ESTIMATION OF COEFFICIENT OF ISOTHERMAL OIL COMPRESSIBILITY AT RESERVOIR PRESSURE GREATER THAN BUBBLE POINT PRESSURE USING GROUP METHOD OF DATA HANDLING

POH CHUAN SIEU

MSc. PETROLEUM ENGINEERING

UNIVERSITI TEKNOLOGI PETRONAS

JULY 2014

CERTIFICATION OF APPROVAL

Estimation of Coefficient of Isothermal Oil Compressibility at Reservoir Pressure Greater Than Bubble Point Pressure Using Group Method of Data Handling.

by

Poh Chuan Sieu

A project dissertation submitted to the

Petroleum Engineering Programme Universiti Teknologi PETRONAS

in partial fulfilment of the requirement for the

MSc. of PETROLEUM ENGINEERING

Approved by,

_____

(Dr. Mohammed Abdalla Ayoub Mohammed)

(Project Supervisor)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

July 2014

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

POH CHUAN SIEU

ACKNOWLEDGEMENT

I would like to take this opportunity to thank the people who have helped me along the route to completion of this project. First of all, I would like to express my sincere appreciation to my Individual Project Supervisor, Dr. Mohammed Abdalla Ayoub Mohammed for his sincere guidance, advices and continuous support throughout the project. Without him, I would not be able to finish my Individual Project on time.

In addition, I would like to express my sincere thanks to my friends who have helped me a lot along the way. Last but not least, deepest gratitude goes to my family members for their morale support, motivation and encouragement during the duration of this project.

ABSTRACT

The procedures of determining the oil compressibility through PVT analyses are usually costly and time consuming. Therefore, various empirical correlations were designed in order to accurately estimate the oil compressibility. This project focuses on designing a model correlation which could accurately predict the oil compressibility coefficient above bubble point pressure. The new model correlation using the Polynomial GMDH method will be compared in terms of its accuracy and reliability with actual PVT data and also current oil compressibility correlation using different types of analysis such as trend analysis, group error analysis, statistical error analysis and graphical error analysis. A total number of 195 data sets were collected from the Mediterranean Basin, Africa, Persian Gulf and the North Sea and after data filtration 183 data were used and divided into 3 sections of training, validation and testing data sets in a ratio of 2:1:1. Using the Polynomial GMDH technique, 3 input parameters were found to be affecting the outputs which are Reservoir Pressure, Solution GOR and Bubble Point Pressure. The new model correlations were then compared with the other correlations and it surpasses all of the other correlations in terms accuracy by having the lowest Root Mean Square Error, Average Percent Relative Errors, Average Absolute Percent Relative Error and Standard Deviations. Furthermore, it is worth to note that the Absolute Percent Relative Error was the main statistical analysis criteria in this study and the Polynomial GMDH model obtained the lowest value. Overall, the Polynomial GMDH model is a robust model and can be affectively applied within its trained data ranges.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AAPE	Average Absolute Relative Error
APE	Average Relative Error
API	American Petroleum Institute
PVT	Pressure-Volume-Temperature
GMDH	Group Method of Data Handling
GOR	Gas Oil Ratio
GUI	Graphical User Interface
IRC	Information Resource Centre
MATLAB	Matrix Laboratory
PD	Partial Description
PNN	Polynomial Neural Networks
RMSE	Root Mean Square Error
SD	Standard Deviation

NOMENCLATURE

Symbol	Definition
c_o	Oil Compressibility (psi^{-1})
$\left(\frac{\delta V}{\delta p}\right)_T$	Slope of isothermal pressure-volume curve
$\left(\frac{\delta Bo}{\delta p}\right)_T$	Slope of isothermal pressure-volume formation factor curve
$\left(\frac{\delta \rho o}{\delta p}\right)_T$	Slope of pressure-density curve
R_s	Gas Solubility (scf/STB)
R_{sb}	Gas solubility at bubble point pressure (scf/STB)
T	Temperature (°F)
P	Pressure (psia)
γ_{gs}	Corrected gas gravity
γ_{API}	Oil API Gravity (°API)
T_{sep}	Separator Temperature (°F)
P_{sep}	Separator Pressure (psia)
c_{ofb}	Oil compressibility using extension of values from bubble point to higher pressure of interest
c_{ofi}	Oil compressibility using initial pressure to estimate oil compressibility at lower pressure
E_i	Relative deviation of the predicted value from the experimental values
E_r	Average Percent Relative Error (%)
E_a	Average Absolute Percent Relative Error (%)
$(c_o)_{meas}$	Measured oil compressibility (psi^{-1})
$(c_o)_{pred}$	Predicted oil compressibility (psi^{-1})
E_{min}	Minimum Absolute Percent Relative Error (%)

E_{max}	Maximum Absolute Percent Relative Error (%)
R	Correlation Coefficient

CHAPTER 1

INTRODUCTION

1.1 Background

The isothermal compressibility of oil is defined as the fractional change of fluid volume per unit change in pressure at constant reservoir temperature and is denoted with the symbol c_o [1]. When pressures exceed bubble point, it can be assumed that the coefficient of isothermal compressibility of oil is the same as that of the coefficient of isothermal compressibility of a gas. However at pressures below the bubble point an additional term has to be added to take into account for the volume of gas which evolves from it [2]. However in this project, it only focuses on the isothermal coefficient of oil compressibility above bubble point pressure. There are various applications of compressibility such as determining the extension of fluid properties from correlations starting at bubble point pressure, to derive a material balance equation for under saturated oil reservoirs and to determine the tangent compressibility for theoretical work [3]. The basic equation to estimate the coefficient of isothermal compressibility of reservoir fluids for a finite change in pressure and volume above bubble point pressure is: -

$$\mathbf{c_o = -\frac{1}{V}\left(\frac{\delta V}{\delta p}\right)_T} \qquad \mathbf{(1.1)}$$

Where,

c_o = isothermal compressibility of oil, psi^{-1} and

$\left(\frac{\delta V}{\delta p}\right)_T$ = slope of the isothermal pressure-volume curve

The isothermal oil compressibility can also be written in terms of formation volume factor, B_o as well as density of oil, ρ_o as given:-

$$\mathbf{c_o = -1/B_o\left(\frac{\delta Bo}{\delta p}\right)_T} \qquad \mathbf{(1.2)}$$

$$\mathbf{c_o = 1/\rho_o\left(\frac{\delta \rho o}{\delta p}\right)_T} \qquad \mathbf{(1.3)}$$

However problems are sometimes encountered when constructing the slope of the isothermal pressure-volume curve since the pressure and volume data are usually taken from laboratory studies on samples which are taken from the wellbores as well as at surfaces. There is a downside to this process as the reservoir samples collected might be jeopardized due to human errors which may result in inaccurate readings. The cost of taking the samples is also expensive which might not be the best of interest of the oil companies. Furthermore, PVT analyses are usually not ready when the data are needed and also to obtain an accurate PVT behavior of each reservoir fluid is costly and time consuming [1]. Therefore, in cases where experimental data are not available, the coefficient of isothermal oil compressibility is determined from empirically derived correlations or equation of state. Therefore in this project, the main objective is to estimate the coefficient of isothermal oil compressibility at reservoir pressure greater than bubble point pressure using Group Method of Data Handling.

1.2 Problem Statement

There are several problems which need to be analyzed in the process of completing this project. The problem statements include:-

1. Laboratory PVT analysis are not available whenever fluid properties such as coefficient of oil compressibility are needed.
2. Difficulty in predicting the oil compressibility due to the complexity of the relationship between the various parameters involved.
3. The reliability of the existing oil compressibility correlations have limitations.

1.3 Objectives and Scope of Study

The main objective of the project is to come up with a new correlation using Polynomial GMDH techniques by modeling it with MATLAB software and compare the accuracy of the results with the current available correlations. To reduce the complexity of this project, the new correlation will only look at coefficient of isothermal oil compressibility above bubble point pressures.

In order to achieve the goal of this project, several objectives were outlined as such:

1. To model a new correlation using Polynomial GMDH technique utilizing MATLAB software by applying the appropriate level of detail.
2. To test the reliability and accuracy of the new model compared to actual PVT data as well as the existing correlations by using several statistical error analysis, group error analysis and graphical error analysis.
3. To identify any problems or constraints faced in modeling or implementing the new correlations

1.4 Benefits of the Research

There are three benefits identified by doing this research, those are:-

1. The new correlation model will assist in the prediction of isothermal oil compressibility with reduced dependency on PVT data.
2. To explore the potential of using Polynomial GMDH techniques in this era.
3. To determine the potentially best empirical correlations to predict isothermal oil compressibility above bubble point pressure.

CHAPTER 2

LITERATURE REVIEW

2.1 Existing Correlations

There are several existing correlations developed in order to overcome problems faced in the field when laboratory PVT analyses are not available. Several correlations were developed by using the fluid properties available in order to estimate the coefficient of oil compressibility. However each of these correlations are accurate up to a certain degree of accuracy compared to the actual data.

2.1.1 Vasquez-Begg's Correlation

Vasquez and Beggs used a total of 4036 experimental data points in a linear regression model to correlate the isothermal oil compressibility coefficients using Rs, T, °API gravity, γ_g and p. Their proposed correlations are given below:-

$$c_o = \frac{-1433+5R_s+17.2(T-460)-1180\gamma_{gs}+12.61\gamma_{API}}{10^5 p} \quad (2.1)$$

Vasquez and Beggs added the adjustment of gas gravity to give a more accurate prediction. The proposed relationship is for the adjustment of gas gravity γ_g to the reference separator pressure γ_{gs} as follows:-

$$\gamma_{gs} = \gamma_g \left[1+5.912\,(10^{-5})(\gamma_{API})(T_{sep} - 460)\mathrm{Log}\left(\frac{P_{sep}}{114.7}\right)\right] \quad (2.2)$$

The pressure limits for Vasquez-Beggs correlation are as follows:- [5]

$140.7 \leq p\ (psia) \leq 9514.7$

2.1.2 Petrosky and Farshad Correlation

The Petrosky and Farshad correlation can be used to estimate the oil compressibility above bubble point. The correlation was developed using samples from the offshore regions mainly in Texas and Louisiana. The authors claim

improved results over those done by Standing, Vasquez and Beggs, Glaso and Al-Marhoun in the Gulf of Mexico region.[5] The correlation is as follows:-

$$\mathbf{c_o = [(1.705*10^{-7})(R_{sb}^{0.69357})(\gamma_g^{0.1885})(\gamma_{API}^{0.3272})(T - 460^{0.6729})(p^{-0.5906})]} \quad \mathbf{(2.3)}$$

However there is a range of values which can be utilized by this correlation which are:- [5]

$\mathbf{2.464*10^{-5} < c_o < 3.507*10^{-5}}$

$\mathbf{114 \leq T\ (°F) \leq 288}$

$\mathbf{1700 \leq p\ (psia) \leq 10692}$

$\mathbf{217 \leq Rs \leq 1406}$

2.1.3 De Ghetto et al Correlations

De Ghetto et al (1994) characterize the fluid samples used in their studies to 4 types of oil which are extra-heavy oils (°API ≤ 10), heavy oils (10 < °API ≤22.3), medium oils (22.3 < °API ≤ 31.1) and light oils (°API > 31.1). He developed the correlations from reservoir fluid samples taken from the Mediterranean Basin, Africa and Persian Gulf. [5] The reported improvements from errors were decreased by five percentage points. In contrast to other existing correlations, De Ghetto correlation requires the pressure and temperature at the separator. The advantage of these correlations is that the oil correlations are more specific for each oil API gravity range. The modified correlations are as follows:-

1. **Extra-heavy oils (Modified Vasquez-Beggs Correlation)**

$$\mathbf{c_o = \frac{-889.6+3.1374R_s+20T-627.3\gamma_{georr}-81.447\gamma_{API}}{p10^5}} \quad \mathbf{(2.4)}$$

2. **Heavy oils (Modified Vasquez-Beggs Correlation)**

$$\mathbf{c_o = \frac{-2841.8+2.9646R_s+25.5439T-1230.5\gamma_{georr}-41.91\gamma_{API}}{p10^5}} \quad \mathbf{(2.5)}$$

3. Medium oils (Modified Vasquez-Beggs Correlation)

$$c_o = \frac{-705.288+2.2246R_s+26.0644T-2080.823\gamma_{georr}-9.6807\gamma_{API}}{p10^5} \quad (2.6)$$

4. Light oils (Modified Labedi's Correlation)

$$c_o = (10^{-6.1646}B_o^{1.8789}\gamma_{API}^{0.3646}T^{0.1966}) - (1-\frac{p_b}{p})(10^{-8.98}B_o^{3.9392}T^{1.349}) \quad (2.7)$$

Where,

$\gamma_{georr} = \gamma_g\,[1+0.5912\,\gamma_{API}T_{sp}Log(\frac{P_{sp}}{114.7})10^{-4}]$

The oil correlation limits for De Ghetto (Heavy and Extra Heavy Oils) are as follows:- [5]

131.4≤ T (°F) ≤250.7

1038.49 ≤ p (psia) ≤7411.54

17.21 ≤ Rs ≤ 640.25

2.1.4 Dinodruk-Christman Correlation

The correlation proposed by Dinodruk and Christman (2001) predicts the oil compressibility values with an average relative error of -0.85% and an average absolute relative error of 6.21%.[1] The following correlations are for undersaturated oil reservoirs.

$$(c_o)_{bp} = (4.487462368 + 0.005197040A + 0.000012580A^2) \times 10^{-6} \quad (2.8)$$

Where,

$$A = \frac{(\frac{R_s^{0.980922372}\gamma_g^{0.021003077}}{\gamma_o^{0.338486128}}+20.00006358(T-60)^{0.300001059}-0.876813622R_s)}{(2.749114986+(T-60)\frac{2R_s^{-1.713572145}}{\gamma_g^{9.999932841}})^2}$$

2.1.5 Spivey et al. Correlation

Spivey et al. (2005) introduced a set of correlations for estimating the isothermal compressibility for 3 applications in reservoir engineering which are to determine the isothermal compressibility from the bubble point to a pressure of interest, to determine the isothermal compressibility from the initial pressure to a

pressure of interest and to determine the isothermal compressibility tangent at some pressure of interest. [2]

1. **Determination of Isothermal Compressibility from Bubble Point to Pressure of interest.**

   This correlation is used to estimate the oil compressibility using extension of values of some fluid properties from bubble point to a higher pressure of interest

$$\ln(c_{ofb}) = 2.434 + 0.475z + 0.048z^2 \quad (2.9)$$

$$z = \sum_{n=1}^{6} z_n \quad (2.10)$$

$$z = C_{o,n} + C_{1,n}\, x_n + C_{2,n}\, x_n^{\,2} \quad (2.11)$$

Table 2.1 shows the coefficients listed to calculate value of z_n for each of the 6 independent variables.

Table 2.1: Coefficients for the c_{ofb} Correlation [2]

Coefficients for the c_{ofb} Correlation				
n	x_n	C_{0n}	C_{1n}	C_{2n}
1	ln °API	3.011	-2.6254	0.497
2	ln γ_{gsp}	-0.0835	-0.259	0.382
3	ln p_b	3.51	-0.0289	-0.0584
4	ln p/p_b	0.327	-0.608	0.0911
5	ln R_{sb}	-1.918	-0.642	0.154
6	ln T_R	2.52	-2.73	0.429

2. **Determination of Isothermal Compressibility from Initial Pressure to a Pressure of Interest**

   This correlation uses the values from isothermal oil compressibility at initial pressure, c_{ofb} to estimate isothermal compressibility at a lower pressure, c_{ofi}.

$$c_{ofi} = \frac{(p-p_b)c_{ofb}p-(p_i-p_b)c_{ofb}p_i}{p-p_i} \quad (2.12)$$

3. Determination of Isothermal Compressibility Tangent at Some Pressure of Interest

This correlation gives the tangent or instantaneous compressibility, c_o.

$$c_o = c_{ofb} + (p-p_b)c_{ofb}\,(0.475+0.096z)\frac{-0.608+0.1822ln\frac{p}{p_b}}{p} \qquad (2.13)$$

2.2 Polynomial GMDH Technique

Group method of Data Handling (GMDH) was developed in the late 1960s by Ivakhnenko to identify the relationships between nonlinear input and output.[7] The GMDH method will generate an optimal structure of the model through successive generations of partial descriptions (PDs) of data using quadratic regression polynomials with 2 input variables [8]. The disadvantage of this method is that it tends to generate complex polynomial for simple data. GMDH also has the tendency to generate complex network models when it comes to highly nonlinear systems. Not only that, GMDH will not produce a versatile structure if there are less than 3 input variables. [8]

The Polynomial Neural Networks (PNNs) uses the concept of the GMDH model where it uses polynomials similar to GMDH such as linear, modified quadratic, cubic, etc.[8] After the selection of the important input variables followed by an order of polynomial are chosen, the vital inputs will be chosen from the extracted PDs according to the selected nodes of each layer. Additional layers will be generated until the model reaches its optimum state. Optimum state is a state whereby any additional layers added will not generate a better prediction value. **Figure 2.1** shows the overall structure of the PNN.

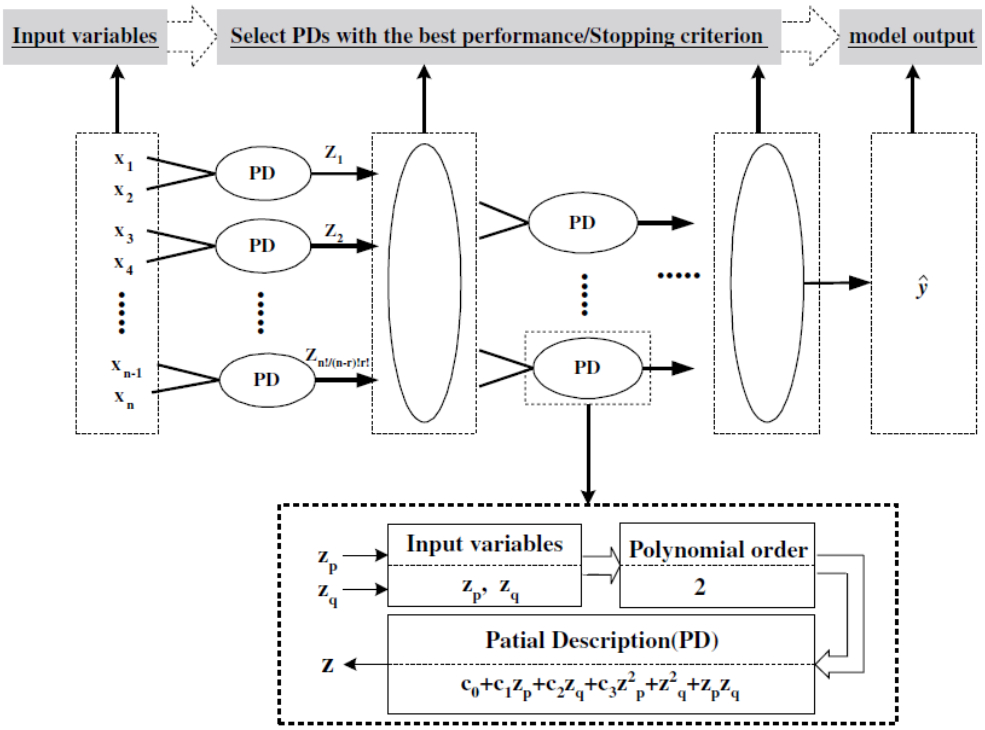

Figure 2.1: Structure of the Polynomial Neural Network [8]

Thus, various techniques have been developed to further improve GMDH technique one of them is where it will be combined with Polynomial Neural Network which will be currently used in this project which is also known as the Polynomial Group Method of Data Handling Technique. It utilizes smart type of regression where it will take 3 steps to reach the final output which consists of representation, selection and stopping. Representation is a phase whereby the network tries all the possible combinations from the input layer in order to achieve the actual oil compressibility values. Next is the selection phase whereby the critical parameters or parameters which have a large impact on the final output are chosen. These selections are done with minimal interference from the user. The iteration process starts here where it uses the "regularity criterion" such as average absolute percentage error to reduce the error between the actual and estimated value at each layer or iteration within a threshold value. [9] The codes function in such a way that it starts with a simple regression relationship and starts deriving a better representation between the input and output with each iteration. Finally is the stopping phase where, the network will stop when the new generated regression equations start to have poorer prediction performance compared to the previously generated predictions.

2.3 Types of Analysis

2.3.1 Trend Analysis

To check whether generated model is physically correct and obeying the real behavior of the input parameter being used. To test the input parameter, the said parameter is set to vary in values where all other parameters are set to a constant mean value. A graph is then plotted for the input parameter value against the oil compressibility and the trend is observed and analysed.

2.3.2 Group Error Analysis

To demonstrate the reliability of the developed model where errors were quantified by each input and grouped to several classes based on average absolute relative error as the indicator. Average absolute relative error was chosen as the indicator as it is a reliable indicator for empirical correlations and for the new developed model being tested. These is done by grouping the input parameters to a few ranges and see the average absolute relative error for each model according to the range and decide which model has the lowest average absolute relative error for the defined range.

