SYSTEM IDENTIFICATION USING GROUP METHOD OF DATA HANDLING

(GMDH)

A THESIS IN MECHATRONICS
Master of Science in Mechatronics

Presented to the faculty of the American University of Sharjah
School of Engineering
in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
YASMEEN MOHAMMED ZUHAIR ABU-KHEIL
B.S. 2005

Sharjah, UAE
January 2009

We approve the thesis of Yasmeen Mohammed Zuhair Abu- Kheil

Date of signature

Dr. Khaled Assaleh, 16/2/2009
Associate Professor, Department of Electrical Engineering
Thesis Advisor

Dr. Tamer Shanableh, 10/2/09
Associate Professor, Department of Computer Science & Engineering
Thesis Co-Advisor

Dr. Rached Dhaouadi, 22/2/09
Associate Professor, Department of Electrical Engineering
Graduate Committee

Dr. Yousef Al Assaf, 16/2/09
Professor, Department of Electrical Engineering
Graduate Committee

Dr. Ameen El-Sinawi, 2/16/2009
Associate Professor, Department of Mechanical Engineering
Graduate Committee (External Examiner)

Dr. Rached Dhaouadi, 22/2/09
Coordinator,
Mechatronics Engineering Graduate Program

Dr. Hany El Kadi, 22/2/09
Associate Dean, College of Engineering

Dr. Yousef Al Assaf, 23/2/09
Dean, College of Engineering

Mr. Kevin Mitchell, 23/2/09
Director, Graduate & Undergraduate Programs

SYSTEM IDENTIFICATION USING GROUP METHOD OF DATA HANDLING (GMDH)

Yasmeen Abu Kheil, Candidate for the Master of Science Degree

American University of Sharjah, 2009

ABSTRACT

Recently, many researchers have had much interest in various methods for system identifications. Such methods involve soft computing techniques such as neural networks and fuzzy logic. Neural networks and fuzzy logics are used to identify and predict non-linear systems based on empirical data. However, using such methods, the nonlinear dynamics aren't explicitly expressed as a mathematical model. Hence, polynomial classifiers and networks were introduced to obtain a mathematical model for the non-linear systems. However, polynomial classifiers require huge storage memory and can lead to instability when it uses higher order polynomials. Therefore, Group Method of Data Handling (GMDH) is introduced. GMDH is a multilayered network with a certain structure determined through training. It has the feature that the nonlinear dynamics are

expressed as a mathematical model as well as the polynomial can have higher order terms without instability problems.

In this thesis, the GMDH networks was implemented and then applied to the identification problem of 2000N MR damper. The GMDH network results were then compared with other nonlinear system identification method such as neural networks and polynomial classifiers. It was found that GMDH network can effectively emulate the behavior of a 2000N MR damper.

Contents

LIST OF ILLUSTRATIONS

Figures Page

LIST OF TABLES

Tables Page

ACKNOWLEDGEMENTS

During the development of my graduate studies at American University of Sharjah, several people collaborated directly and indirectly with my research. Without their support, it would be impossible for me to finish my work. That is why I wish to dedicate this section to recognize their support.

I want to start expressing a sincere acknowledgement to my advisors, Dr. Khaled Assaleh and Dr. Tamer Shanableh because they gave me the opportunity to research under their guidance and supervision. I received motivation, encouragement and support from them during all my thesis work. I would also like to thank the rest of my Professors at American University of Sharjah for their help and support throughout my academic experience at AUS.

Furthermore, I would like to thank Dr. Davaie Markazi, Associate Professor in Iran University of Science & Technology, for providing the experimental data sets that were used in this thesis.

Special thanks go to my colleagues at AUS for their cheerful attitude and ability to make me laugh during stressful situations.

At last, but the most important, I would like to thank my family for their unconditional support, inspiration and love.

1 INTRODUCTION

This chapter provides an overview of system identification and its applications. It begins by introducing the research problem. Then, it proceeds to the aims and objectives of this research. It also highlights my contributions. Finally, the organization of the thesis is outlined.

1.1 System Identification

System identification is defined as mathematical tools and algorithms to describe and predict dynamic behavior of a system or process from measured data (Water, P.R, Kerckhoffs,J.H, & Welden,D.V., 2000). Figure 1 illustrates the task of system identification. In system identification, a model should represent the behavior of a process as closely as possible. Both the process and model are feed with the same inputs $u= [u_1\ u_2\ .\ .\ .\ .\ u_p]$ and their outputs y and y_m are compared yielding the error signal e which determines the quality of the model. Based on the error between the process output and the model output, the model parameters are adjusted and updated (Nelles, 2001).

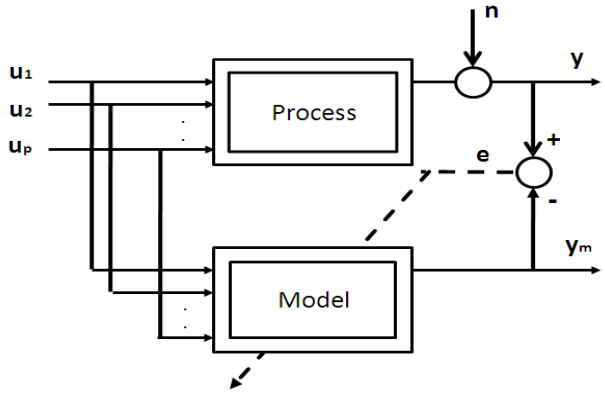


Figure 1: Block Diagram of System Identification

For successful system identification, several major steps have to be performed. These steps involve the choice of model inputs, choice of excitation signals, choice of model architecture, choice of dynamic representation, choice of the model order, choice of the model structure and complexity, choice of the model parameters and finally model validation (Ljung, 1987) . Figure 2 shows system identification steps.

