THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

JOHN AND WILLIE LEONE FAMILY DEPARTMENT OF ENERGY AND MINERAL ENGINEERING

DEVELOPMENT OF AN ARTIFICIAL NEURAL NETWORK FOR PREDICTING FISHBONE WELLBORE PERFORMANCE IN TIGHT GAS SANDS

ERIC DAVID SCHUMACKER

SPRING 2014

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Petroleum and Natural Gas Engineering
with honors in Petroleum and Natural Gas Engineering

Reviewed and approved* by the following:

Turgay Ertekin
Head, John and Willie Leone Family Department of Energy and Mineral Engineering
Thesis Supervisor
Honors Adviser

John Yilin Wang
Assistant Professor of Petroleum and Natural Gas Engineering
Faculty Reader

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

Fishbone wellbores are a type of multi-lateral wellbore structure that can be applied to unconventional natural gas reserves such as tight gas sands and shale gas reservoirs to increase natural gas production and make the development of natural gas plays plausible in otherwise uneconomic conditions.

This study is aimed at developing an artificial neural network tool to forecast monthly production data for fishbone type wellbores in tight gas sand formation. Reservoir, fluid, and wellbore parameters, as well as their resulting monthly production data generated with reservoir simulation software, are used to train the ANN model. The resulting model is able to predict a combination of outputs for any parameter combination within the range of training.

Three neural network structures will be designed. The first, described above, is a typical reservoir depletion study, with reservoir rock and fluid parameters known along with an existing wellbore geometry. In this case, monthly production data will be predicted by the neural network tool. The second structure will focus on wellbore geometry optimization, with reservoir rock and fluid parameters known, along with a desired production target. In this second case, possible wellbore geometry to yield aforementioned production targets will be predicted by the tool. In the final case, a formation test application, a known wellbore geometry along with early production data will be provided as known inputs to the tool. The tool will predict possible reservoir rock and fluid properties responsible for the given production data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge the help, support, and counsel of Dr. Ertekin throughout my years in the College of Earth and Mineral Sciences. In all of my questions, whether regarding career path, technical problems, or meetings regarding my research progress, he was always able to take time out for me in a schedule that could not have been busier.

I would like to thank my Fiancé, Qianwen Ma, who is supportive of me in every aspect of my life. By my side during the long, late hours in the Hosler lab, she gave me the support and endurance to bring this project to the level that it has been developed to today. I would also like to thank my father, David, who instilled in me the unquenchable drive for success and improvement.

I am truly grateful for the advice and support of Sarath Ketineni. His guidance and help proved truly fundamental to my success in the development of my thesis. Additionally, thank you to the friendship of Quan Huang and Peidong Zhao, who were by my side throughout our time in the College of Earth and Mineral Sciences, and my experiences with them will stay with me a lifetime. Additionally, I would like to recognize other friends who helped me with my thesis: Taha Husein, Junjie Yang, Qian Sun, Burak Kulga.

Chapter 1

Introduction

Natural gas plays are one of the primary frontiers of the petroleum and natural gas industry. Natural gas itself is used commercially and domestically for energy and heat. There are other technically advanced applications coming to the market including liquefaction to LNG (liquefied natural gas), as well as gas-to-liquids projects in which natural gas can be converted to produce liquid petroleum products [13]. However, at the present, natural gas prices in the U.S. are depressed, and conventional natural gas recovery through horizontal wellbores is largely un-economic. Limited upside price potential [6] has somewhat limited the development of these unconventional plays. However, new technology is in existence, including massive slickwater fracturing [8] and multilateral fishbone wellbores [4], that are able to increase production from natural gas formations and therefore induce the plausibility of further development of these natural gas plays.

This paper will focus on reservoir development studies of multilateral fishbone type wellbores in tight gas formations. Traditional reservoir studies for observation of reservoir pressure depletion and infill drilling potential are time consuming and tedious, and must be repeated at multiple stages in the development of the reservoir. A new type of study will be explored in which an artificial neural network is trained based on a wide range of reservoir rock, fluid, and wellbore geometry parameters. After the successful training of the network based on real production simulation data, the tool will offer a fast, convenient way of gauging development progress, analyzing the pressure profile in the reservoir, and suggesting possible wellbore geometry and parameters in the case of infill drilling.

The concept of applying a neural network to hydrocarbon recovery has been proven by multiple previous studies [1] [2] [15], specifically by using a back propagation network with conjugate gradient training algorithms. The neural network structure is widely versatile, and can be applied to any problem in which there inputs and outputs.

The neural network is a model in which the system will "learn by example", and can be trained by a variety of inputs and outputs [14]. The network will self-adapt, and modification of parameters within the network will allow the accurate reproduction of outputs with the given inputs. After this stage, the network can be provided any inputs within its range of training, and the network will generate the respective outputs. In this study, inputs and outputs to the network system will include reservoir rock and fluid properties, wellbore geometry information, as well as daily production data for nine years of natural gas production, in intervals of 30 days.

A similar study has been conducted by Burak Kulga [2], in which a neural network system was applied to predict monthly production data from tight gas sand systems with multiple hydraulic fractures. This study will continue further in this line of research, again applying the neural network system to tight gas sand reservoirs, however, this time containing fishbone multilateral wellbores. This study will extend this concept further, by not only predicting monthly gas production, but also allowing the determination of reservoir formation and fluid parameters, as well as the determination of appropriate wellbore geometry selection for required gas production.

Problem Statement

In a natural gas recovery problem such as this, there are three major factors controlling the system. The first being reservoir parameters, including rock and fluid properties downhole. The second aspect of the problem includes wellbore geometry. Parameters of wellbore data include length of the wellbore, any multilateral branches, and subsequent lengths. Finally, the output of these two principle variables would be the final facet of the problem, that is, the gas production of the system.

Neural Network Structure A

The first neural network structure will attempt to model a typical reservoir development study scheme, in which a series of formation evaluation has been performed, to allow for the understanding of formation rock and fluid parameters. In addition, wellbore geometry has been determined and drilled. This network will take these two series of inputs, and will output the predicted daily gas production for nine years of natural gas production in monthly intervals.

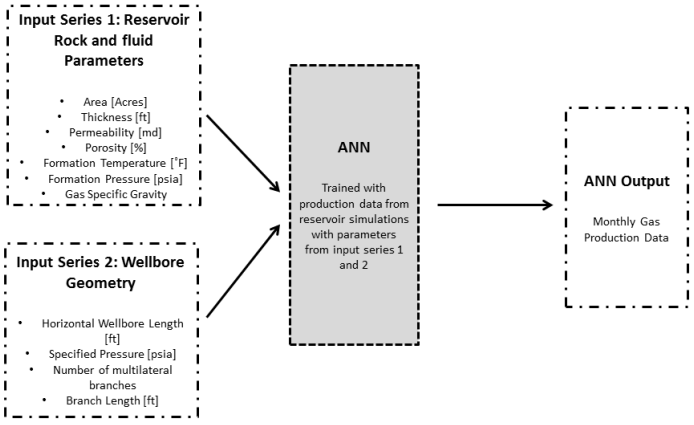

Figure 1-1 Network Structure A Schematic

Neural Network Structure B

The second network architecture will attempt to represent a well-testing and formation evaluation scenario. This case will serve as an inverse case to the previous network structure A. In this case, a wellbore structure is in place and is producing natural gas. Production data has been captured and tabulated. From these two series of inputs, a network will be trained to yield formation characteristics that would satisfy the aforementioned gas rates with the given wellbore geometry. Resulting data would allow for characterization of the formation. Results of this study would allow for planning and modeling of further development and depletion of the field.

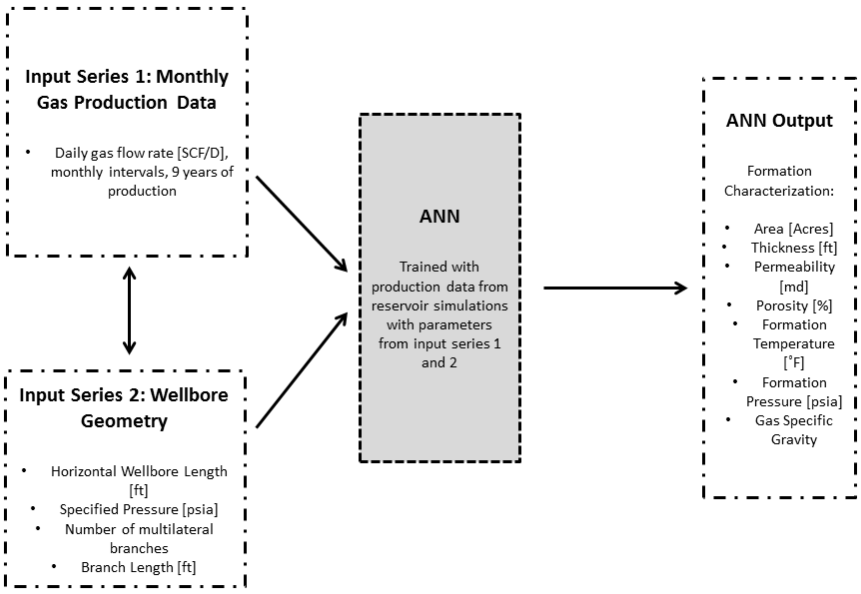

Figure 1-2 Network Structure B Schematic

Neural Network Structure C

The third neural network structure is another replication of the inverse to network structure A. However, in this third network structure, the two known inputs to the problem will include monthly production data over the nine years of natural gas production, along with a suite of reservoir formation evaluation data including rock and fluid properties. The output of the network will be an effective design for a horizontal wellbore with a number of lateral branches, that will be able to meet the respective production rates with the given reservoir rock and fluid parameters presented to the network.

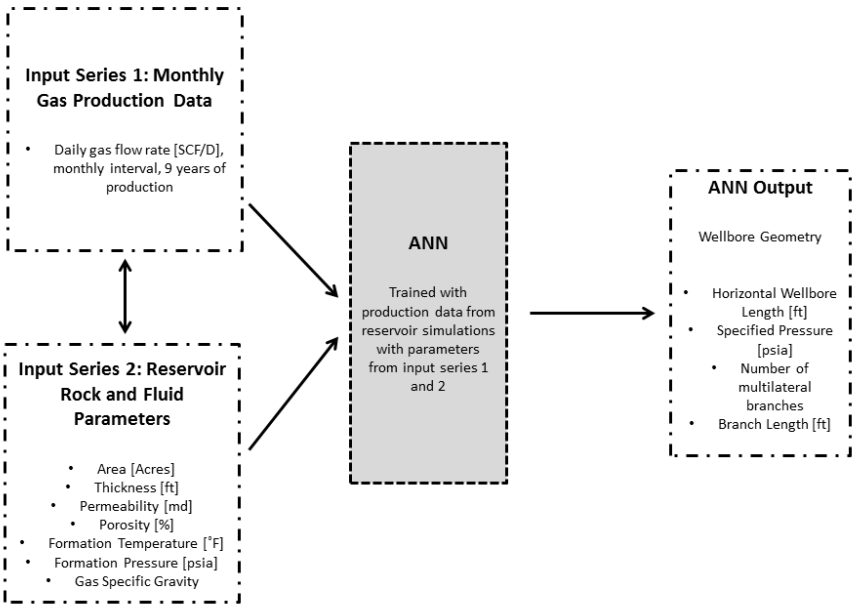

Figure 1-3 Network Structure C Schematic

Chapter 2

Literature Review

Unconventional Gas Reservoirs

Law and Curtis (2002) have defined tight gas reservoir as having a permeability of less than 0.1 millidarcies [6], with an effective gas permeability, on average, of less than 0.6 millidarcies [7]. Pores are connected irregularly, resulting in very low, if any, rates of gas flow [8]. Special methods are therefore required for production of gas from this type of reservoir. Tight gas sands often have very high stresses, due to great burial depth; therefore, tight gas reservoirs are often diagenetically transformed from their initial state at the time of deposition [8].

It is generally known that tight reservoirs cannot be produced economically without stimulation, such as massive slickwater hydraulic fracturing [6].

Initial production will typically be quite high and notice a steep decline, after which, follows a stable production period (with a low rate of decline), that may endure for years and even decades [8]. Water saturation has a large effect on gas permeability in tight gas reservoirs. At 50% water saturation, gas permeability is “significantly reduced”. At water saturations higher than this, there is minimal to no gas permeability [8].

The large volume and areal extent, as well as declining amount of conventional plays in the world today make unconventional plays, including tight gas reservoirs, a very attractive option. The low carbon impact of natural gas is another factor encouraging the leap into unconventional resources including natural gas [8]. An important current-day limitation on unconventional natural gas play development is the limited upside potential for the price of natural gas [6]. Therefore, the key aspect in the delineation between

successful unconventional gas reservoir exploitation and exceeded costs is in maximal recovery with minimal economic footprint. Technologies such as horizontal wellbores, massive slickwater hydraulic fracturing, and multi-lateral wellbores have changed the outlook on the economic feasibility of unconventional resources.

Horizontal Wellbores

A horizontal wellbore has three distinct features. It has a vertical section that will extend from surface to just above the formation of interest. It will then have a "kickoff point" and curve where the well will begin to deviate. The curve can have a radius of anywhere from 300 to 500 feet, with a final inclination of approximately 90 degrees, or running parallel within the reservoir. Finally, in the horizontal section of the wellbore, the well will extend horizontally through the reservoir, staying within the reservoir rock until the end of the wellbore is reached [10].

The first horizontal well was drilling in Texas in 1929 [10], with other countries including China and the Soviet Union making similar attempts in the 1950's [10]. The 1980's, however, is when horizontal wells began to be drilled in large quantities. This was due to the advancement of down hole mud motors, as well as down hole telemetry equipment. Currently, wells are being drilled with lateral lengths in excess of 8,000 feet. MWD and LWD tools have given operators the ability to more accurately stay within the target formation during drilling the lateral. Often, bottomhole assemblies will be configured with various down hole tools that are capable of measuring bottomhole temperature/pressure, and other drilling parameters such as weight on bit, rate of penetration, rotational torque, and so on.

Horizontal wells are more expensive to drill and complete than their vertical counterparts. However, in cases such as low matrix permeability, or water/gas coning is expected, horizontal drilling may be preferred or even required. For instance, such applications as shale gas in the Appalachian Basin require horizontal drilling to maintain profitable recovery of natural gas.

Horizontal wellbores offer the following benefits [10, 13]:

- Operators are able to develop a given reservoir with a smaller number of wells. This will translate to increased well spacing and fewer number of wells drilled. Due to increased surface area exposed to the target formation, a horizontal well is able to drain a significantly larger rock volume than a vertical well.
- Horizontal wells may delay the onset o production problems that could spell low production rates
- Horizontal wells may produce at 2.5 to 7 times the rate/reserves of vertical wells.
- A well that is cased into the producing formation will allow a lower density mud wen drilling the lateral section.

Fishbone Wellbores

Development of Tight Gas Reservoirs

In the last twenty years, horizontal wellbores have become a well-established technology, and a commonly used method to develop tight gas reservoirs.

In recent years, multilateral drilling technology has advanced enough to bring a new option to the table in economic tight gas reservoir development, termed fishbone well. A fishbone well is a multi-lateral well, with all laterals are placed in the same pay zone [3]. Therefore, the drainage area open to the formation is increased from the single wellbore running through the payzone, to the main lateral in addition to the each of the branches diverging from the main lateral. Increased drainage area in fishbone wells has proved to allow high single well production rates and enhanced recovery [4].

Multi-lateral technology and fishbone wellbores are a new technology, and currently much focus is placed on the technological development of the multilateral well. Up to 1995, there were less than 50 multilateral wells. Since 1995, however, there have been hundreds drilled, and even more planned [5]. Multilateral wells are now possible due to advanced drilling technologies, including directional and horizontal drilling techniques, drilling equipment, and coiled tubing drilling [5]. Currently, multilateral completions is the bottleneck to multilateral development, with technology in this area yet to be fully explored and developed.

Multi-Lateral Wellbores and Fishbone Wellbores

A multilateral well is "a single well with one or more wellbore branches radiating from the main borehole", as defined by Schlumberger [5]. The world's first multi-lateral well was drilled by a Soviet drilling engineer by the name of Alexander Grigoryan [5]. Following the idea that production will increase with increasing effective wellbore radius, Grigoryan postulated that a wellbore with many branches in the target formation would also increase surface area open to flow, and therefore increase production [5]. This was the first concept of the fishbone well. Grigoryan's multilateral, named "Well 66/45", performed extremely well. Well 66/45 had a main vertical wellbore drilled to 575 meters (1886 ft), after which, nine producing branches were drilled into the target formation, with individual branch lengths of 80 to 300 meters (260 to 1000 ft) [5]. Compared to other traditional vertical wells in the same field, Well 66/45 was able to contact over 5 times the pay thickness, and produced 17 times the amount of oil [5]. It was slightly more expensive, at about 1.5 times the cost of other wells in the area [5].

There exist two main types of multilateral wells. That is, vertical wells with various horizontal laterals extending into different reservoir layers; and horizontal well bores with horizontally spread laterals. Multilateral wells are classified based on "TAML", "Technology Advancement for Multi-Laterals", referring to the complexity of the branch junction [11, 4, 5]. This rating systems features six levels, labeled '1' through '6', and additionally a '6*S*' category. Level 1 represents an openhole main borehole, with an open and unsupported junction and branch [11, 4, 5]. Level 2 is for a wellbore with a cased and cemented main wellbore, with an openhole lateral (or with a slotted liner hong-off in the open hole). Level 3 features a cased and cemented main wellbore, with a cased lateral, but not cemented. Lateral liner will be anchored to the main wellbore, but not cemented.

Level 4 will have a main wellbore and lateral both cased and cemented. In level 5, pressure integrity is maintained at the junction and through the lateral through the use of completion equipment. Level 6, junction has pressure integrity, with casing, no need/reliance on completion equipment. The subcategory level 6 features a downhole splitter that will divide the main wellbore into two laterals [11, 4, 5].

A fishbone well is a structure with a main horizontal wellbore, with horizontally spaced branches, or laterals, extending off in a tangential direction from the main wellbore. All the branches in a fishbone wellbore structure are intended to target one reservoir interval.

The use of a fishbone wellbore structure allows [5]:

- Increased Production Rates
- Improved Hydrocarbon Recovery
- Maximized Zonal Recovery
- Reduced Drilling and Completions Costs

Where fracture structure is unknown, a fishbone wellbore may increase chances of contacting and connecting natural fractures pre-existing in the rock formation [5].

Fishbone Wellbore Parameters

Later referenced in Chapter 5, fishbone wellbores have several critical parameters that can influence production rates, including branch number and branch deviation angle, branch length, and the distance between branches (referred to as branch spacing). A study done in optimization of fishbone wellbore parameters in thin layer fishbone wellbores observed that production rates will increase with increasing number of branches, however,

with a branch number greater than four, there is little additional benefit [12]. Every additional branch will incur additional drilling time and drilling cost, reducing possible benefits from increased production after completion of the well. Similarly, it was noted that production does not increase with a branch angle higher than about 30 degrees [12]. For both branch length and spacing between branches, the main factor dictating these parameters are safe well operation. An optimal branch length has been observed as 100 to 200 m (330 ft to 660 ft), and an optimal well spacing of 80 to 150 m (250 to 500 ft) [12].

Artificial Neural Networks

The lowest level and most fundamental building block of a neural network architecture is the concept of a neuron.

