

# **Artificial Intelligence Techniques in Reservoir Characterization**

By

**ADENIRAN, AHMED ADEBOWALE**

A Thesis Presented to the  
DEANSHIP OF GRADUATE STUDIES  
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the  
Requirements for the Degree of

**MASTER OF SCIENCE**

In

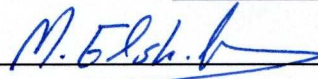
Systems Engineering

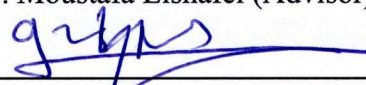
January 2009

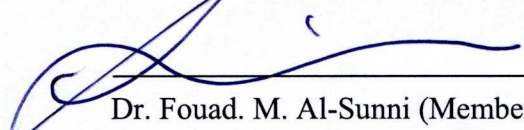
**KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS  
DHAHRAN 31261, SAUDI ARABIA  
DEANSHIP OF GRADUATE STUDIES**

This thesis, written by Adeniran Ahmed Adebowale under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN SYSTEMS ENGINEERING.


Thesis Committee

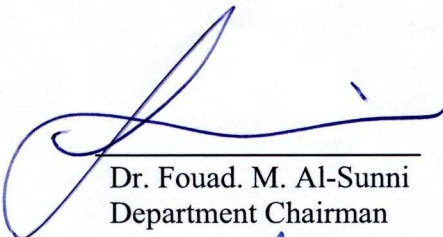
  
\_\_\_\_\_  
Dr. Moustafa Elshafei (Advisor)

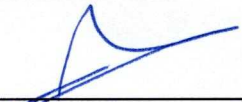
  
\_\_\_\_\_  
Dr. Gharib Hamada (Co-Advisor)

  
\_\_\_\_\_  
Dr. Fouad. M. Al-Sunni (Member)

  
\_\_\_\_\_  
Dr. Amay Khouki (Member)

  
\_\_\_\_\_  
Dr. Samir H. Al-Amer (Member)

  
\_\_\_\_\_  
Dr. Fouad. M. Al-Sunni  
Department Chairman

  
\_\_\_\_\_  
Dr. Salaam Zummo  
Dean of Graduate Studies

19/4/09  
\_\_\_\_\_  
Date



*Dedicated*

*To my Loving Parents*

## **ACKNOWLEDGEMENT**

All gratitude belongs to Almighty Allah the lord of the universe, Who teaches man what he knows not. I thank Allah for having spared my life over the years and for bestowing on me His uncompromising blessings.

First, I like to acknowledge the opportunity given to me by King Fahd University of Petroleum and Minerals and especially Department of Systems Engineering to pursue my master degree in a harmonious environment with state-of-the-art facilities.

My immense appreciation and gratitude goes to my thesis advisor Prof. Moustafa Elshafei and co-advisor Prof. Gharib Hamada, for their guidance, encouragement and support during the research. Most important of all were their enthusiastic approach towards research and their readiness and eagerness to help me at any times. Additionally their innovative thinking and valuable suggestions greatly inspired me. I thank them, as they are indeed not only advisors but fathers to reckon with.

My sincere gratitude is due to my thesis committee members: Professor F. Al-Sunni, Dr. Amar Khouki and Dr. S. H. Al-Amer. Their constructive and positive criticism and suggestions were extremely helpful and an immense privilege.

I also want to acknowledge the support and suggestions of my friends and colleagues: Adetunji Ademola, S. O. Olatunji, Alli Alrayyan, Abdul Fatai and many

others, especially the Nigerians who in one way or another made my stay at KFUPM and in Saudi Arabia a memorable one.

Finally, with a deep sense of affection, I offer my sincere thanks to my wife, and the entire Adeniran family, for their encouragement, support and prayers which have kept my spirit alive while away from home.

## TABLE OF CONTENTS

<b>DEDICATION</b> .....	
ACKNOWLEDGEMENT .....	iii
TABLE OF CONTENTS.....	v
LIST OF TABLES .....	ix
LIST OF FIGURES .....	x
THESIS ABSTRACT .....	xiii
 <b>CHAPTER ONE</b>	
INTRODUCTION .....	1
3.1 Overview.....	1
1.1 Problem Statement .....	5
1.2 Thesis Objectives .....	6
1.3 Scope of Thesis .....	7
1.4 Thesis Organization .....	8
 <b>CHAPTER TWO</b>	
LITERATURE SURVEY .....	9
2.1 Overview.....	9
2.2 Reservoir Characterization and Rock Properties .....	9
2.2.1 Porosity.....	10
2.2.2 Water saturation .....	11
2.3 Well log Analysis.....	12

2.4	Approaches to Predicting Porosity and Water Saturation.....	13
2.5	Review of Estimating Porosity and Water saturation Using Neural Networks.....	15
2.6	Statistical Quality Measures .....	19
CHAPTER THREE		
	NEURAL NETWORKS .....	22
3.2	Overview.....	22
3.3	Structure of Neural Networks .....	23
3.4	Multilayer Perceptron Feedforward Neural networks (MLP) .....	26
	3.3.1 MLP Learning .....	27
	3.3.2 Generalization .....	33
3.5	Advantages and Disadvantages of Neural Networks.....	35
3.6	Advances in Neural Networks .....	37
	3.5.1 Cascade Correlation Neural Networks (CCNN) .....	38
	3.5.2. Polynomial Networks (Abductive Networks).....	41
	3.5.3. General Regression Neural Networks .....	45
CHAPTER FOUR		
	FUNCTIONAL NETWORKS.....	48
4.1	Overview.....	48
4.2	Functional Networks Background and Definition .....	50
4.3	Comparison between Functional Networks and Neural Networks.....	52
4.4	Working with Functional Networks.....	55
4.5	Training Functional Networks .....	59

4.6	Model Selection in Functional Networks .....	60
4.7	Estimation of Porosity and Water Saturation With FN .....	63
4.7.1	Problem Statement .....	63
4.7.2	Functional Networks Initial Architecture .....	64
4.7.3	Uniqueness of Representation .....	65
4.7.4	Simplification of the Model .....	66
4.7.5	Learning procedure for the simplified model .....	68
CHAPTER FIVE		
	PREDICTION OF POROSITY AND WATER SATURATION .....	71
5.1	Overview .....	71
5.2	Data acquisition and Implementation process .....	72
5.3	Experimental Results Using Functional Networks .....	75
5.3.1	Porosity Estimation Results .....	75
5.3.2	Water saturation Estimation Results .....	78
5.4	Experimental Results Using CCNN .....	80
5.5	Experimental Results Using GRNN .....	83
5.6	Experimental Results Using Polynet .....	86
5.7	Experimental Results Using Feedforward Neural Networks .....	89
CHAPTER SIX		
	PERFORMANCE ANALYSIS AND COMPARATIVE STUDIES .....	93
6.1	Overview .....	93
6.2	Discussion .....	109



## CHAPTER SEVEN

CONCLUSION AND RECOMMENDATIONS .....	111
7.1 Summary .....	111
7.2 Conclusion .....	112
7.3 Recommendations for Future Work.....	113
REFERENCE.....	115
APPENDIX A: Performance of other Basis Function used for Functional Networks in the Combined Well .....	120
APPENDIX B: Performance of All Models on Well A and B .....	123
APPENDIX C: Result of FN with reduced inputs from the combined well A and B ....	133
VITA.....	136

## LIST OF TABLES

Table 5.1: Statistics of the combined wells A and B .....	73
Table 5.2: Statistics of well A .....	74
Table 5.3: Statistics of well B.....	74
Table 5.4. The RMSE and correlation coefficient (CC) of different basis functions used for porosity estimation.....	76
Table 5.5: The RMSE and CC of different basis functions used for water estimation.....	78
Table 5.4: Models performance for porosity and water saturation using CCNN .....	81
Table 5.5: Models performance for porosity and water saturation using GRNN .....	84
Table 5.6: Models performance for porosity and water saturation using PolyNet .....	87
Table 5.7: Models performance for porosity and water saturation using FFNN .....	90
Table 6.1A: Models performance for porosity .....	94
Table 6.1B: Models performance for water saturation .....	95
Table 6.2: Models performance for porosity and water saturation for well A.....	97
Table 6.3: Models performance for porosity and water saturation for well B.....	98
Table 6.4: FN, GRNN and EM models performance for porosity and water saturation for well A .....	101
Table 6.5: FN, GRNN and EM Models performance for porosity and water saturation for well B .....	101
Table 6.6: Statistics of well T2.....	106
Table 6.7: Models performance for porosity and water saturation on Test well T2 .....	107
Table 6.8: Brief Summary of all models performance for porosity and water saturation.....	110
Table C1: RMSE and correlation coefficient of different basis functions used for porosity (Three predictors: DT, NPHI, RHOB) .....	133

## LIST OF FIGURES

Figure 3.1: Model of a Neuron .....	24
Figure 3.2: Structure of Feedforward Neural Network.....	27
Figure 3.3 Cascade architecture, initial state and after adding two hidden units. The vertical lines sum all incoming activation. Boxed connections are frozen, X connections are trained repeatedly. ....	40
Figure 3.4: Typical polynomial networks.....	43
Figure 3.5: Typical General Regression networks.....	46
Figure 4.1: Example of Functional Network architecture .....	52
Figure 4.2 :(a) The Standard Neural Network (b) its equivalent functional network. ....	54
Figure 4.3: (a) Initial network, (b) Equivalent simplified network.....	57
Figure 5.1: Generalized functional network in equation (4.10) with two inputs .....	65
Figure 5.2: Simplified generalized model with $k=2$ and $q_1=q_2=q$ .....	67
Figure 5.4 Learning algorithm for porosity and water saturation functional networks model .....	70
Figure 5.1a: Performance plot for estimated porosity using FN.....	77
Figure 6.1b: Scatter plot for estimated porosity versus Core porosity using FN.....	77
Figure 5.2a: Performance plot for estimated water saturation using FN .....	79
Figure 5.2b: Scatter plot for estimated water saturation versus water saturation using functional network. ....	80
Figure 5.3a: Performance plot for estimated porosity using CCNN.....	81
Figure 5.3b: Scatter plot for estimated porosity versus Core porosity using CCNN .....	82
Figure 5.4a: Performance plot for estimated water saturation using CCNN .....	82
Figure 5.4b: Scatter plot for estimated water saturation versus water saturation using CCNN.....	83
Figure 5.5a: Performance plot for estimated porosity using GRNN.....	84
Figure 5.5b: Scatter plot for estimated porosity versus Core porosity using GRNN.....	85

Figure 5.6b: Scatter plot for estimated water saturation versus water saturation using GRNN.....	86
Figure 5.7a: Performance plot for estimated porosity using Polynet.....	87
Figure 5.7b: Scatter plot for estimated porosity versus Core porosity using Polynet.....	88
Figure 5.8a: Performance plot for estimated water saturation using ploynet .....	88
Figure 5.8b: Scatter plot for estimated water saturation versus water saturation using Polynet.....	89
Figure 5.9a: Performance plot for estimated porosity using FFNN.....	90
Figure 5.9b: Scatter plot for estimated porosity versus Core porosity using FFNN.....	91
Figure 5.10b: Scatter plot for estimated water saturation versus water saturation using FFNN.....	92
Figure 6.1: Scatter plot of CC versus RMSE for porosity prediction on testing case for all models.....	95
Figure 6.2: Scatter plot of CC versus RMSE for water saturation prediction on testing case for all models	96
Figure 6.3: Scatter plot of CC versus RMSE for porosity prediction on well A for all models .....	98
Figure 6.4: Scatter plot of CC versus RMSE for water saturation prediction on Well A for all models .....	99
Figure 6.5: Scatter plot of CC versus RMSE for Porosity prediction on Well B for all models .....	99
Figure 6.6: Scatter plot of CC versus RMSE for water saturation prediction on Well B for all models .....	100
Figure 6.7: Performance of FN, Calculated porosity and the core porosity for well A .....	102
Figure 6.8: Performance of FN, Calculated porosity and the core porosity for well B .....	102
Figure 6.9: Performance of FN, Calculated water saturation and the core water saturation for well A .....	103
Figure 6.10: Performance of FN, Calculated water saturation and the core water saturation for well B .....	103
Figure 6.11: Performance of GRNN, Calculated porosity and the core porosity for well A .....	104
Figure 6.12: Performance of GRNN, Calculated porosity and the core porosity for well B .....	104
Figure 6.13: Performance of GRNN, Calculated water saturation and the core water saturation for well A .....	105
Figure 6.14: Performance of GRNN, Calculated water saturation and the core water saturation for well B .....	105
Figure 6.15: Performance of FN, Calculated porosity and the core porosity on Test well T2 .....	107

Figure 6.16: Performance of GRNN, Calculated porosity and the core porosity on Test Well T2 .....	108
Figure 6.17: Performance of FN, Calculated water saturation and the core water saturation on Test well T2 .....	108
Figure 6.18: Performance of GRNN, Calculated water saturation and the core water saturation on Test Well T2.....	109
Figure A1: Scatter plot for estimated porosity versus Core porosity using Logarithm function .....	120

## THESIS ABSTRACT

**NAME:** Adeniran Ahmed Adebowale

**TITLE OF STUDY:** Artificial Intelligence Techniques in Reservoir  
Characterization

**MAJOR FIELD:** Systems Engineering

**DATE OF DEGREE:** January, 2009

One of the major objectives of the petroleum industry is to obtain an accurate estimate of initial hydrocarbon in place before investing in development and production. Porosity, permeability and fluid saturation are the key variables for characterizing a reservoir in order to estimate the volume of hydrocarbons and their flow patterns to optimize the production of a field. Many empirical equations are available to transform well log data to predict these properties. Recently, researchers utilized artificial neural networks (ANNs), particularly feed forward back propagation neural networks (FFNN), to develop more accurate predictions. The success of FFNN opens the door to both machine learning and soft-computing techniques to play a major role in the petroleum, oil, and gas industries. Unfortunately, the developed FFNN correlations have some drawbacks, and as a result several improvements have been proposed.

This thesis investigated the suitability of some of the recently proposed advances in neural networks technique including, functional networks (FN), cascaded correlation neural networks, polynomial networks, and general regression neural networks for predicting porosity and water saturation from well logs. Since there is no fully developed software for functional networks, we described both the steps and procedures in developing functional networks to predict these properties. We also compared the performance of these techniques with standard FFNN as well as the empirical correlation models.

Generally, the results show that the performance of General Regression neural networks, Functional networks and Cascaded Correlation networks outperform that of standard neural networks. In addition, General Regression Neural networks are more robust while Functional networks are easier and quicker to train with no over-fitting problem, and more importantly we have more insight into the coefficients of the network. Therefore, we believe that the use of these better techniques will be valuable for Petroleum Engineering scientists.

## ARABIC ABSTRACT

الاسم: احمد عديبوالي ادينيران

عنوان الرسالة: تقنيات الذكاء الصطناعي لتوصيف المكامن

التخصص: هندسة النظم

تاريخ التخرج: يناير 2009

احد أهم الأهداف في مجال انتاج النفط هو الحصول على تقديرات اولية لكميات الهيدروكربون المتوفرة في اي موقع قبل بدء الاستثمار و التطوير و الانتاج في ذلك الموقع.

وتعتبر المسامية والسماحية و تشبع الموائع من أهم الخصائص لوصف و تصنيف المكمن و من ثم تقدير كميات الهيدروكربونات و نمطية انسيابها، و هما عاملان مهمان يستخدمان لايجاد امثل انتاج للحقل.

وهناك الكثير من المعادلات التجريبية التي تستعمل لتحويل قياسات الآبار لتقدير الخصائص الهامة للحقل. ومؤخرا صار الباحثون يستخدمات الشبكات العصبية الصناعية و خصوصا الشبكات العصبية ذات التغذية الامامية و الانتشار الخلفي (FFNN) للحصول على تقديرات اكثر دقة. وأدى النجاح الذي أحرزه استخدام (FFNN) إلى فتح الباب لتقنيات تعليم الآلة و الحوسبة اللينة لتلعب دورا هاما في مجالات انتاج النفط و الغاز. إلا أن النظم التي تستعمل FFNN تعاني من بعض السلبيات، و لهذا تم اقتراح عدد من الطرق التحسينية.

وهذا البحث يدرس مدى مناسبة استعمال بعض التطويرات الحديثة في مجال الشبكات العصبية و التي تشمل شبكات الدوال و الشبكات العصبية التضامنية المتتالية و شبكات التراجع العامة، لتقدير المسامية و تشبع المياه و الانسيابية. ونظرا لعدم وجود برامج مكتملة البناء خاصة لبناء شبكات الدوال فقد قمنا بوصف كامل للخطوات و المنهجية لبناء نظام شبكات الدوال ليستخدم في تقدير العوامل سألقة الذكر.

كما قمنا أيضا بمقارنة أداء هذه الطرق المختلفة مع أداء طريقة FFNN و مع أداء الطرق التجريبية. وبشكل عام فقد أظهرت النتائج المتحصل عليها أن أداء كلا من شبكات التراجع العامة و الشبكات التضامنية المتتالية و شبكات الدوال كانت متفوقة على تلك الخاصة بالشبكات العصبية التقليدية.

إضافة لذلك فقد وجد ان طريقة الشبكات التراجعية العامة أكثر قدرة على التعميم و التنبؤ من مدخلات جديدة. بينما كانت شبكات الدوال أسرع و أسهل في عملية التدريب مع تجنب مشاكل الإفراط في التدريب، و الأكثر أهمية هو أننا نحصل على معرفة أعمق للعلاقات الضمنية من معاملات شبكات الدوال.

و لذا فإننا نعتقد بان استخدام هذه الطرق سيكون ذا قيمة كبيرة في مجال هندسة النفط.



# CHAPTER ONE

## INTRODUCTION

### 3.1 Overview

Reservoir rocks vary from homogeneous to heterogeneous. There are different kinds and levels of heterogeneity: either fluid or lithology. Understanding the form and spatial distribution of these heterogeneities is important in petroleum reservoir characterization. Porosity, permeability and fluid saturation are the key variables for characterizing a reservoir in order to estimate the volume of hydrocarbons and their flow patterns to optimize the production of a field. These characteristics are different for different rocks.

Porosity is described as the ratio of the aggregate volume of interstices in a rock to its total volume, whereas permeability, defined as the capacity of a rock or sediment for fluid transmission, is a measure of the relative ease of fluid flow under pressure gradients. Knowledge of permeability in formation rocks is crucial for estimating the oil production rate and for reservoir flow simulations for enhanced oil recovery. Reliable predictions of porosity and permeability are also crucial for evaluating hydrocarbon accumulations in a basin-scale fluid migration analysis, and for mapping potential pressure seals to reduce drilling hazards.

No well log can directly determine porosity, but well logging measures the physical properties depending on porosity, and then the formation porosity is deduced. Several relationships have been identified which can relate porosity to wireline readings, such as the sonic transit time and density logs. However, the conversion from density and transit time to equivalent porosity values is not trivial (Helle et al. 2001). The common conversion formulas contain terms and factors that depend on the individual location and lithology. For example, conversion from density depends on clay content, pore fluid type, and grain density while conversion from sonic log depends grain transit time. These terms and factors are unknowns and thus remain to be determined from rock sample analysis.

Finding the distribution and composition of subsurface fluids is another main objective in hydrocarbon exploration, field development and production. Since direct sampling of underground fluids and determination of fluid saturation in the laboratory is an expensive and time-consuming procedure, indirect determination from log measurements is the common approach. The common practice in the industry is to determine water saturation from empirical formulas using resistivity, gamma ray logs and porosity estimates. The hydrocarbon saturation is then calculated from the water saturation, as both water and hydrocarbon form the composite pore fluid.

Water saturation is basically defined as the fraction of the pore volume of formation rock which is filled with water, where the rest of the pore space is filled with either oil or gas. Therefore, inaccurate determination of water saturation leads to either

underestimation or overestimation of reserves. The standard practice in the industry for calculating water saturation is by using different models. But these models should be tuned to the area of work, which requires the estimation of parameters in the laboratory. Thus it would be better to calculate water saturation by using different means that will avoid depending on the auxiliary parameters explicitly.

Artificial intelligence (AI) may be defined as a collection of new analytic tools that attempt to imitate life (Mohaghegh, 2000). AI techniques exhibit an ability to learn and deal with new situations. Artificial neural networks, genetic algorithm, particle swarm optimization, fuzzy logic and functional networks are among the paradigms that are classified as artificial intelligence tools. AI has drawn the attention of many researchers over the last two decades (Sandha et al., 2005). The main interest in AI has its roots in the efficient and fast way of computation in complex tasks such as speech and other pattern recognition problems. For about two decades now, AI techniques such as neural networks, genetic algorithm, and fuzzy logic have continued to draw increasing attention from researchers in the petroleum industry to address fundamental problems. These include, for example, the determination of formation permeability, porosity, and water saturation from well logs, or specific problems such as forecasting the post-fracture well performance in the absence of engineering data, which conventional computing has been unable to solve or posed a lot of problems.

Artificial Neural Networks (ANNs) have been applied in a wide variety of fields to solve problems such as classification, feature extraction, diagnosis, function

approximation and optimization (e.g. Haykin, 1999). Although it seems clear that neural networks should not be used where an effective conventional solution exists, there are many tasks for which neural computing can offer a unique solution, in particular those where the data is noisy, where explicit knowledge of the task is not available, or when unknown non-linearity between input and output may exist. These are in many respects features of conventional earth science data, and are the main reasons for the increasing popularity of ANN in geosciences and petroleum engineering (Mohaghegh, 2000; Nikravesh et al., 2001).

Functional network (FN) is another network structure which is also gaining attention in the field of AI. Functional network is a powerful extension and network-based alternative to the neural networks paradigm (Castillo et al., 2000a; Castillo 1998). In functional networks, neural functions are allowed to be not only multivariate but also truly multiargument and different for all neurons. Thus, neural functions are learned instead of weights. In addition, outputs coming from different neurons can be connected, that is, forced to output the same values. The topology and neuron functions of functional networks can be selected according to the data, the domain knowledge, or a combination of the two. Functional equations play an important role in functional networks, since the preceding types of connections lead to functional equations that impose a substantial reduction in the degrees of freedom of the initial neural functions. This kind of network exhibits more versatility than neural networks, and so it can be successfully applied to several problems. Castillo et al. (2000b) shows that every problem that can be solved by a neural network can also be formulated by functional networks. More importantly, some

problems that cannot be solved by using neural networks can be naturally formulated by using functional networks.

In this thesis, we explored recent advances in neural networks, including functional networks, cascaded correlation neural networks, and polynomial neural networks architectures for predicting porosity and water saturation from well logs, and we also carried out a comparative analysis vis-à-vis the standard neural networks.

## **1.1 Problem Statement**

One of the major objectives of the petroleum industry is to obtain an accurate estimate of the hydrocarbon in place. This is required either at an early stage of the well exploration or in the production management of a developed reservoir. An accurate determination of porosity and water saturation parameters is a must for evaluating any reservoir and for drafting any development plan for a reservoir. Neural networks for quantitative analysis of reservoir properties from well logs have been demonstrated in several practical applications (e.g. Huang et al., 1996; Huang and Williamson, 1997; Zhang et al., 2000; Helle et al., 2001), where the artificial neural network approach is shown to be a simple and accurate alternative for converting well logs to common reservoir properties such as porosity and water saturation. Multilayer perceptrons (MLP) trained by a back-propagation algorithm have been the popular tool for most practical applications over the last decade.

MLP trained by back-propagation learning techniques have been shown to be a universal approximator, implying that they will approximate any static function provided that sufficiently representative input-output sample pairs of the function are given. However, one major problem encountered in the back-propagation algorithm is its slow convergence during learning.

Another problem is the structure of the network. The number of hidden units and their interconnections is defined by the programmer, while the learning rule can modify only the connection weights. There is no rule which allows one to determine the necessary structure from a given application or training set. Lastly, the MLP may be caught by local minima, which reduce the network performance. A common approach is to train as many networks as possible and to select the one that yields the best generalization performance. Hence the solution may not be unique especially for noisy data commonly encounter in well logging. Thus, to overcome the potential drawbacks of neural networks, we are interested in designing and investigating new adequate intelligent system techniques which have been proposed either as an alternative or as an improvement to neural networks, and can be utilized in estimation of water saturation and porosity.

## **1.2 Thesis Objectives**

The main objective of this work is to explore the use of recent advances in artificial intelligence techniques, particularly neural networks, to estimate porosity and water saturation from well logs. This study aims to develop better approaches for the

estimation of these reservoir parameters. More specifically, the study aims to achieve the following:

- Investigate and develop functional networks for the estimation of porosity and water saturation.
- Investigate the suitability of estimating porosity and water saturation from well logs by using other recently proposed neural networks such as cascaded correlation neural networks, polynomial networks and general regression neural networks.
- Compare all the techniques along with standard neural networks (MLP).
- Evaluate the performance and accuracy of the proposed methods, their applications, and the limitations in the reservoir characterization from well logs.

### **1.3 Scope of Thesis**

In this thesis we investigate the use of the some proposed improvements to neural networks to predict porosity and water saturation from well logs. Although there has been an exponential increase in the literature on advances in neural networks, we limit this study to some of these proposed advances: functional networks (FN), cascaded correlation neural networks (CCNN), polynomial networks (POLYNET) and general regression neural networks (GRNN).

## **1.4 Thesis Organization**

In Chapter one, we present an introduction to this study, which clearly highlights the motivation behind this work. Chapter two highlights some of the research done in context of estimation of porosity and water saturation with particular focus on the neural networks modeling approach. A brief discussion on empirical model and statistical models is also presented.

Furthermore, as the main technique under which other techniques are based, neural networks are discussed in detail in Chapter three. Some proposed advances to neural networks which are our concern in this work are also discussed. Functional networks are presented in Chapter four as the main recent advances under study. Chapter five presents functional networks design for the prediction of porosity and water saturation. Chapter six focuses on implementation of FN, CCNN, GRNN, POLYNET and GRNN for the prediction of porosity and water saturation. In Chapter seven, the performance analysis and comparative studies are discussed and recommendation are made for further research



# **CHAPTER TWO**

## **LITERATURE SURVEY**

### **2.1 Overview**

In a number of petroleum engineering problems, there is a necessity to forecast and classify the petrophysical properties in series signals. PVT properties, porosity, water saturation, permeability, lithofacies types, seismic pattern recognition are important properties for the oil and gas industry. Predicting these properties in the laboratories is very expensive and the computations are not accurate. Yet, the accuracy of such predictions is critical and not often known in advance. This chapter presents a brief introduction to the reservoir rock properties concerned in this thesis, and it discusses the most common empirical models and neural network approaches that have been used in predicting porosity and water saturation rock properties..

### **2.2 Reservoir Characterization and Rock Properties**

One of the major obstacles that impact reservoir production is that most reservoirs show some degree of heterogeneity. This heterogeneity is known as the non-uniform, non-linear spatial distribution of rock properties. Understanding the form and spatial distribution of this heterogeneity is important in petroleum reservoir evaluation. Detailed information regarding the type and physical properties of rocks and reservoir fluids (such

as porosity and water saturation) is essential for understanding and evaluating the potential performance or productivity of a given petroleum reservoir (Abhijit, 2006). Wireline logs, core analysis, production data, pressure buildups, and tracer tests provide quantitative measurements of the reservoir rock properties in the vicinity of the wellbore. This wellbore data must be integrated with a geological model to display the properties in three-dimensional space.

### 2.2.1 Porosity

Porosity is an important rock property, as it measures the potential storage volume for hydrocarbons. It is described as the ratio of the aggregate volume of interstices in a rock to its total volume. Porosity in carbonate reservoirs ranges from 1 to 35%. The terms effective porosity or connected pore space are commonly used to denote porosity that is most available for fluid flow. Porosity is a scalar quantity, because it is a function of the bulk volume used to define the sample space.

Porosity is determined by visual methods and by laboratory measurements from core samples. This is an expensive exercise, and hence not a routine in all drilled wells. Several relationships have been identified which can relate porosity to wireline readings, such as the sonic transit time and density logs. Equation 2.1 shows an example of a common relationship (Dresser, 1982) used in this study.

$$\phi_c = \frac{R_m - \rho_b}{R_m - R_f * 0.9} \quad (2.1)$$

where  $R_m$  is the matrix density,  $\rho_b$  is the bulk density measurement and  $R_f$  is the resistivity factor.

This common conversion formula contains terms and factors that depend on the individual location and lithology e.g. clay content, pore fluid type, and grain density, which in general are unknowns and thus remain to be determined from rock sample analysis.

### **2.2.2 Water saturation**

Water saturation is defined as the fraction of the pore volume of formation rock which is filled with water. Water, oil, and gas may be found simultaneously in reservoirs. However, due to specific gravity, fluids tend to segregate with the reservoir. When considering a possible production interval, the fraction portion of the pore space which does not contain formation water is assumed to contain hydrocarbon. This can be expressed mathematically by using the relationship:

$$S_h = 1 - S_w \quad (2.2)$$

where  $S_h$  is hydrocarbon saturation and  $S_w$  is the water saturation. The determination of water saturation allows the log analyst to estimate the amount of hydrocarbons present in a given formation. This formation, along with porosity and permeability, can be used to predict the wells' ultimate production.

The standard practice in industry for calculating water saturation is by using different models. But these models should be tuned to the area of work, which requires the estimation of parameters in the laboratory. A popular model derived from the Archie formulae and the Fertl equation is given below (Dresser, 1982).

$$S_w^n = \frac{aR_w}{\phi^m R_t} \quad (2.3)$$

where  $n$  is the saturation exponent, (normally equal to 2)  $a$  is the constant determined empirically,  $m$  is the cementation exponent,  $R_w$  is the resistivity, and  $R_t$  is the resistivity derived from the induction log.

### **2.3 Well log Analysis**

Well logging is the technique of analyzing and recording the character of a formation penetrated by a drill hole in petroleum exploration and exploitation. Well logs are a fundamental method of formation analysis, since they measure the physical properties of the rock matrix and pore fluids. The primary purpose in the analysis of most wells is to describe the lithology of the reservoir rock and the fluid properties for the section cut by that particular well. Generally, logs are available on all wells drilled in a reservoir, and therefore they represent the most complete set of reservoir descriptive data.

Various types of well logs are commonly used for reservoir evaluation. The types of interest in this study include: Sonic log (DT), Neutron log (NPHI), Density log (RHOB), Gamma Ray (GR), Laterlog (Rt), and Photoelectric Factor log (PEF).