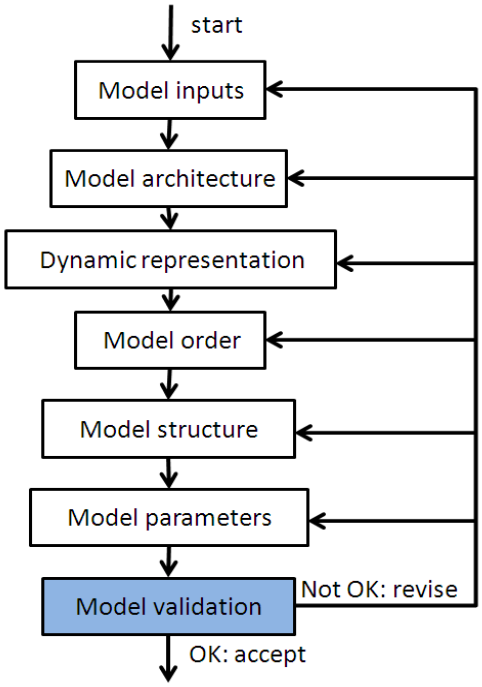


Figure 2: System Identification Steps

Recently, many researchers have had much interest in various methods for system identification. Among them, mathematical modeling methods, such as regression techniques, were widely used to identify and predict the linear systems based on input-output data. However, using mathematical models to express dynamic analysis of non-linear real systems have had many problems in the selection of variables and model

structure (Park, 1999). Therefore, many methods for identifying and modeling non-linear systems were proposed. Such methods involve neural networks, fuzzy inference, genetic algorithm and polynomial classifiers. Most of these methods require large amount of data to estimate the parameters of the model in higher order systems. For example, neural networks and fuzzy logics are used to identify and predict non-linear systems based on empirical data. However, using such methods, the non-linear dynamics aren't explicitly expressed as a mathematical model. Hence, polynomial classifiers and networks were introduced to obtain a mathematical model for the non-linear systems (Zhao, X.M, Song, Z. & Ping,L., 2002). On the other hand, polynomial classifiers require huge storage memory and can lead to instability when it uses higher order polynomials. Therefore, Group Method of Data Handling (GMDH) is introduced. GMDH is a multilayered network with a certain structure determined through training. It has the feature that the non-linear dynamics are expressed as a mathematical model as well as the polynomial can have higher order terms without instability problems. Therefore, GMDH provides an effective approach to identification of higher order non-linear systems. .

1.2 Applications of System Identification

Model of real systems have gained a great importance. They can be used for system analysis to gain a better understanding of the system. Models also allow us to predict and simulate systems' behaviors. Furthermore, models are required for designing new processes, analyzing existing processes, designing controllers, optimizations, supervision, and fault detection and diagnosis. Therefore, various techniques for system identifications

are applied in many fields in order to model and predict the behaviors of complex systems based on given input/output data. They are widely used in control system theory, biomedical, engineering, financial and operation research (Rahim, N.A.,Taib,M.N, Adom, A.H & Mashor, M.Y.) .

1.3 Thesis Objectives and Contribution

The purpose of this thesis is to evaluate the performance of GMDH as a system identification tool. Then, the GMDH algorithm is applied to the problem of identifying the non-linear dynamics of an MR damper. The results, indicated in this thesis, showed that the GMDH network can precisely model the forward and inverse models of MR dampers.

1.4 Thesis Outline

This thesis consists of seven chapters which are organized as follow: Chapter 1 introduces system identification and its applications. Chapter 2 provides the necessary background for various system identification techniques. Chapter 3 describes the GMDH algorithms and its features. Chapter 4 presents MR damper as an example of a highly non-linear system. Applications of GMDH to modeling the dynamic behavior of MR damper appears in Chapter 5. Chapter 6 compares the results of identifying MR models using various system identification techniques. Finally, conclusions and future work are presented in Chapter 7.

2 REVIEW OF SYSTEM IDENTIFICATION TECHNIQUES

In this chapter, the architecture and features of artificial neural networks (ANN), Neuro-fuzzy, polynomial classifiers and GMDH networks are described.

2.1 Artificial Neural Networks (ANNs)

2.1.1 Introduction to Neural Networks

One of the most popular non-linear system identification approaches is based on the application of artificial neural networks (ANNs) (Mrugalski, M., Witczaka,M. & Korbicza, J., 2008). The field of artificial neural networks (ANNs) has its roots in neurobiology. The structure and functionality of ANNs has been motivated by the architecture of the human brain. In other words, neural networks inherit the intelligence and non-linear characteristic of the brain. Furthermore, ANNs show the ability to learn from the environment in an interactive way. They also have remarkable abilities of learning, recall, generalization and adaptation to changes of the operating environments (Liu, 2001).

ANNs are composed of a number of interconnected processing elements arranged together with weighted connections. These elementary units are called neurons. Each neuron in the network operates by taking the sum of its weighted inputs and passing the result through a non-linear activation function. Figure 3 illustrates the architecture of a simple neuron while Figure 4 shows the general architecture of neural network with three neurons in the hidden layer. Learning the network is done through training or exposure

to a set of input and output data where the training algorithm adjusts the weights iteratively (Murphy, 2002). Examples of different neural network architectures are described in the following section.