2.3.3 Statistical Error Analysis

Error analysis to check for the accuracy of the models such as average percent relative error, average absolute percent relative error, minimum and maximum absolute percent error, root mean square error, standard deviation of error and the correlation coefficients. Equations are as below:-

1. **Average Percent Relative Error (APE)**

Used to measure the relative deviation from the experimental data and is given by the equation:-

$$E_r = \frac{1}{n}\sum_{i=1}^{N} E_i \qquad (3.1)$$

Where E_i is the relative deviation of the predicted value from the experimental values and is given by this equation:-

$$Ei = \left[\frac{(C_o)_{meas} - (C_o)_{pred}}{C_{o_{meas}}}\right] X100\% \qquad i = 1,2,3,4,\ldots n \qquad (3.2)$$

Where,

$(c_o)_{meas}$ is the measured oil compressibility

$(c_o)_{pred}$ is the predicted oil compressibility

2. Average Absolute Percent Relative Error (AAPE)

Used to measure the relative absolute deviation from the experimental data and is given by the equation:-

$$E_a = \frac{1}{n}\sum_{i=1}^{N} |E_i| \quad (3.3)$$

This type of statistical error analysis will be used for most of the sections in this study as it is a good representation of variability of each sample of data. Furthermore, it is easy to calculate and easy to interpret as it is in percentage units.

3. Minimum Absolute Percent Relative Error

$$E_{min} = min_{i+1}^{n}|E_i| \quad (3.4)$$

4. Maximum Absolute Percent Relative Error

$$E_{max} = max_{i+1}^{n}|E_i| \quad (3.5)$$

5. Root Mean Square Error (RMSE)

RMSE is used to measure the dispersion around the zero deviation and is given in the equation below:-

$$RMSE = \sqrt{\left[\frac{1}{n}\sum_{i=1}^{N} {E_i}^2\right]} \quad (3.6)$$

6. Standard Deviation

Standard deviation is used to measure the dispersion and is given in the equation below:-

$$SD = \sqrt{\sum_{i=1}^{n} \frac{[\bar{x}_{i\ errors} - x_{i\ errors}]^2}{n-1}} \qquad i = 1,2,3,4 \ldots n$$

where,

$x_{i\ errors} = (c_o)_{meas} - (c_o)_{pred}$

7. Correlation Coefficient

It is used as a measure of comparison of the actual value to the predicted value and is defined by the equation below:-

$$R=\sqrt{1-\frac{\sum_{i=1}^{n}[(c_o)_{meas}-c_{o\ pred}]}{\sum_{i=1}^{n}[(c_o)_{meas}-\overline{\Delta c_o}]}}$$

Where,

$$\overline{\mathbf{\Delta c_o}} = \frac{\mathbf{1}}{\boldsymbol{n}}\sum_{i=1}^{N}[\mathbf{\Delta}c_{o\ meas}]_I$$

The R value can vary from 0 to 1 whereby the closer the value is to 1 the better or more accurate the correlation is to the actual value whereas for a value of 0 means there is no correlation at all among the independent variables. Therefore the R value is a good indicator for a goodness of fit.

2.3.4 Graphical Error Analysis

This type of analysis uses visualization to analyse the performance and accuracy of the correlation models. There are 3 types of graphical error analysis utilized in project which are the cross-plots, error distribution and the residual analysis.

1. Cross-plots

This type of analysis is used to compare the predicted values and the actual values by plotting the predicted values against the latter in a cross plot form and with the aid of a 45° straight line drawn to indicate the perfect correlation line. The closer the unity slope line is to the 45° straight line the more accurate the prediction is to the actual values.

2. Error Distributions

This type of analysis is done to observe the error distribution in the form of histograms. The normal distribution curve will be fitted on the graph and the errors are considered normally distributed if the mean around the 0% and the standard deviation is equal to 1.0. This is a good indicator to observe if the model have skewed error distributions.

3. Residual Analysis

This type of analysis shows the relative difference of the measured with the actual value around the zero line to verify if the models have error trends. This type of analysis is also good to check for model deficiencies.

CHAPTER 3

METHODOLOGY

3.1 Project Planning

After defining the problems and objectives that need to be achieved in Chapter 1, the next step is conducting the literature review on publications, previous works and books relevant to the project. The time allocated for each section is sufficient in order to meet with the project dateline for each chapter. For the studies done by previous authors, most of the information is obtained from books and journals which can be found in the Information Resource Centre (IRC) as well as websites.

Once a better understanding of the methods to determine oil coefficient is attained, the next steps to determine the important parameters from the existing methods which will be used as a screening process in our methodology section. After writing the methodology, the next step is data collection and the simulation process. For the Chapter 4 which is the results and discussion section, the new correlation is compared with the existing correlations using several graphical, statistical and group error analysis. The final Chapter 5 is where the recommendations and conclusions are summarized.

3.2 Project Resources

Resources are defined as an input required by an individual to generate the desired output. It can vary in different forms such as machine, man and money. Thus, it is important to identify the resources needed for the implementation of the work before taking in account the magnitude of the project. Since the type of project conducted is a simulation project, the amount of resources needed is limited.. Given that there is no need for purchasing of materials in this project, resources such as money or capital is not critical and thus will not be discussed.

3.3 Equipment/Software Simulation Software (MATLAB 7.10.0 r2010a)

Since there is no need for physical experimentation in this project, the main equipment used in this project is the author's laptop. However, for the simulation stage, a simulation package (MATLAB 7.10.0 r2010a) was employed to carry out these activities. MATLAB is software which utilizes a high-performance language for technical computing. It integrates computation, visualization and programming to solve problems and the solutions are expressed in mathematical notations. The typical uses of the software include: [6]

1. Mathematics and Computation
2. Developing Algorithms
3. Modelling, simulation and prototyping
4. Data analysis, exploration and visualization
5. Scientific and engineering graphics
6. Application development which includes the Graphical User Interface building

MATLAB allows solving of many technical computing problems specifically those using matrix and vector formulation in a fraction of time compared to other non-interactive language software. For this project, MATLAB is used to come up with the most suitable correlation by applying the GMDH Polynomial Method and using the critical parameters to determine oil compressibility with the highest accuracy above bubble point pressure compared to actual value using a set of PVT data.

3.4 Parameters Screening

Table 3.1 shows the screening of existing correlations done by various authors of 5 different correlations, namely Vasquez-Beggs, Petrosky and Farshad, De Ghetto, Dinodruk-Christman and Spivey. The all 5 correlations are listed out and the parameters used are identified and summarized in the table. Then the parameters that are used by 2 or more authors are identified as crucial parameters which will be used in the simulation process using MATLAB. To identify these parameters, it is done manually by selecting only parameters used by the authors and the parameters that is

overlapped by more than 2 types of correlations is then selected as the critical parameters.

Table 3.1: Screening of Existing Criteria

Parameters / Correlation	Temp, T	Pressure, p	Bubble point pressure, P_b	Oil °API gravity, γ_{API}	Gas specific gravity, γ_g	Oil specific gravity, γ_o	Separator gas specific gravity, γ_{gsp}	Gas solubility, R_s	Oil volume formation factor, B_o	Separator Temp. T	Separator Pressure, P
Vasquez-Beggs	Y	Y		Y	Y			Y		Y	Y
Petrosky and Farshad	Y	Y		Y	Y			Y			
De Ghetto	Y	Y	Y	Y	Y			Y	Y	Y	Y
Dinodruk-Christman	Y				Y	Y		Y			
Spivey	Y	Y	Y	Y			Y	Y			

Y = Parameters used

Based on the screening criteria, 8 critical parameters were identified which are the reservoir temperature, reservoir pressure, bubble point pressure, API gravity, gas specific gravity, solution GOR, separator temperature and separator pressure. Therefore, the using the Polynomial GMDH Method, MATLAB will utilize these 8 inputs and determine the number of critical parameters used to determine the best correlation by generating a relationship between the input and the output variables.

3.5 Project Process Flow

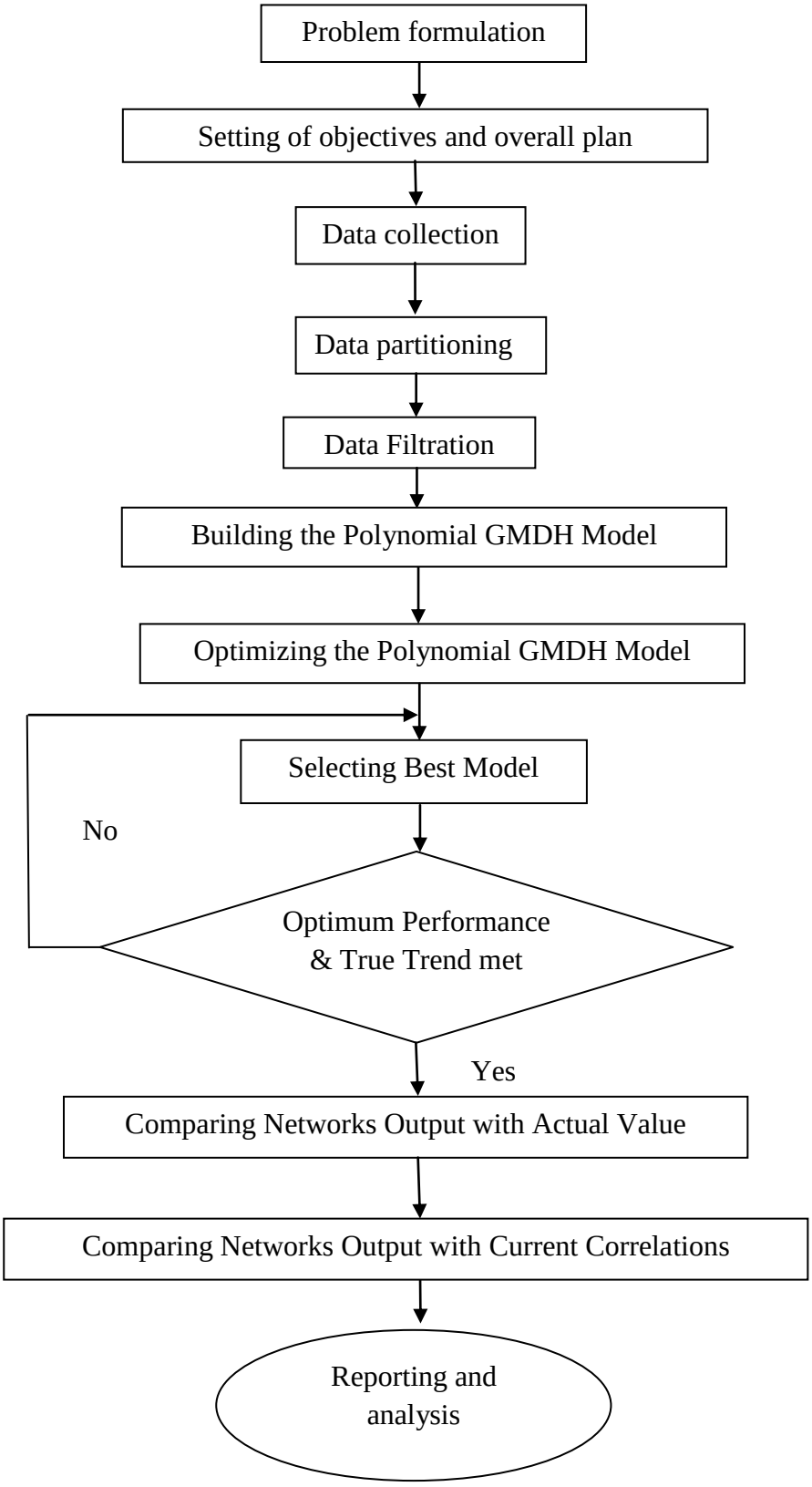

Figure 3.1: Project Process Flow

1. **Problem Formulation**

It is essential to first understand the problem statement. If the problem statement was developed by the analyst or in this case the author, then it is essential for the author to understand and agree with the formulation. In order to achieve the main goal of this project, 3 problems statements identified.

2. **Setting of Objectives and Overall Project Plan**

The second stage is to come up with the objectives and overall project plan where 3 main objectives were set. These objectives will assist the process of solving the problem statement set earlier. In setting the overall project plan, other aspects must be put into considerations such as the cost of this study and also the time aspect which includes the duration to complete this project.

3. **Data Collection**

The purpose of this project is for the validation of numerical predictions using MATLAB software with experimental measurements acquired from PVT data. The data collection is a tedious process where the data of the crucial parameters need to be collected from dependable sources. The source of the data is taken from an SPE paper consisting of 195 crude oil samples from the Mediterranean Basin, Africa, the Persian Gulf and the North Sea. [11] It is necessary to begin data collection at the early stages of the project as it takes an enormous amount of time to get the suitable data. The data sets used in this study is in **APPENDIX A**.

4. **Data Partitioning**

Data partitioning is a phase whereby the total number of data are divided accordingly to this ratio of 2:1:1 which are training, validation and testing data sets respectively. Training data is the main data sets which will be utilized by the software to develop and adjusts the weights in the network, thus the highest amount of data is given to this particular data set. As for validation data set, it is used during the training phase to ensure the optimum generalization of the developed network.[9] The test data set is however used for examining the final performance of the network and it is not seen by the network during the training phase. The testing data sets are also used for comparing the performance of existing correlations to the developed

Polynomial GMDH model as this data set is independent as it is not used to develop the model. Thus, for this project, the number of data allocated for each sets are 91, 46 and 46 for training, validation and testing data sets respectively.

5. Data Filtration

Data filtration also involves removing data that is incomplete as well as outliers. Outliers will affect the output of the generated model and is usually caused by noise when measuring. Another reason of anomalies can be due to errors or improper methods to collect these data. Outliers are usually filtered using linear regression as shown in **Figure 3.2**. The solid line shows the most ideal fit for the data while the dotted line shows the confidence level.[9] The dot in black is the outlier point which is removed from the main data sets. A total of 2 data sets were removed due to outliers and an additional 10 data sets were removed due to missing data. Outliers are identified by running the data sets for all existing correlations as well as the newly developed model and values which have AAPE value of more than 40% for all of the correlations are then identified as outliers. Missing data on the other hand are data sets which are incomplete for example data sets which have missing bubble point pressure, solution GOR, etc. Thus from an initial 195 data sets, 183 data sets were used in this project after data filtration.

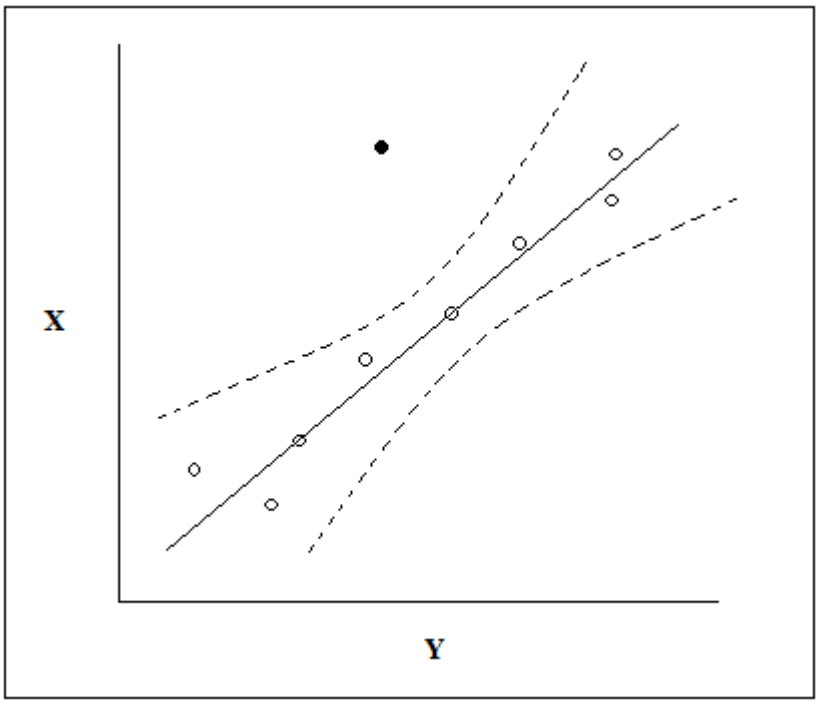

Figure 3.2: Linear Regression Method to Determine Outliers

6. Building the Polynomial GMDH model

To build the Polynomial GMDH model, firstly the data which was earlier screened were used as the input parameters. There are 8 input parameters selected. The source code were obtained from Jekabsons (2010) and were tested using MATLAB 7.10.0 (R2010a). The detailed model input and output notions were defined in **APPENDIX B**.

7. Optimizing the Polynomial GMDH Model

The codes are constantly modified in a trial and error basis in order to achieve the best performance and true trend.

8. Selecting Best Model

Models are then selected based on the optimum performance as well as the true trend analysis. True trend analysis is done by varying the crucial parameters and fixing the other parameters to see whether these parameters follow the actual behavior or trend. The behavior trend is how the parameter for example the oil compressibility changes with a change in pressure. If an increase in reservoir

pressure results in a decrease in oil compressibility. Thus it can be concluded that it is physically correct.

9. Comparing Networks Output with Actual Value Correlations

Once the optimum model is obtained, the values are then compared with the actual measured values obtained from PVT data. Different types of analysis are done to determine its accuracy such Statistical Error Analysis and Graphical Error Analysis.

10. Comparing Networks Output with Current Correlations

The predicted outputs are compared with the current existing correlations using various Statistical Error Analysis methods with the Average Absolute Percent Relative Error (AAPE) being the main criteria.

11. Reporting and Analysis

At this stage, the oil coefficient generated using Polynomial GMDH methods are then compared and analysed with the current correlations to determine its reliability and accuracy. Types of analysis include Group Error Analysis.

3.6 Modeling Process Flow

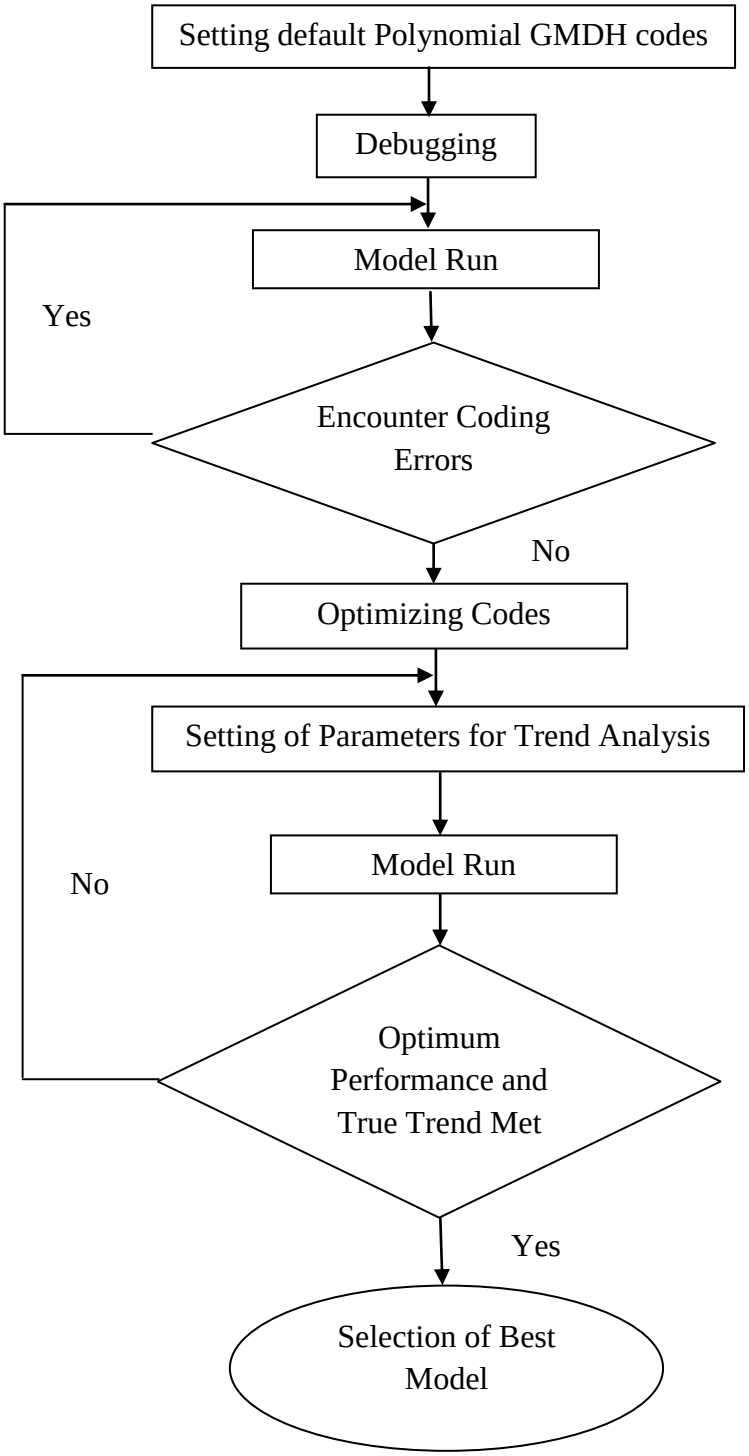

Figure 3.3: Modeling Process Flow

1. **Setting default Polynomial GMDH codes**

The Polynomial GMDH codes are inputted into the MATLAB software and modifying the codes based on the project.