## **2.4 Approaches to Predicting Porosity and Water Saturation**

The most accurate method for the determination of these important properties is core analysis. However, the core data are not always available for most wells in a given field or at every depth, either due to the borehole condition or due to the high cost of obtaining cores. Hence, log analyses which are usually available are utilized by correlating well logs with core data of the cored wells, and subsequently by using the correlation model to predict these properties at the uncored intervals and wells. This is possible not only because log analysis can provide a continuous record over the entire well where coring is impossible but also because they are economical and quick to obtain.

Many empirical equations are available to transform well log data to porosity and water saturation. For example, porosity has been related to wireline readings such as the sonic transit time and density logs. Water saturation, on the other hand, can also be determined from empirical formulas using resistivity, gamma ray logs and porosity estimates. Unfortunately, these models are not universally applicable, as they should be tuned to the area of work by estimating parameters in the laboratory. Moreover, some of these required parameters are not easily obtainable (Helle et al., 2001; Jun, 2002).

Parametric methods, such as statistical regression, provide another versatile way to estimate these properties, where a functional relationship is developed between the core data and well log data (Jun, 2002). This however requires the assumption and satisfaction of multi-normal behavior and linearity (Jong-se and Jungwhan, 2004).

Therefore, neural networks, or more specifically, multilayer perceptrons (MLP), have been increasingly applied to predict reservoir properties by using well log data (Jong-se and Jungwhan, 2004; Balan et al., 1995).

Neural networks are characterized as computational models with abilities to adapt, learn, generalize, recognize, and organize data. The advantages of neural networks include their high computational efficiency, adaptability, non-linear characteristics, generation properties, fault tolerance, freedom from a priori selection of mathematical models, and ease of working with high-dimensional data. Neural networks have been demonstrated in several practical applications for estimation of porosity and water saturation from well logs (e.g. Huang and Williamson, 1997; Helle et al., 2001; Shokir, 2004; Hamada and Elshafei, 2007; Al-Bulushi et al., 2007, etc). Multilayer perceptrons (MLP) trained by a back-propagation algorithm have been the conventional work for most practical applications over the last decade. Despite their wide applications, MLP show two major drawbacks. First, their network architecture and parameters is often by trail and error. Secondly, they have the possibility of being caught by local minima, which reduces network performance. A common approach is to train as many networks as possible and select the one that yields the best generalization performance. Hence the solution may not be unique, especially for the noisy data commonly encountered in well logging.

## **2.5 Review of Estimating Porosity and Water saturation Using Neural Networks**

Neural networks for quantitative analysis of reservoir properties from well logs have been demonstrated in several practical applications (e.g. Huang et al., 1996; Huang and Williamson, 1997; Zhang et al., 2000; Helle et al., 2001). They were shown to be simple and accurate alternative for converting well logs to common reservoir properties such as porosity, permeability and water saturation.

Soto et al. (1997) developed a back propagation neural network with four layers to predict permeability and porosity from log data. Gamma ray and neutron porosity were used as inputs to the porosity predictor network, with two hidden layers of two and three neurons respectively (2-2-3-1). Porosity, resistivity and spontaneous potential were used as inputs to the permeability predictor network, with two hidden layers of fifteen and twelve neurons respectively (3-15-12-1). A relatively good correlation was achieved. However, due to the use of trial and error in obtaining the network, this is time-consuming and cannot guarantee that the network is optimal.

Ahmed et al. (1997) also predicted permeability and porosity with neural networks from 3D seismic data (seismic amplitudes grid coordinate and instantaneous amplitude) and well logs (neutron porosity, density porosity) data. Those researchers tried to make the porosity prediction more reliable by using ten random seed values to generate ten different porosity networks, and the four networks with the least error were averaged

together to give a better result. This was done to avoid the effect of local minimal as a result of initial random weight. This is also time consuming and needs refinement.

Stan and Pedro, (1998) used artificial neural networks to classify and identify lithofacies, and they predicted permeability and porosity from well logs (Gamma ray, density log and deep resistivity) corresponding to each facies. The use of back-propagation suggests the weakness in their approach.

Bhatt et al. (2001a) demonstrated the prediction of porosity and permeability by using single neural networks. While using the committee machine approach of neural networks, Bhatt and Helle (2001) demonstrated better porosity and permeability predictions from well logs by using an ensemble combination of neural networks rather than selecting the single best by trial and error. Bhatt et al. (2001b) successfully applied a committee machine by using a combination of back propagation neural networks and recurrent neural networks for the identification of lithofacies.

Shokir (2004) proposed a back-propagation multilayer artificial neural network to predict water saturation from convectional well-log data (deep resistivity, self-potential, gamma-ray, litho-density, and neutron porosity log values) in a shaly Bahariya formation of the western desert of Egypt. The best performance network topology was found to be two hidden layers, each of 22 neurons. Although the predicted water saturation follows the trend very closely, the use of back-propagation indicates its inherent drawback. El-sayeed (2004) proposed the use of a model with two artificial neural networks to identify



the flow regime, and to estimate the liquid holding in horizontal multiphase flow with superficial gas and liquid velocities, pressure, temperature, and fluid properties.

Jong-se and Jungwhan (2004) and Jong-se (2005) proposed the use of combined fuzzy logic artificial neural networks to predict porosity and permeability. Fuzzy curve analysis was used to select the best inputs for the artificial neural network from the available conventional well log data (NPHI, DT, GR, CAL, LLD, RHOB, and LLS) by identifying the strength of their relationship with core permeability and porosity data. NPHI, CAL, LLD, LLS, RHOB, and SP were selected as inputs for porosity, while NPHI, DT, LLD, RHOB, and SP were selected for permeability prediction. Although reasonable results were obtained, the use of trial and error was a defect.

Bueno et al. (2006) developed artificial neural networks to predict open-hole logs from a cased hole paused neutron log (PNL) in a low porosity naturally fractured formation of the Bermudez basin in southern Mexico due to the drilling obstacles which prohibit the running of conventional open-well logs. The results clearly show good correlation between the predicted and the actual curves. They also observed that the use of artificial neural networks using PNL to detect open natural fractures may be problematic except where the natural fracture system has high radioactivity due to uranium.

Hamada and Elshafei (2007) also estimated formation porosity and water saturation of a sandstone reservoir with relatively satisfactory results by using two separate neural networks from well logging measurements (GR, LLD, RHOB, NPHI, PET, ST). The predicted values were used to calculate index values for porosity, water

saturation and shale volume, and subsequently to predict the possible pay zone in the tested well.

Nabil Al-Buhushi et al. (2007) developed a neural networks model for predicting water saturation in shaly formations by using wireline well logs and core data. A general workflow (methodology) was constructed in order to cover different design issues in the ANN modeling. The workflow shows how the neural network results can be interpreted by investigating the contribution of input variables, finding the optimum number of hidden neurons, and evaluating the different learning algorithms. In addition, the workflow focuses on the relevance of the statistics and on the importance of determining the uncertainties in the original data before using it in the model. A three layered feed-forward neural network model with five hidden neurons and the Resilient Back-Propagation (PROP) algorithm was found to be the best design. The input variables to the model are density, neutron, resistivity and photo-electric wireline logs. The neural network model was able to predict the water saturation directly from wireline logs with a correlation factor of 0.91 and a root mean square error (RMSE) of 2.5%. It was also found that Levenberg-Marquardt (LM) performs similarly to the PROP algorithm and better than back propagation. However, the optimum number of networks was still based on trial and error, which is time-consuming.

## 2.6 Statistical Quality Measures

To compare the performance and accuracy of the desired particular networks model to other models, statistical error analysis is performed. The statistical parameters used for comparison are: average percent relative error, average absolute percent relative error, minimum and maximum absolute percent error, root mean squares error, and the correlation coefficient. The corresponding mathematical equations for those parameters are given below:

**i. Average Percent Relative Error:** It is the measure of the relative deviation from the experimental data, that is,

$$E_r = \frac{1}{n} \sum_{i=1}^n E_i \quad (2.1)$$

where  $e_i$  is a relative error deviation between estimated value  $y_{est}$  and experimental value  $y_{exp}$ .

$$e_i = \left[ \left( \frac{y_{exp} - y_{est}}{y_{exp}} \right)_i \right] \times 100 \quad i = 1, 2, \dots, n. \quad (2.2)$$

**ii. Average Absolute Percent Relative Error:** It measures the relative absolute deviation from the experimental values, that is:

$$E_a = \frac{1}{n} \sum_{i=1}^n |E_i| \quad (2.3)$$

where  $E_i$  is given as:

$$E_i = (y_{est} - y_{exp})_i \quad i = 1, 2, \dots, n. \quad (2.5)$$

**iii. Minimum and Maximum Absolute Percent Relative Error:** This error type is used to define the range of error for each correlation. The calculated absolute percent relative error values are scanned to determine the minimum and maximum values, that is:

$$E_{\min} = \min_i |E_i| \quad (2.6)$$

$$E_{\max} = \max_i |E_i| \quad (2.7)$$

**iv. Root Mean Squares Error:** It measures the data dispersion around zero deviation, that is:

$$RMSE = \sqrt{\left[ \frac{1}{n} \sum_{i=1}^n E_i^2 \right]} \quad (2.8)$$

**v. The Correlation coefficient:** It represents the degree of success in reducing the standard deviation by regression analysis, that is:

$$r = \sqrt{1 - \frac{\sum_{i=1}^n [y_{exp} - y_{est}]^2}{\sum_{i=1}^n [y_{exp} - \bar{y}]^2}} \quad (2.9)$$

where

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n [y_{\text{exp}}]$$

# CHAPTER THREE

## NEURAL NETWORKS

### 3.2 Overview

Artificial neural networks (ANNs) developed significantly in the mid-1980s after major developments in neuroscience. A broad category of computer algorithms were designed to imitate the function of the biological neurons of the human brain. Neural networks are well suited to complex problems. They generally have large degrees of freedom, and thus they can capture the non-linearity of the process being studied better than conventional regression methods. ANNs are relatively insensitive to data noise, as they can determine the underlying relationship between model inputs and outputs, resulting in good generalization ability. They can make extensive use of prior knowledge, and they are not limited by the assumptions of the underlying model (Essenreiter et al., 1998). In predictive modeling, the goal is to map a set of input patterns onto a set of output patterns. Neural networks accomplish this task by learning from a series of input/output data sets presented to the network. The trained network is then used to apply what it has learned to approximate or predict the corresponding output.

Generally, neural networks are capable of solving several types of problems; including function approximation, pattern recognition and classification, optimization and

automatic control (Maren et al., 1990). The major element of the ANNs is the perceptron or artificial neuron. These neurons mimic the action of the abstract biological neuron. The most common type of neural networks is the multi-layer perceptron (MLP).

### 3.3 Structure of Neural Networks

Artificial neural networks (ANNs) are composed of signal processing elements called neurons. The model of a neuron is as shown in Figure 3.1. Its basic elements are:

- i) A set of synapses or connecting links, each characterized by a weight of its own. A signal  $x_j$  at the input of synapse  $j$  connected to neuron  $k$  is multiplied by the synaptic weight  $w_{kj}$ .
- ii) An adder for summing the input signals, weighted by the respective synapses of the neuron.
- iii) An activation function for limiting the amplitude of the output of a neuron. It limits the permissible amplitude range of the output signal to some finite value.

The neuronal model also includes an externally applied bias ( $b_k$ ) which has the effect of increasing or lowering the net input of the activation function, depending on whether it is positive or negative respectively.

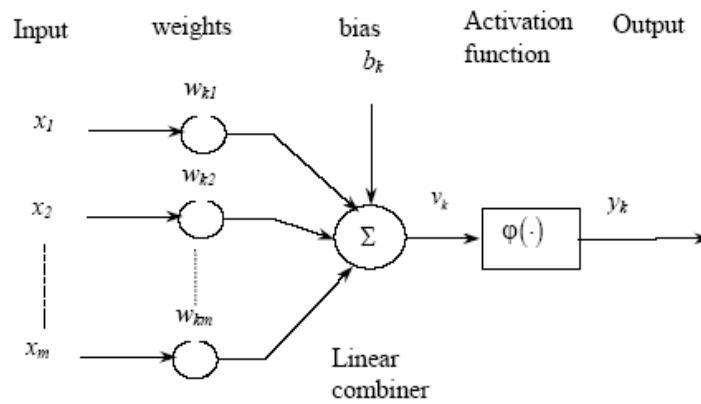


Figure 3.1: Model of a Neuron

Mathematically the function of the neuron  $k$  can be expressed by equation 3.1:

$$y_k = \phi(u_k + b_k) \quad (3.1)$$

where

$$u_k = \sum_{j=1}^m w_{kj} x_j \quad (3.2)$$

and where  $x_j$  is the input signal from an  $m$  dimensional input,  $w_{kj}$  is the synaptic weights of neuron  $k$ ,  $u_k$  is the linear combiner output due to the input signals,  $b_k$  is the bias,  $\phi(\cdot)$  is the activation function, and  $y_k$  is the output signal of the neuron. The relation between the linear combiner output  $u_k$  and the activation potential  $v_k$  is

$$v_k = u_k + b_k \quad (3.3)$$



The activation function  $\phi(v)$  defines the output of a neuron in terms of the induced local field  $v$ . There are different types of activation function, and the most commonly used are:

1. *Threshold Function*. A threshold (hard-limiter) activation is either a binary type or a bipolar type, for example for binary:

$$\phi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$

2. *Piecewise -Linear Function*. This type of activation function is also referred to as saturating linear function and can have either binary or bipolar range for the saturating limits of the output. The mathematical model for a symmetric saturation is described as follows:

$$\phi(v) = \begin{cases} 1 & \text{if } v \geq \frac{1}{2} \\ v & \text{if } +\frac{1}{2} > v > -\frac{1}{2} \\ 0 & \text{if } v \leq -\frac{1}{2} \end{cases}$$

3. *Sigmoid (S shaped) Function*. The sigmoid function is by far the most common form of activation function used in the construction of artificial neural network. It is defined as strictly increasing function that exhibits smoothness and asymptotic properties. An example of the sigmoid is the logistic function, defined by

$$\varphi(v) = \frac{1}{1 + e^{-av}}$$

where  $a$  is the slope parameter of the sigmoid function. By varying the parameter  $a$ , we can obtain sigmoid function of different slopes.

4. *Tangent Hyperbolic Function*: This transfer (activation) function is sometime used in place of sigmoid function and is described by the following mathematical form:

$$\varphi(v) = \tanh(v) = \frac{e^{av} - e^{-av}}{e^{av} + e^{-av}}$$

### 3.4 Multilayer Perceptron Feedforward Neural networks (MLP)

The most common neural networks in the literature are multilayer feedforward neural networks (MLPFFN) and recurrent neural networks (RNN). Generally, feedforward neural networks are the most commonly used neural network in both computational intelligence and petroleum engineering. In feedforward neural networks, the neurons are organized in different layers, and each of the neurons in one layer can receive an input from units in the previous layers without loss of generality. Figure 3.2 gives a simple example of a three-layer neural network that contains an input layer, two hidden layers, and an output layer, interconnected by modifiable weights, represented by

links between layers. A network that has only a single layer is referred to as a single-layer network, the term “single layer” referring to the output layer of computational neurons.

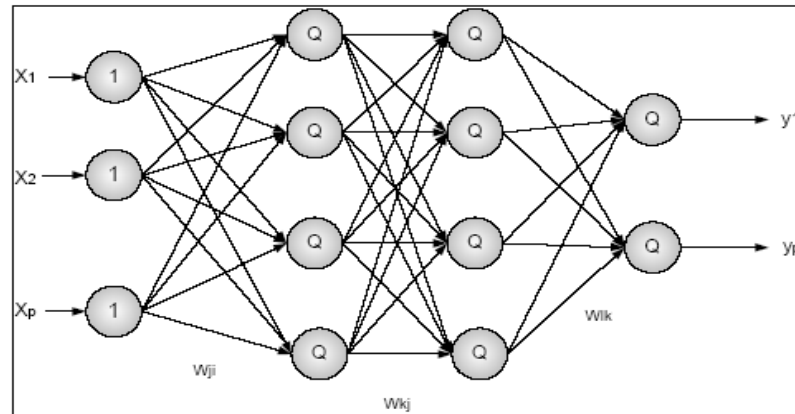


Figure 3.2: Structure of Feedforward Neural Network

The presence of one or more hidden layers, whose neurons are correspondingly called hidden neurons, enables the network to extract higher order statistics. thus network acquires a global perspective, despite its local connectivity, by virtual of the extra set of synaptic connections and the extra dimension of neural interaction. Such a network is called a “multilayer feedforward network”.

### 3.3.1 MLP Learning

The basic idea in the learning procedures is to provide the network with a training set of patterns having inputs and outputs. Real valued  $m$ -dimensional input feature vectors  $\mathbf{x}$  are presented to each of the first hidden layer units through the weight vector  $\mathbf{w}$ .

Hidden layer unit  $k$  receives input  $j$  through the synaptic weight,  $w_{kj}$ ,  $k = 1, 2, \dots, n$ , and  $j = 1, 2, \dots, m$ . Unit  $k$  computes a function of the input signal  $x$  and the weights  $w_{kj}$  and then it passes its output forward to all of the units in the next successive layer. Like the first hidden layer, the units of the second hidden layer are fully connected to the previous layer through the synaptic weights. These units also compute a function of their inputs and their synaptic weight and they pass their output on to the next layer. The output of one layer becomes the input to the following layer. Then, at the output, the unit error is calculated between the target value and the computed value of the pattern. This process is repeated until the final computation is produced by the output unit. The learning algorithm for this type of network is called the backpropagation (BP) algorithm, and it was published in the mid-1980s for multilayer perceptrons. This architecture of the network is the basic unit in this study. Hornik et al. (1989) suggested that, if a sufficient number of hidden units are available, then an MLP with one hidden layer having a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer can approximate any function to any degree of accuracy.

Back-propagation is a systematic method for training multilayer neural networks due to its strong mathematical foundation. Despite its limitations, back-propagation has dramatically expanded the range of problems it can solve. Many successful implementations demonstrate its power. The steps to implement the back-propagation algorithm are given as follows:

The error signal at the output of neuron  $j$  at iteration  $n$  (i.e. presentation of the  $n$ th training pattern) is defined by

$$e_j(n) = d_j(n) - y_j(n) \quad (3.4)$$

where

$d_j(n)$  is the desired response for neuron  $j$

$y_j(n)$  is the function signal appearing at the output of neuron  $j$

$e_j(n)$  refers to the error signal at the output of neuron  $j$ .

The instantaneous value of the sum of squared errors is obtained by summing the square error over all neurons in the output layer; which is written as:

$$\xi(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n) \quad (3.5)$$

The net internal activity level  $v_j(n)$  produced at the input of the nonlinearity associated with neuron  $j$  is therefore written as:

$$v_j(n) = \sum_{i=0}^P w_{ji}(n) y_i(n) \quad (3.6)$$

where  $p$  is the total number of inputs (excluding the threshold ) applied to neuron  $j$  and  $w_{ji}(n)$  denote the synaptic weight connecting the output of neuron  $i$  to the input of neuron  $j$  at iteration  $n$ . Hence the output of neuron  $j$  at iteration  $n$  is given as:

$$y_j(n) = \varphi_j(v_j(n)) \quad (3.7)$$

The instantaneous gradient which is proportional to the weight correction term is given as:

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = -e_j(n) \varphi_j'(v_j(n)) y_j(n) \quad (3.8)$$

The correction  $\Delta w_{ji}(n)$  applied to  $w_{ji}(n)$  is defined by the delta rule as:

$$\Delta w_{ji}(n) = \eta \frac{\partial \xi(n)}{\partial w_{ji}(n)}$$

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)$$

$$\delta_j(n) = e_j(n) \varphi_j'(v_j(n)) \quad (3.9)$$

When neuron  $j$  is located in a hidden layer of the network, the local gradient is redefined as:

$$\delta_j(n) = -\frac{\partial \xi(n)}{\partial y_j(n)} \phi_j'(v_j(n))$$

$$\delta_j(n) = \phi_j'(v_j(n)) \sum_k \delta_k w_{kj}(n) \quad (3.10)$$

where the  $\delta_k$  requires the knowledge of the error signals  $e_k$  for all those neurons that lie in the layer to the immediate right of hidden neuron  $j$ . The  $w_{kj}(n)$  consists of the synaptic weights associated with these connections. We are now ready to put forward the weight correction update for the back-propagation algorithm, which is defined by the delta rule:

$$\Delta w_{ji}(n) = \eta \delta_j y_j \quad (3.11)$$

It is important to note that weight correction term depends on whether neuron  $j$  is an output node or a hidden node:

- a. if neuron  $j$  is an output node, equation 3.10 is used for the computation of the local gradient.
- b. if neuron  $j$  is a hidden node, equation 3.11 is used for the computation of local gradients.

The network performance is checked by monitoring the average square error. The average squared error is obtained by summing  $\xi(n)$  over all  $n$  and then normalizing with respect to  $N$  (number of training patterns)

$$\xi_{av} = \frac{1}{N} \sum_{n=1}^N \xi(n) \quad (3.12)$$

Both the instantaneous and average square errors are functions of free parameters (synaptic weights and biases). The process is repeated several times for each pattern in the training set, until the total output squared error converges to a minimum, or until some limit is reached in the number of training iterations.

One of the major problems with the BP algorithm has been the long training times due to the steepest descent method of minimization, which is simple but slow. The learning rate is sensitive to the weight changes. The smaller the learning rate, the smaller will be the changes to the synaptic weights from one iteration to the next, and the smoother will be the trajectory in the weight space.

On the other hand, if the learning rate is chosen too large in order to speed up the rate of learning, the resulting large changes in the synaptic weights make the network unstable. In order to speed up the convergence of the BP algorithm, along with improved stability, a momentum term is added to the weight update of the BP algorithm. A momentum term is simple to implement, and this significantly increases the speed of convergence. The inclusion of momentum term represents a minor modification to the



weight update. The inclusion of momentum may also have the benefit of preventing the learning process from terminating in shallow local minima on the error surface.

The second method of accelerating the BP algorithm is by using the Levenberg - Marquardt BP (LMBP) algorithm (Hagan et al., 1996). It is based on Newton's optimisation method (Hagan et al., 1996) and differs from the usual BP algorithm in the manner in which the resulting derivatives are used to update the weights. The main drawback of the LMBP algorithm is the need for large memory and storage space of the free parameters in the computers. If the network has more than a few thousand parameters, the algorithm becomes impractical on current machines. In this study, the feedforward network architecture used for our comparison has been designed to have several free parameters to be smaller than the number of training patterns in order for LMBP algorithm to be adequate for training the network.

### **3.3.2 Generalization**

In training MLP by the back-propagation algorithm, we compute the synaptic weights of a MLP by using as many training examples as possible into the network. The hope is that the neural network so designed will generalize. A network is said to generalize well when the input-output mapping computed by the network is correct for test data which is unknown to the network. A well designed neural network will produce a correct input-output mapping, even when the input is slightly different from the examples used to train the network. However, when a neural network has too many hidden neurons, the network may end up memorizing the training data. It may do so by

finding a feature that is present in the training data (noise) but not true of the underlying function that is to be modeled. This phenomenon is referred to as overfitting. Overfitting is the result of more hidden neurons than actually necessary, with the result that undesired contributions in the input space due to noise are stored in synaptic weights. However, if the number of hidden neurons is less than the optimum number, then the network is unable to learn the correct input-output mapping. Hence it is important to determine the optimum number of hidden neurons for a given problem.

Generalization is influenced by three factors (Haykin, 1999): (i) the size of the training set, (ii) the architecture of the neural network, and (iii) the complexity of the problem at hand. We cannot control the latter. In the context of the first two, for a good generalisation to occur, we may vary the size of the training set by keeping the architecture of the network fixed or vice versa. This problem can be resolved in terms of the Vapnik-Chervonenkis (VC) dimension, which is a measure of the capacity or expressive power of the family of classification functions realised by a network. It can be defined as the maximum number of training examples for which a function can correctly classify all the patterns in a test dataset (Haykin, 1999). The bounds specified by the VC dimension can be simply stated as follows: for an accuracy level of at least 90%, one should use ten times as many training examples as there are weights in the network. It has been suggested that, to achieve a good generalization, the size of the training set,  $N$ , should satisfy the condition

$$N = O\left(\frac{w}{\epsilon}\right) \quad (3.13)$$

where  $w$  is the total number of free parameters (i.e. synaptic weight and biases) in the network, and  $\epsilon$  denotes the fraction of classification errors permitted on the test data, and  $O(\cdot)$  denotes the order of quantity enclosed within.

### **3.5 Advantages and Disadvantages of Neural Networks**

An MLP network generates a nonlinear relationship between inputs and outputs by the interconnection of nonlinear neurons. The nonlinearity is distributed throughout the network. It does not require any assumption about the underlying data distribution for designing the networks. Hence the data statistics do not need to be estimated. MLP parallel structure makes it realizable in parallel computers. The network exhibits a great degree of robustness or fault tolerance because of built-in redundancy. Damage to a few nodes or links thus need not impair overall performance significantly. It can form any unbounded decision region in the space spanned by the inputs. Such regions include convex polygons and unbounded convex regions. The network has a strong capability for function approximation. The abilities to learn and generalize are additional qualities. Previous knowledge of the relationship between input and output is not necessary, unlike for statistical methods. The MLP has a built-in capability to adapt its synaptic weights to changes in the surrounding environment by adjusting the weights to minimize the error.

Experience with neural networks has revealed a number of drawbacks for the technique. For an MLP network, the topology is important for the solution of a given problem, i.e. the number of hidden neurons, the size of the training dataset, and the type of transfer function(s) for neurons in the various layers. Learning algorithm parameters to

be determined include initial weights, learning rate, and momentum. With no analytical guidance on the choice of many design parameters of the network, the developer often follows an ad hoc, trial-and-error approach of manual exploration that naturally focuses on just a small region of the potential search space. Although acceptable results may be obtained with effort, it is obvious that potentially superior models can be overlooked, and the considerable amount of user intervention certainly slows down model development. Because of the distributed nonlinearity and the high connectivity of the network, a theoretical analysis of the network response is difficult to undertake. The use of hidden neurons makes the learning process harder to visualize. The associated lack of explanation capabilities is a handicap in many decision support applications such as medical diagnostics, where the user would usually like to know how the model came to a certain conclusion. Additional analysis is required to derive explanation facilities from neural network models; e.g. through rule extraction [Kartoatmodjo and Schmidt, 1994]. Model parameters are buried in large weight matrices, making it difficult to gain insight into the modeled phenomenon or to compare the model with available empirical or theoretical models. Large-scale MLP networks have extremely low training rates when the back-propagation algorithm is used, since the networks are highly nonlinear in the weights and thresholds. Using the LM algorithm, as mentioned previously, is one of the ways used now to overcome this problem. The mean square error surface of a multiple network can have many local minima and a global minimum, and the network may get stuck in the local minima instead of converging into the global minimum. As a result, the

output of a single network may not be satisfactory. In order to address some of these problems, a number of improvements have been proposed.

### **3.6 Advances in Neural Networks**

The problems of back propagation as discussed above, and the need for finding the appropriate architecture, are still challenging tasks. This, however, has produced some algorithms that either provide alternative approaches to neural networks (e.g. function networks) or improvements to the neural networks design (e.g. cascaded correlation neural networks) to learn the necessary architecture from data. Over the years, some methods have been developed to improve neural network topology design. These are categorized into three major groups according to how they handle the problem of constructing network structures (Stepniewskir and Keane, 1997). A particular problem can allow a topology to (i) shrink only, (ii) expand only, (iii) shrink and expand. The shrinking approach starts with a large network, and then, it gradually removes the unnecessary nodes and connections. This is also known as pruning (Thomas et al., 1997). In the expand approach, also known as the constructive approach, the algorithm starts with a small network, and it constructs the network by adding nodes and connections. Cascade correlation neural networks are probably the most known example that utilizes this approach (Stepniewskir and Keane, 1997). The shrink-expand approach utilizes optimization techniques such as genetic algorithm for network pruning and also for the schemes that allow networks to grow and shrink. These types of neural network are

called evolutionary neural networks. Below are some of the advances in neural networks used in this thesis.

### **3.5.1 Cascade Correlation Neural Networks (CCNN)**

Cascade-correlation Neural Network (CCNN) is a constructive approach generally preferred to pruning. It was designed by Fahlman and Lebiere (1990) to optimize the network topology. CCNN combines two main ideas. The first is, the cascade architecture, in which hidden units are added to the network one at a time, and do not change after they have been added. The second is the learning algorithm, which creates and installs the new hidden units. For each new hidden unit, an attempt is made to maximize the magnitude of the correlation between the new unit's output and the residual error signal. The algorithm is summarized as follows:

The network starts with a minimal topology, consisting only of the required input and output units (and a bias input that is always equal to 1). This net is trained until no further improvement is obtained. The error for each output unit is then computed (summed over all training patterns).

Next, one hidden unit is added to the net in a two-step process. During the first step, a candidate unit is connected to each of the input units, but not to the output units. The weights on the connections from the input units to the candidate unit are adjusted to maximize the correlation between the candidate's output and the residual error at the

output units. The residual error is the difference between the target and the computed output, multiplied by the derivative of the output unit's activation function, i.e. the quantity that would be propagated back from the output units in the back-propagation algorithm. When this training is completed, the weights are frozen, and the candidate unit becomes a hidden unit in the net. The second step, in which the new unit is added to the net, now begins. The new hidden unit is then connected to the output units, with the weights on the connections being adjustable. Now all connections to the output units are trained. (Here the connections from the input units are trained again, and the new connections from the hidden unit are trained for the first time.)

A second hidden unit is then added by using the same process. However, this unit receives an input signal from the both input units and the previous hidden unit. All weights on these connections are adjusted and then frozen. The connections to the output units are then established and trained. The process of adding a new unit, training its weights from the input units and the previously added hidden units, and then freezing the weights, followed by training all connections to the output units, is continued until the error reaches an acceptable level or the maximum number of epochs (or hidden units) is reached. Figure 3.3 shows the cascaded correlation architecture.