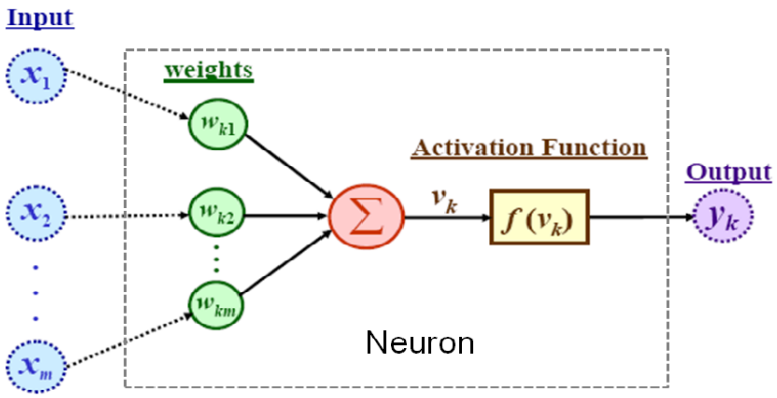


Figure 3: Architecture of a Simple Neuron

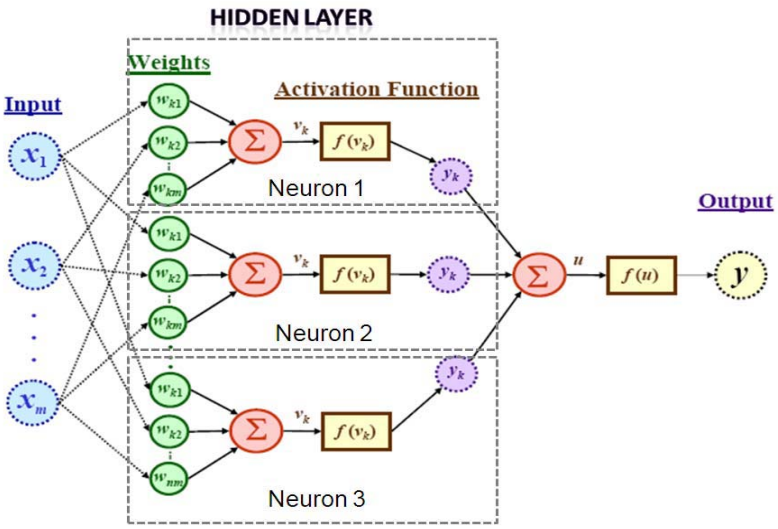


Figure 4: General Neural Network Architecture

2.1.2 Neural Network Architecture

Neural networks have many types that differ in their architecture, usages and applications. In system identification theory, feedforward and recurrent neural networks are the most commonly used neural networks (Murphy, 2002). A feedforward network consists of an input and output layer, one or more layers of neurons in between called the hidden layers, which propagate the inputs forward to the output layer (Rahim, N.A.,Taib,M.N, Adom, A.H & Mashor, M.Y.). The general structure of a multilayer feedforward network is shown in Figure 5. On the other hand, in recurrent network, there is a feedback loop from the output to the input layer or from the output of each hidden layer. The recurrent neural network is also called a dynamic network. Figure 6 shows the recurrent network architecture. Information about other neural networks models can be found in any introductory book on this field (Kung, 1993).

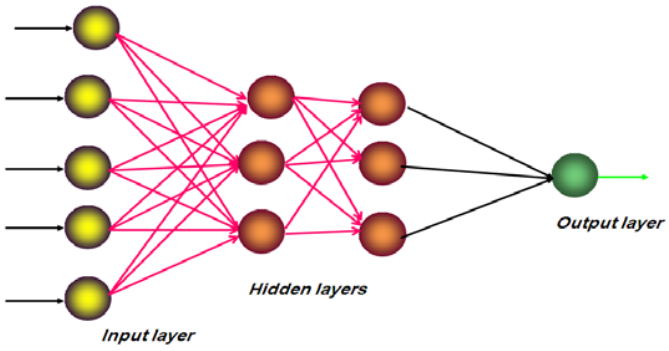


Figure 5: Feedforawrd Network with Two Hidden Layers

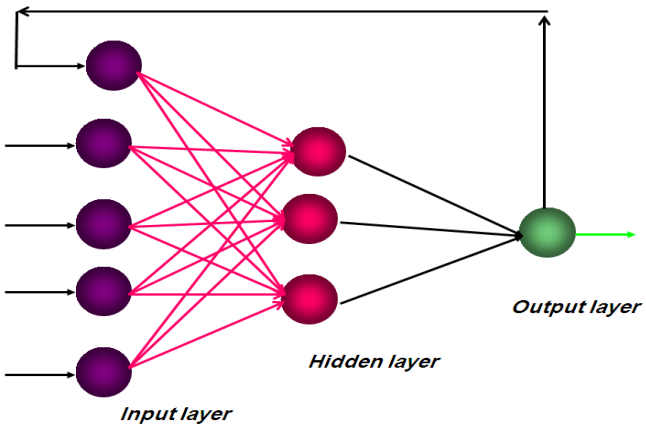


Figure 6: Recurrent Neural Network Architecture

2.1.3 Neural Network System Identification Steps

System identification in neural networks is defined as the procedure used to find the inputs and output relationship of a dynamic model. This step involves four steps: collection of high quality input/output data, selection of a network structure, training of the network and validation of the trained network (Chang, 1998) . The quality of the trained network is related to the quality of the training data which can be obtained by either an experiment of the real system to be modeled or from a simulation using a mathematical model of the system. To produce a meaningful model of the identified system, the training data must include information in the entire operating range of the system. Then, the network architecture should be selected. This issue could be difficult especially for a nonlinear system since both the regressors and the model architecture must be determined a priori. The third and forth steps involve training and evaluating the model, respectively. A commonly used method of validation is to investigate prediction

errors by running sets of test data on the trained model. Figure 7 represent the required steps in neural network system identification.

