Prediction of Oil Density using Group Method of Data Handling (GMDH) Approach and the Effect of Reducing Correlating Parameters; A Comparative Study

by

Nur Syazwani binti Moktar
12152

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
Petroleum Engineering

May 2013

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Prediction of Oil Density using Group Method of Data Handling (GMDH) Approach and the Effect of Reducing Correlating Parameters; A Comparative Study

by

Nur Syazwani binti Moktar
12152

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
Petroleum Engineering

Approved by,

_____

(MR. ALI F. MANGI ALTA'EE)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
May 2013

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 acknowledgement, and the original work contained herein have not been undertaken or done by unspecified sources or persons.

(NUR SYAZWANI MOKTAR)

ACKNOWLEDGEMENT

Alhamdulillah, I would like to express my greatest gratitude to Allah S.W.T. for His will upon completion of this project dissertation.

I would like to express my deepest appreciation and special thanks to my supervisor, Mr. Ali F. Mangi Alta'ee for his commitment, encouragement, guidance and support throughout this project. Not to be forgotten, my appreciation goes to my family and colleagues for their continuous supports upon my project completion. Thank you.

ABSTRACT

Reservoir fluid or PVT properties are one of the most important elements in petroleum engineering, especially in reservoir studies. It is required in material balance, reservoir simulation, volumetric calculations and others. With the laboratory studies and the aids of PVT correlations, PVT properties can be effectively obtained. PVT correlations existed in the oil and gas industry is widely used when the experimental data cannot be obtained or no fluid samples are available. However, some of the empirical correlations in the literature are controversial in aspects of its accuracy, validity and range of applicability. Recently, group method of data handling (GMDH) is introduced in the petroleum industry as another alternative to improve the accuracy of existing PVT correlations. This research proposes GMDH approach as a modeling tool for predicting crude oil density at bubble-point pressure. The objective of this research is to study the capability of GMDH in modeling oil density. The new oil density model incorporates three (3) correlating parameters: (1) bubble-point oil formation volume factor, (2) solution gas-oil ratio and (3) API gravity. A comparative study is carried out to compare the performance of the new oil density model with other existing correlations. The results obtained show that the oil density model with GMDH is more accurate and outperforms other known correlations.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

3.6: $SD = \left[\frac{1}{(n_d - 1)}\right]\sum_{i=1}^{n_d} E_i^2$ 24

3.7: $R = \sqrt{1 - \frac{\sum_{i=1}^{m}\left[\left(x_{exp} - x_{est}\right)_i\right]^2}{\sum_{i=1}^{m}\left[\left(x_{exp} - \bar{x}\right)_i\right]^2}}$ 25

3.8: $\bar{x} = \left(\frac{1}{n_d}\right)\sum_{i=1}^{n_d}\left(x_{exp}\right)_i$ 25

4.1: 29

$X_1 = a_1 + a_2API + a_3R_s + a_4B_{ob} + a_5R_sAPI + a_6B_{ob}API + a_7B_{ob}R_s + a_8(API)^2 + a_9(R_s)^2 + a_{10}(B_{ob})^2$

$X_2 = b_1 + b_2X_1 + b_3\gamma_g + b_4R_s + b_5\ X_1\gamma_g + b_6\ X_1R_s + b_7R_s\gamma_g + b_8(X_1)^2 + b_9(\gamma_g)^2 + b_{10}(R_s)^2$

$X_3 = c_1 + c_2X_2 + c_3API + c_4R_s + c_5X_2API + c_6X_2R_s + c_7R_sAPI + c_8(X_2)^2 + c_9(API)^2 + c_{10}(R_s)^2$

$X_4 = d_1 + d_2X_3 + d_3\gamma_g + d_4B_{ob} + d_5\ X_3\gamma_g + d_6X_3B_{ob} + d_7B_{ob}\gamma_g + d_8(X_3)^2 + d_9(\gamma_g)^2 + d_{10}(B_{ob})^2$

$\rho_{ob} = y_1 + y_2X_4 + y_3R_s + y_4B_{ob} + y_5X_4R_s + y_6X_4B_{ob} + y_7B_{ob}R_s + y_8(X_4)^2 + y_9(R_s)^2 + y_{10}(B_{ob})^2$

4.2 34

$\rho_{ob} = x_1 + x_2API + x_3R_s + x_4B_{ob} + x_5R_sAPI + x_6B_{ob}API + x_7B_{ob}R_s + x_8(API)^2 + x_9(R_s)^2 + x_{10}(B_{ob})^2$

LIST OF ABBREVIATIONS

1. PVT - Pressure-volume-temperature
2. GMDH - Group method of data handling
3. AAPE - Average absolute percentage error
4. AAPRE - Average absolute percent relative error
5. APRE - Average percent relative error

NOMENCLATURES

ENGLISH

API	unit for stock tank oil gravity
B_{ob}	bubble-point oil formation volume factor (bbl/STB)
E_r	average relative error (%)
E_a	average absolute relative error (%)
E_i	relative error (%)
m	number of data sets
n_d	number of data points
P_b	bubble-point pressure (psia)
R_s	solution gas-oil ratio (scf/STB)
R	coefficient correlation
R^2	coefficient of determination
SD	standard deviation
T_f	reservoir temperature (^oF)
$\bar{x}$	average value for x_{exp}

GREEK

ρ_o	oil density (lb/ft^3)
γ_g	gas specific gravity (air = 1.0)
γ_o	stock-tank oil specific gravity

SUBSCRIPTS

b	bubble point
est	estimated from correlation
exp	experimental
g	gas
o	oil
s	solution

CHAPTER 1
INTRODUCTION

1.1 Background of Study

In the oil and gas industry, especially in reservoir studies, reservoir fluid or pressure-volume-temperature (PVT) properties are very important in the determination of reservoir performance and the calculation of its reserve. Reservoir fluid properties are always required in order to perform petroleum engineering calculations such as estimation of hydrocarbon properties, the in-place volumes and transport parameters (Dindoruk & Christman, 2001). One of the important reservoir fluid properties of primary interest in petroleum engineering studies is crude oil density.

Crude oil density, ρ_o is one of the most important oil properties as its value impacts the calculations of oil volume (Ahmed, 2007).

According to Ahmed (2007),

> The crude oil density is defined as the mass of a unit volume of the crude at a specified pressure and temperature, mass/volume. The density usually is expressed in pounds per cubic foot and it varies from 30 lb/ft^3 for light volatile oil to 60 lb/ft^3 for heavy crude oil with little or no gas solubility.

Ideally, crude oil density is experimentally measured in the laboratory. However, it is very expensive in predicting this property at laboratory. The accuracy of the prediction is critical and sometimes not known in advance (Nagi *et al*, 2009). Therefore, when the crude oil density measurements are not available, PVT correlations from the literature are often used.

There are many PVT correlations that have been proposed in order to determine the crude oil density. The correlations are divided into 2 categories: correlations that use the

crude oil composition and correlations that use limited PVT data (Ahmed, 2007). However, in this study, the author focuses on correlations that use limited PVT data only.

The example of PVT correlations widely used in determining the crude oil density is Standing and Katz (Ahmed, 2007). However, limitations concerning the validity of the correlations for different types of hydrocarbon systems, accuracy and range of applicability have been controversial (Elsharkawy *et al*, 1995).

So, in order to improve the accuracy and validity of PVT correlations, the researchers are struggling to come out with new ideas on the correlations by using different approaches. Some of the approaches been done are neural networks, regression analysis and graphical networks.

Recently, a modeling tool called group method of data handling (GMDH) approach has been introduced in oil and gas industry. GMDH is an inductive modeling method built on the principles of self-organization. This modeling approach has been used widely in many areas such as medical diagnostics, weather modeling, marketing and environment systems (Osman & Abdel-Aal, 2002).

In last 35 years, GMDH is developing as a method of inductive modeling and forecasting of complex systems (Godefroy *et al*, 2012). Therefore, GMDH modeling approach has been proposed as an alternative modeling tool to predict the PVT properties which can avoid the limitations of the existing PVT correlations.

1.2 Problem Statement

Although the existing PVT correlations are widely used in oil and gas industry, there are some problems arise when dealing with it. One of the main problems is **accuracy of the existing correlations.**

The developed PVT correlations have some limitations on its accuracy and are suitable only for certain types of hydrocarbon systems. Crude oil from different regions has different properties. The PVT correlations were originally developed for some range of reservoir fluid characteristics and geographical area with similar fluid composition.

Therefore, the accuracy of the correlations is critical and the suitability of those correlations must be verified before it is used for PVT predictions.

Another problem regarding PVT correlations is **limitations of available data.** The most important parameters usually taken into account before using the PVT correlations are API gravity, reservoir temperature and gas-oil ratio. However, some fields might not have enough data to be measured and analyzed in the laboratory. Therefore, it will be more difficult in determining PVT properties using correlations when the fields don't have many/enough data.

1.2.1 Problem Identification

The problems identified are:

a) Difficulty to decide which correlations have the best accuracy
b) Limitations of available data from the field
c) The validity of the correlations

1.3 Objectives

The main objectives of this study are:

a) To study the capability of GMDH in modeling crude oil density.
b) To reduce the number of correlating parameters that needed in the PVT correlations.
c) To compare the performance of this GMDH modeling approach with the existing correlations.

1.4 Scope of Study

The scope of study is mainly to model a new correlation for crude oil density. The new correlation is modeled by using MATLAB software. The study is divided into two stages; the first stage involves the modeling of the correlation associated with programming and graphic visualizations. GMDH algorithm is developed by MATLAB during this stage. After GMDH algorithm is successfully done, the new correlation will be obtained. The second stage focuses on testing the accuracy of the new modeled correlation. Moreover, its performance also will be compared with the existing correlations.

This project involves the understanding and ability to develop mathematical model from MATLAB and also involves the understanding in PVT properties and correlations. Proper understanding in these two topics is important in order to keep the project work on the right track.

1.5 Relevancy of the Project

This project is relevant to the author's field of study since PVT properties and correlations is one of the most important areas in petroleum engineering. PVT correlations topic is fall under reservoir engineering disciplinary where reservoir engineers are still doing research on how to improve the capability of the existing correlations.

In this project, the author has to deal with MATLAB programming to develop a mathematical model and GMDH algorithm for crude oil density. Although the author's knowledge in MATLAB programming is still new, it is not a major hurdle as long as the author is determined and keeps on learning and doing the research on MATLAB.

1.6 Feasibility of the Project

The project is feasible since it is within the scope and time frame. The author has completed the research and literature review by the end of the first semester. Moreover, the author also has done some tutorials on MATLAB to get to know more about its programming. By the end of Final Year Project I (FYP I) period, the author is completely clear about the PVT properties, the mechanism of GMDH and the programming behind MATLAB. For the second semester (i.e Final Year Project II), the author has started doing the programming for GMDH algorithm. Eventually, the author has successfully modeled a new crude oil density within the stipulated time frame.