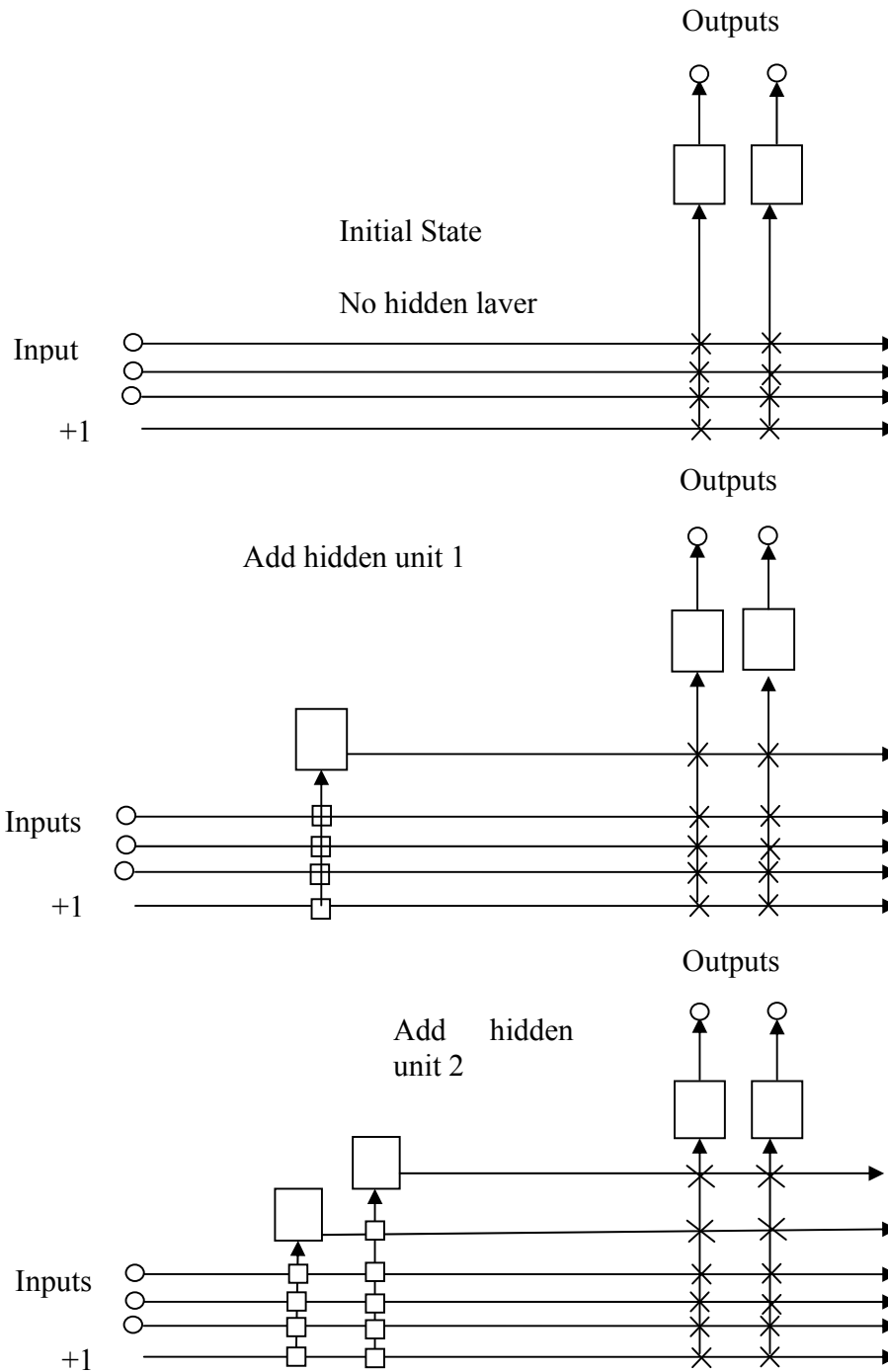


Figure 3.3 Cascade architecture, initial state and after adding two hidden units. The vertical lines sum all incoming activation. Boxed connections are frozen, whereas X connections are trained repeatedly.



The CCNN architecture is faster, and it also avoids trial and error in selecting the number of hidden layer neurons with less error (Mckenna et al., 1997; Schetinin, 2003; Hamidreza and karim, 2004; Chandra and Paul, 2007). However, cascaded correlation neural networks have been criticized for not finding the desired solution sometimes due to the freezing of the existing weight which allows the solution to be found only in an affine subset of the weight space (Oya and Ethhem, 2003). This thesis investigates the application CCNN in reservoir characterization as one of the advancement to neural networks.

### **3.5.2. Polynomial Networks (Abductive Networks)**

An abductive network, like a neural network, is a set of interconnected nodes. In most types of neural nets, each node performs the same kind of simple computation. In an abductive network, the nodes' computations can differ from one another and become mathematically complex. Abductive modeling is the search for the types of nodes and the architecture of their interconnections that minimize the predictive error in a set of training data. The goal is to build a model that generalizes well to a set of test data.

Abductive networks are an alternative modeling tool that avoids many of the neural networks limitations. While the processing elements in neural networks are restricted by the neuron analogy, abductive networks use various types of more powerful polynomial functional elements based on prediction performance. Based on the self-organizing algorithm called the group method of data handling (GMDH) (Farlow, 1984),

this technique uses well-proven optimization criteria to automatically determining the network size and connectivity, the element types and coefficients for the optimum model. Thus, GMDH reduces the modeling effort and the need for user intervention. The abductive network model automatically selects influential input parameters, and the input-output relationship can be expressed in polynomial form. This enhances explanation capabilities, and it allows comparison of the resulting data-based machine learning models with existing first principles or empirical models.

A typical structure of a polynomial network used in this thesis is shown in Figure 3.4. The models take the form of layered feed-forward abductive networks of functional elements (nodes). Functions are selected from the set of bivariate quadratic polynomials. The training algorithm selects the best polynomial for each combination of the input variables by using the method of least squares. At each layer, for each pair of input variables, the best transfer polynomial is selected and its parameters are optimized to fit the given data in the least squared error sense as follows:

$$J(i, j) = \min_{k, w} \left\{ \sum_{m=1}^M (y(m) - f_k(w_k, v_i(m), v_j(m)))^2 \right\} \quad (3.15)$$

where  $\{v_j(m)\}$ , for  $j = 1, 2, \dots, I_1$  are the outputs of the previous layer,  $\{f_k\}$  is a given set of functions,  $w_k$  is the vector of the parameters of the function, i.e. polynomial coefficients.

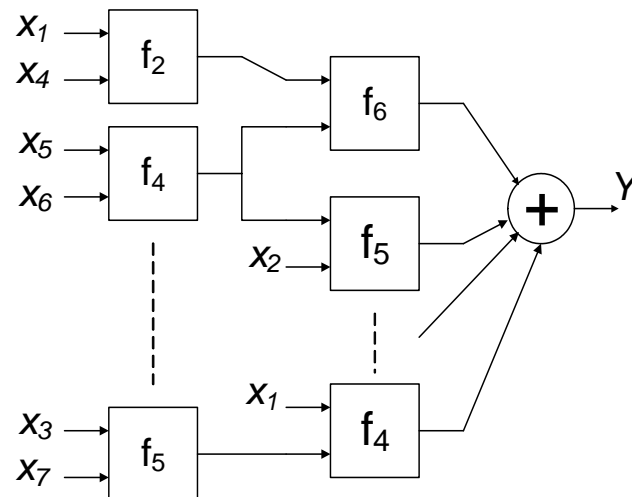


Figure 3.4: Typical polynomial networks

The training algorithm proceeds as follows:

1. At the input layer, for each pair of input variables, we select the best transfer polynomial which fits the given data in the least squared error sense.

$$J(i, j) = \min_{k, w} \left\{ \sum_{m=1}^M (y(m) - f_k(w_k, x_i(m), x_j(m)))^2 \right\} \quad \text{For } i, j=1, 2, \dots, N \quad (3.16)$$

The polynomial is defined by its index  $k$ , and its optimized coefficient vector  $w_k$ . For each  $k$ , we apply the LS technique to find the parameter vector  $w_k$  and we compute the residual mean squared error. The best polynomial is the one which achieves the least residual sum of squared error. Clearly, the number of the generated nodes is  $N^2$ .

2. Next, we sort the nodes in ascending order of their LSE, and we select the best  $l_1$  nodes, where  $l_1$  is the desired maximum number of nodes in the first layer.
3. Compute the outputs  $\{v_j^1(m)\}$ , for  $j = 1, 2, \dots, l_1$
4. The nodes of the subsequent layers are computed by repeating the same steps, but replacing the input  $x$  by  $v$  :

$$J(i, j) = \min_{k, w} \left\{ \sum_{m=1}^M (y(m) - f_k(w, v_i(m), v_j(m)))^2 \right\} \quad (3.17)$$

The residual minimum error is then sorted and the best  $l_i$  node is retained. The output from the retained nodes are computed and used in the subsequent layers.

5. Pruning: starting from the last hidden layer, we trace back the nodes, identifying the signal path from each node in the final layer to the input variables. We keep only the nodes along these paths, while the unused nodes are deleted. The network in this case will have at most  $l_1$  node in the first layer, at most  $l_2$  nodes in the second layer, and so on.
6. Finally, the weight of the output node is computed by minimizing the squared error

$$E = \min_w \left\{ \sum_{m=1}^M (y(m) - \sum_{i=1}^{l_f} w_{f,i} v_i^f(m))^2 \right\} \quad (3.18)$$

where  $l_f$  is the number of nodes of the final hidden layer.

### 3.5.3. General Regression Neural Networks

GRNN (Specht, 1991) falls into the category of probabilistic neural networks. Although GRNN may not be considered as one of the significant advances from the standard neural network, it is a special type neural network, and so it has some important advantages over standard FFNN, which include:

- GRNN network is usually much faster to train than an MLP network.
- GRNN networks are often more accurate than MLP networks.
- GRNN networks are relatively insensitive to outliers (wild points).
- The additional knowledge needed to get a satisfying fit is relatively small, no additional input by the user is required.

This makes GRNN a very useful tool to perform predictions and comparisons of system performance in practice. The architecture of GRNN is discussed below and illustrated in Figure 3.5:

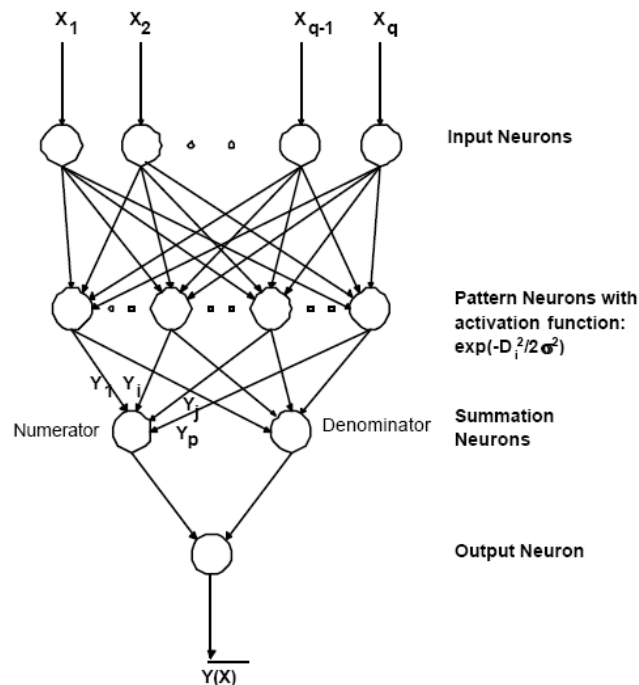


Figure 3.5: Typical General Regression network

All GRNN networks have four layers: Input layer – There is one neuron in the input layer for each predictor variable. The input neurons (or processing before the input layer) standardize the range of the values by subtracting the median and dividing by the interquartile range. The input neurons then feed the values to each of the neurons in the pattern layer.

Pattern layer – This layer has one neuron for each case in the training data set. The neuron stores the values of the predictor variables for the case along with the target value. When presented with the  $x$  vector of input values from the input layer, a pattern neuron computes the Euclidean distance of the test case from the neuron's center point,

and a radial basis function (RBF) (also called a kernel function) is applied to the distance to compute the weight (influence) for each point,  $Weight = RBF(distance)$  (equation 3.16). The calculations performed in each pattern neuron of GRNN are  $\exp(-D_i^2/2\sigma^2)$ , with the normal distribution centered at each training sample. The radial basis function is so named because the radius distance is the argument to the function. The farther some other point is from the new point, the less influence it has. Different types of radial basis functions can be used, but the most common is the Gaussian function. The peak of the radial basis function is always centered on the point it is weighting. The function's sigma value ( $\sigma$ ) of the function determines the spread of the RBF function; that is, how quickly the function declines as the distance increases from the point. With larger sigma values and more spread, distant points have a greater influence. Then the resulting value is passed to the neurons in the summation layer.

Summation layer – There are only two neurons in the summation layer. One neuron is the denominator summation unit, whereas the other is the numerator summation unit. The denominator summation unit adds up the weight values coming from each of the hidden neurons. The numerator summation unit adds up the weight values multiplied by the actual target value for each hidden neuron.

Decision layer/ Output – The decision layer divides the value accumulated in the numerator summation unit by the value in the denominator summation unit, and it uses the result as the predicted target value.

The primary work of training a GRNN network is the selection of the optimal sigma values to control the spread of the RBF functions.

$$Y(x) = \frac{\sum_{i=1}^n y_i \exp(-D_i^2/2\sigma^2)}{\sum_{i=1}^n \exp(-D_i^2/2\sigma^2)} \quad (3.16)$$
$$D_i^2 = (x - x_i)^T (x - x_i)$$



# CHAPTER FOUR

## FUNCTIONAL NETWORKS

### 4.1 Overview

Functional networks were introduced by Castillo et al. (1998) as a powerful alternative to neural networks. Unlike neural networks, functional networks have the advantage that they use domain knowledge in addition to data knowledge. The network initial topology is derived from the modeling of the properties of the real world. Once this topology is available, functional equations allow a much simpler equivalent topology to be obtained. Although functional networks also can deal with data only, the class of problems where functional networks are most convenient is the class where the two sources of knowledge about domain and data are available (Castillo et al., 2000a).

Functional networks as a new modeling scheme has been used in solving both prediction and classification problems. It is a general framework useful for solving a wide range of problems in engineering, statistics, and functions approximations. Several studies have been done to compare its performance with the performance of the most popular predictive modeling techniques for data mining, and machine learning schemes in the literature (Castillo, 1998). The results show that functional networks outperform most of the popular modeling schemes including neural networks.

In this chapter, we describe the functional networks, their architecture, and the steps needed for working with them. The learning and training techniques, and the selection criteria for choosing the best network model are discussed. The differences between functional networks and the standard neural networks are also pointed out.

## 4.2 Functional Networks Background and Definition

Functional network is defined as a pair  $\langle X, \Gamma \rangle$  where  $X$  is a set of nodes and  $\Gamma = \{\langle Y_j, f_j, Z_j \rangle, j=1,2,\dots,p\}$  is a set of neuron functions over  $X$ , such that every node  $x_j \in X$  must be either an input or an output node of at least one neuron function in  $\Gamma$ . The node  $x_j \in X$  for all  $j$  is called a multiple node, if it is an output of more than one neuron functions. Otherwise, it is called a simple node.

Functional Unit (also called a neuron)  $U$  over the set of nodes  $X$  is a triplet  $\langle Y, f, Z \rangle$ , where  $Y, Z, \subset X$ ;  $Y \neq \emptyset, Z \neq \emptyset, Y \cap Z = \emptyset$ , and  $f: \chi \rightarrow \xi$  is a given function. We say that  $Y, Z$ , and  $f$  are respectively the set of input nodes, the set of output nodes, and the processing function of the functional unit  $U$ .

Input Node in a Functional Network  $\langle X, \Gamma \rangle$  is the input node of at least one functional unit in  $\Gamma$  and is not the output of any functional unit in  $\Gamma$ .

Output Node in a Functional Network  $\langle X, \Gamma \rangle$  is the output node of at least one functional unit in  $\Gamma$  and is not the input of any functional unit in  $\Gamma$ .

Intermediate Node in a Functional Network  $\langle \{x_5, x_5\}, f_4, \{x_7\} \rangle$  is the input node of at least one functional unit in  $\Gamma$  and, at the same time, is the output node of at least one functional unit in  $\Gamma$ .

Figure 4.1 illustrates an example of functional networks  $N = \langle X, \Gamma \rangle$  where  $X = \langle x_1, x_2, \dots, x_7 \rangle$  and  $\Gamma$  consist of functional units of:

$$\langle \{x_1, x_2\}, f_1, \{x_5\} \rangle$$

$$\langle \{x_2, x_3\}, f_2, \{x_6\} \rangle$$

$$\langle \{x_3, x_4\}, f_3, \{x_6\} \rangle$$

$$\langle \{x_5, x_5\}, f_4, \{x_7\} \rangle$$

It is clear from the above definitions and figure that functional networks consist of:

- a) A layer of nodes for receiving the input data ( $x_i; i = 1, 2, 3, 4$ ), another layer for the output data ( $x_7$ ) and none or one or more layers for intermediate information ( $x_5$  and  $x_6$ );

- b) Processing units that evaluate a set of input values and deliver a set of output values ( $f_i$ ); and
- c) A set of directed links that connect the inputs or intermediate layers to neurons and neurons to intermediate or output units.

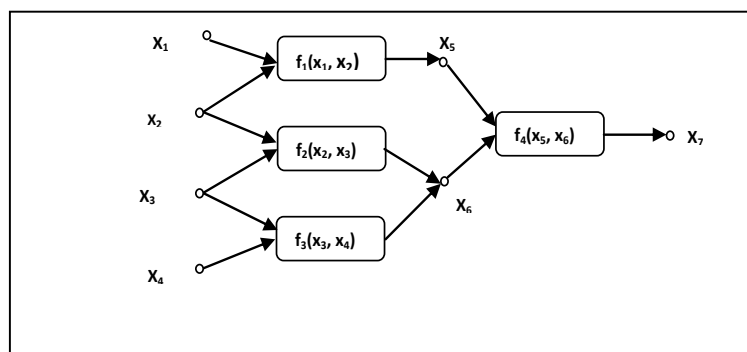


Figure 4.1: Example of Functional Network architecture

### 4.3 Comparison between Functional networks and Neural Networks

Although functional networks extend neural networks, both networks are different in the way they handle problems and in structure. The characteristics and key features of functional networks, as compared with those of neural networks, are for example as shown in Figure 4.2 (Castillo et al., 2001).

1. In selecting the topology of functional networks, the required information can be derived from the data, from domain knowledge, or from different combinations of the two. In the case of standard neural networks, only the data are used. This implies that, in addition to the data information, other properties of the function being modeled by the

functional network can be used for selecting its topology (associativity, commutativity, invariance, etc.). This information is available in some practical cases.

2. In standard neural networks, the neuron functions are assumed to be fixed and known, and only the weights are learned. In functional networks, however, the functions are learned during the structural learning (which obtains simplified network and functional structures) and estimated during the parametric learning (which consists of obtaining the optimal neuron function from a given family).

3. Arbitrary neural functions can be assumed for each neuron (e.g., neurons  $f_1$ ,  $f_2$ ,  $f_3$ ,  $f_4$ , and  $f_5$  in Figure 4.2(b)), while in neural networks they are fixed sigmoidal functions.

4. In functional networks, weights are not needed, since they can be incorporated into the neural functions.

5. The neural functions are allowed to be truly multiargument [e.g., neural functions  $f_1$ ,  $f_2$ ,  $f_3$ ,  $f_4$ , and  $f_5$  in Figure 4.2(b)]. However, in many cases, they can be equivalently replaced by functions of single variables. Note that in standard neural networks the neural sigmoidal functions are of a single argument, though this is a linear combination of all inputs (pseudo-multiargument functions).

6. In functional networks, intermediate or output units can be connected (linked) to several storing units, say  $m$  units, indicating that the associated values must be equal. Each of these common connections represents a functional constraint in the model, and allows writing the value of these output units in different forms (one per different link).

Intermediate layers of units are introduced in functional network architectures to allow several neuron outputs to be connected to the same units, which is not possible in neural networks.

7. Functional networks are extensions of neural networks. In other words, neural networks are special cases of functional networks. For example, in Figure 4.2, the neural network and its equivalent functional network are shown. Note that weights are subsumed by the neural functions.

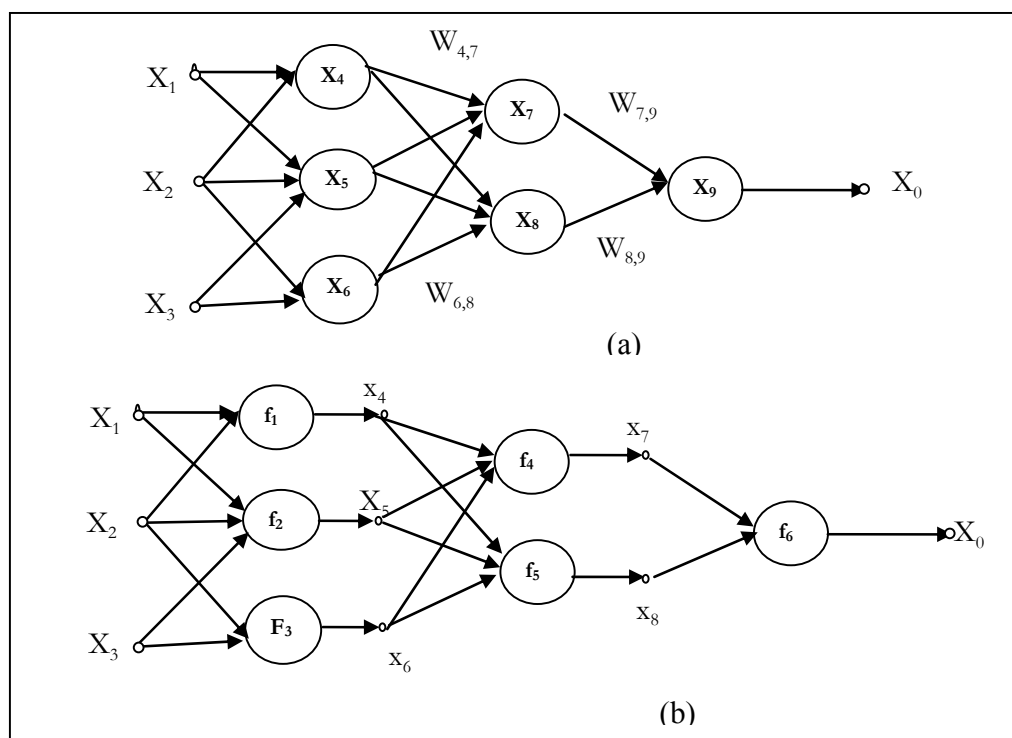


Figure 4.2 : (a) Standard Neural Network (b) its equivalent functional network.

#### 4.4 Working with Functional Networks

Functional networks methodology can be more easily understood by organizing it into the following steps:

**Step 1:** Statement of the problem: Understanding the problem to be solved.

**Step 2:** Select the suitable initial architecture: The selection of the initial topology of a functional network is based on the characteristics and the knowledge of the problem at hand, which usually leads to a single clear network structure.

**Step 3:** Simplifying the initial functional network: The initial functional network is simplified by using functional equations. Given a functional network, we wish to determine whether there exists another functional network giving the same output for any given input. Functional equations are the main tool for simplifying functional networks. Further information about the functional equations and their solutions can be found in (Castillo et al, 2005). In general, functional network architecture can be represented by functional equations, and their solutions lead to an equivalent but simpler one. This will lead to the idea of network equivalence.

**Equivalent Functional Networks:** We say that the two functional networks  $\langle X, \Gamma_1 \rangle, \langle X, \Gamma_2 \rangle$  are equivalent, if they give the same output for any given input. For example, the functional network in Figure 4.3(a) corresponds to the functional equation:

$$F[P(x, y), Z] = G[x, Q(y, Z)] \quad (4.1)$$

which can be simplified as follows:

A general solution of functional equations in (4.1) is:

$$\begin{aligned}
 F(x, y) &= k[f(x) + r(y)], \\
 P(x, y) &= f^{-1}[p(x) + q(y)], \\
 G(x, y) &= k[n(x) + p(y)], \\
 Q(x, y) &= n^{-1}[q(x) + r(y)],
 \end{aligned}
 \tag{4.2}$$

where  $f, r, k, m, p, q$  and  $n$  are arbitrary continuous and strictly monotonic functions.

Substituting equation 4.2 in equation 4.1, we obtain

$$d = D(x, y, z) = k[p(x) + q(y) + r(z)] \tag{4.3}$$

Therefore, the functional network in equations 4.1 and 4.2 are equivalent but equation 4.3 is much simpler, as can be seen in Figure 4.3(b). We note that the initial functional network in equation 4.1 has six functions:  $F, G, I, P,$  and  $Q$  and each of them depend on two arguments. On the other hand, the simplified functional network in equation 4.3 has only four functions:  $p, q, r$  and  $k$  and each one depends only on one argument; yet the two networks are equivalent. The problem then reduces to estimating these four functions based on the available data.



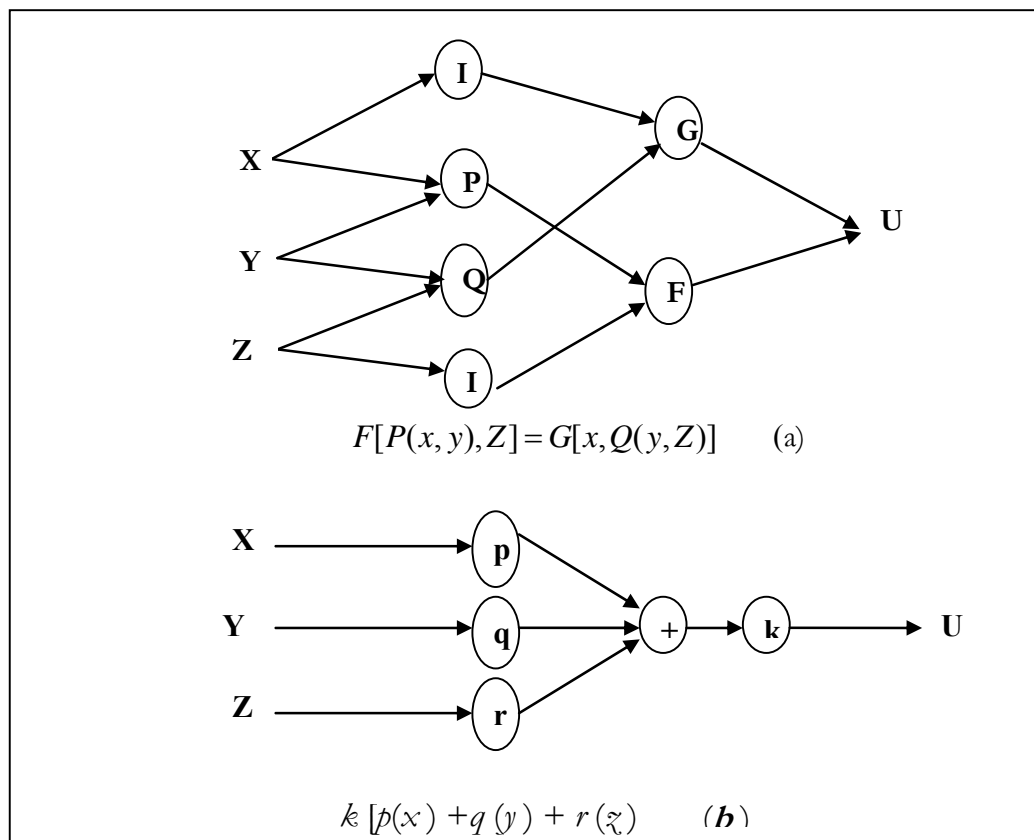


Figure 4.3: (a) Initial network, (b) Equivalent simplified network.

**Step 4:** Checking uniqueness of representation: Before learning a functional network, we have to find the conditions that make the neural functions of the simplified functional network unique. That is, for a given topology, there are several sets of neuron functions leading to exactly the same outputs for any input. Thus, we must check the uniqueness conditions on representation of this functional network, and we must find the constraints that neural functions of the simplified functional network must satisfy. For example, we can find the uniqueness of equation 4.3 represented by the functional

network shown in figure 4.3(b) by using theorem 6.5 in Castillo et al (2005, p.100).

Assume that there are two set of functions  $\{k_1, p_1, q_1, r_1\}$  and  $\{k_2, p_2, q_2, r_2\}$  such that they give the same output for the same input, i.e. the functional equation:

$$k_1 [p_1(x) + q_1(y) + r_1(z)] = k_2 [p_2(x) + q_2(y) + r_2(z)] \quad (4.4)$$

gives a general solution of :

$$\begin{aligned} k_2(u) &= k_1 \left( \frac{u - b - c - d}{a} \right), \\ p_2(x) &= ap_1(x) + b, \\ q_2(y) &= aq_1(y) + c, \\ r_2(z) &= ar_1(z) + d \end{aligned} \quad (4.5)$$

Where a, b, c and d are arbitrary constants. Thus, uniqueness in this case requires fixing the function k, p, q, and r at a point. For example, force  $f(x^*) = y^*$  to have uniqueness.

**Step 5:** Learning Algorithm: Once the structure of the functional network is known in Step 2, the neural functions of the network must be learned (estimated) in order to approximate the associated parameters of the neuron functions.

**Step 6:** Network model validation: After the learning algorithm process is done, it is essential to do the test for quality of the functional network model, to assess its performance. At this step, a test for quality and/or the cross-validation of the model is performed. Checking the obtained error is important to see whether or not the selected

families of approximating functions are suitable. A cross-validation of the model, using testing data as opposed to the training set, is also important.

**Step 7:** Use of the functional network model: Once the functional network model has been satisfactorily validated, it is ready to be used for the purpose in real world.

## 4.5 Training Functional Networks

In functional networks, two types of learning are used: (a) structural and (b) parametric (Castillo et al., 2001). Structural learning is the simplification of the initial topology of the network, as discussed earlier. It is based on some properties available to the designer, initially to arrive at a topology and finally to simplify the architecture by using functional equation. Parametric learning, on the other hand, is concerned with estimation of the neuron functions. This can be done by considering the combination of shape functions (basis functions), and by estimating the associated parameters from the given data. A neuron function is represented as

$$f_i = \sum_{j=1}^{m_i} a_{ij} \varphi_{ij} \quad (4.6)$$

where  $a_i$  is the weight to be estimated;  $\varphi_i = \{\varphi_{i1}, \varphi_{i2}, \dots, \varphi_{im_i}\}$  is a family of linearly independent functions, such as polynomial  $(1, x, x^2 \dots x^n)$  or Fourier function  $(1, \sin(x), \cos(x), \sin(2x), \cos(2x), \dots, \sin(nx), \cos(nx))$  or exponential functions or any other linearly independent function. The coefficients  $a_{ij}$  are the parameters of the functional

networks to be determined. The learning method consists of obtaining the neural functions based on a set of data  $\{D = (I_i, O_i); i=1,2,\dots,n\}$ . The learning process is based on minimizing the Euclidean Norm of error function given by

$$E = \frac{1}{2} \sum_{i=1}^{m_i} \{O_i - f_i\}^2 \quad (4.7)$$

The associated optimization function may lead to a system of linear or nonlinear algebraic equations, which can be solved numerically by using optimization methods such as least square, min-max, etc. The least square method is adopted in this study.