To illustrate the architecture of the neuron, the one-input neuron should be considered. In this case, a single scalar input will be multiplied by a scalar weight. This product will be added to another input (in this case, 1) that is multiplied by a bias. Following, the sum of these two products will be input to a transfer function, which will output the output of the neuron/network. In this case, the bias and the weight are both adjustable parameters that will be alternated and modified as the network is trained, and can be adjusted according to a variety of user-selected learning rules [17]. The specific quantity of inputs and outputs provided to and from the network will be dependent on the external specifications of the problem [17]. Theoretically, any problem that has inputs and outputs can be described by a neural network architecture. In this problem, various reservoir and well parameters, as well as gas production rates output by the reservoir-well system, are conveniently described as inputs and outputs to the neural network.

Moving upwards in complexity from the single input neuron, next should be considered neurons with multiple inputs. In this case, each of the inputs will be multiplied by the weight, and added together along with the bias. This entire value will be sent to the transfer function, generating the output of the neuron [17]. The neuron output will again be a scalar. Addition of neurons would mean that the output of the network is no longer a scalar, but is instead a vector. In this case, multiple neurons on the same level is known as a 'layer' of neurons. Here, every input will be sent to every neuron. That is to say, every input of the input matrix will be sent to each neuron. At the neuron, the values will be multiplied by the weight of the neuron, summed, and sent to the transfer function, after which will yield the output. There will be as many outputs to the network as there are neurons in the output layer.

Additional layers can be added. Until this point, the network structure developed in this overview has included an input layer, as well as an output layer. This type of network, with a series, 'layer', of neurons is known as the Perceptron Network. Neural networks can also feature 'hidden' layers that will connect between the input matric and the output layer. In this way, the hidden layers' outputs will never be seen by the network user, but will only effect the internal calculations performed by the network. This case will function in the same way as before. Each of the inputs will be sent to each of the neurons in the first layer. They will be operated on by the neurons, being multiplied by each neurons' weight, as well as bias added after summation. Finally, this value is given as input to the transfer function, whose output consists of the output of the neuron. After all outputs are found, each of the outputs from every neuron are then sent as input to each of the neurons from the next layer. The process is repeated, with inputs being operated on by a new set of weights and biases,

and output generated by another transfer function. Finally, after an unknown, user-defined number of layers, the final output layer will be reached, and network outputs generated [17]. Multi-layer networks make for a large amount of creativity, design, and testing when it comes to network architecture. Networks with multiple layers are more powerful than networks of a single layer [17]. Able to have two different transfer functions, the opportunity to have a series of parameters able to be trained through learning rules (weights and biases), enables the network to have a better ability to learn and reproduce any given function. To increase accuracy and performance of the network, additional neurons may also be added. Additional neurons will provide the network with more flexibility, because it will have more parameters that it is able to optimize [16]. This type of network, with multiple layers of neurons with integrated operation is known as the Multiple Layer Perceptron (MLP) network [14].

Other types of networks include Radial Basis Function network, Kohonen's Self-Organizing Feature Map, ART – Adaptive Resonance Theory network, Hopfield network, BAM – Bidirectional Associative Memory network, Counter-Propagation network, and the Cognitron & Neo-Cognitron networks [14].

Transfer Functions

The transfer function in a network will take the summation of the inputs and respective biases as an input, and the output of the function will be the output of the neuron. There are several types of transfer functions, which will each be discussed briefly [17].

- Hard Limit. With an input of less than zero, the function will output 0. With an input of greater than or equal to zero, the function will output 1.

$$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

- Symmetrical Hard Limit. With an input of less than zero, a function value of -1 will be assigned. With an input of greater than or equal to zero, a function value of 1 will be assigned.

$$f(x) = \begin{cases} -1, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

- Linear. With the linear transfer function, the function value will be equal to the input value.

$$f(x) = \{x\}$$

- Saturating Linear. With an input of less than zero, a function value of 0 will be assigned. With an input value greater than or equal to zero and less than or equal to one, a function value of the input value itself. With a function value of greater than one, a function value of one will be assigned.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & 0 \leq x \leq 1 \\ 1, & x > 1 \end{cases}$$

- Symmetric Saturating Linear. With an input of less than negative one, a function value of negative one will be output. For an input value of less than or equal negative one to a value of less than or equal to one, a function value of the input value itself will be output. For an input value of greater than one, a function value of one will be output.

$$f(x) = \begin{cases} -1, & x < -1 \\ x, & -1 \leq x \leq 1 \\ 1, & x > 1 \end{cases}$$

- Log-Sigmoid. The Log-sigmoid transfer function will take the input value and contract its' value to between zero and one.

The log-sigmoid function is written as:

$$s_c(x) = \frac{1}{1 + e^{-cx}}$$

Where c is an arbitrary constant. As can be seen from the figure below, the value of c will determine the shape that the sigmoid will take. As can be seen from the figure, increasing values of c will cause the sigmoid function to closer and closer represent the step function.

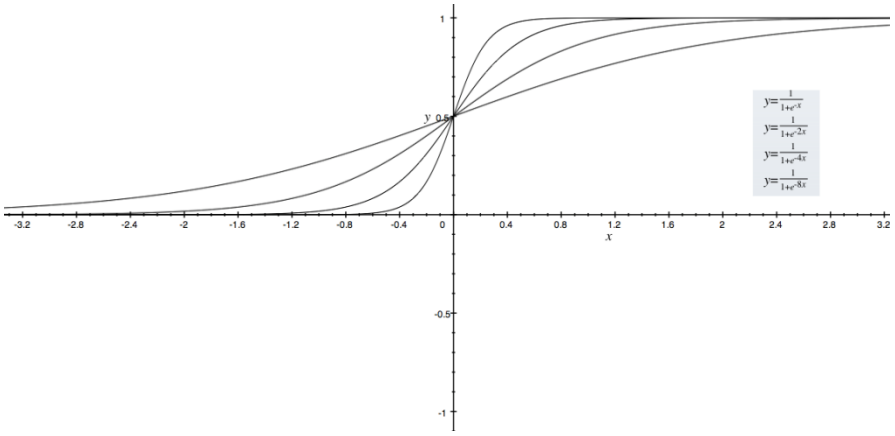

Figure 2-1 Log-Sigmoid Function

- Hyperbolic Tangent Sigmoid. The tangent sigmoid transfer function is similar to the log-sigmoid transfer function, except that in this case, the output of the function will scale any given input to a value between negative one and one.

  The hyperbolic tangent-sigmoid transfer function is written as:

$$tansig(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

  Addition of a constant in from of the input variables will again, as stated with the log-sigmoid function, will influence the severity of the slope between negative one and one.

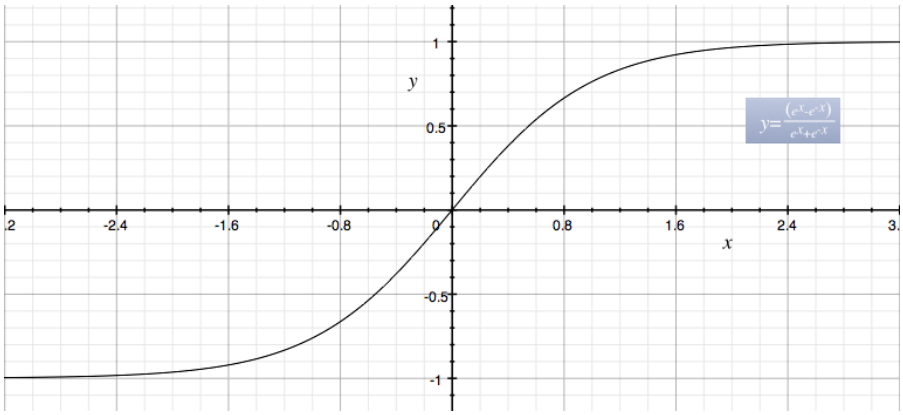

Figure 2-2 Tan-Sigmoid Function

- Positive Linear. With an input of less than zero, the function value will be zero. With an input of greater than or equal to zero, a function value of the input itself will be output.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

- Competitive. The competitive transfer function is a function that will compare values with other neurons in the same layer. For all neurons in a layer, the function will return a value of one for the neuron with the highest input value. For all other neurons, the function will return a value of zero.

$$f(x) = \begin{cases} 1, & MAX(x) \\ 0, & All\ other\ neurons \end{cases}$$

This research will focus on specifically the log-sigmoid and tan-sigmoid transfer functions. For use with backpropagation networks, transfer functions must me differentiable [14]. As such, these two transfer functions lend themselves well to use with a backpropagation network.

Training

After the network structure has been determined, training can begin, at which point the inputs will be given to the network, operated on, and output by the various neurons and layers of the network. A function, called the 'performance function' will allow the network, with the use of learning rules, to know to what extent to adjust the weights and biases of the neurons.

Learning Rules

As mentioned above, the weights and the biases of every neuron are modified through training according to a user-selected set of learning rules. That is, a *learning rule* is simply defined as a procedure for modifying the weights and biases of a network [17].

There are three basic types of learning rules for neural networks, including *supervised learning, reinforcement learning,* and *unsupervised learning* [17]. In supervised learning, a training set is utilized to provide an example of proper network operation. Network outputs will be compared against the targets, and the network weights and biases will then be adjusted based on this comparison. Reinforcement learning is similar to supervised learning. However, in reinforcement learning, the network is not provided with a set of examples, but instead each output is provided with a grade of rating regarding network performance. From here, the network is able to update its weights and biases. In the final learning rule type, unsupervised learning, there is no ideal network output available. In this case, network weights and biases are solely modified based on inputs supplied to the network.

Some important types of learning rules include Hebbian, Perceptron, Delta, Least Mean Square (Widrow-Hoff), Outstar (Grossberg), and Winner Takes All [14].

The backpropagation Algorithm

Backpropagation algorithm will adjust the network's weights and biases in the direction of steepest descent, which is the negative of the gradient [16]. In the backpropagation algorithm, computations will be performed backwards throughout the network, and is derived from the chain rule, calculus [16]. A single algorithm iteration can be written in the following fashion:

$$x_{k+1} = x_k - \alpha_k g_k$$

Where x_k is representative of the weights and biases of the current iteration, α_k is representative of the learning rate of the algorithm, and g_k is the gradient of the current iteration. The algorithm can be implemented in one of two separate ways: batch mode, as well as incremental mode [16]. In batch mode, the entire training set will be applied to the network before the update of the weights and biases. In the incremental mode, however, gradient computation and weight update will take place after each and every input is submitted to the network.

There are various other forms of backpropagation with faster training algorithms, and they include *Variable Learning Rate, Resilient Backpropagation, Conjugate Gradient Algorithms, Quasi-Newton Algorithms, Levenberg-Marquardt, Reduced Memory Levenberg-Marquardt* [16]. This study will focus on the use of conjugate gradient algorithms. Previous studies done at The Pennsylvania State University [1] [2] [15] have suggested that significant results were achieved when using this training algorithm. In the conjugate gradient training algorithm, searches are performed along the conjugate directions. This will yield a faster convergence [16] [17]. In the conjugate gradient training algorithm, the learning rate (which will be applied to control the length of the weight update) will be

adjusted during each individual iteration [16]. Conjugate Gradient Training Algorithm is comprised of four major different learning techniques:

- Scaled Conjugate Gradient, *trainscg*

  This training algorthim combines portions of technique used for Levenberg-Marquardt algorithm (specifically, the model-trust region approach), and applies it to the conjugate gradient approach. In the scaled conjugate gradient algorithm, no line search is performed for every iteration. However, it may take a larger number of iterations for convergance to be reached. There is a balance, however, because every iteration will be considerable shorter. The intense line search, repeated forevery iteration, will be avoided [16].

- Fletcher-Reeves Update, *traincgf*

  In the Fletcher-Reeves Update training algorithm, searching is started at the negative of the gradient on the first iteration. That is, $P_0 = -g_0$. Afterwards, a line serach will be performed. This is to calculate the optimal distance to move in the search direction. After, in order to determine the next search direction, the newest steepest descent (negative of the gradient) is combined with the previous search direction [16]:

  $$p_k = -g_k + \beta_k p_{k-1}$$

  For Fletcher-Reeves Update, the constant β_k is calculated as the ratio of the normal squared of the current gradient to the norm squared of the previous gradient [16]:

  $$\beta_k = \frac{g_k g_k}{g_{k-1} g_{k-1}}$$

- Powell-Beale Restarts, *traincgb*

  In Powell-Beale Restarts, if the condition: $|g_{k-1}g_k| \leq 0.2|g_k|^2$is fulfilled, the search direction will be reset to the negative of the gradient [16].

- Polak-Ribiere Update, *traincgp*

  In Polak-Ribiere Update, the search direction for every repetition of the algorithm is the same as in Fletcher-Reeves Update, as calculated by:

  $$p_k = -g_k + \beta_k p_{k-1}$$

  However, the constant β_k will be calculated using the inner product of the previous gradient change, Δg_{k-1}, with the current gradient g_k, divided by the norm squared of the gradient of the previous iteration [16]:

  $$\beta_k = \frac{\Delta g_{k-1} g_k}{g_{k-1} g_{k-1}}$$

Chapter 3

Generation of Training, Validation, and Testing Sets

Combinations of reservoir, fluid, and wellbore properties were generated using CMG®-IMEX® black-oil simulator. The following assumptions were utilized for initial reservoir conditions:

- 3 Dimensional Model (three layer)
- Square reservoir, non-uniform grid distribution
- Dry Gas Reservoir
- Oil and Gas Saturations both zero ($S_o = 0$; $S_g = 0$)
- One horizontal well in the center of j direction
- Production with specified pressure (P_{sf})

Models

In development of the ANN tool, two cases were selected. These two cases are a horizontal wellbore in a tight gas reservoir, containing no branches or fishbone-type well structure. The other case is horizontal wellbore in tight gas reservoir, containing a number of fishbone branches, with branches varying in length, number, and angle from the main lateral.

Case One: Single Horizontal Wellbore in Tight Gas Reservoir

For the development of a neural network tool for horizontal fishbone wellbores in tight gas reservoirs, a range of rock and fluid parameters were determined with the appropriate parameters to model a tight gas reservoir. MATLAB™ was then utilized to generate various combinations of these parameters.

In both cases, a number of assumptions were used in the development of the target tight gas reservoir. Two dimensional, square reservoirs, with non-uniform, dynamically sized gridblocks were modeled. The reservoir is a dry gas reservoir, with no oil saturation and connate water saturation. Table 5.1 contains the ranges of reservoir rock, fluid, and wellbore properties that were used to model all dry gas reservoirs for case one.

For case one, an important condition of parameter set generation was to limit wellbore length. Wellbore length was randomly generated from the parameters given below (from 1,000' to 7,000'). But before the value was assigned, a check had to be employed to ascertain that wellbore length did not exceed the boundary length of the reservoir. The wellbore length maximum was set as 80 percent of total reservoir length. If the randomly generated value was less than this 80 percent value, it was accepted. However, if the code generated value did exceed this 80 percent limit, the wellbore length value was rejected and re-assigned as 80 percent of the reservoir grid block boundary length.

Parameter	Minimum Value	Maximum Value	Units
Area (A)	100	1,000	Acres
Thickness (h)	50	300	ft
Matrix Permeability (k)	0.0001	0.1	md
Porosity (φ)	5	20	%
Reservoir Temperature (T_i)	100	300	°F
Initial Reservoir Pressure (P_i)	1000	7000	psia
Specified Pressure (P_{sf})	14.7	0.5*Pi + 14.7	psia
Horizontal Wellbore Length (L_{wb})	1,000	7,000	ft
Gas Specific Gravity (γ_g)	0.6	0.8	

Table 3-1 Reservoir and Fluid Parameters, Case One

Case Two: Multilateral Horizontal Wellbore in Tight Gas Reservoir with Fishbone Branches

Case two is for a lateral wellbore with a varying number of fishbone branches drilled deviating laterally from the main wellbore. In this case, parameters Area (A), Thickness (h), permeability (k), porosity (φ), reservoir Temperature (Ti), Initial Reservoir Pressure (Pi), Specified Pressure (Psf), Horizontal Wellbore Length (Lwb), and Gas Specific Gravity (γ_g) were all held constant with the first case.

However, case two marks the addition of various parameters unique to the fishbone wellbore geometry. Chiefly among these are branch number (n), branch length (L_b), and angle of deviation from the main lateral (θ). The branch geometry will be fixed as a planar quadrilateral structure [4]. In this wellbore structure, the main wellbore lateral will extend laterally in the i-direction through the reservoir grid blocks in the center of the reservoir thickness (that is, layer 2 of three). In the fishbone well bore geometry, there are many parameters that will influence recovery from the reservoir. These parameters include number of branches, branch length, angle of branch deviation from main horizontal wellbore, and location of branch junction on the main horizontal wellbore (branch spacing). In this series of simulations, the following variables will be tested and controlled: branch number (0 to 2 branches), branch length (see Table 5.2 for parameter range specifications), and branch spacing on the main horizontal wellbore. The angle of branch deviation will remain constant at 45˚. This is due to the fact that a Cartesian grid block structure was imposed on the series of simulations performed in this research. Working under a time constraint, to maintain an acceptable speed of simulation execution, the grid block system was not developed at a fine enough resolution to allow the alteration of branch deviation angle.

In the parameter generation combination performed in MATLAB™, special consideration must be made for the dimensions of the branch geometry in relation to the area of the reservoir for the specific data set, as well as the wellbore length. In the MATLAB™ coding procedure, code was developed to automatically orient the placement of the branches at a given distant from the beginning of the main horizontal wellbore.

Branch length will be limited such as to not exceed the boundary of the reservoir. After random parameter assignment, a conditional parameter modification will trim the branch length such that it will end at the reservoir boundary, if it will originally exceed the reservoir boundary.

Parameter	Minimum Value	Maximum Value	Units
Area (A)	100	1,000	Acres
Thickness (h)	50	300	ft
Matrix Permeability (k)	0.0001	0.1	md
Porosity (φ)	5	20	%
Reservoir Temperature (T_i)	100	300	°F
Initial Reservoir Pressure (P_i)	1000	7000	psia
Specified Pressure (P_{sf})	14.7	0.5*Pi + 14.7	psia
Horizontal Wellbore Length (L_{wb})	1,000	7,000	ft
Gas Specific Gravity (γ_g)	0.6	0.8	
Number of Branches (n)	1	2	
Branch Length (L_b)	300	1000	ft
Branch Spacing (x)	0.1*L_{wb}	0.8*L_{wb}	ft
Branch Deviation Angle (θ)	45		°

Table 3-2 Reservoir, Fluid, and Wellbore Parameters, Case Two

Data Set Simulation and Processing

After the ranges of parameters for both cases were established, MATLAB™ code was employed to create a number of sets to be used for training and testing of the neural network. A different code was then used to generate CMG input batch files, to be used in the batch simulation of the different reservoir condition cases generated by the input files. This code was designed to receive an input of the reservoir, fluid, and wellbore properties generated in previous code, and generate an output batch file to conduct simulations and obtain an output of monthly production data from CMG. Appendix A contains all code used during this procedure, as well as a sample batch file script used as an input in CMG IMEX in the simulation of reservoir depletion with the previously descripted reservoir, fluid, and well bore geometry parameters.

The outputs of the batch generation (CMG_CASE1.m and CMG_CASE2.m) were then executed with IMEX in an automated fashion. Upon completion of each simulation, the resulting pressure depletion and production data for each case were generated in the same directory.