2. **Debugging**

The codes are then debugged for errors which prevents the network from proceeding further.

3. **Model Run**

Model is then put to an initial run to see if it can perform error free and the graphs produced are observed for errors or strange patterns.

4. **Optimizing Codes**

The codes are optimized in a trial and error basis to obtain the optimized performance as well as obeying the true trend.

5. **Setting Of Parameters for Trend Analysis**

Parameters are set in order to produce graphs for trend analysis. Usually the parameter being investigated is allowed to vary while the remaining parameters are set to the constant mean value.

6. **Model Run**

The model is run to observe whether it reaches optimum performance and obeying true trend. Optimum performance is in terms of correlation coefficient and the graphical error analysis.

7. **Selection of Best Model**

The best models based on optimum performance are then selected for analysis purposes in the next phase of the project.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

In this chapter, firstly the trend analysis is done to ensure the model is physically correct. Then, the results from the generated model by the Polynomial GMDH method will be analysed by the statistical and graphical methods. The Polynomial GHDH method will then be compared with the 5 existing correlations which were mentioned earlier. The method of comparison would be the group error analysis. Then the reliability of this new generated model will be discussed comparing with the current correlations.

4.2 Polynomial GMDH Model Run

The MATLAB software was used to build the GMDH model. The model consists of 2 layers and 28 neurons were tried in the first layer and only 2 neurons are included at the end of first layer run. As for the second layer, one neuron was included as per default and that is the oil compressibility. During the run, 3 input parameters have shown significant impact on the final oil compressibility predictions which were Reservoir Pressure, Solution GOR and Bubble point Pressure. The selections of these inputs are done with minimal interference of the user as it was done from the mapping of values inside the network to come up with the closest actual oil compressibility values. **Figure 4.1** shows the schematic diagram of the Polynomial GMDH model.

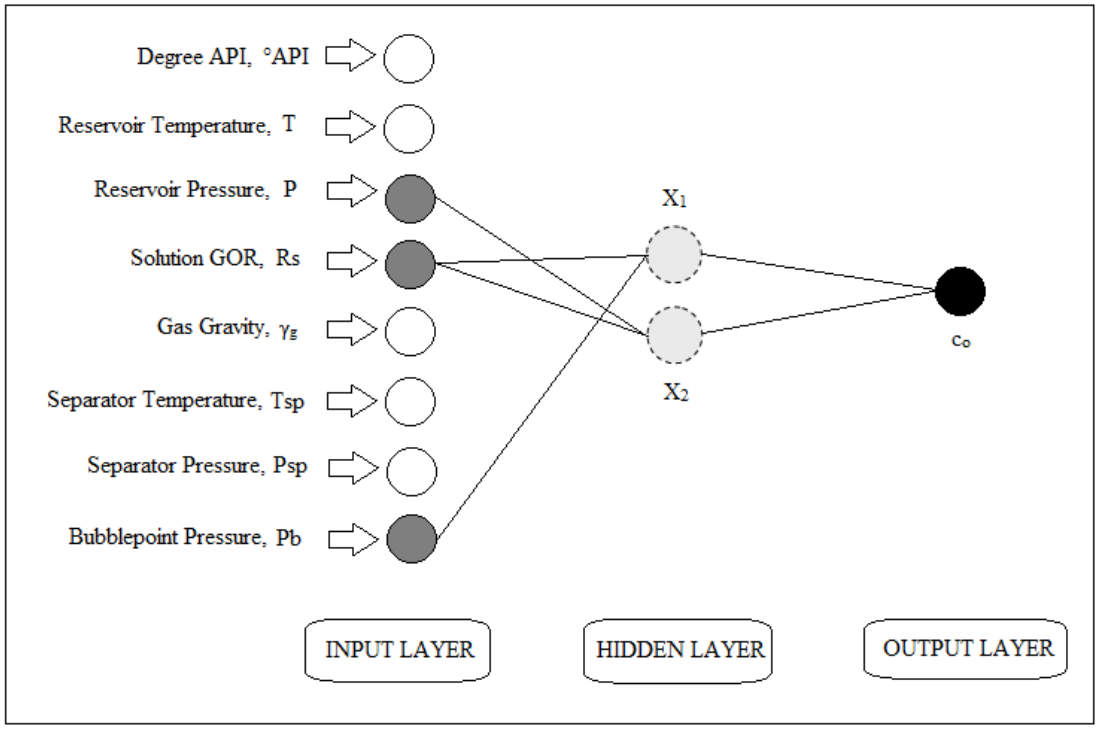

Figure 4.1: Schematic Diagram of Polynomial GMDH Model

4.3 Summary of the Model Equation

The 2 layers from the GMDH Polynomial model are as follows:-

Layer number 1

Number of neurons: 2

$$X_1 = a_1 - a_2 P_b + a_3 R_s - a_4 R_s P_b + a_5 P_b^2 - a_6 R_s^2$$
$$X_2 = a_7 + a_8 R_s + a_9 P - a_{10} R_s P + a_{11} R_s^2 + a_{12} P$$

Where,

$a_1 = 5.92e-6,$ $a_3 = 2.78e-8,$ $a_5 = 1.25e-12,$ $a_7 = 4.84e-6,$ $a_9 = 2.80e-10,$
$a_2 = 5.19e-9,$ $a_4 = 4.11e-12,$ $a_6 = 3.53e-13,$ $a_8 = 1.59e-8,$ $a_{10} = 7.22e-13,$

P= Reservoir Pressure

P_b = Bubble Point Pressure

R_s = Solution GOR

Layer number 2

Number of neurons: 1

$$c_0 = b_0 + b_1 X_2 + b_2 X_1 - b_3 X_1 X_2 + b_4 X_2^2 + b_5 X_1^2$$

Where,

$b_0 = 3.29e-7$, $b_2 = 0.244$, $b_4 = 6.6e+4$

$b_1 = 0.641$, $b_3 = 1.24e+5$ $b_5 = 6.26e+4$

c_o = oil compressibility $(psia)^{-1}$

4.4 Trend Analysis for the GMDH Polynomial Model

Trend analysis is to ensure that the model is physically correct. Thus, 2 of the crucial input parameters were run and only the parameters tested is varying while the rest of the parameters are set constant. The trend is plotted against oil compressibility to see if the behavior of the parameters follows the true trend of the actual behavior. The 2 input parameters were Reservoir Pressure and Solution GOR.

For reservoir pressure against oil compressibility in **Figure 4.2**, it shows the true trend whereby as reservoir pressure increase, the oil compressibility decreases. This shows the real relationship between pressure and oil compressibility assuming all data points of reservoir pressure is above bubble point pressure.

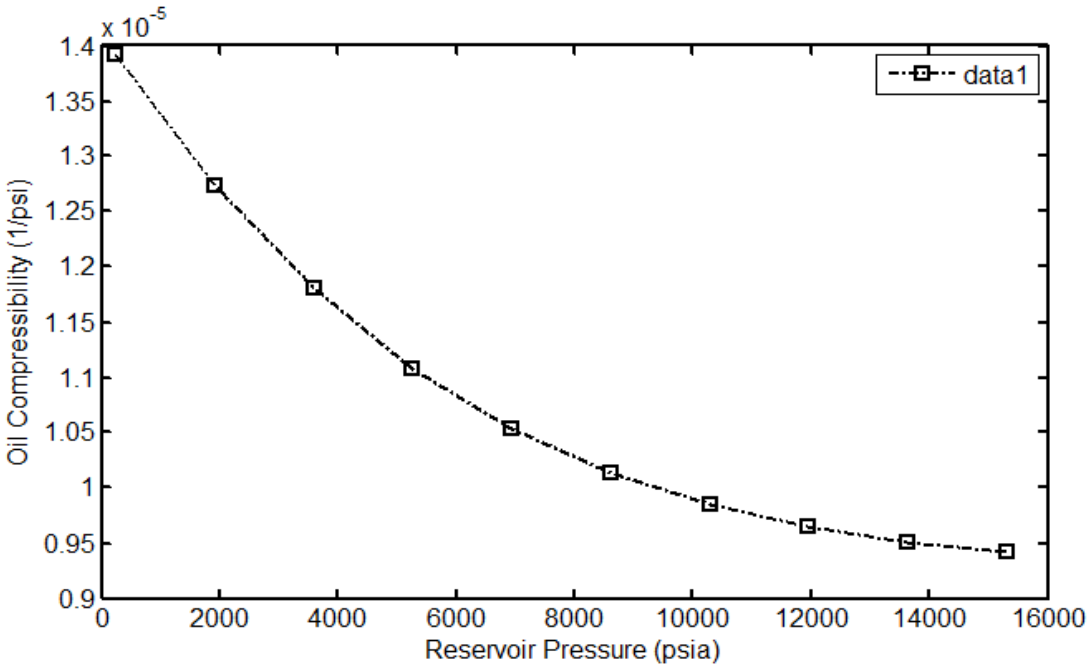

Figure 4.2: Reservoir Pressure vs Oil Compressibility

As for the trend of solution GOR against oil compressibility as shown in **Figure 4.3**, the oil compressibility is increasing with an increase in solution GOR. This behavior trend is also true as more gas is in the solution thus gives an increase in oil compressibility. However one of the correlation study done by Ahmed show a decrease in oil compressibility with increase solution GOR as shown in **Figure 4.4**.[10] Therefore, it can be safely concluded that the trend analysis for solution GOR vs Oil compressibility is also obeying the real physical behavior.

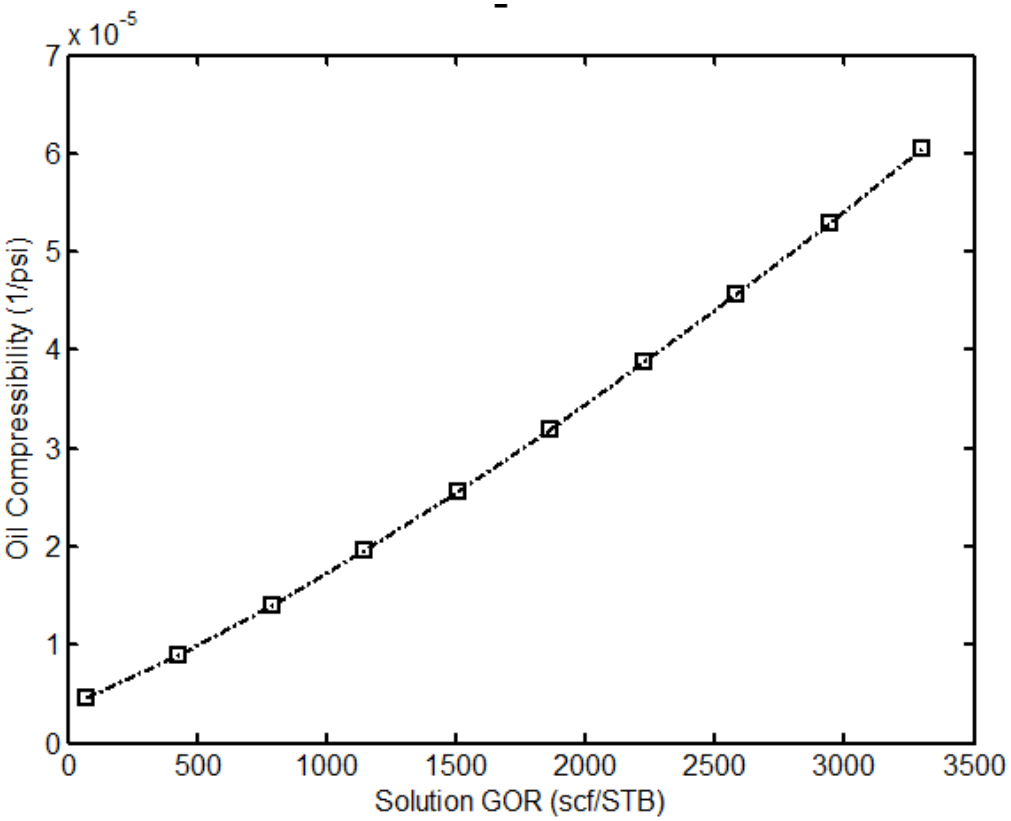

Figure 4.3: Solution GOR vs Oil Compressibility

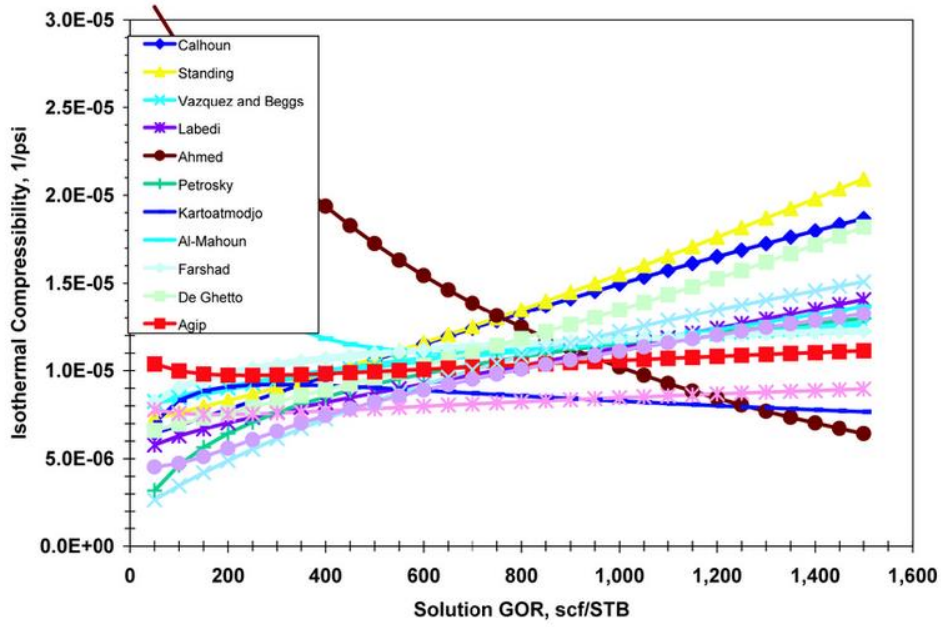

Figure 4.4: Correlation Trend by Other Authors [10]

4.5 Group Error Analysis

This analysis was done to show the reliability of the new model compared to the current existing correlations. For this analysis, the average absolute relative error was used to evaluate the accuracy of all the empirical correlations as well as the polynomial GMDH model. This is done to the 3 crucial input parameters selected earlier by the GMDH Polynomial Method. The input parameter is grouped into different ranges and is plotted with corresponding values attained by the average absolute relative error for the oil compressibility for each data set.

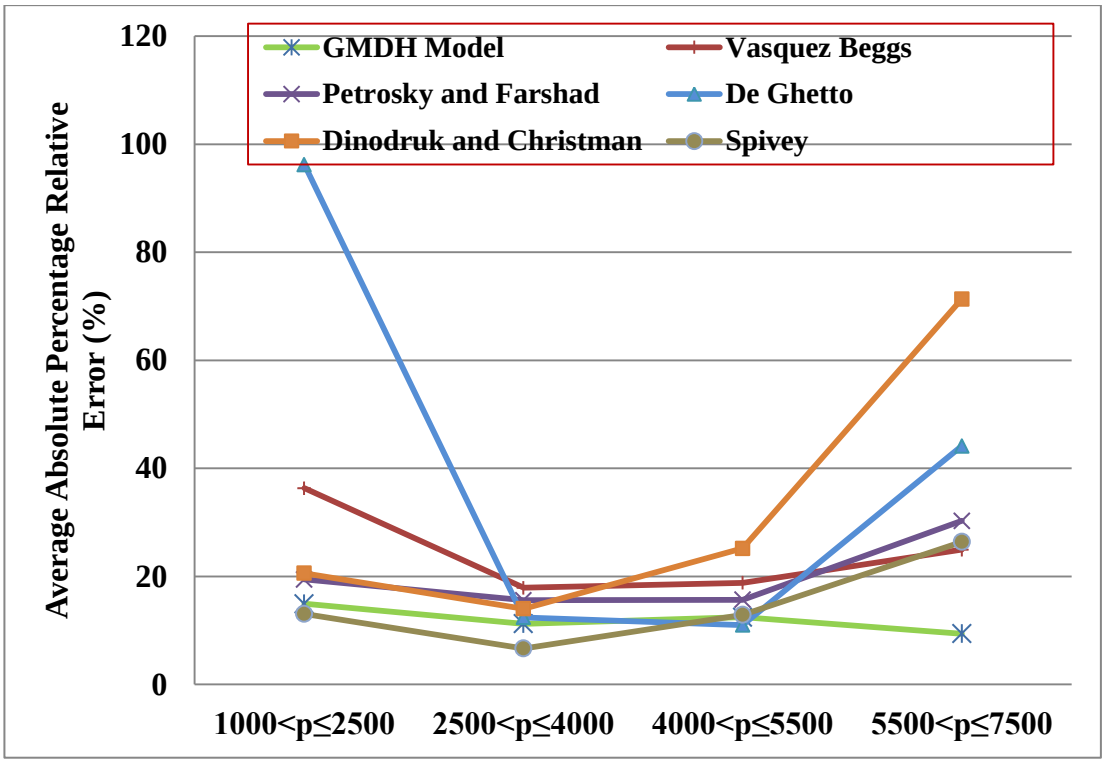

Figure 4.5: Statistical Accuracy for Oil Compressibility for Different Pressure Range

Figure 4.5 shows the AAPE for different pressure range and the GMDH Model shows a reasonable low AAPE for all pressure range. For the lowest pressure range which was from 1000 to 2500psia, the Spivey correlation model has the lowest AAPE value and the GMDH model is ranked second. For the next pressure range which was from 2500 to 4000 psia, the Spivey model was still the best with the lowest AAPE value followed by the GMDH model. However for the following pressure ranges which was from 4000 to 5500 psia, De Ghetto correlation model now has the lowest AAPE value followed by GMDH model. For the final pressure range

which is at the higher pressure range, the GMDH model now has the lowest AAPE value far outranking the other correlations with a big margin. Thus the GMDH model is much stable compared to the other correlations as the AAPE for all pressure ranges does not exceed 20%.

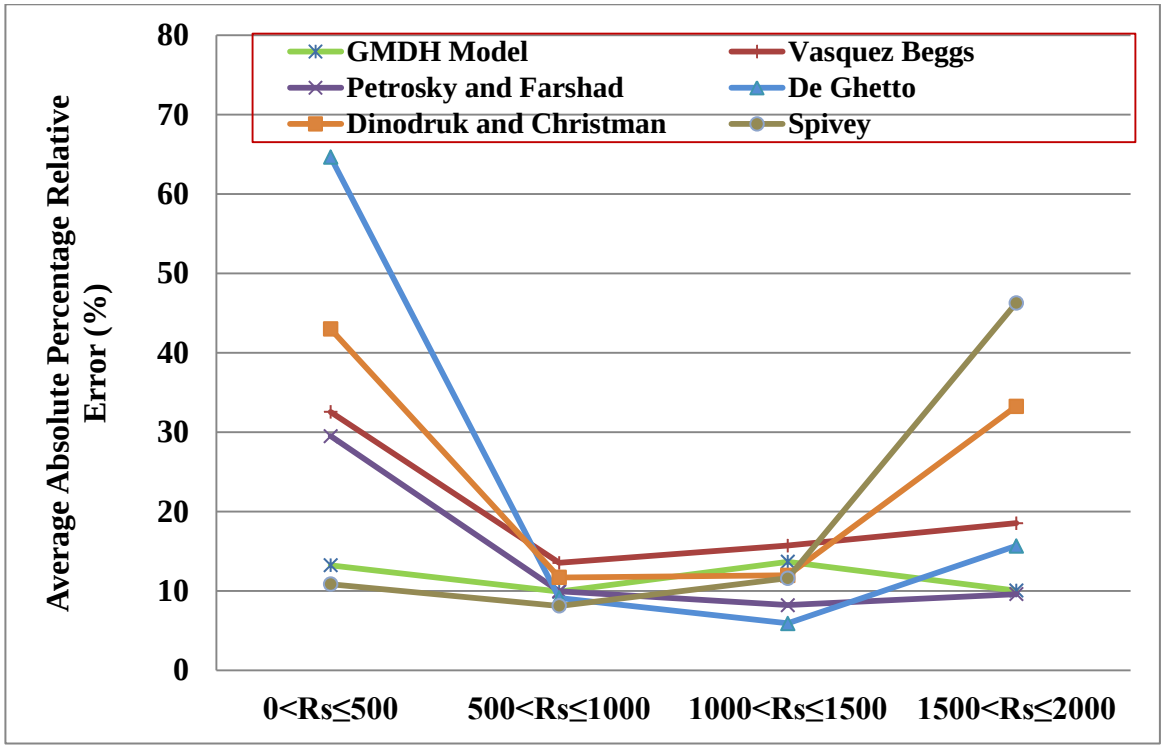

Figure 4.6: Statistical Accuracy for Oil Compressibility for Different Solution GOR Range

In **Figure 4.6**, the models are now compared with the ranges in solution GOR whereby for the lowest solution GOR range from 0 to 500 scf/STB, Spivey has the lowest AAPE followed by GMDH model coming second. For the next solution GOR range from 500 to 1000 scf/STB, Spivey model still has the lowest AAPE followed by De Ghetto and GMDH model coming third. As for solution GOR range from 4000 to 5500 scf/STB, De Ghetto correlation model has the lowest AAPE followed by Petrosky and Farshad, Spivey, Dinodruk and Christman and GMDH model coming at 5th. In terms of solution GOR ranges, although GMDH model did not achieve the lowest AAPE value in all of the ranges, it outperforms the existing correlations at the lowest and highest limits of solution GOR.