#### 4.6 Model Selection in Functional Networks

To learn (parametric) functional networks, we can choose different sets of linearly independent functions for the approximation of the neuron functions. This requires us to select the best model according to some criteria of optimality. The Minimum Description Length (MDL) principle is one of the model selection principles we can use (Castillo et al. 1999). The idea behind the MDL measure is to find the minimum information required to store the given training set by using the functional network model. Therefore, we can say that the best functional network model for a given problem corresponds to that with the minimum description length value. The code length  $L(x)$  of  $x$  is defined as the amount of memory needed to store the information  $x$ . For example, to store the data in the functional network in (4.3), we have two options:

Option 1: Store Raw Data:

Store the triplets  $\{(x_i, y_i, z_i, u_i) | i = 1, 2, 3, \dots, n\}$ . In this case, the initial description length of the data set is

$$DL = \sum_{i=1}^n [L(x_i) + L(y_i) + L(z_i) + L(u_i)] \quad (4.8)$$

Option 2: Use a Model:

By selecting a model, we try to reduce this length as much as possible. In this case, we can store the parameters of the model  $\{a_i\}_{i=1}^m$  and then the residuals are  $e_i = u - \hat{k}[\hat{p}(x) - \hat{q}(y) - \hat{r}(z)], i = 1, 2, \dots, n$  where  $\hat{k}, \hat{p}, \hat{q},$  and  $\hat{r}$  are the approximate neuron functions of the model. The description length becomes

$$DL_{\text{model}} = \sum_{i=1}^n [L(x_i) + L(y_i) + L(z_i)] + \sum_{i=1}^k L(a_i) + \sum_{i=1}^n L(e_i | \text{model}) \quad (4.9)$$

where  $L(a_i)$  is the code length of the estimated parameters  $\{a_i\}_{i=1}^m$ .

Generally, the description length is a measure for comparing not only the quality of different approximations, but also different functional network models. The description length measure can be calculated for any model. In addition, it is used to compare models with different parameters, because it has a penalty term for overfitting. Moreover, it is distribution independent. This makes the minimum description length a

convenient method for solving the model selection problem. Accordingly, the best functional network model for a given problem corresponds to the one with the smallest description length value. To achieve this goal, the following methods can be used:

**The Exhaustive Search:** This method computes the MDL measure for all possible models, and it chooses the one leading to the smallest value of the error measure. The obvious shortcoming of this method is its computational complexity.

**The Backward-Forward method:** The backward process starts with the complete model with all parameters, and it sequentially removes the one leading to the smallest value of the MDL measure, repeating the process until there is no further improvement in the measure. Next, the forward process is applied, but starting from the final model of the backward process, and it sequentially adds the one variable that leads to the smallest value of MDL measure, repeating the process until there is no further improvement in the measure. This process is repeated until there is no further improvement in MDL measure is obtained, whether by removing or by adding a single variable.

**The Forward-Backward method:** The forward process starts with all models of a single parameter, and it selects the one leading to the smallest value of  $L(x)$ . Next, it incorporates one more parameter with the same criterion, and the process continues until there is no further improvement in  $L(x)$  can be obtained by adding an extra parameter to the previous model. Then the backward process is applied, but starting from the final model of the forward process, and sequentially removing the one variable that is leading

to the smallest value of MDL, repeating the process until no improvement in  $L(x)$  is possible. The double process is repeated until no further improvement in  $L(x)$  is obtained, whether by adding or removing a single variable.

## 4.7 Estimation of Porosity and Water Saturation With FN

### 4.7.1 Problem Statement

The main task in this problem is to estimate porosity ( $\phi$ ) and water saturation ( $S_w$ ) (refer to as responses) from well logs data, (refer to as predictor variables), while minimizing the error as much as possible using a selected criteria such as mean square error.

Mathematically, these relationships can be represented as:

$$S_w = f_1(x_1, \dots, x_5) \quad \text{and} \quad \phi = f_2(x_1, \dots, x_6)$$

where

$f_1(\cdot)$  &  $f_2(\cdot)$  are models

$x_i$  is the predictor variables from well logs

$S_w$  and  $\phi$  are estimated water saturation and porosity respectively.

However, since a general procedure is presented, we can write a general representation as:

$$\hat{y} = F(x_1, \dots, x_n)$$

where  $\hat{y}$  is the response,  $F$  is the model, and  $x_i$  are the predictors.

The objective is to minimize  $[\frac{1}{n} \sum_{i=1}^n \{y_i - \hat{y}_i\}^2]$ , which is the mean square of the error.

#### 4.7.2 Functional Networks Initial Architecture

The first step in functional networks is the specification of the initial topology which is problem driven. However, since there is no idea of the problem domain, models such as associative and uniqueness (Castillo et al. 1992) were first tried without any encouraging result. A recommended generalized model of the form (4.10) for this kind of situation was then used:

$$y = \sum_{r_1=1}^{q_1} \dots \sum_{r_k=1}^{q_k} C_{r_1, \dots, r_k} \varphi_{r_1}(x_1) \dots \varphi_{r_k}(x_k) \quad (4.10)$$

where  $C_{r_1, \dots, r_k}$  are unknown parameters (coefficients) and the set of functions  $\varphi_s = \{\varphi_{r_s}(x_s), r_s = 1, 2, \dots, q_s\}, s = 1, 2, \dots, k$  are linearly independent. For simplicity, Figure 5.1 shows the corresponding generalized model in equation 4.10 with two input variables.



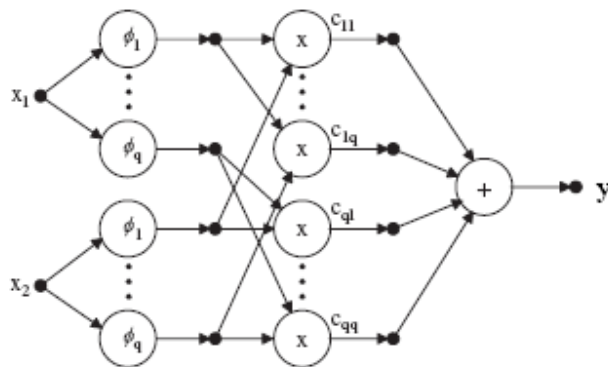


Figure 5.1: Generalized functional network in equation 4.10 with two inputs

### 4.7.3. Uniqueness of Representation

After selection of the model, the uniqueness of representation needs to be checked before the neural functions are learned from data. This uniqueness property is very important for some estimation methods that solve systems of equations. To check for the uniqueness of generalized model in equation 4.10, we assumed there are two sets of parameters  $C_{r_1, \dots, r_k}$  and  $C_{r_1, \dots, r_k}^*$  such that

$$\sum_{r_1=1}^{q_1} \dots \sum_{r_k=1}^{q_k} C_{r_1, \dots, r_k} \varphi_{r_1}(x_1) \dots \varphi_{r_k}(x_k) = \sum_{r_1=1}^{q_1} \dots \sum_{r_k=1}^{q_k} C_{r_1, \dots, r_k}^* \varphi_{r_1}(x_1) \dots \varphi_{r_k}(x_k) \quad (4.11)$$

Then, this can be written as

$$\sum_{r_1=1}^{q_1} \dots \sum_{r_k=1}^{q_k} (C_{r_1, \dots, r_k} - C_{r_1, \dots, r_k}^*) \varphi_{r_1}(x_1) \dots \varphi_{r_k}(x_k) = 0 \quad (4.12)$$

Since the family of functions  $\varphi_s$  are linearly independent, then  $C_{r_1, \dots, r_k} - C_{r_1, \dots, r_k}^* = 0$  for all  $r_1, r_2, \dots, r_k$ . This implies  $C_{r_1, \dots, r_k} = C_{r_1, \dots, r_k}^*$  for all  $r_1, r_2, \dots, r_k$ . Hence equation (4.10) is a unique representation.

#### 4.7.4 Simplification of the Model

For the problem concerned in this study, the model represented by equation 4.10 was simplified further to reduce the complexity of terms to be estimated, by assuming that all the coefficients of the cross-multiplication terms between the function of different variables other than one is zero. This reduces equation 4.10 to the form

$$y = h_1(x_1) + h_2(x_2) + \dots + h_k(x_k) \quad (4.13)$$

where  $h(x)$  denotes the sum of the function basis for each predictor variable with coefficients to be learned. This can be written as:

$$y = \sum_{r_1=1}^{q_1} C_{r_1} \varphi_{r_1}(x_1) + \sum_{r_2=1}^{q_2} C_{r_2} \varphi_{r_2}(x_2) + \dots + \sum_{r_k=1}^{q_k} C_{r_k} \varphi_{r_k}(x_k) \quad (4.14)$$

This can be written in condensed form as:

$$y = \sum_{r=1}^{q_s} C_{kr} \varphi_{kr}(x_k), \quad k = 1, 2, 3, \dots, s \quad (4.15)$$

Or simply as:

$$y = \sum_{k=1}^s \sum_{r=1}^{q_k} C_{kr} \phi_{kr}(x_k) \quad (4.16)$$

Figure 5.2 shows the functional networks with two predictor variables for equations 4.1)-4.1). The uniqueness of equation (4.16) follows that of the original generalized model, which is means this simplified model is also unique.

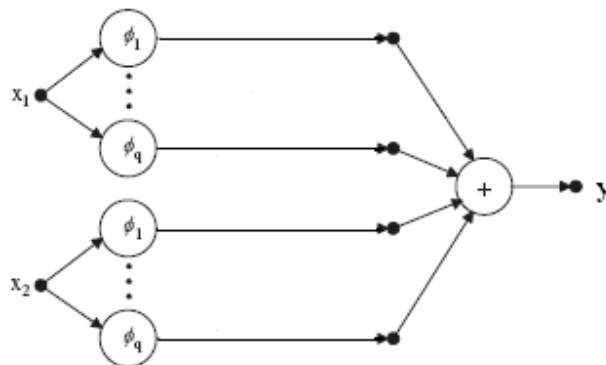


Figure 5.2: Simplified generalized model with  $k=2$  and  $q_1=q_2=q$

Note that without loss of generality, since the first term of all the basis functions starts with a constant and the sum will also give a constant, the constant term can be included only in the  $h_1(x_1)$ .

### 4.7.5 Learning procedure for the simplified model

Let  $D = \{y_i, x_{i1}, x_{i2}, \dots, x_{ik}; i = 1, 2, \dots, n\}$  denotes the available data set, which consists of  $n$  observations of the response (output) and  $k$  predictor variables (inputs). Then, for the  $i^{\text{th}}$  observation in  $D$ , we have for the general case of the simplified model as:

$$y_i = \sum_{k=1}^s \sum_{r=1}^{q_k} C_{kr} \varphi_{kr}(x_{ik}), \quad i = 1, 2, \dots, n \quad (4.17)$$

Since the combinations of linearly independent functions ( $\varphi_{kr}$ ) are known, our goal is only to learn the coefficients  $C_{kr}$ . Examples of linearly the independent function used in this study include:

1). Polynomial family

$$\varphi = \{1, x, x^2, \dots, x^q\}$$

2). Exponential family

$$\varphi = \{1, ex, e^{-x}, e^{2x}, e^{-2x}, \dots, e^{qx}, e^{-qx}\}$$

3). Fourier family

$$\varphi = \{1, \sin(x), \cos(x), \sin(2x), \cos(2x), \dots, \sin(qx), \cos(qx)\}$$

4). Logarithms function

$$\varphi = \{1, \log(x+2), \log(x+3), \dots, \log(x+q)\}$$

Assuming that the model in equation 4.17 provides an acceptable approximation to the relation between the response and the predictor variables, then we can write 4.17 in matrix form as:

$$\hat{Y} = W\beta \quad (4.18)$$

The error  $\varepsilon$  between the real output  $Y$  and the estimated output  $\hat{Y}$  is thus given by:

$$\varepsilon = Y - \hat{Y} = Y - W\beta \quad (4.19)$$

Using the least square optimization technique, we can learn the coefficients by minimizing the error ( $\varepsilon$ ).

$$L = \varepsilon^t \varepsilon = (Y - W\beta)^t (Y - W\beta) \quad (4.20)$$

$$\beta = (W^t W)^{-1} W^t Y \quad (4.21)$$

Solving equation 4.2 gives the solution to the unknown coefficients,  $\beta$  in equation 4.1) or coefficients  $C_{kr}$  in equation 4.1. The learning algorithm is summarized in Figure 4.4.

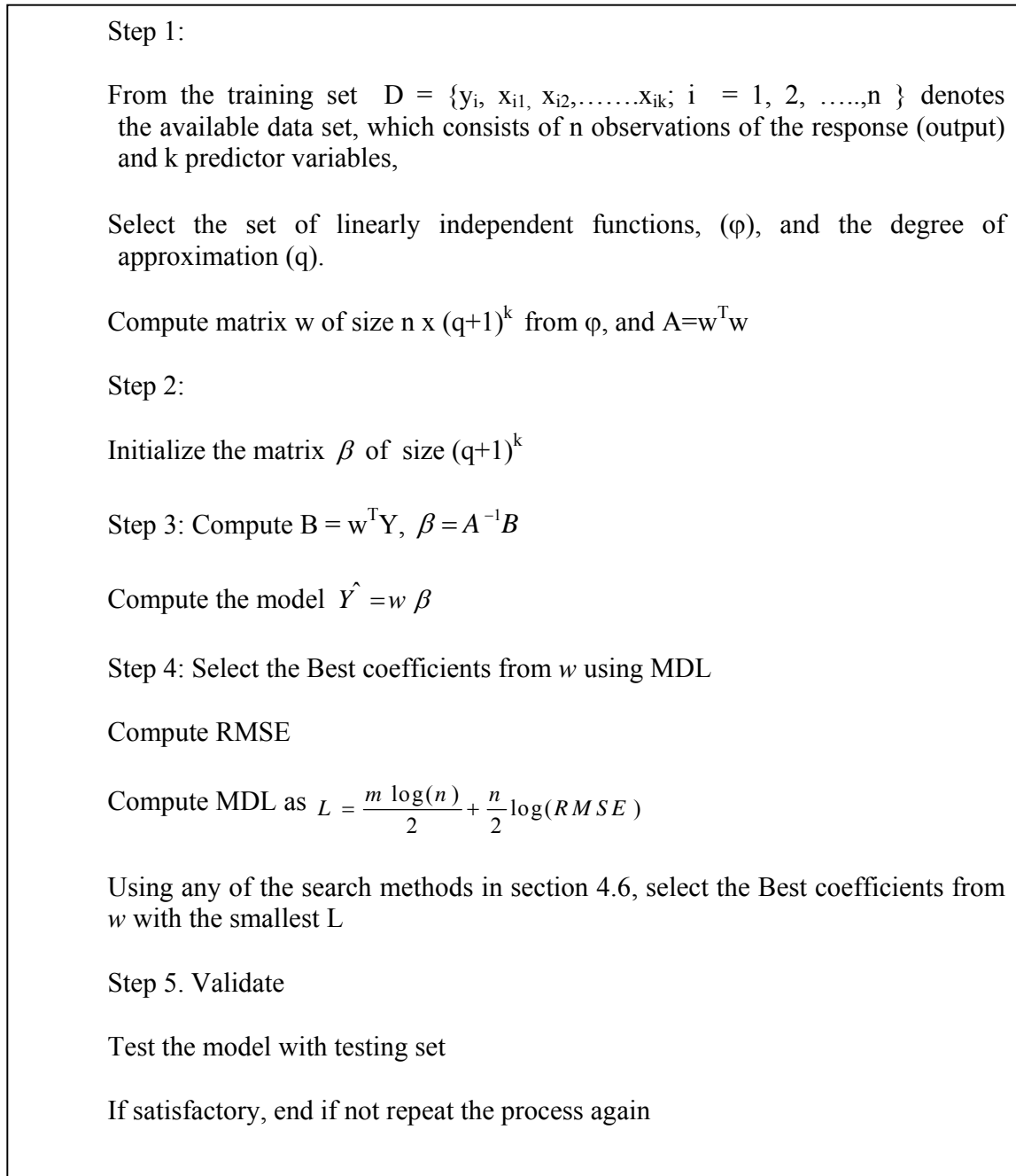


Figure 5.4 Learning algorithm for porosity and water saturation functional networks model

## CHAPTER FIVE

# PREDICTION OF POROSITY AND WATER SATURATION

### 5.1 Overview

The main contribution of this thesis is to investigate and develop some recent advances in neural networks (such as functional networks, cascaded correlation neural networks, and polynomial networks) to estimate porosity ( $\phi$ ) and water saturation ( $S_w$ ) for reservoir rock characterization.

In this chapter, we present the implementation process for estimation of formation porosity ( $\phi$ ) and water saturation ( $S_w$ ) from well logs based on the framework discussed in the chapter 3, 4 and 5. Empirical results are presented from the implementation of functional networks (FN) and other methods, including cascaded correlation neural networks (CCNN), general regression neural networks (GRNN), polynomial networks (POLYNET) as well as standard feed forward neural networks.

## 5.2 Data acquisition and Implementation process

This study uses a data set from the Middle Eastern region which consists of well logs, core porosity and water saturation from 206 and 211 observations of two wells labeled A and B of respectively. We investigate these data sets, and we perform the required statistical analysis plots to determine the hidden pattern of the relationship among the actual outputs and the provided input features. This helps us to get more knowledge about all the data. The two wells were combined together and divided randomly into two sets, training and testing set of 70% and 30% respectively. The training set was then used to build the model while the testing set was used to evaluate the predictive capability of the model. For investigating prediction of porosity, six well logs namely; Sonic log (DT), Neutron log (NPHI), Density log (RHOB), Gamma Ray (GR), Resistivity (Rt), and Photoelectric Factor log (PEF) were used as inputs to all the networks. Five inputs namely Neutron log (NPHI), Density log (RHOB), Gamma Ray (GR), Resistivity (Rt), and Photoelectric Factor log (PEF) were used for the prediction of water saturation. Statistical descriptions of the data are given in Tables 5.1 to 5.3.

In order to make sure the predictors variables are independent from measurement units (since they are of different units), the predictor variables (inputs) were normalized to interval [0, 1] by using the formula:

$$x_i^{new} = \frac{x_i^{old} - \min(x_i)}{\max(x_i) - \min(x_i)}; i = 1, 2, \dots, n \quad (5.1)$$



To determine the performance and accuracy of the models, we made use of some the statistical quality measures mentioned in Chapter two, namely: correlation coefficient, absolute relative errors, average absolute error, and root mean squared error. A good model should have a high correlation coefficient (CC) and a low root mean square error (RMSE), an average absolute error (EA), and average absolute relative errors (ER).

Table 5.1: Statistics of the combined wells A and B

<b>LOG TYPE (Predictors)</b>	<b>MIN</b>	<b>MAX</b>	<b>AVERAGE</b>	<b>STDEV</b>
Sonic Travel time (DT)	50.00948	85.73618	64.39576	9.759414
Neutron porosity	0.011684	0.265368	0.122452	0.063873
Bulk density (RHOB)	2.20866	2.822594	2.483926	0.137241
Gamma Ray	3.487981	27.01702	11.65997	3.963161
Resistivity	1.592936	165.3348	22.21751	19.8768
Photoelectric Factor (PEF)	2.693318	5.512973	4.272576	0.516099

Table 5.2: Statistics of well A

<b>LOG TYPE (Predictors)</b>	<b>MIN</b>	<b>MAX</b>	<b>AVERAGE</b>	<b>STDEV</b>
Sonic Travel time (DT)	50.00948	83.64293	65.11951	10.02312
Neutron porosity	0.011684	0.233358	0.121226	0.063654
Bulk density (RHOB)	2.217509	2.732041	2.467093	0.14167
Gamma Ray	3.487981	22.17721	11.01905	3.724572
Resistivity	5.962086	165.3348	23.35686	21.47114
Photoelectric Factor (PEF)	2.88618	5.39553	4.375203	0.47247

Table 5.3: Statistics of well B

<b>LOG TYPE (Predictors)</b>	<b>MIN</b>	<b>MAX</b>	<b>AVERAGE</b>	<b>STDEV</b>
Sonic Travel time (DT)	50.02477	85.73618	63.68916	9.465285
Neutron porosity	0.025846	0.265368	0.123649	0.064215
Bulk density (RHOB)	2.20866	2.822594	2.50036	0.131033
Gamma Ray	5.546964	27.01702	12.2857	4.095807
Resistivity	1.592936	115.2886	21.10515	18.16862
Photoelectric Factor (PEF)	2.693318	5.512973	4.17238	0.537877

### 5.3 Experimental Results Using Functional Networks

The function network model described in section 5.4 was implemented by trying all basis functions given in section 5.5. First, different basis was used for each function in equation 5.4. Secondly, the same basis was used for all the functions ( $h_1(x_1), \dots, h_n(x_n)$ ) in order to select the one that will give best approximation. The coefficients of the network were optimized by the backward-forward search method by using the criterion of Minimum description length (MDL). This criterion determines the best coefficients contributing to the network, and it chooses the best functional basis (the one with lowest value of MDL as described in section 4.6). However, in both cases (estimation of porosity and water saturation), it is found that using different basis functions does not give good approximation compared to having same basis functions. Therefore, we present only the results second try (same basis functions).

#### 5.3.1 Porosity Estimation Results

The functional network model is able to predict formation porosity with the Fourier basis function as the best selected basis, producing a root mean square error (RMSE) of 0.0293 and a correlation coefficient (CC) of 0.9158 for the training set, but RMSE of 0.0245 and CC of 0.9343 for the testing set. The results for the estimation are shown in Table 5.2 while Figures 5.1a and 5.1b show the performance plot for the best selected basis function (Fourier basis). Table 5.2 shows that the Fourier basis function is the best basis, as it achieved the highest testing set correlation and lowest RMSE. For the

sake of simplicity, performances of other basis functions are shown in appendix A. The results show that there is a good matching between the core porosity and the estimated porosity. Equation 5.2 shows the relationship between the predictor variable and the core porosity for the best model.

$$y = h_1(x_1) + h_2(x_2) + h_3(x_3) + h_4(x_4) + h_5(x_5) + h_6(x_6) \quad (5.2)$$

where

$$h_1(x_1) = -0.42587 + 0.73647 \sin(2x_1) + 0.40639 \cos(2x_1) - 0.36971 \sin(3x_1)$$

$$h_2(x_2) = 0.33693 \sin(x_2) - 0.10597 \sin(2x_2)$$

$$h_3(x_3) = 1.638 \sin(x_3) - 1.4767 \sin(2x_3) + 0.43745 \sin(3x_3)$$

$$h_4(x_4) = 0.023313 \cos(2x_4)$$

$$h_5(x_5) = 0.028385 \sin(2x_5)$$

$$h_6(x_6) = 0.029636 \sin(2x_6)$$

Table 5.4 RMSE and correlation coefficient (CC) of different basis functions used for Porosity estimation

MODEL	Training Set		Testing Set	
	RMSE	CC	RMSE	CC
<b>Function</b>				
Fourier	0.0293	0.9158	0.0245	0.9343
Exponential	0.0292	0.9159	0.0248	0.9332
Polynomial	0.0300	0.9158	0.0249	0.9314
Logarithm	0.0302	0.9100	0.0247	0.9328
Poly&Log	0.0294	0.9148	0.0247	0.9331

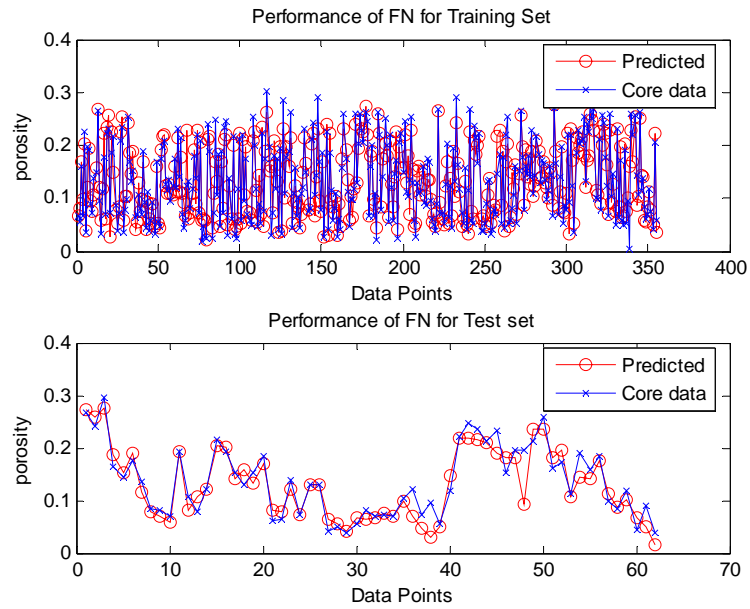


Figure 5.1a: Performance plot for estimated porosity using FN

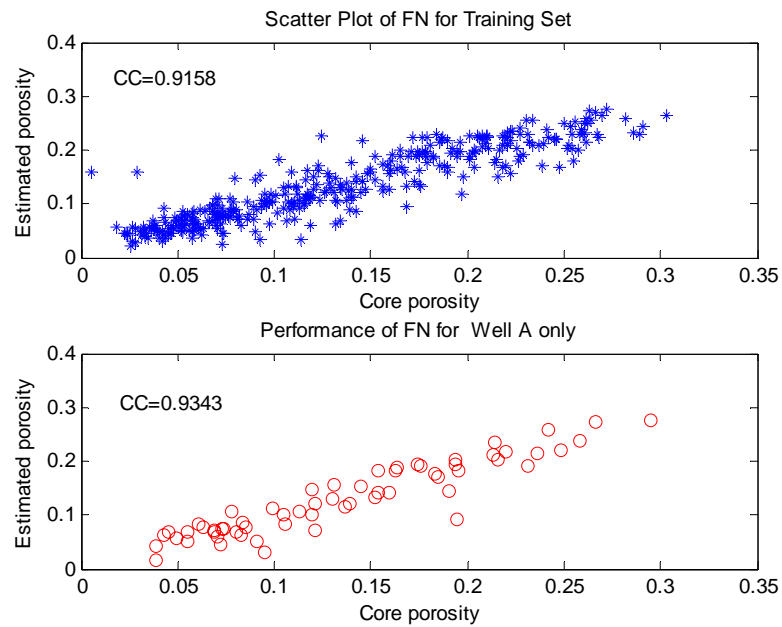


Figure 6.1b: Scatter plot for estimated porosity versus core porosity using FN

### 5.3.2 Water saturation Estimation Results

Similarly, water saturation is determined with the same model as presented in section (5.4) using five log inputs (NPHI, RHOB, GR, Rt, and PEF). The model is able to predict water saturation with logarithm function as the best selected basis function, achieving a root mean square error, (RMSE) of 0.1143 and correlation coefficient (CC) of 0.9491 for the training set, but RMSE of 0.0805 and CC of 0.9743 for the testing set. The results for the estimation are shown in Table 5.3. Figures 5.2a & 5.2b show the performance plot for the best selected basis within the function basis. Also, performances of other bases were shown in appendix for simplicity. The results also show that there is a good match between the core and estimated water saturation. Equation 5.3 shows the relationship between the predictor variable and the water saturation for the best selected model.

Table 5.5: RMSE and CC of different basis functions used for water estimation

MODEL	Training Set		Testing Set	
	RMSE	CC	RMSE	CC
Fourier	0.1076	0.9549	0.0866	0.9730
Exponential	0.1075	0.9549	0.0864	0.9702
Polynomial	0.1150	0.9483	0.0827	0.9730
Logarithm	0.1143	0.9491	0.0805	0.9743

$$y = h_1(x_1) + h_2(x_2) + h_3(x_3) + h_4(x_4) + h_5(x_5) \quad (5.3)$$

where

$$h_1(x_1) = 974.02 - 2549 \log(x_1 + 2) + 9745.6 \log(x_1 + 3) - 7942 \log(x_1 + 4)$$

$$h_2(x_2) = -391042 \log(x_2 + 3) + 512.75 \log(x_2 + 4)$$

$$h_3(x_3) = 1441.6 \log(x_3 + 2) - 5683.3 \log(x_3 + 3) + 4709.4 \log(x_3 + 4)$$

$$h_4(x_4) = -2053.4 \log(x_4 + 2) + 7943.3 \log(x_4 + 3) - 6517 \log(x_4 + 4)$$

$$h_5(x_5) = -2.0153 \log(x_5 + 2)$$

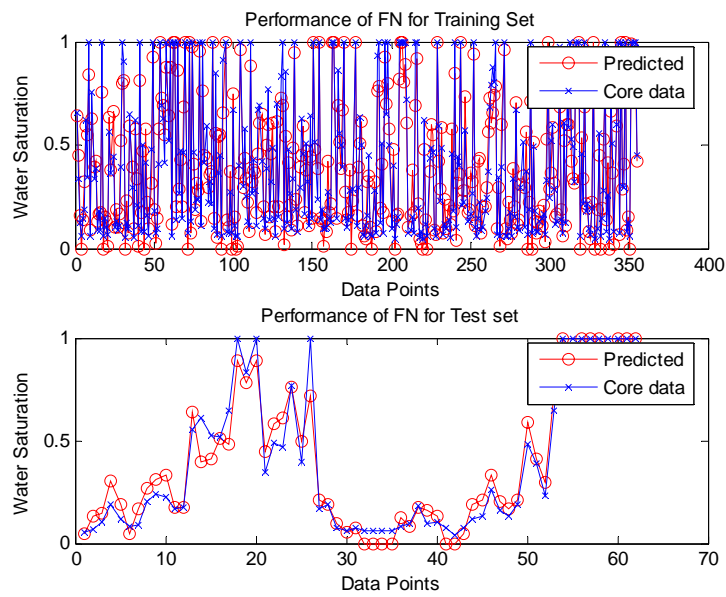


Figure 5.2a: Performance plot for estimated water saturation using FN

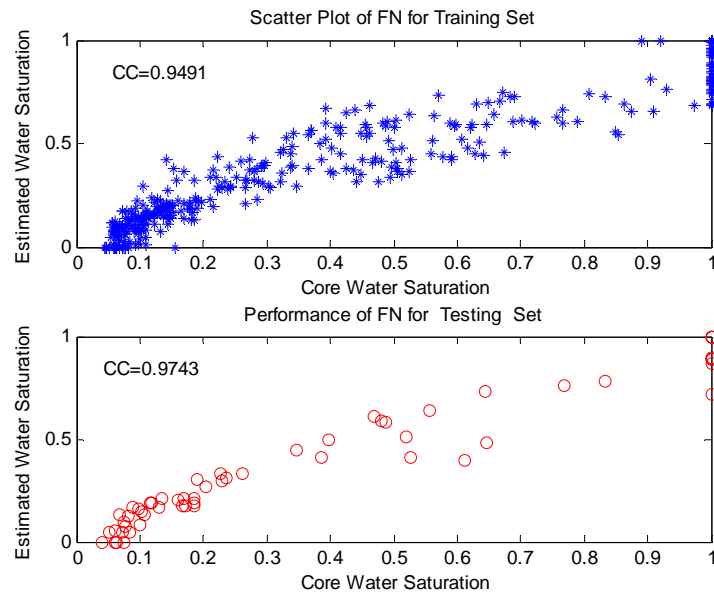


Figure 5.2b: Scatter plot for estimated water saturation versus water saturation using FN

## 5.4 Experimental Results Using CCNN

For the prediction of porosity from well logs by using CCNN, we tried different numbers of neurons in the hidden layer and we found the optimum to be a single hidden layer with five neurons. The hidden layer used the Gaussian activation function, while the output layer used the sigmoid function and it trained up to 250 epochs. The water saturation network, on the other hand, used ten neurons in the hidden layer with the Gaussian activation function and the linear function at the output layer. The results are shown in Figures 5.3 and 5.4 for porosity and water saturation respectively. Table 5.4 summarizes the result from the networks.