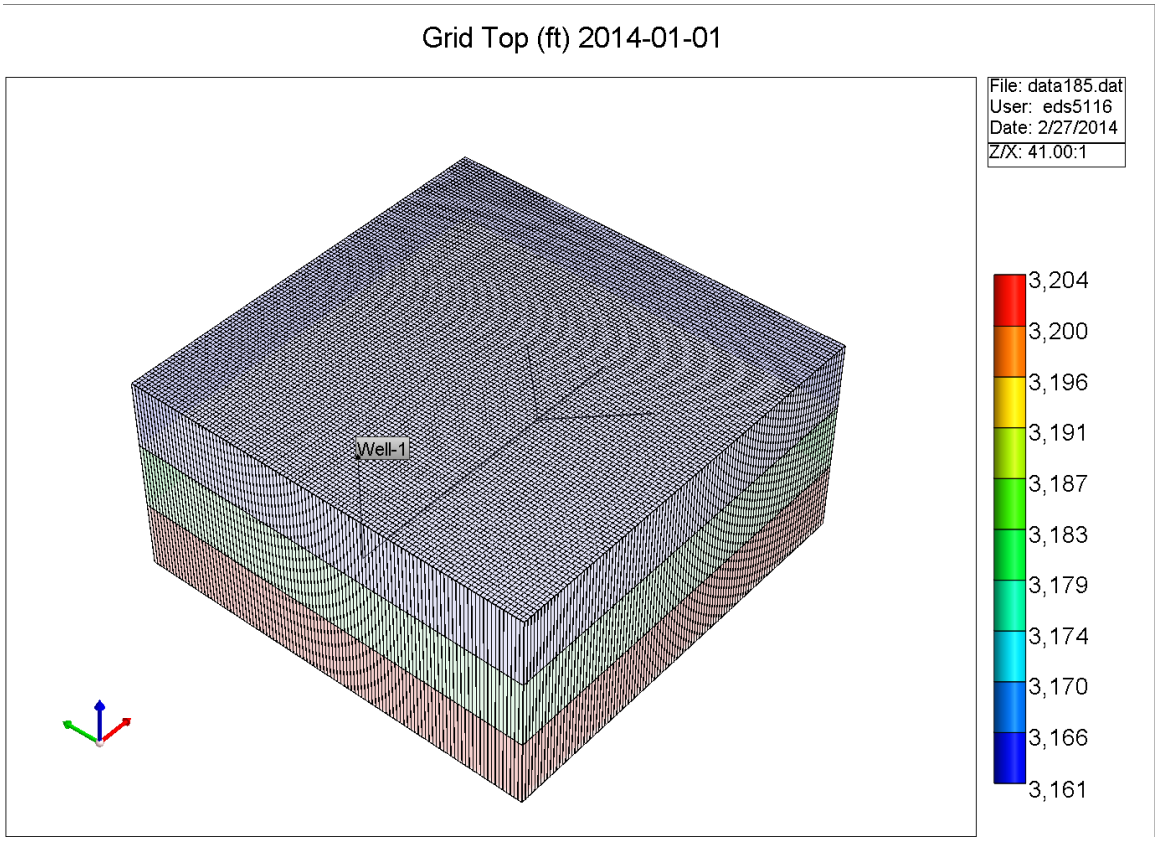

Figure 3-1 Data Set # 185, grid block structure and well bore geometry in IMEX[1]

[1] IMEX Black Oil Simulator used for reservoir pressure depletion simulation in this study. ©CMG Computer Modelling Group LTD.

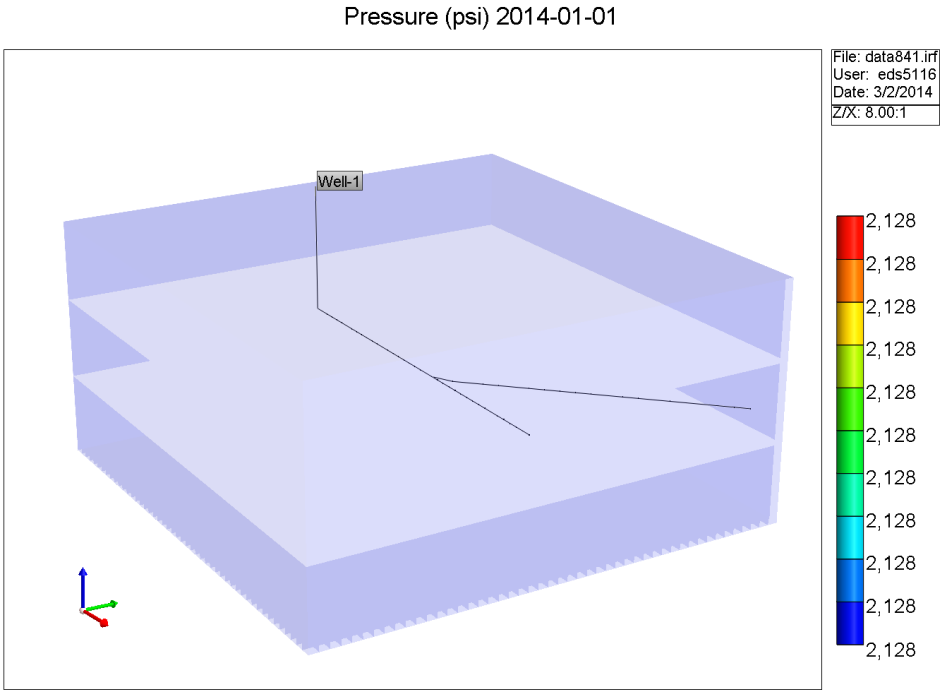

Figure 3-2 Data Set #841 Grid Block System Hidden to Reveal Wellbore Structure

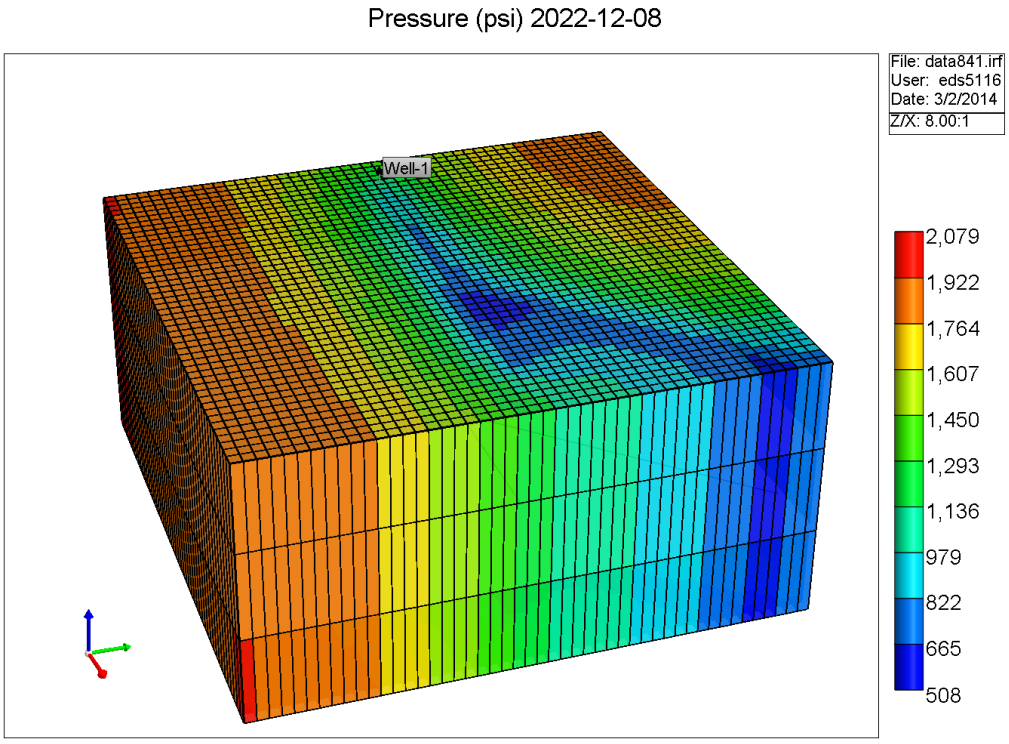

Figure 3-3 Data Set #841 Reservoir Pressure Depletion, Nine Years of Natural Gas Production

Following the successful simulation of each data set and the resulting production data output, procedure was executed to extract, compile, and output into format useable by the neural network. Within this step, the data sets were examined and cleaned. Several non-operational data sets were generated. These entire sets were removed. Additionally, due to the entirely random nature of set generation, there may have been combinations of parameters that are altogether entirely unfeasible to be applied in actuality. Several of these sets were also removed. Finally, data was also tabulated and output in excel for reference.

Chapter 4

Development of ANN Model

For case one, nine input variables will be used, including reservoir area (A), thickness (h), formation permeability (k) and porosity (φ), initial reservoir temperature (Ti) and pressure (Pi), formation gas specific gravity (γg), horizontal wellbore length (L_{wb}), and specified pressure (P_{sf}). The output of case one will include nine years (108 months) of daily production data, given in monthly intervals, accounting for 109 outputs for the network. Case two A will note the addition of three input parameters, including branch length, L_b, number of lateral branches, *n,* as well as the location of branch junction in terms of the main wellbore, *x.* The output for case two A will be identical to case one, using monthly gas production data as the main output.

Network A can also be known as a "forward-looking" case, because it will take two sets of parameters, and predict future time-dependent outputs. The next two cases to be developed in this research are "inverse" cases, in which this time dependent data will be used in prediction of multiple original properties. Network B will take input of all nine years, by month, of daily gas production data (109 inputs), along with five wellbore geometry inputs of main wellbore length L_{wb}, specified pressure p_{sf}, number of branches n, branch length L_b, and branch junction location on the main wellbore x (5 inputs, 114 inputs total), to predict reservoir rock and fluid properties, including area (A), thickness (h), formation permeability (k) and porosity (φ), initial reservoir temperature (Ti) and pressure (Pi), formation gas specific gravity (γg)(7 outputs). Network C will be another alteration of the inverse case. In this case, again all 109 gas rate data points will be included as inputs, along with all reservoir rock and fluid properties, including area (A), thickness (h),

formation permeability (k) and porosity (φ), initial reservoir temperature (Ti) and pressure (Pi), formation gas specific gravity (γg)(7 inputs, 116 total inputs). The outputs of this network structure will be wellbore geometry data, including main wellbore length L_{wb}, specified pressure p_{sf}, number of branches n, branch length L_b, and branch junction location on the main wellbore x.

In optimization of the neural network structure, several variables will be examined and varied through a series of multiple network training trials to determine the most effective network structure. Network parameters to be modified through experimentation include:

- The number of hidden layers
- The type of transfer function between layers
- The number of neurons contained in each layer
- Network types (feed-forward, cascade-forward)
- Training algorithm used
- Relationship of network inputs and outputs, including functional links

For initial trials, there will be one hidden layer employed. The number of hidden units, or neurons, will be set to half of the sum of the number of inputs and outputs. *Tansig* and *logsig* will be predominantly utilized as activation functions. This is in part due to reference to previous studies analyzing neural network application to hydrocarbon recovery noted that back propagation networks performed more effectively with transfer functions *logsig* and *tansig.*

In order to train a neural network, a large base of data (consisting of inputs and outputs) must be provided to the network. The data will be divided randomly, with user-defined percentage allocations to each set of data, into three sets. These sets are known as the training set,

the validation set, and the testing set. The largest portion of the data, about 70% to 80%, will be allocated to the training set. The inputs and the outputs of the training set will be used by the network and its learning rules and training algorithms to manipulate network parameters to yield the outputs from the inputs, memorizing the data. The validation set will constantly be monitored, but not trained, to represent approximate performance the network yields when exposed to untrained data. Finally the training set, the smallest portion of the data (roughly 5% to 10%), will be seen by the network only once, at the end of training. The testing set will test the network with data points it has never been exposed to, and network prediction error will be calculated from statistical methods, comparing the actual outputs (in this case, outputs generated in the numerical reservoir simulation model) and the outputs predicted by the ANN.

Various trials will be executed, in which the previously stated network properties will be altered and shuffled, to find a network structure that is most suitable to each individual application.

In general, it was found over the course of different network trials, that a cascade-forward back propagation network, *newcf,* yielded better results for all network structures where output consisted of time dependent outputs (Network Structure A, output of gas production schedule).

Case One

The goal of the network outputs in this study has been fixed as the output of monthly production data for nine years of cumulative gas production. Outputs were left unmodified as the daily gas rate values (in units of SCF per day) monthly for nine years, yielding a total of 109 network outputs.

Table 4-1 Network Input and Output Structure, Case One

Inputs (9)	Outputs (109)
• Area, A [Acres] • Formation Thickness, h [ft] • Formation Permeability, k [md] • Formation Porosity, φ [%] • Initial Pressure, Pi [psia] • Initial Temperature, Ti [˚F] • Formation Gas Specific Gravity • Horizontal Wellbore Length, L_{wb}, [ft] • Specified Pressure, P_{sf}, [psia]	• Daily Gas production rate, Day 1, q_{gsc} [SCF/D] • … • … • Daily Gas Production rate, Day 3240, q_{gsc} [SCF/D]

In the trials of case one, a cascade-forward back propagation network, *newcf*, was implemented. To assess network performance, the function *msereg*, mean squared error with regularization performance, will be utilized. As described above, the initial trial for case one utilized a single hidden layer, with 59 neurons. Increasing the number of layers while decreasing the number of neurons per layer (for example, from one layer with 59 neurons to two layers with 40 and 20 neurons, respectively) resulted in a much slower training, and a larger error from the values generated by CMG. It was found that due to the

large number of inputs and outputs, a larger number of neurons per layer was beneficial in increasing the strength of the network.

Table 4-2 Case One Neural Network Trials

Trial Number	Iterations to goal	Hidden Layers	Neurons per layer	Training Method	Transfer Functions	Mean Percentage Error
1	5212	1	59	newcf	tansig	1.73%
2	10052	2	40, 20	newcf	tansig, tansig	2.18%
3	3659	2	50, 30	newcf	tansig, logsig	1.97%
4	2912	2	20, 10	newcf	tansig, logsig	1.82%
5	1381	1	59	newcf	logsig	1.76%
6	3618	2	40,20	newff	tansig, tansig	1.70%
7	1594	2	42, 24	newff	tansig, logsig	1.81%
8	3245	2	42, 24	newcf	tansig, logsig	1.90%
9	1037	1	55	newcf	logsig	1.76%
10	1090	1	50	newcf	logsig	1.64%
11	7583	1	50	newcf	tansig	1.90%

The final network design for case one included a cascade forward backpropagation network, *newcf,* with scaled conjugate gradient backpropagation, *trainscg.* One hidden layer is used, with 50 neurons. *Logsig* transfer function is used. It was found that with one hidden layer and a large number of outputs, *logsig* yielded significantly better results than any combination of transfer functions when two hidden layers were utilized. *Logsig* also performed significantly better than the use of transfer function *tansig* when only one hidden layer was employed. This network configuration yielded the best combination of lowest

mean percentage error over all the testing sets, while also achieving the best training performance of all trials conducted.

Case Two, Network Structure A

This network structure is an analog of the network structure applied in case one, extended to include the capability of handling multiple lateral branches including the main wellbore. As with Case One, the target output of this network is the nine year, monthly production schedule (gas rates, in SCF/Day) generated by a reservoir with given inputs of rock and fluid properties, along with a given input of a wellbore geometry. For this problem, there were 12 inputs and 109 outputs.

Table 4-3 Network Input and Output Structure, Case Two A

Inputs (12)	Outputs (109)
• Area, A [Acres] • Formation Thickness, h [ft] • Formation Permeability, k [md] • Formation Porosity, φ [%] • Initial Pressure, Pi [psia] • Initial Temperature, Ti [˚F] • Formation Gas Specific Gravity • Horizontal Wellbore Length, L_{wb}, [ft] • Specified Pressure, P_{sf}, [psia] • Number of Lateral Branches, n • Branch Length, L_b, [ft] • Location of Junction on main Wellbore Lateral, x, [% wellbore length]	• Daily Gas production rate, Day 1, q_{gsc} [SCF/D] • … • … • Daily Gas Production rate, Day 3240, q_{gsc} [SCF/D]

As above, initial network trials were begun with a single hidden layer of neurons, with number of neurons equal to half of the sum of inputs and outputs. In this case, testing

was begun with a single layer of 70 neurons. A cascade forward back propagation network was primarily tested, noting its' effectiveness with the prior, similarly structured, case. It has been noted that a cascade-forward network will be better suited to handle data with time dependent outputs. In this case, with data outputs being directly dependent on time, cascade-forward networks seemed to be the most logical place to begin testing. As trials were continually ran, it was apparent the respective strengths and weaknesses of different network configurations. For this case, it was clear that a two hidden-layer network performed far better than either one hidden layer or three hidden layers. Feed forward networks were tested multiple times, however cascade-forward still excelled. Finally, the transfer function combination of *logsig* followed by *tansig* consistently yielded the best performance and error results. Performance function used was *msereg,* mean squared error with regularization. Below, Table 5-4 as documented a selection of the trials that were conducted with network structure A.

As can be seen highlighted in yellow in Table 4-4, the final network structure selected for prediction of monthly production data from reservoir properties and wellbore structure with multiple lateral branches was s a cascade-forward backpropagation network, *newcf,* with scaled conjugate gradient, *trainscg* backpropagation technique. Two hidden layers were chosen with 70 neurons and 40 neurons, respectively. The transfer functions found to function it the greatest performance were *logsig* followed by *tansig*

Table 4-4 Case Two A Neural Network Trials

Trial	Iterations	Hidden Layers	Neurons	Training Method	Transfer Functions	Testing Sets Averaged MPE	Stop
1	414	1	70	newcf	logsig	4.96%	Early Stopping Tech.
2	933	2	70, 30	newcf	logsig, logsig	3.33%	Early Stopping Tech.
3	3216	2	70, 30	newcf	logsig, tansig	2.28%	Early Stopping Tech.
4	823	2	50,20	newcf	logsig, tansig	6.50%	Early Stopping Tech.
5	4367	2	70, 40	newcf	logsig, tansig	2.34%	Early Stopping Tech.
6	3842	3	70, 40, 20	newcf	logsig, tansig, logsig	3.12%	Early Stopping Tech.
7	3983	2	70, 30	newcf	logsig, logsig	4.04%	Early Stopping Tech.
8	2547	2	70, 30	newcf	logsig, tansig	3.00%	Early Stopping Tech.
9	2296	2	70, 30	newcf	logsig, tansig	4.10%	Early Stopping Tech.
10	3082	2	70, 30	newff	logsig, tansig	4.70%	Early Stopping Tech.
11	3079	2	70, 40	newcf	logsig, tansig	2.76%	Early Stopping Tech.

Case Two, Network Structure B

The network structure of ANN B is a solution to the inverse problem initially proposed. In this problem case, gas production data is assumed to be a known input, along with the wellbore geometry and design data. The output of this network will be the various reservoir rock and fluid pararmeters, including area, thickness, permeability, porosity, initial reservoir temperature and pressure, as well as gas specific gravity.