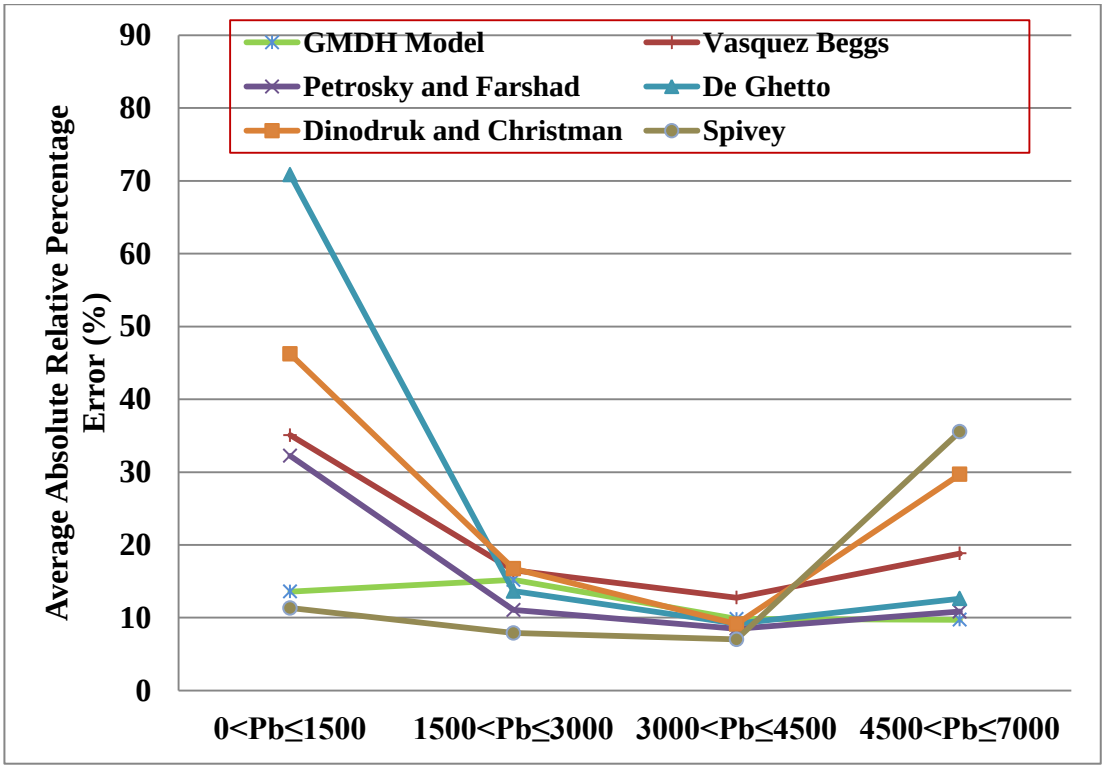

Figure 4.7: Statistical Accuracy for Oil Compressibility for Bubble Point Pressure Range

In **Figure 4.7**, the AAPE is compared to the bubble point pressure ranges and for the low bubble point range of 0 to 1500 psia, Spivey has the lowest AAPE followed by the GMDH model. For the second range of bubble point pressure from 1500 to 3000psia, Spivey still has the lowest AAPE with GMDH model coming at 4th lowest after Petrosky and Farshad and De Ghetto correlations. For the following bubble point pressure range which was from 3000 to 4500 psia, GMDH model is the 5th lowest where Spivey is having the lowest AAPE. However for the final range of bubble point pressure which was from 4500 to 7000 psia, the GMDH model obtained the lowest AAPE value. This shows that the GMDH model is also stable if bubble point pressure ranges are used as the limits for low and high bubble point pressure still displayed good AAPE values.

4.6 Statistical and Graphical Analysis

For statistical error analysis, the results are tabulated in **Table 4.1** in terms of Absolute Percent Relative Error (E_r), Average Absolute Percent Relative Error (E_a), Minimum Absolute Percent Relative Error (E_{min}), Maximum Absolute Percent Relative Error (E_{max}), Root Mean Square Error (RMSE), Correlation Coefficient (R) and Standard Deviation (SD). Graphical Error Analysis is used to visualize the performance of the Polynomial GMDH model compared to other current correlation models and these includes cross-plots, error distribution and residual analysis.

Table 4.1: Comparisons Of Statistical Error Analysis For Polynomial GMDH Model (Training, Validation and Testing)

Parameters	Data Sets		
	Training	Validation	Testing
E_a	15.45	15.81	12.08
E_r	-3.30	-5.08	-2.91
E_{Max}	91.98	66.49	31.33
E_{Min}	0.54	0.17	0.11
RMSE	21.08	21.88	14.64
R "fraction"	0.94	0.94	0.96
SD	2.27E-06	3.05E-06	1.94E-06

For the 3 data sets used, the results from the statistical analysis does not show any unique trend as all 3 data sets showed reasonable good results in terms of RMSE, correlation coefficient and standard deviation. The testing data set are independent from the training and validation data sets. Thus the values seen for the testing data set shows the performance of the GMDH model.

1. Cross-Plots Analysis

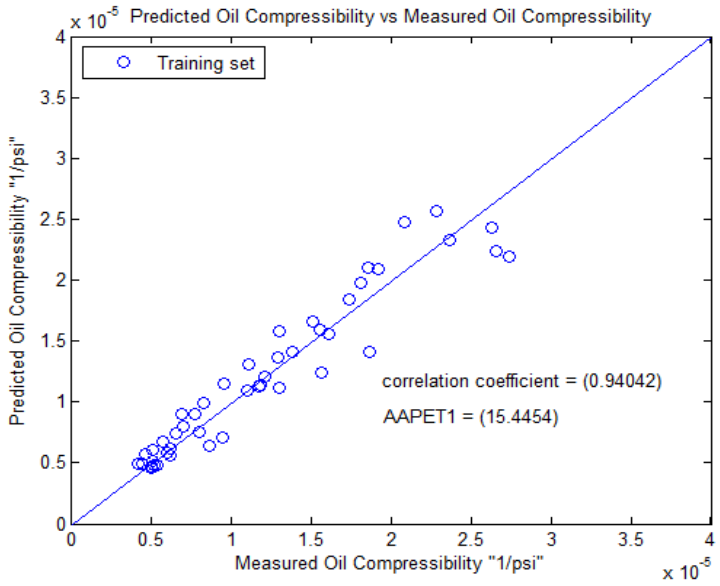

Figure 4.8: Cross plot for Predicted vs Measured Oil Compressibility for Training Data Sets

Figure 4.8 shows the cross plots for the training data sets of the GMDH Model and a correlation coefficient of 0.94042 was obtained. As can be seen from the cross plots, the values of predicted oil compressibility is very close to the ideal correlation line which is set at 45° at lower oil compressibility values. This can mean that at lower compressibility values, the GMDH model has a more accurate prediction of oil compressibility. However, the cross plots are not the only indication of accuracy as it is just a visualization as it does not give a clear insight on the actual error trends.

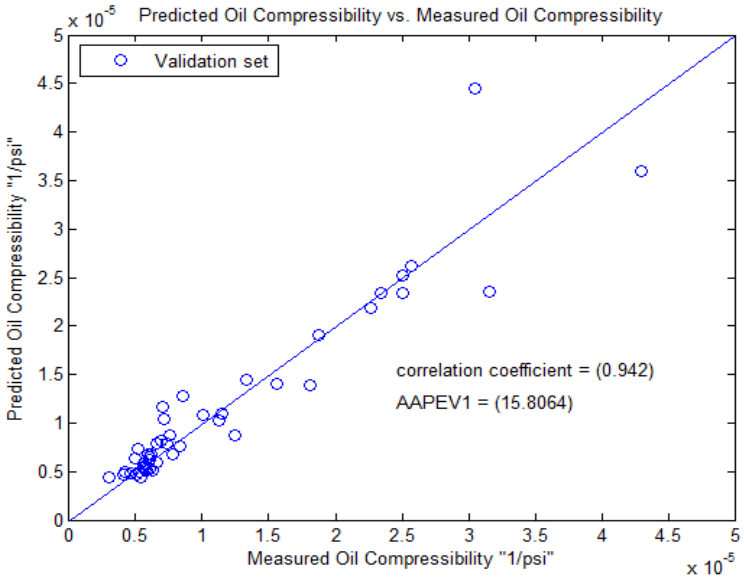

Figure 4.9: Cross plot for Predicted vs Measured Oil Compressibility for Validation Data Sets

The validation set was introduced to the training of the model to avoid overtraining.[9] In **Figure 4.9**, the cross plots are used to compare the validation data sets and a correlation coefficient of 0.942 was obtained which was slightly higher compared to the one obtained using the training data sets. As can be seen from the trend, it also shows a good representation of oil compressibility at the lower oil compressibility range and not so accurate representation at a higher oil compressibility range.

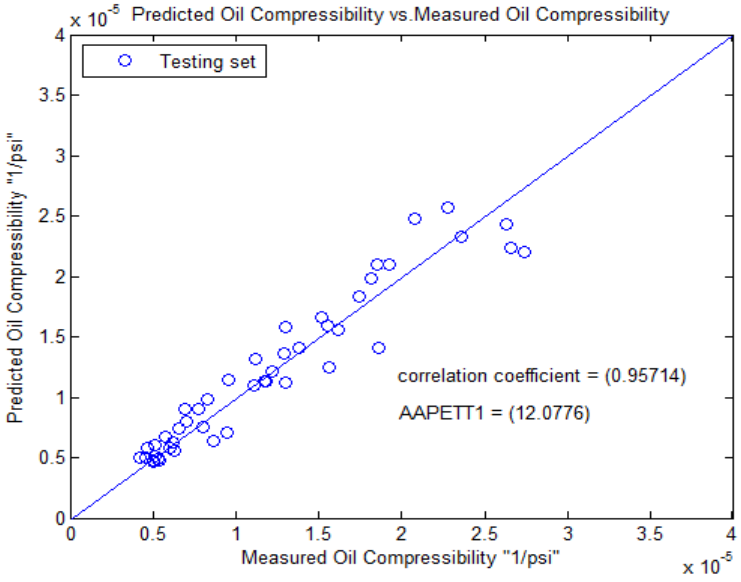

Figure 4.10: Cross plot for Predicted vs Measured Oil Compressibility for Testing Data Sets

As for the testing data sets, these data are not seen by the network during the training process, thus it can be considered independent and the values here can show the true trend of the GMDH model. From **Figure 4.10**, the measured and predicted oil compressibility shows the highest correlation coefficient compared to its earlier counterparts with a value of 0.95714. Same trend with the earlier 2 data sets can also be observed at lower oil compressibility values where it shows higher tendency towards unity compared to higher oil compressibility values.

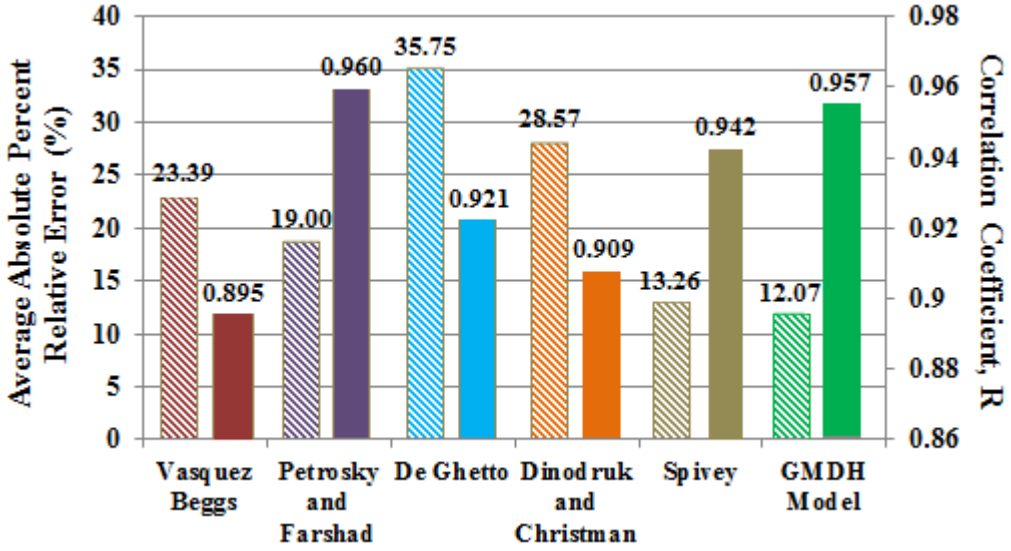

Figure 4.11: Comparison of Correlation Coefficient, Fraction (R)

Figure 4.11 shows the comparison in terms of correlation coefficient whereby and the data sets used for these comparisons come from the testing data sets. The Petrosky and Farshad correlation model gives the highest value of 0.96 followed by the GMDH model coming second at 0.957. However this comparison is not very accurate as it does not take into account the relative errors between actual and predicted values. Hence, the AAPE values are inserted as reference to give a clearer graphical representation of the correlation coefficient, R. The GMDH Model has the second highest correlation coefficient however it does have the lowest AAPE value which gives it an advantage over the Petrosky and Farshad correlation as its AAPE is 19 compared to 12.07 of the GMDH Model.

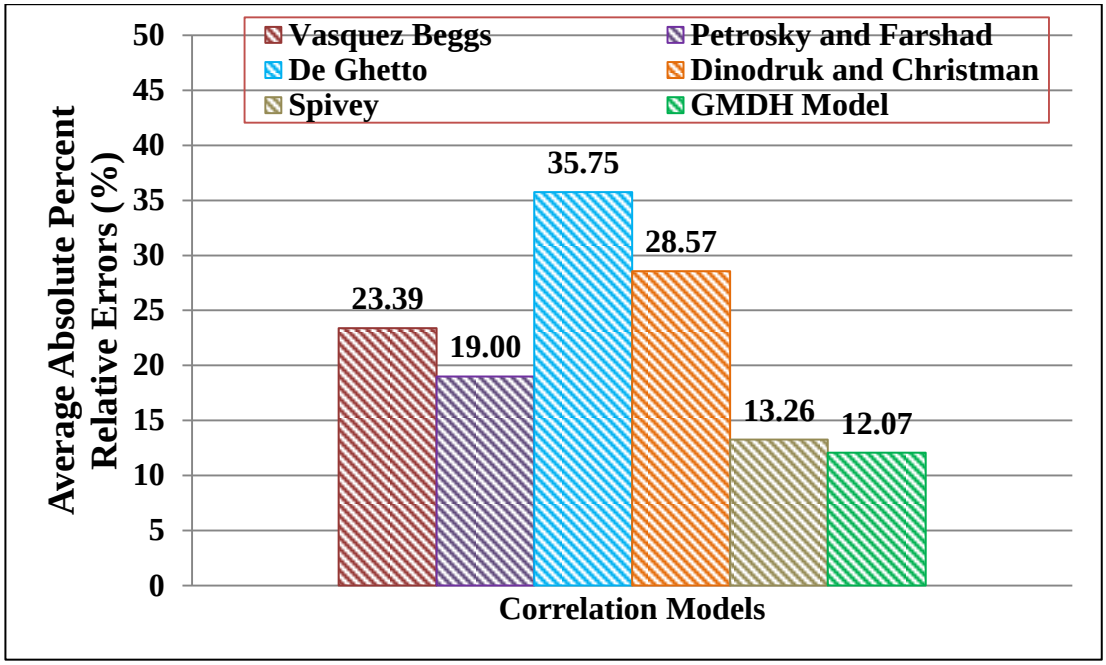

Figure 4.12: Comparison of Average Absolute Percent Relative Errors (%)

The comparison for average absolute percent relative errors (AAPE) is shown in **Figure 4.12** and again GHMD model has the lowest value of 12.07% followed by Spivey with 13.26%. This statistical analysis is the main criteria used in this project. Thus, this means that the error from actual value of the GMDH model is the smallest.

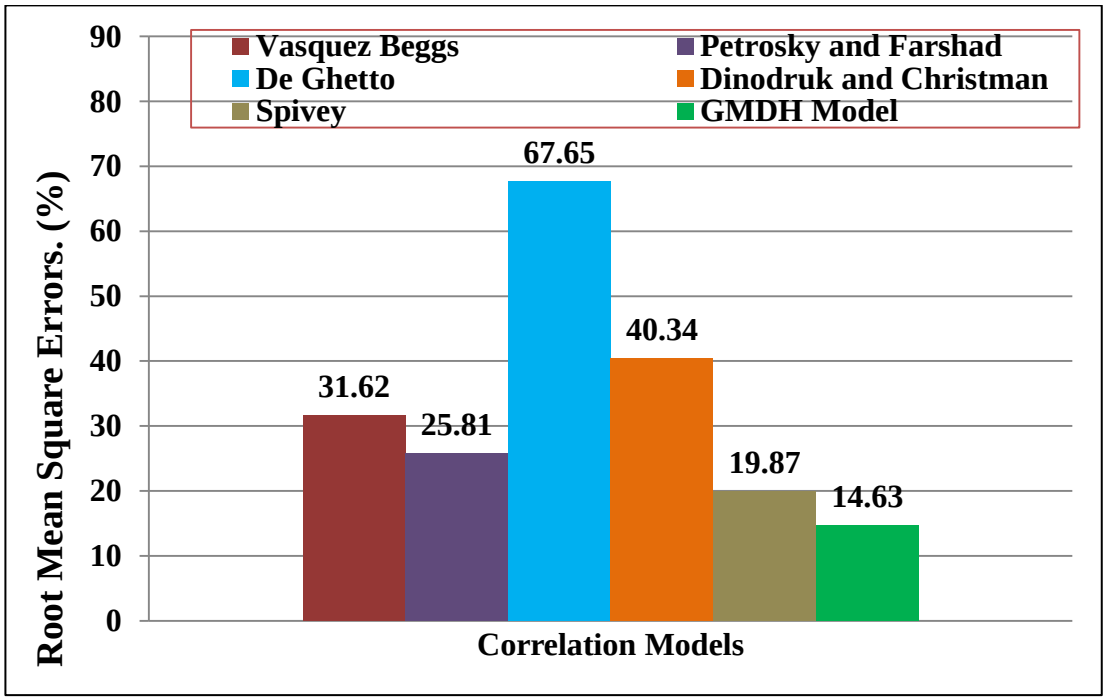

Figure 4.13: Comparison of Root Mean Square Errors

In **Figure 4.13**, a comparison is done in terms of RMSE and the GMDH model has the lowest value of 14.63%. This means that the GMDH model has the lowest data dispersion around the zero deviation compared to other correlation models. Spivey correlation comes second with an RMSE value of 19.869%.

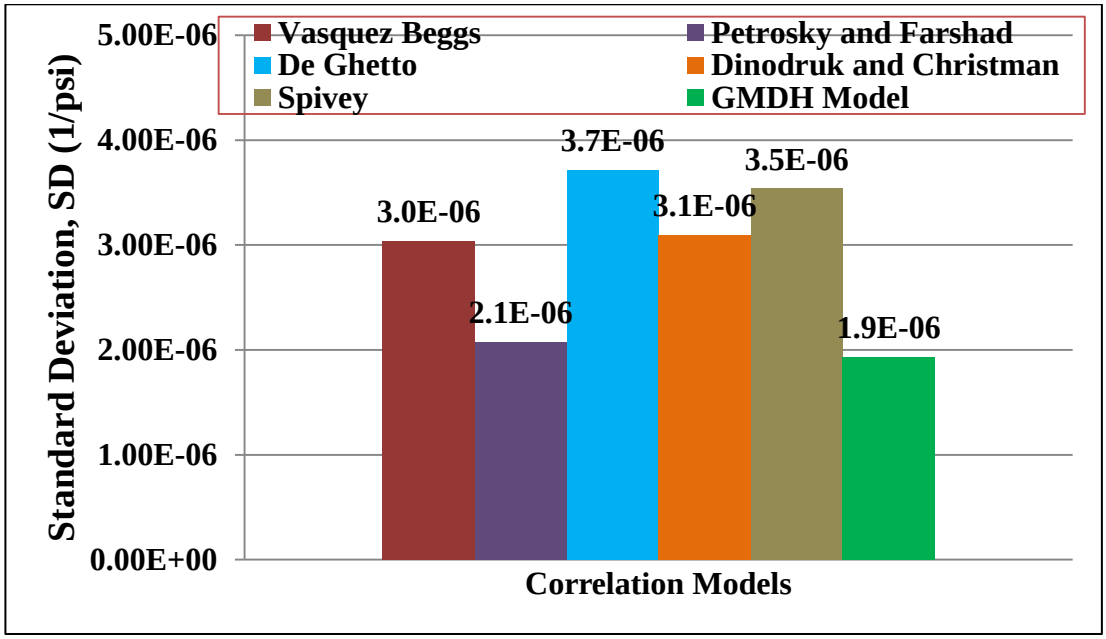

Figure 4.14: Comparison of Standard Deviation, SD (psi^{-1})

In **Figure 4.14**, a comparison in terms of Standard Deviation was done and the GMDH Model obtained the lowest value of 1.94E-06 psi^{-1} followed by Petrosky and Farshad. This means the values obtained by the GMDH model are not as dispersed compared to the other correlations. However the Spivey correlation shows quite a high standard deviation value this time with a value of 3.54E-06 psi^{-1}. Although Spivey correlations showed good results in the AAPE and RMSE, the SD of the Spivey correlation is considerably high means the values are quite dispersed.