Table 5.4: Models performance for porosity and water saturation using CCNN

MODEL	Network	TRAINING				TESTING			
		$E_A$	$E_R$	RMS	CC	$E_A$	$E_R$	RMS	CC
Porosity	6-5-1	0.0200	0.1853	0.0257	0.9268	0.0188	0.2041	0.0259	0.9260
Water	5-10-1	0.0534	0.5261	0.0866	0.9706	0.0638	0.1526	0.1048	0.9596

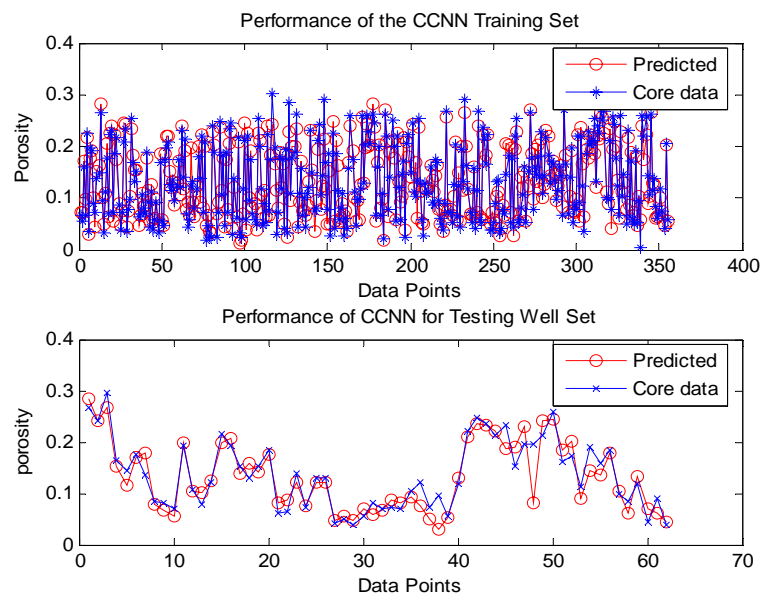


Figure 5.3a: Performance plot for estimated porosity using CCNN

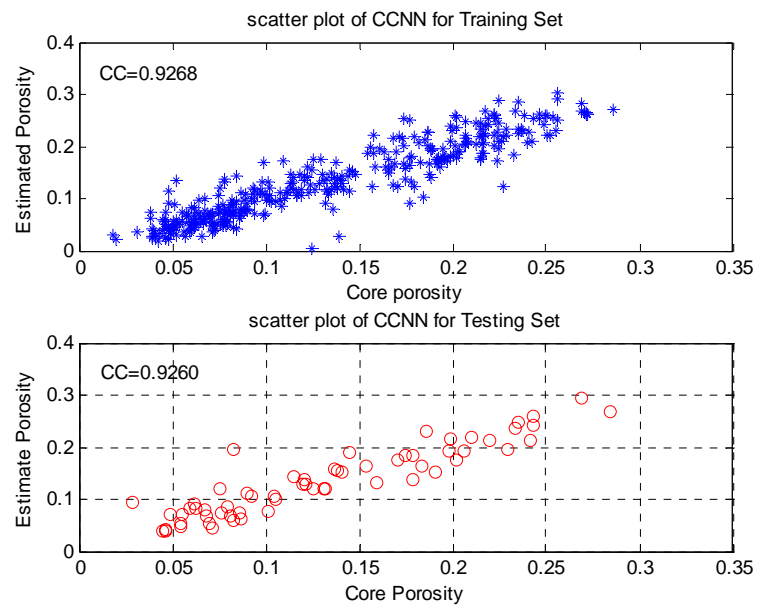


Figure 5.3b: Scatter plot for estimated porosity versus Core porosity using CCNN

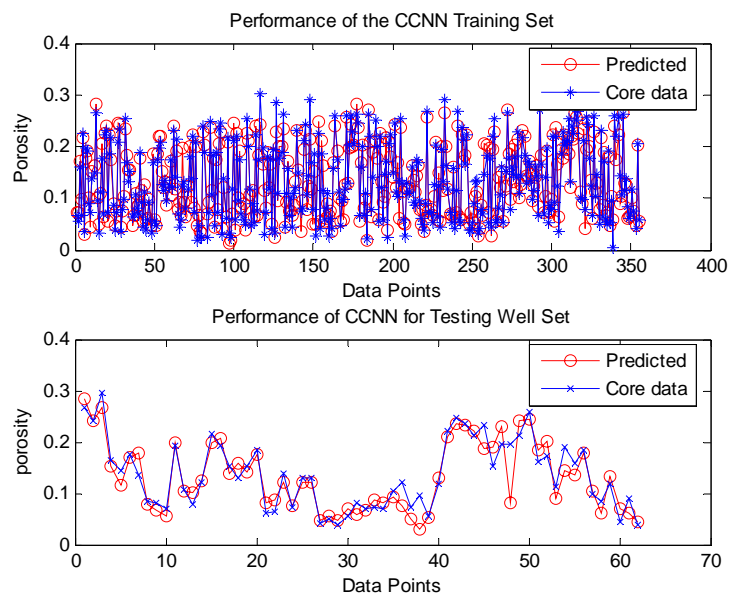


Figure 5.4a: Performance plot for estimated water saturation using CCNN

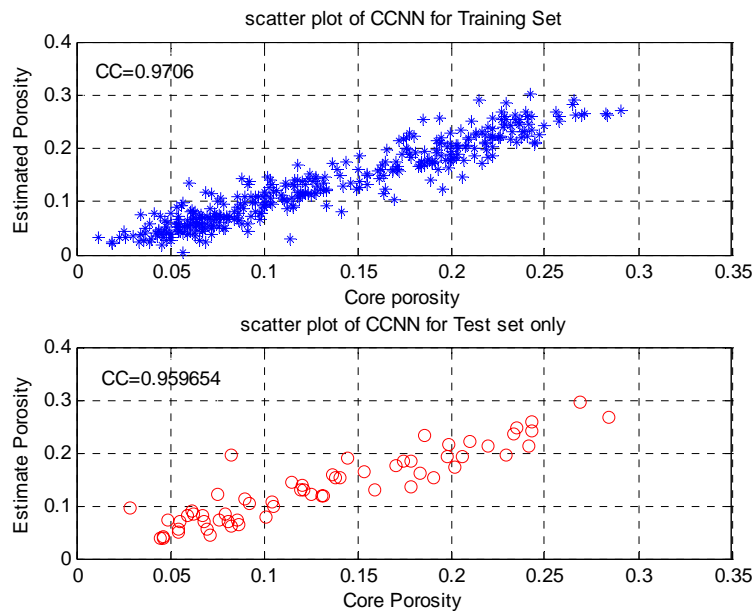


Figure 5.4b: Scatter plot for estimated water saturation versus water saturation using CCNN

## 5.5 Experimental Results Using GRNN

GRNN was also designed to determine porosity and water saturation as done in previous sections. The network structure was determined after a number of trials of different numbers of spread centers. A spread of 0.07 was selected for porosity prediction, while a spread of 0.055 was chosen for prediction of water saturation. The summary of the results is shown in Table 5.5 and illustrated in Figures 5.5 and 5.6 below.

Table 5.5: Models performance for porosity and water saturation using GRNN

MODEL	TRAINING				TESTING			
	$E_A$	$E_R$	RMS	CC	$E_A$	$E_R$	RMSE	CC
Porosity	0.0101	0.1120	0.0145	0.97323	0.0221	0.2322	0.0329	0.9003
Water	0.0270	.08158	0.0476	0.9907	0.0569	0.2273	0.0858	0.9716

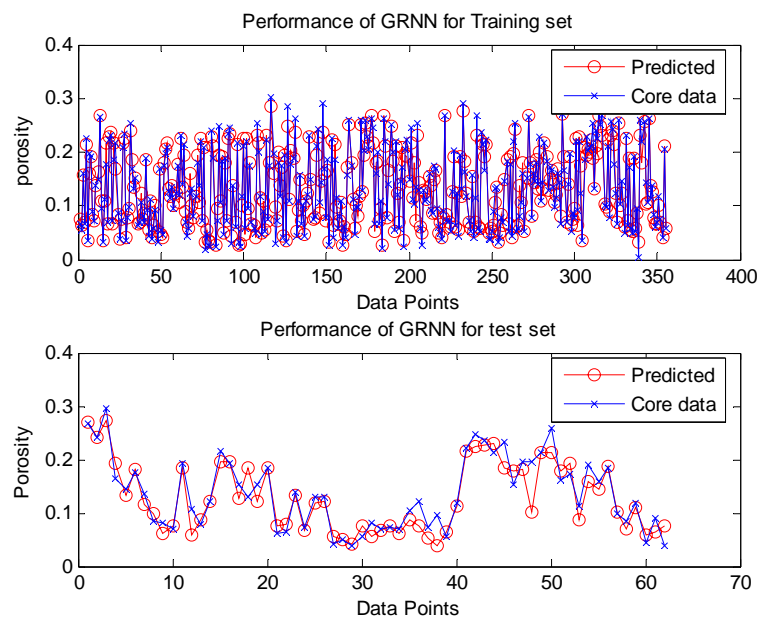


Figure 5.5a: Performance plot for estimated porosity using GRNN

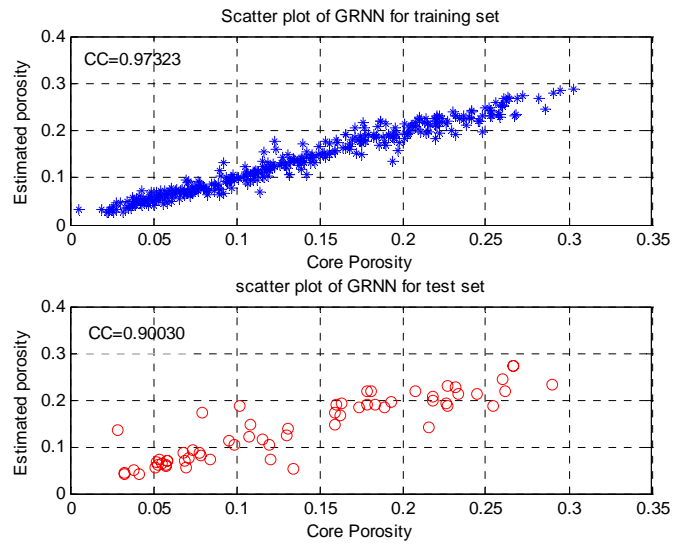


Figure 5.5b: Scatter plot for estimated porosity versus Core porosity using GRNN



Figure 5.6a: Performance plot for estimated water saturation using GRNN

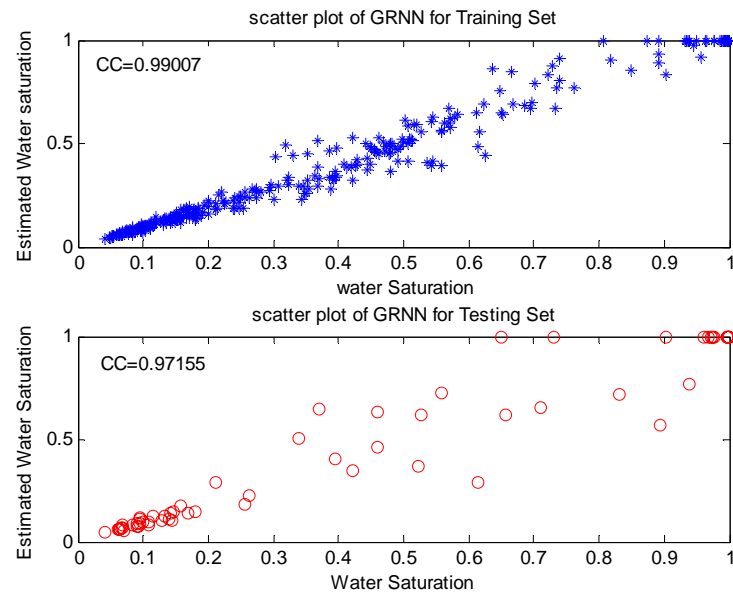


Figure 5.6b: Scatter plot for estimated water saturation versus water saturation using GRNN

## 5.6 Experimental Results Using Polynet

For prediction of porosity and water saturation by using polynomial networks, two hidden layers were used. For porosity networks, the first layer and second layer were constructed by only 2 nodes and 1 node respectively. In water saturation network 8 nodes were used in the first layer and 2 nodes were found to be appropriate for the second layer. Table 5.6 summarizes the results and Figures 5.7 and 5.8 show the scatter plots.

Table 5.6: Models performance for porosity and water saturation using PolyNet

MODEL	Network	TRAINING				TESTING			
		$E_A$	$E_R$	RMS	CC	$E_A$	$E_R$	RMS	CC
Porosity	6-2-1--1	0.0221	0.2955	0.0311	0.9040	0.0182	0.1553	0.0245	0.9329
Water	5-8-2-1	0.0885	0.3480	0.1194	0.9346	0.0758	0.3338	0.1013	0.9593

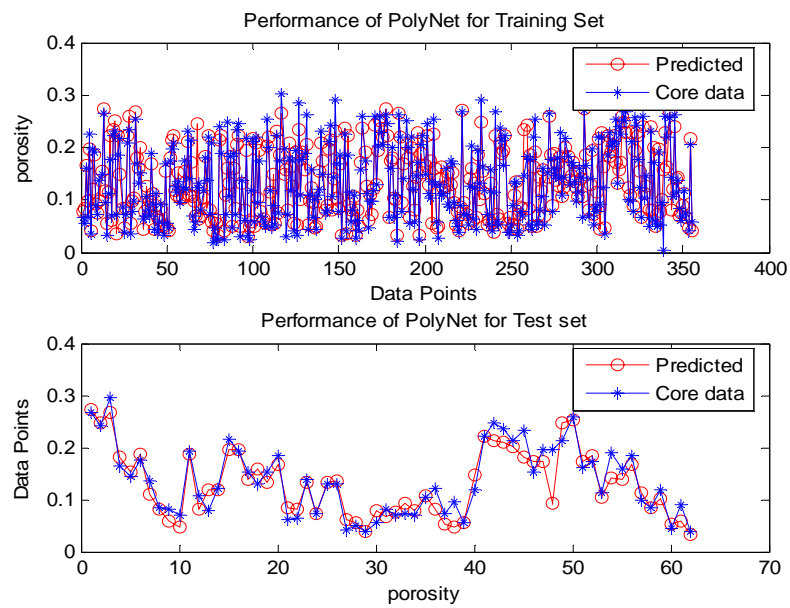


Figure 5.7a: Performance plot for estimated porosity using PolyNet

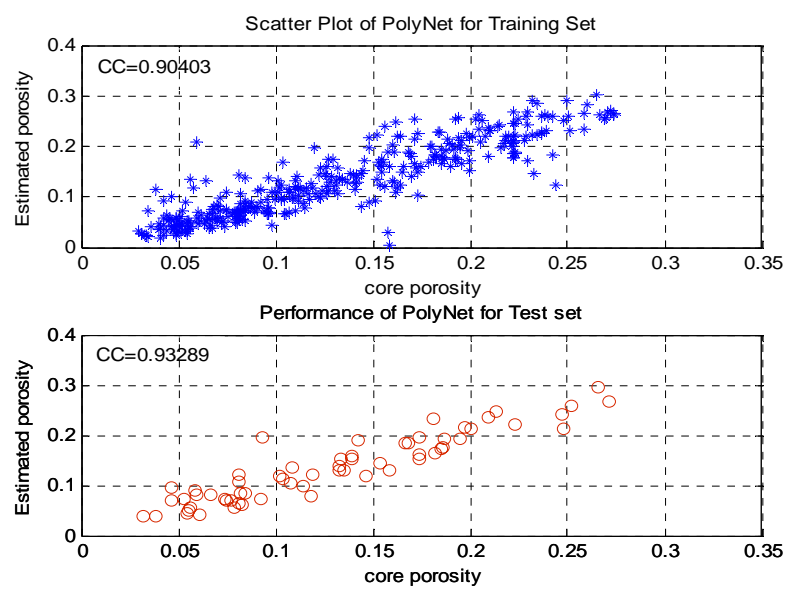


Figure 5.7b: Scatter plot for estimated porosity versus Core porosity using PolyNet

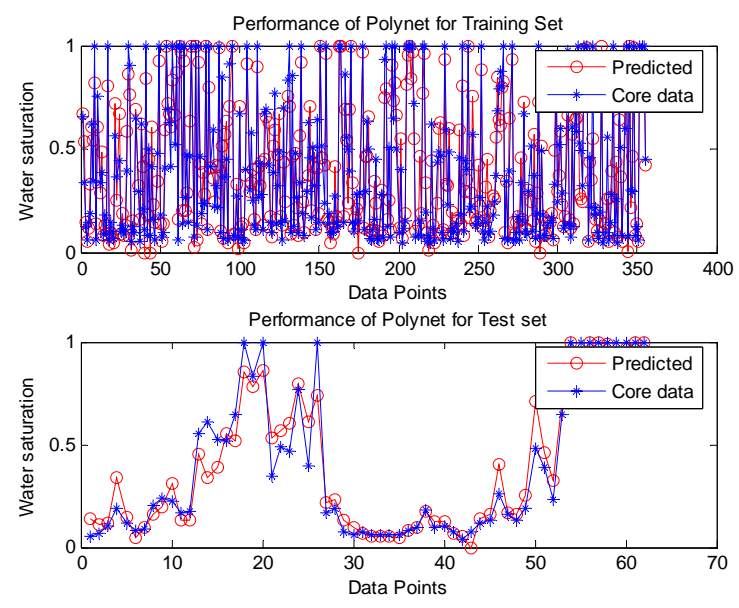


Figure 5.8a: Performance plot for estimated water saturation using ploynet



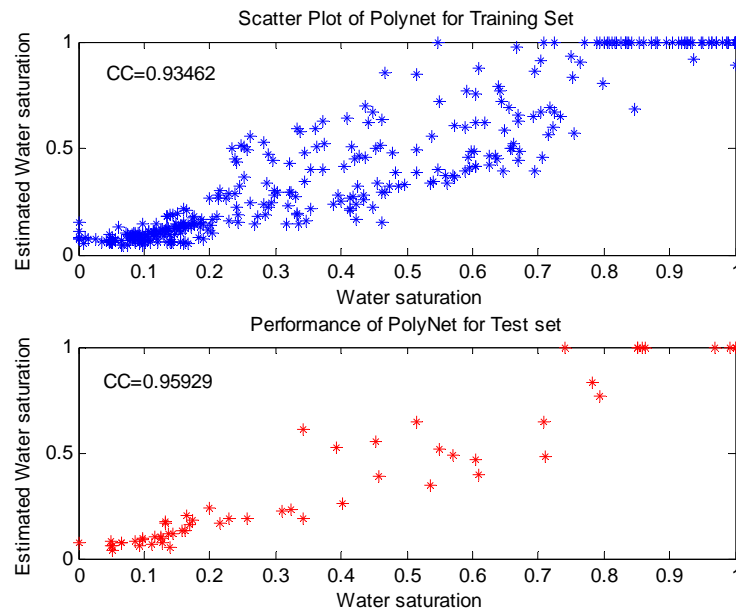


Figure 5.8b: Scatter plot for estimated water saturation versus water saturation using Polynet

## 5.7 Experimental Results Using Feedforward Neural Networks

To make clear the advantages or disadvantages of the models above for standard feed forward neural networks, we also developed a feed forward neural network, FFNN, for the same purpose. After some trial-and-error, and experiences derived from cascaded correlation neural networks, a network of the same structure as CCNN was used but with the LM training algorithm. The hidden layer used tan-sigmoid while the output layer used log-sigmoid. The network was trained with 500 epochs. Table 5.7 presents the summary of the result obtained and Figures 5.9 and 5.10 show the plot with core values.

Table 5.7: Models performance for porosity and water saturation using FFNN

MODEL	Network	TRAINING				TESTING			
		$E_A$	$E_R$	RMSE	CC	$E_A$	$E_R$	RMS	CC
Porosity	6-5-1	0.0186	0.2599	0.0265	0.9316	0.0210	0.2287	0.0286	0.9043
Water	5-10-1	0.0222	0.1016	0.0353	0.9946	0.0495	0.1567	0.0807	0.9682

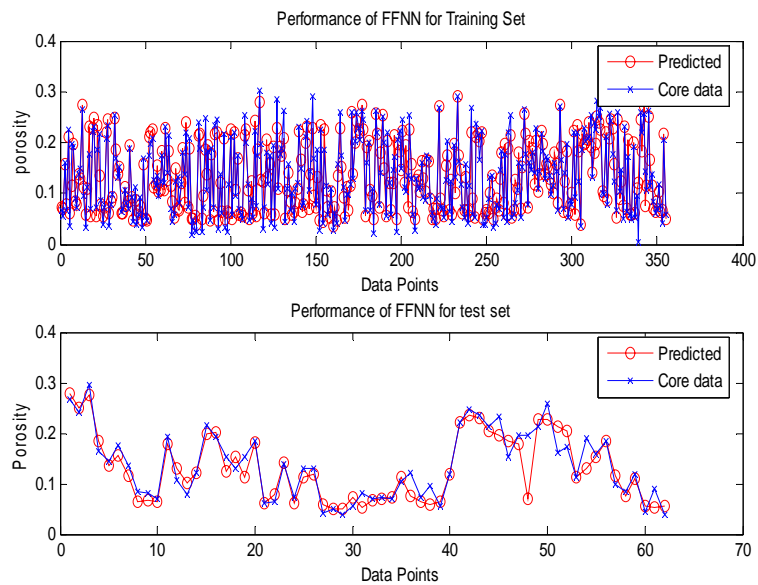


Figure 5.9a: Performance plot for estimated porosity using FFNN

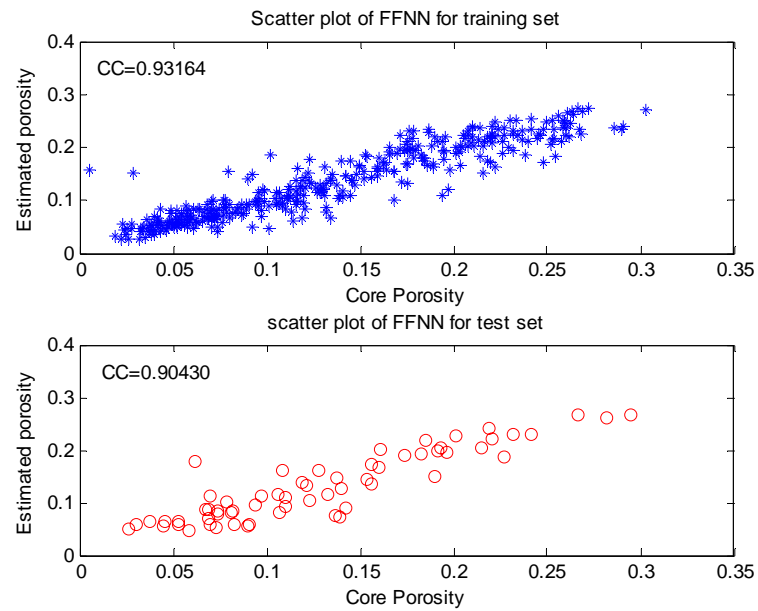


Figure 5.9b: Scatter plot for estimated porosity versus Core porosity using FFNN

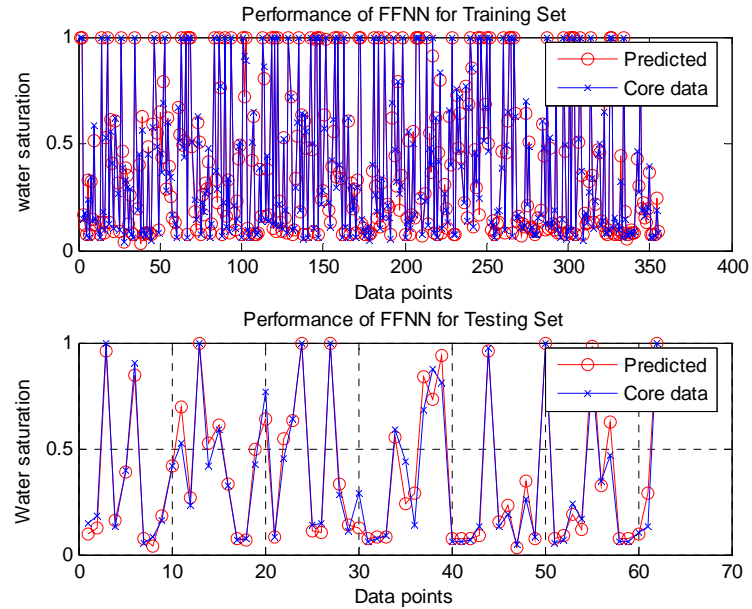


Figure 5.10a: Performance plot for estimated water saturation using FFNN

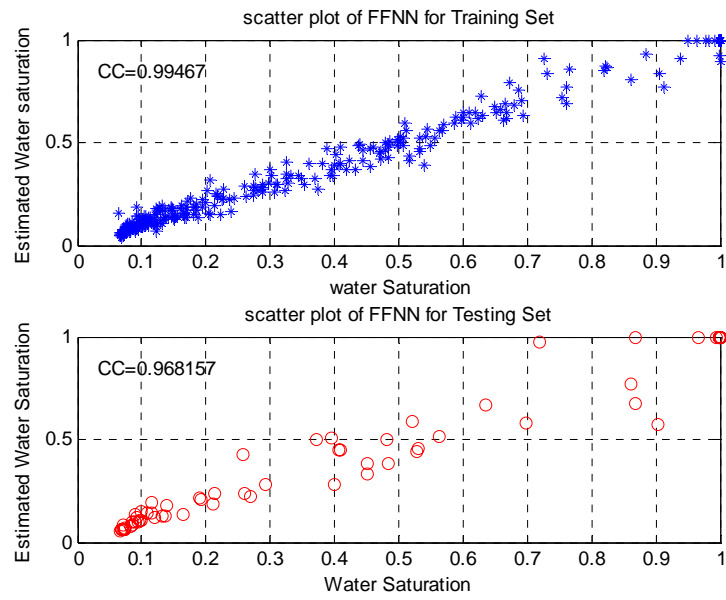


Figure 5.10b: Scatter plot for estimated water saturation versus water saturation using FFNN

## **CHAPTER SIX**

# **PERFORMANCE ANALYSIS AND COMPARATIVE STUDIES**

### **6.1 Overview**

The first objective of this thesis is the development and investigation of functional networks and other forms of recent advances in neural networks for the estimation of reservoir rock porosity and water saturation. The other objective is to compare their performance. In this regard, we present the comparison between the techniques presented in Chapter six. Furthermore, we also present the comparisons of the best model(s) with empirical models (equations 2.1 and 2.2).

We compare the performance and accuracy of all the models presented in chapter six in four stages. The first stage involved the performance of all the models with the combined well A and B as presented in chapter six. In the second stage, each well was run on the models to determine further the performance and the robustness of each model. The third stage involved comparing the best selected model(s) from the first and second stages with the empirical models discussed in Chapter two (equations 2.1 and 2.2). In the last stage, a test well T2, which did not involve in the training of the model,

was tested with the best selected model(s) from the first and second stages. Results from these four stages are discussed below.

In the first stage, the results from the models as presented in chapter six are very close but the functional networks modeling scheme outperforms all other techniques. The model (functional network) shows a high accuracy in predicting the porosity and water saturation values. It achieves the lowest network weights (N.wgt) with the lowest root mean square error, and the highest correlation coefficient for the testing case (Table 6.1). Functional network also shows relatively low execution time (Ex. Time). Scatter plots (Figure 6.1 and Figure 6.2) of the RMSE versus the CC for all the computational intelligence models is drawn to show the pictorial view of the performance. Each modeling scheme is represented by a symbol, and the best modeling scheme should appear in the upper left corner of the graph. We observe that the symbol corresponding to functional networks scheme falls in the upper left corner.