CHAPTER 2
LITERATURE REVIEW AND THEORY

2.1 PVT Properties and Its Importance

In the oil and gas industry, especially in reservoir studies, reservoir fluid characterization is vital for developing a strategy to manage the reservoir production scheme effectively (Godefroy *et al*, 2012). PVT properties (e.g bubble-point pressure, formation volume factor, gas-oil ratio, oil density and oil viscosity) are very important and are always required in order to perform petroleum engineering computations such as estimation of hydrocarbon properties, the in-place volumes calculations, transport parameters, reservoir simulation, design of production equipment, oil and gas recovery estimation, material balance and well test analysis (Dindoruk & Christman, 2001; Elsharkawy, 1998; Godefroy *et al*, 2012; Nagi *et al*, 2009).

PVT properties data can be obtained by conducting a laboratory study. These data also can be estimated from empirical correlations. Although laboratory results are better in terms of high accuracy where reservoir conditions can be controlled, the results are dependent on the validity of the reservoir fluid samples, especially when the reservoir pressure has decreased below the bubble-point pressure (Omar & Todd, 1993).

In situations where the experimental data cannot be obtained, or the laboratory results must be cross checked, or no fluid samples are available, one must rely on empirical correlations. In past few decades, more than 30 empirical correlations have been published as other alternatives to estimate and predict PVT properties (Godefroy *et al*, 2012).

2.2 PVT Correlations and Its Limitations

PVT properties can be predicted by using empirically derived correlations from the literature. Some of the well-known empirical correlations are Standing, Vasquez and Beggs, Lasater, Petrosky and Farshad, McCain, Al-Marhoun, Glaso and Labedi (Ahmed, 2007; Godefroy *et al*, 2012).

One of the most important PVT properties is crude oil density. There are several correlations available to determine the saturated crude oil density (at or below bubble-point pressure) such as correlation by Standing and Katz and also correlation by Standing (Ahmed, 2007).

Ahmed (2007) reported that Katz introduced apparent liquid density of the dissolved gas, ρ_{ga} at 14.7 psia and 60°F and correlated it with solution gas-oil ratio, R_s, gas specific gravity, γ_g and stock-tank oil specific gravity (or API gravity).

$$\rho_{sc} = \frac{350.376\gamma_o + \left(\frac{R_s\gamma_g}{13.1}\right)}{5.615 + \left(\frac{R_s\gamma_g}{13.1\rho_{ga}}\right)} \tag{2.1}$$

$$\rho_{ga} = (38.52)10^{-0.00326API} + [94.75 - 33.93\ \log(API)]\log(\gamma_g) \tag{2.2}$$

where

γ_g = gas specific gravity (air = 1.0)

γ_o = stock-tank oil specific gravity

R_s = solution gas-oil ratio, scf/STB

ρ_o = oil density at standard condition, lb/ft^3

ρ_{ga} = apparent liquid density of the dissolved gas, lb/ft^3

Ahmed (2007) also reported that Standing expressed the crude oil density as a function of R_s, API gravity, γ_g and the system temperature, T. No composition of the oil is

required for both correlations. The density of a crude oil at a specified pressure and temperature can be calculated from the following equation:

$$\rho_o = \frac{62.4\gamma_o + 0.0136 R_s \gamma_g}{0.972 + 0.000147\left[R_s\left(\frac{\gamma_g}{\gamma_o}\right)^{0.5} + 1.25(T - 460)\right]^{1.175}} \quad (2.3)$$

where

T = system temperature, ^{o}R

ρ_o = oil density, lb/ft^3

However, the success of the existing empirical correlations is sometimes controversial as they depend on the range of data at which they were originally developed.

There are also limitations concerning the validity of the correlations for different types of hydrocarbon systems, accuracy, non-hydrocarbon content and range of applicability (Elsharkawy *et al*, 1995). These correlations were developed by using linear, nonlinear, multiple regression or graphical techniques. A regression model imposes a given form for the relation between independent and dependent variables. Modern learning algorithm techniques overcome some of the limitations of regression analysis.

Neural network as the alternative to regression analysis have been proposed. In general, artificial neural networks have been proposed in solving many problems in the industry, such as seismic pattern recognition, permeability and porosity prediction, prediction of PVT properties and estimating pressure drop in pipes and wells (Osman & Abdel-Aal, 2002).

However, still, this modeling technique has some limitations which are long training times, the complexity of the design space, over-fitting or poor network generalization with new data during actual use and the opacity or black-box nature of the models (Osman & Abdel-Aal, 2002; Abdel-Aal *et al,* 1997).

2.3 Group Method of Data Handling (GMDH)

So, self-organizing group method of data handling (GMDH) is introduced in the petroleum industry as an alternative modeling approach that helps to overcome the above limitations. GMDH combines the advantages of neural networks with those of advanced statistical methods to provide a faster, easier to use and more accurate modeling tool (Abdel-Aal *et al*, 1997).

GMDH is an inductive learning algorithm for complex processes and systems modeling (GMDH Applications). It was invented in the late 1960s by Prof. Alexey Grigorevich Ivahnenko, an academician from the Ukrainian Academy of Sciences, Institute of Cybernetics, Ukraine.

GMDH or also known as polynomial neural networks, abductive and statistical learning networks is an algorithm modeling tool for identifying nonlinear relations between input and output variables (Oh & Pedrycz, 2002). GMDH algorithm can be represented as set of neurons in which different pairs of neurons in each layer are connected through a quadratic polynomial, and later produce new neurons in the next layer (Ma *et al*, 2009).

GMDH works by building consecutive layers with link. The layers are simple polynomial terms which are created by using linear and nonlinear regressions. The first layer is built by computing regressions of the input variables and then choosing the best ones. The second layer is made by computing regressions of the values in the first layer along with the input variables. This process continues until the net stops getting better (Ward Systems Group Inc., 2008).

The problem is to find a function *f* so that can be approximately used to predict output Y for a given input vector $x = (x_1, x_2, x_3, .. , x_n)$ as close as possible to its actual input Y. Assume the output variable Y is a function of the input variables x:

$$Y = f(x_1, x_2, x_3, .. , x_n) \tag{2.4}$$

Polynomial reference function is used in this multilayered algorithm (Semenov *et al*, 2010) as shown below:

$$F(x) = a_0 + \sum_{i=1}^{d} a_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} a_{ij}\, x_i\, x_j + \ \ldots \qquad (2.5)$$

The above function can simulate the input output perfectly and it has been used as a complete description of the system model. By combining the partial polynomial of two variables in the multilayers, the GMDH algorithm can solve the problems.

By this self-organizing method, inaccurate, small and noisy data will be removed, thus the accuracy of the model is higher and the structure also is simpler than structure of usual physical model. The workflow of this GMDH algorithm is presented on **Figure 2.1**:

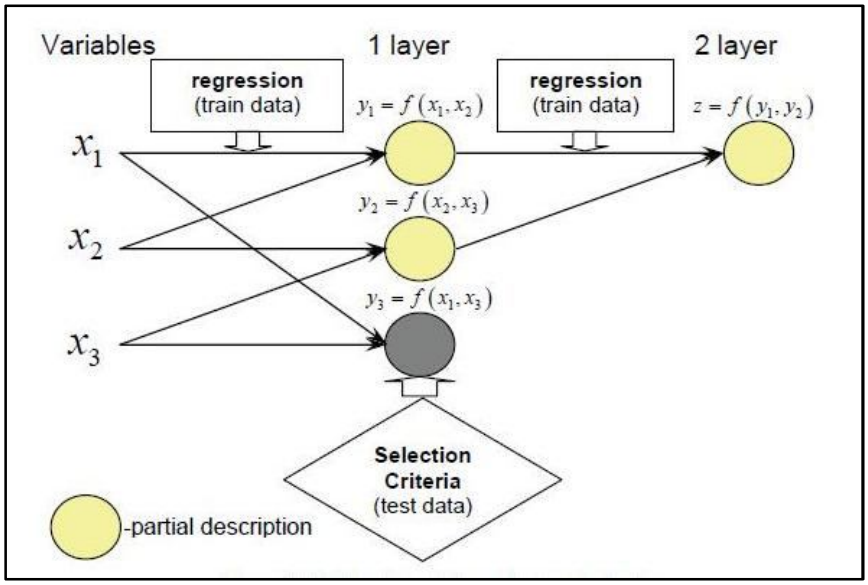

Figure 2.1: GMDH network structure (Semenov *et al*, 2010)

2.4 Applications of GMDH in the Oil and Gas Industry

GMDH is now technically and practically used in various applied fields such as economic systems, ecological systems, demographic systems, econometric modeling and military systems (GMDH Applications).

However, this approach is rarely being practiced in the petroleum and gas industry. A search has revealed only a few studies have been done using GMDH modeling such as the prediction of tool life in drilling (Lee *et al*, 1995), the prediction PVT properties (Osman & Abdel-Aal, 2002), the prediction of permeability from well logs (Lim *et al*, 2006) and the improvement of porosity prediction (Semenov *et al*, 2010).

Lee *et al* (1995) presented an abductive network for predicting tool life in drilling operations. The abductive network consists of several functional nodes which later on, were represented by drill diameter, cutting speed and feed rate. By these three (3) functional elements, tool life can be predicted.

Based on the experimental results, the abductive network presented by (Lee *et al*, 1995) can be effectively used to predict drill life under varying cutting conditions, and moreover, the prediction error is less than 9%.

Semenov *et al* (2010) had introduced an application of GMDH for geological modeling of Vankor Field. A study had been conducted at Dolgan, a gas-and-water saturated formation of Vankor Field and the objective was to develop the best mathematical model for Dolgan reservoir rock characteristics estimation using all available well logs information.

Dolgan reservoir gamma ray log cannot be applied for porosity interpretation because the sandstones consist of potash feldspar by 30-40%, thus some parts of the reservoir have high radioactivity. The conventional methods (e.g Willie equation and Fomenko equation) cannot resolve the petrophysical relations.

Therefore, Semenov *et al* (2010) came out with a solution by using GMDH modeling to develop the best prediction model for porosity. Correlation coefficient was chosen by the authors as a statistical feature for the evaluation criteria. The closer the correlation coefficient to 100%, the better the model is.

GMDH shows the highest core data correlation coefficient of 38% (resistivity, neutron and density logs were used), outperforming other two models; the linear regression model (spontaneous potential log was used with a correlation coefficient of 24%) and the conventional neural network model (spontaneous potential, neutron and density logs were used with a correlation coefficient of 27%).

2.5 Applications of GMDH for PVT Properties Prediction

In order to improve the accuracy of existing PVT correlations, researchers are struggling to come out with new ideas by using different techniques and one of them is by using GMDH modeling.