Table 4.2: Statistical Analysis of GMDH Model against Other Correlations

Model Name	Statistical Feature						
	E_a	E_r	E_{Max}	E_{Min}	RMSE	R	SD
Vasquez Beggs	23.39	6.78	128.64	0.78	31.62	0.90	3.04E-06
Petrosky and Farshad	19.00	5.33	61.86	0.64	25.81	0.96	2.07E-06
De Ghetto	35.75	30.89	260.53	0.05	67.65	0.92	3.71E-06
Dinodruk and Christman	28.57	-23.72	132.83	1.62	40.34	0.91	3.10E-06
Spivey	13.26	-8.20	70.53	0.25	19.87	0.94	3.54E-06
GMDH Model	12.07	-2.88	31.35	0.06	14.63	0.96	1.94E-06

Table 4.2 is the summary of the statistical analysis done between GMDH model and the other correlations. It can be seen from the table that the GMDH Model gives a good correlation trend with having the lowest values in terms of E_a, E_r, E_{Max}, E_{Min}, RMSE and Standard Deviation. This means that the values predicted by the new GMDH model is consistent as it does not produce outliers or values which are too far out of range to the actual value which can be seen from the minimum and maximum average percent relative error. The values are the least dispersed which can be seen from RMSE and SD values.

2. Error Distribution Analysis

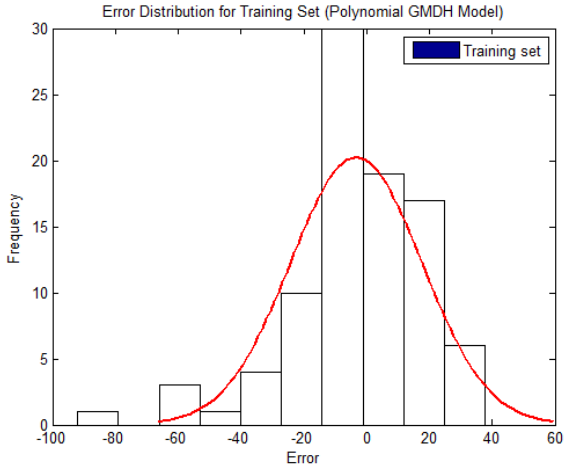

Figure 4.15: Error Distribution for Training Set (Polynomial GMDH Model)

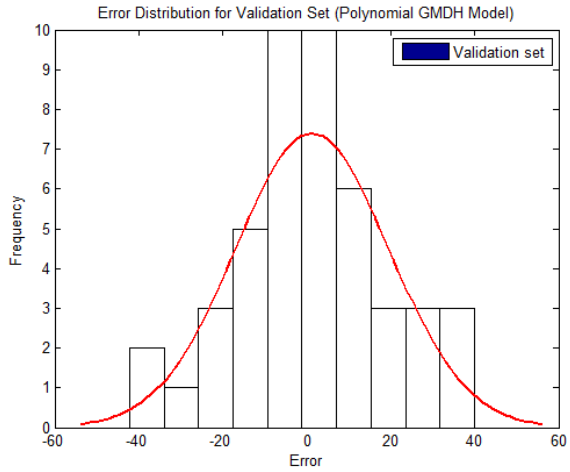

Figure 4.16: Error Distribution for Validation Set (Polynomial GMDH Model)

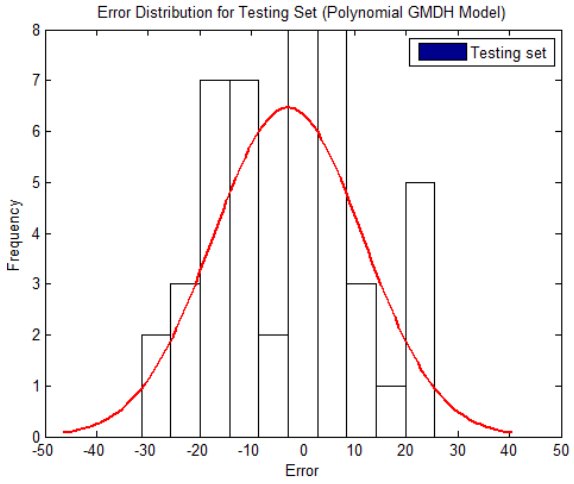

Figure 4.17: Error Distribution for Testing Set (Polynomial GMDH Model)

Figure 4.15 shows the error distribution for the training sets of the Polynomial GMDH Model. As can be seen from the figure, there is a slight shift of mean of errors to the negative side of the plot. From **Figure 4.16,** the mean for the error distribution for the validation set is slightly skewed to the positive side of the plot. However for the testing data sets as shown in **Figure 4.17**, the mean of the error distribution of about less than 5% is towards the negative side of the plot. From these 3 Figures, it can be observed that there is no trend for the error distribution to go towards the negative or positive side as the bell curve for all 3 data set peaks at zero error.

Table 4.3: Evaluating Model Performances in Terms of Average Absolute Percent Errors and Correlation Coefficient

Model Name	Statistical Feature		
	Average Absolute Percent Errors, E_a	Correlation Coefficient, R	Rating
GMDH Model	12.07	0.957	**1**
Spivey	13.26	0.942	**2**
Petrosky and Farshad	19.00	0.960	**3**
Vasquez Beggs	23.39	0.895	**4**
Dinodruk and Christman	28.57	0.909	**5**
De Ghetto	35.75	0.921	**6**

To rank the performances of each model, the model performances in terms of AAPE and Coefficient Correlation as well as RMSE and Standard Deviation are summarized in **Table 4.3** and **4.4** respectively. The reason why AAPE is compared together with correlation coefficient is due to the nature of estimating the correlation coefficient as there are cases whereby, the values are parallel to one another and it also displays a perfect correlation which is 1. Thus the purpose of the AAPE is to verify the reliability of the correlation coefficient. For the first evaluation, the GMDH model is the best by having the lowest AAPE and the second highest correlation coefficient followed by the Spivey correlation ranked at second place.

GMDH model is placed at rating 1 due to the fact that the AAPE of the GMDH model is far lower than Petrosky and Farshad which has the highest correlation coefficient. This proves that the correlation coefficient of Petrosky and Farshad does not display the actual situation.

Table 4.4: Evaluating Model Performances in Root Mean Square Errors and Standard Deviation

Model Name	Statistical Feature		
	Root Mean Square Errors	Standard Deviation	Rating
GMDH Model	14.63	1.94E-06	**1**
Spivey	19.87	3.54E-06	**2**
Petrosky and Farshad	25.81	2.07E-06	**3**
Vasquez Beggs	31.62	3.04E-06	**4**
Dinodruk and Christman	40.34	3.10E-06	**5**
De Ghetto	67.65	3.71E-06	**6**

For evaluation in terms of RMSE and Standard Deviation, the GMDH model is again the best model as it is having the lowest RMSE as well as Standard Deviation followed by Spivey coming second again. This is a measure of dispersion where RMSE is the measure of dispersion around the zero and SD is the measure of dispersion. GMDH model attained the best rating which means the values predicted by the model is not as dispersed compared to the other existing correlations.

3. Residual Analysis Error Distributions

The residual analysis error distributions are to check for model consistency. It is the difference between the predicted and measured value and is plotted with a zero line to see where the error values are more distributed to the negative or positive side. **Figure 4.18, 4.19** and **4.20** shows the error distribution for the training, validation and testing data sets of the GMDH Model respectively.

For the training data sets in **Figure 4.18**, the minimum and maximum residual errors are -6.9E-06 and 5.26E-06 respectively. As for the validation data sets in **Figure 4.19**, the minimum and maximum residual errors are -8E-06 and 1.4E-05 respectively and the testing data sets shown in **Figure 4.20** are having minimum and maximum residual errors of -5.5E-06 and 3.86E-06 respectively. The residual error shown by these 3 figures doesn't seem to show strong tendency to skew to the positive or negative sides.

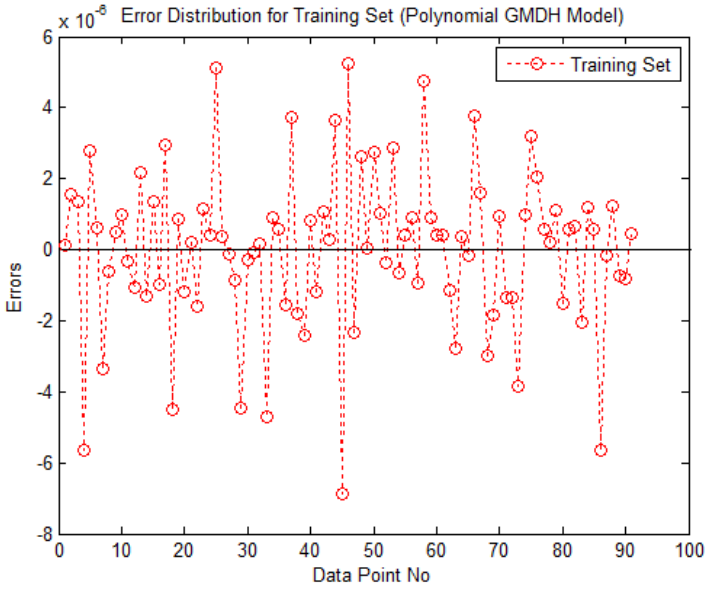

Figure 4.18: Residual Graph for Training Set

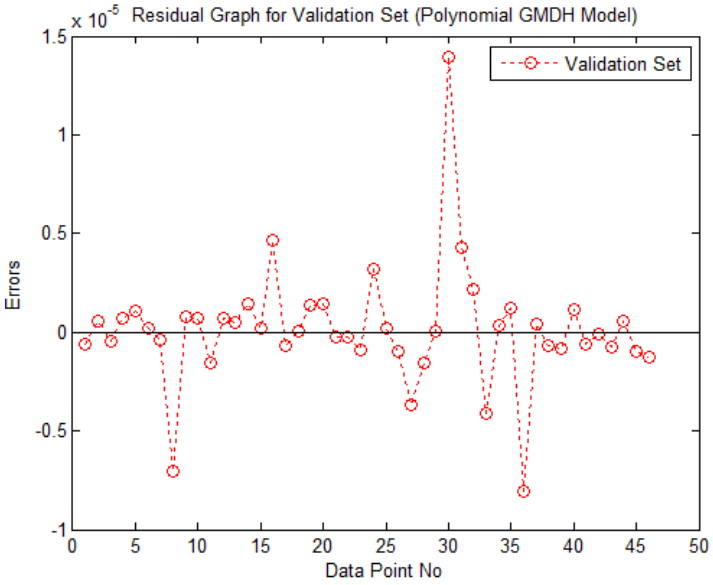

Figure 4.19: Residual Graph for Validation Set

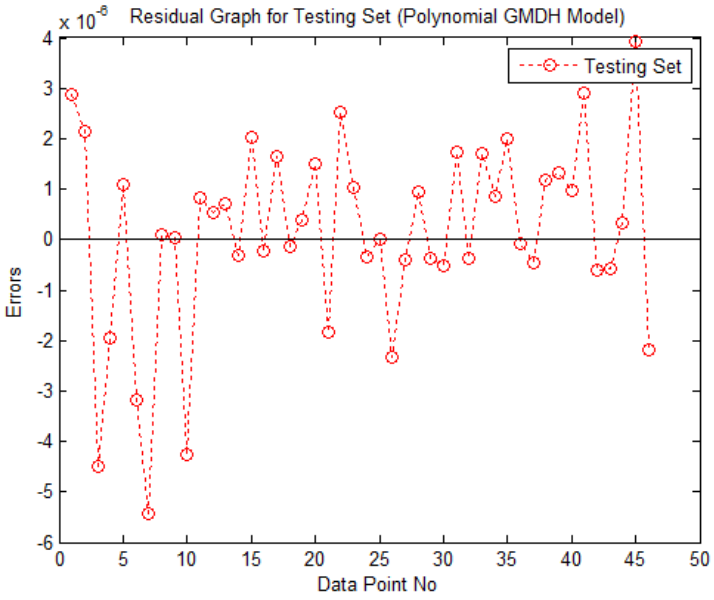

Figure 4.20: Residual Graph for Testing Set

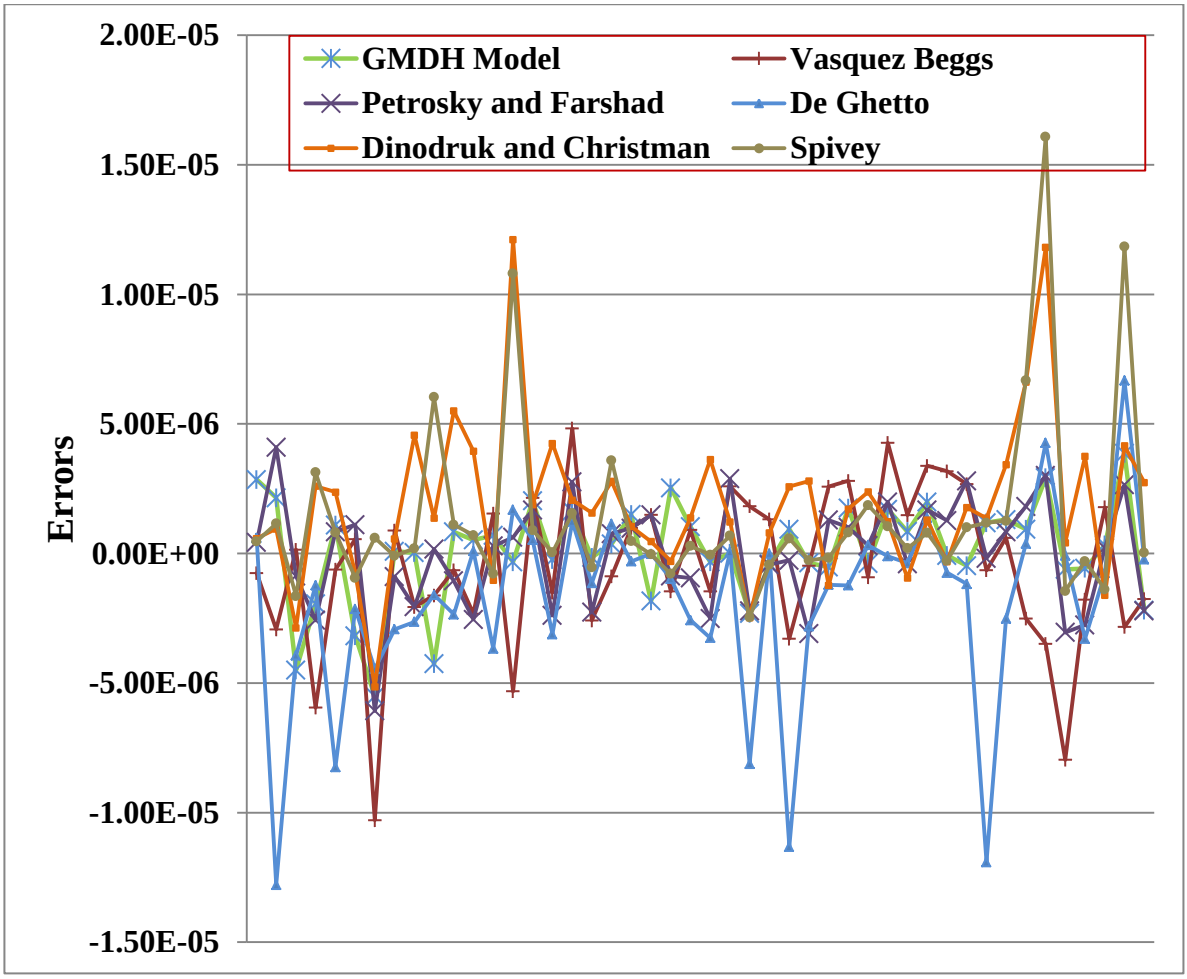

Figure 4.21: Comparison of Residual Errors of GMDH Model with Existing Correlations

Table 4.5: Residual Limits of Polynomial GMDH Model against Other Correlation Models

Model Name	Residual Limits	
	Maximum	Minimum
Vasquez Beggs	4.82E-06	-1.03E-05
Petrosky and Farshad	4.10E-06	-6.07E-06
De Ghetto	6.67E-06	-1.28E-05
Dinodruk and Christman	1.21E-05	-5.14E-06
Spivey	1.61E-05	-2.47E-06
GMDH Model	3.86E-06	-5.51E-06

Figure 4.21 shows the residual limits of Polynomial GMDH model compared with the existing correlations. **Table 4.5** summarizes the values of maximum and minimum residual limits of the GMDH model against all investigated correlation models. From the table, it can be seen that Spivey shows the highest maximum residual limit of 1.61E-05 and GMDH model has the lowest maximum residual limit. As for the minimum residual limits, De Ghetto has the highest minimum residual limit of -1.28E-05 and the GMDH Model is ranked 4th highest while Spivey is having the smallest minimum residual limit. The residual limits shows how far off the predictions of the model compared to the actual values. The larger the range, the higher the tendency for the model to have inconsistent predictions.

4.7 Testing Data Set

Table 4.6 shows the data ranges used for the all the correlations being investigated which was the testing data set. The data range is fairly wide thus making it a reasonable set of data to ensure no set of data is focused or gathered in a certain range.

Table 4.6: Data Ranges for Testing Data Set

Data Range	Temeprature (°F)	Pressure, (Psia)	Bubble Point Pressure (Psia)	Oil °API Gravity	Gas Specific Gravity	Oil Specific Gravity
Min	105.80	1038.49	171.15	6.50	0.62	0.80
Max	305.10	7411.54	6613.82	45.40	1.52	1.03

Table 4.6: Data Ranges for Testing Data Set (continued)

Data Range	Separator Gas Specific Gravity	Gas Solubility (scf/STB)	Oil Volume Formation Factor	Separator Temperature (°F)	Separator Pressure (Psia)
Min	0.61	54.13	1.06	68.00	14.50
Max	1.52	1897.08	2.10	194.00	867.34

Table 4.7 shows the types of data being used by all the correlation models being investigated and as can be seen the GMDH Model used the least amount of data types which was 3 and was able to produce the best results in terms of statistical and graphical analysis. The more data used constitutes to higher cost as more PVT data are needed. The performance of the Spivey model came second after GMDH model, but the amount of data used by the Spivey model is twice as that of the GMDH model thus giving the GMDH model a huge advantage in terms of cost.

Table 4.7: Types of Data Used by Different Correlations

Correlation	Types of Data Used										Total Data
	T	P	P_b	°API	γ_g	R_s	B_o	γ_{gsep}	T_{sep}	P_{sep}	
Vasquez Beggs	√	√	×	√	√	√	×	×	√	√	7
Petrosky & Farshad	√	√	×	√	√	√	×	×	×	×	5
De Ghetto	√	√	√	√	√	√	√	×	√	√	9
Dinodruk & Christman	√	×	×	√	√	√	×	×	×	×	4
Spivey	√	√	√	√	×	√	×	√	×	×	6
GMDH Model	×	√	√	×	×	√	×	×	×	×	**3**

Where,

T = Temperature,

P = Pressure,

P_b = Bubble Point Pressure,

°API = Oil API Gravity,

γ_g = Gas Specific Gravity,

R_s = Gas Solubility,

B_o = Oil Volume Formation Factor,

γ_{gsep} = Separator Gas Specific Gravity

T_{sep} = Separator Temperature

P_{sep} = Separator Pressure

4.8 Sensitivity Analysis

The sensitivity analysis is done to determine which input parameter has the largest impact on the GMDH Model. For this purpose, one parameter is change at a time by a variation of (+, -) 25%, 50%, 75% and 100% while the other input parameters are set constant as can be seen from **Figure 4.22**. The percentage change in oil compressibility of the GMDH Model is then recorded and the results are summarized in **Table 4.8**.