Table 6.1A: Models performance for porosity

MODEL	N.Wgt	Training			Testing		
		RMSE	CC	Ex. Time	RMS	CC	Ex. Time
FN	12	0.0293	0.9158	1.278	0.0245	0.9343	0.01592
FFNN	35	0.0265	0.9316	6.975	0.0286	0.9043	0.1643
CCNN	35	0.0257	0.9268	8.75	0.0259	0.9260	0.010
GRNN	350	0.0145	0.9732	0.529	0.0329	0.9003	0.0341
POLYNET	18	0.0311	0.9040	0.2103	0.0245	0.9328	0.00570

Table 6.1B: Models performance for water saturation

MODEL	N.Wgt	Training			Testing		
		RMSE	CC	Ex. Time	RMSE	CC	Ex. Time
FN	13	0.1143	0.9491	0.1809	0.0805	0.9743	0.01528
FFNN	60	0.03525	0.9946	7.7346	0.0807	0.9681	0.02258
CCNN	60	0.0866	0.9706	10.08	0.1048	0.9596	0.0089
GRNN	350	0.0476	0.9907	0.7482	0.0858	0.9716	0.04786
POLYNET	60	0.1194	0.9346	0.3036	0.1013	0.9592	0.003093

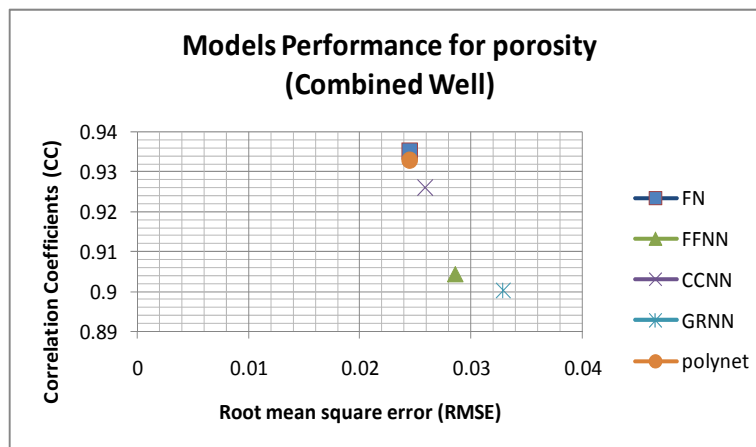


Figure 6.1: Scatter plot of CC versus RMSE for porosity prediction on testing case for all models

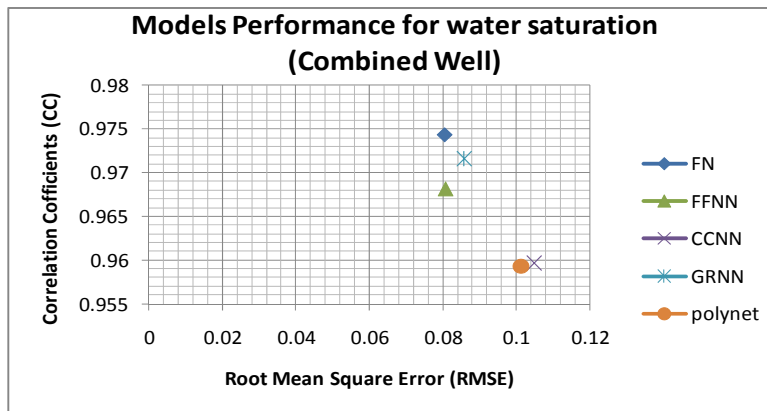


Figure 6.2: Scatter plot of CC versus RMSE for water saturation prediction on testing case for all models

In the second stage, we tested well A and B separately on all the models developed in the first stage. This was to test the robustness of the models, especially that of FN, which showed the best performance in the first stage. Here, it was observed generally that the performance of the models in well A is lower than that of well B especially in the prediction of porosity. The statistics of these wells were compared to those of the combined well, and it is found that the statistics of well B dominates the training data. This might be a reason for the better performance in well B. Although all the models show close performance, GRNN result takes a leading role in this stage. The results also show that GRNN is consistent. This indicates that GRNN is more robust than FN and other models. However, if we consider the GRNN model complexity from the network weights (N.Wgt) and the execution time (Ex. Time) to that of FN, there may be



need to make a compromise between accuracy and time complexity. The plot of each models are presented in appendix B. However, the scatter plots for the comparison are shown in Figures 6.3 to 6.6 and Tables 6.2 and 6.3 show the summary of the results.

Table 6.2: Models performance for porosity and water saturation for well A

MODEL	Porosity			Water saturation		
	RMSE	CC	Ex. Time	RMSE	CC	Ex. Time
FN	0.0399	0.8538	0.01844	0.1390	0.9532	0.0156
FFNN	0.0480	0.8589	0.02003	0.1849	0.9193	0.02702
CCNN	0.0357	0.8761	0.0042	0.1594	0.9505	0.0072
GRNN	0.0312	0.9048	0.066212	0.1031	0.9696	0.0873
POLYNET	0.0396	0.86873	0.00147	0.3368	0.92396	0.00912

Table 6.3: Models performance for porosity and water saturation for well B

MODEL	Porosity			Water saturation		
	RMSE	CC	Ex. Time	RMSE	CC	Ex. Time
FN	0.0194	0.9613	0.01628	0.0310	0.9211	0.02172
FFNN	0.0248	0.9406	0.0160	0.1014	0.9556	0.0274
CCNN	0.0241	0.9405	0.0064	0.1417	0.9136	0.00812
GRNN	0.0149	0.9776	0.05353	0.0895	0.9584	0.08914
POLYNET	0.022	0.94996	0.00171	0.1413	0.90785	0.00927

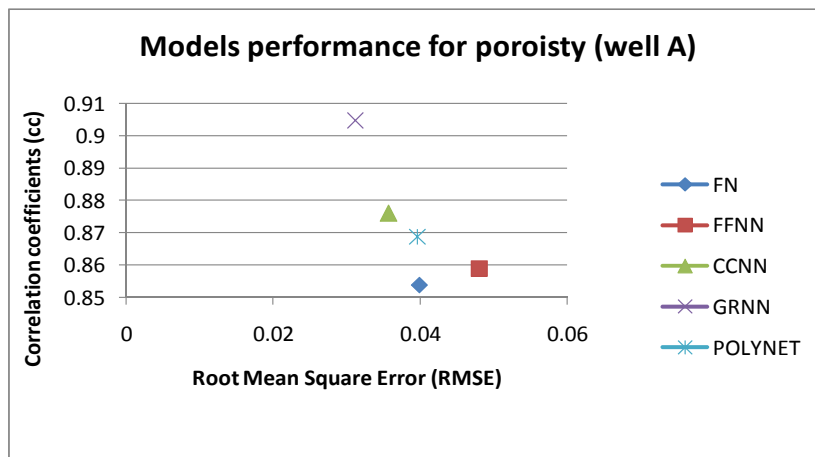


Figure 6.3: Scatter plot of CC versus RMSE for porosity prediction on well A for all models

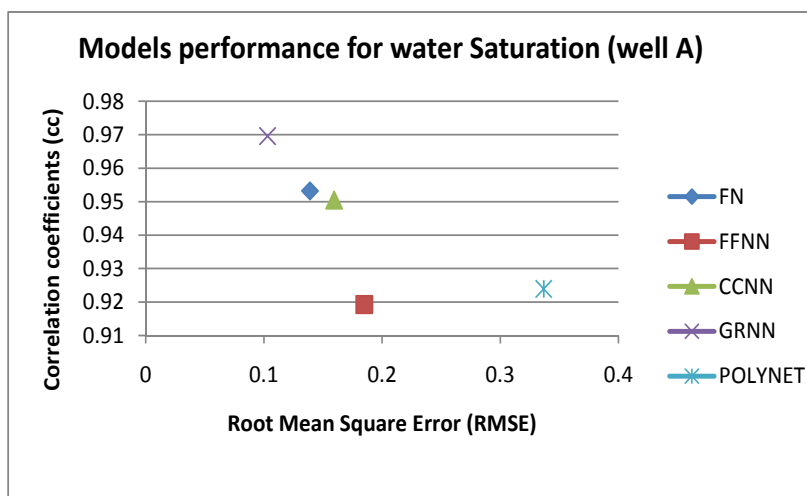


Figure 6.4: Scatter plot of CC versus RMSE for water saturation prediction on Well A for all models

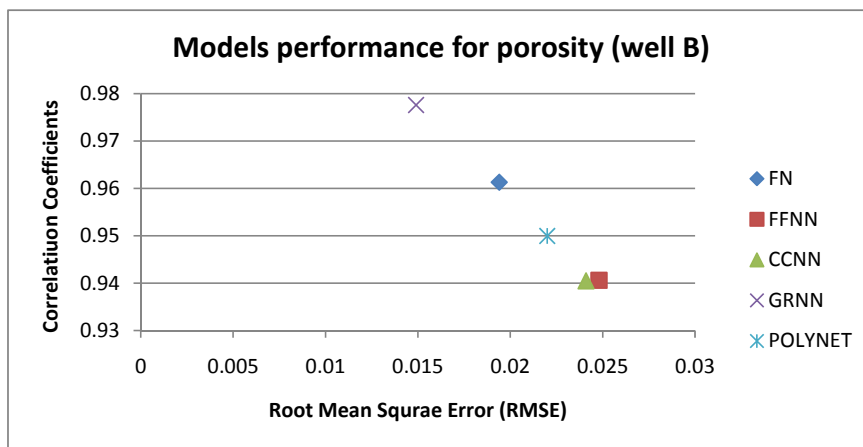


Figure 6.5: Scatter plot of CC versus RMSE for Porosity prediction on Well B for all models

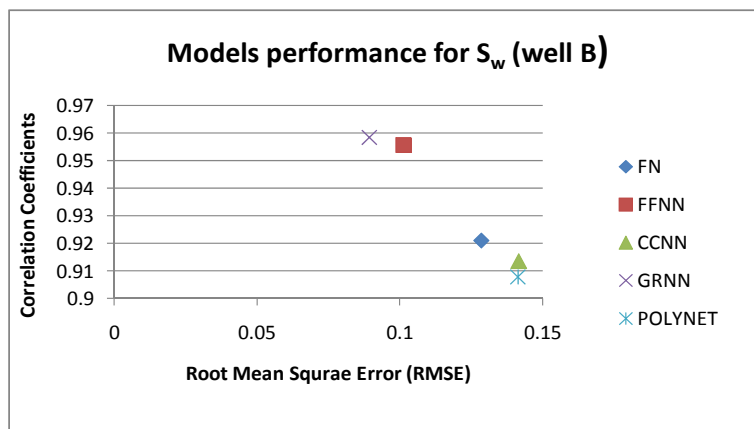


Figure 6.6: Scatter plot of CC versus RMSE for water saturation prediction on Well B for all models

In the third stage, the best models from the first and second stages (FN and GRNN) were compared with the empirical models (EM) (calculated porosity (cal\_por) and water saturation (cal\_swt)). The plots below (Figures 6.7 to 6.14) show that FN and GRNN are better than EM even in porosity prediction where the accuracy is low. This may be because the uncertainty in the empirical models can not be handled accurately or intelligently. Therefore, FN and GRNN can always be used in place of these empirical models.

Table 6.4: FN, GRNN and EM models performance for porosity and water saturation for well A

MODEL	Porosity				Water saturation			
	E <sub>A</sub>	E <sub>R</sub>	RMS	CC	E <sub>A</sub>	E <sub>R</sub>	RMSE	CC
FN	0.0293	0.499	0.03840	0.8577	0.1417	0.666	0.1849	0.9489
GRNN	0.0227	0.303	0.0312	0.9048	0.0584	0.2689	0.1031	0.9696
EM	0.0351	0.510	0.04391	0.8317	0.325	0.785	0.2187	0.8003

Table 6.5: FN, GRNN and EM Models performance for porosity and water saturation for well B

MODEL	Porosity				Water saturation			
	E <sub>A</sub>	E <sub>R</sub>	RMSE	CC	E <sub>A</sub>	E <sub>R</sub>	RMSE	CC
FN	0.01613	0.166	0.02057	0.95658	0.1102	0.406	0.147	0.90401
GRNN	0.0104	0.105	0.0149	0.9776	0.0533	0.141	0.089	0.9584
EM	0.01923	0.193	0.02815	0.92134	0.2541	0.512	0.236	0.80144

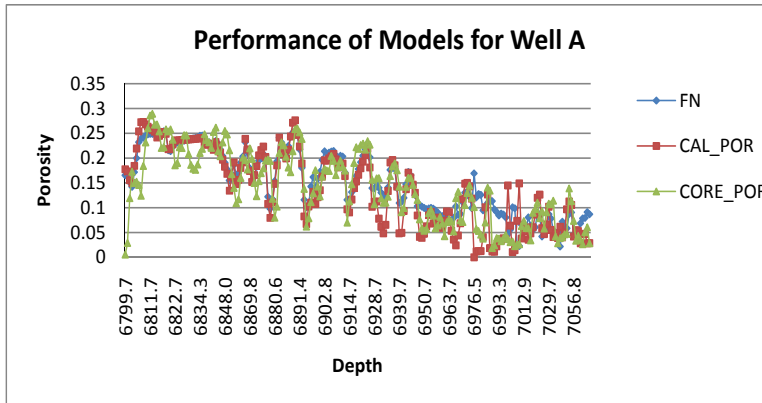


Figure 6.7: Performance of FN, Calculated porosity and the core porosity for well A

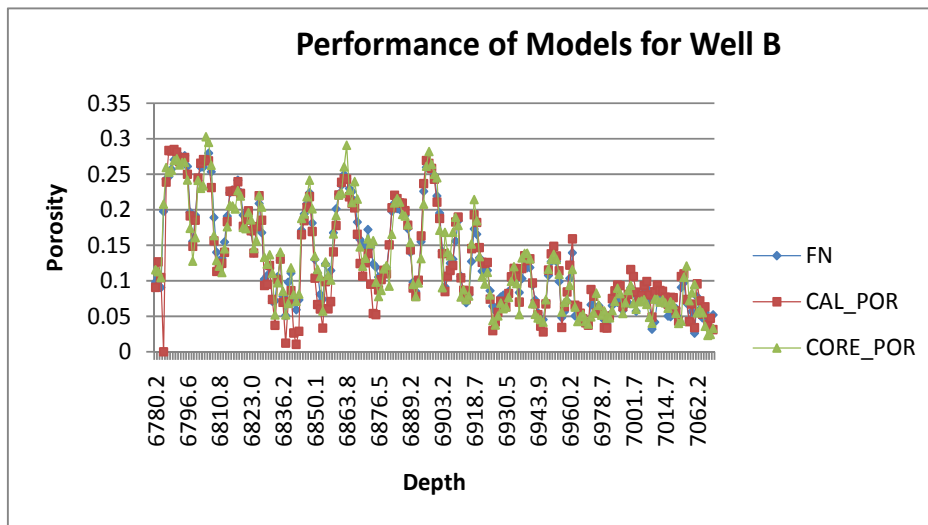


Figure 6.8: Performance of FN, Calculated porosity and the core porosity for well B

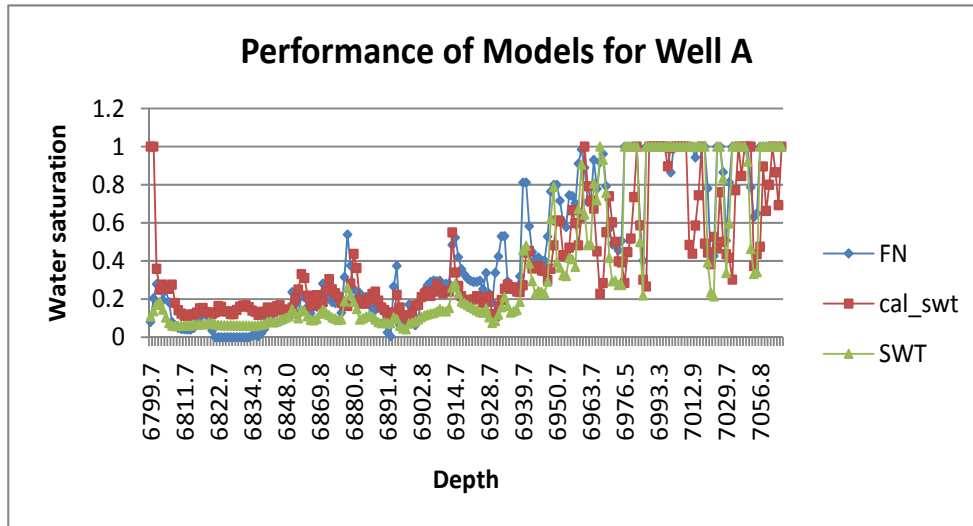


Figure 6.9: Performance of FN, Calculated water saturation and the core water saturation for well A

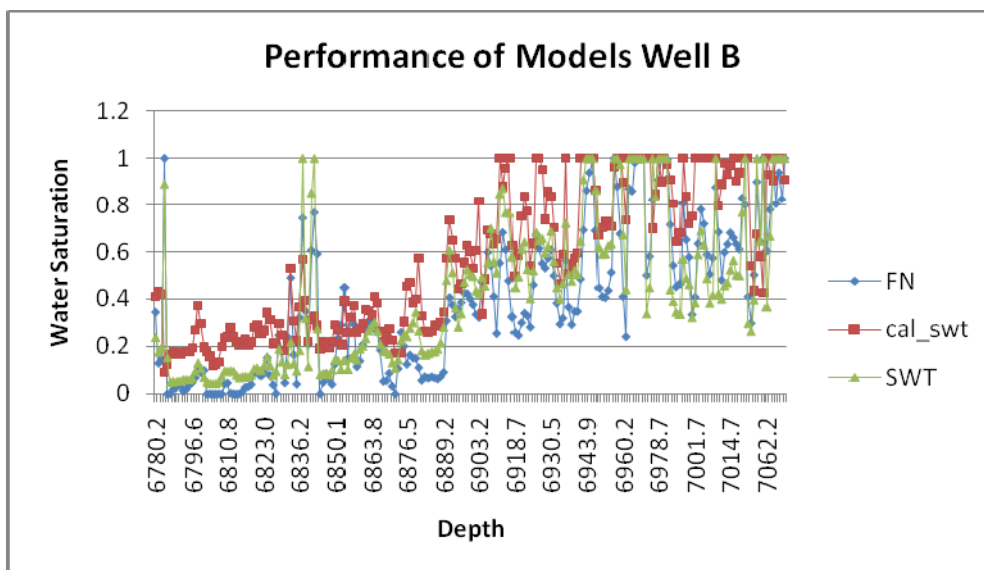


Figure 6.10: Performance of FN, Calculated water saturation and the core water saturation for well B

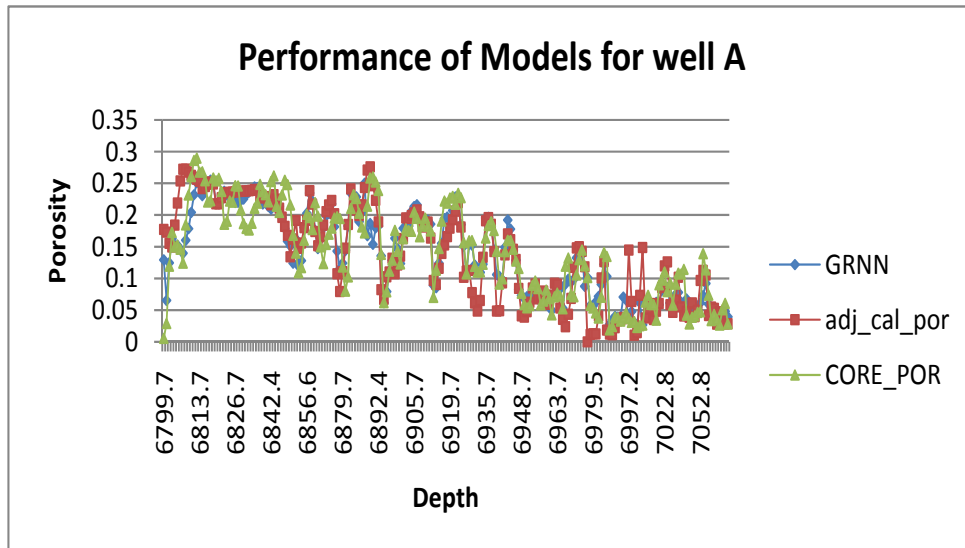


Figure 6.11: Performance of GRNN, Calculated porosity and the core porosity for well A

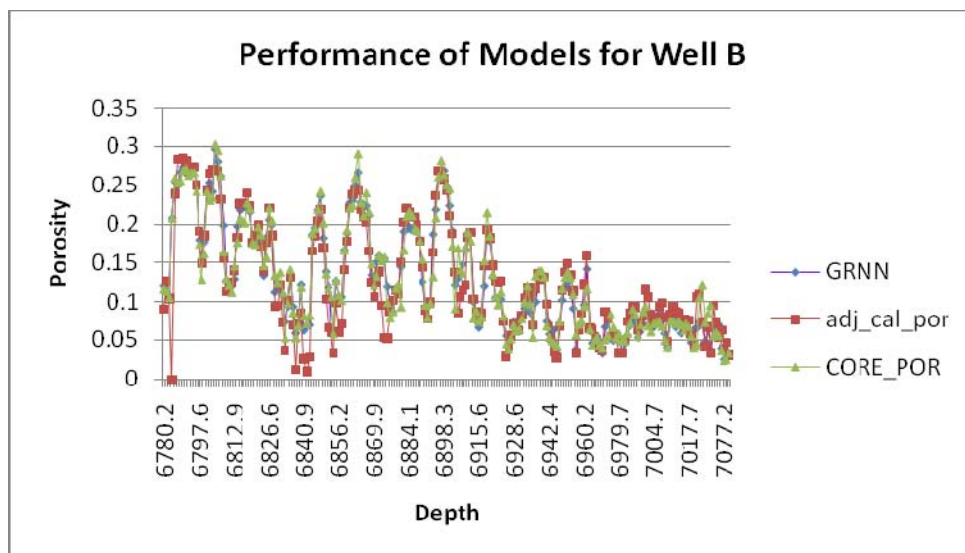


Figure 6.12: Performance of GRNN, Calculated porosity and the core porosity for well B



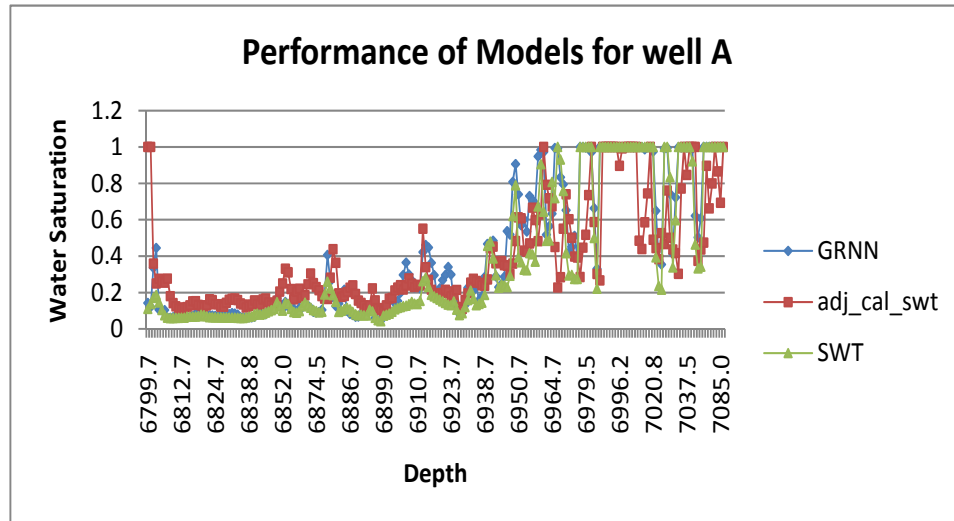


Figure 6.13: Performance of GRNN, Calculated water saturation and the core water saturation for well A

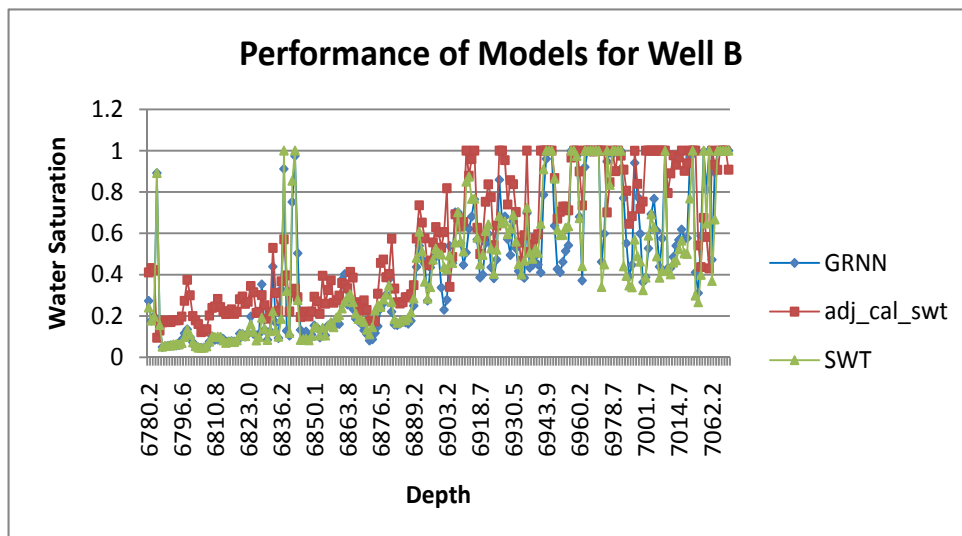


Figure 6.14: Performance of GRNN, Calculated water saturation and the core water saturation for well B

In the last stage of our comparison, the FN and GRNN models were tested with another well label T2 whose statistics are shown in Table 6.6. This well did not participate in the training of the models. This is to further test for the generalization of these models and to particularly determine if the model performance can still be accurate or acceptable on a new well belonging to the same reservoir. The success of this will go a long way to save money and time for the core testing of the new well.

The results of the two best selected models (FN and GRNN) on well T2, as observed clearly from the Table 6.7 and Figures 6.15 to 6.18, are not better than the empirical models. In fact the EM tends to have slightly lower errors and better correlation coefficients. This may be because the uncertainty in the model variables is properly taken care of in this case. However, the results of the FN and GRNN are still acceptable, especially those of the functional network models which are better and can therefore be recommended for nearby wells whose statistics are close to the training data.

Table 6.6: Statistics of Well T2

<b>LOG TYPE (Predictors)</b>	<b>MIN</b>	<b>MAX</b>	<b>AVERAGE</b>	<b>STDEV</b>
Sonic Travel time (DT)	51.21629	82.67955	65.06333	8.247997
Neutron porosity	0.020514	0.273245	0.126999	0.056829
Bulk density (RHOB)	2.185501	2.7303	2.472135	0.126199
Gamma Ray	8.7209	35.82097	16.86412	5.156644
Resistivity	5.600847	146.1045	26.3729	22.11964
Photoelectric Factor (PEF)	2.97236	5.429557	4.14652	0.483861

Table 6.7: Models performance for porosity and water saturation on Test well T2

MODEL	Porosity				Water saturation			
	$E_A$	$E_R$	RMS	CC	$E_A$	$E_R$	RMSE	CC
FN	0.0229	0.3396	0.0297	0.9141	0.1603	0.796	0.211	0.8883
GRNN	0.0224	0.2442	0.0311	0.8961	0.147	0.727	0.213	0.8719
EM	0.0185	0.1001	0.0289	0.94771	0.0616	0.565	0.096	0.88385

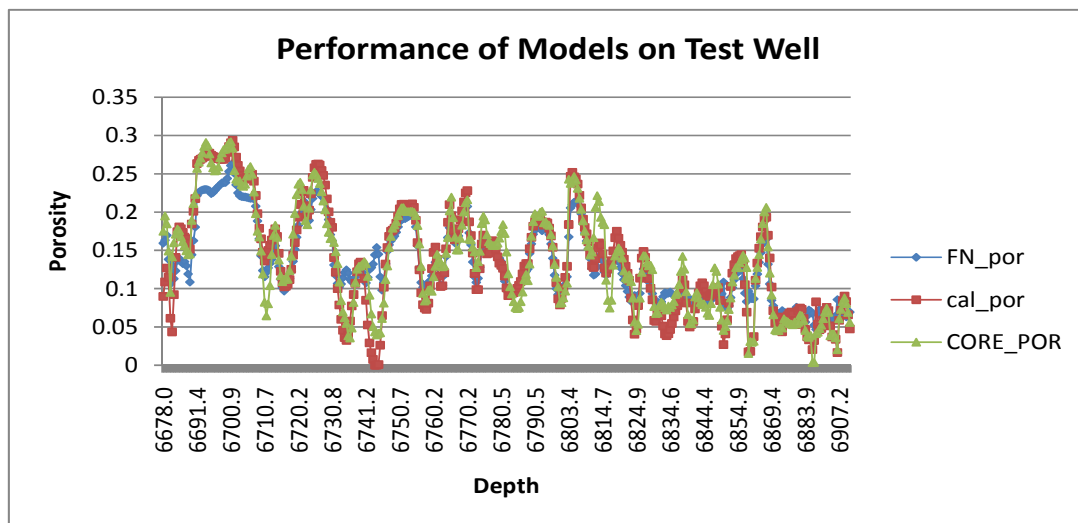


Figure 6.15: Performance of FN, Calculated porosity and the core porosity on Test well T2

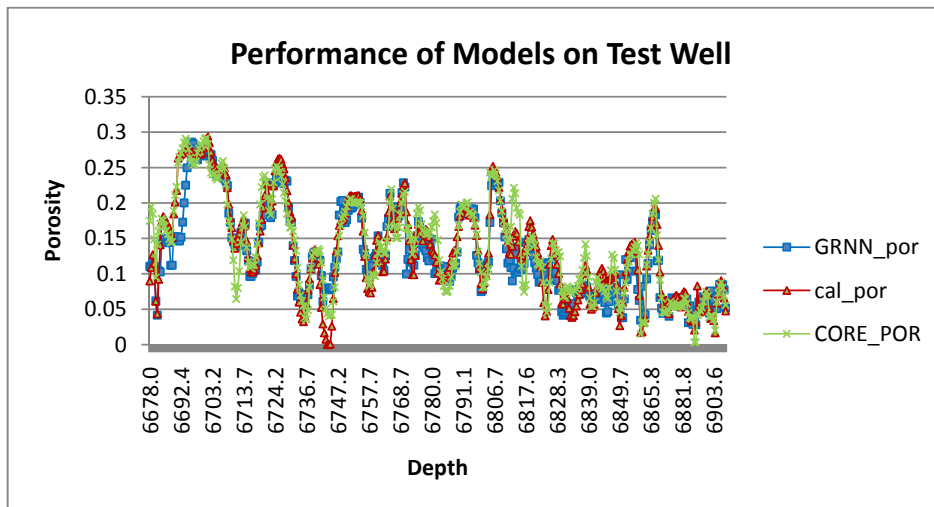


Figure 6.16: Performance of GRNN, Calculated porosity and the core porosity on Test Well T2

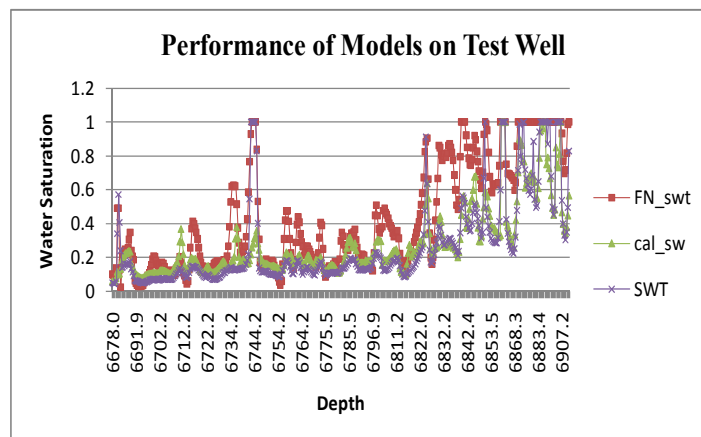


Figure 6.17: Performance of FN, Calculated water saturation and the core water saturation on Test well T2

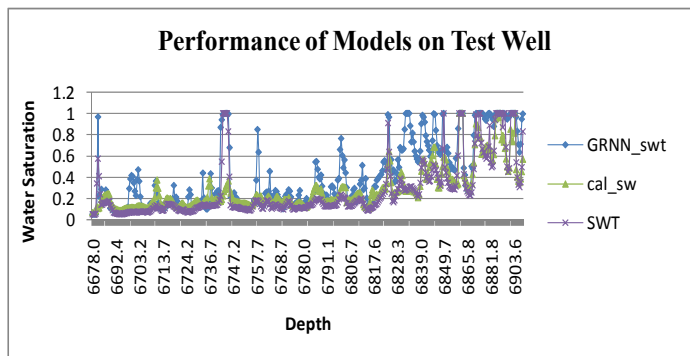


Figure 6.18: Performance of GRNN, Calculated water saturation and the core water saturation on Test Well T2

## 6.2 Discussion

The comparative performance analysis of the five models investigated in the previous section gives a lucid picture of the best models to use for estimation of porosity and water saturation and also the conditions that surrounds their use. For clarity, the summary of the performance is shown in Table 6.8. In the performance analysis, although GRNN and FN are selected to be the best models from the five studied models, the three other models (CCNN, Polynet & FFNN) are also adequate for the prediction of porosity and water saturation though with some deficiency as shown in Table 6.8 (e.g. robustness and number of network weight).

