figure 5.6. Good agreement between the actual values and predictions can be observed.

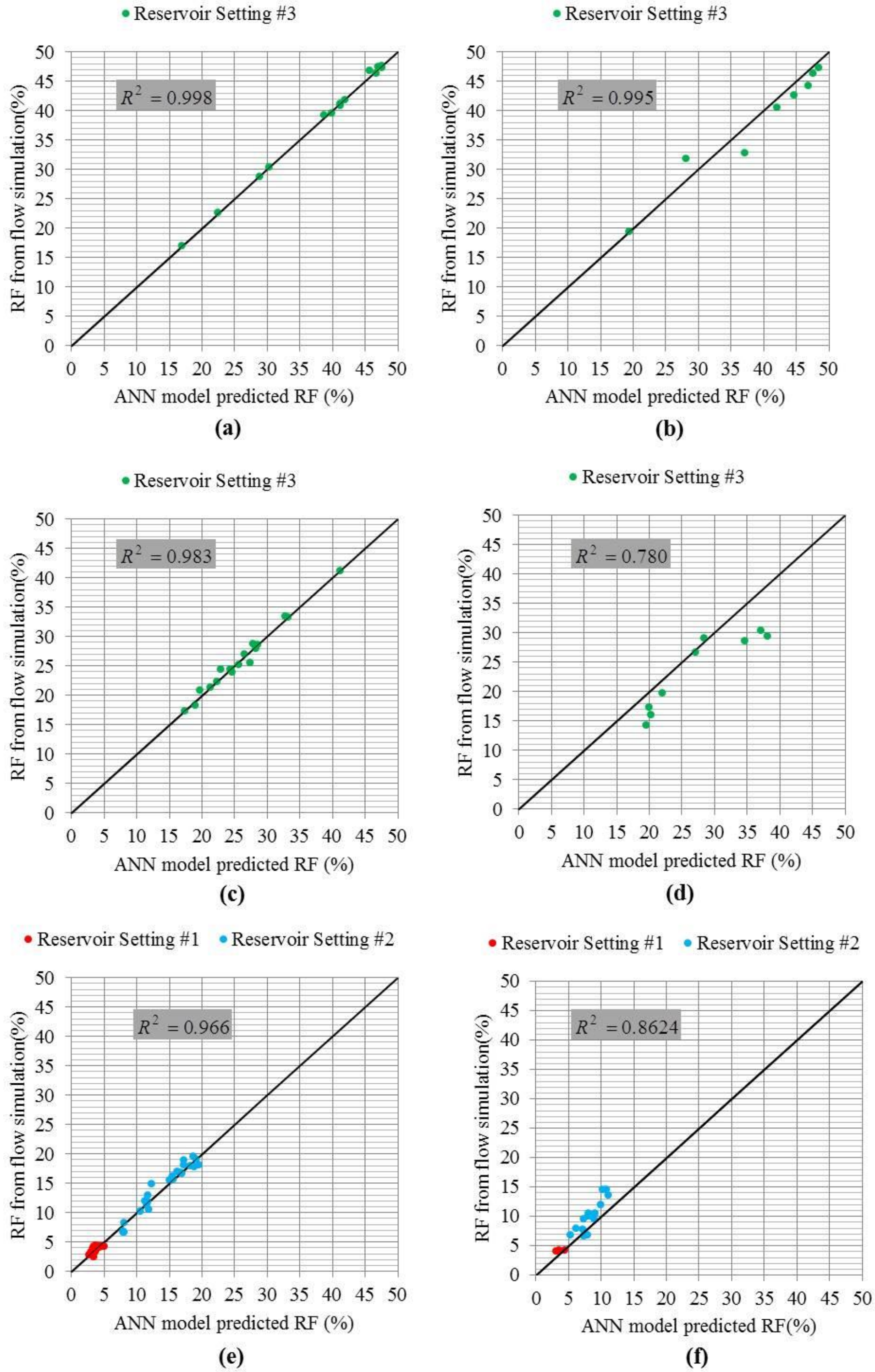


Figure 5.6 Cross plot of actual flow simulation results (target values) against network predictions using 9 PSs for case study 1:
a) Cluster 1, training data set, b) Cluster 1, testing data set
c) Cluster 2, training data set, d) Cluster 2, testing data set
e) Cluster 3, training data set, f) Cluster 3, testing data set

Comparing Figure 5.3 & Figure 5.6 reveals that clustering the original data set into subsets has increased the ANN predictability. Data in clusters 1 and 2 are all coming from reservoir setting #3 shown in green color, and data from reservoir settings #1 and #2 indicated by red and blue markers, respectively, are included in cluster 3. It suggests that there is more variability among the models generated using reservoir setting #3 than those using reservoir settings #1-2. This is because the non-linear relationship between permeability and porosity is more exaggerated for high porosity values (as depicted in reservoir setting #3). Reducing the number of PSs to either 7 or 5 does not have a significant impact on the network predictability. As shown in Figure 5.7 and Figure 5.8, the predicted values are still in a good agreement with the actual values. Comparing Figures 5.6-5.8, it is interesting to note that reducing the number of PSs could potentially improve the ANN predictability with the testing data for cluster no. 2. This observation might be attributed to the ability to avoid overfitting with fewer input variables.

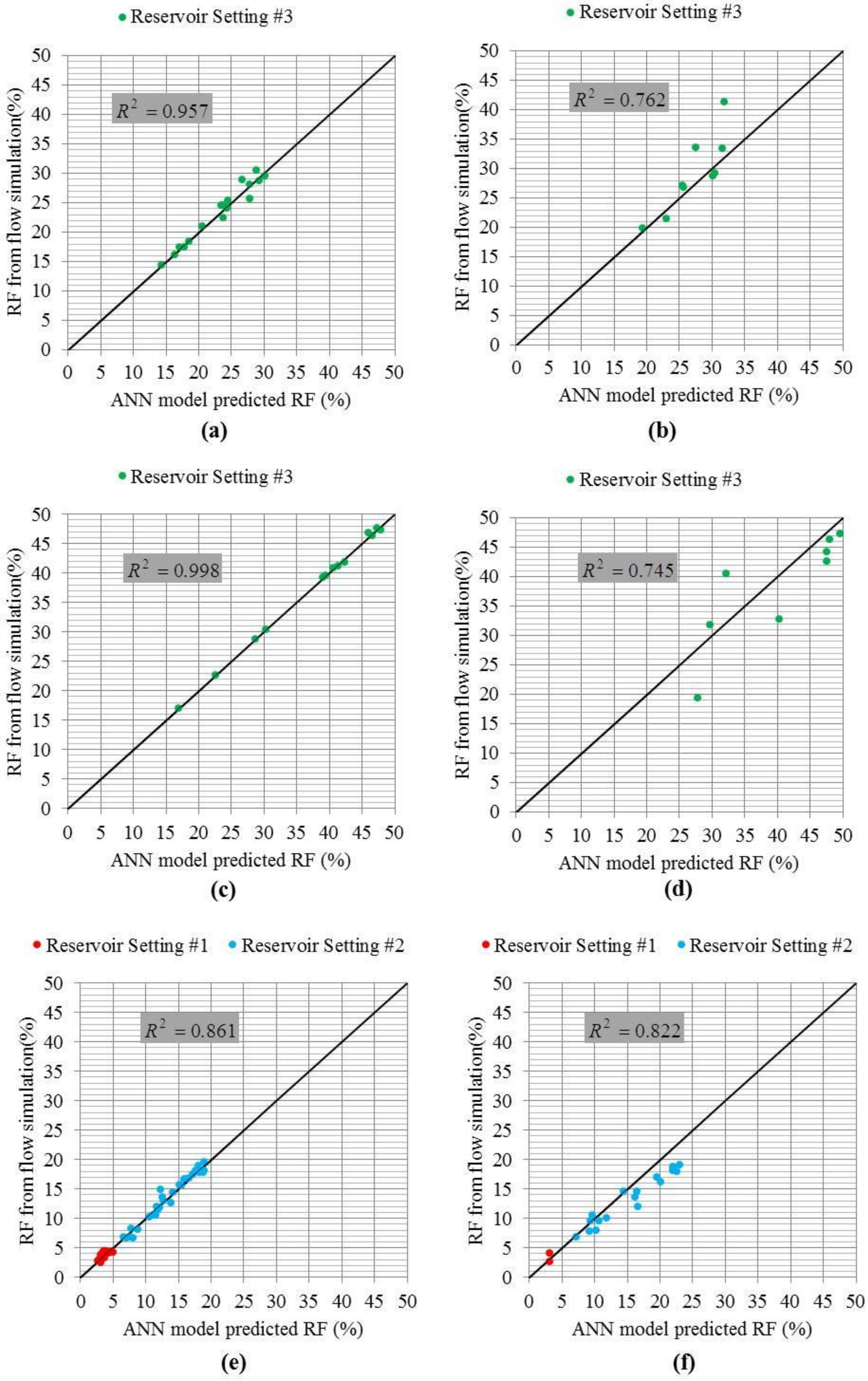


Figure 5.7 Cross plot of actual flow simulation results (target values) against network predictions using 7 PSs for case study 1:
a) Cluster 1, training data set, b) Cluster 1, testing data set
c) Cluster 2, training data set, d) Cluster 2, testing data set
e) Cluster 3, training data set, f) Cluster 3, testing data set