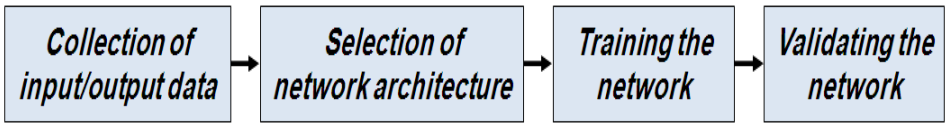


Figure 7: System Identification Steps in Neutral Network

2.1.4 Advantages and Disadvantages of Neural Networks

ANNs has several properties that make them an attractive tool for system identification. Such properties include the ability to learn, adopt and parallel data processing. Other properties are simplicity of implementation, generalizations abilities and good approximation properties. Furthermore, neural networks do not require a priori knowledge of the process being identified. They learn by extracting information from empirical data that describe the relationship between the inputs and the outputs. They are also capable of handling complex and non-linear problems (Rahim, N.A.,Taib,M.N, Adom, A.H & Mashor, M.Y.). However, the successful application of the ANNs in the system identification task depends on a proper selection of the neural network architecture. In the case of the classical ANNs such as Multi-Layer Perceptron (MLP), the problem reduces to the selection of number of layers and the number of neurons in a particular layer. If the obtained network doesn't satisfy pre-specified requirements, then, a new network structure is selected and the parameter estimation is repeated again. Determination of the appropriate structure and parameters of the model in the presented

way is a complex task. Furthermore, an arbitrary selection of the ANNs structure can be a source of model uncertainty (Korbicz, J. & Mrugalski, M., 2008)

2.2 Neuro Fuzzy Networks

2.2.1 Fuzzy Models

Fuzzy logic, introduced by Zadeh in 1965, emulates the way in which the human brain deals with concepts such as uncertainty, vagueness, and imprecision. It allows human to express and process relationships in form of rules (Nelles, 2001). The rules of fuzzy models are generally determined by an expert who knows the system very well or by numerical methods. A typical fuzzy model consists of a rule-base module that contains a number of fuzzy if-then rules, a database that defines the membership functions (MFs) of the fuzzy sets used by the fuzzy rules, a decision-making subsystem that performs the inference operations on the rules, a fuzzification module that transforms inputs to degrees of membership to different fuzzy sets and finally, a defuzzification module that transforms the fuzzy results into an output. Figure 8 shows the block diagram of a typical fuzzy interface system (Vlachos, D. & Tolias, Y.A., 2003).

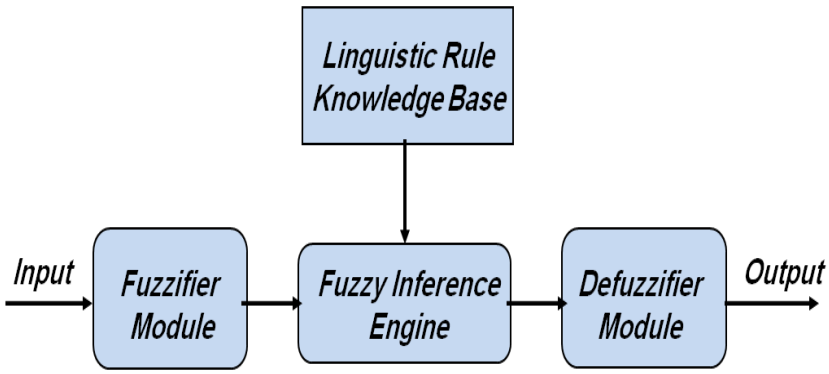


Figure 8: Block Diagram of Fuzzy Interface System

2.2.2 Motivation for Neuro-Fuzzy Networks

One of the major problems with the use of fuzzy models alone in system identification is the difficulty of choice and design of membership functions to suit a given application. In other words, there is no systematic procedure for choosing the types and ranges of membership functions. Furthermore, qualitative knowledge isn't sufficient enough to build a high accuracy fuzzy model. On the other hand, neural networks have the ability to learn non-linear systems from a set of input/output data. Therefore, the learning capabilities of neural network as well as the control capabilities of fuzzy systems are combined together to form a Neuro-fuzzy model. A Neuro-Fuzzy model will incorporate prior knowledge into the model before and during identification using fuzzy models. It will also interpret the obtained identification model using neural networks models (Mohannad, K. & Shaik.K., 2001). Furthermore, a combination of both neural networks and fuzzy logic can improve speed, fault tolerance, and adaptiveness (Galina, 2008).

One of the most commonly used Neuro-fuzzy systems is the Adaptive Neuro-Fuzzy Interface System (ANFIS). Figure 9 represents the architecture of ANFIS. ANFIS is a multi-layer feedforward network that consists of five layers; the first layer is composed of *n* MFs, each of which implements a fuzzy decision rule, the second layer computes every possible conjunction of the *n* decision rules, the third layer normalizes the conjunctives MFs to rescale the inputs, the fourth layer is a standard neuron perceptron that associates every normalized MF with an output, and the fifth layer sums the evidences to produce a final output (Roger, 1993)

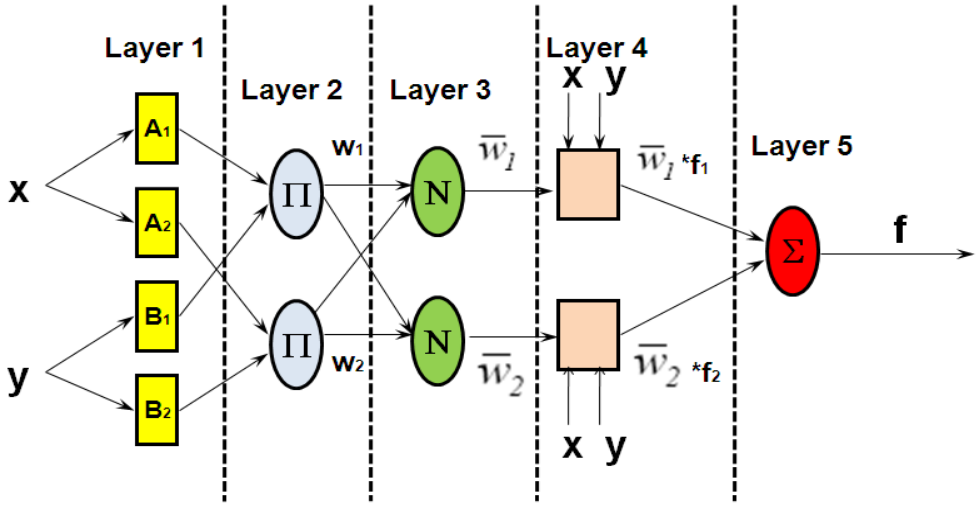


Figure 9: Adaptive Neuro-Fuzzy System Architecture

2.3 Polynomial Classifiers

2.3.1 Polynomial Classifiers Model and Algorithm

Polynomial classifiers are non-linear system identification techniques that provide an effective way to describe complex non-linear input/output relationships. The higher