Osman and Abdel-Aal (2002) had successfully proved the capability of abductive networks based on the using of GMDH modeling approach for predicting PVT properties. Bubble-point pressure (P_b) model and bubble-point oil formation volume factor (B_{ob}) model have been successfully developed.

The total of 283 data records from different fields were used for this work. 198 out of 283 data points were randomly selected to train each model and another 85 data points were used to test the model to evaluate its accuracy.

2.5.1 Bubble-Point Pressure (P_b) Model

Osman and Abdel-Aal (2002) successfully developed P_b model by using four (4) correlating parameters including reservoir temperature (T_f), solution gas-oil ratio (R_s), gas gravity (γ_g) and API gravity. **Figure 2.2** illustrates the structure of the model and the equations of the functional elements:

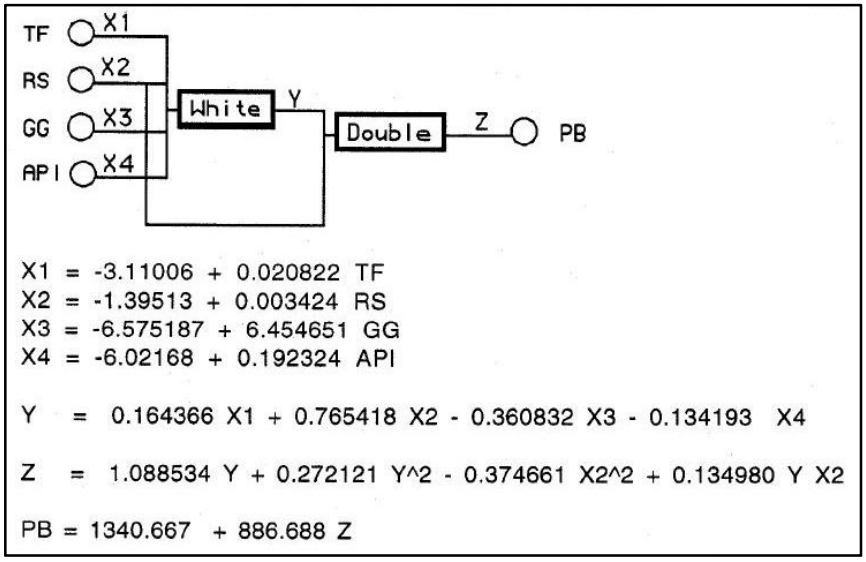

Figure 2.2: Abductive network model for P_b (Osman & Abdel-Aal, 2002)

Two (2) statistical features were chosen by the authors as the evaluation criteria; those are coefficient correlation and average absolute percentage error (AAPE). From this model, the coefficient correlation is 98.98% and the AAPE is 5.62%. When comparing with other empirical correlations, it was proved that this abductive network model outperforms all other correlations because those other correlations give approximately the AAPE of 13%.

2.5.2 Bubble-Point Oil Formation Volume Factor (B_{ob}) Model

Another abductive model developed by Osman and Abdel-Aal (2002) is B_{ob} model. **Figure 2.3** shows the structure of the model and the equations of the functional elements:

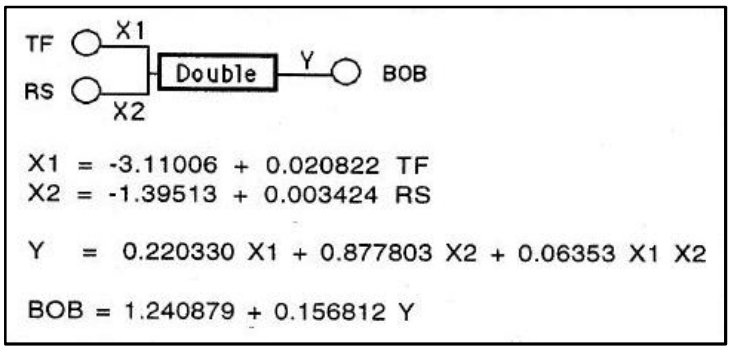

Figure 2.3: Abductive network model for B_{ob} (Osman & Abdel-Aal, 2002)

The B_{ob} model was found to be a function of only T_f and R_s. The correlation coefficient is 99.59% and the AAPE is 0.86% and usually, the AAPE value varies in the range of 1-2% for other empirical correlations.

2.6 Summary

PVT properties is very important for petroleum engineering industry, as these properties are used in performing volumetric calculations, material balance, EOR, reservoir simulations and others. PVT properties like crude oil density are basically reported determined from the field data.

However, in some cases, where PVT measurements from the laboratory are not available due to some problems like high cost for equipment, or the measured data are not so reliable to be used, PVT correlations are the best solution. PVT correlations are mathematical expressions and plots that have been used in reservoir engineering. There are almost 30 PVT correlations developed by researchers to determine the PVT properties including oil density. Some of the correlations available for oil density are Katz and Standing.

However, limitations of the existing correlations on accuracy, validity, range of applicability, data available and on-hydrocarbon content have been controversial. Researchers are still working on the development of the correlations that are very good in accuracy, validity and applicable for oil types of hydrocarbon systems.

Recently, GMDH is introduced into oil and gas industry. GMDH or polynomial network is an algorithm modeling tool for identifying nonlinear relations between input and output. Some studies have been conducted on using GMDH modeling approach to model the PVT properties and petrophysical properties.

Osman & Abdel-Aal (2002) proposed an abductive network based on GMDH technique to predict bubble-point pressure and bubble-point oil formation volume factor. The results showed that GMDH modeling outperforms other empirical correlations. Since GMDH is proven to be successful in predicting P_b and B_{ob}, it is suggested that GMDH modeling is being used for other PVT properties, generally in other areas of petroleum engineering.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Project Workflow

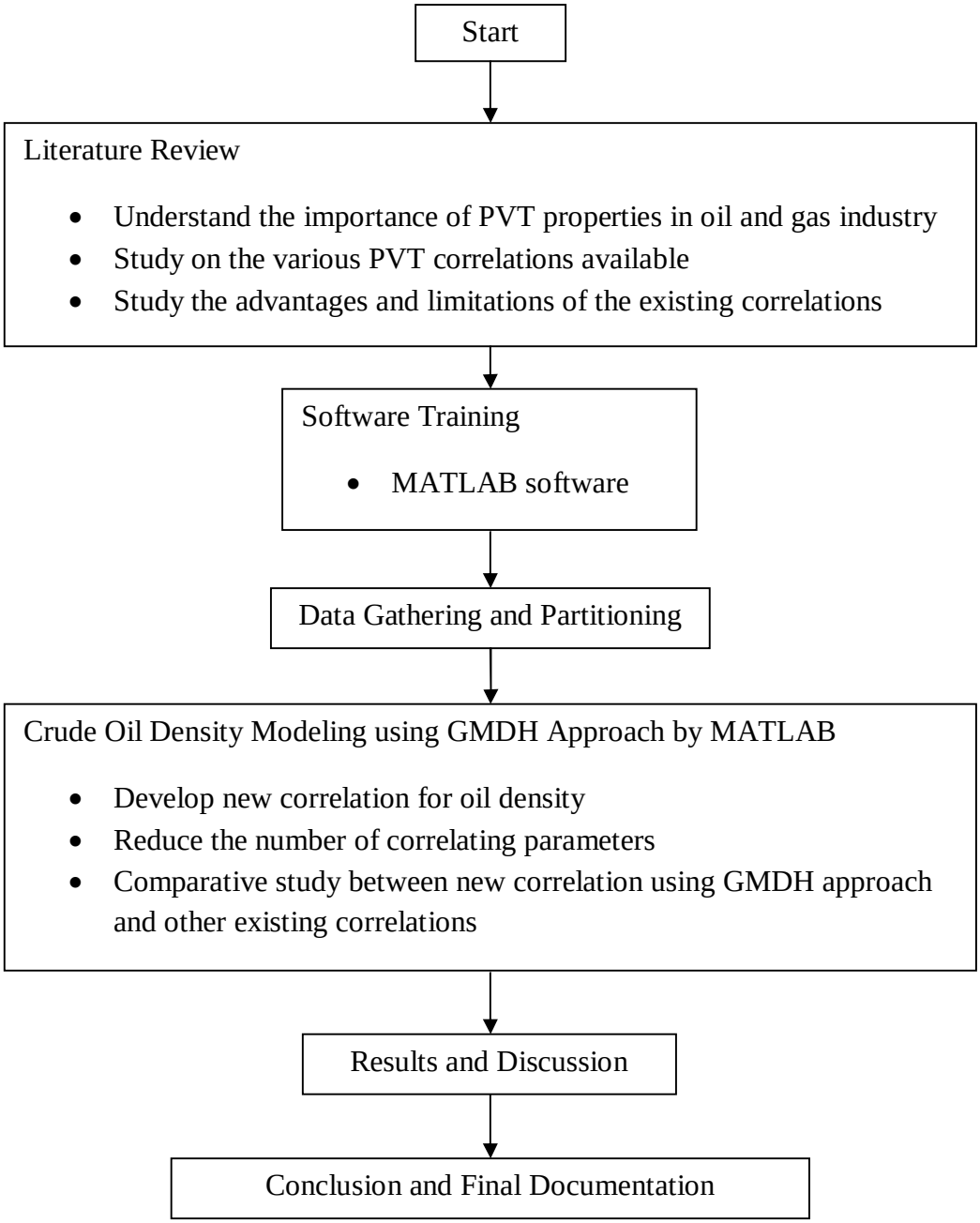

Figure 3.1: Project workflow

3.2 Project Gantt Chart and Key Milestone

Table 3.1: Project Gantt chart (First semester)

Detail/Week	1	2	3	4	5	6	7		8	9	10	11	12	13	14
Selection of Project Topic								Mid-Semester Break							
Prelim. Research Work/Lit. Review															
Submission of Extended Proposal							○								
Software Training															
Proposal Defense															
Software Training + Project Continues															
Submission of Interim Report															○

Table 3.2: Project Gantt chart (Second semester)

Detail/Week	1	2	3	4	5	6	7		8	9	10	11	12	13	14
Project Work Continues								Mid-Semester Break							
Submission of Progress Report									○						
Project Work Continues															
Submission of Draft Report												○			
Submission of Technical Paper												○			
Pre-SEDEX														○	
Oral Presentation															○
Submission of Project Dissertation															○

3.3 Data Gathering and Partitioning for GMDH Model

A total of 290 data sets were collected from three (3) fields (i.e Malaysian, UAE and Middle East oil fields). These data has been utilized for the generation of the GMDH model. Range of collected data is presented in **Table 3.3** below:

Table 3.3: Range of collected data

Bubble-point pressure, psia	147 - 3851
Bubble-point oil formation volume factor, bbl/STB	1.032 – 1.997
Solution gas-oil ratio, scf/STB	26 - 1602
Gas specific gravity (air=1)	0.627 - 1.367
API gravity, ^{o}API	19.4 – 50.5
Reservoir temperature, ^{o}F	80 - 254

Relevant input parameters were selected based on the most commonly used empirical correlations in the industry. Six (6) parameters were thought to have a strong effect on the prediction of crude oil density: bubble-point pressure, bubble-point oil formation volume factor, solution gas-oil ratio, gas specific gravity, API gravity and reservoir temperature.