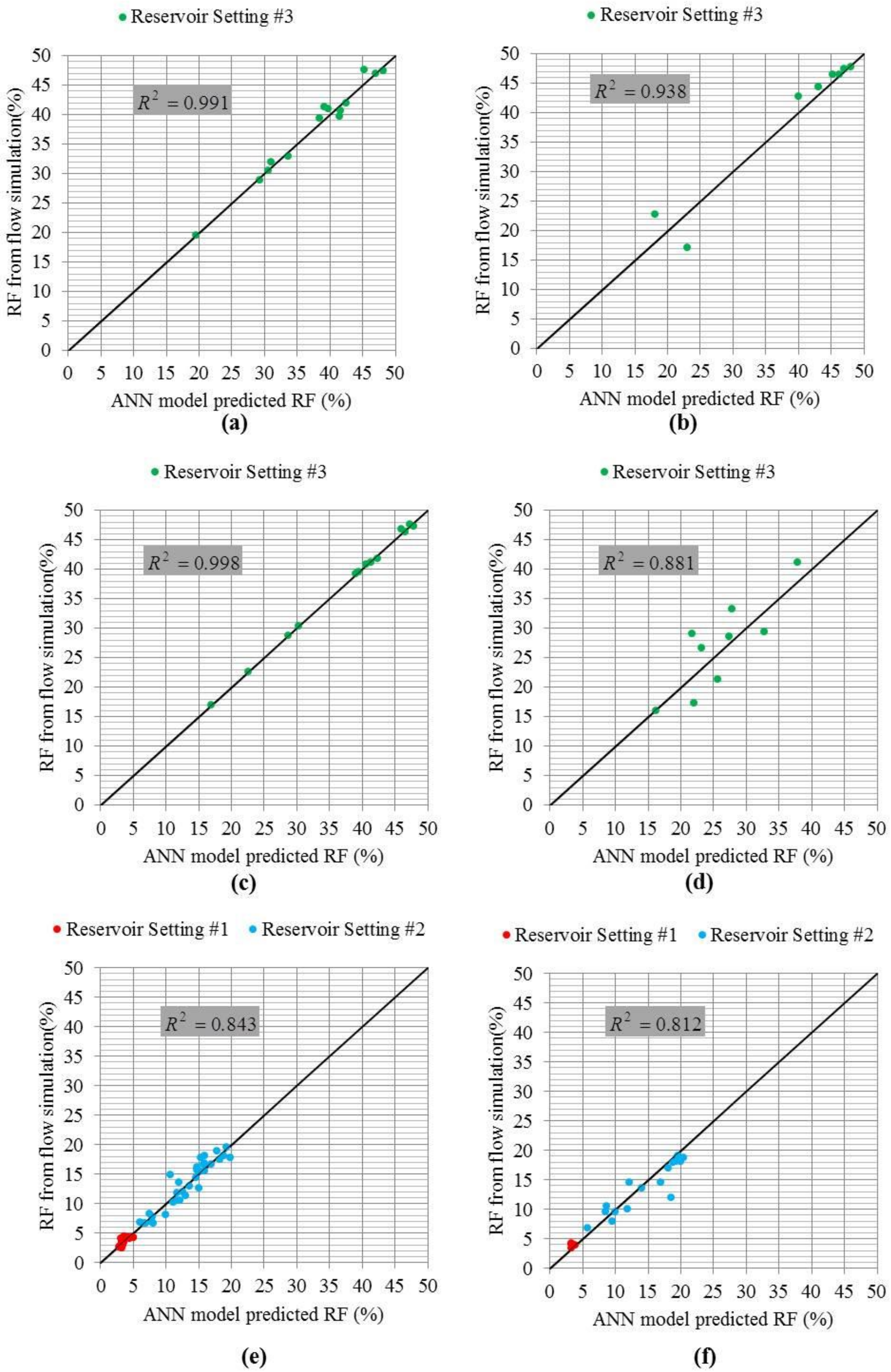


Figure 5.8 Cross plot of actual flow simulation results (target values) against network predictions using 5 PSs for case study 1:
a) Cluster 1, training data set, b) Cluster 1, testing data set
c) Cluster 2, training data set, d) Cluster 2, testing data set
e) Cluster 3, training data set, f) Cluster 3, testing data set

Finally, a set of 9 new reservoir models are generated for further validation/testing purposes. As each data point is presented to the cluster-ANN model, it is assigned to a particular cluster with the minimum Euclidean distance between the data point and the cluster center. Weights of the ANN model trained and tested in the previous step are applied to this new data point, and a predicted value of RF is obtained. This result is compared against the actual RF obtained by subjecting the reservoir model to detailed flow simulation. Figure 5.9 compares the predicted values from ANN model with actual values obtained from flow simulation for these nine cases. The validation results show good agreement with simulation results.

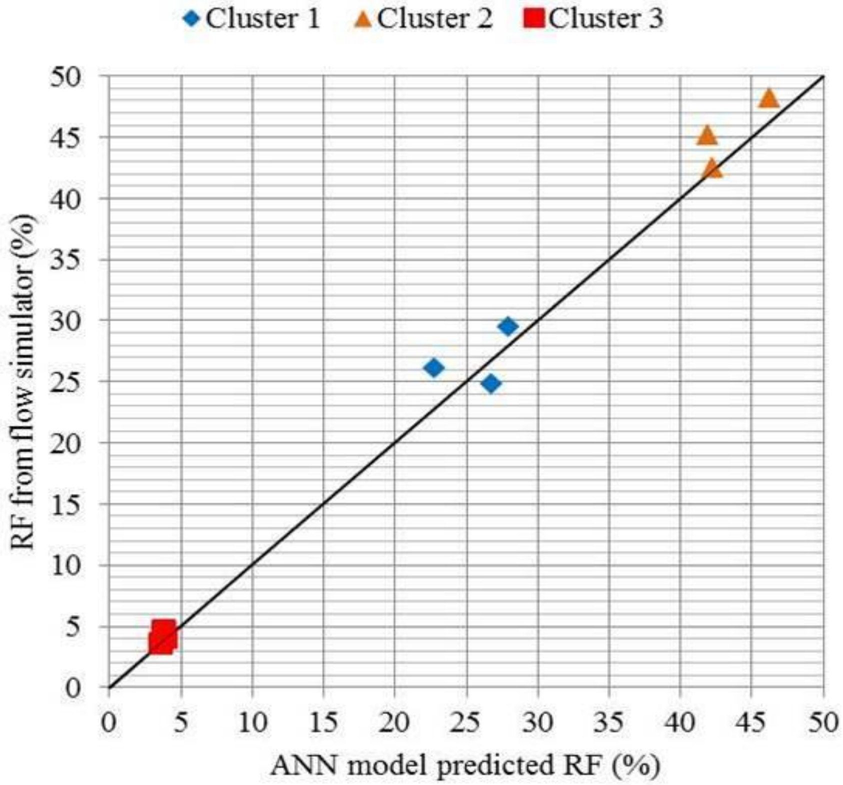


Figure 5.9 Cross plot of actual flow simulation results (target values) against network predictions for nine new cases for case study 1

It is clear from the results that clustering data into several subsets prior to ANN modeling has significantly improved the general predictability and robustness of the model over a range of reduced data dimensionality. The integrated procedure helps to recognize hidden patterns within the data set, and training separate ANN model for each identified pattern is important in capturing the highly non-linear relationship exhibited by the data. PCA aims to project the data in the original space onto a set of orthogonal variables, without altering the relative distance among the data points. Therefore, k-means clustering in either the physical space (original data space) or the transformed space does not have significant impact on the ensuing solution. However, performing clustering prior to PCA can capture local variance exhibited within an individual cluster, but global variance exhibited by the entire data set might not be fully taken into account.

5.4 Case Study 2: Fuzzy-Based Artificial Neural Networks for Oil Production Time Series Prediction

The data-driven modeling techniques are applied to predict cumulative oil production (Q_p) profile. Empirical Arps decline parameters (b, D_i, q_i) are used for parameterization of cumulative production profile of all 150 reservoir models and considered as outputs of the ANN models. A curve-fitting procedure that minimizes the RSME described in Eq.(3-28) is applied to the cumulative production data for all 150 reservoir model; the corresponding Arps decline parameters are calculated and recorded.

Parameterization results reveal that the empirical parameter b is approximately one for the 50 reservoir models constructed based on setting #1, which corresponds to a harmonic Arps decline curve equation Eq. (3-25. As a result, only two output variables, D_i *and* q_i, are assigned to these 50 reservoir models in the data set. Now that the data set contains cases with different number of output variables, it is proposed that two separate data-driven models should be considered for those reservoir models with only two output variables (D_i and q_i) and those with all three output variables (b, D_i, q_i).

To investigate the feasibility of parameterization with Arps decline variables, ANN modeling is performed with the 50 cases constructed based on reservoir setting #1, in which 44 cases are used for training, while 6 cases are selected as testing data. Cumulative oil production profile of actual flow simulation results (target values) are compared against network predictions for each testing case in Figure 5.10, where 12 PSs are used as input variables. Modeling is later repeated using 6 PSs to investigate the effect of dimensionality reduction, and the results are illustrated in Figure 5.11. Comparing Figure 5.10 with Figure 5.11, it is noted that reducing the number of input variables would result in good agreement between actual target values and network predictions.