the degree of the polynomial, the more flexible the model becomes. The general polynomial model can be expressed by a p-dimensional polynomial function of degree l which is given in Equation 2.1:

$$g = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i1=1}^{p} \sum_{i2=i1}^{p} w_{i1i2} x_{i1} x_{i2} + \cdots \sum_{i1=1}^{p} \ldots \sum_{il=i1-1}^{p} w_{i1\ldots il} x_{i1} \ldots x_{il} \quad 2.1$$

The offset and the first sum describe a linear model, the second sum describes the second order term such as x_1^2, $x_1 x_2$, …etc., and the last sum describes the l^{th} order term such as x_1^l, $x_1^{l-1} x_2$,… etc.

Equation 2.1 can be expressed in another form:

$$g(x) = w_0 + \sum_{i=1}^{l} w_i x_i + \sum_{i=1}^{l-1} \sum_{m=i+1}^{l} w_{im} x_i x_m + \sum_{i=1}^{l} w_{ii} x_i^2 \quad 2.2$$

Where:

$M = \frac{(l+p)!}{l!p!} - 1$, $x_0 = 1$

The polynomial model parameters can be estimated by least squares. The polynomial classifier algorithm can be described in the following steps (Liu, 2001):

1. Collect inputs-output data set

$$\text{Input} = \begin{bmatrix} x_{11} & x_{21} & x_{31} & . & x_{m1} \\ x_{12} & x_{22} & x_{32} & . & x_{m2} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{1N} & x_{2N} & x_{3N} & . & x_{mN} \end{bmatrix}_{Nxm} \qquad \text{Ouput} = \begin{bmatrix} Y_1 \\ Y_2 \\ . \\ . \\ Y_N \end{bmatrix}_{Nx1}$$

Where: N→ number of data samples, m → number of features

2. Expand the input to a 2nd or higher order polynomial (Z)

$$Z = \begin{bmatrix} 1 & x_{11} & x_{21} & . & x_{m1} & x_{11}x_{21} & . & x_{11}x_{(m-1)1} & {x_{11}}^2 & . & {x_{m1}}^2 \\ 1 & x_{12} & x_{22} & . & x_{m2} & x_{12}x_{22} & . & x_{12}x_{(m-1)2} & {x_{12}}^2 & . & {x_{m2}}^2 \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ 1 & x_{1N} & x_{2N} & . & x_{mN} & x_{1N}x_{2N} & . & x_{1N}x_{(m-1)N} & {x_{1N}}^2 & . & {x_{mN}}^2 \end{bmatrix}_{Nx(2m+1+\frac{m(m-1)}{2})}$$

3. Compute the weights as follow:

$$W = (Z^T Z)^{-1} Z^T y$$

4. Using the calculated weights, evaluate the estimated output (Ym) using the following equation:

$$Ym = Z*W$$

5. Calculate the least square error to evaluate the performance of the identified system:

$$\text{Error} = \frac{\sum_{i=1}^{N}(Y_i - Y_{mi})^2}{\sum_{i=1}^{N}(Y_i)^2}$$

2.3.2 Advantages and Disadvantages of Polynomial Classifiers:

The polynomial classifier algorithm, described in the previous sections, is simple, fast and easy to implement. However, it requires a huge storage memory. Furthermore, the number of parameters and thus the model complexity grows strongly with an increasing number of inputs p and/or polynomial degree l. Therefore, the estimation of a full polynomial model isn't practical in terms of computational cost and storage space. Therefore, for most system identification problems, polynomial models can only be applied in combination with structure selection techniques (Liu, 2001). Another

drawback of polynomial models is their tendency to oscillatoric interpolation and extrapolation behavior, especially with high degree polynomials. To solve the problems associated with polynomial classifier, GMDH algorithm is proposed and briefly explained in the following section.

2.4 Group Method of Data Handling (GMDH)

GMDH is a multilayered network with a certain structure determined through training. It has the feature that the non-linear dynamics are expressed as a mathematical model as well as the polynomial can have higher order terms without instability problems (Korbicz, J. & Mrugalski, M., 2008). Furthermore, GMDH is used to detect input-output relationships for various models. It aims to find relationships between one output and a frequently large set of possible inputs. The network decides which of the possible inputs are actually relevant to the system being identified. Therefore, the network is built up layer by layer during training. In each layer, there are neurons with only two inputs; the output of each neuron it a quadratic function of its both inputs. The parameters of the quadratic functions are obtained using linear regression analysis. Before adding a new layer, the previous layer is trained. During this training, for each unique combination of two inputs, a neuron is trained and on the basis of a certain selection criteria, only the best performing neurons are selected. Then, a new layer is added, and the whole procedure of training is performed again on this new layer. Adding new layers is done if some stopping criteria are achieved (Water, P.R, Kerckhoffs,J.H, & Welden,D.V., 2000). Figure 10 shows the architecture of GMDH network. A detailed description of GMDH,

its application in system identification, external criteria and its algorithm is provided in the following chapter.