Partitioning the data is the process of dividing the data into three (3) different sets: training set, validation set and testing set. The training set is used to develop the model; the validation set is used to ensure the optimum generation of the developed model and the testing set is used to examine the final performance of the model.

A partitioning ratio of 2:1:1 is used in this study. This corresponds to one half of the data (144 data points) are used for training; one quarter (73 data points) are used for validation and another one quarter (73 data points) are used for testing the new model performance.

3.4 GMDH Algorithm Workflow

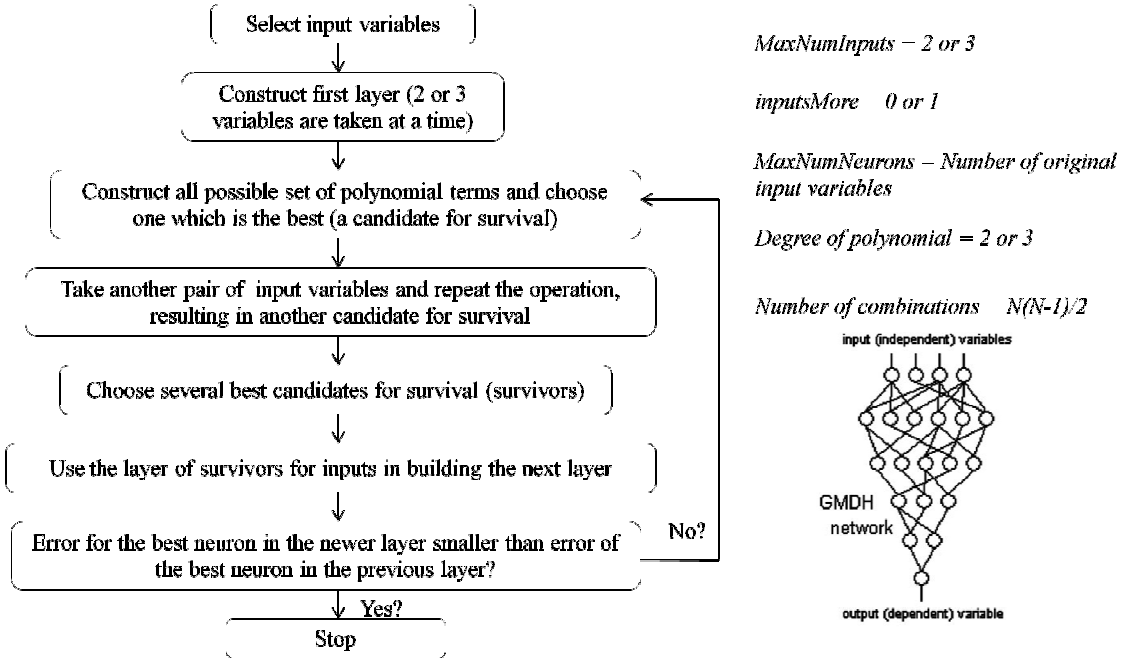

Figure 3.2: GMDH algorithm workflow

3.5 Software Used

MATLAB software (version R2009b) is a high-level language and interactive environment for numerical computation, visualization and programming (The MathWorks Inc., 2013). The author can analyze data, develop algorithms and create models by using MATLAB software.

Other software that is used in this study is summarized in **Table 3.4** below:

Table 3.4: Summary of software used

Tool	Function
MATLAB software	To develop GMDH modeling approach for new oil density correlation
Microsoft Office Word	To write reports, data etc
Microsoft Office Excel	To prepare data sheets and calculations
Microsoft Office Power Point	To prepare presentations

3.6 Model Efficiency Evaluation

The model efficiency evaluation will be conducted by statistical error analysis and graphical error analysis.

3.6.1 Statistical Error Analysis

There are five (5) main statistical parameters that are being considered in this study: average percentage relative error, average absolute percentage relative error, standard deviation, correlation coefficient and coefficient of determination. These parameters will be used to help in evaluating the accuracy of the predicted crude oil density correlations. Those parameters are well known for their capabilities to analyze models' performances and they have been utilized by several authors, (Hemmati & Kharrat, 2007) and (Omar & Todd, 1993).

3.6.1.1 Average Percentage Relative Error (APRE)

This is an indication of the relative deviation in percent from the experimental values. It is given as:

$$E_r = \left(\frac{1}{n_d}\right)\sum_{i=1}^{n_d} E_i \quad (3.1)$$

E_i is the relative deviation in percent of an estimated value from an experimental value. It is defined as:

$$E_i = \left[\frac{(x_{est} - x_{exp})}{x_{exp}}\right]_i \times 100, \quad i = 1,2..n_d \quad (3.2)$$

where x_{est} represents the estimated values while x_{exp} represents the experimental values. The lower the value of E_r, the more equally distributed is the errors between positive and negative values.

3.6.1.2 Average Absolute Percentage Relative Error (AAPRE)

It measures the average value of the absolute relative deviation of the measured value from the experimental data. This value is expressed in percent. It is defined as:

$$E_a = \left(\frac{1}{n_d}\right)\sum_{i=1}^{n_d} |E_i| \quad (3.3)$$

It indicates the relative absolute deviation from the experimental values. The lower the value of AAPRE, the better the agreement between the estimated and experimental values.

3.6.1.3 Minimum and Maximum Absolute Percentage Relative Error

The range of error is determined by the values of APRE, where the highest and the lowest values are identified.

$$E_{min} = min_{i=1}^{n_d} |E_i| \tag{3.4}$$

$$E_{max} = max_{i=1}^{n_d} |E_i| \tag{3.5}$$

A higher accuracy is achieved when the maximum value is small.

3.6.1.4 Standard Deviation (SD)

Standard deviation, SD is a measure of dispersion. It is expressed as:

$$SD = \left[\frac{1}{(n_d - 1)}\right] \sum_{i=1}^{n_d} E_i^2 \tag{3.6}$$

The lower value of standard deviation, the smaller degree of scatter, thus the accuracy is higher.

3.6.1.5 Correlation Coefficient (R) and Coefficient of Determination (R^2)

Correlation coefficient, R describes the extent of the association between experimental and calculated values.

$$R = \sqrt{1 - \frac{\sum_{i=1}^{m}\left[\left(x_{exp} - x_{est}\right)_i\right]^2}{\sum_{i=1}^{m}\left[\left(x_{exp} - \bar{x}\right)_i\right]^2}} \tag{3.7}$$

where

$$\bar{x} = \left(\frac{1}{n_d}\right)\sum_{i=1}^{n_d}\left(x_{exp}\right)_i \tag{3.8}$$

The value of R varies from -1.0 to +1.0. A coefficient of zero indicates no relationship between experimental and calculated values. A +1.0 coefficient indicates a perfect positive relationship while a -1.0 coefficient indicates a perfect negative relationship.

Coefficient of determination, R^2 is the square value of correlation coefficient. It is defined as the proportion of the validity in the predicted values that is encountered for by the experimental values.

3.6.2 Graphical Error Analysis

Graphical error analysis helps in visualizing the accuracy of a correlation. In this study, one (1) graphical analysis technique will be used: cross plot.

All the calculated values are plotted against the experimental values. Thus, a cross plot is formed. A 45^{o} straight line is drawn on the cross plot on which the calculated value is equal to the experimental value. The closer the plotted data points are to this line, the better the correlation.

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Data Availability

In this study, 290 PVT data points from three (3) different oil fields were used, which are Malaysian oil fields (Omar & Todd, 1993), UAE fields (Dokla & Osman, 1990) and Middle East fields (Al-Marhoun, 1998). The reasons why these data were chosen are these 3 fields produced crude oil in nature and its availability of complete PVT reports for further evaluation and development of PVT correlations. 89 data points from 38 Malaysian oil fields, 43 data points from 51 bottom hole sample of UAE reservoirs and 158 data points from 69 Middle East oil fields were used for this study. The range of data is summarized in **Table 4.1** below and the list of all PVT data points is available in the **Appendix I**.

Table 4.1: Range of data from 3 field regions

	Malaysia		Middle East		UAE	
	Min	Max	Min	Max	Min	Max
P_b (psia)	790	3851	147	3573	601	3840
B_{ob} (bbl/STB)	1.092	1.954	1.032	1.997	1.216	1.946
R_s (scf/STB)	142	1440	26	1602	209	1408
γ_g (air=1)	0.628	1.315	0.752	1.367	0.798	1.29
API (^{o})	29.1	50.4	19.4	44.6	31.2	40.3
Temperature (^{o}F)	127	250	80	240	212	254

4.2 The Study of Capability of GMDH in Modeling Crude Oil Density

The list of GMDH programs and codes generated or modified by the author is available in the **Appendix II**.

All six (6) correlating parameters were used to study the capability of GMDH in modeling the oil density. After several runs by MATLAB, it was found that the desired correlating parameters to determine the oil density are B_{ob}, R_s, γ_g and API. The summary of the best 10 results are shown in **Table 4.2**:

Table 4.2: Summary of results (1)

Run	Layers	Desired Parameters						AAPRE (%)
		P_b	B_{ob}	R_s	γ_g	API	Temp	
1	3		√	√	√	√		0.39
2	4		√	√	√	√		1.09
3	6	√	√	√	√	√		0.18
4	4		√	√	√	√		0.84
5	8		√	√	√	√		6.28
6	6		√	√	√	√		0.15
7	6	√	√	√	√	√		0.10
8	6		√	√	√	√		0.07
9	5		√	√	√	√		0.09
10	6		√	√	√	√		0.08

From the table above, it is shown that GMDH is capable to model the oil density by giving low absolute percentage relative error as low as 0.07%. From these runs, it is found that Run 9 gives the best result as the layers or equation generated by MATLAB is 5 with the AAPRE of 0.09%.