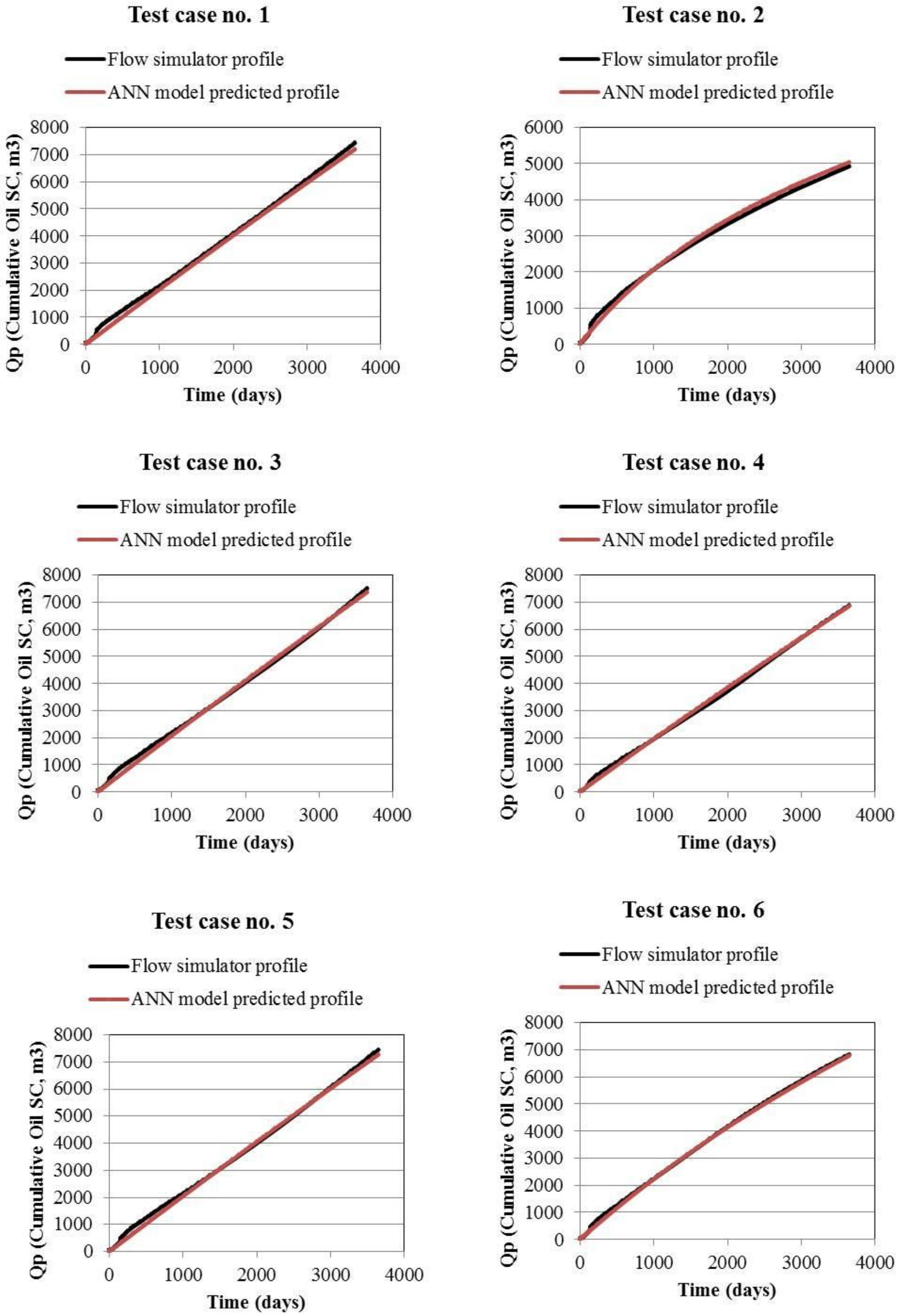


Figure 5.10 Cumulative oil production profile of actual flow simulation results (target values) against network predictions for each testing case using 12 PSs for reservoir setting #1 in case study 2

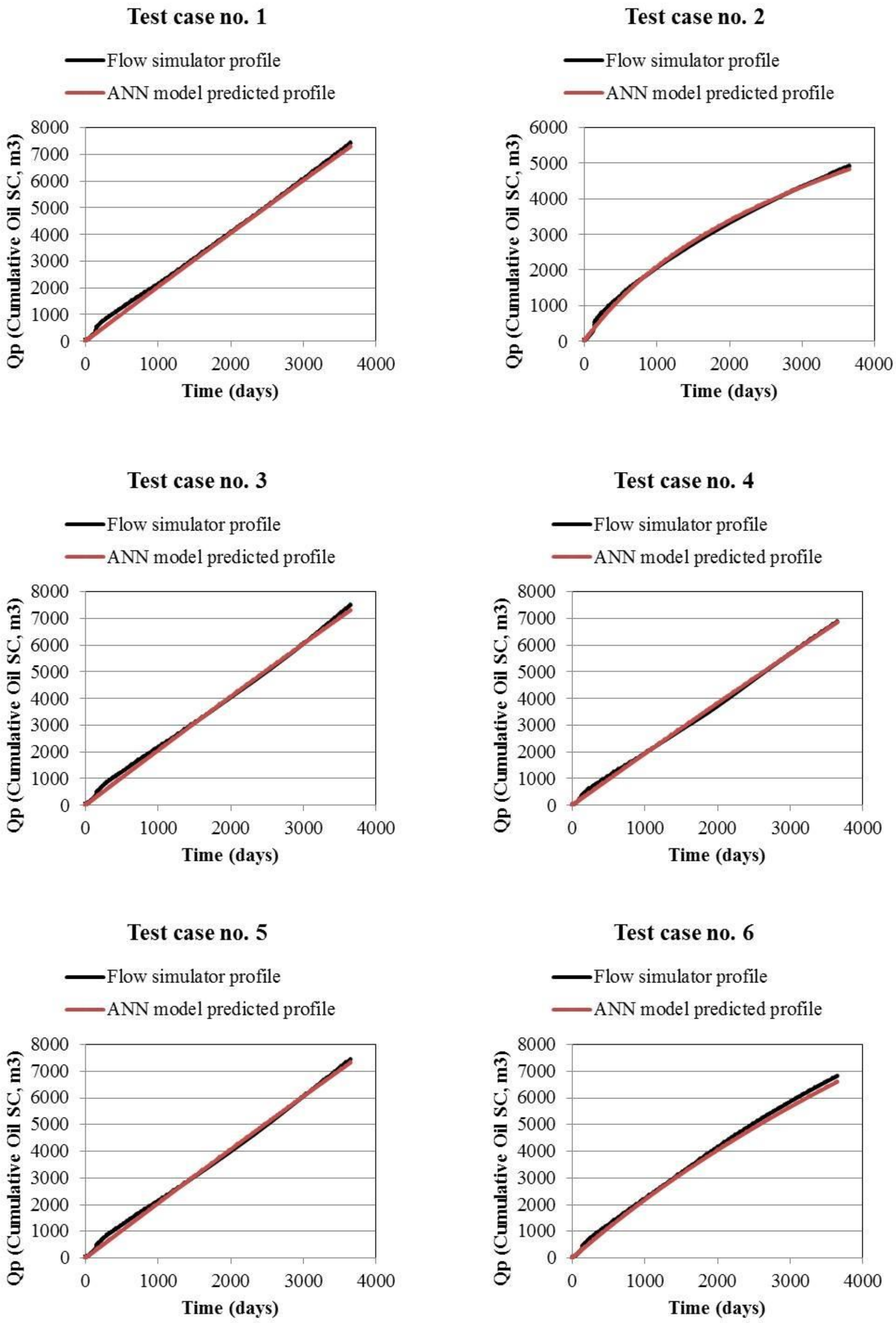


Figure 5.11 Cumulative oil production profile of actual flow simulation results (target values) against network predictions for each testing case using 6 PSs for reservoir setting #1 models in case study 2

Table 5-3 summarizes quantitatively the root mean square error (RMSE) of the cumulative production match using Eq.(3-28 for all 6 testing cases; the RSME values are close to one, indicating an excellent match.

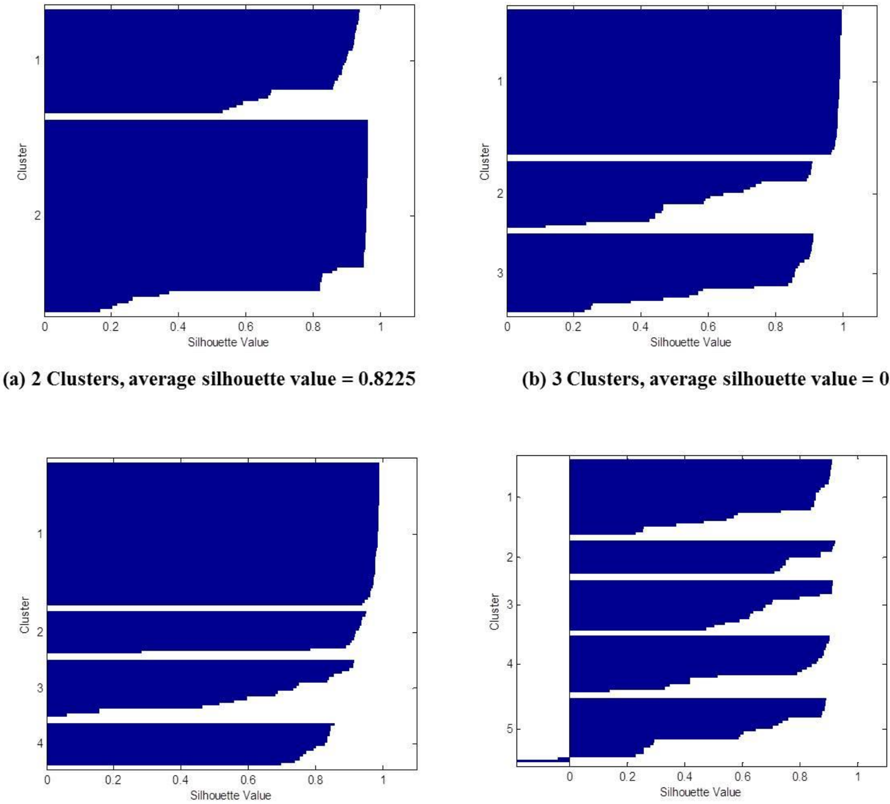
It is interesting to note that incorporating *b* (which is supposed to be constant in this case) into the ANN modeling degrades the overall prediction quality. It suggests that output variables should be selected carefully by excluding parameters that are redundant but including those that are strongly correlated to the input variables. In this case, the prediction variables, D_i and q_i, are highly sensitive to reservoir heterogeneities and properties of shaly layers.

Table 5-3 RMSE values calculated using Eq. 3-28 for the entire production period for the 6 testing cases corresponding to reservoir setting #1 in case study 2

RMSE		
Case no.	12 PSs	6 PSs
1	0.998	0.998
2	0.996	0.995
3	0.999	0.999
4	0.999	0.998
5	0.999	0.999
6	0.999	0.999

5.4.1 Integration of Fuzzy C-Means Clustering

There are 100 cases (corresponds to reservoir settings #2 and #3) remaining from the original data set. A subset of 88 out of 100 cases is selected as training data set, while the other 12 cases are designated as validation/testing data. K-means clustering is initially attempted. Based on Figure 5.12, the maximum average of silhouette value corresponds to 3 clusters. The training data set is subsequently assigned into 3 clusters as shown in Figure 5.13, and the 12 validation cases are denoted by blue circle markers.