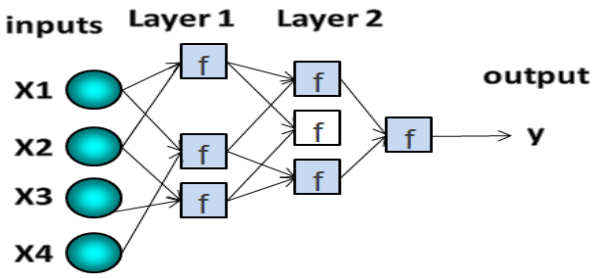


Figure 10: GMDH Network Architecture

3 GROUP METHOD OF DATA HANDLING (GMDH)

In this chapter a detailed description of system identification using GMDH algorithm is provided. It includes the partial model construction, sorting out procedure description, external criteria of accuracy. It also outlines the implementation and algorithm of GMDH network.

3.1 Outline of GMDH and Its Applications

GMDH is a modeling technique that provides an effective approach to the identification of higher order non-linear systems. It was first introduced by A.G Ivakhnenko (Ivakhnenko, 1968). Furthermore, GMDH is an inductive self-organizing algebraic model since it is not necessary to know the exact physical model in advance. Instead, GMDH automatically learns the relations that dominate the system variables during the training process. In other words, the optimal neuron's structure is selected automatically in a way that minimizes the values of the prediction error criteria and unnecessary neurons are eliminated from the network. Therefore, the GMDH has good generalization ability and can fit the complexity of non-linear systems (Kondo, 1998). Based on this basic algorithm, many developed studies have been done to apply the GMDH approach in various applications such as data mining, forecasting, prediction and system identification, pattern recognition, and fault detection and isolation (FDI). For example, in the system identification filed, Konodo (1998) applied GMDH algorithm to the medical image recognition problem. In his study, the shapes of the livers, obtained using stomach X-ray CT image, were recognized automatically and the interests regions

were extracted using GMDH algorithm. The utilized image features were the statistics of the image density such as mean, standard deviation, variance, median, minimum, maximum and range. The final GMDH network had six layers with three neurons in each layer. The useful image features were found to be the mean, standard deviation and variance. Once the GMDH network was obtained, the regions of the livers were extracted by after processing of the resultant image. Finally, the output image of the GMDH network was compared with the original image. It was shown that GMDH algorithm had a good prediction accuracy for medical image recognition. Konodo (1998) also applied GMDH algorithm to the non-linear pattern identification problem. The input variables were the x, y and z coordinates while the output data was the pattern being identified. The inputs and output were trained using GMDH algorithm. The final network consisted of ten layers with three neurons in each layer. The network was evaluated by comparing the actual values of the data with the predicted data from the GMDH network. The obtained results indicated that GMDH network was a very useful method for non-linear pattern identification problem. Another study of the use of GMDH in system identification was proposed by Water.et al (2000). They applied GMDH on two synthetic examples in which the mathematical descriptions were known. The first example was a synthetic linear system while the second example was a synthetic switched randomized sine-wave system. It was shown that at least for these two examples, GMDH can be used in identification of dynamic systems. However, the authors stated that the excellent results obtained from synthetic examples don't guarantee such results for real systems.

Furthermore, Sakaguchi.et al (2004) applied the GMDH approach to identify an atmospheric distillation process system. The set of inputs/output variables were obtained from real experimental setup. The inputs of the network were anti-forming chemical rate, feed rate, production rate of product A, production rate of product B, Pre-fractionators reflux rate, Pre-fractionators pressure and Pre-fractionators feed temperature. The trained network consisted of four layers and eleven neurons; five neurons in the first layer, three neurons in the second layer, two neurons in the third layer and one neuron in the last layer. The network was then evaluated by comparing the predicted network output with the actual output. The obtained results indicated that GMDH can effectively identify an atmospheric distillation process system. In addition, Konodo (2004) proposed a revised GMDH network and applied it to a non-linear system whose mathematical description was known. The non-linear system was identified using two approaches; revised GMDH and conventional radial basic function (RBF) neural network. In the first approach, the revised-GMDH network was trained using four inputs and one output. The revised-GMDH network had a feedback loop and the network parameter such as number of neurons, useful input variables and the number of feedback loop calculations were automatically determined so as to minimize the prediction sum of squares (PSS) criteria. The constructed GMDH network consisted of only three input variables since the fourth variable was irrelevant to the system, four neurons in the hidden layer and three feedback loop calculations. On the other hand, in the second approach, the developed RBF neural network consisted of three layers, three inputs, one output and twenty hidden neurons.

The weights of the neural network were estimated using regression analysis. Finally, the results obtained from both revised GMDH network and conventional (RBF) neural network were compared. It was shown that both estimation and prediction errors of the revised GMDH algorithm were smaller than conventional (RBF) neural network. Therefore, the revised GMDH algorithm provided an accurate identification method for the non-linear system.

The details of the neuron architectures, sorting out procedure, external criteria of accuracy and algorithm will be discussed in the following sections.

3.2 Partial Model Construction

GMDH network is a data -driven modeling technique. It uses mathematical functions to characterize the complex nonlinear relationships among the given inputs-output data sets. The GMDH network consists of a number of layers containing neurons. Each neuron has two inputs and one single output (Ehsan, N., Saeed, B. & Vesal, H., 2004). The output of each neuron is calculated using Ivakhnenko polynomial described in Equation 3.1 :

$$g(x_1, x_2) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2 + w_4 {x_1}^2 + w_5 {x_2}^2 \qquad 3.1$$

The GMDH neuron architecture is shown in Figure 11.