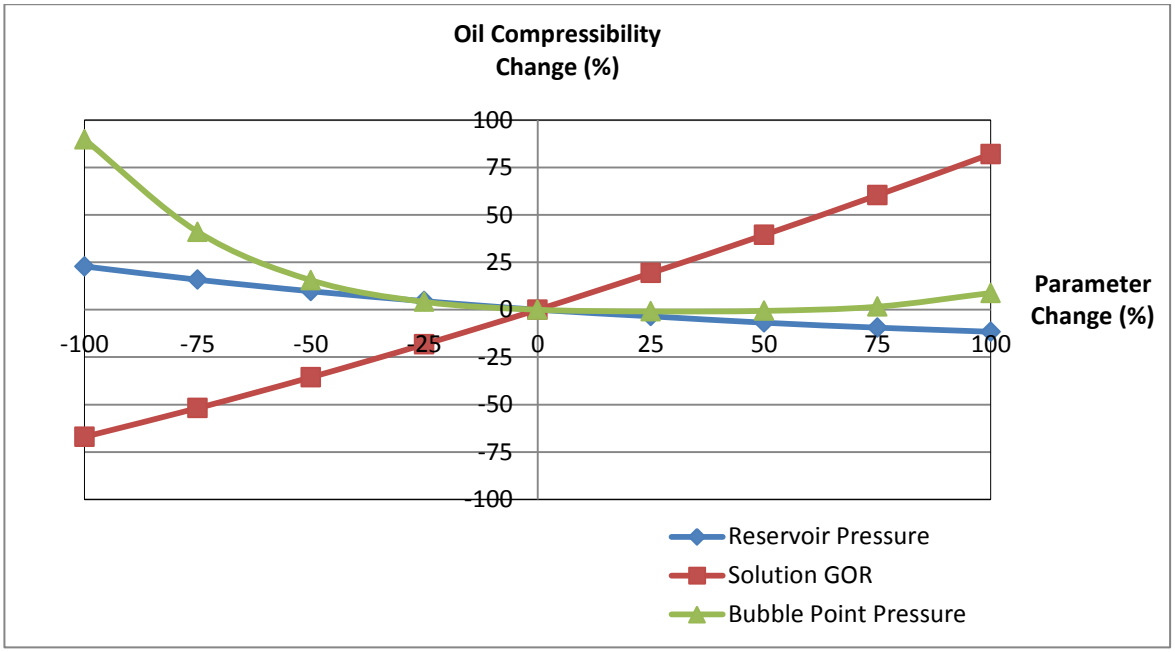

Figure 4.22: Sensitivity Analysis of the 3 Input Parameters

Table 4.8: Summary of Sensitivity Analysis

Parameters	Parameters Change (%)								
	-100	-75	-50	-25	0	25	50	75	100
	Oil Compressibility Change (%)								
Reservoir Pressure	22.85	15.81	9.74	4.52	0	-3.48	-6.83	-9.45	-11.61
Solution GOR	-67.02	-51.86	-35.58	-18.25	0	19.35	39.46	60.39	82.09
Bubble Point Pressure	89.85	41.05	15.54	4.13	0	-0.89	-0.60	1.61	8.84

From **Table 4.8**, a parameter change of solution GOR of 100% gives an oil compressibility change of 82.09% compared to that of reservoir pressure with a value of -11.61 and bubble point pressure which is 8.84. Thus, the solution GOR has the largest impact on the GMDH model where the reservoir pressure is having the second largest impact and lastly the bubble point pressure. However due to the nature of the bubble point pressure trend, the oil compressibility change tend to increase drastically towards the lower end of bubble point pressure range making it having the largest impact at extreme low values.

4.9 Limitations of GMDH Model

The limitations of this generated model may be due to the nature of the data value ranges as the model is depending on the parameters fed to it to determine the oil compressibility correlation. Therefore caution must be practiced if values out of the range of the data used in the Polynomial GMDH model generation are used.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The conclusions that can be drawn from this study are:-

1. The Proposed Polynomial GMDH model outperforms the other investigated correlation models for estimating the isothermal oil compressibility above bubble point pressure with an outstanding average absolute percent relative error of 12.07%.
2. Statistical analysis shows that the Polynomial GMDH Model achieved the lowest average percent relative error, average absolute percent relative error, root mean square error and standard division.
3. Superiority of the Polynomial GMDH model can be seen in group error analysis when it is compared with other correlations for different ranges of pressure, solution GOR and bubble point pressure with outstanding performances.
4. The GMDH Model can be applied confidently within the trained data ranges.
5. The GMDH Model used the least amount of data but achieved the highest performance in isothermal oil compressibility prediction.
6. Reservoir Pressure, Solution GOR and Bubble Point Pressure were found to be the parameters that were strongly influencing the prediction of isothermal oil compressibility.

5.2 Recommendations

1. A wider range of data with higher and lower limits can be collected from different fields to construct a more robust model using the Polynomial GMDH technique.
2. The Polynomial GMDH model should be incorporated into any commercial oil compressibility predicting software to serve as an easy programmable tool.
3. A double verification of the existing correlation models should be conducted using smart simulators.

REFERENCES

1. Olaoluwa O. Adepoju (2007). *Coefficient of Isothermal Oil Compressibility*. United Kingdom: VDM Verlag Dr. Muller. p1-78.
2. William D. McCain, JR (1990). *The Properties of Petroleum Fluids*. 2nd ed. USA: PennWell Publishing Company. p231-250.
3. J.P. Spivey, SPE, Phoenix Reservoir Engineering, and P.P. Valko, SPE, and W.D. McCain, SPE, Texas A&M U. (2005). SPE 96415. *Applications of the Coefficient of Isothermal Compressibility to Various Reservoir Situations With New Correlations for Each Situation*.
4. Ahmed, Tarek (2008). *Equations of State and PVT Analysis - Applications for Improved Reservoir Modeling*. -: Gulf Publishing Company.
5. *Oil Correlations.* Available: http://www.fekete.com/SAN/WebHelp/FeketeHarmony/Harmony_WebHelp/Content/HTML_Files/Reference_Material/Calculations_and_Correlations/Oil_Correlations.htm. Last accessed 31st Mar 2014.
6. *What is MATLAB.* Available: cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm. Last accessed 24th April 2014.
7. Ivahnenko AG. Polynomial theory of complex systems. IEEE Trans Systems Man Cybernet 1971;SMC-1:364–78.
8. Sung-Kwun Oh, Witold Pedrycz, Byoung-Jun Park . (2011). Polynomial neural networks architecture. *Polynomial neural networks architecture: analysis and design.* .p704-724.
9. M.A.Ayoub (2011). *Development and Testing of Universal Pressure Drop Models in Pipelines using Abductive and Artificial Neural Networks*. Malaysia: UTP. p40-50.
10. SPE. (2013).*PEH: Oil System Correlations.* Available: http://petrowiki.org/PEH%3AOil_System_Correlations#Solution_GOR. Last accessed 18th May 2014.
11. Giambattista De Ghetto, Francesco Paone, Marco Villa. (1994). Reliability Analysis on PVT Correlations. *SPE28904*. p385-387.

APPENDIX A

MAIN DATA SET (source taken from SPE28904 [10])

Oil Compressibility, c_o (1/psia)	°API	Temperature, Tr (°F)	Pressure, Pr (psia)	Solution GOR, Rs (scf/STB)	Total surface gas gravity, γ_g	Separator Temperature, Tsp (°F)	Separator Pressure, Psp (psia)	Bubble Point Pressure, Pb (psia)
2.00E-05	41	277.2	6528.25	1398.8	0.826	152.6	440.92	5106.86
5.67E-06	10.5	152.6	2916.75	260	0.815	158	156.64	2076.97
5.02E-06	7.9	165.2	5518.77	250.5	0.768	129.2	369.85	2902.25
2.71E-05	37.9	228.2	4345.4	1357.94	0.855	86	227.71	4267.08
5.07E-06	6.3	165.2	5391.14	323.62	0.675	129.2	227.71	4021.96
8.23E-06	21.2	183.2	3721.73	404.01	1.062	100.4	725.2	2432.32
1.08E-05	28	114.8	1166.12	225.74	0.928	78.8	369.85	1166.12
2.09E-05	44.5	289.4	6433.97	1404.3	0.795	194	583.06	5170.68
4.48E-06	9.6	217.4	4895.1	108.54	1.129	133.5	58.02	967.42
2.12E-05	41	287.6	6747.26	1575.35	0.763	122	440.92	5675.42
5.07E-06	14.6	205.9	3684.02	41.92	1.178	167	85.57	337.94
1.10E-05	33	219.2	3456.3	477.24	1.051	86	440.92	2483.08
1.45E-05	37	186.8	4879.15	1021.05	0.902	86	156.64	2809.42
1.08E-05	43	154.4	1601.24	356.21	1.292	80.6	156.64	1045.74
5.31E-06	19.3	154.4	1877.54	175.44	1.406	80.6	71.07	796.27
1.65E-05	35.6	190.4	2944.31	852.05	0.919	96.8	242.22	2774.62
1.54E-05	41.7	167	3032.79	1046.87	0.895	86.5	85.57	2944.31
1.31E-05	56.8	140.7	1170.47	300.91	0.649	71.6	156.64	1170.47
1.32E-05	41.5	159.8	3420.04	769.83	0.872	69.8	85.57	2603.47
1.26E-05	27	271	3713.02	595.05	1.28	149.9	725.2	3627.45
4.78E-06	21.3	179.6	6272.98	100.93	1.035	59	49.31	654.13
2.32E-05	41.5	302	7147.57	1555.36	0.818	116.6	440.92	5589.84
1.50E-05	42.5	167	3663.71	931.61	0.893	78.4	298.78	2944.31
9.51E-06	34.5	210.2	5293.96	516.83	1.04	95	156.64	2005.9
2.61E-05	53	237.2	5956.79	2191.33	0.883	86	156.64	3826.16
1.09E-05	38.8	183.2	2271.33	524.1	1.46	86	298.78	1863.76
4.52E-06	7.3	221.7	4732.66	18.82	1.134	131	42.06	249.47
5.85E-06	21.2	190.4	1209.63	27.76	1.421	86	85.57	213.21
2.34E-05	38.5	224.6	3684.02	1134.14	0.923	76.5	156.64	3527.37
4.95E-06	14.9	207.9	3727.53	25.04	1.307	167	50.76	208.86
1.08E-05	38.8	152.6	2423.62	475.41	1.109	94.1	298.78	1493.91
1.65E-05	37.2	269.6	5697.17	1081.29	0.866	158	440.92	4243.87
3.18E-05	43.6	197.6	3044.39	1593.4	1.016	71.6	440.92	2760.11
1.16E-05	43	222.8	5589.84	708.53	1.007	71.6	171.15	2120.48
9.14E-06	37.2	180.5	2588.96	432.6	0.797	87.8	440.92	2517.89
1.51E-05	38.5	204.8	3215.54	725.86	0.983	96.8	242.22	2496.14
8.41E-06	36.2	140	5788.55	665.17	1.057	89.6	440.92	1891.32

1.37E-05	27.8	260.6	3691.27	629.86	1.19	73.4	213.21	3507.07
1.48E-05	39	271.4	6187.41	763.39	1.025	125.6	440.92	3243.09
8.84E-06	21	185.2	4873.34	500.23	0.965	68	156.64	2369.95
1.12E-05	39	194	2233.62	436.43	1.464	86	440.92	1721.62
1.05E-05	38	171.5	3114.01	583.01	0.936	104	440.92	2687.59
6.53E-06	39.7	325.4	14466.29	387.52	0.939	86	583.06	2219.11
6.58E-06	42.4	341.6	15304.62	690.16	1.045	158	156.64	2578.81
2.71E-05	42	165.7	3480.96	1206.98	0.857	86	156.64	3362.03
5.72E-06	25	262.4	3699.97	564.63	1.28	194	511.99	3100.96
8.18E-06	35.7	212	3562.18	136.02	1.095	113	156.64	661.38
1.63E-05	42.2	167	3201.03	1099.33	0.953	89.4	440.92	3201.03
7.03E-06	40	334.4	14863.7	410.79	1.095	158	298.78	2133.54
2.02E-05	40	303.1	6336.8	1585.4	0.752	158.9	227.71	5959.69
1.08E-05	27.9	212	5533.28	686.33	0.934	86	156.64	2645.53
8.00E-06	40	158	3079.2	217.47	1.349	102.2	87.02	511.99
5.01E-06	11.4	153.1	2858.74	305.8	0.776	158	156.64	2622.32
5.12E-06	8	215.6	4494.79	51.13	1.415	131	42.06	619.32
4.41E-06	8.2	215.6	4851.59	84.06	1.334	141.8	71.07	725.2
1.47E-05	42.5	167	3612.95	884.58	0.995	97.5	440.92	2631.03
7.24E-06	36	118.4	384.36	105.04	1.137	91.4	29.01	384.36
1.41E-05	42.5	167	3612.95	1113.66	0.991	98.6	440.92	2944.31
1.57E-05	41.2	159.8	2760.11	913.18	0.993	75.2	440.92	2760.11
4.57E-06	8.3	212	4883.5	89.27	1.47	122	71.07	639.63
7.38E-06	31	138.2	1501.16	259.22	1.094	122	58.02	1282.15
6.94E-06	36.6	118.4	351	72.12	1.077	86	85.57	327.79
1.27E-05	25.7	271.4	3698.52	484.29	1.213	122	85.57	2831.18
5.85E-06	28.6	111.7	1293.76	131.58	0.93	77	298.78	654.13
2.49E-05	37.5	303.1	6301.99	550.92	0.764	158.9	227.71	6272.98
1.27E-05	42.5	167	3567.98	942.99	0.885	92.3	227.71	2660.03
5.10E-06	6	147.9	3428.75	231.46	0.696	131	58.02	2503.39
7.93E-06	31.6	242.6	3860.96	69.23	1.072	77	29.01	503.29
1.45E-05	34.1	237.9	4467.23	704.54	0.804	122	511.99	4039.36
6.53E-06	40	334.4	14913.01	444.49	1.136	179.6	342.29	2140.79
1.41E-05	38.4	224.6	4104.63	711.2	0.896	86	227.71	2924.01
8.34E-06	38	257.7	13242.15	363.7	1.113	129.9	227.71	1670.86
2.58E-05	37.2	300.9	6248.32	1396.86	0.749	149	440.92	6088.78
7.05E-06	11.2	154.8	2850.04	316.51	0.812	158	156.64	2546.9
1.98E-05	41.5	159.8	3655.01	1405.19	1.027	78.8	725.2	3655.01
1.33E-05	40.8	186.8	4739.91	912.73	0.972	95	156.64	2657.13
7.65E-06	30.7	141.4	1419.94	289.14	0.949	122	129.09	1419.94
9.14E-06	46.9	90.5	1644.75	365.76	0.959	77	227.71	1380.78
2.24E-05	41.1	296.6	6898.1	1717.76	0.82	176	440.92	5760.99
5.96E-06	14	183.2	2552.7	40.97	1.295	158	156.64	1180.63

4.61E-06	8	210.2	4708	103.1	1.491	177.8	85.57	658.63
5.58E-06	19.8	150.8	1749.47	147.96	1.256	69.8	56.57	839.78
1.23E-05	50.9	183.9	2658.58	367.48	1.408	95	85.57	661.38
1.25E-05	28.6	258.8	3726.08	763.66	1.249	174.2	440.92	3726.08
5.96E-06	19.2	165.2	1792.26	166.33	1.402	69.8	71.07	796.27
2.65E-05	41.5	226.4	4267.08	1306.64	0.977	86	440.92	3684.02
4.72E-06	19.5	167	4238.07	25.37	1.105	75.2	58.02	256.72
4.88E-06	11	167	5739.23	234.18	0.735	129.2	227.71	2588.96
6.47E-06	29.7	134.6	1927.58	89.22	1.156	83.3	156.64	384.36
6.27E-06	36.6	116.6	478.63	56.96	1.157	77	227.71	369.85
6.58E-06	19	217.4	6557.26	330.12	0.914	86	114.58	2319.19
1.15E-05	41	180.5	3369.28	529.15	1.047	77	440.92	1834.76
6.18E-06	19.7	170.6	1877.54	186.54	1.336	80.6	71.07	967.42
5.76E-06	24.8	178.2	4281.58	116.26	0.932	104	56.57	796.27
1.01E-05	39.4	181.4	3370.73	533.87	1.057	88	440.92	2062.47
1.33E-05	33	211.1	5006.78	859.77	0.914	86	156.64	2713.7
6.05E-06	23.1	112.3	1315.51	141.02	0.83	77	298.78	796.27
5.02E-06	15.1	207.7	3727.53	25.21	1.344	167	85.57	227.71
4.30E-05	45.5	231.4	4793.57	2323.75	0.968	68	227.71	4082.88
5.96E-06	19.4	172.4	1649.98	177.83	1.411	69.8	71.07	825.28
4.20E-06	8.6	217.4	4996.63	86.55	1.479	128.8	71.07	626.57
1.56E-05	37.5	271.4	5482.51	866.21	0.919	125.6	440.92	3652.11
4.04E-06	18.8	244.4	7411.54	111.76	1.206	100.4	156.64	999.33
7.38E-06	25.2	198.5	4381.66	323.68	1.2	100.4	265.42	1366.28
3.02E-06	12.8	215.6	4519.45	17.21	1.323	176	85.57	227.71
4.62E-06	24.5	129.2	1520.02	8.61	1.53	98.6	14.5	107.33
7.03E-06	11	152.6	2916.75	586.67	1.253	158	56.57	2916.75
5.86E-06	24.5	162.5	1437.35	36.75	1.38	86	85.57	210.31
2.51E-05	40.1	302	5888.62	1760.56	0.924	122	440.92	5760.99
6.86E-06	10.9	154.2	2893.55	331.34	0.81	158	156.64	2802.17
4.92E-06	7.5	153.5	3563.63	208.7	0.756	131	58.02	2082.77
5.65E-06	23.7	176	4216.31	120.09	0.864	107.6	42.06	768.71
5.27E-06	19.6	231.8	6927.11	140.52	1.092	71.6	227.71	1209.63
1.12E-05	37	222.8	4026.31	521.27	1.011	86	440.92	2375.76
7.17E-06	10	154.8	2850.04	486.9	1.236	158	156.64	2665.84
5.55E-06	23.8	80.6	242.22	58.02	1.197	140	43.51	200.16
7.74E-06	47	140	2517.89	121.81	1.789	71.6	56.57	147.94
1.24E-05	22.3	276.8	3740.58	396.41	1.218	131.9	440.92	2674.54
2.50E-05	39.6	292.1	6899.55	1700.94	0.788	156.2	868.79	5868.32
2.34E-05	44	277.9	6521	1678.62	0.847	194	867.34	5705.87
3.05E-05	51	226.8	5974.2	2987.14	0.881	86	440.92	4210.51
8.54E-06	27.8	134.6	3826.16	702.32	0.792	71.6	185.65	3826.16
5.18E-06	12.4	152.6	2916.75	269.99	0.714	62.6	42.06	2432.32

1.81E-05	31	260.1	3713.02	781.99	1.001	113	440.92	3713.02
1.87E-05	33.8	192.7	5395.49	1256.95	0.859	86	227.71	4326.54
6.63E-06	19.5	178.7	5305.56	332.61	1.169	122	156.64	1322.76
3.16E-05	41	266	4992.28	1559.47	0.663	107.6	227.71	4992.28
6.05E-06	19.8	163.4	1806.47	167.89	1.333	75.6	71.07	896.35
8.27E-06	29.6	163.4	1459.1	245.28	0.997	122	129.09	1295.21
2.27E-05	37.8	231.8	5349.08	1467.2	0.861	104	440.92	4737.01
7.58E-06	17.6	194	4873.34	429.16	0.934	68	156.64	2236.52
6.54E-06	35.1	154.4	2716.6	120.7	1.27	100.4	298.78	526.5
5.64E-06	29.8	128.1	1544.68	71.18	1.257	89.6	87.02	321.99
5.98E-06	16.5	188.1	3328.67	97.32	1.188	86	227.71	697.64
2.57E-05	49.2	172.4	3769.59	1617.27	0.928	64.4	114.58	3161.87
5.34E-06	15.4	203	3665.16	21.49	1.276	134.6	43.51	355.35
6.29E-06	34.6	226.4	5135.87	108.76	0.794	122	156.64	839.78
1.30E-05	31.7	192.7	5405.64	1006.56	0.884	86	230.61	3862.42
6.86E-06	16.8	140	1153.07	320.34	1.517	122	14.5	1074.75
1.86E-05	37	221	3328.67	773.38	1.045	96.8	440.92	2858.74
2.63E-05	41.5	235.4	5433.2	1657.52	0.901	77	440.92	4850.14
5.69E-06	19	163.4	1806.47	188.82	1.292	86	71.07	952.91
1.56E-05	29	275	3676.76	667.28	1.255	78.8	227.71	3676.76
2.74E-05	37.2	300.9	6248.32	1396.86	0.749	149	440.92	6088.78
6.14E-06	25	117.5	1279.25	135.8	0.933	75.2	227.71	739.7
5.09E-06	17	250.7	7411.54	146.4	1.232	104	56.57	1082
2.66E-05	42.8	253.4	4024.86	1390.75	0.928	82.4	440.92	3968.29
4.14E-06	19.8	244	7137.42	135.47	1.347	100.4	156.64	1124.06
4.46E-06	6.5	210.2	4808.08	93.77	1.429	128.3	71.07	697.64
1.29E-05	31.1	185	3755.09	756.67	0.879	104	440.92	3755.09
2.36E-05	45	276.8	6510.85	1664.63	0.85	194	867.34	5697.17
1.11E-05	42.5	150.1	3295.31	641.47	1.113	84.2	327.79	1517.12
4.99E-06	19	238.3	7047.49	113.7	1.172	100.4	156.64	1047.19
8.23E-06	37.4	150.8	2503.39	417	1.268	77	440.92	1351.77
5.97E-06	24.7	162.5	4011.81	162.67	0.866	104	56.57	1045.74
1.55E-05	45.4	225.5	5064.8	971.75	0.695	104	298.78	4523.8
1.51E-05	43.6	159.1	3498.36	941.05	0.861	104.9	116.03	2488.89
1.30E-05	26	275	3713.02	578.51	1.241	78.8	227.71	3314.16
1.85E-05	37.2	176	4873.34	1365.21	0.927	68	298.78	3753.64
6.96E-06	16	211.3	4281.58	338	0.784	116.6	298.78	3769.59
5.19E-06	19.9	231.8	6856.04	121.64	1.005	104	227.71	1067.49
1.21E-05	26.4	255.2	3669.51	643.58	1.228	87.8	227.71	3669.51
9.42E-06	28.6	111.2	1216.89	200.31	0.986	75.2	440.92	994.97
7.94E-06	37.8	145.4	2423.62	234.01	1.416	91.9	440.92	783.22
5.08E-06	19.2	158	1593.12	109.93	1.412	69.8	71.07	469.93
5.00E-06	15.2	214	3784.09	54.13	1.064	122	129.09	570.01