Table 4-5 Network B Inputs and Outputs

Inputs (114)	Outputs (7)
• Daily Gas production rate, Day 1, q_{gsc} [SCF/D] • … • … • Daily Gas Production rate, Day 3240, q_{gsc} [SCF/D] • Main Wellbore Length, L_{wb} [ft] • Well Specified Pressure, P_{sf} [ft] • Number of branches, n • Branch length, L_b [ft] • Junction location on main wellbore, x	• Area, A [Acres] • Formation Thickness, h [ft] • Formation Permeability, k [md] • Formation Porosity, φ [%] • Initial Pressure, Pi [psia] • Initial Temperature, Ti [˚F] • Formation Gas Specific Gravity

The initial network structure design, as defined in the neural network literature review, will consist of one hidden layer, with neurons amounting to half of the sum of network inputs and outputs. The initial trial consisted of a single layer with 60 neurons. A feed-forward backpropagation network was utilized due to its relatively better performance when outputs are not time related. Backpropagation technique used was the scaled conjugate gradient, *trainscg.* For the learning function, *learngd*, gradient descent weight and bias learning function, was selected. Transfer function was the sigmoid function *logsig*, however, these parameters were altered in subsequent trials. The performance function selected for the trials was *msereg*, mean squared error with regularization.

In initial trials, validation and training results were very poor. The error of the training set saw continued reduction, however validation proved to be impossible, with the network consistently failing validation tests. The network type, backpropagation technique, number of

neurons and layers were all varied. To an extent, it was found that an increased number of neurons and layers would increase early time validation performance. However, after the network assed about 300 to 500 iterations, no matter the network configuration, validation performance was poor, and the network could consistently fail validation checks. This is due to the non-unique nature of the problem solution. A various combination of reservoir parameters may exist for a given gas production schedule. Therefore, network performance was measured through another criteria, using ANN generated reservoir parameters to generate another gas production curve, and compare results with the original gas production curve. Results will be discussed in Chapter 7.

Due to the non-unique nature of the solution to the structure of network B, it was determined whether or not functional links would need to be added to the network outputs to increase accuracy of the solution. To an extent, it was beneficial, however, overuse of functional links led to impossible reservoir parameter outputs. Without functional links in the output data sets, the network was able to generate parameters that were near the testing set values, in the same order of magnitude. However, with an excess of functional links applied, values were generated to different orders of magnitude (impossibly small reservoir area values, impossibly large porosity and permeabilities). As further discussed in Chapter 7, it was found that more consistent results were obtained without the use of functional links. A lower standard deviation amongst average mean percentage errors in all the parameters means a larger relative likelihood of the ability to apply a forcing function to generate more realistic outputs after network output. Additionally, the non-unique nature of the problems makes for a scenario in which parameter "error" does not even necessarily correlate to network inoperability, but instead, the prediction of one of many different possible solutions.

Table 4-6 Functional Links Utilized in Network B

#	**Network Output**
1	$\frac{A}{L_{wb} * n * L_b}$
2	$A * h$
3	$\frac{k}{L_{wb} * n * L_b}$
4	$A * h * \phi$
5	$\frac{P_{sf}}{P_i}$
6	$P_i * T_i$
7	$T_i * \delta_g$

A final network structure was determined as three hidden layers, containing 80, 60, and 40 neurons, respectively. Feed-forward backpropagation network was used with scaled conjugate gradient training algorithm. Mean squared error with regularization, *msereg*, was the performance function, and *learngd*, gradient descent weight and bias learning function was used. The transfer functions used between each layer were *logsig*, *logsig*, and *logsig*, respectively.

Case Two, Network Structure C

Table 4-7 Network Structure C Inputs and Outputs

Inputs (116)	Outputs (5)
• Daily Gas production rate, Day 1, q_{gsc} [SCF/D] • … • … • Daily Gas Production rate, Day 3240, q_{gsc} [SCF/D] • Area, A [Acres] • Formation Thickness, h [ft] • Formation Permeability, k [md] • Formation Porosity, φ [%] • Initial Pressure, Pi [psia] • Initial Temperature, Ti [˚F] • Formation Gas Specific Gravity	• Main Wellbore Length, L_{wb} [ft] • Well Specified Pressure, P_{sf} [ft] • Number of branches, n • Branch length, L_b [ft] • Junction location on main wellbore, x

Initial network structure for network C was a feed-forward backpropagation network with scaled gonjugate gradient backpropagation, *trainscg*. The mean squared error with regulariziaton, *msereg*, function was applied, with learning function gradient descent weight and bias function, *learngd*. A single hidden layer with 60 neurons was used. As can be seen from Table 5-7 above, there were 116 inputs and 5 outputs for the network.

Initial trials offered unsatisfactory results with failure of network validation testing. A number of training algorithms were employed in the trials, including a variety of conjugate gradient algorithms (Fletcher-Reeves Update *traincgf,* Polak-Ribiere Update *traincgp,* Powell-Beale Restarts *traincgb,* and Scaled Conjugate Gradient *trainscg*), as well as Bayesian Regularization *trainbr*, and Levenberg-Marquardt *trainlm*. Addtionally, the number of neurons and layers were varied. As mentioned above, an additional number of neurons and layers, providing more parameters for the network to optimize, increased network performance. However, convergence could not be reached after a certain point, and no matter how network

structure was shuffled between training algorithms, network structure, # of neurons and layers, training performance would not increase. Functional links were added. Addition of functional links immediate improved validation performance during early training. As training proceeded, it was noted that validation performance would still return to its previous poor behavior. This was again observed due to the non-unique nature of the solution. As such, two different testing procedures were used. Testing procedure one included using ANN C predicted wellbore geometry with the original reservoir parameters and analyzing production curve error. Testing procedure two consisted of quantifying the errors of the actual properties generated.

The final network structure was without functional links. While functional links led to better network training, the resulting properties were higher in error. Although unable to train the network to as small an error as with functional links, the network generating wellbore properties with the most plausible results was without functional links. The final structure selected for this problem structure was a feed-forward backpropagation network with Powell-Beale Restarts, *traincgb*. Mean squared error with regularization was the performance function, with the learning function *learndgm*, gradient descent with momentum weight and bias.

Chapter 5

Results and Discussion

Error Analysis of ANN Outputs

To assess network accuracy of reproduction of target outputs, a statistical average known as "Mean Percentage Error" will be used. This error is defined as:

$$MPE = \frac{100\%}{n}\sum_{i=1}^{n}\frac{forecast - actual}{actual}$$

where the 'forecast' term is applicable to the neural network generated output, and the 'actual' term is applicable to the actual output value generated by reservoir simulations in CMG. MPE is an average of the percentage difference of CMG and ANN predicted production values for each time step considered in this study. It is important to recognize the level of uniqueness in each class of solutions that the different network structures attempt to represent. In Case One and Case Two A, the target outputs of the networks are gas production profiles. For given inputs of reservoir parameters and wellbore geometry, the production profile will be a unique solution to this problem. The network easily learns the time-dependent behavior of hydrocarbon production via reservoir pressure depletion, and the network can be trained to very low error with little to no validation checks failing during the training procedure. In this case, a simple mean percentage error is sufficient to effectively describe the accuracy of the network. However, in Case Two B and C, the target outputs of the networks are reservoir system parameters, including rock and fluid properties, as well as wellbore geometry. In this case, the inherent non-uniqueness of the problem solution presents a difficulty for validation and testing processes of the network. In this case, more than one set of reservoir parameters will lead to the reproduction of a given

gas production profile. To understand how a variety of solutions may exist for one set of a given production profile and reservoir properties, one need consider the variables that can be altered within a wellbore geometry. To increase production, the main wellbore length may be increased, the number of branches may be increased, the branch length may be increased, the specified pressure at the wellhead may be decreased or any simultaneous combination of these parameters. As such, the trained network may find a correct answer to the parameter values that will yield a given production schedule, but it may not be the solution that was provided to the tool for validation and testing, and will therefore fail validation. In this case, error should not be assessed from the original properties that were utilized in the simulation to yield production values. Instead, the outputs artificially generated by the system should then be used as inputs to the previous network structure (to predict gas production schedule), and a production schedule generated. After this, production values can then be compared with the original production schedule used as input to the network to generate the parameters, and the gas production schedules can then be evaluated for accuracy.

Case One

Thee mean percentage error was calculated for each of the testing data sets and averaged to determine the overall accuracy of the neural network's reproduction of the CMG generated outputs. The final network structure yielded an average MPE of 1.7434%.

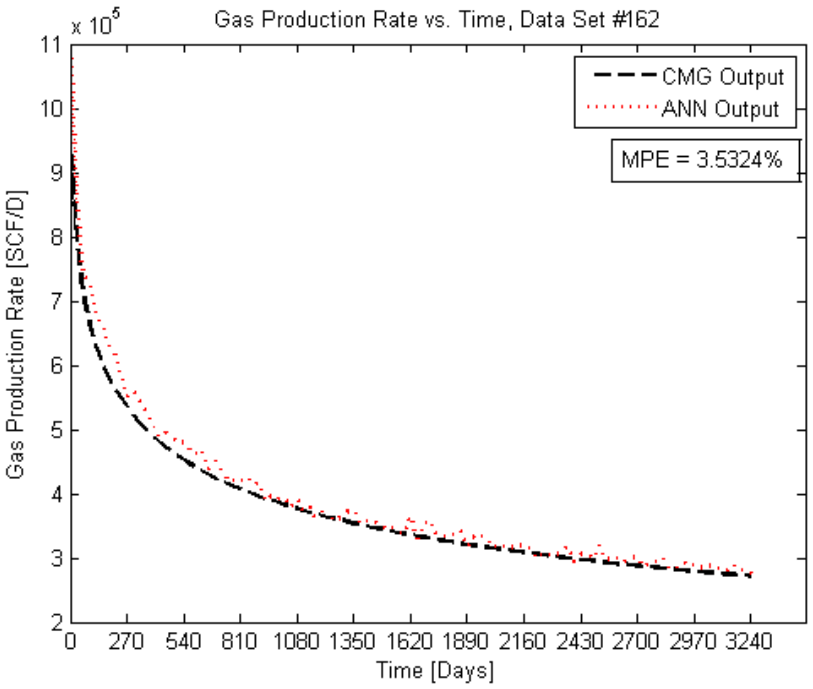

Figure 5-1 Case One, Data Set #162 Comparison of CMG Output vs. ANN Output

It was observed that data sets with a more moderate rate of decline experienced slightly larger error than those with a steeper decline. Figure 7-1 shows data set #162, with a mean percentage error of 3.53%. ANN output drastically overestimates early time production rate, yielding a gas rate of nearly 11x10^5 SCF/D, with CMG output only at about 9x10^5 SCF/D. At later times, the ANN data more accurately meets the CMG produced output.

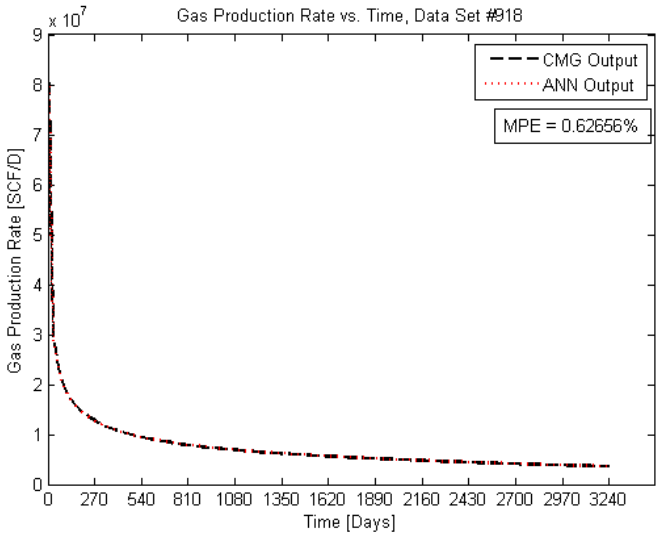

Figure 5-2 Case One, Data Set # 918 Comparison of CMG Output vs. ANN Output

Figure 7-2 features data set #918. This data set had production rate seeing a much steeper decline than the previous data set #162. In this case, ANN predicted output meets CMG output very accurately, with a mean percentage error of only 0.62%.

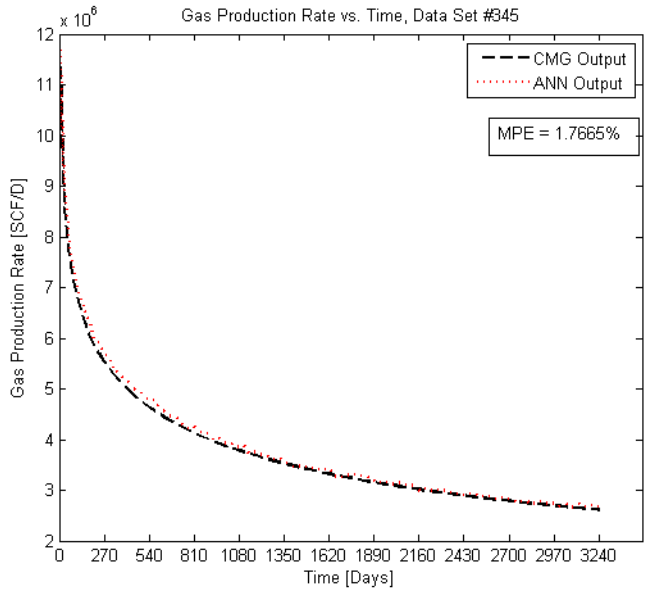

Figure 5-3 Case One, Data Set #345 Comparison of CMG Output vs. ANN Output

A final data set, set #345, is displayed in Figure 7-3. This data set has gas rates that see a decline between the two extremes illustrated above. Mean percentage error is 1.766%.

It is conclusive that the network is more accurate at reproducing gas production rates that see a sharper decline over time.

Case Two, Network Structure A

Network structure A, identical to the network developed in case 1, will be tested in the same way as case 1. That is, the output production data predicted by ANN A will be compared to the production data from CMG, and error analyzed for each production rate over the domain of the output.

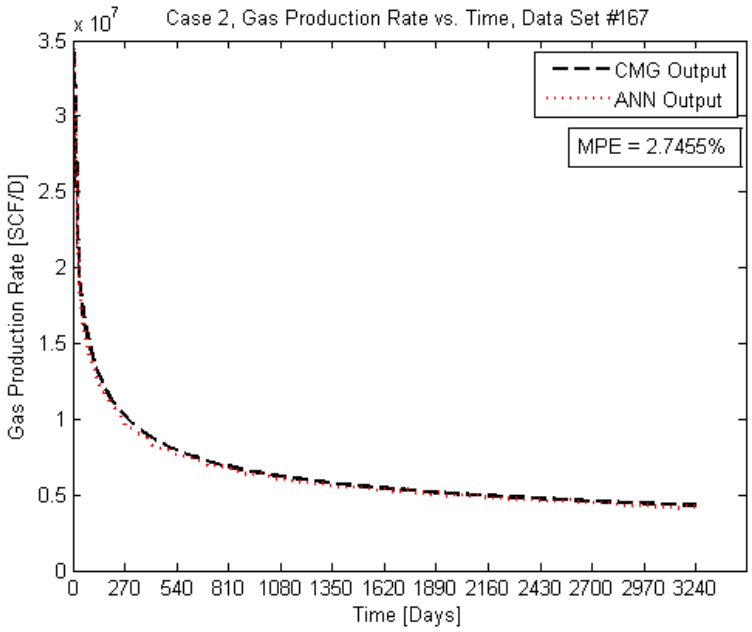

Figure 5-4 Case Two A, Data Set #167, CMG Output vs. ANN Output

Figure 7-4 shows a typical gas production schedule predicted by ANN A. Data set #167 had a mean percentage error of 2.75%, which is very accurate. The network is much more adept at predicting the later section of the gas production curve, where change in production rate is not so severe. At early times, the ANN is less accurate at data reproduction. Overall, the average MPE of all the testing sets was found as 2.08%, proving

that network A is very accurate in the prediction of gas production rate from given reservoir and well parameters.

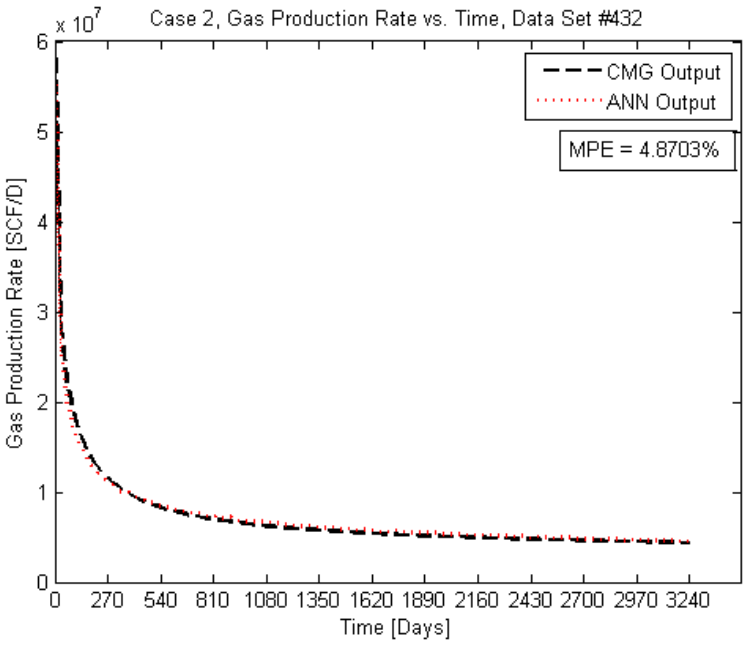

Figure 5-5 Case Two A, Data Set #432 CMG Output vs. ANN Output

Data set #432 is representative of the largest errors amongst all the test data sets inspected for error, with an MPE of 4.9%. As can be seen in Figure 7-6, the largest errors are found in the initial period with highest production rates, after which error decreases. In later times, another section of larger deviation is noted.

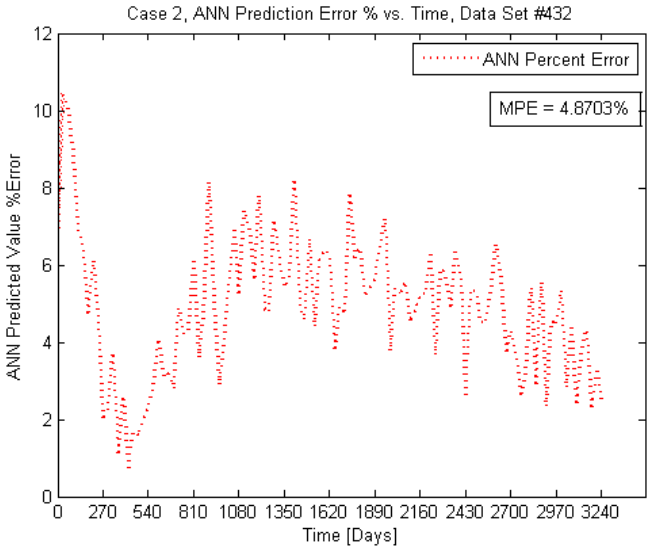

Figure 5-6 Case Two A, Data Set #432 ANN Prediction Percentage Error

Case Two, Network Structure B

Network structure B, as pictured below in Figure 7-7, features the input data of the gas production schedule of the well over 9 years (daily gas production, in monthly interval) as well as the wellbore geometry. The desired outputs include the reservoir rock and fluid parameters of the well, including Area A, thickness h, permeability k, porosity φ, reservoir temperature Ti, initial reservoir pressure Pi, and gas specific gravity γ_g.