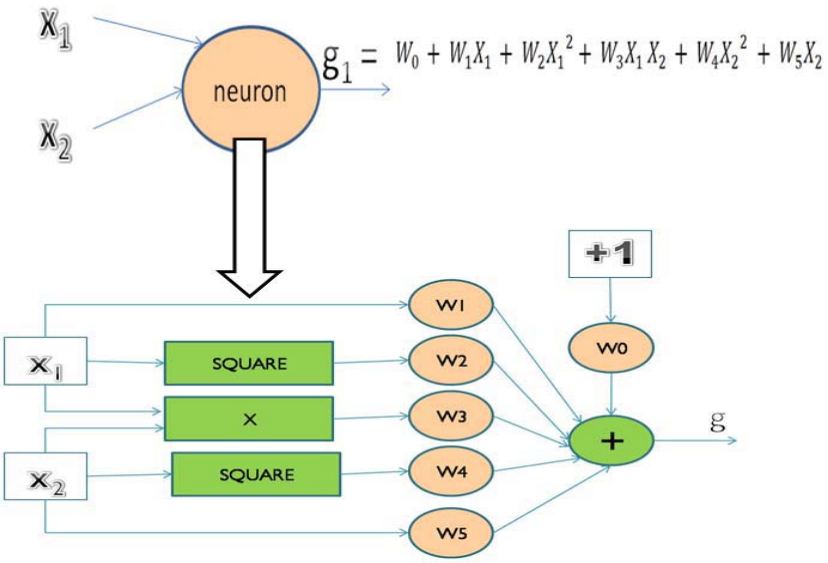


Figure 11: GMDH Neuron Architecture

To construct a complete GMDH network, the combination of each two input variables in each layer is generated. Then, for each combination and its associated output, the coefficients of the polynomials are estimated by least square fitting algorithm or regression analysis that will be described later in Section 3.5. Once the polynomial coefficients are obtained, the output of the polynomials is then evaluated and tested using external criteria of accuracy.

3.3 External Criteria of Accuracy

The external criterion, known as the regularity criteria, is used to test the model adequacy. It evaluates and tests the output of each neuron in the GMDH network by determining the mean square error in the model between the actual and predicted output of each neuron. The regularity criterion also indicates which of the input combination is

more relevant to the network. Furthermore, the regularity criterion tests the capability and fitness of the neuron polynomial to the desired system output. The smaller the regularity criterion, the better fit of the neuron polynomial on the data. The regularity criteria can be expressed by Equation 3.2.

$$R^2 = \frac{\sum_{i=1}^{N}(y_i - g_i)^2}{\sum_{i=1}^{N}(y_i)^2} \quad 3.2$$

Where:

R: Regularity criteria measure

N: Number of samples

y: Desired system output

g: GMDH neuron output

In GMDH network, the regularity criterion for each neuron output is evaluated. Then, it is used to determine the surviving neurons to the next layer according to the sort out procedure described in the following section.

3.4 Sorting Out Procedure Description

As stated in Section 2.4, GMDH modeling is self- organizing since neither the number of neurons nor the number of layer is predefined. The best performing neurons in each layer of the GMDH is selected based on the external criteria of accuracy described in the previous section. The neurons with regularity criteria that are less than a pre-defined threshold value will be selected while other neurons will be eliminated and discarded from the network. Figure 12 shows a GMDH network after eliminating unnecessary neurons. Furthermore, the smallest regularity criterion in each layer is saved.

If the smallest regularity criterion in the next layer is larger than the smallest regularity criterion in the previous layer, adding new layers will stop. The final output of the GMDH network will be the output of the neuron with the smallest regularity criterion in the last layer (Water, P.R, Kerckhoffs,J.H, & Welden,D.V., 2000).

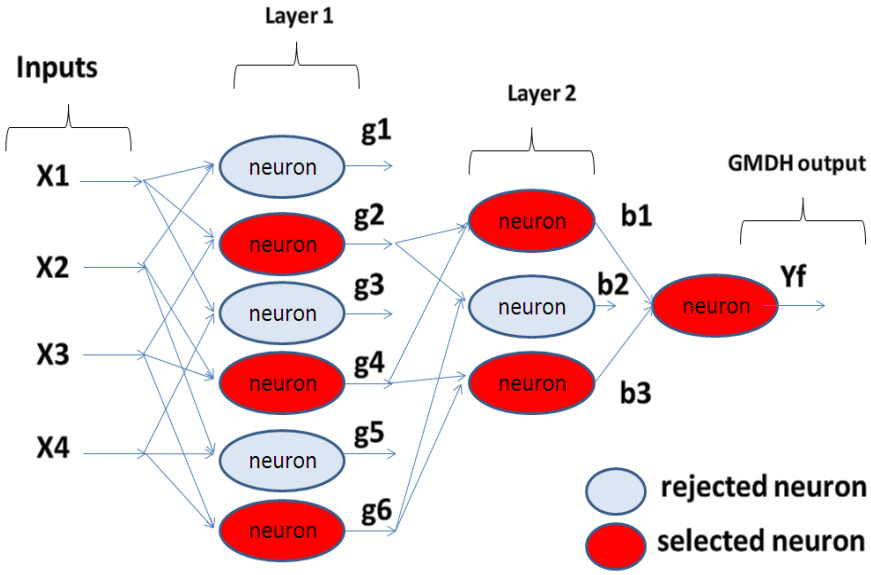


Figure 12: GMDH Network with Rejected Neurons

3.5 GMDH Implementation and Algorithm

The main procedure for GMDH algorithm implementation is described in steps. The first step is the separation of the data into training and checking sets. The training data are used for the estimation of the weights of GMDH neurons while the checking data are used for organizing the network architectures. The division of data is conducted heuristically either by selecting random points for each set or based on the data variance. For example, the points with high variance are used in the checking set to ensure that the selected model is generalized and can extrapolate outside the data in the training set

(Onwubolu, 2007). The second step involves the generation of all possible combinations of two inputs among all the input variables. The number of combinations is given by n= $\frac{m\,(m-1)}{2}$ where m and n are the number of input variables and the number of combinations, respectively. Then, for each combination, expand the inputs to a quadratic polynomial Z.