The diagram, equations and coefficients generated from Run 9 are illustrated in **Figure 4.1**, **Equation 4.1** and **Table 4.3** below:

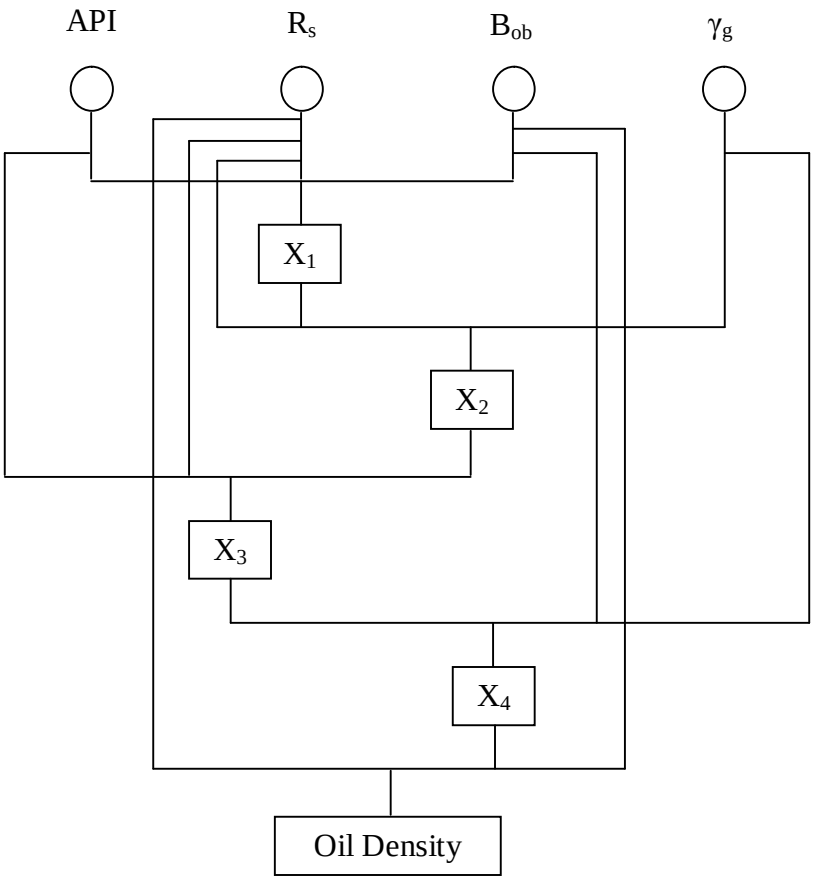

Figure 4.1: GMDH network structure for the new oil density model (1)

$$X_1 = a_1 + a_2API + a_3R_s + a_4B_{ob} + a_5R_sAPI + a_6B_{ob}API + a_7B_{ob}R_s + a_8(API)^2 + a_9(R_s)^2 + a_{10}(B_{ob})^2$$

$$X_2 = b_1 + b_2X_1 + b_3\gamma_g + b_4R_s + b_5\ X_1\gamma_g + b_6\ X_1R_s + b_7R_s\gamma_g + b_8(X_1)^2 + b_9(\gamma_g)^2 + b_{10}(R_s)^2$$

$$X_3 = c_1 + c_2X_2 + c_3API + c_4R_s + c_5X_2API + c_6X_2R_s + c_7R_sAPI + c_8(X_2)^2 + c_9(API)^2 + c_{10}(R_s)^2$$

$$X_4 = d_1 + d_2X_3 + d_3\gamma_g + d_4B_{ob} + d_5\ X_3\gamma_g + d_6X_3B_{ob} + d_7B_{ob}\gamma_g + d_8(X_3)^2 + d_9(\gamma_g)^2 + d_{10}(B_{ob})^2$$

$$\rho_{ob} = y_1 + y_2X_4 + y_3R_s + y_4B_{ob} + y_5X_4R_s + y_6X_4B_{ob} + y_7B_{ob}R_s + y_8(X_4)^2 + y_9(R_s)^2 + y_{10}(B_{ob})^2 \quad (4.1)$$

Table 4.3: Coefficients for the new oil density model (1)

a_1	121.179602744772
a_2	-0.227272134044463
a_3	0.0181272876170419
a_4	-82.8099268150002
a_5	0.000172426679241986
a_6	0.144135493161753
a_7	-0.0191686348658074
a_8	-0.00439176423174856
a_9	$6.39706982913085 \times 10^{-6}$
a_{10}	22.2552872667849

b_1	18.770194916396
b_2	0.26940382241424
b_3	-7.03965342672555
b_4	-0.00759462126436576
b_5	0.13484213332
b_6	$2.15190193348182 \times 10^{-5}$
b_7	0.00941334638446587
b_8	0.00699148204780506
b_9	-0.00336572263020571
b_{10}	$-6.65482604780846 \times 10^{-7}$

c_1	8.78871028187614
c_2	0.805752729447875
c_3	-0.853480306484748
c_4	0.021286948075414
c_5	0.0100027693895368
c_6	-0.000257162963189024
c_7	-0.000185406777991506
c_8	0.00145462972781034
c_9	0.0080285292657463
c_{10}	$-1.75575998048839 \times 10^{-6}$

d_1	8.15137320777751
d_2	0.540671491854118
d_3	9.34032345581186
d_4	-3.81785815662668
d_5	-0.123458375662134
d_6	0.120763196956461
d_7	-2.4772985824412
d_8	0.00457549430353984
d_9	0.00688563899256617
d_{10}	0.36111425366507

y_1	-51.2603461278923
y_2	1.60121001078334
y_3	-0.037807875417156
y_4	71.5989096914056
y_5	0.000260407365771556
y_6	-0.452923214065124
y_7	0.0241655435335066
y_8	-0.00147360975332288
y_9	$-5.75100228657471 \times 10^{-6}$
y_{10}	-24.0059003601657

The statistical error analysis for this section is summarized in **Table 4.4** and a cross plot is illustrated in **Figure 4.2**:

Table 4.4: Statistical error analysis of the new oil density model (1)

	This Study
AAPRE (%)	0.09
Min. APRE (%)	0.001
Max. APRE (%)	0.56
Standard Deviation, SD (%)	0.09
Correlation Coefficient, R	0.999
Coefficient of Determination, R^2	0.999

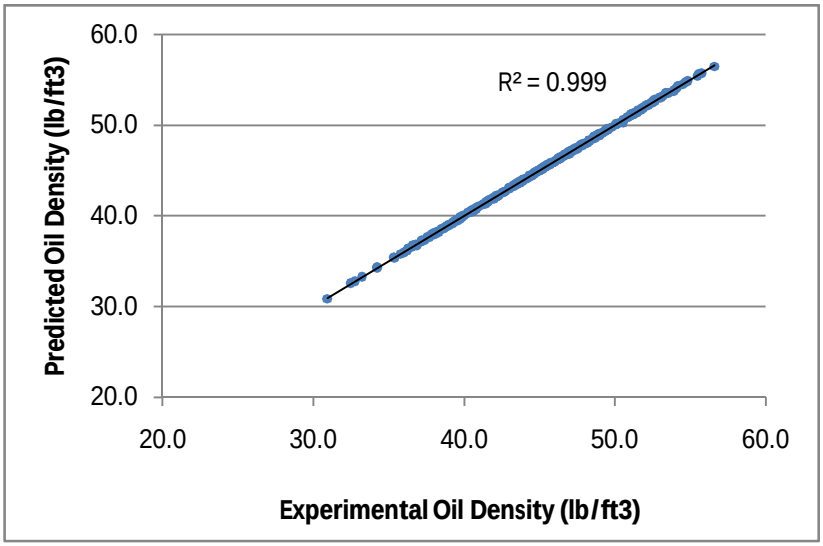

Figure 4.2: Cross plot of the new oil density model (1)

From the results, it is proven that GMDH is capable to model the crude oil density with a very low absolute percentage relative error and a high accuracy. However, the network structure is a little bit complex as it involves 5 layers and it has too many coefficients in order to determine the oil density at bubble-point pressure.

4.3 Reducing the Number of Correlating Parameters

B_{ob}, R_s, γ_g and API were selected as the best correlating parameters to be used in this study. Although the new oil density model shows the best accuracy and the lowest percent error, it involves many layers (equations) and coefficients, and the structure is complex.

It is decided that among these four (4) parameters, some of them should be removed so that the oil density model is simpler than previous one.

Firstly, API was removed then followed by R_s, B_{ob} and γ_g. The summary of the results is shown in **Table 4.5**:

Table 4.5: Summary of results (2)

No.	Layers	Parameters				AAPRE (%)	R^2
		API	R_s	B_{ob}	γ_g		
1	2		√	√	√	1.73	0.962
2	2		√	√	√	1.77	0.961
3	2		√	√	√	1.86	0.958
4	1	√		√	√	1.44	0.973
5	1	√		√	√	1.43	0.973
6	1	√		√	√	1.44	0.973
7	1	√	√		√	3.05	0.894
8	1	√	√		√	3.02	0.891
9	1	√	√		√	2.98	0.893
10	2	√	√	√		1.17	0.983
11	2	√	√	√		1.13	0.981
12	1	√	√	√		1.14	0.982

From the results, the best result for each case is chosen. The summary of the statistical error analysis for all cases are shown in **Table 4.6** below:

Table 4.6: Statistical error analysis for the new oil density model (eliminating one parameter)

	All	All (except API)	All (except R_s)	All (except B_{ob})	**All (except γ_g)**
AAPRE (%)	0.09	1.73	1.43	2.94	**1.14**
Min. APRE (%)	0.001	0.004	0.004	0.009	**0.01**
Max. APRE (%)	0.56	8.34	6.38	11.64	**5.53**
Standard Deviation, SD (%)	0.09	2.20	1.94	3.89	**1.02**
Correlation Coefficient, R	0.999	0.981	0.986	0.945	**0.991**
Coefficient of Determination, R^2	0.999	0.962	0.973	0.893	**0.982**

Based on the statistical error analysis shown above, it is concluded that by eliminating γ_g, GMDH produces the oil density model with the lowest AAPRE (i.e 1.14%), the lowest standard deviation (i.e 1.53%), the maximum APRE (i.e 5.53%) and the highest correlation coefficient (i.e 0.991).