1.61E-05	37.2	190.4	3215.54	871.65	0.92	100.4	242.22	3037.14
1.92E-05	43.5	167	3612.95	1264.28	0.913	77	440.92	3441.8
1.17E-05	38.8	238.1	5305.56	645.41	0.782	122	227.71	3456.3
1.81E-05	42.6	167	3314.16	1165.01	0.945	90.3	440.92	3214.09
6.52E-06	29	105.8	1579.49	233.4	0.624	71.6	156.64	1351.77
9.49E-06	37.5	146.3	3103.86	576.95	0.909	104	440.92	2574.46
1.10E-05	37.2	152.6	2423.62	495.45	1.054	104	227.71	1607.04
1.18E-05	27.3	262.4	3710.12	591.17	1.295	194	511.99	3186.53
4.57E-06	15.6	131.4	1038.49	102.82	0.788	78.8	29.01	754.21
7.66E-06	28.8	208.4	4822.58	416.73	1.252	109.9	440.92	1550.48
1.74E-05	40.6	269.6	6358.55	1253.73	0.861	158	440.92	4565.86
2.28E-05	42	287.6	6747.26	1897.08	0.81	176	440.92	5788.55
6.19E-06	31.3	120.2	1337.27	55.19	1.218	74.3	58.02	171.15
5.35E-06	19.5	240.8	7211.39	115.98	1.059	100.4	227.71	1038.49
1.38E-05	31.7	176	3499.82	785.26	0.981	98.6	513.44	3385.23
2.08E-05	37.7	305.1	6613.82	1654.36	0.738	158.9	227.71	6613.82
8.58E-06	40.4	252	5518.77	233.24	0.985	113	156.64	1276.35

APPENDIX B

PROGRAM CODE LIST

GMDH Build Function

```
function [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs,
inputsMore, ...
maxNumNeurons, decNumNeurons, p, critNum, delta, Xv, Yv, verbose)
% GMDHBUILD
% Builds a GMDH-type polynomial neural network using a simple
% layer-by-layer approach
%
% Call
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons, p, critNum, delta, Xv, Yv,
verbose)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons, p, critNum, delta, Xv, Yv)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons, p, critNum, delta)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons, p, critNum)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons, p)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons,
%                     decNumNeurons)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore,
maxNumNeurons)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs, inputsMore)
%   [model, time] = gmdhbuild(Xtr, Ytr, maxNumInputs)
%   [model, time] = gmdhbuild(Xtr, Ytr)
%
% Input
% Xtr, Ytr     : Training data points (Xtr(i,:), Ytr(i)), i =
1,...,n
% maxNumInputs : Maximum number of inputs for individual neurons -
if set
%                to 3, both 2 and 3 inputs will be tried (default =
2)
% inputsMore   : Set to 0 for the neurons to take inputs only from
the
%                preceding layer, set to 1 to take inputs also from
the
%                original input variables (default = 1)
% maxNumNeurons: Maximal number of neurons in a layer (default =
equal to
%                the number of the original input variables)
% decNumNeurons: In each following layer decrease the number of
allowed
%                neurons by decNumNeurons until the number is equal
to 1
%                (default = 0)
```

```
% p            : Degree of polynomials in neurons (allowed values
are 2 and
%                3) (default = 2)
% critNum      : Criterion for evaluation of neurons and for
stopping.
%                In each layer only the best neurons (according to
the
%                criterion) are retained, and the rest are
discarded.
%                (default = 2)
%                0 = use validation data (Xv, Yv)
%                1 = use validation data (Xv, Yv) as well as
training data
%                2 = use Corrected Akaike's Information Criterion
(AICC)
%                3 = use Minimum Description Length (MDL)
%                Note that both choices 0 and 1 correspond to the so
called
%                "regularity criterion".
% delta        : How much lower the criterion value of the network's
new
%                layer must be comparing the the network's preceding
layer
%                (default = 0, which means that new layers will be
added as
%                long as the value gets better (smaller))
% Xv, Yv       : Validation data points (Xv(i,:), Yv(i)), i =
1,...,nv
%                (used when critNum is equal to either 0 or 1)
% verbose      : Set to 0 for no verbose (default = 1)
%
% Output
% model        : GMDH model - a struct with the following elements:
%    numLayers     : Number of layers in the network
%    d             : Number of input variables in the training data
set
%    maxNumInputs  : Maximal number of inputs for neurons
%    inputsMore    : See argument "inputsMore"
%    maxNumNeurons : Maximal number of neurons in a layer
%    p             : See argument "p"
%    critNum       : See argument "critNum"
%    layer         : Full information about each layer (number of
neurons,
%                    indexes of inputs for neurons, matrix of
exponents for
%                    polynomial, polynomial coefficients)
%                    Note that the indexes of inputs are in range
[1..d] if
%                    an input is one of the original input
variables, and
%                    in range [d+1..d+maxNumNeurons] if an input is
taken
%                    from a neuron in the preceding layer.
% time         : Execution time (in seconds)
%
% Please give a reference to the software web page in any
publication
% describing research performed using the software e.g., like this:
% Jekabsons G. GMDH-type Polynomial Neural Networks for Matlab,
2010,
% available at http://www.cs.rtu.lv/jekabsons/
```

```
% This source code is tested with Matlab version 7.1 (R14SP3).

%
================================================================
=====
% GMDH-type polynomial neural network
% Version: 1.5
% Date: June 2, 2011
% Author: Gints Jekabsons (gints.jekabsons@rtu.lv)
% URL: http://www.cs.rtu.lv/jekabsons/
%
% Copyright (C) 2009-2011  Gints Jekabsons
%
% This program is free software: you can redistribute it and/or
modify
% it under the terms of the GNU General Public License as published
by
% the Free Software Foundation, either version 2 of the License, or
% (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License
% along with this program. If not, see
<http://www.gnu.org/licenses/>.
%
================================================================
=====

display !!
if nargin < 2
    error('Too few input arguments.');
end

[n, d] = size(Xtr);
[ny, dy] = size(Ytr);
if (n < 2) || (d < 2) || (ny ~= n) || (dy ~= 1)
    error('Wrong training data sizes.');
end

if nargin < 3
    maxNumInputs = 2;
elseif (maxNumInputs ~= 2) && (maxNumInputs ~= 3)
    error('Number of inputs for neurons should be 2 or 3.');
end
if (d < maxNumInputs)
    error('Numbet of input variables in the data is lower than the
number of inputs for individual neurons.');
end
if nargin < 4
    inputsMore = 1;
end
if (nargin < 5) || (maxNumNeurons <= 0)
    maxNumNeurons = d;
end
if maxNumNeurons > d * 2
```

```
    error('Too many neurons in a layer. Maximum is two times the
number of input variables.');
end
if maxNumNeurons < 1
    error('Too few neurons in a layer. Minimum is 1.');
end
if (nargin < 6) || (decNumNeurons < 0)
    decNumNeurons = 0;
end
if nargin < 7
    p = 2;
elseif (p ~= 2) && (p ~= 3)
    error('Degree of individual neurons should be 2 or 3.');
end
if nargin < 8
    critNum = 2;
end
if any(critNum == [0,1,2,3]) == 0
    error('Only four values for critNum are available (0,1 - use
validation data; 2 - AICC; 3 - MDL).');
end
if nargin < 9
    delta = 0;
end
if (nargin < 11) && (critNum <= 1)
    error('Evaluating the models in validation data requires
validation data set.');
end
if (nargin >= 11) && (critNum <= 1)
    [nv, dv] = size(Xv);
    [nvy, dvy] = size(Yv);
    if (nv < 1) || (dv ~= d) || (nvy ~= nv) || (dvy ~= 1)
        error('Wrong validation data sizes.');
    end
end
if nargin < 12
    verbose = 1;
end

ws = warning('off');
if verbose ~= 0
    fprintf('Building GMDH-type neural network...\n');
end
tic;

if p == 2
    numTermsReal = 6 + 4 * (maxNumInputs == 3); %6 or 10 terms
else
    numTermsReal = 10 + 10 * (maxNumInputs == 3); %10 or 20 terms
end

Xtr(:, d+1:d+maxNumNeurons) = zeros(n, maxNumNeurons);
if critNum <= 1
    Xv(:, d+1:d+maxNumNeurons) = zeros(nv, maxNumNeurons);
end

%start the main loop and create layers
model.numLayers = 0;
while 1
```

```
    if verbose ~= 0
        fprintf('Building layer #%d...\n', model.numLayers + 1);
    end

    layer(model.numLayers + 1).numNeurons = 0;
    modelsTried = 0;
    layer(model.numLayers + 1).coefs = zeros(maxNumNeurons,
numTermsReal);

    for numInputsTry = maxNumInputs:-1:2

        %create matrix of exponents for polynomials
        if p == 2
            numTerms = 6 + 4 * (numInputsTry == 3); %6 or 10 terms
            if numInputsTry == 2
                r = [0,0;0,1;1,0;1,1;0,2;2,0];
            else
                r =
[0,0,0;0,0,1;0,1,0;1,0,0;0,1,1;1,0,1;1,1,0;0,0,2;0,2,0;2,0,0];
            end
        else
            numTerms = 10 + 10 * (numInputsTry == 3); %10 or 20
terms
            if numInputsTry == 2
                r = [0,0;0,1;1,0;1,1;0,2;2,0;1,2;2,1;0,3;3,0];
            else
                r =
[0,0,0;0,0,1;0,1,0;1,0,0;0,1,1;1,0,1;1,1,0;0,0,2;0,2,0;2,0,0; ...

1,1,1;0,1,2;0,2,1;1,0,2;1,2,0;2,0,1;2,1,0;0,0,3;0,3,0;3,0,0];
            end
        end

        %create matrix of all combinations of inputs for neurons
        if model.numLayers == 0
            combs = nchoosek(1:1:d, numInputsTry);
        else
            if inputsMore == 1
                combs = nchoosek([1:1:d
d+1:1:d+layer(model.numLayers).numNeurons], numInputsTry);
            else
                combs =
nchoosek(d+1:1:d+layer(model.numLayers).numNeurons, numInputsTry);
            end
        end
        %delete all combinations in which none of the inputs are
from the preceding layer
        if model.numLayers > 0
            i = 1;
            while i <= size(combs,1)
                if all(combs(i,:) <= d)
                    combs(i,:) = [];
                else
                    i = i + 1;
                end
            end
        end

        makeEmpty = 1;
```

```
        %try all the combinations of inputs for neurons
        for i = 1 : size(combs,1)

            %create matrix for all polynomial terms
            Vals = ones(n, numTerms);
            if critNum <= 1
                Valsv = ones(nv, numTerms);
            end
            for idx = 2 : numTerms
                bf = r(idx, :);
                t = bf > 0;
                tmp = Xtr(:, combs(i,t)) .^ bf(ones(n, 1), t);
                if critNum <= 1
                    tmpv = Xv(:, combs(i,t)) .^ bf(ones(nv, 1), t);
                end
                if size(tmp, 2) == 1
                    Vals(:, idx) = tmp;
                    if critNum <= 1
                        Valsv(:, idx) = tmpv;
                    end
                else
                    Vals(:, idx) = prod(tmp, 2);
                    if critNum <= 1
                        Valsv(:, idx) = prod(tmpv, 2);
                    end
                end
            end

            %calculate coefficients and evaluate the network
            coefs = (Vals' * Vals) \ (Vals' * Ytr);
            modelsTried = modelsTried + 1;
            if ~isnan(coefs(1))
                predY = Vals * coefs;
                if critNum <= 1
                    predYv = Valsv * coefs;
                    if critNum == 0
                        crit = sqrt(mean((predYv - Yv).^2));
                    else
                        crit = sqrt(mean([(predYv - Yv).^2; (predY -
Ytr).^2]));
                    end
                else
                    comp = complexity(layer, model.numLayers,
maxNumNeurons, d, combs(i,:)) + size(coefs, 2);
                    if critNum == 2 %AICC
                        if (n-comp-1 > 0)
                            crit = n*log(mean((predY - Ytr).^2)) +
2*comp + 2*comp*(comp+1)/(n-comp-1);
                        else
                            coefs = NaN;
                        end
                    else %MDL
                        crit = n*log(mean((predY - Ytr).^2)) +
comp*log(n);
                    end
                end
            end

            if ~isnan(coefs(1))
                %add the neuron to the layer if
                %1) the layer is not full;
```

```
                %2) the new neuron is better than an existing worst
one.
                maxN = maxNumNeurons - model.numLayers *
decNumNeurons;
                if maxN < 1, maxN = 1; end;
                if layer(model.numLayers + 1).numNeurons < maxN
                    %when the layer is not yet full
                    if (maxNumInputs == 3) && (numInputsTry == 2)
                        layer(model.numLayers +
1).coefs(layer(model.numLayers + 1).numNeurons+1, :) = [coefs'
zeros(1,4+6*(p == 3))];
                        layer(model.numLayers +
1).inputs(layer(model.numLayers + 1).numNeurons+1, :) = [combs(i, :)
0];
                    else
                        layer(model.numLayers +
1).coefs(layer(model.numLayers + 1).numNeurons+1, :) = coefs;
                        layer(model.numLayers +
1).inputs(layer(model.numLayers + 1).numNeurons+1, :) = combs(i, :);
                    end
                    layer(model.numLayers +
1).comp(layer(model.numLayers + 1).numNeurons+1) = length(coefs);
                    layer(model.numLayers +
1).crit(layer(model.numLayers + 1).numNeurons+1) = crit;
                    layer(model.numLayers +
1).terms(layer(model.numLayers + 1).numNeurons+1).r = r;
                    if makeEmpty == 1
                        Xtr2 = [];
                        if critNum <= 1
                            Xv2 = [];
                        end
                        makeEmpty = 0;
                    end
                    Xtr2(:, layer(model.numLayers + 1).numNeurons+1)
= predY;
                    if critNum <= 1
                        Xv2(:, layer(model.numLayers +
1).numNeurons+1) = predYv;
                    end
                    if (layer(model.numLayers + 1).numNeurons == 0)
|| ...
                       (layer(model.numLayers + 1).crit(worstOne) <
crit)
                        worstOne = layer(model.numLayers +
1).numNeurons + 1;
                    end
                    layer(model.numLayers + 1).numNeurons =
layer(model.numLayers + 1).numNeurons + 1;
                else
                    %when the layer is already full
                    if (layer(model.numLayers + 1).crit(worstOne) >
crit)
                        if (maxNumInputs == 3) && (numInputsTry ==
2)
                            layer(model.numLayers +
1).coefs(worstOne, :) = [coefs' zeros(1,4+6*(p == 3))];
                            layer(model.numLayers +
1).inputs(worstOne, :) = [combs(i, :) 0];
                        else
                            layer(model.numLayers +
1).coefs(worstOne, :) = coefs;
```

```
                            layer(model.numLayers +
1).inputs(worstOne, :) = combs(i, :);
                        end
                        layer(model.numLayers + 1).comp(worstOne) =
length(coefs);
                        layer(model.numLayers + 1).crit(worstOne) =
crit;
                        layer(model.numLayers + 1).terms(worstOne).r
= r;
                        Xtr2(:, worstOne) = predY;
                        if critNum <= 1
                            Xv2(:, worstOne) = predYv;
                        end
                        [dummy, worstOne] =
max(layer(model.numLayers + 1).crit);
                    end
                end
            end

        end

    end

    if verbose ~= 0
        fprintf('Neurons tried in this layer: %d\n', modelsTried);
        fprintf('Neurons included in this layer: %d\n',
layer(model.numLayers + 1).numNeurons);
        if critNum <= 1
            fprintf('RMSE in the validation data of the best neuron:
%f\n', min(layer(model.numLayers + 1).crit));
        else
            fprintf('Criterion value of the best neuron: %f\n',
min(layer(model.numLayers + 1).crit));
        end
    end

    %stop the process if there are too few neurons in the new layer
    if ((inputsMore == 0) && (layer(model.numLayers + 1).numNeurons
< 2)) || ...
       ((inputsMore == 1) && (layer(model.numLayers + 1).numNeurons
< 1))
        if (layer(model.numLayers + 1).numNeurons > 0)
            model.numLayers = model.numLayers + 1;
        end
        break
    end

    %if the network got "better", continue the process
    if (layer(model.numLayers + 1).numNeurons > 0) && ...
       ((model.numLayers == 0) || ...
        (min(layer(model.numLayers).crit) -
min(layer(model.numLayers + 1).crit) > delta) )
%(min(layer(model.numLayers + 1).crit) <
min(layer(model.numLayers).crit)) )
        model.numLayers = model.numLayers + 1;
    else
        if model.numLayers == 0
            warning(ws);
            error('Failed.');
        end
```

```
        break
    end

    %copy the output values of this layer's neurons to the training
    %data matrix
    Xtr(:, d+1:d+layer(model.numLayers).numNeurons) = Xtr2;
    if critNum <= 1
        Xv(:, d+1:d+layer(model.numLayers).numNeurons) = Xv2;
    end

end

model.d = d;
model.maxNumInputs = maxNumInputs;
model.inputsMore = inputsMore;
model.maxNumNeurons = maxNumNeurons;
model.p = p;
model.critNum = critNum;

%only the neurons which are actually used (directly or indirectly)
to
%compute the output value may stay in the network
[dummy best] = min(layer(model.numLayers).crit);
model.layer(model.numLayers).coefs(1,:) =
layer(model.numLayers).coefs(best,:);
model.layer(model.numLayers).inputs(1,:) =
layer(model.numLayers).inputs(best,:);
model.layer(model.numLayers).terms(1).r =
layer(model.numLayers).terms(best).r;
model.layer(model.numLayers).numNeurons = 1;
if model.numLayers > 1
    for i = model.numLayers-1:-1:1 %loop through all the layers
        model.layer(i).numNeurons = 0;
        for k = 1 : layer(i).numNeurons %loop through all the
neurons in this layer
            newNum = 0;
            for j = 1 : model.layer(i+1).numNeurons %loop through
all the neurons which will stay in the next layer
                for jj = 1 : maxNumInputs %loop through all the
inputs
                    if k == model.layer(i+1).inputs(j,jj) - d
                        if newNum == 0
                            model.layer(i).numNeurons =
model.layer(i).numNeurons + 1;

model.layer(i).coefs(model.layer(i).numNeurons,:) =
layer(i).coefs(k,:);

model.layer(i).inputs(model.layer(i).numNeurons,:) =
layer(i).inputs(k,:);

model.layer(i).terms(model.layer(i).numNeurons).r =
layer(i).terms(k).r;
                            newNum = model.layer(i).numNeurons + d;
                            model.layer(i+1).inputs(j,jj) = newNum;
                        else
                            model.layer(i+1).inputs(j,jj) = newNum;
                        end
                        break
                    end
                end
```

```
            end
        end
    end
end

time = toc;
warning(ws);

if verbose ~= 0
    fprintf('Done.\n');
    used = zeros(d,1);
    for i = 1 : model.numLayers
        for j = 1 : d
            if any(any(model.layer(i).inputs == j))
                used(j) = 1;
            end
        end
    end
    fprintf('Number of layers: %d\n', model.numLayers);
    fprintf('Number of used input variables: %d\n', sum(used));
    fprintf('Execution time: %0.2f seconds\n', time);
end

return

%====================  Auxiliary functions  ====================

function [comp] = complexity(layer, numLayers, maxNumNeurons, d,
connections)
%calculates the complexity of the network given output neuron's
connections
%(it is assumed that the complexity of a network is equal to the
number of
%all polynomial terms in all it's neurons which are actually
connected
%(directly or indirectly) to network's output)
comp = 0;
if numLayers == 0
    return
end
c = zeros(numLayers, maxNumNeurons);
for i = 1 : numLayers
    c(i, :) = layer(i).comp(:)';
end
%{
%unvectorized version:
for j = 1 : length(connections)
    if connections(j) > d
        comp = comp + c(numLayers, connections(j) - d);
        c(numLayers, connections(j) - d) = -1;
    end
end
%}
ind = connections > d;
if any(ind)
    comp = comp + sum(c(numLayers, connections(ind) - d));
    c(numLayers, connections(ind) - d) = -1;
end
%{
%unvectorized version:
```

```
for i = numLayers-1:-1:1
    for j = 1 : layer(i).numNeurons
        for k = 1 : layer(i+1).numNeurons
            if (c(i+1, k) == -1) && (c(i, j) > -1) && ...
               any(layer(i+1).inputs(k,:) == j + d)
                comp = comp + c(i, j);
                c(i, j) = -1;
            end
        end
    end
end
%}
for i = numLayers-1:-1:1
        for k = 1 : layer(i+1).numNeurons
            if c(i+1, k) == -1
                inp = layer(i+1).inputs(k,:);
                used = inp > d;
                if any(used)
                    ind = inp(used) - d;
                    ind = ind(c(i, ind) > -1);
                    if ~isempty(ind)
                        comp = comp + sum(c(i, ind));
                        c(i, ind) = -1;
                    end
                end
            end
        end
end
return
```