(a) 2 Clusters, average silhouette value = 0.8225

(b) 3 Clusters, average silhouette value = 0.8401

(c) 4 Clusters, average silhouette value = 0.8330

(d) 5 Clusters, average silhouette value =6990

Figure 5.12 Silhouette plots and average silhouette values for 2, 3, 4 and 5 clusters for case study 2

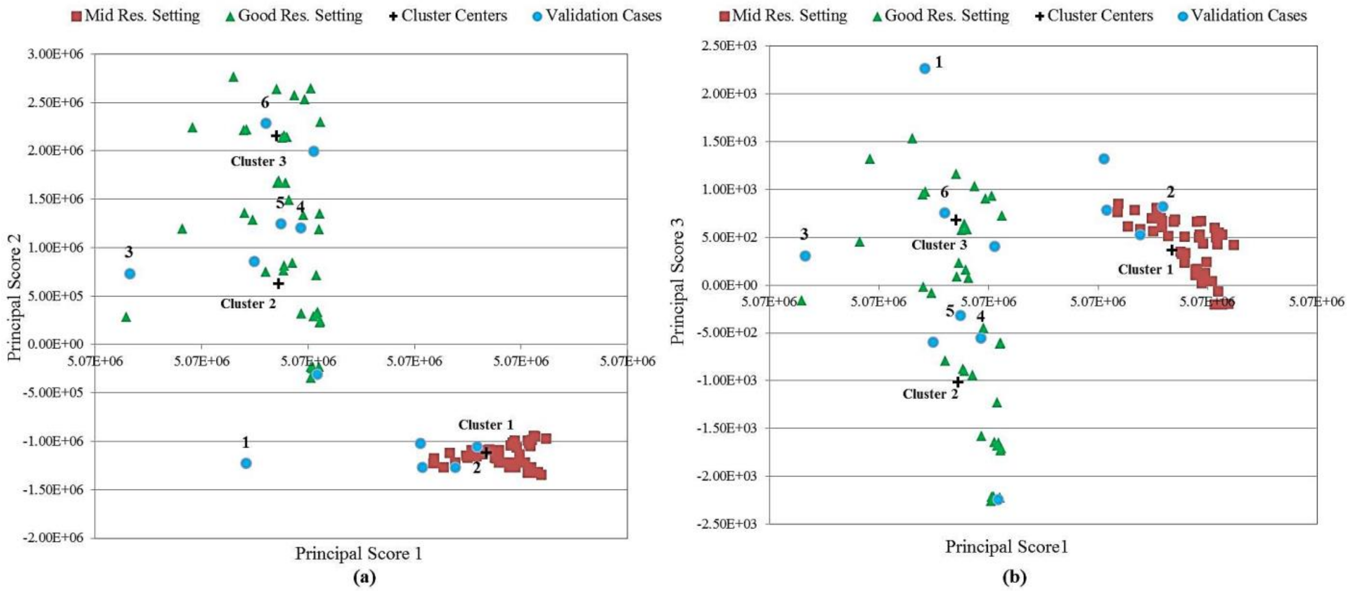


Figure 5.13 Scatter-plot between principal scores for case study 2: (a) Principal scores 1 & 2; (b) Principal scores 1 & 3

Applying k-means clustering prior to ANN modeling, in a fashion similar to the procedure described in case study 1, does not produce satisfactory results this time with three output variables (b, D_i, q_i). More training data is needed when the number of output variables increases. The problem is further exacerbated for data points that are located near the cluster boundaries. K-means clustering categorizes all cases deterministically based on minimum Euclidian distance, which leads to the 0 and 1 binary belonging degrees.

We then explore the use of fuzzy c-means (FCM) clustering method alongside with ANN model. FCM is applied to the data set and the degrees of membership are calculated according to Eq. 3-23. Summation of the three membership values for the three clusters equals to one. Once again, separate ANN model is trained and tested for each cluster, with 75% of the data for training and the remaining 25% for testing.

The split percentage of training and testing data set in this case study is based on the proposed split by the previous researchers. Our sensitivity analysis also confirmed this percentage to be efficient in terms of network predictability. The number of validation cases should be also between 10 to 15 % of the original data set. In this case study, the models are tested using the 12 validation cases (12% of the original data set), which have not been presented to the cluster-ANN modeling previously. Cluster membership values, together with the corresponding ANN model predictions, are computed. A defuzzification scheme is proposed to calculate a weighted average, where individual ANN model prediction is weighted by the associated membership value. The approach is repeated with 12 PSs and 6 PSs as ANN input variables. The predicted oil production profiles after defuzzification for 6 randomly-selected validation cases are shown in Figure 5.14 & Figure 5.15 for 12PSs and 6 PSs, respectively.

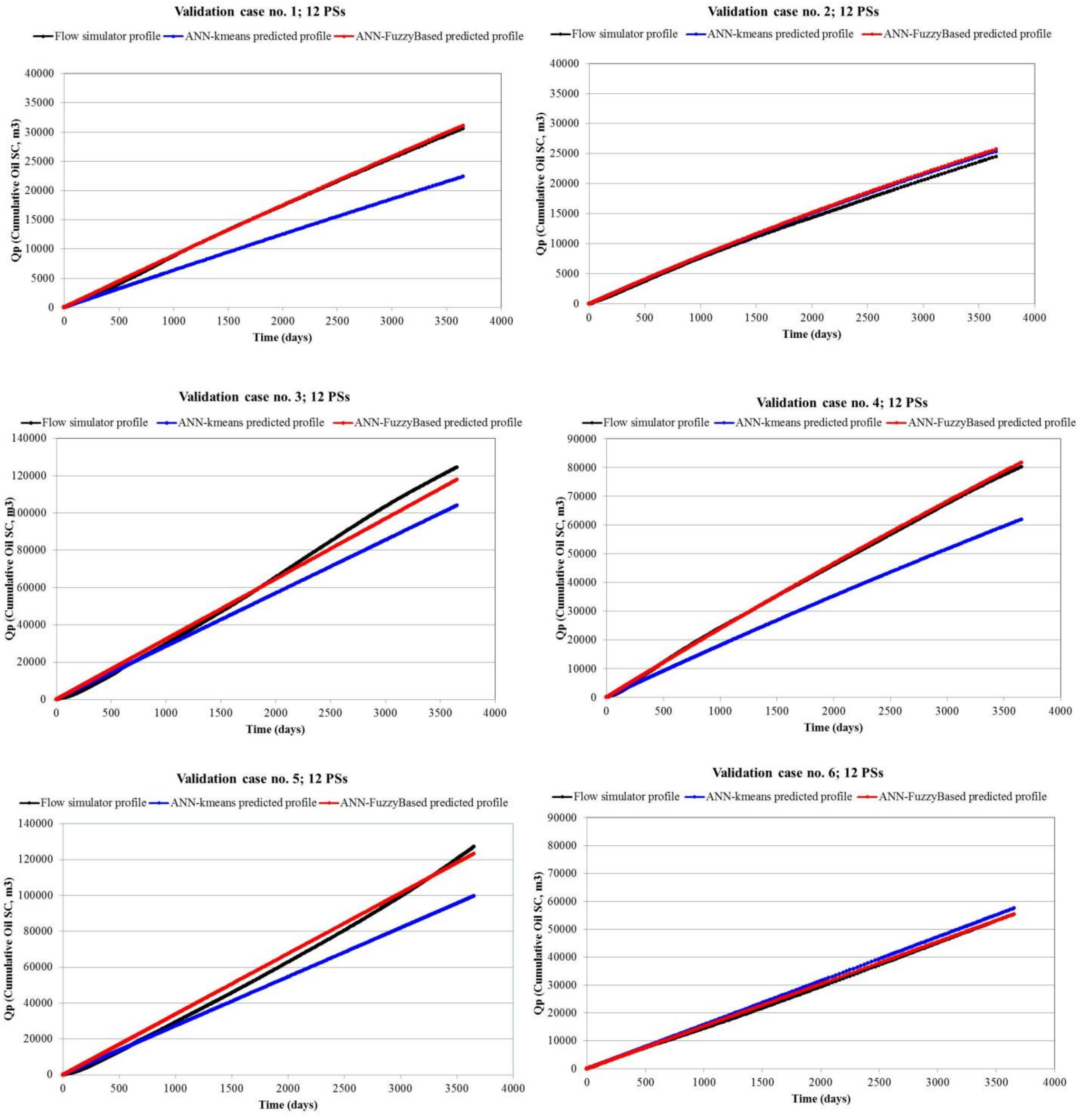


Figure 5.14 Comparison of cumulative oil production profiles of actual flow simulation results (target values) and network predictions using k-means and fuzzy-based clustering for validation cases no. 1-6 using 12 PSs in case study 2