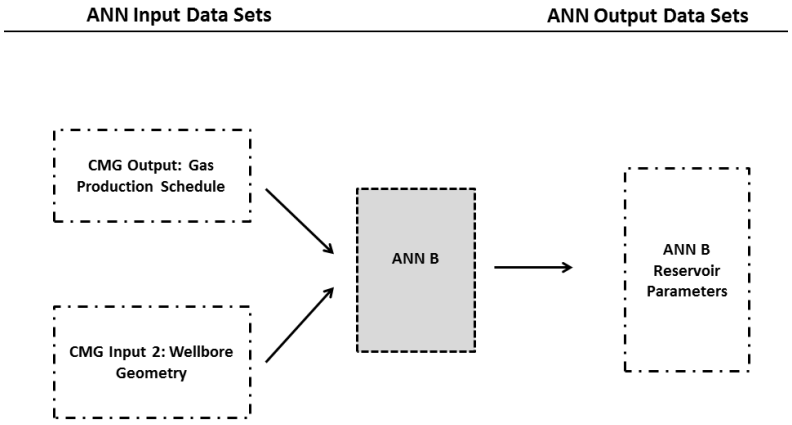

Figure 5-7 Case Two B Inputs and Outputs

As mentioned previously, the solution to the given problem is non-unique. A number of various reservoir parameters may exist to allow the generation of a given production profile. As such, a direct mean percent error analysis of the outputs (reservoir parameters predicted through use of the ANN) may not be compared with the original reservoir parameters provided to CMG for generation of the production profile. An additional method of analysis will be utilized. As proven in the Network Structure A case, the network structure A (allowing prediction of gas production schedule from reservoir parameters and wellbore geometry) will be assumed to be accurate. With this assumption, then the artificial inputs predicted from ANN B may be used as inputs to ANN A. The outputs (gas production schedules) should be identical, proving the feasibility of ANN 2's ability to generate reservoir parameters that would allow for the replication of a given gas production schedule with wellbore geometry held constant. This case will prove that ANN B

is able to capture the inverse relationship between reservoir properties and gas production, but may not be able to predict the unique solution.

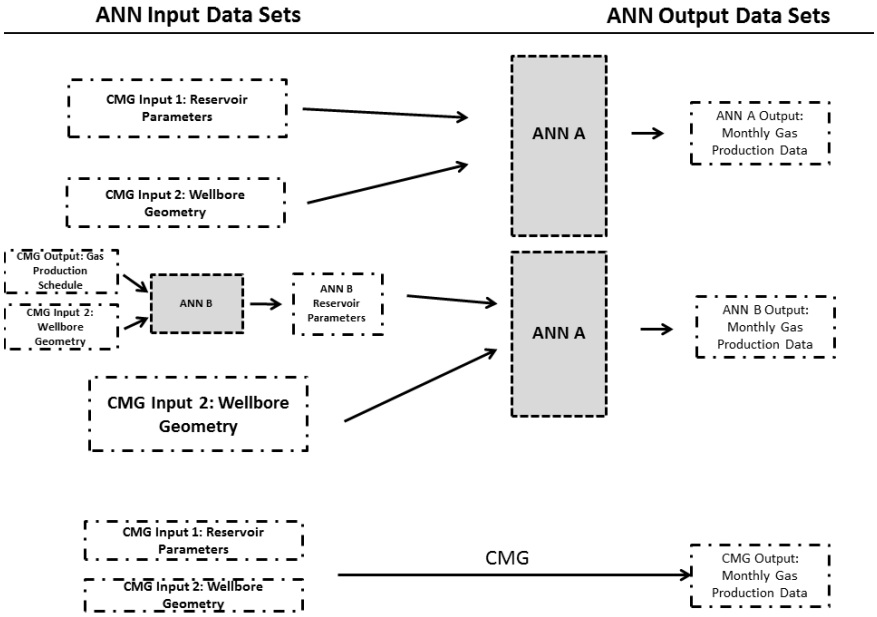

Figure 5-8 Network Two B Testing Procedure

Figure 7-8 depicts the testing procedure, described above, that will validate network two B. Three production profiles will be graphed. The first is simply the original production profile generated in IMEX (*CMG Output*) that was simulated from the original IMEX inputs (*CMG Input*). The second profile is the production profile artificially predicted with ANN A (*ANN A Output*) generated by using the same inputs as were given to IMEX in the initial numerical reservoir simulation (*CMG Input*). The final profile that will be analyzed is the production profile predicted with ANN A and ANN B (*ANN B Output*) using inputs that were generated from ANN B.

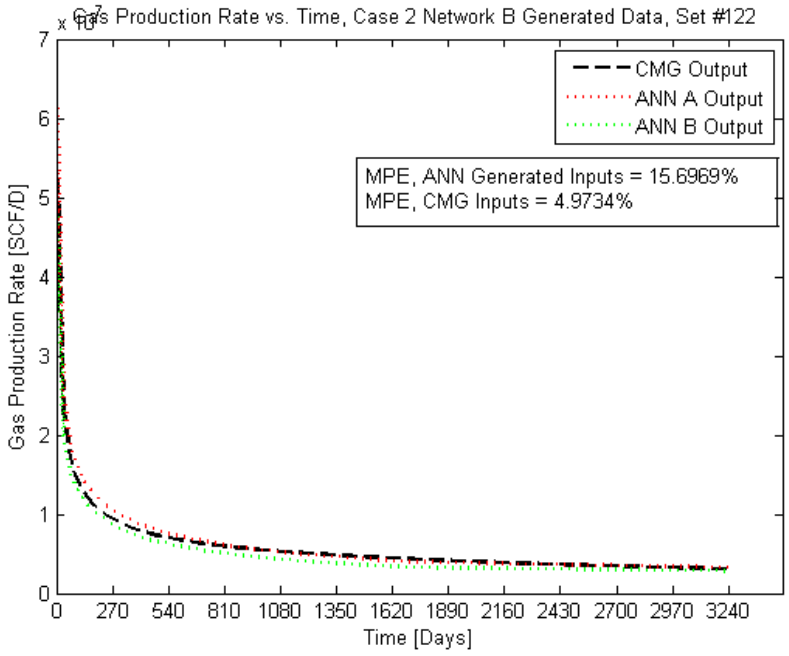

Figure 5-9 Case Two B, Data Set #122, CMG, ANN A and ANN B Outputs vs. Time

Figure 7-9 is representative of the testing procedure used to assess the performance of the ANN B network. Gas production profiles were generated using ANN A, with both the original inputs supplied to CMG (red curve), as well as with the artificial inputs from ANN B (green curve). Both curves were plotted in comparison to the original case of CMG production profile, ie. target input and output (black curve). While it can be observed that error is slightly greater for the ANN B output (15.7% vs. 5%, respectively), however, it is noted that the artificial inputs, when used by ANN A to produce a production profile, are able to produce a significantly close reproduction of the target gas production profile. For further comparison, the reservoir properties generated by ANN B are provided in Table 7-

Table 5-1 Case Two B, Data Set #122 Reservoir Parameter Prediction

Parameter	Original Parameter (CMG Input)	ANN B Prediction	Percentage Error
A [Acres]	812	874	7.64%
h [ft]	179	193	7.82%
k [md]	0.0952	0.09	5.46%
φ [%]	0.0836	0.0641	23.33%
Ti [˚F]	192	211.3	10.05%
Pi [psia]	2681	2876	7.27%
γg	0.74	0.683	7.70%

Multiple reservoir parameters were predicted to under 10% error. The parameters in this case with the largest variation from the target value were porosity (23.33% error) and reservoir temperature (10.05% error).

After removal of 4 outliers, the mean percentage error averaged over all 50 testing sets examined (that the ANN had not been exposed to in training) for the production profile generated with ANN predicted reservoir parameters was 23.89%. Likewise, the mean percentage error averaged over all 50 testing sets for the production profile generated with original CMG inputs was 20.87%. Therefore, with an error of only 3.02% between the MPE of production profiles generated with target/predicted inputs, and 9.8% error of predicted to target parameters, the network 2B is fairly accurate in prediction of a probable set of reservoir parameters resulting in a given production profile with known wellbore geometry. However, the network will not be able to predict the exact solution to the problem, because this network was developed without the use of functional links. The further employment of functional links on the output parameters may link the input and

output sets to a level such that the network may be able to be trained to predict the unique solution.

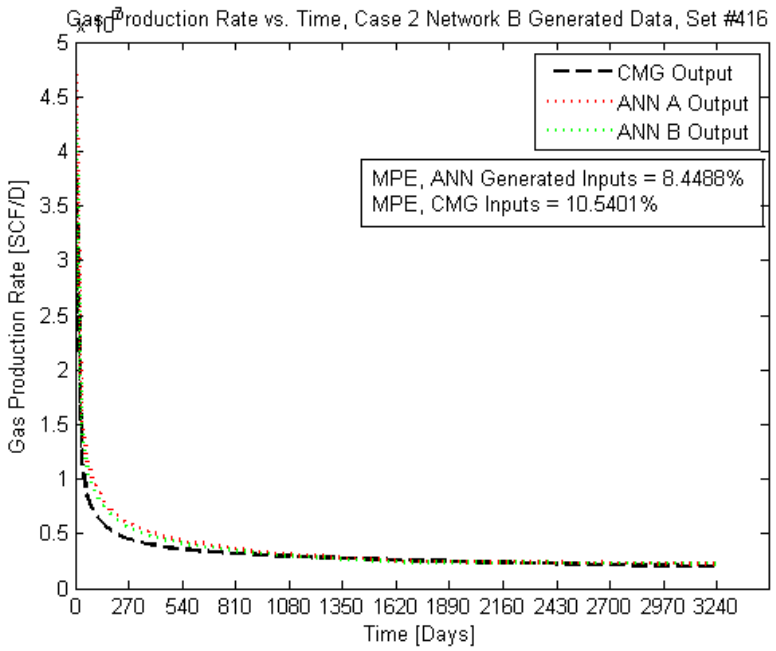

Figure 5-10 Case Two B, Data Set # 416, CMG, ANN A, and ANN B Outputs vs. Time

Data Set #416 is representative of the failure of ANN B to predict the unique set of parameters leading to the gas production data generated for set #416 in CMG. As can be seen from Figure 7-10, the production curves have a very low error, and both ANN A and ANN B are able to accurately reproduce the gas production curve. However, in Table 7-2, it is found that the target reservoir parameters (used by CMG for production profile generation) are very different from the solution predicted by ANN B. In this way, it is conclusive that there may exist may combinations of parameters to evolve a given gas production curve.

Table 5-2 Case Two B, Data Set #416, Reservoir Parameter Predictions

Parameter	Original Parameter (CMG Input)	ANN B Prediction	Percentage Error
A [Acres]	398	601	51.01%
h [ft]	72	82	13.89%
k [md]	0.0715	0.075	4.90%
φ [%]	0.1947	0.132	32.20%
Ti [˚F]	264	184	30.30%
Pi [psia]	5336	5107	4.29%
γg	0.66	0.71	7.58%

While the mean percentage error of the production profile generated by the two different sets of inputs may have resulted in errors of 8% and 10%, the reservoir parameters themselves have much larger error. Area [acres] varied as much as 51%, while still generating a near identical gas production profile. As such, while the network may be able to produce a representative set of reservoir parameters, it is not the ideal solution where parameter accuracy is necessary.

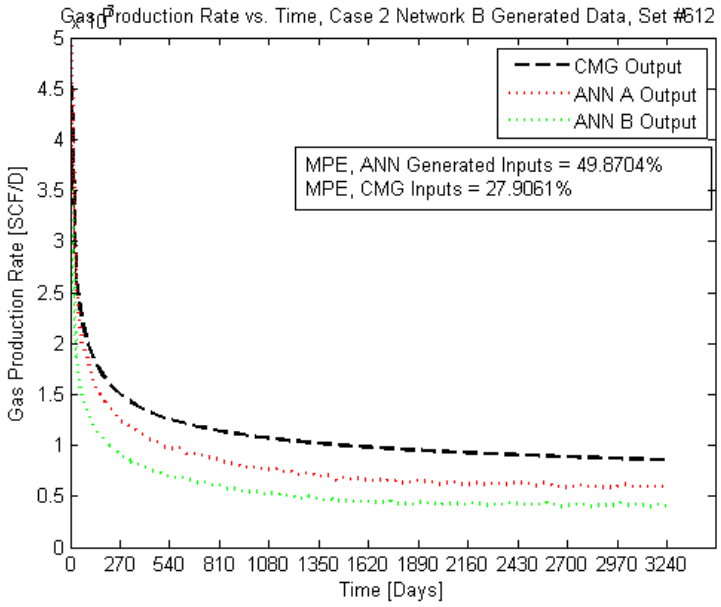

Figure 5-11 Case Two B, Data Set #612, CMG, ANN A, and ANN B Outputs vs. Time

Data set #612 is representative of a third type of output seen by the ANN system. In this case, there appears to be a consistent error throughout the whole domain of the production schedule. This case may present a situation in which some type of multiplier or forcing function may be applied post-ANN output to serve as error correction.

Table 5-3 Case Two B, Data Set #612, Reservoir Parameter Prediction

Parameter	Original Parameter (CMG Input)	ANN B Prediction	Percentage Error
A [Acres]	985	861.3	12.56%
h [ft]	238	203	14.71%
k [md]	0.057	0.0454	20.35%
φ [%]	0.1798	0.1647	8.40%
Ti [°F]	169	165	2.37%
Pi [psia]	6758	6615	2.12%
γg	0.77	0.7122	7.51%

To descriptively analysis the error present for each of the parameters predicted by ANN B, the mean percentage error for every parameter was averaged for the entire set of testing data sets. In this way, it can be quantitatively determined how well the network was able to reproduce the exact solution from the initial problem.

Table 5-4 ANN B, Average MPE, by Parameter. With and Without Functional Links.

Parameter	Average MPE, Without Functional Links	Average MPE, Highest Performing Configuration with Functional Links
A [Acres]	18.49	18.89
h [ft]	21.40	12.14
k [md]	22.40	49.16
φ [%]	22.08	27.54
Ti [˚F]	21.33	22.54
Pi [psia]	22.10	29.66
γg	21.28	9.77
Mean [%]	21.30	24.24191

Reservoir parameter mean percentages errors (from the initial input values given to CMG as input) were tabulated in this way for two different network configurations. One in which functional links were included, and another without the inclusion of function links. In this way, the intention was to test the network development and response to the use of functional links. It was found that with the use of functional links, the network was better able to be trained to output the unique solution. Validation and testing performance was much higher, and the network was trained to a much lower error level. However, due to the structure of the functional links used, when any given functional link contained more than one output parameter (ie. reservoir rock and fluid parameters), increased error would be

induced when parameters were calculated from an initial artificial parameter. Several parameters expressed lowest error when the network was trained with functional links. However, the standard deviation in error for outputs when generated with functional links was also larger, yielding error margins that are too high to be plausible for use in a real-time field study.

Case Two, Network Structure C

Network C inputs consist of 109 inputs of daily natural gas production rates (9 years, monthly basis), as well as 7 reservoir parameter inputs including area (A), thickness (h), permeability (k) , porosity (ϕ), reservoir temperature (Ti), initial reservoir pressure (Pi), and gas specific gravity (γ_g), for a total of 116 inputs. Outputs include 5 wellbore geometry parameters, including Horizontal Wellbore Length (L_{wb}), Specified Pressure (P_{sf}), Number of Lateral Branches (n), Branch Length (L_b), Location of Junction on main Wellbore Lateral (x).

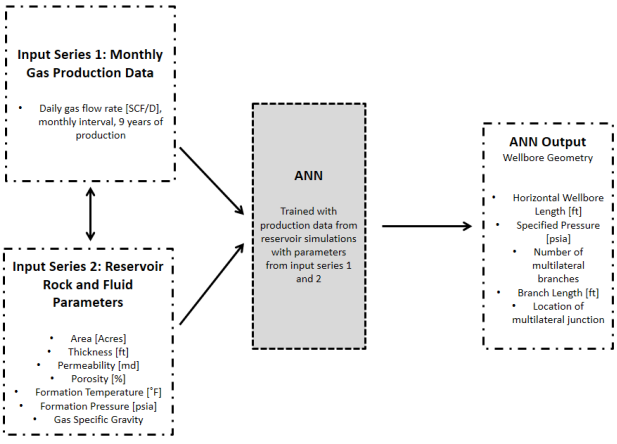

Figure 5-12 Case Two C Inputs and Outputs

As in Case Two B, the solution to this problem is non-unique. A multiple number of wellbore geometries are able to yield a given natural gas production schedule given a

certain combination of reservoir parameters. As such, two testing method will be employed. The first testing method is comprised of an error analysis between the original wellbore geometry used to generate a given gas production schedule and the wellbore geometry predicted by the ANN with the same gas production schedule used as network input. The second testing method is to use the wellbore geometry predicted by ANN C as an input to ANN A along with constant reservoir parameters, and compare the resulting gas production schedule with the original production schedule generated in CMG by the inputs given to CMG.

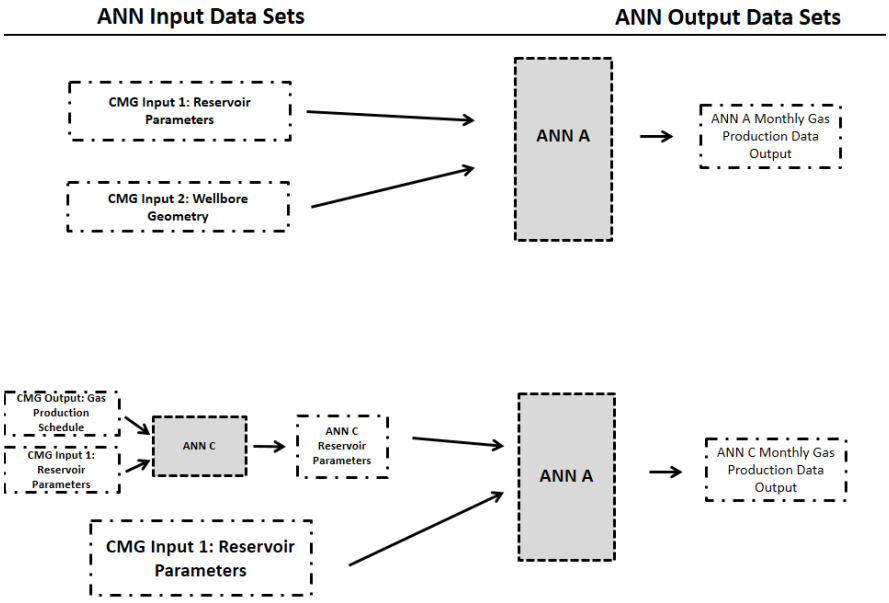

Figure 5-13 ANN C Testing Procedure

Figure 7-13 depicts the testing procedure, described above, that will validate network C. Three production profiles will be graphed. The first production profile is simply the original production profile generated in the numerical reservoir simulation model (*CMG Output*) that was simulated from the original inputs supplied to the numerical simulation (*CMG Input*). The second profile is the production profile artificially predicted with ANN A

(*ANN A Output*) generated by using the same inputs as were given to the numerical simulation (*CMG Input*). The final profile that will be analyzed is the production profile predicted with ANN A and ANN C (*ANN C Output*) using inputs that were generated from ANN C.