GMDH EQ Function

```
function gmdheq(model, precision)
% gmdheq
% Outputs the equations of GMDH model.
%
% Call
%   gmdheq(model, precision)
%   gmdheq(model)
%
% Input
%   model         : GMDH-type model
%   precision     : Number of digits in the model coefficients
%                   (default = 15)

% This source code is tested with Matlab version 7.1 (R14SP3).

%
================================================================================
% GMDH-type polynomial neural network
% Version: 1.5
% Date: June 2, 2011
% Author: Gints Jekabsons (gints.jekabsons@rtu.lv)
% URL: http://www.cs.rtu.lv/jekabsons/
%
% Copyright (C) 2009-2011  Gints Jekabsons
%
% This program is free software: you can redistribute it and/or
modify
% it under the terms of the GNU General Public License as published
by
% the Free Software Foundation, either version 2 of the License, or
% (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License
% along with this program. If not, see
<http://www.gnu.org/licenses/>.
%
================================================================================

if nargin < 1
    error('Too few input arguments.');
end
if (nargin < 2) || (isempty(precision))
    precision = 15;
end

if model.numLayers > 0
    p = ['%.' num2str(precision) 'g'];
    fprintf('Number of layers: %d\n', model.numLayers);
    for i = 1 : model.numLayers %loop through all the layers
        fprintf('Layer #%d\n', i);
```

```
        fprintf('Number of neurons: %d\n',
model.layer(i).numNeurons);
        for j = 1 : model.layer(i).numNeurons %loop through all the
neurons in the ith layer
            [terms inputs] = size(model.layer(i).terms(j).r);
%number of terms and inputs
            if (i == model.numLayers)
                str = ['y = ' num2str(model.layer(i).coefs(j,1),p)];
            else
                str = ['x' num2str(j + i*model.d) ' = '
num2str(model.layer(i).coefs(j,1),p)];
            end
            for k = 2 : terms %loop through all the terms
                if model.layer(i).coefs(j,k) >= 0
                    str = [str ' +'];
                else
                    str = [str ' '];
                end
                str = [str num2str(model.layer(i).coefs(j,k),p)];
                for kk = 1 : inputs %loop through all the inputs
                    if (model.layer(i).terms(j).r(k,kk) > 0)
                        for kkk = 1 :
model.layer(i).terms(j).r(k,kk)
                            if (model.layer(i).inputs(j,kk) <=
model.d)
                                str = [str '*x'
num2str(model.layer(i).inputs(j,kk))];
                            else
                                str = [str '*x'
num2str(model.layer(i).inputs(j,kk) + (i-2)*model.d)];
                            end
                        end
                    end
                end
            end
            disp(str);
        end
    end
else
    disp('The network has zero layers.');
end

return
```

GMDH Predict Function

```
function Yq = gmdhpredict(model, Xq)
% GMDHPREDICT
% Predicts output values for the given query points Xq using a GMDH
model
%
% Call
%   [Yq] = gmdhpredict(model, Xq)
%
% Input
% model     : GMDH model
% Xq        : Inputs of query data points (Xq(i,:)), i = 1,...,nq
%
% Output
% Yq        : Predicted outputs of query data points (Yq(i)), i =
1,...,nq

% This source code is tested with Matlab version 7.1 (R14SP3).

%
=========================================================================
=====
% GMDH-type polynomial neural network
% Version: 1.5
% Date: June 2, 2011
% Author: Gints Jekabsons (gints.jekabsons@rtu.lv)
% URL: http://www.cs.rtu.lv/jekabsons/
%
% Copyright (C) 2009-2011  Gints Jekabsons
%
% This program is free software: you can redistribute it and/or
modify
% it under the terms of the GNU General Public License as published
by
% the Free Software Foundation, either version 2 of the License, or
% (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License
% along with this program. If not, see
<http://www.gnu.org/licenses/>.
%
=========================================================================
=====

if nargin < 2
    error('Too few input arguments.');
end
if model.d ~= size(Xq, 2)
    error('The matrix should have the same number of columns as the
matrix with which the network was built.');
end

[n, d] = size(Xq);
Yq = zeros(n, 1);
```

```
for q = 1 : n
    for i = 1 : model.numLayers
        if i ~= model.numLayers
            Xq_tmp = zeros(1, model.layer(i).numNeurons);
        end
        for j = 1 : model.layer(i).numNeurons

            %create matrix for all polynomial terms
            numTerms =  size(model.layer(i).terms(j).r,1);
            Vals = ones(numTerms,1);
            for idx = 2 : numTerms
                bf = model.layer(i).terms(j).r(idx, :);
                t = bf > 0;
                tmp = Xq(q, model.layer(i).inputs(j,t)) .^ bf(1, t);
                if size(tmp, 2) == 1
                    Vals(idx,1) = tmp;
                else
                    Vals(idx,1) = prod(tmp, 2);
                end
            end

            %predict output value
            predY = model.layer(i).coefs(j,1:numTerms) * Vals;
            if i ~= model.numLayers
                %Xq(q, d+j) = predY;
                Xq_tmp(j) = predY;
            else
                Yq(q) = predY;
            end

        end
        if i ~= model.numLayers
            Xq(q, d+1:d+model.layer(i).numNeurons) = Xq_tmp;
        end
    end
end

return
```

GMDH Test Function

```
function [MSE, RMSE, RRMSE, R2] = gmdhtest(model, Xtst, Ytst)
% GMDHTEST
% Tests a GMDH-type network model on a test data set (Xtst, Ytst)
%
% Call
%   [MSE, RMSE, RRMSE, R2] = gmdhtest(model, Xtst, Ytst)
%
% Input
% model     : GMDH model
% Xtst, Ytst: Test data points (Xtst(i,:), Ytst(i)), i = 1,...,ntst
%
% Output
% MSE       : Mean Squared Error
% RMSE      : Root Mean Squared Error
% RRMSE     : Relative Root Mean Squared Error
% R2        : Coefficient of Determination

% Copyright (C) 2009-2011  Gints Jekabsons

if nargin < 3
    error('Too few input arguments.');
end
if (size(Xtst, 1) ~= size(Ytst, 1))
    error('The number of rows in the matrix and the vector should be
equal.');
end
if model.d ~= size(Xtst, 2)
    error('The matrix should have the same number of columns as the
matrix with which the model was built.');
end
MSE = mean((gmdhpredict(model, Xtst) - Ytst) .^ 2);
RMSE = sqrt(MSE);
if size(Ytst, 1) > 1
    RRMSE = RMSE / std(Ytst, 1);
    R2 = 1 - MSE / var(Ytst, 1);
else
    RRMSE = Inf;
    R2 = Inf;
end
return
```

Polynomial GMDH Script

```
clc;
% the aim is to clear all input and output from the Command Window
% display, giving you a "clean screen."
clf; % it deletes from the current figure all graphics objects
clear all;%Clears all variables and other classes of data too.
close all;% it force deletes all figures (hidden and non-hidden
strings)
tic;
%
% Step (1) Reading the input file
% ================================
% Loads data and prepares it for a neural network.
%ndata= xlsread('all_data.xls');
ndata= xlsread('OC_Data.xlsx');
%50% of data will be used for training
%25% of data will be used for cross-validation
%25% of data will be used for testing
for i=1:91
    atr(i,:)=ndata(i,:);
end
for i=92:137
    aval(i-91,:)=ndata(i,:);
end
%
for i=138:length(ndata)
    atest(i-137,:)=ndata(i,:);
end
Ytr=atr(:,1);
Xtr=atr(:,2:9);
Xtst=atest(:,2:9);
Ytst=atest(:,1);
Yv=aval(:,1);
Xv=aval(:,2:9);
[model, time] = gmdhbuild(Xtr, Ytr, 2, 0, 2, 0, 2, 1, 0.9, Xv,
Yv,1);
gmdheq(model, 3);
[Yqtst] = gmdhpredict(model, Xtst);
[Yqval] = gmdhpredict(model, Xv);
[Yqtr] = gmdhpredict(model, Xtr);
[MSE, RMSE, RRMSE, R2] = gmdhtest(model, Xtst, Ytst);

% Evaluating Relative Error for training set:
%============================================
Et1=(Ytr-Yqtr)./Ytr*100;
[q,z] = size(Et1);
figure
plot(Ytst,Yqtst,'o')
grid off
set(gcf, 'color', 'white')
axis ([0 4e-5 0 4e-5])

title('Predicted Oil Compressibility vs Measured Oil
Compressibility');
xlabel('Measured Oil Compressibility "1/psi"');
ylabel('Predicted Oil Compressibility "1/psi"')
legend('Training set', 'location', 'Northwest')
% Addding Reference Line with 45 degree slope
line([0 ; 5e-5],[0 ; 5e-5])
%HINT: Select the y-value based on your data limits
```

```
hold
% Evaluating the correlation coefficient for training set:
% ========================================================
Rt1=corrcoef(Yqtr,Ytr);
Rt11=min(Rt1(:,1));
gtext(['correlation coefficient = (' num2str(Rt11) ')']);
hold

% Addding Reference Line with 45 degree slope
%line([0 ; 300],[0 ; 300])
%HINT: Select the y-value based on your data limits

% Evaluating Relative Error for validation set:
%==============================================
Ev1=(Yv-Yqval)./Yv*100;
[m,n] = size(Ev1);
figure

plot(Yv,Yqval,'o')
grid off
set(gcf, 'color', 'white')
axis ([0 5e-5 0 5e-5])
title('Predicted Oil Compressibility vs. Measured Oil
Compressibility');
xlabel('Measured Oil Compressibility "1/psi"');
ylabel('Predicted Oil Compressibility "1/psi"')
legend('Validation set', 'location', 'Northwest')
% Addding Reference Line with 45 degree slope
line([0 ; 5e-5],[0 ; 5e-5])
%HINT: Select the y-value based on your data limits

% Evaluating the correlation coefficient for validation set:
% ========================================================
% for the first target Pressure Drop
Rv1=corrcoef(Yqval,Yv);
Rv11=min(Rv1(:,1));
gtext(['correlation coefficient = (' num2str(Rv11) ')']);
hold

% Evaluating Relative Error for testing set:
%===========================================
% for the first target Pressure Drop
Ett1=(Ytst-Yqtst)./Ytst*100;
[m,n] = size(Ett1);
figure
%
plot(Ytst,Yqtst,'o')
grid off
set(gcf, 'color', 'white')
axis([0 4e-5 0 4e-5])

title('Predicted Oil Compressibility vs.Measured Oil
Compressibility');
xlabel('Measured Oil Compressibility "1/psi"');
ylabel('Predicted Oil Compressibility "1/psi"')
legend('Testing set', 'location', 'Northwest')
% Addding Reference Line with 45 degree slope
line([0 ; 5e-5],[0 ; 5e-5])
%HINT: Select the y-value based on your data limits
```

```
% Evaluating the correlation coefficient for testing set:
% ========================================================
Rtt1=corrcoef(Yqtst,Ytst);
Rtt11=min(Rtt1(:,1));
gtext(['correlation coefficient = (' num2str(Rtt11) ')']);
hold
% plotting the histogram of the errors for training set:
% ========================================================
figure
histfit(Et1,10)
%hist(Et1,10)
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
title('Error Distribution for Training Set (Polynomial GMDH
Model)');
legend('Training set')
xlabel('Error');
ylabel('Frequency')
set(gcf, 'color', 'white')
hold

% plotting the histogram of the errors for validation set:
% ========================================================
figure
histfit(Ev1,10)
%hist(Ev1,10)
h = findobj(gca, 'Type', 'patch');
set(h,'FaceColor','w','EdgeColor','k')
title('Error Distribution for Validation Set (Polynomial GMDH
Model)');
legend('Validation set')
xlabel('Error');
ylabel('Frequency')
set(gcf, 'color', 'white')
hold

% plotting the histogram of the errors for testing set:
% ========================================================
figure
histfit(Ett1,10)
%hist(Ett1,10)
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
title('Error Distribution for Testing Set (Polynomial GMDH Model)');
legend('Testing set')
xlabel('Error');
ylabel('Frequency')
set(gcf, 'color', 'white')
hold
% Estimating the residuals for training set:
% ==========================================
figure
Errort1 = Yqtr-Ytr;
plot(Errort1,':ro');
grid off
set(gcf, 'color', 'white')
title('Error Distribution for Training Set (Polynomial GMDH Model)')
legend('Training Set')
xlabel('Data Point No')
ylabel('Errors')
hold
```

```
% Estimating the residuals for validation set:
% ============================================
figure
Errorv1 = Yqval-Yv;
plot(Errorv1,':ro');
grid off
set(gcf, 'color', 'white')
title('Residual Graph for Validation Set (Polynomial GMDH Model)')
legend('Validation Set')
xlabel('Data Point No')
ylabel('Errors')
hold
% Estimating the residuals for testing set:
% ========================================
figure
Errortt1 = Yqtst-Ytst;
plot(Errortt1,':ro');
grid off
set(gcf, 'color', 'white')
title('Residual Graph for Testing Set (Polynomial GMDH Model)')
legend('Testing Set')
xlabel('Data Point No')
ylabel('Errors')

% ********************
% STATISTICAL ANALYSIS:
% ********************
% Training set:
% =============
% Determining the Maximum Absolute Percent Relative Error
MaxErrt1 = max(abs(Et1));

% Evaluating the average error
Etavg1 = 1/q*sum(Et1);

% Evaluating the standard deviation
STDT1 = std(Errort1);

% Determining the Minimum Absolute Percent Relative Error
MinErrt1 = min(abs(Et1));

% Evaluating Average Absolute Percent Relative Error
% ================================================
AAPET1 = sum(abs(Et1))/q;

% Evaluating Average Percent Relative Error
% ========================================
APET1 = 1/q*sum(Et1);

% Evaluating Root Mean Square
% ===========================
RMSET1 = sqrt(sum(abs(Et1).^2)/q);

% Validation set:
% ===============
% Determining the Maximum Absolute Percent Relative Error
MaxErrv1 = max(abs(Ev1));

% Determining the Minimum Absolute Percent Relative Error
MinErrv1 = min(abs(Ev1));
```

```
% Evaluating the average error
Evavg1 = 1/m*sum(Ev1);

% Evaluating the standard deviation
STDV1 = std(Errorv1);

%
% Evaluating Average Absolute Percent Relative Error
% ================================================
AAPEV1 = sum(abs(Ev1))/m;

% Evaluating Average Percent Relative Error
% =======================================
APEV1 = 1/m*sum(Ev1);

% Evaluating Root Mean Square
% ==========================
RMSEV1 = sqrt(sum(abs(Ev1).^2)/m);

% Testing set:
% ============
% Determining the Maximum Absolute Percent Relative Error
MaxErrtt1 = max(abs(Ett1));

% Determining the Minimum Absolute Percent Relative Error
MinErrtt1 = min(abs(Ett1));

% Evaluating the average error
Ettavg1 = 1/m*sum(Ett1);

% Evaluating the standard deviation
STDTT1 = std(Errortt1);

% Evaluating Average Absolute Percent Relative Error
% ================================================
AAPETT1 = sum(abs(Ett1))/m;

% Evaluating Average Percent Relative Error
% =======================================
APETT1 = 1/m*sum(Ett1);

% Evaluating Root Mean Square
% ==========================
RMSETT1 = sqrt(sum(abs(Ett1).^2)/m);



% ================================================
%-----------------------------------------------------------------------
-----------
% Simulation: Variation Reservoir Pressure while fixing the other
parameters
% ------------RESERVOIR PRESSURE VARIATION-------------------------
------------

ps1=[linspace(30.18,30.18,10); %DEGREE API [min=6    max=56.8
mean=30.18]
```

```
linspace(201.37,201.37,10);%RESERVOIR TEMPERATURE[min=80.6
max=341.6 mean=201.37]
linspace(242.22,15304.62,10);%RESERVOIR PRESSURE [min=242.22
max=15304.62  mean=4204.37]
linspace(614.60,614.60,10);%SOLUTION GOR [min=8.61    max=3298.66
mean=614.60]
linspace(1.044,1.044,10);%GAS GRAVITY[min=0.624    max=1.789
mean=1.044]
linspace(108.58,108.58,10);%SEPARATOR TEMPERATURE [min=59   max=194
mean=108.58]
linspace(254.88,254.88,10);%SEPARATOR PRESSURE [min=14.5
max=868.79  mean=254.88]
linspace(2357.18,2357.18,10)]';%BUBBLEPOINT PRESSURE [min=107.33
max=6613.82    mean=2357.18]

% Now simulate
[Yq_length] = gmdhpredict(model, ps1);
% Plot Figures for Reservoir Pressure Variation
figure
px1=plot(ps1(:,3),Yq_length(:,1),'-rs');
set(gca,'YGrid','off','XGrid','off')
set(gca,'FontSize',12,'LineWidth',2);
set(px1,'LineStyle','-.','LineWidth',1.5,'Color','k','MarkerSize',6)
xlabel('Reservoir Pressure (psia)','FontSize',12)
ylabel('Oil Compressibility (1/psi)', 'fontsize',12)

%-------------------------------------------------------------------
-----------
% Simulation: Variation of Solution GOR while fixing the other
parameters
% ------------SOLUTION GOR VARIATION-------------------------------
------

ps2=[linspace(30.18,30.18,10); %DEGREE API [min=6    max=56.8
mean=30.18]
linspace(201.37,201.37,10);%RESERVOIR TEMPERATURE[min=80.6
max=341.6 mean=201.37]
linspace(4204.37,4204.37,10);%RESERVOIR PRESSURE [min=242.22
max=15304.62  mean=4204.37]
linspace(68.61,3298.66,10);%SOLUTION GOR [min=8.61    max=3298.66
mean=614.60]
linspace(1.044,1.044,10);%GAS GRAVITY[min=0.624    max=1.789
mean=1.044]
linspace(108.58,108.58,10);%SEPARATOR TEMPERATURE [min=59   max=194
mean=108.58]
linspace(254.88,254.88,10);%SEPARATOR PRESSURE [min=14.5
max=868.79  mean=254.88]
linspace(2357.18,2357.18,10)]';%BUBBLEPOINT PRESSURE [min=107.33
max=6613.82    mean=2357.18]

% Now simulate
[Yq_angle] = gmdhpredict(model, ps2);
% Plot Figures for Solution GOR Variation
figure
px2=plot(ps2(:,4),Yq_angle(:,1),'-rs');
set(gca,'YGrid','off','XGrid','off')
set(gca,'FontSize',12,'LineWidth',2);
set(px2,'LineStyle','-.','LineWidth',1.5,'Color','k','MarkerSize',6)
xlabel('Solution GOR (scf/STB)','FontSize',12)
ylabel('Oil Compressibility (1/psi)', 'fontsize',12)
```

```
%-----------------------------------------------------------------
-----------
% Simulation: Variation of Bubble Point Pressure while fixing the
other parameters
% ------------BUBBLE POINT PRESSURE VARIATION----------------------
---------------

ps3=[linspace(30.18,30.18,10); %DEGREE API [min=6    max=56.8
mean=30.18]
linspace(201.37,201.37,10);%RESERVOIR TEMPERATURE[min=80.6
max=341.6 mean=201.37]
linspace(7613.82,7613.82,10);%RESERVOIR PRESSURE [min=242.22
max=15304.62  mean=4204.37]
linspace(614.60,614.60,10);%SOLUTION GOR [min=8.61    max=3298.66
mean=614.60]
linspace(1.044,1.044,10);%GAS GRAVITY[min=0.624    max=1.789
mean=1.044]
linspace(108.58,108.58,10);%SEPARATOR TEMPERATURE [min=59   max=194
mean=108.58]
linspace(254.88,254.88,10);%SEPARATOR PRESSURE [min=14.5
max=868.79  mean=254.88]
linspace(107.33,6613.82,10)]';%BUBBLEPOINT PRESSURE [min=107.33
max=6613.82    mean=2357.18]

% Now simulate
[Yq_angle] = gmdhpredict(model, ps3);
% Plot Figures for Bubble Point Variation
figure
px3=plot(ps3(:,8),Yq_angle(:,1),'-rs');
set(gca,'YGrid','off','XGrid','off')
set(gca,'FontSize',12,'LineWidth',2);
set(px3,'LineStyle','-.','LineWidth',1.5,'Color','k','MarkerSize',6)
xlabel('Bubble Point Pressure (psia)','FontSize',12)
ylabel('Oil Compressibility (1/psi)', 'fontsize',12)
```