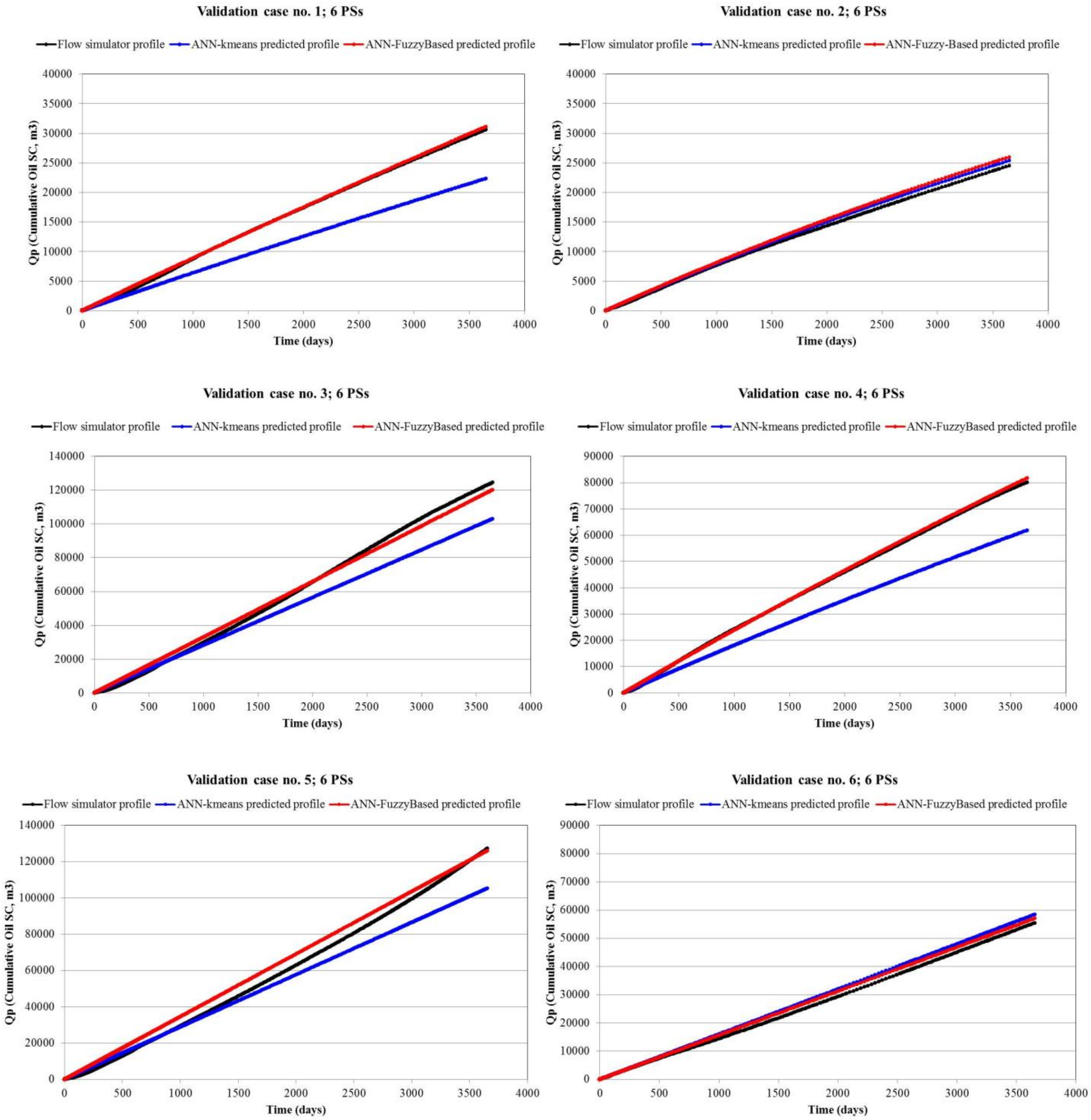


Figure 5.15 Comparison of cumulative oil production profiles of actual flow simulation results (target values) and network predictions using k-means and fuzzy-based clustering for validation cases no. 1-6 using 6 PSs in case study 2

Results for validation cases no. 1, 3, 4 and 5, which are located near the boundaries between clusters, demonstrate substantial improvement in comparison to the k-means assisted scheme. On the other hand, for validation cases no. 2 & 6 that are situated close to a particular cluster center, both k-means and fuzzy-based clustering methods yield similar results because the membership value assigned to the closest cluster is approximately equal to 1. Similar observations can be made when using 6 PSs. RMSE values representing the mismatch between predicted production profile with the actual flow simulation results are computed and tabulated in Table 5-4. Results in Figure 5.14 & Figure 5.15 and Table 5-4 encourage the integration of fuzzy-based clustering techniques with ANN model for enhancing overall predictability and efficiency of these data-driven proxies.

Table 5-4 RMSE values calculated using Eq. 3-28 for the entire production period for 6 randomly-selected validation cases in case study 2

RMSE				
	ANN-kmeans		ANN-FuzzyBased	
Validation case	12 PSs	6 PSs	12 PSs	6 PSs
1	0.843	0.861	0.997	0.995
2	0.977	0.969	0.981	0.978
3	0.876	0.872	0.974	0.979
4	0.839	0.845	0.997	0.998
5	0.848	0.852	0.985	0.980
6	0.976	0.982	0.999	0.997

5.5 Conclusion

Materials presented in this chapter can be concluded as following:

- Case study 1: integration of cluster analysis into proxy data-driven model was presented in this study. PCA and kmeans clustering methods were employed to highlight the efficient use of cluster analysis in ANN. In addition, 9 validation cases were used to verify the framework predictability. Presented results revealed that clustering data into several subsets prior to ANN modeling has significantly improved the general predictability and robustness of the model over a range of reduced data dimensionality.

- Case study 2: unlike the previous case studies, here, data-driven modeling techniques were applied to predict cumulative oil production (Q_p) profile. Empirical Arps decline parameters (b, D_i, q_i) were utilized for parameterization of cumulative production profile of all 150 reservoir models and considered as outputs of the ANN models. Furthermore, fuzzy c-means clustering method was also integrated into the proxy model. 12 validation cases, which have not been presented to the cluster-ANN modeling previously, were used to verify the network predictability. The results assured the integration of fuzzy-based clustering techniques with ANN model for enhancing overall predictability and efficiency of the data-driven proxy.

Chapter 6: Conclusion

This thesis has described the application of data-driven proxy modeling for recovery performance prediction in SAGD operations. ANN was employed to model a data-driven proxy to predict SAGD recovery in heterogeneous reservoirs. The research methodology consists of static reservoir modeling, numerical flow simulation and proxy model development, model testing and validation against flow simulation results.

The artificial neural network (ANN) approach for recovery performance prediction in SAGD operations using operational and geological predicting parameters seems very promising approach for characterizing heterogeneous reservoirs during SAGD process. This chapter summarizes the key points that can be concluded from this research.

6.1 Conclusion

- Present workflow that entails static reservoir modeling and numerical flow simulations is generally quite cumbersome and time-consuming, limiting its application in real-time optimization. Data-driven modeling processes such as artificial neural networks are still considered recent advancements and have not been widely adopted in most sectors of the oil sands industry.
- In this study, numerical flow simulations are performed to construct a training data set consisting of various input attributes describing reservoir heterogeneities and relevant production/injection parameters with the

corresponding recovery factor as output. A series of ANN models are trained using the training data set and later used as a proxy to predict the recovery factor and cumulative oil production profile as target (output) variables during SAGD process.

- In order to select the optimal network configuration, sensitivity studies are performed by comparing the error/mismatch in network prediction.
- To enhance the robustness of the modeling procedure, new schemes are implemented to identify extrapolations and periodically update the network parameters.
- Various pertinent predicting parameters in relation to SAGD recovery prediction in heterogeneous reservoirs are investigated. In addition, a new normalized shale continuity indicator (SI), which quantifies the impact of the closest shaly barrier to the injection well on the recovery factor during SAGD process, is introduced.
- Empirical Arps decline parameters are tested successfully for parameterization of cumulative production profile.
- To reduce dimensionality of input data, PCA is performed. Results of the case study suggest that prediction quality is improved, while over-fitting is limited, when using a reduced set of principal components as input attributes.
- Schemes for incorporating various deterministic and fuzzy-based clustering techniques with ANN modeling are presented. It is shown that robustness and accuracy of the prediction capability are greatly enhanced

when cluster analysis are performed to identify internal data structures and groupings prior to ANN modeling.

- In particular, FCM could improve neural network predictability when input vectors are located along cluster boundaries. A membership weighting scheme is proposed to defuzzify the model outputs in order to obtain a unique prediction.
- In terms of academic contribution, this study investigated the feasibility of integrating a number of data-based modeling approaches with artificial neural networks (ANN) for SAGD recovery performance prediction in heterogeneous reservoirs.
- In terms of industrial contribution, modeling approaches presented in this work can be applied to analyze the vast amounts of complex field data (and its uncertainty) available to recovery process evaluation. The periodic updating procedure discussed here can be used to integrate continuous dynamic data in a convenient fashion.
- The approaches presented in this work can be integrated directly into most existing reservoir management routines. They represent a significant potential of assisting real-time decision-making for field activities, development planning, and uncertainty quantification.

6.2 Recommendations for Future Work

For further data-driven proxy modeling studies during enhanced oil recovery (EOR) processes, the following future research is recommended:

- The idealization of pertinent input parameters for ANN model during SAGD process requires further investigation. Selection of more representative input parameters associated with the desired output parameter would improve the ANN network predictability. For example, anisotropic permeability tensor (instead of scalar isotropic permeability values) and variogram parameters can be used as input attributes. For instance, permeability can be used as a tensor instead of a single value for each layer or even each grid block within reservoir model.
- Other types of neural networks such as: probabilistic neural networks can be used to solve the classification problem associated with the variability of data set.
- Proper application of other artificial intelligence methods such as genetic algorithm and fuzzy logic in a hybrid manner alongside with neural networks could be used for practical applications. To hybridize the solution, each problem has to be divided into some sub-problems and any of these sub-problems have to be resolved by one of the artificial intelligence techniques. Optimization aspects can be solved by genetic algorithm, handling of linguistic and qualitative term or imprecision in model boundaries can be handled by fuzzy logic and prediction or approximation of non-linear relation between inputs and desired outputs can be addressed by artificial neural network.
- Given that quantitative ranking of operating areas, robust forecasting, and optimization of heavy oil recovery processes are major challenges faced

by the industry, the proposed research highlights the significant potential of applying effective data-driven modeling approaches in analyzing other solvent-additive steam injection projects.

References

Ahmadloo, F., Asghari, K. & Renouf, G., 2010. *A new diagnostic tool for performance evaluation of heavy oil waterfloods: case study of western Canadian heavy oil reservoirs.* Anaheim, CA, USA, SPE Western Regional Meeting, May 27-29.

Al-Anazi, A., Gates, I. & Azaiez, J., 2009. *Innovative Data-Driven Permeability Prediction in a Heterogeneous.* Amsterdam, The Netherlands, SPE EUROPEC/EAGE Annual Conference and Exhibition, June 8-11.