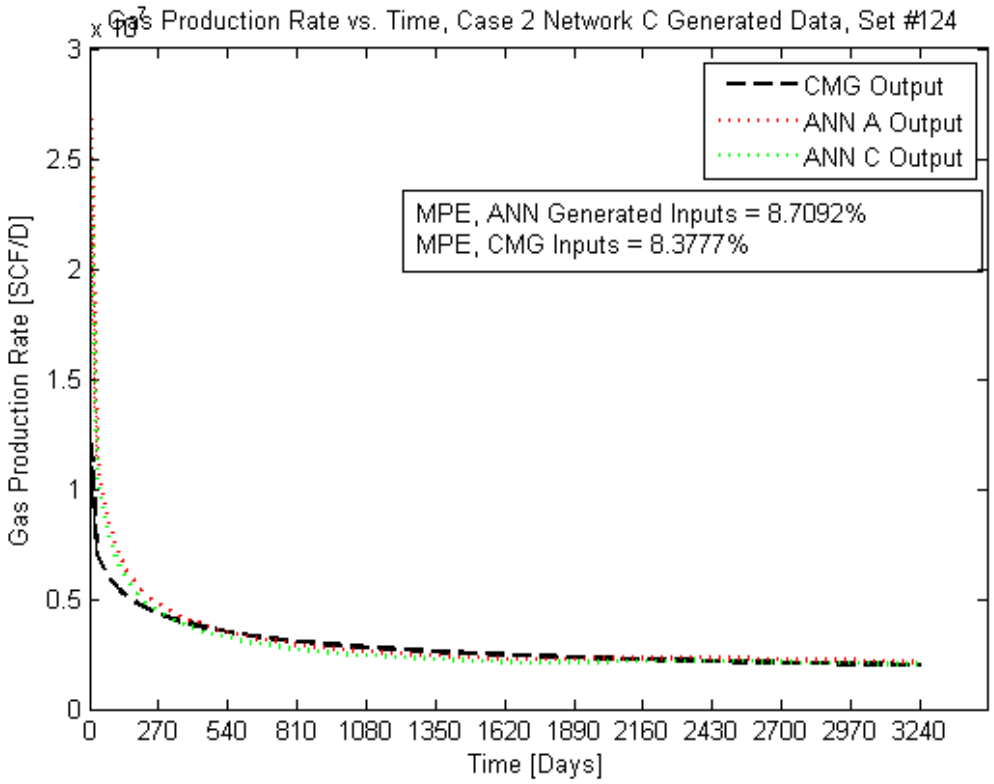

Figure 5-14 Case Two Network C, Set #124, Comparison of CMG, ANN A, and ANN C Gas Production Profiles

Figure 5-14 is a comparison of the production profiles from CMG (CMG Output), ANN A output with original wellbore geometry (ANN A Output), and ANN A output with wellbore geometry predicted with ANN C. Deviation from the original CMG data is only 8.7% when curve is generated from wellbore geometry predicted by ANN C. The network has trouble reproducing accurate gas production values when there is a steep decline in daily production rate, as can be seen from the larger deviation of production rate at early

times. However, in later times when production rate decline has begun to stabilize, the network is very accurate in the prediction of gas production rate.

Table 5-5 Case Two C, Data Set #124 Wellbore Geometry Prediction

Parameter	Original Parameter (CMG Input)	ANN C Prediction	Percentage Error
Main Horizontal Wellbore Length, Lwb [ft]	1840	3864.86	109.88%
Specified Pressure, Psf [psia]	261.95	484.085	84.80%
Number of Branches, n	1	1.00027	0.02%
Branch Length, Lb [ft]	510	685.502	34.41%
Multilateral Junction Location, x	0.7	0.3463	50.52%

Table 5-5 is an error analysis of the individual wellbore geometry parameters generated by ANN C for Data Set #124 vs. the original values used as input by CMG for generation of the gas production profile. While the generated production profiles are very similar, the wellbore geometries seem to diverge by quite a large margin. However, given that the specified pressure suggested by ANN C is possible to be maintained by the surface production facilities, and the main wellbore length does not exceed the extent of the reservoir, the configuration predicted by the network is still feasible.

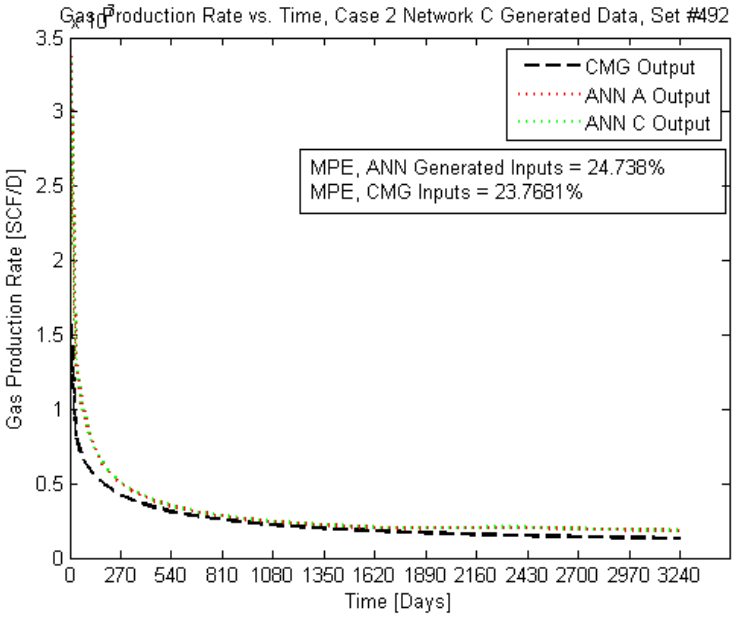

Figure 5-15 Case Two Network C, Set #492, Comparison of CMG, ANN A, and ANN C Gas Production Profiles

Figure 5-15 is another representative comparison of the outputs generated by ANN A, ANN C, and the original gas production schedule produced by CMG. Error in production rate generated by the network is quite variable. In some sets, error as low as 4% to 10% was seen, with other sets over 50% error. Therefore, more work should be conducted in the functional links employed by the network. However, being a design parameter, many combinations will exist.

Table 5-6 Case Two C, Data Set #492 Wellbore Geometry Prediction

Parameter	Original Parameter (CMG Input)	ANN C Prediction	Percentage Error
Main Horizontal Wellbore Length, Lwb [ft]	2740	2749.27	0.34%
Specified Pressure, Psf [psia]	48.8	267	447.11%
Number of Branches, n	2	2	0%
Branch Length, Lb [ft]	820	864.16	5.39%
Multilateral Junction Location, x	0.3	0.34363	14.55%

Table 5-6 shows the wellbore geometries predicted by ANN C vs. the actual wellbore geometry used to generate the gas production profile in CMG. Specified pressure was quite divergent, but other parameters were predicted quite accurately. Critical design parameters such as wellbore length, branch length, number of branches, and even junction location were predicted with minimal error.

The average deviation from the target (CMG generated) gas production schedule over all the testing sets while using wellbore geometries predicted by ANN C was 29% after removal of 2 large outliers.

Further Work

The work done in this thesis has laid a very convenient backdrop for further neural network and batch reservoir depletion simulations with the fishbone type wellbore structure. The MATLAB™ code developed to modify and generate batch files contains a sturdy and reproducible algorithm for dynamic sizing and placement of main horizontal wellbore and branches with a given number of branches. The code will automatically test and resize for wellbore lengths that will exceed other reservoir (Area) parameters. In further work, cases can be added to handle an additional number of wellbores. Additionally, the code developed in this thesis has allowed for the dynamic and random placement of one branch junction. With the future increased number of deviations, this technique can be replicated and duplicated to allow for multiple junctions and a variety of wellbore geometries.

In the testing and development of neural network structure, additional testing can be done on functional links to be used when studying the inverse production problem in which reservoir and wellbore parameters are generated from field gas production data. At this point in the research, time constraint became a major factor in the results seen to date. With an additional investment of time, various combinations of parameters can be tested and a more robust network structure developed.

Chapter 6

Summary

In this research, three different types of problems were solved by using a neural network. In the first, or "forward-looking" case, reservoir rock and fluid parameters along with wellbore parameters were used by the ANN as inputs to predict a gas production schedule for nine years of daily gas production data. Along with this, two inverse cases were developed, one which can take the input of a gas production schedule along with wellbore geometry and attempt to predict the formation rock and fluid properties, and another which can take the input of a gas production schedule along with reservoir rock and fluid parameters, and predict the necessary wellbore geometry to produce at the given rates.

In order to train an ANN, a large set of data were generated within a range of parameters determined by the parameters of the target formation (tight gas sand), target fluid (natural gas, with no water or oil saturations), and wellbore (multilateral fishbone horizontal wellbore). These data sets were input into CMG to generate the respective gas production schedules for each data set. The data were then trimmed for any error data sets (or unrealistic sets) and used for the training of the ANN. Data was given to the ANN randomly in three data sets: a training set in which the ANN will attempt to "memorize" or manipulate its parameters to obtain the given outputs from the given inputs, a validation set in which these data sets are run to gauge error performance in sets which the network has not been trained with, and a testing set in which the network will only be exposed to the data one time, at the end of training, to estimate an error percentage (performance) of the network to data sets which it has not yet been exposed to.

In the forward looking ANN case, gas production schedule data was able to be predicted to within 5% - 10% error from the original data generated in CMG.

In the inverse case, reservoir parameters were able to be predicted within 20% error from the original reservoir parameters that were used to generate the production data in CMG and train the forward looking ANN. A variety of plausible reservoir rock and fluid parameter combinations were able to be generated that, when provided as inputs to the forward looking ANN, were capable of generating gas production profiles that very accurately reproduced the original gas production schedule, to within 24%.

In the second inverse case, wellbore geometries were predicted, and when used to generate a gas production curve, saw deviation from the target gat production curve of 29% or less error. As a design parameter, a number of viable combinations could exist. As long as specified pressure predicted by the network is feasible to be handled by the surface production equipment, and wellbore length does not exceed the extent of the reservoir, wellbore geometry parameters predicted by the ANN were plausible.

A number of conclusions are able to be drawn from this research.

- Neural Networks with conjugate gradient algorithms are very capable of predicting natural gas production when given reservoir and well properties as input parameters (also found by previous studies [1], [2], [15]).
- The parameter x, the location of the multilateral junction on the main wellbore, is a very difficult parameter predict, and a neural network cannot do so accurately without functional links.
- For inverse problems, outputs generated from the inverse ANN are able to be used as inputs to the forward looking ANN, and error from the original solution may be calculated to determine accuracy of the inverse ANN.

The overall impression of this work is that given known reservoir rock, fluid, and wellbore parameters, such a trained network will be able to very accurately predict probable gas production rates for years in the future. Given the existence of a unique solution, the neural network is able to be trained (while avoiding overtraining [16]) to a very high accuracy. However, when the inverse problem is considered, the networks designed in this study experienced very low validation performance when trained in the typical fashion. This result has arisen from the non-unique solution to the problem. Whether it be the prediction of reservoir rock and fluid parameters (Network Structure B), or the prediction of wellbore characteristics that should be drilled (Network Structure C), both systems have a various number of parameter combinations that will fulfill the input parameters. Therefore, as a formation evaluation tool, the ANN presented in this study, at this point, may not be a plausible application. However, if a company desires to drill a horizontal multilateral well based on a desired production goal and a known set of reservoir parameters, the ANN C in this study may be indeed be a viable route for the company to determine necessary wellbore geometry.

Appendix A

ANN Development Procedure

Part A. Generation of data sets. This phase of the neural network design process will generate a variety of reservoir rock and fluid properties, along with wellbore geometry and production scheme parameters. Parameters generated within ranges characterized by target reservoir type and subsequent reservoir fluid. See Table 5.1 and 5.2 for ranges of each parameter.

Table 6-1 Generation of Data Sets

```
%Generate random combinations of reservoir, fluid, and wellbore properties
%given ranges of each property and number of sets to generate

sets = 1000;

%Parameter number one, reservoir
a = randi([100,1000],[1,sets]);
a_ft = a*43560;
%Parameter Number two, reservoir thickness
h = randi([50,300],[1,sets]);
%Parameter number three, permeability
k = unifrnd(0.0001, 0.1, [1,sets]);
%Parameter number four, porosity
phi = unifrnd(0.05, 0.20, [1,sets]);
%Parameter number 5, reservoir temp
t = randi([100,300],[1,sets]);
%Parameter 6, initial reservoir pressure
pi = unifrnd(1000,7000,[1,sets]);
%Parameter 7, specified pressure
psf_multiplier = unifrnd(0,0.2,[1,sets]);
psf = psf_multiplier.*pi + 14.7;
%Parameter 8, wellbore length
boundlen = (a_ft).^0.5; %First, establish a parameter that defines the gridblock boundary
length
lwb = randi([1000,7000],[1,sets]);
%check for ambiguity between reservoir gridblock boundary length and wellbore length.
%Set a wellbore length maximum of 0.8*reservoir gridblock boundary length
for i = 1:sets
   if lwb(i) > boundlen(i)*0.8
      lwb(i) = boundlen(i)*0.8;
   end
end
lwb = roundn(lwb, 1);
%Parameter 9, Gas Specific Gravity
```

```
g_sg = unifrnd(0.6, 0.8, [1,sets]);

%Next, generate parameters related to fishbone branch geometry. For case
%#1, these parameters will be ignored in the black-oil simulator

%Parameter 10, Branch number
n = randi(2, [1,sets]);


%Parameter 11, Branch Placement
%X referes to the distance from the beginning of the horizontal wellbore
%length, at which the branch junction will occur

x = [1,sets];
x = unifrnd(0.1,0.8, [1,sets]);
x = roundn(x, -1);

%Parameter 12, Branch Deviation Angle
%%theta = randi([15,45],[1,sets]); Future chamge: to randominze the angle
%%of deviation. For first revision of the code, simply assign theta to
%%equal the value of 45 degrees.
theta = 45*(3.1415/180);

%Parameter 13, Branch Length. Adjust in the case that branch runs outside
%of reservoir boundary length.
lb = randi([300,1000],[1,sets]);
for i = 1:sets
    if (x(i)*lwb(i) + lb(i)*cos(theta)) > boundlen(i)
       lb(i)= (boundlen(i) - x(i))/cos(theta);
    end
end
lb = roundn(lb,1);

%Debug the branch length condition system. Check or the length remainin
%btwen the ed of the fishbone branch and the reservoir boundary
debugendlength = [1:sets];
debugwell_length = [1:sets];
for i=1:sets
    debugendlength(i) = boundlen(i) - lb(i)*cos(theta) - x(i)*lwb(i);
    debugwell_length(i) = lwb(i)/boundlen(i);
end
%Generation of 2 matrices to summarize all parameters. setcompilation will
%be exported to text and converted to excel as a record of the parameters
%of every set. indata will be used with another MATLAB algorithm [1] to
%generate input files to run with CMG for production data simulation.


setcompilation=[a;boundlen;h;k;phi;t;pi;psf;g_sg;lwb;n;lb;x];
setcompilation = transpose(setcompilation);

input=[boundlen;h;k;phi;t;pi;g_sg;psf;lwb;n;lb;x];
```

```
        input = transpose(input);
        %%Export files to .txt format
        save setcompilation.txt setcompilation -ASCII
        save input.txt input -ASCII

xlswrite('input.xls',input);
```

Part B. Generation of CMG input script. Computer Modelling Group's (CMG) IMEX black oil simulator operates with an input file ('.dat'). The following MATLAB code will execute by searching the root directory for file 'input.txt' containing all data set and parameter information generated in Part A, and will generate an individual and unique '.dat' input file to run with CMG IMEX simulator.

Part B.1. Generation of CMG IMEX input script for Case 1. As can be seen in section 5.1.1, Case One is simply a single horizontal wellbore of randomly generated length (wellbore length unique to each data set). This code will automatically center the wellbore in the reservoir, in both lateral directions (i and j directions). In the thickness (k direction) direction of the reservoir, the code will automatically generate a grid block system with three equally sized layers, with the wellbore in the middle layer.