Furthermore, GRNN and FN show different strengths by the evaluation criteria as shown in Table 6.8. FN is better than GRNN in terms of the sensitivity to a new well, the number of network weights, the testing time, and the insight into the network. GRNN is

better in terms of training time, accuracy, and robustness. One of the strongest points of FN that is not shown in Table 6.8 is the access to the input/output relationship of the network, which can be obtained from the insight of the network. For example, equation 6.2 shows that the first three variables (DT, NPHI and RHOB) are more significant in the prediction of porosity while in equation 6.3 one variable (PEF) is insignificant in the prediction of water saturation. The result of using the significant variables, as shown in appendix C, is as good as using six and five variables for the prediction porosity and water saturation respectively. This is an indication that trade-off in using either the GRNN or the FN model should be considered.

Table 6.8: Brief Summary of all models performance for porosity and water saturation

<b>Model</b> <b>Measure</b>	<b>FFNN</b>	<b>CCNN</b>	<b>polynet</b>	<b>FN</b>	<b>GRNN</b>
Accuracy	fair	fair	fair	good	very good
No. of Network Weight	large	large	medium	small	very large
Training Time	high	Very	very low	low	low
Testing Time	high	low	very low	low	very high
Robustness	weak	weak	weak	good	very good
Sensitivity to new well	-	-	-	Very good	good

# CHAPTER SEVEN

## CONCLUSION AND RECOMMENDATIONS

### 7.1 Summary

This study investigated the application of recent advances in neural networks techniques in developing an accurate model for the prediction of rock properties (porosity and water saturation). A simplified functional networks (FN) model was developed for the prediction of porosity and water saturation. Four basis functions were tested as the training functions on the model, and the best was selected.

The other part of the study focused on designing and investigating Cascaded Correlation neural networks (CCNN), Polynomial neural networks (Polynet), General regression neural networks (GRNN), and Feedforward neural networks (FFNN) for the prediction of these rock properties.

In all the modeling techniques, data from real carbonate field were used. Three wells were studied, labeled; well A, well B and well T2. While data from well A and B participated in the training of the models, well T2 did not but it was used for further tests on the models to check their robustness.

Finally, comparative analyses were carried out between these models, to determine not only the most accurate on the trained wells but also the robustness of the model on untrained well. The section below gives the conclusion from the study.

## 7.2 Conclusion

Based on the research, experiment and analysis discussed previously in this study, the following conclusions are reached:

- Generally, all the techniques investigated in this study, (GRNN, FN, polynet and CCNN) can be said to at least match FFNN in accuracy, and they are better in their design and in the speed of training in predicting porosity and water saturation. However, since CCNN is not free from the problem of random initialization of weights, its result is not unique, which is similar to that of standard FFNN.
- Functional network is a good model, but not robust enough compared to GRNN if the spread parameter of GRNN is well chosen. However, the training of FN is much faster.
- FN proved to have good generalization, better than GRNN. According, FN can be used for nearby wells with statistics close to those of the training data.
- FN shows that three inputs (DT, NPHI and RHOB) are adequate for the prediction of porosity, while four inputs (NPHI, RHOB, RT and GR) are adequate for the prediction of water saturation.



- GRNN can be used when more accuracy is required on the trained well only.
- Feedforward neural networks (FFNN) can have good generalization if there is enough time and knowledge of the network topology.
- Cascaded correlation neural network (CCNN) can be a guide in designing FFNN as it gives a picture of the topology automatically but it is not always as accurate as FFNN.

### **7.3 Recommendations for Future Work.**

Although good results were obtained from the investigation carried out in this thesis, better accuracy and generalization may be obtained from the following recommendations:

- 1) Investigating combined networks in the form of committee machine or modular networks where different networks such as Ploynet, FN and GRNN are combined. These three models were suggested because of the unique results that can be obtained from them. However, many factors need to be considered in developing such architecture, including:
  - There are several different methods of combining networks, depending on the task at hand.
  - Determination of training strategy for different networks to achieve more accurate network model can be challenging. For example, do we train each

network with the same data set, or different data sets, with overlapping or without overlapping?

- Determination of the techniques to combine is also not an easy task.
  - Model complexity versus gained accuracy is another factor to consider.
- 2) Investigating means to establish a relationship between a trained model and the data of a new well such that the model can be adjusted by a factor to suit the statistics of the new well.
- 3) It is also recommended to explore more FN models and basis functions. Other optimization techniques, besides the least square method, may also be employed for training.

It is hoped that implementation of these recommendations will lead to better models for the prediction of rock properties.

## REFERENCES

- [1] Ahmed T., Porter, K.W., Wideman. C. J, Himmer, P., and. Braun, J Application of Neural Network Parameter Prediction in Reservoir Characterization and Simulation- A case History: The Rabbit Hills field.” SPE 38985. Fifth Latin American and Caribbean Petroleum Engineering Conference and Exhibition Rio de Janeiro, Brazil, 30 August-3 September 1997.
- [2] Ajith A. and Baikunth N., “Optimal design neural nets using hybrid algorithm” proceedings of 6th Pacific Rim International Conference on Artificial Intelligence PRICAI, 2000
- [3] Abhijit Y. D. “Petroleum Reservoir Rock and Fluid Properties”, Taylor and Francis Group, LLC, London, 2006.
- [4] Alfonso I. , Bernardino A. , Cotos J. M. , Taboada J. A. and Carlos D., “A comparison between functional networks and artificial neural networks for the prediction of fishing catches” Neural Computing & Applications (2004) 13: 24–31
- [5] Balan, B., Mohaghegh, S., Ameri, S. “State-Of-The-Art in Permeability Determination From Well Log Data: Part 1- A Comparative Study, Model Development” SPE 30978, SPE Eastern Regional Conference & Exhibition, Morgantown, West Virginia, U.S.A., 17-21 September, 1995.
- [6] Barron, A.R.: “Predicted squared error - a criterion for automatic model selection,” in Self-Organizing Methods in Modeling: GMDH Type Algorithms, Farlow, S.J., ed., Marcel-Dekker, New York, 1984.
- [7] Bueno E.O., Peez I. C., Scamilla G., Mesa H., Ponce B., Graham R., Kessie C. and Muillo J. ”Applications of Artificial Neural Networks and Dipole Sonic Anisotropy in Low-porosity Naturally Fractured, Complex Lithology Formations in the Southern Land of Mexico” SPE 103662. First International Oil Conference and Exhibition in Mexico, Cancun, Mexico, 31 August–2 September 2006.
- [8] Castillo E. “Functional Networks” Neural Processing Letters 7: 151–159, 1998.
- [9] Castillo E, Cobo A., Manuel J. G., and Pruneda E, “Functional Networks with Application: A Neural based Paradigm” Boston Kluwer Academic Publisher 1999.
- [10] Castillo E, Cobo A., Manuel J. G., and Pruneda E, (2000a) “Functional Networks: A New Network-Based Methodology” Computer-Aided Civil and Infrastructure Engineering, Volume 15 Issue 2 Page 90-106, March 2000
- [11] Castillo E., Cobo A, Gómez-Nesterkin R. and Ali S. H. (2000b) “A General Framework for Functional Networks” NETWORKS, Vol. 35(1), 70.82 2000

- [12] Castillo E, Cobo A., Manuel J. G., and Castillo C.(2000c) "Some Learning Methods in Functional Networks" *Computer-Aided Civil and Infrastructure Engineering*, Volume 15 Issue 2 Page 427-439, March 2000
- [13] Castillo E, Manuel J. G., Hadi A. S. and Lactruz B., "Some Applications of Functional Networks in Statistics and Engineering", *Technometrics*, vol. 43, no.1, Feb. 2001.
- [14] Castillo E., Alfonso I., and Ruiz-cobo R. "Functional Equation in Applied sciences" *Mathematics in Science and Engineering Volume 199 series Editor C.I. CHUI, ELSEVER, 2005.*
- [15] Chandra B. and Paul V.P. "Applications of Cascade Correlation Neural Networks for Cipher System Identification" *Proceedings of World Academy of Science, Engineering and Technology Volume 20 April 2007 ISSN 1307-6884*
- [16] Changhua Z., Changxing P., Jiandong L., "Functional Networks Based Internet End-to-End Delay Dynamics" *Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA'04), The Computer Society IEEE, 2004.*
- [17] Dresser A, "Well Logging and Interpretation Techniques" 3rd edition, Dresser Industries Inc. 1982.
- [18] El-Sebakhy E. A., Kanaan A., Helmy T., Azzedin F, and Al-Suhaim A., "Evaluation of Breast Cancer Tumor Classification with Unconstrained Functional Networks Classifier" *IEEE, 2006.*
- [19] Essenreiter R, Karrenbach M. Treite S., "Multiple reflection attenuation in seismic data using backpropagation" *IEEE Transaction on Signal Processing*, 46 (7), 1998.
- [20] Fahlman S. and Lebiere C., "The Cascaded-Correlation Learning Architecture". *Neural Information Processing Systems*, 2 1990.
- [21] Farlow S. J., "The GMDH algorithm," in *Self-Organizing Methods in Modeling: GMDH Type Algorithms*, S. J. Farlow, ed., Marcel-Dekker, New York , 1984.
- [22] Gharib H. and Moustafa E. "Evaluation Of Petrophysical Properties Of Sandstone Reservoirs Using Artificial Neural Network (Ann) Approach" *Icmsa07, UAE, March 2007.*
- [23] Hagan M. T. and Fun M. H. "Levenberge-Marquardt Training Algorithm for Modular Networks," *IEEE International Conference on Neural Networks*, vol. 1, pp. 468-473, 1996.
- [24] Hamidreza R. K and Karim F. "Wave Height Forecasting Using Cascade Correlation Neural Network "WSCG POSTERS proceedings WSCG'2004, Plzen, Czech Republic, February 2-6, 2004.

- [25] Hassibi, B. and Stork, D. G. "Second order derivatives for network pruning: Optimal Brain Surgeon", Proceedings of the Neural Information Processing Systems-5. S. J. Hansen, J. D. Cowan, and C. L. Giles (eds.), Morgan-Kaufmann 1992.
- [26] Haykin S. "Neural networks: A comprehensive foundation". Macmillan International 1999.
- [27] Helle H.B., Bhatt A. and Ursin B. (2001a). Porosity and permeability prediction from wireline logs using artificial neural networks: a North Sea case study. *Geophysical Prospecting* 49, 431-444, 2001.
- [28] Helle H.B., Bhatt A. and Ursin B. (2001b). Application of Committee machine in reservoir Characterization while drilling: A Novel Neural Networks Approach in log analysis", Proceedings of the 6<sup>th</sup> Nordic Symposium on Petrophysics, Trondheim Norway, 15-16 May 2001.
- [29] Helle H.B. and Bhatt A. "Fluid saturation from well logs using committee neural networks", *Petroleum Geosciences* 8 May 2002. Expanded abstract presented in EAGE 64th conference and technical exhibitions-Florence, Italy, 2002.
- [30] Hornik K .M., Stinchcombe M. and White H "Multilayer feedforward networks are universal approximators". *Neural networks* 2(5), 359-366 1986.
- [31] Huang Z., Shimeld J., Williamson M. and Katsube J. "Permeability prediction with artificial neural network modeling in the Venture gas field, offshore eastern Canada. *Geophysics* 61 422-436 1996.
- [32] Huang Z. and Williamson M.A. "Determination of porosity and permeability in reservoir intervals by artificial neural network modeling, offshore eastern Canada. *Petroleum Geoscience* 3, 245-258, 1997.
- [33] James B. "Applied open-hole log Analysis" Vol. 2, Gulf Publishing Company Houston 1986.
- [34] Jerry L. F. "Carbonate Reservoir Characterization" Springer-Verlag 1999.
- [35] Jong-Se L. "Reservoir properties determination using fuzzy logic and neural networks from well data in offshore Korea" Division of Ocean Development Engineering, Korea Maritime University, Busan, 606-791, Republic of Korea Accepted 20 May 2005.
- [36] Jong-Se L. and Jungwhan K. "Reservoir Porosity and Permeability Estimation from Well log using Fuzzy Logic and Neural Network". SPE 88476. SPE Asia Pacific Oil and Gas Conference and Exhibition, Perth, Australia, 18–20 October 2004.
- [37] Jun Y. " Reservoir Parameter Estimation from well log and core data: A case study from the north sea " *Petroleum Geosciences*, Vol. 8 pp 63-69, 2002.
- [38] Kartoatmodjo, T. and Schmidt, Z.: "Large data bank improves crude physical property correlations," *Oil and Gas Journal* 51, July 4, 1994 .

- [39] Maren A. J., Pap R. M., Harston, C. T. Handbook of neural computing applications, Academic Press, New York, 1990.
- [40] McKenna S. J., Ricketts I. W., Cairns A. Y, Hussein K. A Cascade-Correlation Neural Networks For The Classification Of Cervical Cells, 1997
- [41] Mohaghegh S. “Application of virtual intelligence to petroleum Engineering” Computers & Geoscience 26 (Special Issue) 2000.
- [42] Nabil A., Mariela A. and Martin K. “ Predicting Water Saturation Using Artificial Neural Networks” Proceedings of the 25<sup>th</sup> IASTED International Multi-Conference Artificial Intelligence And Applications Innsbruck, Austria, Feb 12-14, 2007.
- [43] Nikravesh M., Amnizadeh F. and Zadeh L.A. (eds) Soft computing and earth science. Journal of Petroleum Science & Engineering, 29 (Special Issue) 2001.
- [44] Attoh-Okine N. O., “Modeling incremental pavement roughness using functional networks” Can. J. Civ. Eng. 32(5): 899–905, 2005.
- [45] Oya A., Ethem A. “An Incremental Neural Network Construction Algorithm for Training Multilayer Perceptrons” ICANN'03, Istanbul, June 2003
- [46] Patterson D.W. “Artificial Neural Networks: theory and applications” Prentice Hall, 477p, 1996.
- [47] Ridha B.C. and Mansoori A. G. “An introduction to artificial intelligence applications in petroleum exploration and production“, Editorial, Journal of Petroleum Science and Engineering 49 (2005) 93– 96.
- [48] Soto R. B., Ardilla J.F., Ferneynes H., and Bejarano H. “Neural Network To Predict the Permeability and Porosity of Zone “C” of The Cantagalo Field in Colombia” SPE 38134. petroleum Computer Conference Dallas, Tx, USA, 8-11 June 1997.
- [49] Rajasekaran S. Thiruvengatasamy K, Tsong-Lin L, “Tidal level forecasting using functional and sequential learning neural networks” Applied Mathematical Modelling 30 (2006) 85–103
- [50] Sandha A. Agha K. Islam R. “Artificial intelligence as a decision support system for petroleum engineers”, petroleum science and technology, vol. 23, no5-6, pp. 555-57, 2005.
- [51] Schetinin V. “A Learning Algorithm for Evolving Cascade Neural Networks”. Neural Processing Letters 17(1): 21-31 (2003) Kluwer.
- [52] Schiffmann W., Joost M., Werner R. “Comparison of Optimized Backpropagation Algorithms” Proc. of ESANN'93, Brussels 1993.

- [53] Shokir E. M, "Prediction of Hydrocarbon Saturation in low Resistivity Formation via Artificial Neural Network" SPE 87001. SPE Asia Pacific Conference on Integrated Modelling for Asset Management, Kuala, Lumpur, Malaysia, 29-30 March 2004.
- [54] Specht D. F, "A General Regression Neural Network" IEEE Transaction on Neural Networks Vol. 2. No. 6. November 1991.
- [55] Stan P. and Pedro A. R. "Application Of Core-Log Correlation And Artificial Neural Networks To Better Define Permeability, Porosity And Lithology" Sca-9836-1998.
- [56] Stepniewski S.W. and Keane A.J. "Pruning backpropagation neural networks using modern stochastic optimization techniques" Neural Computing and Applications, 5, (2), 76-98, 1997
- [57] Thomas R., Steffen G. and Hai M. S., "Automatic Determination of Optimal Network Topologies Based on Information Theory and Evolution", Proceedings of the 23rd EUROMICRO Conference, vol., no., pp.549-555, 1-4 Sep 1997.
- [58] Zhang Y., Salisch H.A. and McPherson J.G. Application of neural networks to identify lithofacies from well logs. Exploration Geophysics 30, 45-49 1999.
- [59] Zhang Y., Salisch H.A. and Arns C. Permeability evaluation in a glauconite-rich formation in the Carnarvon Basin Western Australia. Geophysics 65, 46-53 2000.

## APPENDIX A: Performance of other Basis Functions used for Functional Networks in the Combined Well

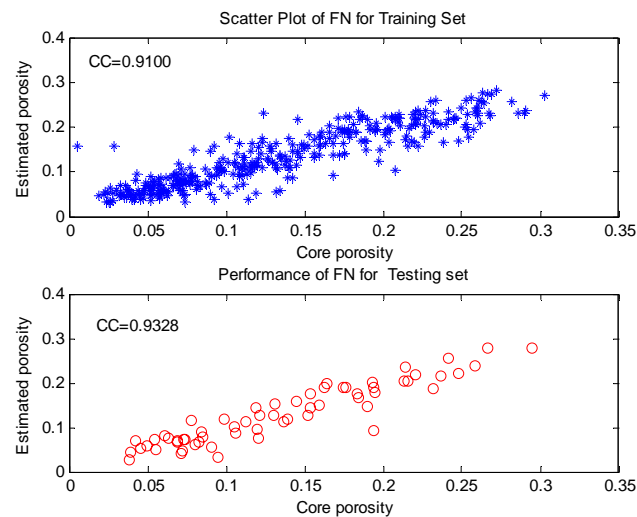


Figure A1: Scatter plot for estimated porosity versus Core porosity using Logarithm function

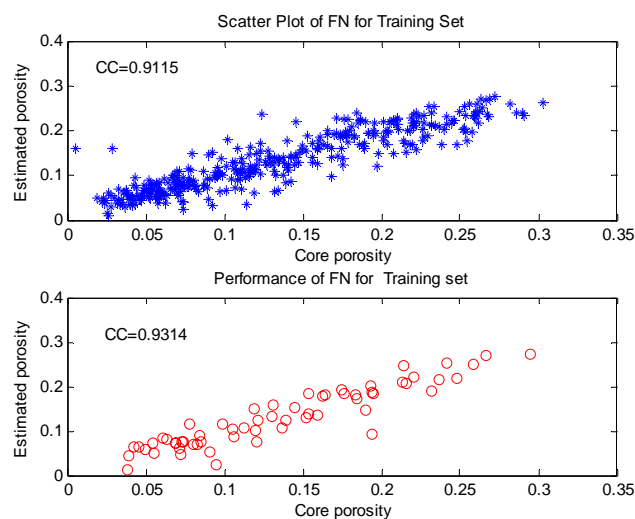


Figure A2: Scatter plot for estimated porosity versus Core porosity using Polynomial function



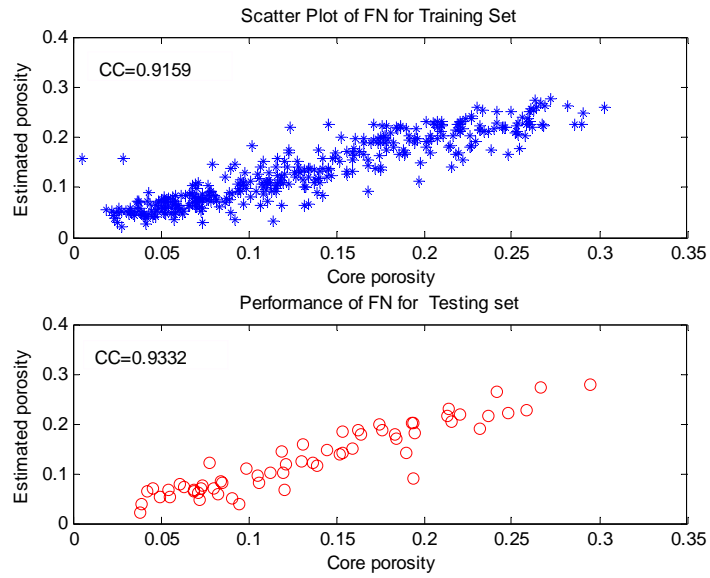


Figure A3: Scatter plot for estimated porosity versus Core porosity using Exponential function

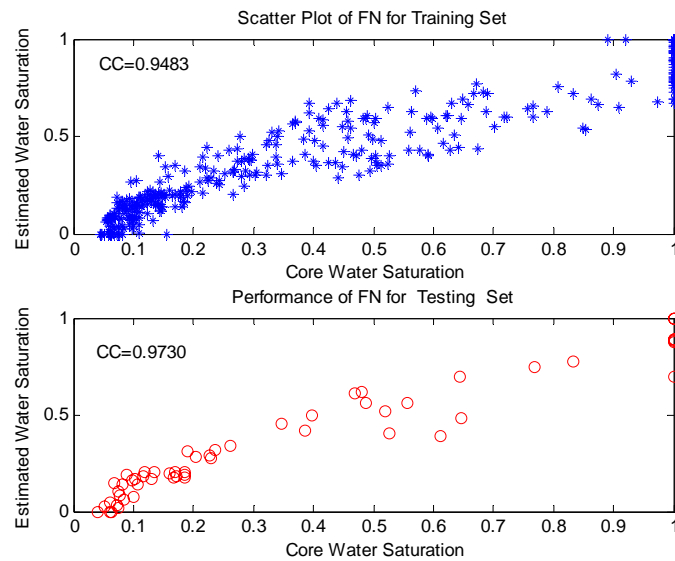


Figure A4: Scatter plot for estimated Water Saturation versus Water Saturation using polynomial function

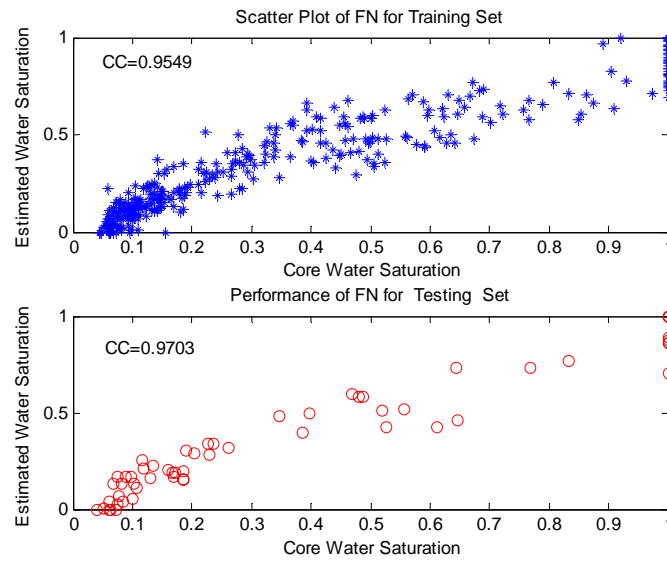


Figure A5: Scatter plot for estimated Water Saturation versus Water Saturation using Exponential function

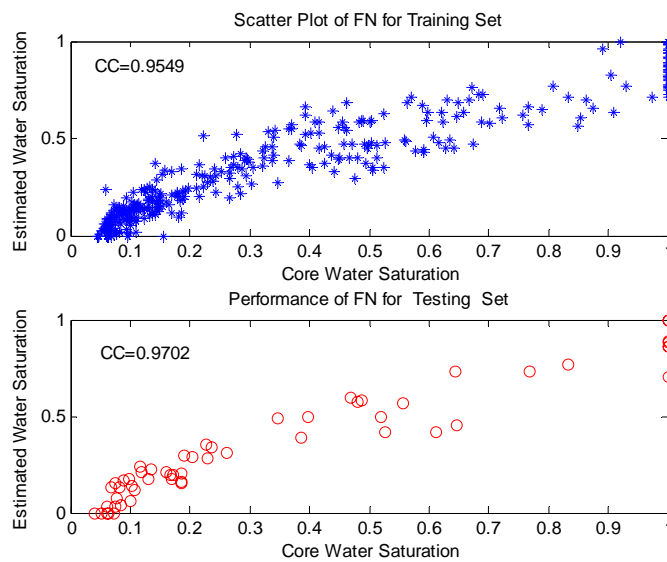


Figure A6: Scatter plot for estimated Water Saturation versus Water Saturation using Fourier function

## APPENDIX B: Performance of All Models on Wells A and B

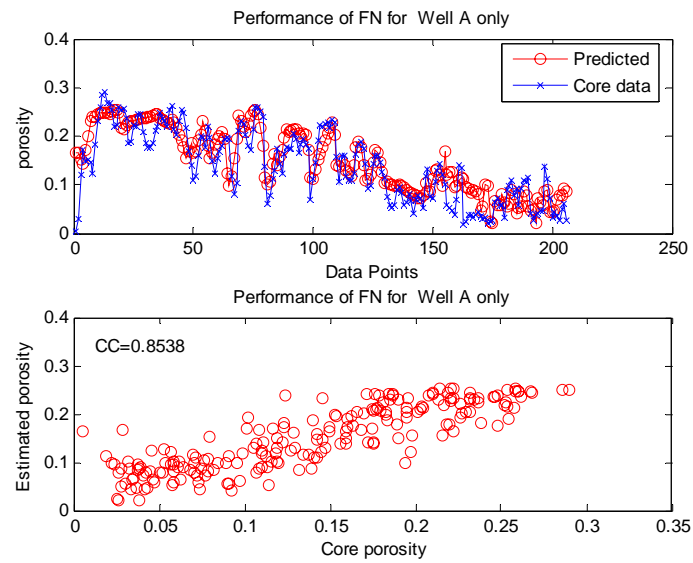


Figure B1: Scatter plot for estimated Porosity versus Core Porosity using functional Network function on Well A

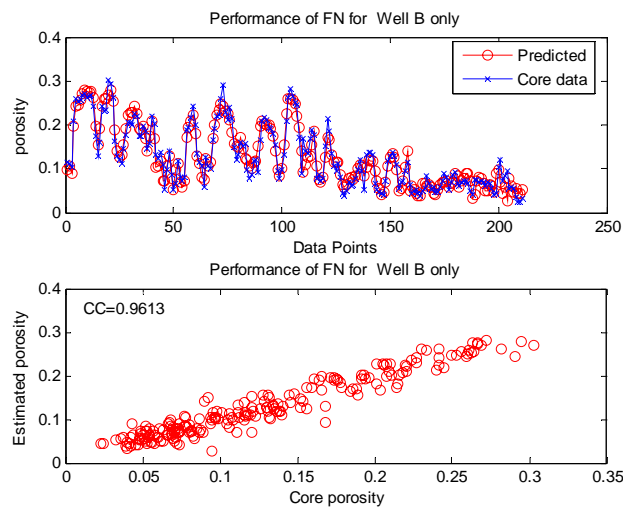


Figure B2: Scatter plot for estimated Porosity versus Core Porosity using functional Network function on Well B

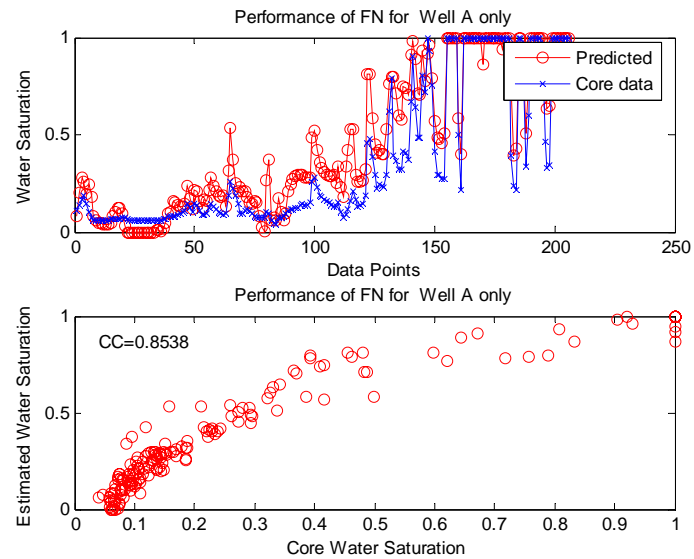


Figure B3: Scatter plot for estimated Water Saturation versus Core Water Saturation using functional Network function on Well A

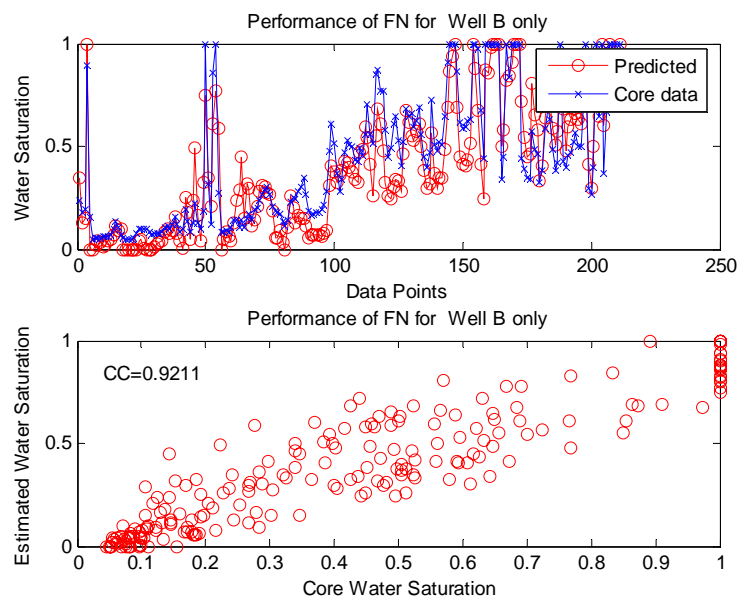


Figure B4: Scatter plot for estimated Water Saturation versus Core Water Saturation using functional Network function on Well B