Al-Bulushi, N. I., King, P. R., Blunt, M. J., & Kraaijveld, K., 2012. Artificial neural networks workflow and its application in the petroleum industry. *Neural Computing & Applications,* 21(3), pp. 409-421.

Al-Gosayir, M., Leung, J., & Babadagli, T., 2012. Design of solvent-assisted SAGD processes in heterogeneous reservoirs using hybrid optimization techniques. *Journal of Canadian Petroleum Technology,* 51(6), pp. 437-448.

Alimonti, C. & Falcone, G., 2004. Integration of Multiphase Flowmetering, Neural Networks, and Fuzzy Logic in Field, Performance Monitoring. *SPE Production & Facilities,* 19(1), pp. 25-32.

Aminian, K., Ameri, S., Oyerokun, A. & Thomas, B., 2003. *Prediction of flow units and permeability using artificial neural networks.* Long Beach, CA, USA., SPE Western Regional/AAPG Pacific Section Joint Meeting, May 19-24.

Aminzadeh, F. & Brouwer, F., 2006. *Integrating Neural Networks and Fuzzy Logic for Improved Reservoir Property Prediction and Prospect Ranking.* New Orleans, Society of Exploration Geophysicists (SEG), October 1 - 6.

Amirian, E., Leung, J. Y., Zanon, S. & Dzurman, P., 2013. *Data-Driven Modeling Approach for Recovery Performance Prediction in SAGD Operations.* Calgary, Canada, SPE Heavy Oil Conference, June 11-13.

Arps, J. J., 1945. Analysis of Decline Curves. *Transactions of the American Institute of Mining, Metallurgical and Petroleum Engineers,* 160, pp. 228-247.

Artun, E., Ertekin, T., Watson, R. & Miller, B., 2012. Designing cyclic pressure pulsing in naturally fractured reservoirs using an inverse looking recurrent neural network. *Computational Geosciences,* 38(1), pp. 68-79.

Attia, M., Mahmoud, M. A., Abdulraheem, A. & Al-Neaim, S. A., 2013. *Evaluation of the Pressure Drop due to Multi Phase Flow in Horizontal Pipes using Fuzzy Logic and Neural Networks.* Manama, Bahrain, SPE Middle East Oil and Gas Show and Conference, March 10-13.

Awoleke, O. O. & Lane, R. H., 2011. Analysis of Data From the Barnett Shale Using Conventional Statistical and Virtual Intelligence Techniques. *SPE Reservoir Engineering & Evaluation,* 14(5), pp. 544-556.

Bezdek, J. C., 1981. *Pattern Recognition with Fuzzy Objective Function Algoritms.* New York: Plenum Press.

Bhattacharya, S. & Nikolaou, M., 2013. Analysis of Production History for Unconventional Gas Reservoirs With Statistical Methods. *SPE Journal*, 18(5), pp. 878-896.

Bravo, C., Saputelli, L., Rivas, F., Pérez, A. G., Nikolaou, M., & Zangl, G., 2012. *State-of-the-art application of artificial intelligence and trends in the E&P industry: A technology survey*. Utrecht, The Netherlands: SPE Intelligent Energy International, March 27-29.

Cai, W., Chen, S. & Zhang, D., 2007. Fast and Robust Fuzzy C-Means Clustering Algorithms Incorporating Local Information for Image Segmentation. *Elsevier: Pattern Recognition,* 40(3), pp. 825-838.

Canuto, A., 2001. Ph.D. Thesis: *Combining Neural Networks and Fuzzy Logic for Applications in Character Recognition, Canterbury*: University of Kent.

Charles, R. (2008). Rubencharles, *Retrieved from http://www.rubencharles.com*

Chen, Y. & Durlofsky, . L. J., 2008. Ensemble-Level Upscaling for Efficient Estimation of Fine-Scale Production Statistics. *SPE Journal*, 13(4), pp. 400-411.

Chung, T. H., Carroll, H. B. & Lindsey, R., 1995. *Application of Fuzzy Expert Systems for EOR Project Risk Analysis.* Dallas, USA, SPE Annual Technical Conference & Exhibition, October 22-25.

Computer Modeling Group (CMG) Ltd., 2009. *STARS User's Manual, Version 2009.10*. Calgary, Alberta, Canada.

Demuth, H., & Beale, M., 1998. *Neural Network Toolbox User's Guide 3rd ed.*, Natick, MA, USA, The MathWorks, Inc.

Dunn, J. C., 1973. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics,* 3, pp. 32-57.

Dykstra, H., & Parsons, R. L., 1950. *The Prediction of Oil recovery by Waterflooding, Secondary Recovery of Oil in the United States* (2nd ed.). New York, USA, API.

El-Sebakhy, E. A., Asparouhov, O., Abdulraheem, A., Al-Majed, A., Wu, D., & Latinski, K., 2012. Functional networks as a new data mining predictive paradigm to predict permeability in a carbonate reservoir. *Expert Systems with Applications - Elsevier*, 39(12), pp. 10359-10375.

Everitt, B. S., Landau, S., Leese, M. & Stahl, D., 2011. Cluster Analysis. 5th ed. King's College London, UK: John Wiley & Sons, Ltd.

Ferreira, I., Gammiero, A., & Llamedo, M., 2012. *Design of a neural network model for predicting well performance after water shutoff treatments using polymer gels*. Mexico City, Mexico: SPE Latin America and Caribbean Petroleum Engineering Conference, April 16-18.

Fetkovich, M. J., Fetkovich, E. J. & Fetkovich, M. D., 1996. Useful Concepts for Decline Curve Forecasting, Reserve Estimation, and Analysis. *SPE Reservoir Engineering,* 11(1), pp. 13-22.

Francis, L., 2001. *The Basics of Neural Networks Demystified. Contingencies,* 11(12), pp. 56-61.

Garrouch, A. A. & Labbabidi, H. M., 2001. Development of an expert system for underbalanced drilling using fuzzy logic. *Elsevier: Journal of Petroleum Science and Engineering,* 31, pp. 23-39.

Gilbert, R. J., Liu, Y., William, A. & Preece, R., 2004. Reservoir Modeling: Integrating Various Data at Appropriate Scales. *The Leading Edge,* 23(8), pp. 784-788.

Hambalek, N. & Gonzalez, R., 2003. *Fuzzy Logic Applied to Lithofacies and Permeability Forecasting, Case Study: Sandstone of Naricual, El Furrial Field, Eastern Venezuela Basin.* Port-of-Spain, Trinidad, West Indies, SPE Latin American and Caribbean Petroleum, April 27-30.

Haykin, S., 2005. *Neural Networks, a Comprehensive Foundation.* 2nd ed. Delhi, India: Prentice-Hall of India.

Holdaway, K. R., 2012. *Oilfield data mining workflows for robust reservoir characterization.* Utrecht, The Netherlands, SPE Intelligent Energy International, March 27-29.

Karambeigi, M. S., Zabihi, R. & Hekmat, Z., 2011. Neuro-simulation modeling of chemical flooding. *Journal of Petroleum Science and Engineering,* 78(2), pp. 208-219.

Kaufman, L. & Rousseeuw, P. J., 1990. *Finding Groups in Data: An Introduction to Cluster Analysis.* Hoboken, NJ: John Wiley & Sons, Inc..

Khan, M. S., & Coulibaly, P., 2006. Bayesian neural network for rainfall-runoff modeling. *Water Resources Research*, 42, W07409.

Lechner, J. P. & Zangl, G., 2005. *Treating uncertainties in reservoir performance prediction with neural networks.* Madrid, Spain, SPE Europec/EAGE Annual Conference, June 13-16.

Lim, J. S., 2005. Reservoir properties determination using fuzzy logic and neural networks from well data in offshore Korea. *Elsevier: Journal of Petroleum Science and Engineering,* Volume 49, pp. 182-192.

Lin, C. T. & Lee, G., 1996. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Upper Saddle River, NJ: Prentice-Hall.

Lohninger, H., 1993. Evaluation of neural networks based on radial basis functions and their applications to the prediction of boiling points from structural parameters. *Journal of Chemical Information and Computer Science*, 33(5), pp. 736-744.

Luis, F., Ayala, H. & Ertekin, T., 2007. Neuro-simulation analysis of pressure maintenance operations in gas condensate reservoirs. *Journal of Petroleum Science and Engineering,* 58(1-2), pp. 207-226.

MacQueen, J. B., 1967. Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability,* 1, pp. 281-297.

McCulloch, W. S. & W. Pitts., 1943. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, pp. 115-133.

Malallah, A. & Sami Nashawi, I., 2005. Estimating the fracture gradient coefficient using neural networks for a field in the Middle East. *Journal of Petroleum Science and Engineering*, 49(3-4), pp. 193-211.

Mehrotra, K., Mohan, C. K. & Ranka, S., 1997. *Elements of Artificial Neural Networks.* Cambridge, MA: A Bradford Book, MIT Press.

Mohaghegh, S., 2000. Virtual-Intelligence Applications in Petroleum Engineering: Part 3—Fuzzy Logic. *Journal of Petroleum Technology*, 52(11), pp. 82-87.

Mohaghegh, S., 2002. Virtual-Intelligence Applications in Petroleum Engineering: Part1 Artificial Neural Networks. *Journal of Petroleum Technology*, 52(9), pp. 64-73.

Mohaghegh, S., Reeves, S. & Hill, D., 2000. *Development of an Intelligent Systems Approach for Restimulation Candidate Selection.* Calgary, Alberta, SPE/CERI Gas Technology Symposium, April 3-5.

Mohammadpoor, M., Qazvini Firouz, A. & Torabi, F., 2012. *Implementing simulation and artificial intelligence tools to optimize performance of the CO2 sequestration in coalbed methane reservoirs.* Orlando, FL, USA, Carbon management technology conference, February 7-9.

Murillo, A., Neuman, J. & Samuel, R., 2009. *Pipe Sticking Prediction and Avoidance Using Adaptive Fuzzy Logic and Neural Network Modeling.* Oklahoma City, Oklahoma, USA, SPE Production and Operations Symposium, April 4-8.