Table 6-2 Generation of CMG IMEX Input Script, Case One

```
        load input.txt;
        q = 1000;
        for i=1:q;
        numb=num2str(i);
        temp=['data' numb '.dat'];
        fid=fopen(temp,'wt');
        fprintf(fid,'\n RESULTS SIMULATOR IMEX 201211');
        fprintf(fid,'\n INUNIT FIELD');
        fprintf(fid,'\n WSRF WELL 1');
        fprintf(fid,'\n WSRF GRID TIME');
        fprintf(fid,'\n WSRF SECTOR TIME');
        fprintf(fid,'\n OUTSRF WELL LAYER NONE');
        fprintf(fid,'\n OUTSRF RES ALL');
        fprintf(fid,'\n OUTSRF GRID SG SW PRES');
        fprintf(fid,'\n WPRN GRID 0');
        fprintf(fid,'\n OUTPRN GRID NONE');
        fprintf(fid,'\n OUTPRN RES NONE');
```

```
        fprintf(fid,'\n **$ Distance units: ft');
        fprintf(fid,'\n RESULTS XOFFSET 0.0000');
        fprintf(fid,'\n RESULTS YOFFSET 0.0000');
        fprintf(fid,'\n RESULTS ROTATION 0.0000 **$ (DEGREES)');

        fprintf(fid,'\n **');
        fprintf(fid,'\n **');
        fprintf(fid,'\n **$ ************************************************************');
        fprintf(fid,'\n **$ Definition of fundamental cartesian grid');
        fprintf(fid,'\n **$ ************************************************************');
        fprintf(fid,'\n GRID VARI 30 29 3');
        fprintf(fid,'\n KDIR DOWN');
        fprintf(fid,'\n **');
        fprintf(fid,'\n **');
        fprintf(fid,'\n DI IVAR');
        %I-direction gridblock assignment
        %5 gridblocks from reservoir edge to beginning of horizontal wellbore
        %20 gridblocks for the length of the main horizontal wellbore
        %5 gridblockss from the end of the main horizontal wellbore to the far edge
        %of the reservoir
        fprintf(fid,'\n 5*%d 20*%d 5*%d', (input(i,1)-input(i,9))/10, input(i,9)/20,(input(i,1)-
input(i,9))/10);
        fprintf(fid,'\n **');
        fprintf(fid,'\n **');

        fprintf(fid,'\n DJ JVAR');
        %J-direction gridblock assignment. Divide j direction into 29 equal block
        %lengths. 14 above wellbore, 1 for wellbore, and 14 below wellbore
        fprintf(fid, '\n 29*%d', input(i,1)/29);
        fprintf(fid, '\n **');
        fprintf(fid, '\n **');

        fprintf(fid, '\n DK ALL');
        %K-direction gridblock assignment.
        fprintf(fid, '\n 2610*%d', input(i,2)/3);
        fprintf(fid, '\n DTOP');
        fprintf(fid, '\n 870*%d', input(i,6));

        fprintf(fid, '\n **');
        fprintf(fid, '\n **');
        fprintf(fid, '\n **$ Property: Permeability I (md)   Max: %d Min: %d' ,input(i,3),input(i,3));
        fprintf(fid, '\n PERMI CON       %d', input(i,3));
        fprintf(fid, '\n **$ Property: NULL Blocks  Max: 1  Min: 1');
        fprintf(fid, '\n **$  0 = null block, 1 = active block');
        fprintf(fid, '\n NULL CON          1');
        fprintf(fid, '\n **$ Property: Porosity  Max: %d  Min: %d', input(i,4),input(i,4));
        fprintf(fid, '\n POR CON        %d',input(i,4));
        fprintf(fid, '\n **$ Property: Permeability J (md)   Max: %d  Min: %d', input(i,3), input(i,3));
        fprintf(fid, '\n PERMJ CON       %d', input(i,3));
        fprintf(fid, '\n **$ Property: Permeability K (md)   Max: %d  Min: %d', input(i,3), input(i,3));
        fprintf(fid, '\n PERMK CON       %d', input(i,3));
```

```
        fprintf(fid, '\n **$ Property: Pinchout Array  Max: 1  Min: 1');
        fprintf(fid, '\n **$  0 = pinched block, 1 = active block');
        fprintf(fid, '\n PINCHOUTARRAY CON          1');
        fprintf(fid, '\n CPOR 3.0E-6');
        fprintf(fid, '\n MODEL GASWATER');
        fprintf(fid, '\n TRES %d', input(i,5));
        fprintf(fid, '\n PVTG ZG 1');

        %%%%%%Code for generationg of PVT tables, viscosity. Created by Yogesh
        %%%%%%Bansal, inplemented also by Burak Kulga [1,2]
        Psc = 14.7;
        T_r = 460;
        T_res = input(i,5) + T_r;
        MW = input(i,7)*28.96;
        Ppc = 756.8 - 131*input(i,7) - 3.6*input(i,7)^2;    %%Determine pseudo-critical pressure for
the gas
        Tpc = 169.2 + 349.5*input(i,7) - 74.0*input(i,7)^2; %%Determine pseudo-critical
temperature for the gas
        P = (Psc:(8000-Psc)/20:8000);
        Pr = P./Ppc;    %%Calculate reduced pressure of the gas
        Tr = T_res/Tpc; %%calculate reduced temperature of the gas

        %%Loop to calculate Z-values for PVT tables (DRANCHUK)
        A = [0.3265; -1.07; -0.5339; 0.01569; -0.05165; 0.5475; -0.7361; 0.1844; 0.1056; 0.6134;
0.721];
        [u,v] = size(P);
        c1 = A(1) + A(2)/Tr + A(3)*Tr^-3 + A(4)*Tr^-4 + A(5)*Tr^-5;
        c2 = A(6) + A(7)/Tr + A(8)/Tr^2;
        c3 = A(7)/Tr + A(8)/Tr^2;
        for h=1:u
          for g=1:v
            if P(h,g) ~= 0
              ro(h,g) = 0.27*Pr(h,g)/Tr;
              diff=1;
              while diff > 0.0001
                f = 1 - 0.27*Pr(h,g)/(ro(h,g)*Tr) + c1*ro(h,g) + c2*ro(h,g)^2 - A(9)*c3*ro(h,g)^5 +
A(10)*(1 + A(11)*ro(h,g)^2*exp(-1*A(11)*ro(h,g)^2/Tr^3));
                fd = 0.27*Pr(h,g)/(ro(h,g)^2*Tr) + c1 + 2*c2*ro(h,g) - 5*A(9)*c3*ro(h,g)^4 +
2*A(10)*ro(h,g)*(1+A(11)*ro(h,g)^2 - A(11)^2*ro(h,g)^4)*exp(-1*A(11)*ro(h,g)^2)/Tr^3;
                ron(h,g) = ro(h,g) - f/fd;
                diff = abs(ron(h,g) - ro(h,g));
                ro(h,g) = ron(h,g);
              end
              z(h,g) = 0.27*Pr(h,g)/(ro(h,g)*Tr);
            end
          end
        end

        %%Viscosity calculation, created by Yogesh Bansal [1], used by Barak Kulga
        %%[2], from Lee

        X = 3.448 + 986.4/T_res + (0.01009*MW);
```

```
Y = 2.447 - (0.2224*X);
K = (9.379 + 0.01607*MW)*T_res^1.5/(209.2 + 19.26*MW + T_res);

ro_calc = P*MW./z/10.732/T_res;
mu_g = 10^-4*K*exp( X * (ro_calc./62.4).^Y);
P_V_T = [P;z;mu_g];
fprintf(fid, '\n %10.3f %5.6f %5.6f', P_V_T);
fprintf(fid, '\n **');
fprintf(fid, '\n **');
fprintf(fid, '\n **');
fprintf(fid, '\n **');
fprintf(fid, '\n **');
fprintf(fid, '\n **');
fprintf(fid, '\n BWI 1');
fprintf(fid, '\n CVW  0.0');
fprintf(fid, '\n CW  3.51118e-006');
fprintf(fid, '\n DENSITY WATER  62.4');
fprintf(fid, '\n REFPW  8000');
fprintf(fid, '\n VWI  1');
fprintf(fid, '\n GRAVITY GAS  %d', input(i,7));
fprintf(fid, '\n PTYPE CON 1');
fprintf(fid, '\n ROCKFLUID');
fprintf(fid, '\n RPT  1');

fprintf(fid, '\n SWT');
fprintf(fid, '\n **$       Sw      krw');
fprintf(fid, '\n 0 0.0 0');
fprintf(fid, '\n 1 1 0');
fprintf(fid, '\n SGT');
fprintf(fid, '\n **$       Sg      krg');
fprintf(fid, '\n 0 0.0');
fprintf(fid, '\n 1 1');
fprintf(fid, '\n INITIAL');
fprintf(fid, '\n USER_INPUT');
fprintf(fid, '\n PRES CON %d', input(i,6));
fprintf(fid, '\n SW CON  0');
fprintf(fid, '\n NUMERICAL');
fprintf(fid, '\n RUN');
fprintf(fid, '\n DATE  2014  1  1');

fprintf(fid, '\n **$');
fprintf(fid, '\n WELL "Well-1"');
fprintf(fid, '\n PRODUCER "Well-1"');
fprintf(fid, '\n OPERATE  MIN  BHP  %d  CONT', input(i,8));
fprintf(fid, '\n **$    rad geofrac     wfrac      skin');
fprintf(fid, '\n GEOMETRY  I  0.375  0.37  1  0');
fprintf(fid, '\n PERF  GEOA  "Well-1"');

%%%%Code for the perforation geometry
   fprintf(fid, '\n **$ UBA    ff  Status  Connection');
   fprintf(fid, '\n 6 15 2 1. OPEN  FLOW-TO "SURFACE" REFLAYER');
   j=1;
```

```
    for i=7:25
        fprintf(fid, '\n %d 15 2 1. OPEN FLOW-TO %d', i, j);
        j= j+1;
    end

fprintf(fid, '\n **'); %following is code obtaining monthly production data

for day=2:30
    fprintf(fid, '\n DATE  2014  1  %d.00000', day);
end
for month=2:12
    if month == 2
        for day=1:28
            fprintf(fid, '\n DATE  2014  %d  %d.00000',month, day);
        end
    else
        for day=1:30
            fprintf(fid, '\n DATE  2014  %d  %d.00000',month, day);
        end
    end
end

for year=2015:2022
    for month=1:12
        if month == 2
            for day=1:28
                fprintf(fid, '\n DATE  %d  %d  %d.00000',year, month, day);
            end
        else
            for day=1:30
                fprintf(fid, '\n DATE  %d  %d  %d.00000',year, month, day);
            end
        end
    end
end
fprintf(fid, '\n STOP');
fclose(fid);
end
```

Part B.2. Generation of CMG IMEX input script for Case Two. Section 5.1.2 describes the wellbore geometry of Case Two. This MATLAB script will generate a CMG code with perforations corresponding to a number of branches (one to two branches) with a varying branch length, as well as a varying junction location on the main wellbore lateral. Junction departure angle (the orientation of the branch with the main lateral) was held constant at 45°.

Table 6-3 Generation of CMG IMEX Input Script, Case Two

```
load input.txt;
q = 1000;
for i=1:q;
numb=num2str(i);
temp=['data' numb '.dat'];
fid=fopen(temp,'wt');
fprintf(fid,'\n RESULTS SIMULATOR IMEX 201211');
fprintf(fid,'\n INUNIT FIELD');
fprintf(fid,'\n WSRF WELL 1');
fprintf(fid,'\n WSRF GRID TIME');
fprintf(fid,'\n WSRF SECTOR TIME');
fprintf(fid,'\n OUTSRF WELL LAYER NONE');
fprintf(fid,'\n OUTSRF RES ALL');
fprintf(fid,'\n OUTSRF GRID SG SW PRES');
fprintf(fid,'\n WPRN GRID 0');
fprintf(fid,'\n OUTPRN GRID NONE');
fprintf(fid,'\n OUTPRN RES NONE');



fprintf(fid,'\n **$ Distance units: ft');
fprintf(fid,'\n RESULTS XOFFSET 0.0000');
fprintf(fid,'\n RESULTS YOFFSET 0.0000');
fprintf(fid,'\n RESULTS ROTATION 0.0000 **$ (DEGREES)');

fprintf(fid,'\n **');
fprintf(fid,'\n **');
fprintf(fid,'\n **$ ************************************************************');
fprintf(fid,'\n **$ Definition of fundamental cartesian grid');
fprintf(fid,'\n **$ ************************************************************');
xblock = round(input(i,1)/50);
yblock = round(input(i,1)/50);
fprintf(fid,'\n GRID VARI %d %d 3', xblock, yblock);
fprintf(fid,'\n KDIR DOWN');
fprintf(fid,'\n **');
fprintf(fid,'\n **');
fprintf(fid,'\n DI IVAR');
%I-direction gridblock assignment
%5 gridblocks from reservoir edge to beginning of horizontal wellbore
%20 gridblocks for the length of the main horizontal wellbore
%5 gridblockss from the end of the main horizontal wellbore to the far edge
%of the reservoir
fprintf(fid,'\n %d*50 ',xblock);
fprintf(fid,'\n **');
fprintf(fid,'\n **');

fprintf(fid,'\n DJ JVAR');
%J-direction gridblock assignment. Divide j direction into 29 equal block
%lengths. 14 above wellbore, 1 for wellbore, and 14 below wellbore
fprintf(fid, '\n %d*50', yblock);
fprintf(fid, '\n **');
```

```
            fprintf(fid, '\n **');

            fprintf(fid, '\n DK ALL');
            %K-direction gridblock assignment.
            fprintf(fid, '\n %d*%d', xblock*yblock*3,input(i,2)/3);
            fprintf(fid, '\n DTOP');
            fprintf(fid, '\n %d*%d', xblock*yblock,input(i,6));

            fprintf(fid, '\n **');
            fprintf(fid, '\n **');
            fprintf(fid, '\n **$ Property: Permeability I (md)   Max: %d Min: %d' ,input(i,3),input(i,3));
            fprintf(fid, '\n PERMI CON       %d', input(i,3));
            fprintf(fid, '\n **$ Property: NULL Blocks  Max: 1  Min: 1');
            fprintf(fid, '\n **$  0 = null block, 1 = active block');
            fprintf(fid, '\n NULL CON          1');
            fprintf(fid, '\n **$ Property: Porosity  Max: %d  Min: %d', input(i,4),input(i,4));
            fprintf(fid, '\n POR CON        %d',input(i,4));
            fprintf(fid, '\n **$ Property: Permeability J (md)   Max: %d  Min: %d', input(i,3), input(i,3));
            fprintf(fid, '\n PERMJ CON       %d', input(i,3));
            fprintf(fid, '\n **$ Property: Permeability K (md)   Max: %d  Min: %d', input(i,3), input(i,3));
            fprintf(fid, '\n PERMK CON       %d', input(i,3));
            fprintf(fid, '\n **$ Property: Pinchout Array  Max: 1  Min: 1');
            fprintf(fid, '\n **$  0 = pinched block, 1 = active block');
            fprintf(fid, '\n PINCHOUTARRAY CON          1');
            fprintf(fid, '\n CPOR 3.0E-6');
            fprintf(fid, '\n MODEL GASWATER');
            fprintf(fid, '\n TRES %d', input(i,5));
            fprintf(fid, '\n PVTG ZG 1');

            %%%%%%Code for generationg of PVT tables, viscosity. Created by Yogesh
            %%%%%%Bansal, inplemented also by Burak Kulga [1,2]
            Psc = 14.7;
            T_r = 460;
            T_res = input(i,5) + T_r;
            MW = input(i,7)*28.96;
            Ppc = 756.8 - 131*input(i,7) - 3.6*input(i,7)^2;    %%Determine pseudo-critical pressure for
the gas
            Tpc = 169.2 + 349.5*input(i,7) - 74.0*input(i,7)^2; %%Determine pseudo-critical
temperature for the gas
            P = (Psc:(8000-Psc)/20:8000);
            Pr = P./Ppc;    %%Calculate reduced pressure of the gas
            Tr = T_res/Tpc; %%calculate reduced temperature of the gas

            %%Loop to calculate Z-values for PVT tables (DRANCHUK)
            A = [0.3265; -1.07; -0.5339; 0.01569; -0.05165; 0.5475; -0.7361; 0.1844; 0.1056; 0.6134;
0.721];
            [u,v] = size(P);
            c1 = A(1) + A(2)/Tr + A(3)*Tr^-3 + A(4)*Tr^-4 + A(5)*Tr^-5;
            c2 = A(6) + A(7)/Tr + A(8)/Tr^2;
            c3 = A(7)/Tr + A(8)/Tr^2;
            for h=1:u
              for g=1:v
```

```
            if P(h,g) ~= 0
               ro(h,g) = 0.27*Pr(h,g)/Tr;
               diff=1;
               while diff > 0.0001
                  f = 1 - 0.27*Pr(h,g)/(ro(h,g)*Tr) + c1*ro(h,g) + c2*ro(h,g)^2 - A(9)*c3*ro(h,g)^5 +
A(10)*(1 + A(11)*ro(h,g)^2*exp(-1*A(11)*ro(h,g)^2/Tr^3));
                  fd = 0.27*Pr(h,g)/(ro(h,g)^2*Tr) + c1 + 2*c2*ro(h,g) - 5*A(9)*c3*ro(h,g)^4 +
2*A(10)*ro(h,g)*(1+A(11)*ro(h,g)^2 - A(11)^2*ro(h,g)^4)*exp(-1*A(11)*ro(h,g)^2)/Tr^3;
                  ron(h,g) = ro(h,g) - f/fd;
                  diff = abs(ron(h,g) - ro(h,g));
                  ro(h,g) = ron(h,g);
               end
               z(h,g) = 0.27*Pr(h,g)/(ro(h,g)*Tr);
            end
         end
      end

      %%Viscosity calculation, created by Yogesh Bansal [1], used by Barak Kulga
      %%[2], from Lee

      X = 3.448 + 986.4/T_res + (0.01009*MW);
      Y = 2.447 - (0.2224*X);
      K = (9.379 + 0.01607*MW)*T_res^1.5/(209.2 + 19.26*MW + T_res);

      ro_calc = P*MW./z/10.732/T_res;
      mu_g = 10^-4*K*exp( X * (ro_calc./62.4).^Y);
      P_V_T = [P;z;mu_g];
      fprintf(fid, '\n %10.3f %5.6f %5.6f', P_V_T);
      fprintf(fid, '\n **');
      fprintf(fid, '\n **');
      fprintf(fid, '\n **');
      fprintf(fid, '\n **');
      fprintf(fid, '\n **');
      fprintf(fid, '\n **');
      fprintf(fid, '\n BWI 1');
      fprintf(fid, '\n CVW  0.0');
      fprintf(fid, '\n CW  3.51118e-006');
      fprintf(fid, '\n DENSITY WATER  62.4');
      fprintf(fid, '\n REFPW  8000');
      fprintf(fid, '\n VWI  1');
      fprintf(fid, '\n GRAVITY GAS  %d', input(i,7));
      fprintf(fid, '\n PTYPE CON 1');
      fprintf(fid, '\n ROCKFLUID');
      fprintf(fid, '\n RPT  1');

      fprintf(fid, '\n SWT');
      fprintf(fid, '\n **$       Sw      krw');
      fprintf(fid, '\n 0 0.0 0');
      fprintf(fid, '\n 1 1 0');
      fprintf(fid, '\n SGT');
      fprintf(fid, '\n **$       Sg      krg');
      fprintf(fid, '\n 0 0.0');
```

```
            fprintf(fid, '\n 1 1');
            fprintf(fid, '\n INITIAL');
            fprintf(fid, '\n USER_INPUT');
            fprintf(fid, '\n PRES CON %d', input(i,6));
            fprintf(fid, '\n SW CON  0');
            fprintf(fid, '\n NUMERICAL');
            fprintf(fid, '\n RUN');
            fprintf(fid, '\n DATE  2014  1  1');

            fprintf(fid, '\n **$');
            fprintf(fid, '\n WELL "Well-1"');
            fprintf(fid, '\n PRODUCER "Well-1"');
            fprintf(fid, '\n OPERATE  MIN  BHP  %d  CONT', input(i,8));
            fprintf(fid, '\n **$    rad geofrac     wfrac      skin');
            fprintf(fid, '\n GEOMETRY  I  0.375  0.37  1  0');
            fprintf(fid, '\n PERF  GEOA  "Well-1"');

            %%%%Code for the perforation geometry
            %define all lengths in terms of wellblocks
            n = input(i,10);
            y_loc = round(yblock/2);
            main_count = round(input(i,9)/50);
            branch_count = round(input(i,11)/50);
            wellblock = round((xblock-main_count)/2);
            junctionblock = round(wellblock+main_count*input(i,12));
            fprintf(fid, '\n **$ UBA    ff  Status  Connection');
            %First, code to draw the main lateral wellbore
            fprintf(fid, '\n %d %d 2 1. OPEN  FLOW-TO "SURFACE" REFLAYER',wellblock,y_loc);
            j=1;
            for i=wellblock+1:(wellblock+main_count)
                fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', i, y_loc, j);
                if j == junctionblock-wellblock
                   connect_to_junction = j;
                end
                j=j+1;
            end

            %Next, conditional perforation geometry based on one or two laterals.
            switch n
              case 1
                %first perforation off the junction block
                fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+1, y_loc-1,
connect_to_junction);
                j=j+1;
                %loop for subsequent blocks containing branch
                for i=2:branch_count
                   fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+i, y_loc-i, j);
                   j=j+1;
                end
              case 2
                %first perforation off the junction block, above wellbore
                fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+1, y_loc-1,
```

```
connect_to_junction);
                j=j+1;
                %loop for subsequent blocks containing branch
                for i=2:branch_count
                   fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+i, y_loc-i, j);
                   j=j+1;
                end
                %first perforation off the junction block, below wellbore
                fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+1, y_loc+1,
connect_to_junction);
                j=j+1;
                %loop for subsequent blocks containing branch, below wellbore
                for i=2:branch_count
                   fprintf(fid, '\n %d %d 2 1. OPEN FLOW-TO %d', junctionblock+i, y_loc+i, j);
                   j=j+1;
                end
             end

             %inside the conditional statement, include code to perforate at the exact
             %location determined by x(i) give by the runs/user assigned.



             fprintf(fid, '\n **'); %following is code obtaining monthly production data

             for day=2:30
                fprintf(fid, '\n DATE  2014  1  %d.00000', day);
             end
             for month=2:12
                if month == 2
                   for day=1:28
                      fprintf(fid, '\n DATE  2014  %d  %d.00000',month, day);
                   end
                else
                   for day=1:30
                      fprintf(fid, '\n DATE  2014  %d  %d.00000',month, day);
                   end
                end
             end

             for year=2015:2022
                for month=1:12
                   if month == 2
                      for day=1:28
                         fprintf(fid, '\n DATE  %d  %d  %d.00000',year, month, day);
                      end
                   else
                      for day=1:30
                         fprintf(fid, '\n DATE  %d  %d  %d.00000',year, month, day);
                      end
                   end
                end
```

```
end


fprintf(fid, '\n STOP');

fclose(fid);

end
```

Part C. Extraction of production data from '.irf' results file (generated by CMG IMEX). This code, using the "Results" module of CMG, will extract cumulative gas production and daily gas production rate for each of the simulations and output respective data to a '.txt' file. Simulation data is daily gas production data for nine years.