It is concluded that the new oil density model is a function of:

i) Bubble-point oil formation volume factor, B_{ob}

ii) Solution gas-oil ratio, R_s

iii) API gravity, ^oAPI

The diagram, equation and coefficients for the new oil density model are illustrated in **Figure 4.3**, **Figure 4.4**, **Equation 4.2** and **Table 4.7** below:

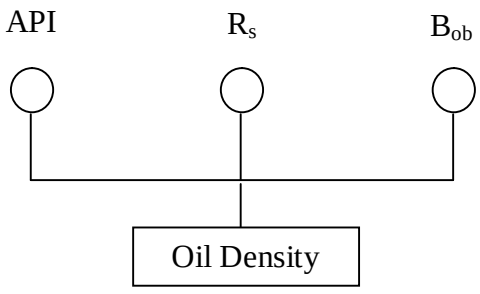

Figure 4.3: New oil density model structure

$$\rho_{ob} = x_1 + x_2API + x_3R_s + x_4B_{ob} + x_5R_sAPI + x_6B_{ob}API + x_7B_{ob}R_s + x_8(API)^2 + x_9(R_s)^2 + x_{10}(B_{ob})^2 \quad (4.2)$$

Table 4.7: Coefficient for the new oil density model (2)

x_1	135.589520128539
x_2	0.456588393494152
x_3	0.0244115554523796
x_4	-122.398701658527
x_5	0.000547574111295958
x_6	-0.819975901365503
x_7	-0.0328725121942872
x_8	0.00157004676268715
x_9	$6.84652618646429 \times 10^{-6}$
x_{10}	51.0210381099486

4.4 A Comparative Study of the Performance of the New Oil Density Model by GMDH with the Existing Correlations

The performance of this new oil density model is now compared with other existing correlations. There are three (3) available correlations for oil density at bubble-point pressure, which are:

i) Standing
ii) Hanafy *et al* (Hanafy *et al*, 1997)
iii) Standing-Katz

The correlating parameters needed before using these correlations are as below:

i) Standing = f (γ_g, γ_o, R_s, T)
ii) Hanafy *et al* = f (B_{ob})
iii) Standing-Katz = f (API, γ_g, γ_o, R_s, P)

The statistical error analysis for the new oil density model and the known correlation is summarized in **Table 4.8** below:

Table 4.8: Statistical error analysis for the new oil density model as compared with the known correlations

	This Study	Standing	Hanafy *et al*	Standing-Katz
AAPRE (%)	**1.14**	2.26	4.49	1.96
Min. APRE (%)	**0.01**	0.003	0.004	0.02
Max. APRE (%)	**5.53**	11.05	14.93	8.74
Standard Deviation, SD (%)	**1.02**	2.06	3.63	1.70
Correlation Coefficient, R	**0.991**	0.971	0.948	0.981
Coefficient of Determination, R^2	**0.982**	0.943	0.898	0.963

From the results, it shows that amongst the correlations, the new oil density correlation by GMDH approach gives the lowest values of AAPRE, standard deviation and maximum APRE of 1.14%, 1.02% and 5.53% respectively. Furthermore, the correlation coefficient of 0.991 produced by the new oil density correlation is close to an ideal value of 1.0.

The cross plots of the experimental against the predicted oil density of all correlations are presented in **Figure 4.4 through 4.7**. The cross plot of the new oil density correlation in **Figure 4.4** shows that most of the data points fall along the 45^{o} line. This is reflected with a good coefficient of determination, R^2 of 0.982. The high value of R^2 indicates a better accuracy of the new correlation in estimating oil density at bubble-point pressure. From the results, it shows that the new correlation predicts oil density at bubble-point pressure better than any other known correlations.

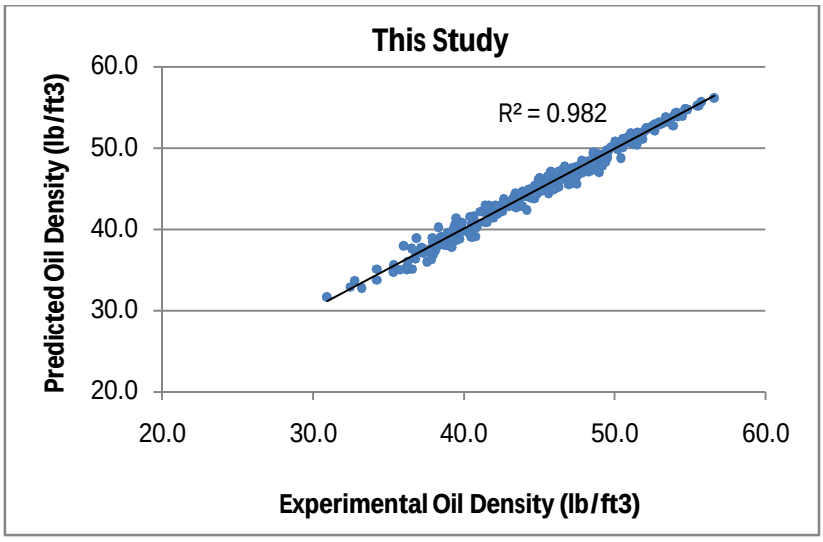

Figure 4.4: Cross plot of the new oil density model (2)

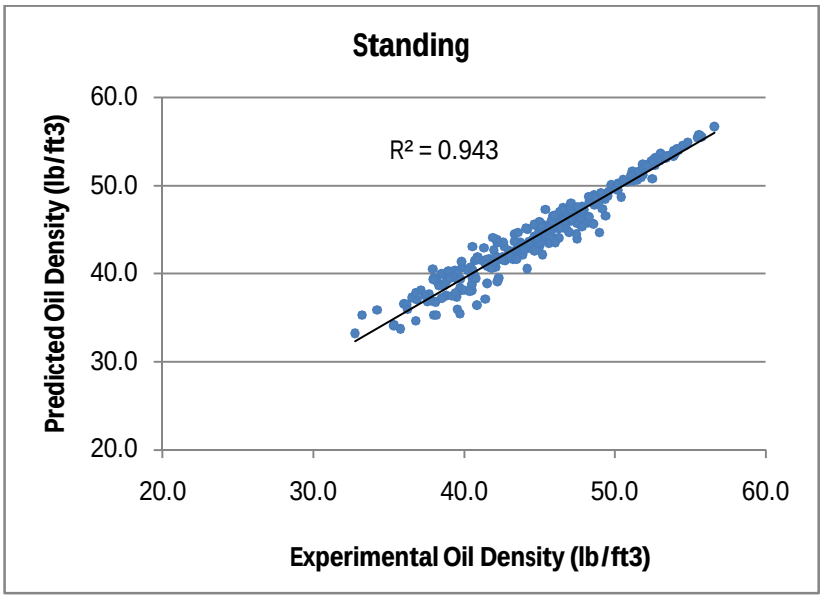

Figure 4.5: Cross plot of Standing's correlation

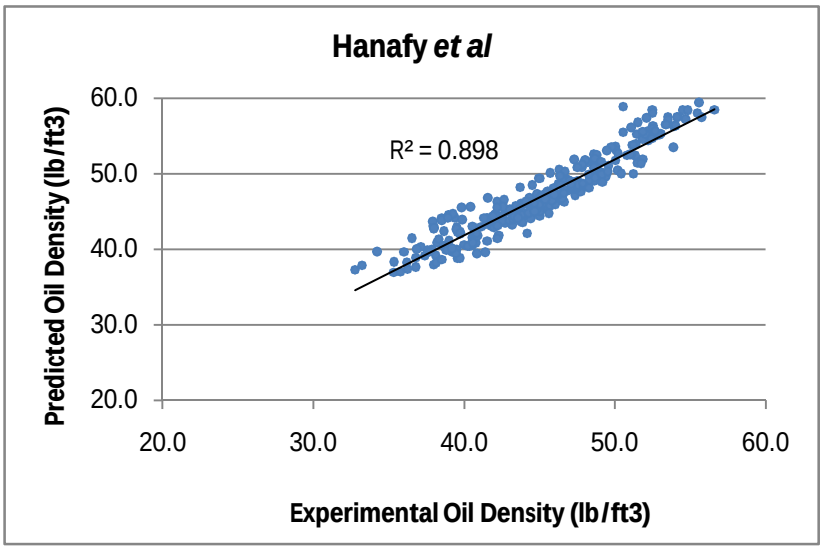

Figure 4.6: Cross plot of Hanafy *et al*'s correlation

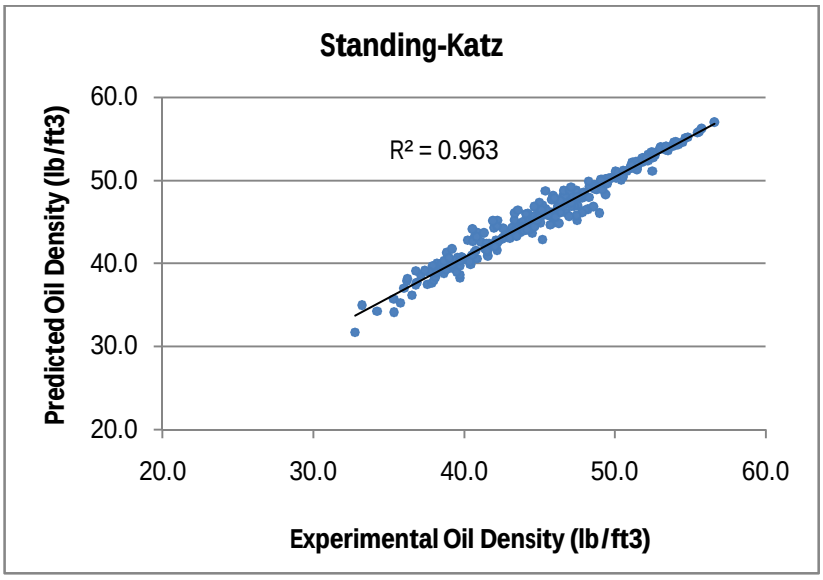

Figure 4.7: Cross plot of Standing-Katz's correlation

The new correlation in estimating oil density at bubble-point pressure also requires only three (3) correlating parameters which are bubble-point oil formation volume factor, solution gas-oil ratio and API gravity. So, it is proven that this new oil density correlation is suitable to be used when there is lack of available PVT data.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATION

Six (6) correlating parameters were used in this study, which are bubble-point pressure, bubble-point oil formation volume factor, solution gas-oil ratio, gas specific gravity, API gravity and reservoir temperature.

GMDH proves that it can model the crude oil density at bubble-point pressure by using these parameters. The best four (4) parameters needed by GMDH to determine oil density are B_{ob}, R_s, γ_g and API gravity. However, the network structure is complex as it involves many layers or equations and coefficients.

So, in order to reduce the complexity of the new modeled oil density, among these 4 correlating parameters, one (1) parameter will be removed.

It is found that by eliminating gas specific gravity, the structure is now simpler which involves only 1 equation and it has less coefficients. Even though the correlating parameters are reduced (where only B_{ob}, R_s and API gravity are used), the new correlation still maintains its lowest percentage error and highest accuracy.

The performance of the new oil density correlation proves that it outperforms all other known correlation in the industry with its lowest percent error and highest accuracy.

Further development and improvement of other PVT properties can be done in the future using GMDH approach.

In conclusion, all the objectives of this study have been achieved successfully.

REFERENCES

Abdel-Aal, R., Al-Nassar, Y., & Al-Garni, A. (1997). Modelling and Forecasting Monthly Electric Energy Consumption in Eastern Saudi Arabia Using Abductive Networks. *Energy Vol. 22, No. 9* , 911-921.

Ahmed, T. (2007). *Equations of State and PVT Analysis.* Houston, TX: Gulf Publishing Company.

Al-Marhoun, M. A. (1998). PVT Correlations for Middle East Crude Oils. *Journal of Petroleum Technology*.