Nash, J. E. & Sutcliffe, J. V., 1970. River flow forecasting through conceptual models part I - A discussion of principles. *Journal of Hydrology,* 10(3), pp. 282-290.

Nikravesh, M., Dobie, C. A. & Patzek, T. W., 1997. *Field-Wise Waterflood Management in Low Permeability, Fractured Oil Reservoir: Neuro-Fuzzy Approach.* Bakersfield, California, SPE International Thermal Operations and Heavy Oil Symposium, February 10-12.

Oberwinkler, C., Ruthammer, G., Zangl, G. & Economides, M. J., 2004. *New tools for fracture design optimization.* Lafayette, LA , USA, SPE International Symposium and Exhibition on Formation Damage control, February 18-20.

Parada, C. H. & Ertekin, T., 2012. *A new screening tool for improved oil recovery methods using artificial neural networks.* Bakersfield, CA, USA, SPE Western Regional Meeting, March 21-12.

Popa, A., Cassidy, S. & Mercer, M., 2011. *A data mining approach to unlock potential from an old heavy oil field.* Anchorage, Alaska, USA, SPE Western North American Regional Meeting, May 7-11.

Popa, A. S., 2013. *Identification of Horizontal Well Placement Using Fuzzy Logic.* New Orleans, Louisiana, USA, SPE Annual Technical Conference and Exhibition, 30 September–2 October.

Popa, A. S. & Cassidy, S., 2012. *Artificial Intelligence for Heavy Oil Assets: The Evolution of Solutions and Organization Capability.* San Antonio, Texas, USA, SPE Annual Technical Conference and Exhibition, October 8-10.

Popa, A. S. & Patel, A., 2012. *Neural networks for production curve pattern recognition applied to cyclic steam optimization in diatomite reservoirs.* Bakersfield, CA, USA, SPE Western Regional Meeting, March 21-23.

Poulton, M. M., 2002. Neural networks as an intelligence amplification tool: A review of applications. *Geophysics,* 67(3), pp. 979-993.

Poulton, M. M., 2001. *Computational neural networks for geophysical data processing. Oxford.* Oxford: Elsevier.

Queipo, N. V., Goicochea, J. V. & Pintos, S., 2002. Surrogate modeling-based optimization of SAGD Processes. *Journal of Petroleum Science and Engineering,* 35(1-2), pp. 83-93.

Raeesi, M. M., Moradzadeh, A., Ardejani, F. D. & Rahimi, M., 2012. Classification and identification of hydrocarbon reservoir lithofacies and their heterogeneity using seismic attributes, logs data and artificial neural networks. *Journal of Petroleum Science and engineering,* 82-83, pp. 151-165.

Ramagulam, A., Ertekin, T. & Flemings, P. B., 2007. *Utilization of Artificial Neural Networks in the Optimization of History Matching.* Buenos Aires, Argentina: Latin American & Caribbean Petroleum Engineering Conference, April 15-18.

Rousseeuw, P. J., 1987. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Journal of Computational and Applied Mathematics,* 20, pp. 53-65.

Rumelhart, D. E., Hinton, G. E., & McClelland, J. L., 1986. A general framework for parallel distributed processing. *Parallel Distributed Processing: Explorations in the Macrostructure of Cognition*, 1, pp. 45-76. Cambridge, MA: MIT Press.

Saemi, M., Ahmadi, M., & Varjani, A. Y., 2007. Design of neural networks using genetic algorithm for the permeability estimation of the reservoir. *Journal of Petroleum Science and Engineering*, 59(1-2), pp. 97–105.

Saputelli, L., Malki, H., Canelon, J. & Nikolaou, M., 2002. *A critical overview of artificial neural network applications in the context of continuous oil field optimization.* San Antonio, TX. USA, SPE Annual Technical Conference and Exhibition, 29 September-2 October.

Scheevel, J. R. & Payrazyan, K., 2001. Principal Component Analysis Applied to 3D Seismic Data for Reservoir Property Estimation. *SPE Reservoir Evaluation & Engineering*, 4(1), pp. 64-72.

Scheidt, C. & Caers, J., 2009. Uncertainty Quantification in Reservoir Performance Using Distances and Kernel Methods-Application to a West Africa Deepwater Turbidite Reservoir. *SPE Journal*, 14(4), pp. 680-692.

Sharma, A., 2008. *Classification of Hydrocarbon Recovery Factor Based on Reservoir Databases*: The University of Texas at Austin.

Smith, L. I., 2002. *A tutorial on Principal Components Analysis,* Dunedin, New Zeland: John Wiley & Sons Inc.

Stoisits, R. F. et al., 1999. *Production optimization at the Kuparuk River field* *utilizing neural networks and genetic algorithms.* Oklahoma City, OK, USA: SPE Mid-Continent Operations Symposium, March 9-11.

Strebelle, S., Payrazyan, K. & Caers, J., 2003. Modeling of a Deepwater Turbidite Reservoir Conditional to Seismic Data Using Principal Component Analysis and Multiple-Point Geostatistics. *SPE Journal*, 8(3), pp. 227-235.

Stundner, M., 2001. *How data-driven modeling methods like neural networks can help to integrate different types of data into reservoir management.* Bahrain, SPE Middle East Oil Show, March 17-20.

Suh, S. C., 2012. *Practical applications of data mining,*: Jones & Bartlett Learning, USA.

Takagi, H. & Hayashi, I., 1991. NN-driven fuzzy reasoning. *International Journal of Approximate Reasoning (Special Issues of IIZUKA'88),* 5(3), pp. 191-212.

Tang, H., Toomey, N. & Meddaugh, W. S., 2011. Using an artificial-neural-network method to predict carbonate well log facies successfully. *SPE Reservoir Evaluation Engineering,* 14(1), pp. 35-44.

Wang, H., Echeverría-Ciaurri, D., Durlofsky, L. J. & Cominelli, A., 2012. Optimal Well Placement Under Uncertainty Using a Retrospective Optimization Framework. *SPE Journal*, 17(1), pp. 112-121.

Werbos, P. J., 1994. *The Roots of Backpropagation. From Ordered Derivatives to Neural Networks and Political Forecasting*. New York, NY: John

Wiley & Sons, Inc.

Yager, R. R., 1992. Expert Systems Using Fuzzy Sets. In: *An Introduction to Fuzzy Logic Applications in Intelligent Systems.* Norwell, MA: Kluwer, pp. 27-44.

Yeten, B. & Durlofsky, L. J., 2003. Optimization of nonconventional well type, location, and trajectory. *SPE Journal,* 8(3), pp. 200-210.

Zadeh, L. A., 1965. Fuzzy Sets. *Information and Control,* 8(3), pp. 338-353.

Zangl, G., Graf, T. & Al-Kinani, A., 2006. *Proxy modeling in production optimization.* Vienna, Austria, SPE Europec/EAGE Annual Conference and Exhibition, June 12-15.

Zerafat, M. M., Ayatollahi, S., Mehranbod, N. & Barzegari, D., 2011. *Bayesian network analysis as a tool for efficient EOR screening.* Kuala Lumpur, Malaysia, SPE Enhanced Oil Recovery Conference, July 19-21.

Zhanggui, L., Di, J., Jun, F. & Kaihong, Z., 1998. *Integration of Fuzzy Methods into Geostatistics for Petrophysical Property Distribution Simulation of a Reservoir at Exploration Stage.* Perth, Australia, SPE Asia Pacific Oil & Gas Conference, October 12-14.

Zhou, C. D., Wu, X. L. & Cheng, J. A., 1993. *Determining Reservoir Properties in Reservoir Studies Using a Fuzzy Neural Network.* Houston, Texas, SPE Annual Technical Conference and Exhibition, October 3-6.