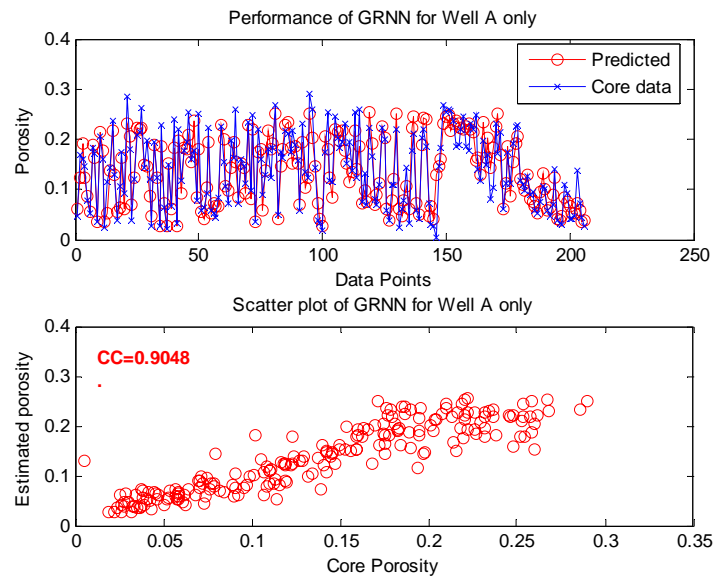


Figure B5: Scatter plot for estimated Porosity versus Core Porosity using GRNN on Well A

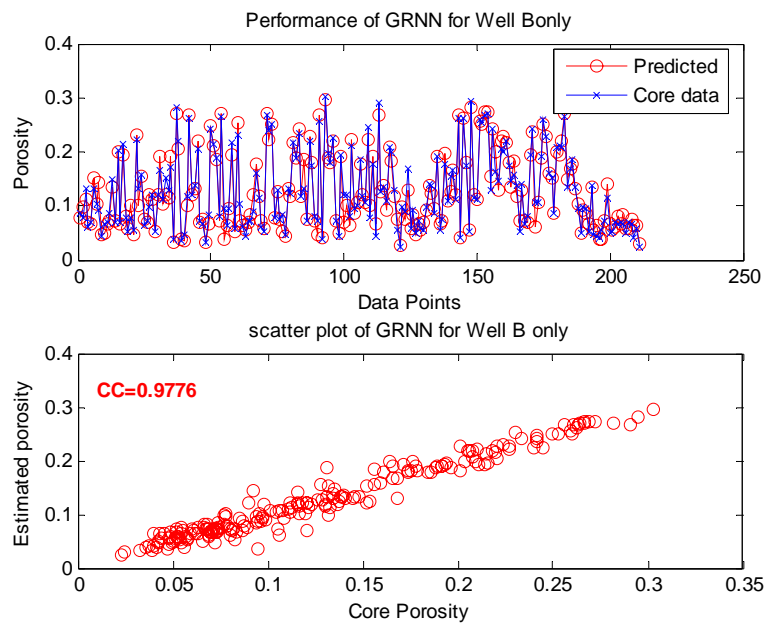


Figure B6: Scatter plot for estimated Porosity versus Core Porosity using GRNN on Well B

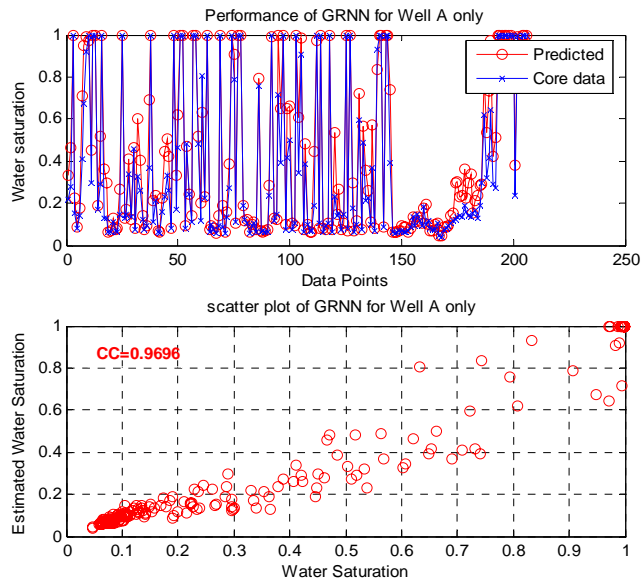


Figure B7: Scatter plot for estimated Water Saturation versus Core Water Saturation using GRNN on Well A

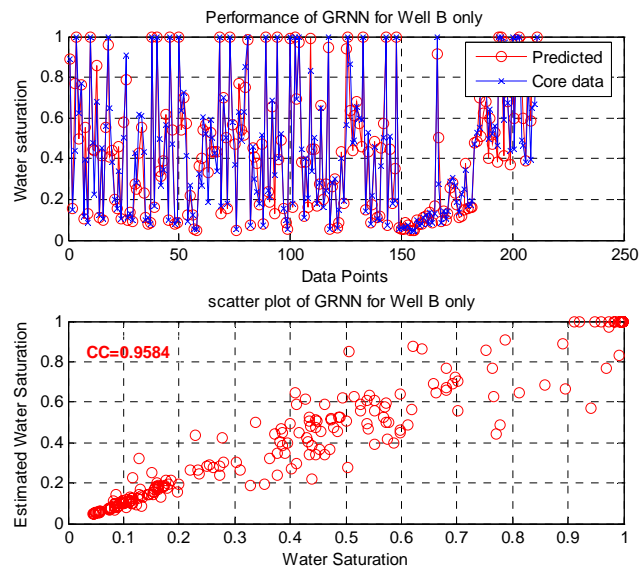


Figure B8: Scatter plot for estimated Water Saturation versus Core Water Saturation using GRNN on Well B

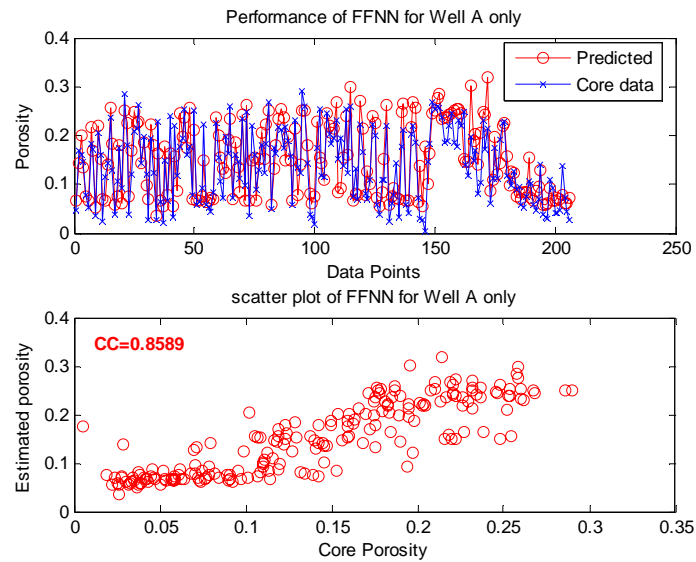


Figure B9: Scatter plot for estimated Porosity versus Core Porosity using FFNN on Well A

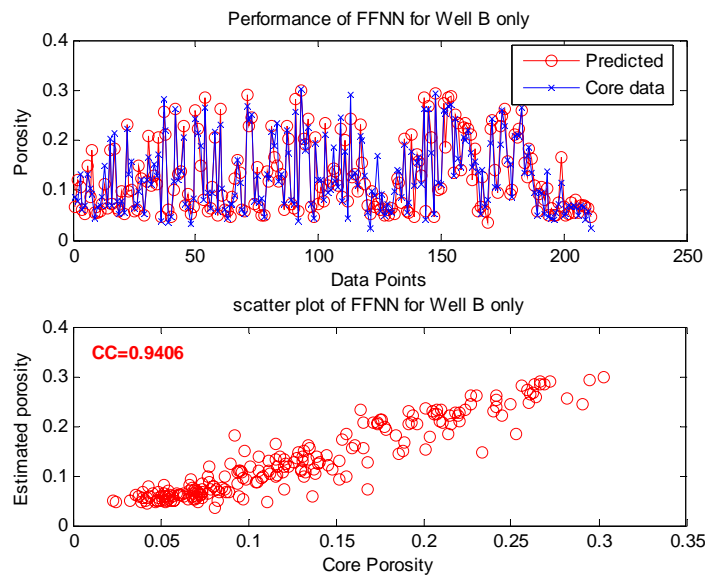
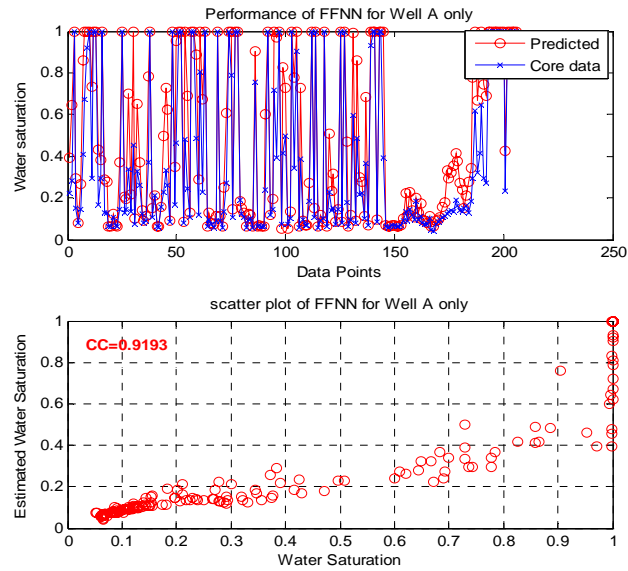


Figure B10: Scatter plot for estimated Porosity versus Core Porosity using FFNN on Well B



FigureB11:Scatter plot for estimated Water Saturation versus Core Water Saturation using FFNN on Well A

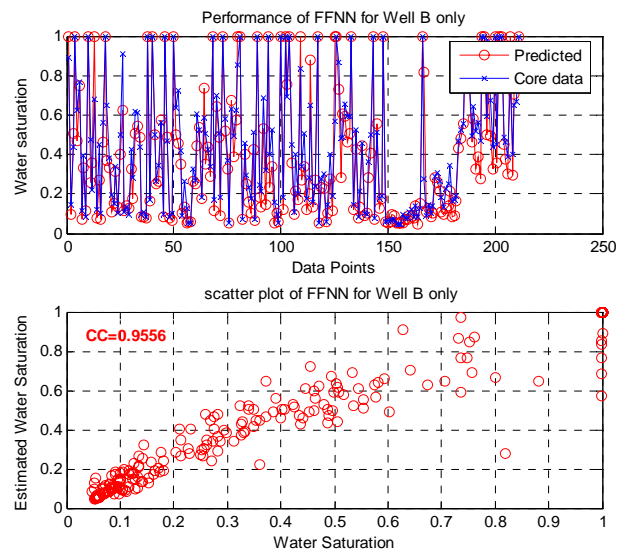


Figure B12: Scatter plot for estimated Water Saturation versus Core Water Saturation using FFNN on Well B



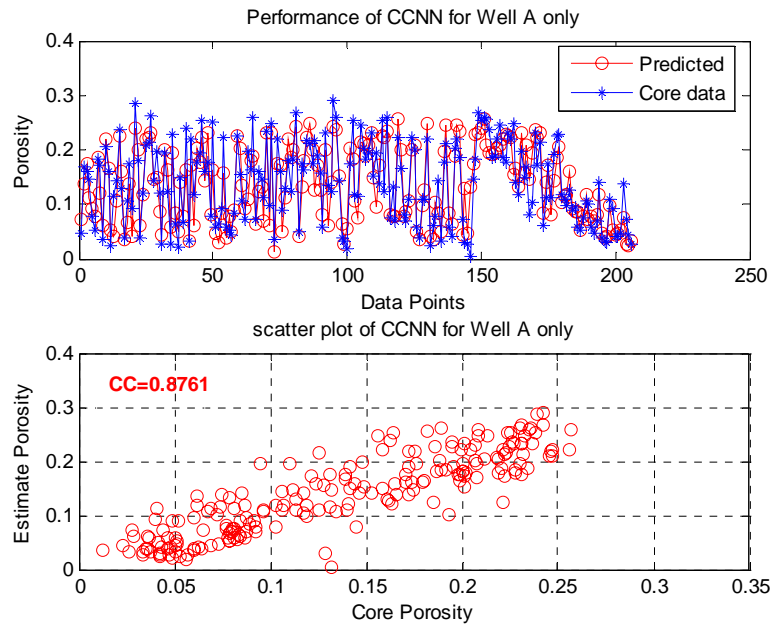


Figure B13: Scatter plot for estimated Porosity versus Core Porosity using CCNN on Well A

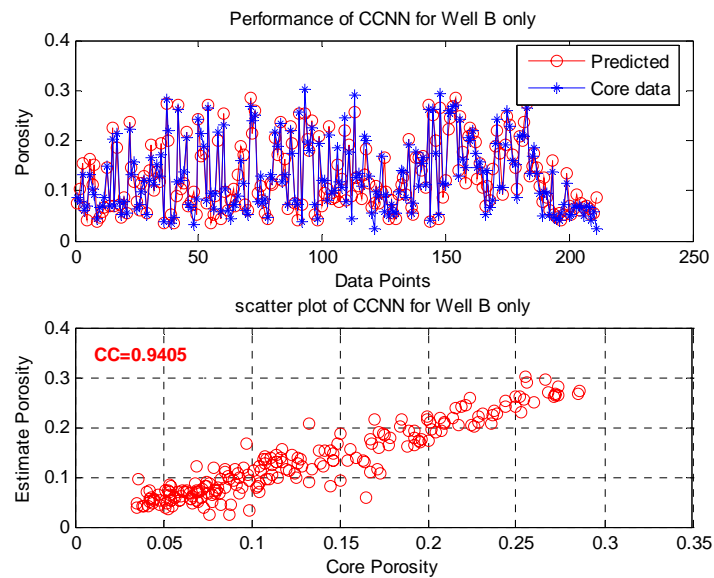


Figure B14: Scatter plot for estimated Porosity versus Core Porosity using CCNN on Well B

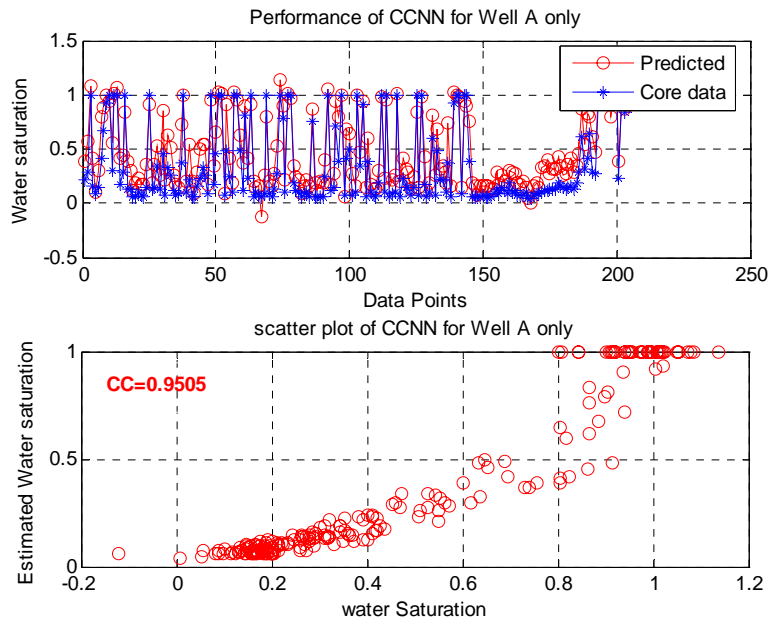


Figure B15: Scatter plot for estimated Water Saturation versus Core Water Saturation using CCNN on Well A

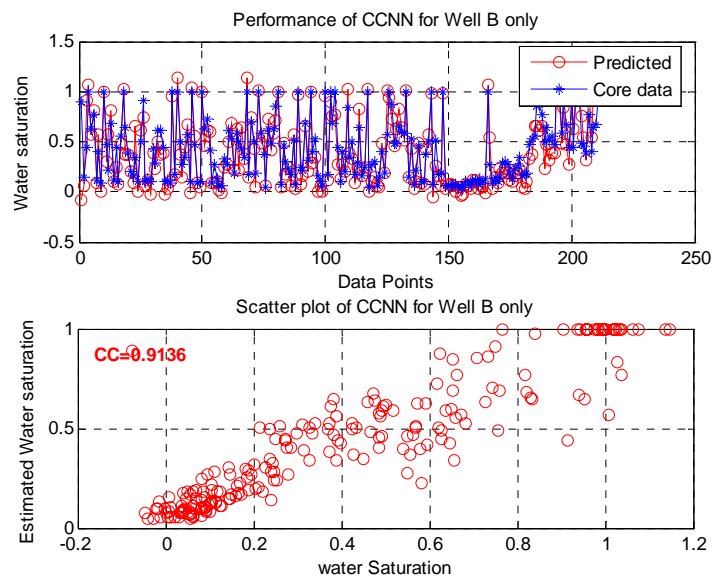


Figure B16: Scatter plot for estimated Water Saturation versus Core Water Saturation using CCNN on Well B

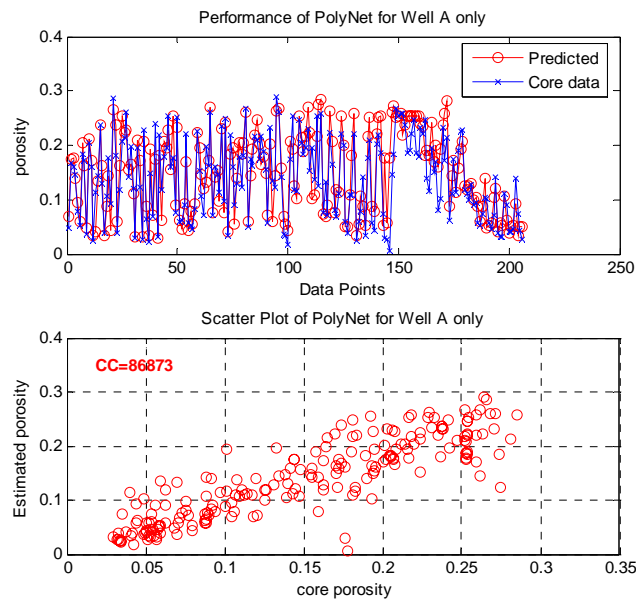


Figure B17: Scatter plot for estimated Porosity versus Core Porosity using Polynet on Well A

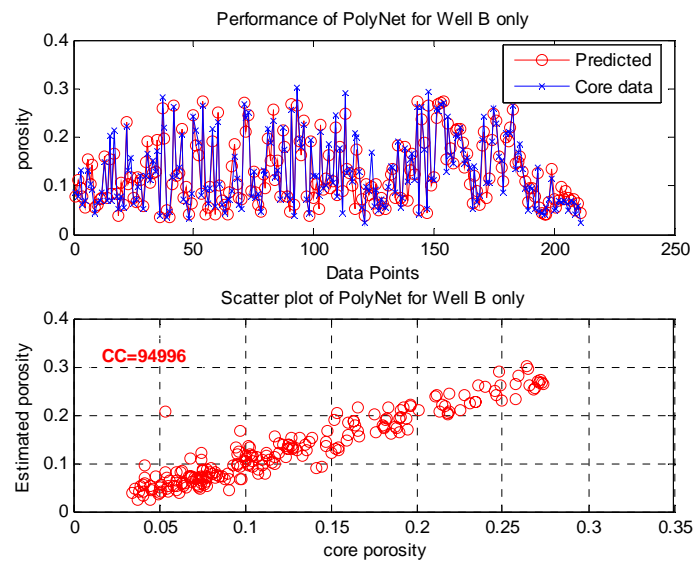


Figure B18: Scatter plot for estimated Porosity versus Core Porosity using Polynet on Well B

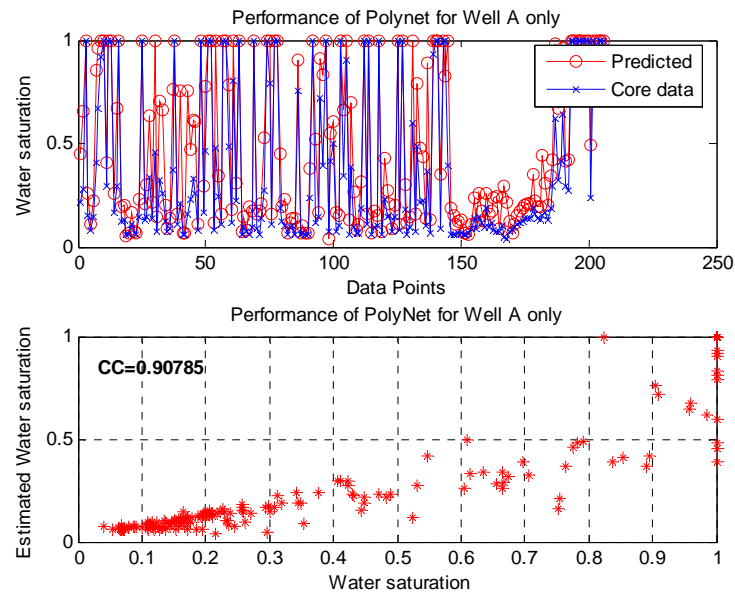


Figure B19: Scatter plot for estimated Water Saturation versus Core Water Saturation using Polynet on Well A

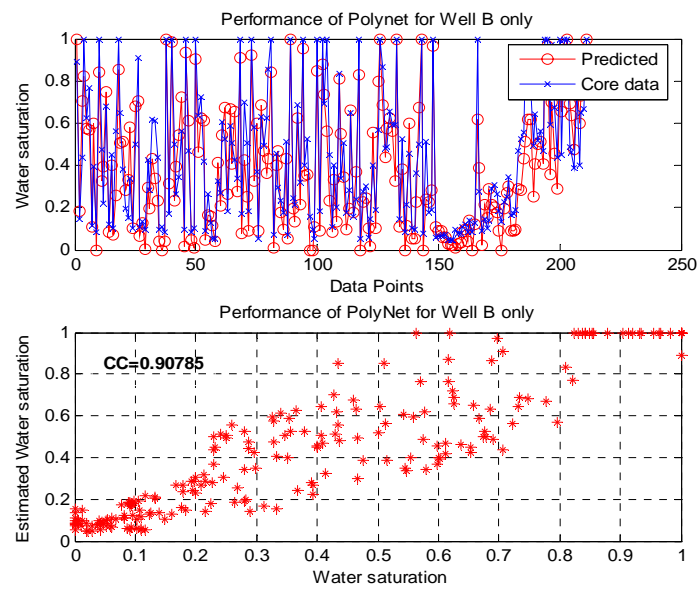


Figure B20: Scatter plot for estimated Water Saturation versus Core Water Saturation using Polynet on Well B

## APPENDIX C: Result of FN with reduced inputs from the combined well A and B

Table C1: The RMSE and correlation coefficient of different basis functions used for porosity (Three predictors: DT, NPHI, RHOB)

MODEL		Training Set		Testing Set	
Function	N.Wght	RMSE	CC	RMSE	CC
Fourier	11	0.0296	0.9138	0.0244	0.9354
Exponential	12	0.095	0.9141	0.0243	0.9355
Polynomial	6	0.030	0.9110	0.0247	0.9313
Logarithm	7	0.030	0.9104	0.0248	0.9310

Table C2: The RMSE and correlation coefficient of different basis functions used for water saturation estimation (Four predictors: NPHI, RHOB, GR, RT)

MODEL		Training Set		Testing Set	
Function	N.Wght	RMSE	CC	RMSE	CC
Fourier	14	0.1105	0.9519	0.0856	0.9705
Exponential	14	0.1106	0.9524	0.0844	0.9713
Polynomial	11	0.1187	0.9443	0.0824	0.9728
Logarithm	11	0.1172	0.9465	0.0810	0.9738

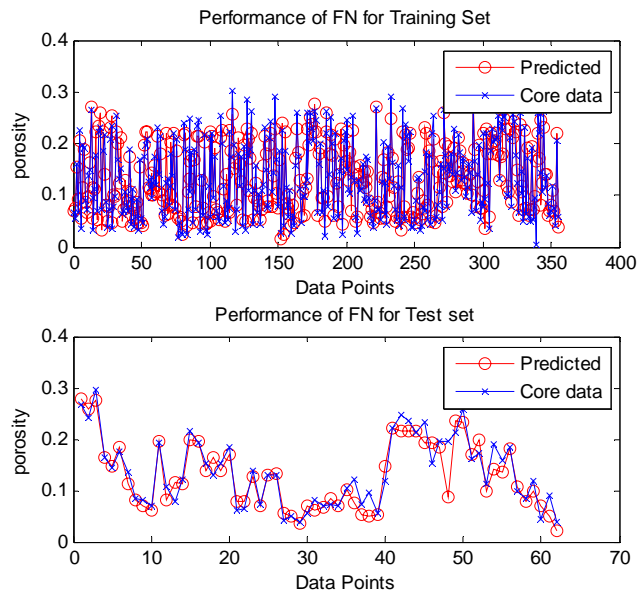


Figure C1: Performance plot for estimated Porosity versus Core Porosity using three inputs FN on combined well

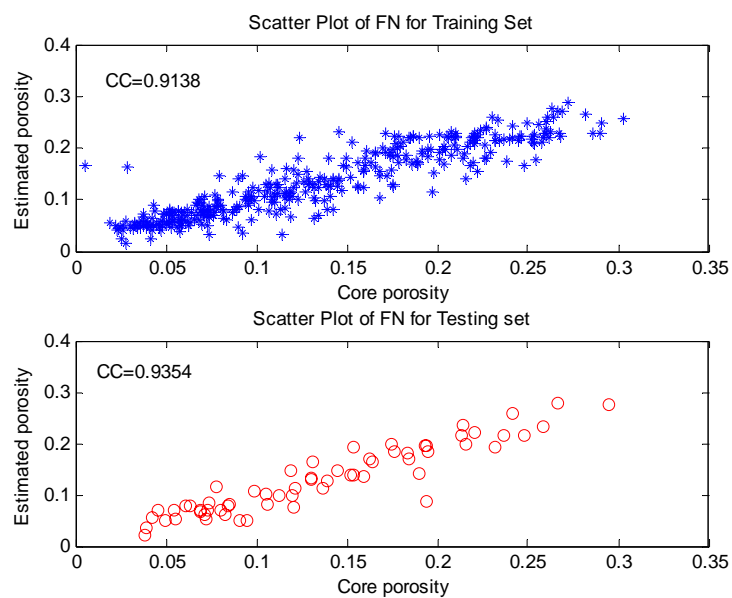


Figure C2: Scatter plot for estimated Porosity versus Core Porosity using three inputs FN on combined well

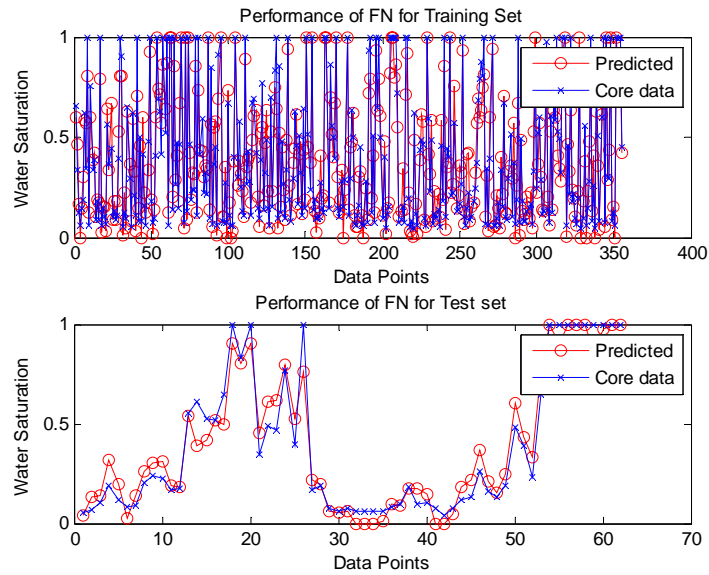


Figure C3: Performance plot for estimated Water Saturation versus Core Water Saturation using four inputs FN on combined well

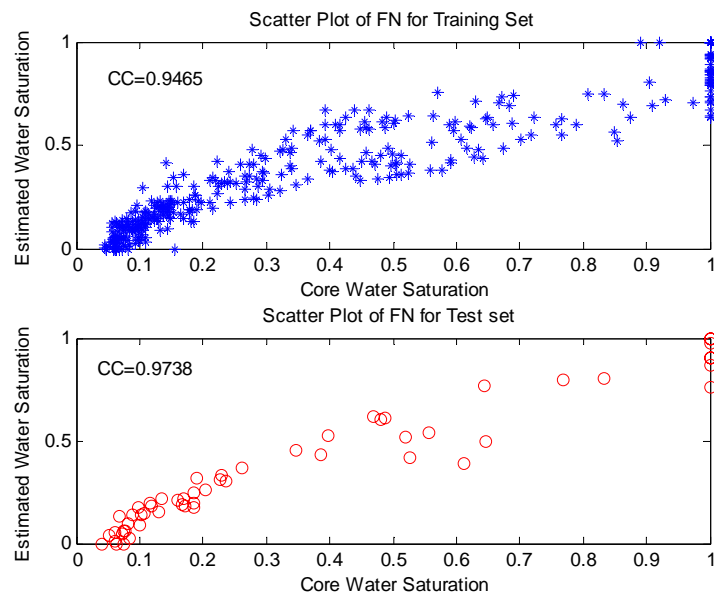


Figure C4: Scatter plot for estimated Water Saturation versus Core Water Saturation using four inputs FN on combined well

## VITA

### Personal Information

Born: 1974, Lagos, Nigeria

Marital Status: Married

### Education

- B.S., Electrical/Electronics Engineering, 2001 University of Ibadan, Ibadan, Nigeria,
- M.S., Computer Science, 2007, University of Ibadan, Ibadan, Nigeria,
- M.S., Systems Engineering, Control and Instrumentation Engineering option, Feb.2009  
King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia

### Research Interests

I have broad interests in optimization, artificial intelligence, and Modern Control Systems

### Publication

- Ahmed Adeniran, Moustafa Elshafei and Gharib Hamada. “Functional networks softsensor for formation porosity and water saturation in oil Wells” IEEE International Instrumentation and Measurement Conference, I2MTC May, 2009.

### Memberships

International Society of Automation (ISA) – Student Member