Al-Marhoun, M., & Osman, E. (2002). Using Artificial Neural Networks to Develop New PVT Correlations for Saudi Crude Oils. *SPE 78592 at the 10th Abu Dhabi International Petroleum Exhibition and Conference, October 13-16.* Abu Dhabi, UAE.

Dindoruk, B., & Christman, P. G. (2001). PVT Properties and Viscosity Correlations for Gulf of Mexico Oils. *SPE 71633 at 2001 SPE Annual Technical Conference and Exhibition, September 30-October 3.* New Orleans, Louisiana.

Dokla, M. E., & Osman, M. E. (1990). Correlation of PVT Properties for UAE Crudes. *SPE 21342.* Abu Dhabi National Oil Company/ Society of Petroleum Engineers.

Elsharkawy, A. M. (1998). Modeling the Properties of Crude Oil and Gas Systems Using RBF Network. *SPE 49961 presented at the 1998 SPE Asia Pacific Oil and Gas Conference and Exhibition.* Perth, Australia, October 12-14 1998: Society of Petroleum Engineers.

Elsharkawy, A. M., Elgibaly, A. A., & Alikhan, A. A. (1995). Assessment of the PVT Correlations for Predicting the Properties of Kuwaiti Crude Oils. *Journal of Petroleum Science and Engineering 13* , 219-232.

GMDH Applications. (n.d.). Retrieved April 9, 2013, from GMDH: http://www.gmdh.net/

Godefroy, S., Siew, H. K., & Emms, D. (2012). Comparison and Validation of Theoretical and Empirical Correlations for Black Oil Reservoir Fluid Properties. *OTC 22792 presented at the Offshore Technology Conference.* Houston, Texas, April 30-May 3 2012: Offshore Technology Conference.

Hanafy, H., Macary, S., Elnady, Y., Bayomi, A., & El Batanony, M. (1997). A New Approach for Predicting The Crude Oil Properties. *SPE 37439 presented at the 1997 SPE Production Operations Symposium.* Oklahoma, March 9-11 1997: Society of Petroleum Engineers.

Hemmati, M., & Kharrat, R. (2007). A Correlation Approach for Prediction of Crude-Oil PVT Properties. *SPE 104543 presented ath the 15th SPE Middle East Oil & Gas Show and Conference.* Bahrain, March 11-14 2007: Society of Petroleum Engineers.

Jekabsons, G. (2011). *Open source regression software for Matlab/Octave*. Retrieved April 2013, from Gints Jekabsons: http://www.cs.rtu.lv/jekabsons/regression.html

Lee, B., Liu, H., & Tarng, Y. (1995). An Abductive Network for Predicting Tool Life in Drilling. *IEEE Transactions on Industry Application*.

Lim, J.-S., Park, H.-J., & Kim, J. (2006). A New Neural Network Approach to Reservoir Permeability Estimation from Well Logs. *SPE 100989 presented at the 2006 SPE Asia Pacific Oil & Gas Conference and Exhibition.* Adelaide, Australia, September 11-13 2006: Society of Petroleum Engineers.

Ma, S.-W., Kou, C.-H., Chen, L., & Wang, A.-P. (2009). Application of Group Method of Data Handling to Stream-Way Transition. *2009 International Joint Conference on Artificial Intelligence.* Taiwan.

Nagi, F., Nagi, J., Tiong, S. K., & Ahmed, S. K. (2009). Prediction of PVT Properties In Crude Oil Systems Using Support Vector Machines. *ICEE 2009 3rd International Conference on Energy and Environment.* Malacca, Malaysia, December 7-8 2009.

Oh, S.-K., & Pedrycz, W. (2002). The Design of Self-Organizing Polynomial Neural Networks. *Information Science 141* , 237-258.

Omar, M., & Todd, A. (1993). Development of new Modified Black Oil Correlations for Malaysian Crudes. *SPE 25338 presented at the SPE Asia Pacific Oil and Gas Conference & Exhibition.* Singapore, February 8-10 1993: Society of Petroleum Engineers.

Osman, E., & Abdel-Aal, R. (2002). Abductive Networks: A New Modeling Tool for the Oil and Gas Industry. *SPE 77822 at SPE Asia Pacific Oil and Gas Conference and Exhibition, October 8-10.* Melbourne, Australia.

Semenov, A., Oshmarin, R., Driller, A., & Butakova, A. (2010). Application of Group Method of Data Handling for Geological Modeling of Vankor Field. *SPE 128517 at SPE North Africa Technical Conference and Exhibition, February 14-17.* Cairo, Egypt.

The MathWorks Inc. (2013). *MATLAB: The Lannguage of Technical Computing*. Retrieved April 12, 2013, from MathWorks: http://www.mathworks.com/products/matlab/

Ward Systems Group Inc. (2008). *Network Architecture - GMDH*. Retrieved April 9, 2013, from NeuroShell2 Help: http://www.wardsystems.com/manuals/neuroshell2/

APPENDICES

APPENDIX I

This Appendix is devoted for the list of PVT data used by the author.

ρ_{ob} (lb/ft^3)	P_b (psia)	B_{ob} (bbl/STB)	R_s (scf/STB)	γ_g (air=1)	API (^{o})	T (^{o}F)
41.41	3851	1.466	819	0.663	34.1	243
38.32	3780	1.581	1023	0.658	40.2	209
40.65	3449	1.503	899	0.769	39.3	195
42.09	3440	1.455	863	0.764	37.4	192
36.89	3420	1.683	1212	0.685	42.3	194
39.52	3387	1.505	919	0.673	41.4	194
36.05	3160	1.707	1213	0.705	45.4	186
32.75	3148	1.954	1440	0.788	50.3	250
41.15	3142	1.484	761	0.723	33.3	247
46.16	3063	1.301	577	0.737	31.2	180
45.80	3063	1.287	586	0.628	32.2	180
41.69	2970	1.445	737	0.707	34.6	239
44.94	2692	1.23	393	0.631	38.6	179
36.54	2632	1.578	888	0.73	49.3	228
43.72	2616	1.371	667	0.842	37.3	177
39.54	2611	1.525	810	0.789	39.6	225
40.54	2609	1.622	1019	1.038	40.4	198
39.51	2562	1.491	741	0.795	42	234
33.23	2550	1.884	1170	0.858	48.9	231
34.27	2540	1.712	1020	0.73	50.4	239
35.34	2500	1.843	1355	0.877	48.8	228
43.41	2480	1.357	686	0.737	38.2	171
41.51	2470	1.429	760	0.758	40	166
42.10	2423	1.399	713	0.765	40	169
43.02	2408	1.384	683	0.821	38.6	166
38.19	2402	1.619	844	0.919	40.7	242
39.72	2390	1.538	956	0.811	43.2	226
45.52	2368	1.282	440	0.756	32.5	235
45.19	2360	1.299	694	0.765	40	167
44.35	2350	1.352	680	0.818	37	169
41.54	2344	1.429	791	0.743	40.4	184
43.81	2310	1.345	636	0.801	38.3	161
37.12	2290	1.653	990	0.801	43.1	208
37.96	2274	1.451	546	0.689	45.2	245
40.45	2221	1.362	547	0.693	45.3	238
39.92	2194	1.438	664	0.75	42.9	214
39.38	2193	1.425	634	0.717	45.3	214
44.88	2168	1.297	544	0.789	37.1	164
39.71	2165	1.517	856	0.916	46.6	211
37.56	2145	1.697	1022	1.045	47.9	216
48.64	2106	1.194	344	0.648	28.9	161

37.84	2090	1.68	1011	1.05	48.2	210
41.61	2081	1.315	494	0.677	44.5	230
38.65	2058	1.52	765	0.939	48.8	205
44.47	2020	1.321	491	1.051	39.2	211
47.45	1982	1.246	415	1.14	36.1	224
45.02	1951	1.23	367	0.627	37.5	173
46.55	1910	1.238	384	0.733	32.6	152
46.69	1838	1.208	366	0.664	34.8	153
50.80	1818	1.153	285	0.704	26.6	152
38.55	1810	1.423	606	0.77	50.5	189
38.91	1805	1.424	599	0.767	48.1	204
38.04	1790	1.496	686	0.8	47.1	224
42.63	1780	1.362	509	0.853	37.8	205
39.24	1769	1.401	585	0.765	49.1	204
47.81	1765	1.184	345	0.695	34	151
49.41	1760	1.222	372	1.195	31	211
38.55	1758	1.442	628	0.762	48.4	199
38.00	1755	1.48	694	0.79	49.5	190
37.97	1750	1.5	714	0.82	48.7	189
42.65	1744	1.325	524	0.727	40.5	190
38.96	1741	1.409	563	0.759	48.4	217
44.50	1728	1.259	397	0.941	41.8	215
46.77	1700	1.232	364	1.028	36.6	206
42.58	1698	1.408	646	0.964	40	193
48.98	1660	1.221	421	1.298	37.1	203
45.71	1658	1.212	368	0.865	41.4	186
43.70	1620	1.265	404	0.847	42.9	188
45.98	1593	1.268	421	1.181	39.8	203
47.00	1570	1.241	366	1.315	39	207
47.49	1562	1.261	463	1.281	38.9	196
47.55	1530	1.24	355	1.228	35	209
42.17	1530	1.334	566	0.817	45.2	185
39.87	1510	1.365	522	0.73	47.8	189
46.29	1492	1.201	341	0.716	37.4	159
48.61	1450	1.214	359	1.25	35.4	208
46.33	1414	1.249	425	1.155	41	185
48.83	1390	1.154	287	0.718	33.4	141
47.84	1370	1.192	313	1.174	38.2	205
48.64	1302	1.17	242	0.824	31.4	180
50.07	1271	1.139	198	0.775	29.2	187
48.20	1225	1.176	267	1.263	38	211
48.05	1225	1.17	260	1.168	38	211
48.95	1220	1.173	267	0.884	31.4	174
48.58	1195	1.152	214	0.664	31.9	180
50.04	1085	1.128	169	0.638	29.1	187
49.80	1058	1.13	220	0.79	32.3	127
52.23	952	1.092	142	0.667	26.9	146
47.34	790	1.168	274	1.005	39.8	150
37.97	3573	1.875	1507	0.951	39.3	225
43.21	3571	1.471	898	0.802	32.7	175
43.81	3426	1.451	898	0.802	32.7	150
35.37	3405	1.997	1579	0.93	42.8	235
43.35	3354	1.431	825	0.779	34.2	185