Appendixes

Appendix A: Code for BPNN Training

```
%----------------------BPNN Training Algorithm-------------------
clear; clc;
close all force;
%------------------------Import Training Data--------------------
X=importdata('input.txt');
T=importdata('ouput.txt');
[e,f]=size(X);
[ee,ff]=size(T);
%--------------------Normalizing X(inputs) and T(Outputs)--------
for i=1:f;
    X(:,i)=(X(:,i)-min(X(:,i)))/(max(X(:,i))-min(X(:,i)));
end
for i=1:ff;
    T(:,i)=(T(:,i)-min(T(:,i)))/(max(T(:,i))-min(T(:,i)));
end
eta=0.5; % Velocity term
%-----------------------Assigning Network Architecture-----------
nh=input('Enter the Number of Hidden Layers= ');
for i=1:nh
    nnodes(i)=input('Enter the Number of Nodes in Hidden Layers=
');
end
nnodes=horzcat(e,nnodes,ee); % Matrix including no. of nodes in
each layer
DoA=max(nnodes);
%------------------Randomly Initialize Weights 3D-Matrix, A------
for m=1:nh+1
    A(:,:,m)= randi([-10 10],DoA,DoA)/10;
    for j=1:DoA
        for i=1:DoA
        if A(j,i,m)==0  %Make the weights to be a nonzero value--
            A(j,i,m)=0.15;
        end
        end
    end
end
for ii=2:nh+2
    for i=1:nnodes(ii-1)
        for j=nnodes(ii)+1:DoA
            A(i,j,ii-1)=0;
        end
    end
end
for ii=2:nh+2
    for i=nnodes(ii-1)+1:DoA
        for j=1:DoA
            A(i,j,ii-1)=0;
        end
    end
end
```

```
%------------------Randomly Initialize Threshold 2D-Matrix, B-----
C=zeros(DoA);
for i=1:nh+1
    threshold=rand([1 -1],DoA)';
    C=horzcat(threshold,C);
end
for j=1:nh+1
    for i=nnodes(j+1)+1:DoA
        C(i,j)=0;
    end
end
for i=1:DoA
    for j=1:nh+1
        B(i,j)=C(i,j);
    end
end
%-----------------------------Training Cycles---------------------
H=zeros(DoA,nh+2);
net=zeros(DoA);
deltaW=zeros(DoA,DoA,nh+1);
deltaThres=zeros(DoA,nh+1);
for run_count=1:2000  % No. of training epoches
    dY_out(run_count)=0;
    for l=1:f
        net=zeros(DoA);
        for rr=1:e
            H(rr,1,l)=X(rr,l);
        end

        for m=1:nh+1
            for i=1:nnodes(m+1)
                for j=1:nnodes(m)
                    net(m,i)=net(m,i)+A(j,i,m)*H(j,m,l);
                end
                net(m,i)=net(m,i)-B(i,m);
                H(i,m+1,l)=1/(1+exp(-net(m,i)));
            end
        end

        Y_out(1:nnodes(nh+2),l)=H(1:nnodes(nh+2),nh+2,l);% Output

        for icc=1:nnodes(nh+2)
            dY(icc,l)=H(icc,nh+2,l)*(1-H(icc,nh+2,l))*(T(icc,l)-
H(icc,nh+2,l));
dY_out(run_count,l)=dY_out(run_count,l)+abs(H(icc,nh+2,l)-
T(icc,l)); %The total mismatch in each training epoch
        end
%-----------dh Matrix (includes the deltaj and deltah)-----------
        %dh= differences in the net values of nodes in each layer
        dh=zeros(DoA,nh+1);
        for i=1:nnodes(nh+2)
            dh(i,1)=dY(i,l);
        end
        for m=nh+2:-1:3
            for i=1:nnodes(m-1)
                for j=1:nnodes(m)
                dh(i,nh-m+4)=dh(i,nh-m+4)+dh(j,nh-m+3)*A(i,j,m-
1);
                end
                dh(i,nh-m+4)=H(i,m-1,l)*(1-H(i,m-1,l))*dh(i,nh-
m+4);
            end
        end
%--------Adjustment Value of the Weights and Treshholds----------
        for i=1:nnodes(nh+2)
            for j=1:nnodes(nh+1)
                deltaW(j,i,nh+1)=eta*dh(i,1)*H(j,nh+1);
                deltaThres(i,nh+1)=-eta*dh(i,1);
            end
        end
        for m=1:nh
            for i=1:nnodes(m)
                for j=1:nnodes(m+1)
                    deltaW(i,j,m)=eta*dh(j,nh+2-m)*H(i,m,l);
                    deltaThres(j,m)=-eta*dh(j,nh+2-m);
                end
            end
        end
%--------------Updating Weight & Treshhold Matrixes, A & B-------
        for m=1:nh+1
            for i=1:nnodes(m)
                for j=1:nnodes(m+1)
                    A(i,j,m)=A(i,j,m)+deltaW(i,j,m);
                    B(j,m)=B(j,m)+deltaThres(j,m);
                end
            end
        end
    end
end
%%
%------------------------------Plotting options-------------------
----------
l=1:1:size(T');
figure(1);
plot(l,Y_out(1,:),'ko','MarkerFaceColor','k','MarkerSize',6);hold
on;plot(l,T(1,:),'r^','MarkerFaceColor','r','MarkerSize',6);grid
on;set(gca, 'GridLineStyle', '-');grid(gca,'minor')
xlabel('Realization No. for Training Data Set','FontName','Times
New Roman','FontSize', 12');ylabel('Di(Decline Rate, t^(-
1))','FontName','Times New Roman','FontSize', 12);hleg =
legend('ANN Model Predicted Di','Flow simulator
Di','FontName','Times New Roman','FontSize',
30);set(hleg,'FontName','Times New Roman','FontSize', 12);
figure(2);
plot(l,Y_out(2,:),'ko','MarkerFaceColor','k','MarkerSize',6);hold
on;plot(l,T(2,:),'r^','MarkerFaceColor','r','MarkerSize',6);grid
on;set(gca, 'GridLineStyle', '-');grid(gca,'minor')
xlabel('Realization No. for Training Data Set','FontName','Times
New Roman','FontSize',
12');ylabel('qi(m^3/day)','FontName','Times New
Roman','FontSize', 12);hleg = legend('ANN Model Predicted
qi','Flow simulator qi','FontName','Times New Roman','FontSize',
30);set(hleg,'FontName','Times New Roman','FontSize', 12);
```

```
figure(3);
plot(Y_out(1,:),T(1,:),'r.','MarkerSize',9');xlim([0 1]),ylim([0
1]);xlabel('ANN Model Predicted Di','FontName','Times New
Roman','FontSize', 12');ylabel('CMG STARS-CF Di
)','FontName','Times New Roman','FontSize', 12');
hold on;
xx=1:1:100;yy=xx;plot(xx,yy); grid on;
hold off;
figure(4);
plot(Y_out(2,:),T(2,:),'r.','MarkerSize',9');xlim([0 1]),ylim([0
1]);xlabel('ANN Model Predicted qi','FontName','Times New
Roman','FontSize', 12');ylabel('CMG STARS-CF qi
)','FontName','Times New Roman','FontSize', 12');
hold on;
xx=1:1:100;yy=xx;plot(xx,yy); grid on;
hold off;
figure(3);plot(dY_out,'k', 'linewidth',2);xlabel('No. of
Epoch','FontName','Times New Roman','FontSize',
12');ylabel('Error','FontName','Times New Roman','FontSize',
12');
```

Appendix B: Code for Validation or Testing

```
%-------------------------Validation or Testing -----------------
clear; clc;
close all force;
%------------------------Import Testing Data---------------------
X_test=importdata('input_test.txt');
T_test=importdata('ouput_test.txt');
[e,f]=size(X_test);
[ee,ff]=size(T_test);
%-----------Normalizing X_test(input_test) and
T_test(output_test)---------
for i=1:f
    X_test(i,:)=(X_test(i,:)-min(X_test(i,:)))/(max(X_test(i,:))-
min(X_test(i,:)));
end
for i=1:ff;
    T_test(:,i)=(T_test(:,i)-min(T_test(:,i)))/(max(T_test(:,i))-
min(T_test(:,i)));
end
for l=1:f
        net=zeros(DoA);
        for rr=1:e
            H_test(rr,1,l)=X_test(rr,l);
        end

        for m=1:nh+1
            for i=1:nnodes(m+1)
                for j=1:nnodes(m)
                    net(m,i)=net(m,i)+A(j,i,m)*H_test(j,m,l);
                end
                net(m,i)=net(m,i)-B(i,m);
                H_test(i,m+1,l)=1/(1+exp(-net(m,i)));
            end
        end


Y_out_test(1:nnodes(nh+2),l)=H_test(1:nnodes(nh+2),nh+2,l);%
Predicted outputs
end
%-----------------------------Plotting options-------------------
l=1:1:size(T_test');
figure(4);
plot(l,Y_out_test,'ko','MarkerFaceColor','k','MarkerSize',6);hold
on;plot(l,T_test,'r^','MarkerFaceColor','r','MarkerSize',6);grid
on;set(gca, 'GridLineStyle', '-');grid(gca,'minor');
```

Appendix C: Code for Principal Component Analysis (PCA)

```
%------------------------------PCA-------------------------------
clear; clc;
close all force;
%------------------------------Import Data-----------------------
data=importdata('all.txt');
data1=data';
[m,n]=size(data1);
for i=1:m
    for j=1:n
        data_zero_mean(i,j)=data1(i,j)-mean(data1(m,:));
    end
end
[V,F,U]=svd(data_zero_mean);
singv = svd(data_zero_mean);% singv is a vector containg the
singular values
D = sort(singv,'descend');% arrange the singular values in
descending order
% sort the columns of P to match the sorted columns of D (going
from largest to smallest)
S=input('Enter the number of components to be considered=  ');
V1=V(:,1:S);
scores = zeros(S,n);
for i=1:n
    scores(:,i)=V1\data_zero_mean(:,i);
end
all_scores=scores';
save('all_scores','all_scores');
%------Silhouette plot and optimum number of k-means clusters----
NC=input('Enter the number of clusters to be considered= ');
[cidx,ccntr] = kmeans(all_scores,NC,'distance','sqeuclid');
[s,h] = silhouette(all_scores,cidx,'sqeuclid');
Silh_Avg=(sum(s(:,1)))/150;
```

Appendix D: Code for K-Means Clustering

```
%-----------------------K-Means Clustering-----------------------
clear; clc;
close all force;
%--------------------------Import Data---------------------------
load('all_scores');
all_rf=importdata('all_rf.txt');
k=4; %No. of clusters
[IDX,C] = kmeans(all_scores,k);
%--Adding Cluster Numbers,Scores Matrix and rf for Each Data
Point-----
all_scores_rf_IDX=horzcat(all_scores,all_rf,IDX);
[e,f]=size(all_scores_rf_IDX);
A=zeros(100,f,k);j=1;
for iii=1:k
    for i=1:e
        if all_scores_rf_IDX(i,f)==iii
           A(j,:,iii)=all_scores_rf_IDX(i,:);
           j=j+1;
        end
    end
end
```

Appendix E: Code for Fuzzy C-Means Clustering

```
%-----------------------Fuzzy C-Means Clustering-----------------
clear; clc;
close all force;
%------------------------------Import Data-----------------------
load('all_scores');
Data=all_scores;
[m,n]=size(Data);
for i=1:n
    for j=1:m
        X(j,i)=(Data(j,i)-min(Data(1:m,i)))/(max(Data(1:m,i))-
min(Data(1:m,i)));
    end
end
[center,U,obj_fcn] = fcm(Data, 3);
maxU = max(U);
index1 = find(U(1,:) == maxU);
index2 = find(U(2, :) == maxU);
index3 = find(U(3, :) == maxU);
%%
%------------------------Group Each Cluster Data-----------------
Cluster1=zeros(length(index1),n+1);
for i=1:length(index1)
    Cluster1(i,1:n)=Data(index1(1,i),1:n);
end
Cluster2=zeros(length(index2),n+1);
for i=1:length(index2)
    Cluster2(i,1:n)=Data(index2(1,i),1:n);
end
```

```
Cluster3=zeros(length(index3),n+1);
for i=1:length(index3)
    Cluster3(i,1:n)=Data(index3(1,i),1:n);
end
```