Table 6-4 Extraction of Production Data

```
%%% Extraction of monthly production data generated by CMG
%%% Created with reference code developed by Yogesh bansal, 2009 [1]

load input.txt

cmgdata = ['CMGEXTRACT.bat'];
fidext = fopen(cmgdata,'wt');
%Loop to extract production data from first directory of CMG IMEX output
%data
for i=1:1000;
   counter = num2str(i);
   filehandle = ['data' counter '.rwd'];
   fid = fopen(filehandle, 'wt');

   fprintf(fid, '%s','FILE "data',num2str(i));
   fprintf(fid, '%s','.irf"');
   fprintf(fid, '\nLINES-PER-PAGE 10000\n');
   fprintf(fid, '\nTIME ON\n');
   fprintf(fid, '\n*TIMES-FOR\n');


   %%TIME LOOP
   for x = 1:15
      for y = 1:108
         fprintf(fid, '%d\n',x);
         fprintf(fid, '%d\n',y*30);
      end
   end
```

```
            fprintf(fid, 'SPREADSHEET\n');
            fprintf(fid, 'TABLE-FOR\n');
            fprintf(fid, '%s\n','COLUMN-FOR *PARAMETERS "Cumulative Gas SC" *WELLS "Well-1"');
            fprintf(fid, '%s\n\n','COLUMN-FOR *PARAMETERS "Gas Rate SC" *WELLS "Well-1"');
            fprintf(fid, '%s\n','TABLE-END');
            fclose(fid);

            fprintf(fidext, '%s','call "C:\\Program Files
(x86)\\CMG\\BR\\2012.20\\Win_x64\\EXE\\report.exe" -f "data');
            fprintf(fidext, num2str(i));
            fprintf(fidext, '%s','.rwd"');
            fprintf(fidext, '%s',' -o "data_extract_case2_',num2str(i),'.txt"');
            fprintf(fidext, '\n');
        end
fclose(fidext);
```

Part D. Preparation of Data for use as input and target of neural network. Each subsequent '.txt' file containing all production rate and cumulative production data will be read into a matrix. All erroneous sets will be deleted and removed, and the matrices will be formatted for the next step, the neural network.

Table 6-5 Data Preparation for ANN Training

```
load input.txt
input = transpose(input);
q = 1;
for i=1:length(input(1,:))
   j=num2str(i);
   data = ['data_extract_case2_' j '.txt'];
   %if statement to check for bad cases that have been deleted
   if exist(data)
      [time cumgas scfd] = textread(data, '%f %f %f','headerlines',6);
      production_rate(:,i) = scfd;
      cumulativegas(:,i) = cumgas;
   else
      skipped(q,:) = i;
      q=q+1;
   end
end

save time.txt time -ASCII
save production_rate_2.txt production_rate -ASCII
save cumulativegas_2.txt cumulativegas -ASCII
```

Part E. Declaration and Training of Neural Network in Matlab. A number of parameters in this matlab code were varied, and the training was done in a series of trials to determine the optimal network configuration. The code will output the Matlab workspace containing the trained network, for further operation in the next step.

Table 6-6 Neural Network Training in Matlab

```
load input_CASE2_B.txt
load target_CASE2_B.txt

%%Add functional links, tranforming solution from non-unique to unique
output =
[target_CASE2_B(1,:)./(input_CASE2_B(110,:).*input_CASE2_B(112,:).*inpu
t_CASE2_B(113,:)); %parameter 1, A/(Lwb*n*Lb)

target_CASE2_B(1,:).*target_CASE2_B(2,:);
                        %parameter 2, A*h

target_CASE2_B(2,:).*target_CASE2_B(3,:);
                        %parameter 3, k*h

target_CASE2_B(1,:).*target_CASE2_B(2,:).*target_CASE2_B(4,:);
                         %parameter 4, A*h*phi

target_CASE2_B(6,:)./input_CASE2_B(111,:);
                        %parameter 5, pi/psf

target_CASE2_B(6,:).*target_CASE2_B(5,:);
                        %parameter 6, Pi*Ti

target_CASE2_B(5,:).*target_CASE2_B(7,:);
                       %parameter 7, Ti*g_sg

target_CASE2_B(3,:).*target_CASE2_B(6,:).*(input_CASE2_B(110,:)+input_C
ASE2_B(112,:).*input_CASE2_B(113,:))./(input_CASE2_B(1,:)+input_CASE2_B
(109,:));

input_CASE2_B(112,:).*input_CASE2_B(113,:).*input_CASE2_B(114,:)./targe
t_CASE2_B(1,:);

target_CASE2_B(6,:)./input_CASE2_B(111,:).*input_CASE2_B(114,:).*input_
CASE2_B(113,:).*target_CASE2_B(1,:)];




in = log(input_CASE2_B);
tar = log(output);

%Normalize data between -1 and 1
```

```
[irun,is] = mapminmax(in,-1,1);
[Trun,Ts] = mapminmax(tar,-1,1);

[mi,ni] = size(irun);
[mo,no] = size(Trun);

%Define variables used in the network
N_in = mi;%Number of inputs
N_out = mo;%Number of outputs
Total_in = ni;%Number of simulations

%Assign sets for training, validation, and testing
[irun_train,irun_val,irun_test,trainInd,valInd,testInd] =
dividerand(irun,0.8,0.15,0.05);
[Trun_train,Trun_val,Trun_test] =
divideind(Trun,trainInd,valInd,testInd);

val.P = irun_val;
test.P = irun_test;
val.T = Trun_val;
test.T = Trun_test;

NNeu1 = 60;
NNeu2 = 40;


netB = newff(irun,Trun,[NNeu1,
NNeu2],{'logsig','tansig'},'trainscg','learngdm','msereg');

%Training Parameters
netB.trainParam.goal = 0.00005;
netB.trainParam.epochs = 12000;
netB.trainParam.show = 1;
netB.trainParam.max_fail = 10000;
NET.efficiency.memoryReduction = 60;
netB.trainParam.showWindow = true; %Show Training Window

%Train Network
[netB,tr] = train(netB,irun_train,Trun_train,[],[],test,val);



%Save the workspace to use the network for later application
save('Case2B_FN_ANN.mat');
```

Part F. Testing of the Neural Network. This code will use the network to predict all outputs for the testing data sets, and then will compare predicted outputs with target outputs.

Table 6-7 Neural Network Testing in Matlab

```
load Case2B_FN_ANN.mat;
load time.txt;
Trun_train_ANN = sim(netB, irun_train);
test_ANN = sim(netB, irun_test);

Trun_train = mapminmax('reverse',Trun_train,Ts);
test_ANN = mapminmax('reverse',test_ANN,Ts);
Trun_train_ANN = mapminmax('reverse',Trun_train_ANN,Ts);
Trun_test = mapminmax('reverse',Trun_test,Ts);

irun_test = mapminmax('reverse',irun_test,is);

Trun_test = exp(Trun_test);
test_ANN = exp(test_ANN);
irun_test = exp(irun_test);

%Reverse Functional Links to provide output Artificial Reservoir
Parameters
%Area: ANN output 1 = A/(Lwb*n*Lb)
A_ANN =
test_ANN(1,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
A =
Trun_test(1,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
%h
h_ANN = test_ANN(2,:)./A_ANN;
h = Trun_test(2,:)./A;
%k
k_ANN =
test_ANN(3,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
k =
Trun_test(3,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
%phi
phi_ANN =
test_ANN(4,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
phi =
Trun_test(4,:).*irun_test(110,:).*irun_test(112,:).*irun_test(113,:);
%Pi
Pi_ANN = test_ANN(5,:).*irun_test(111,:);
Pi = Trun_test(5,:).*irun_test(111,:);
%Ti
Ti_ANN = test_ANN(6,:)./Pi_ANN;
Ti = Trun_test(6,:)./Pi;
%g_sg
g_sg_ANN = test_ANN(7,:)./Ti_ANN;
g_sg = Trun_test(7,:)./Ti;


%Find Percent Error For Each of the Outputs.
ANN_er_A = abs(A_ANN - A)./A.*100;
ANN_er_h = abs(h_ANN - h)./h.*100;
ANN_er_k = abs(k_ANN - k)./k.*100;
ANN_er_phi = abs(phi_ANN - phi)./phi.*100;
```

```
ANN_er_Pi = abs(Pi_ANN - Pi)./Pi.*100;
ANN_er_Ti = abs(Ti_ANN - Ti)./Ti.*100;
ANN_er_gsg = abs(g_sg_ANN - g_sg)./g_sg.*100;

props = [mean(ANN_er_A);mean(ANN_er_h);mean(ANN_er_k);mean(ANN_er_phi);
mean(ANN_er_Pi);mean(ANN_er_Ti);mean(ANN_er_gsg)];
save properties.txt props -ASCII
%Save outputs to import into Network A model for testing artificial
inputs
%production = irun_test(1:109,:);
%wellbore = irun_test(110:114,:);
%case2Ainput_ANNinput = [Trun_test_ANN; wellbore];
%case2Aoutput_target = production;
%case2input_targetinput = [Trun_test; wellbore];
```

Part G. In the final testing procedure of the neural network, outputs generated from ANN B and C will be used as inputs for ANN A, and the respective gas production schedule outputs will be compared against the target gas production profile generated by CMG.

Table 6-8 Neural Network Testing of Predicted Reservoir and Wellbore Parameters

```
load time.txt;
load Case2AANN.mat;
load case_2BTO2A_ANN_GEN_INPUT.txt;
load case_2BTO2A_TARGET_INPUT.txt;
load case_2BTO2A_CMG_OUTPUT.txt;

%Test artificially generated inputs with Case 2A Network

%input as the training set
testinput = case_2BTO2A_ANN_GEN_INPUT;
actualinput = case_2BTO2A_TARGET_INPUT;
ideal_Target = case_2BTO2A_CMG_OUTPUT;

testinput = log(testinput);
actualinput = log(actualinput);
test_target = log(ideal_Target);

[in_trial, s1] = mapminmax(testinput,-1,1);
```

```
[in_actual, s3] = mapminmax(actualinput,-1,1);
[out_trial, s2] = mapminmax(test_target,-1,1);

Trial_output = sim(net, in_trial);
Actual_output = sim(net, in_actual);

Trial_output = mapminmax('reverse',Trial_output, s2);
Actual_output_reverse = mapminmax('reverse',Actual_output, s2);
Trial_output = exp(Trial_output);
Actual_output_reverse = exp(Actual_output_reverse);

%Calculate Mean Percentage Error For Each Data Set
mpe_ANNparam = zeros(50,1);
mpe_CMGparam = zeros(50,1);
for j = 1:50
    for i = 1:109
        mpe_ANNparam(j) = mpe_ANNparam(j)+(abs(ideal_Target(i,j) -
Trial_output(i,j))/ideal_Target(i,j))*100;
        mpe_CMGparam(j) = mpe_CMGparam(j)+(abs(ideal_Target(i,j) -
Actual_output_reverse(i,j))/ideal_Target(i,j))*100;
    end
    mpe_ANNparam(j) = mpe_ANNparam(j)/109;
    mpe_CMGparam(j) = mpe_CMGparam(j)/109;
end
AV_MPE_ANNparam = mean(mpe_ANNparam);
AV_MPE_CMGparam = mean(mpe_CMGparam);



q=36
figure
plot(time, ideal_Target(:,q),'--k','LineWidth',1.5,'MarkerSize',10)
hold on
plot(time,
Actual_output_reverse(:,q),':r','LineWidth',1.5,'MarkerSize',10)
plot(time, Trial_output(:,q),':g','LineWidth',1.5,'MarkerSize',10)
hold off
%Note - Change the string label in the next line based on which run is
%selected for chart output
title('Gas Production Rate vs. Time, Case 2 Network B Generated Data,
Set #612 ')
xlabel('Time [Days]')
ylabel('Gas Production Rate [SCF/D]')
set(gca, 'XTick',[0:270:3240])
legend('CMG Output', 'ANN A Output', 'ANN B Output')
%error = num2str(AV_MPE_ANNparam);
%error2 = num2str(AV_MPE_CMGparam);
%str = ['AVG MPE, Artificial Inputs = ' error '%'];
%str2 = ['AVG MPE, CMG Inputs = ', error2, '%'];
%teststr = {str,str2};
%annotation('textbox',[0.46,0.65,0.44,0.10],'String', teststr);
thisset = {['MPE, ANN Generated Inputs = ' num2str(mpe_ANNparam(q))
'%'],[ 'MPE, CMG Inputs = ' num2str(mpe_CMGparam(q)) '%']};
annotation('textbox',[0.40,0.65,0.50,.10],'String', thisset);
```

```
ANN_er_A = abs(ideal_Target(1,:) -
Trial_output(1,:))./ideal_Target(1,:).*100;
ANN_er_h = abs(ideal_Target(2,:) -
Trial_output(2,:))./ideal_Target(2,:).*100;
ANN_er_k = abs(ideal_Target(3,:) -
Trial_output(3,:))./ideal_Target(3,:).*100;
ANN_er_phi = abs(ideal_Target(4,:) -
Trial_output(4,:))./ideal_Target(4,:).*100;
ANN_er_Pi = abs(ideal_Target(6,:) -
Trial_output(6,:))./ideal_Target(6,:).*100;
ANN_er_Ti = abs(ideal_Target(5,:) -
Trial_output(5,:))./ideal_Target(5,:).*100;
ANN_er_gsg = abs(ideal_Target(7,:) -
Trial_output(7,:))./ideal_Target(7,:).*100;

props = [mean(ANN_er_A);mean(ANN_er_h);mean(ANN_er_k);mean(ANN_er_phi);
mean(ANN_er_Pi);mean(ANN_er_Ti);mean(ANN_er_gsg)];
save properties.txt props -ASCII
```

REFERENCES

[1] Y. Bansal. Conducting in-situ combustion tube experiments using artificial neural networks. Master's thesis, The Pennsylvania State University, 2003.

[2] I. B. Kulga. Development of an artificial neural network for hydraulically fractured horizontal wells in tight gas sands. Master's Thesis, The Pennsylvania State University, 2010.

[3] X. Yu and B. Guo. A Comparison between Multi-Fractured Horizontal and Fishbone Wells for Development of Low-Permeability Fields. SPE 120579, Society of Petroleum Engineers 2009.

[4] Husain, Taha Murtuza, Leong Chew Yeong, Aditya Saxena, Ugur Cengiz, Sarath Ketineni, Amey Khanzhode, and Hadi Muhamad. "Economic Comparison of Multi-Lateral Drilling over Horizontal Drilling for Marcellus Shale Field Development." EME 580 Final Report, The Pennsylvania State University. (2011): n. page. Web. 17 Nov. 2013. <http://www.ems.psu.edu/~elsworth/courses/egee580/2011/Final Reports/fishbone_report.pdf>.

[5] Bosworth, Steve, Hussein Saad El-Sayed, Gamal Ismail, Herve Ohmer, Mark Stracke, Chris West, and Albertus Retnanto. "Key Issues in Multilateral Technology." *Oilfield Review*. 10.4 (1998): n. page. Web. 17 Nov. 2013. <http://www.slb.com/resources/publications/industry_articles/oilfield_review/1998/or1998win02_multilateral_technology.asp&xgt;.

[6] Naik, G.C. "Tight Gas Reservoir- An Unconventional Natural Energy Source for the Future." *Pinedale Online*. (2003): <www.pinedaleonline.com/socioeconomic/pdfs/tight_gas.pdf>.

[7] Law, B. E., and J. B. Curtis, 2002, Introduction to Unconventional Petroleum Systems: AAPG Bulliten, v. 86, p. 1851 – 1852.

[8] Seto, Carolyn. "Role of Technology in Unconventional Gas Resources." *Future of Natural Gas, MIT.* (2011): Web. 16 Nov. 2013. <http://mitei.mit.edu/publications/reports-studies/future-natural-gas>.

[9] Khlaifat, Abdelaziz. "Taking Up Unconventional Challenge is a Game Changer in Oil and Gas Industry." n. page. Web. 16 Nov. 2013. <http://www.jeaconf.org/UploadedFiles/Document/f64916ee-9c55-44eb-8c13-3105ccb413c9.pdf>.

[10] Helms, Lynn. "Horizontal Drilling." *DMR Newsletter*. Vol. 35.No. 1 n. Web. 16 Nov. 2013. <https://www.dmr.nd.gov/ndgs/newsletter/NL0308/pdfs/Horizontal.pdf>.

[11] "TAML." *Technology Advancement for Mulit-Laterals*. TAML. Web. 17 Nov 2013. <http://taml-intl.org/taml-classification/>.

[12] Xing, Guangyu, and Fuxiang Guo. "Fishbone Well Drilling and Completion Technology in Ultra-Thin Reservoir." *SPE/IADC Paper 155958*. (2012): n. page. Print.

[13] “Discover the processes at Pearl GTL”. Shell. 14 Mar. 2014. <http://www.shell.com/global/future-energy/natural-gas/gtl/acc-gtl-processes.html>.

[14] van Rooij, A. J. F., L. C. Jain, and R. P. Johnson. *Neural Network Training Using Genetic Algorithms*. Singapore: World Scientific, 1996. Print.

[15] P. Thararoop. A neural network approach to predict well performance in conjunction with infill drilling strategies. Master’s thesis, The Pennsylvania State University, 2007.

[16] Beale, Mark Hudson, Martin T. Hagan, and Howard B. Demuth. *Matlab Neural Network Toolbox User's Guide*. R2014a. MathWorks, 2014. Web.

[17] Hagan, Martin T., Howard B. Demuth, and Mark Hudson Beale. *Neural Network Design*. Boulder, Colorado: Campus Publishing Service, Colorado University Bookstore, Print.

ACADEMIC VITA

Eric David Schumacker

424 Waupelani Drive Apartment L-31
State College, PA 16801
eds5116@psu.edu
eric.schumacker@gmail.com

Education

B.S., Petroleum and Natural Gas Engineering, 2014, The Pennsylvania State University, University Park, PA

Minor, Chinese, 2014, The Pennsylvania State University, University Park, PA

Honors and Awards

Victor and Anna Mae Beghini PNG Scholarship for Petroleum and Natural Gas (Fall 213, Spring 2014)
Shell Technical Scholarship, Shell Oil Company (Fall 2103, Spring 2014)
Hess Scholarship in Earth and Mineral Engineering (2011, 2012)
College of Earth and Mineral Sciences General Academic Scholarship (2010 through 2014)
Dean's List (Every semester)
President Sparks Award (Spring 2012)
President's Freshman Award, United State Government (2010)
Schreyer Honors College Excellence Scholarship (2010 through 2013)

Association Memberships/Activities

Society of Petroleum Engineers (2010 to Current)
SPE Petrobowl Penn State Team (2012 to 2014)
Penn State Investor's Association (2010 to 2012)

Research Experience

Undergraduate Research Assistant, Mallouk Research Lab, Penn State Department of Chemistry (2011)

Professional Experience

Shell Oil Company, Pittsburg, PA May 2014 – August 2014 Drilling Engineering Intern

International

CIEE Study Abroad, Shanghai, China (December 2011 – June 2012)
Habitat for Humanity, Chengdu, China (March, 2012)

Language Proficiency

Mandarin Chinese – Native conversational skill, Advanced Reading and Writing

Honors Research and Investigative Repots

Drilling Fluid Selection for Clay Rich Shale Formations, Dr. John Yilin Wang, Spring 2013

Real-Time Logging Applications: Measurement While Drilling (MWD) and Logging While Drilling (LWD), Dr. John Yilin Wang, Fall 2013

Well Test Analysis of Shale Gas Wells, Dr. Turgay Ertekin, Spring 2014

Surface Production Facility Equipment Design and Application, Dr. Luis F. Ayala H., Spring 2014