43.53	3311	1.425	825	0.779	34.2	175
42.75	3297	1.458	867	0.799	35.4	180
44.45	3279	1.43	898	0.802	32.7	125
38.10	3250	1.747	1203	0.925	40.2	240
43.51	3228	1.413	775	0.783	34.4	175
44.82	3223	1.387	750	0.8	32	175
38.85	3218	1.686	1151	0.894	39.9	220
44.75	3204	1.372	742	0.752	32.6	160
36.79	3201	1.92	1579	0.93	42.8	190
35.78	3198	1.986	1602	0.96	44.6	230
43.94	3160	1.392	730	0.757	33.1	175
44.30	3155	1.384	700	0.774	32.2	185
43.49	3155	1.427	818	0.789	34.2	170
45.05	3127	1.411	898	0.802	32.7	100
44.55	3101	1.376	700	0.774	32.2	175
45.41	3090	1.36	680	0.755	29.7	175
43.89	3066	1.42	867	0.799	35.4	140
42.57	3057	1.445	811	0.812	36.5	185
44.63	3057	1.371	679	0.778	32	175
40.04	3030	1.636	1151	0.894	39.9	180
45.77	3003	1.34	665	0.766	30.8	175
43.29	2941	1.421	811	0.812	36.5	160
43.32	2925	1.406	693	0.774	33.2	175
45.34	2901	1.352	700	0.774	32.2	140
45.47	2900	1.365	818	0.789	34.2	100
38.14	2896	1.852	1579	0.93	42.8	145
45.34	2871	1.368	825	0.779	34.2	100
46.27	2865	1.327	742	0.752	32.6	100
39.51	2845	1.682	1143	0.951	39.4	240
43.84	2836	1.403	811	0.812	36.5	140
40.53	2831	1.642	1203	0.925	40.2	160
45.03	2804	1.384	867	0.799	35.4	100
45.47	2789	1.352	775	0.783	34.4	100
46.63	2751	1.333	750	0.8	32	100
47.36	2687	1.304	680	0.755	29.7	100
41.44	2652	1.718	1507	0.951	39.3	100
46.34	2639	1.323	700	0.774	32.2	100
40.35	2636	1.647	1143	0.951	39.4	200
44.87	2617	1.371	811	0.812	36.5	100
46.53	2607	1.315	679	0.778	32	100
47.77	2588	1.284	665	0.766	30.8	100
39.55	2559	1.786	1579	0.93	42.8	100
45.54	2558	1.323	602	0.803	33	170
45.15	2530	1.349	693	0.774	33.2	100
42.98	2521	1.44	746	0.907	36.1	200
42.32	2504	1.548	1151	0.894	39.9	100
45.19	2445	1.329	585	0.815	33.3	180
42.23	2413	1.576	1203	0.925	40.2	100
44.93	2401	1.318	567	0.782	34.5	175
41.87	2392	1.479	805	0.929	39.1	200
46.95	2365	1.279	498	0.798	30.1	175
47.34	2359	1.274	521	0.801	30.1	160
39.72	2350	1.789	1602	0.96	44.6	100

41.56	2344	1.599	1143	0.951	39.4	150
47.98	2259	1.257	521	0.801	30.1	135
46.20	2256	1.3	585	0.815	33.3	140
47.44	2249	1.272	469	0.824	28.8	165
44.27	2231	1.398	746	0.907	36.1	150
44.37	2230	1.316	580	0.802	38.1	175
51.22	2177	1.213	421	0.799	21.9	145
47.33	2172	1.273	602	0.803	33	100
40.88	2172	1.734	1493	1.008	43.6	100
46.70	2148	1.286	585	0.815	33.3	120
43.25	2133	1.432	805	0.929	39.1	150
48.64	2132	1.24	521	0.801	30.1	110
42.08	2124	1.406	692	0.876	41.9	185
47.22	2035	1.272	585	0.815	33.3	100
43.88	2016	1.452	803	1.013	36.2	160
49.36	1990	1.222	521	0.801	30.1	85
43.03	1988	1.375	692	0.876	41.9	150
48.97	1981	1.226	498	0.798	30.1	100
45.71	1962	1.354	746	0.907	36.1	100
49.13	1928	1.228	469	0.824	28.8	100
47.78	1912	1.257	585	0.815	33.3	80
46.38	1890	1.259	580	0.802	38.1	100
44.65	1847	1.387	805	0.929	39.1	100
43.51	1834	1.425	755	1.004	39.3	170
44.02	1824	1.344	692	0.876	41.9	115
44.16	1766	1.533	1087	1.056	38	100
45.06	1641	1.313	692	0.876	41.9	80
45.61	1631	1.397	803	1.013	36.2	100
50.23	1630	1.203	347	0.933	26.1	165
44.70	1603	1.387	755	1.004	39.3	125
46.71	1480	1.28	412	0.973	31	180
44.87	1477	1.327	560	1.002	38.6	150
47.22	1472	1.267	417	0.98	31.2	185
49.42	1437	1.226	389	1.002	28.2	150
51.87	1405	1.165	347	0.933	26.1	100
47.49	1405	1.259	412	0.973	31	160
47.86	1378	1.25	417	0.98	31.2	160
49.06	1377	1.21	331	0.921	28.4	160
46.03	1367	1.347	755	1.004	39.3	80
48.29	1292	1.238	412	0.973	31	130
45.45	1282	1.291	469	0.96	36.5	155
48.68	1265	1.229	417	0.98	31.2	130
49.55	1230	1.188	302	0.931	28.9	160
51.48	1205	1.177	389	1.002	28.2	80
47.10	1193	1.246	469	0.96	36.5	130
49.17	1180	1.216	412	0.973	31	100
51.35	1180	1.156	331	0.921	28.4	100
47.10	1159	1.262	512	1.01	37	100
49.53	1153	1.208	417	0.98	31.2	100
48.32	1095	1.268	433	1.188	31.2	190
51.73	1094	1.18	265	1.058	22.8	185
51.10	1061	1.152	302	0.931	28.9	100
49.21	966	1.245	433	1.188	31.2	150

51.00	874	1.152	232	0.989	27.2	160
49.50	854	1.141	196	0.942	32.1	175
53.92	847	1.132	265	1.058	22.8	100
50.42	804	1.215	433	1.188	31.2	100
52.67	697	1.102	189	1.031	27.9	80
51.49	696	1.097	196	0.942	32.1	100
46.41	642	1.22	266	1.192	37.3	165
47.54	601	1.191	266	1.192	37.3	145
52.20	584	1.114	127	1.025	25.1	160
51.19	545	1.125	141	1.072	27.5	155
48.68	518	1.163	266	1.192	37.3	105
53.06	515	1.096	127	1.025	25.1	120
51.88	508	1.11	141	1.072	27.5	130
50.03	477	1.169	158	1.308	27.1	220
49.13	444	1.173	168	1.367	30.5	205
52.51	421	1.045	62	0.875	31.6	170
52.06	408	1.098	104	1.126	27.4	160
50.20	392	1.148	168	1.367	30.5	165
52.95	370	1.099	79	1.146	23.5	185
51.39	368	1.124	100	1.247	26	205
51.22	343	1.125	168	1.367	30.5	125
52.57	331	1.078	74	1.093	27.4	160
53.89	327	1.08	79	1.146	23.5	145
53.51	293	1.059	74	1.093	27.4	120
52.48	290	1.108	103	1.335	25.4	155
54.02	263	1.079	45	1.123	21.8	190
50.53	261	1.093	44	1.05	30.2	205
52.53	255	1.086	61	1.272	26.2	160
54.73	246	1.065	45	1.123	21.8	160
53.51	240	1.066	61	1.272	26.2	140
51.52	238	1.072	44	1.05	30.2	165
52.66	236	1.09	61	1.356	25.4	190
51.88	236	1.091	80	1.297	28.5	155
55.46	231	1.051	45	1.123	21.8	130
54.48	214	1.047	61	1.272	26.2	100
52.50	214	1.052	44	1.05	30.2	125
53.40	211	1.075	61	1.356	25.4	160
55.77	205	1.061	39	1.251	19.4	160
54.20	186	1.059	61	1.356	25.4	130
53.39	186	1.075	29	1.185	23.6	190
56.63	179	1.045	39	1.251	19.4	120
54.10	174	1.061	29	1.185	23.6	160
50.54	174	1.039	46	1.105	38.9	100
51.12	163	1.083	26	1.182	29.2	200
54.82	161	1.047	29	1.185	23.6	130
55.62	148	1.032	29	1.185	23.6	100
52.13	147	1.062	26	1.182	29.2	160
38.55	3840	1.801	1408	0.838	33.9	216
39.23	3798	1.711	1260	0.851	36.6	218
39.48	3647	1.722	1295	0.831	34.0	218
36.78	3220	1.779	1184	0.798	36.4	238
39.78	3212	1.536	886	0.806	40.3	219
36.19	3200	1.852	1246	0.91	39.6	250

37.67	3187	1.707	1102	0.861	40.3	228
40.22	3184	1.647	1018	0.865	31.2	226
37.38	3172	1.753	1186	0.825	37.6	230
36.24	2946	1.946	1439	0.924	36.9	240
38.65	2944	1.65	1008	0.841	37.5	230
38.84	2768	1.686	1016	0.942	36.8	218
39.22	2568	1.677	941	1.036	36.6	230
40.58	2509	1.572	963	0.865	36.8	220
40.78	2482	1.619	948	1.061	37.2	229
40.70	2425	1.571	816	0.873	31.3	250
39.00	2417	1.602	889	0.899	39.6	220
40.57	2310	1.62	882	1.063	35.2	229
40.92	2254	1.556	765	0.923	31.8	243
40.82	2061	1.533	737	0.936	34.5	234
41.35	1920	1.422	523	0.838	35.6	250
43.39	1719	1.416	554	0.975	31.7	216
41.97	1625	1.489	631	1.047	33.5	244
42.23	1591	1.475	583	1.054	32.2	239
43.61	1490	1.424	537	0.989	29.4	239
40.58	1430	1.478	554	0.958	35.8	226
45.08	1401	1.342	490	0.959	31.7	212
42.17	1345	1.364	390	0.923	36.3	254
45.21	1325	1.345	439	1.145	32.1	213
46.59	1261	1.29	364	0.987	28.4	215
45.94	1207	1.322	405	1.079	29.7	212
41.96	1197	1.412	457	1.05	36.0	220
44.21	1179	1.334	406	1.048	34.5	220
44.08	1141	1.335	446	0.98	35.4	190
45.85	1110	1.328	409	1.087	29.5	234
44.98	1104	1.346	408	1.069	30.2	232
45.17	1065	1.305	392	1.061	34.2	213
44.65	1062	1.34	393	1.09	32.0	234
45.43	1030	1.322	333	1.055	28.2	230
46.03	994	1.301	343	1.16	30.6	230
47.04	901	1.24	242	1.12	30.1	235
47.12	710	1.252	265	1.144	29.4	216
48.26	601	1.216	209	1.29	29.0	218

APPENDIX II

This Appendix is devoted for the list of programs generated or modified by the author.

GMDH code generated by Gints Jekabsons (Jekabsons, 2011)

function gmdhbuild

```
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/>.
% ============================================================

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

    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

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
```

function gmdheq

```
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
```

function gmdhpredict

```
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
```