$$Z = \begin{bmatrix} 1 & x_{11} & x_{21} & x_{11}x_{21} & {x_{11}}^2 & {x_{21}}^2 \\ 1 & x_{12} & x_{22} & x_{12}x_{22} & {x_{12}}^2 & {x_{22}}^2 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 1 & x_{1N} & x_{2N} & x_{1N}x_{2N} & {x_{1N}}^2 & {x_{2N}}^2 \end{bmatrix}_{Nx6}$$

Then, using the training data set, the coefficients of the polynomial (g), described previously in Equation 3.1, are estimated for each combination in the training set. The weights of the polynomials are found by least square fitting algorithm and calculated by Equation 3.3.

$$W = (Z^T Z)^{-1} Z^T y \qquad 3.3$$

The third step is to evaluate and test the output of each polynomial using the data points in the checking data. The output of each polynomial can be calculated using Equation 3.4

$$g = Z*W \qquad 3.4$$

After calculating the output of each polynomial, the regularity criteria, for each neuron in the first layer, are calculated. Based on the regularity criteria calculations, the neurons with a regularity criterion that is less than a pre-defined is allowed to proceed to the next layer where the outputs of the selected neurons become the new input values. On the other hand, the remaining neurons are eliminated from the network. Finally, the

whole procedure from the second step is repeated until the condition for terminating the GMDH network is satisfied. The GMDH network will stop when the lowest regularity criteria in the current layer is no longer smaller than that of the previous layer. To obtain the final GMDH model, the path of the neurons that corresponds to the lowest regularity criteria in each layer is tracked back. The flow chart that describes the GMDH algorithm is shown in Figure 13.

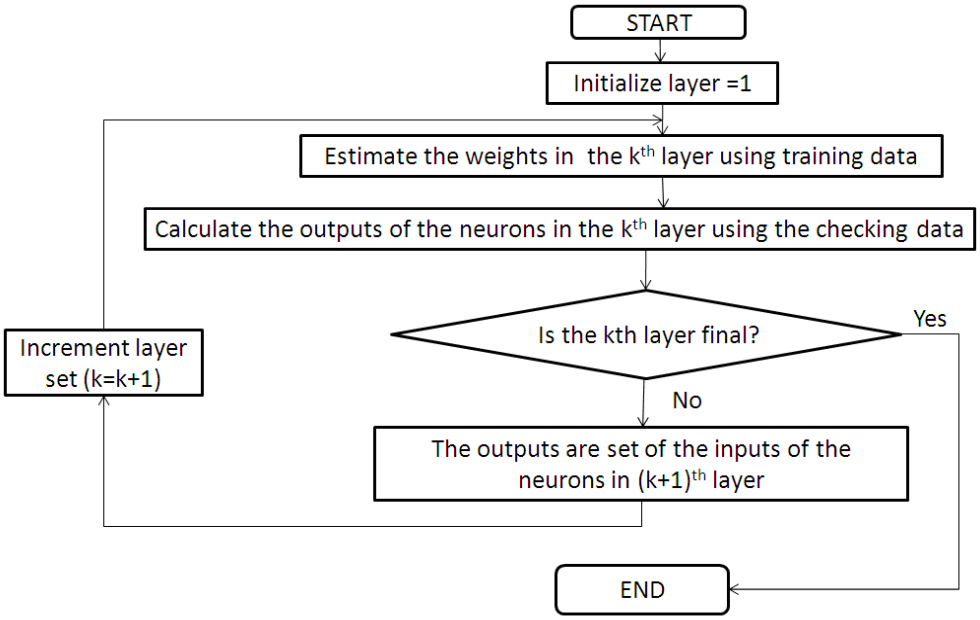


Figure 13: GMDH Network Flow Chart

3.6 GMDH Verses Neural Network

The GMDH algorithm can be viewed as a polynomial neural network, where the processing function of the node is a polynomial rather than a sigmoid function. Therefore, the optimization in the GMDH is based on a series of least-squares fitting rather than an iterative method of minimizing the errors like the back propagation neural

networks. Training in the GMDH is a matter of performing linear algebra rather than a numerical method which requires a large amount of time for convergence. Another advantage of the GMDH is that it cannot over-train. Overtraining a neural network can be a problem when not enough points are given and too much iteration is used to train the network. Due to the training algorithm for the GMDH model, training stops when it reaches the best possible configurations (Kondo, T., Ueno, J. & Kondo, K., 2005). Furthermore, the final model of GMDH is expressed mathematically in terms of high order polynomials. However, the final model extracted by neural networks is still hidden and distributed over the network. In addition, GMDH algorithm has systematical approach on the contrary of the neural network model which is based on a trial-and-error process (Müller, J. A. & Lemke, F., 1999). In summary, the major differences between GMDH and neural networks are the ability of GMDH to objectively select the optimal model, avoid over fitting problems, and select the most relevant input variables. Table1 summarizes the differences between GMDH and neural network modeling (Anastasakis, 2001).

Table1: Features of Both GMDH and Neural Network Modeling

Features	GMDH	Neural Network
Data Analysis	Structure identifier	Universal approximator
Analytical Model	Direct approximation	Indirect approximation
Architecture	Bounded network structure generated during the estimation process	- Pre-selected unbounded network structure - selection of an adequate architecture requires time and experience
Network synthesis	Adaptive synthesized structure	Globally optimized fixed network structure
Threshold	Threshold objective functions	Threshold transfer function
Self-organization	Inductive: number of layers and nodes are estimated by minimum of external criterion	Deductive: require to specify number of layers and number of nodes
Parameter estimation	- Estimation in sets by means of maximum likelihood techniques using all the observational data - extremely short samples	recursive way that demands long samples
ON/OFF line	Data are usually stores and repeatedly accessible	Observation is available transiently in a real-time environment
Regularization	Estimation on training set, selection on testing set	Without: only internal information
Knowledge	Necessary knowledge about the task and the class of system (linear, non-linear)	Requires knowledge about the theory of neural networks
Convergence	Existence of a model of optimal Complexity	Global convergence is difficult to guarantee

table continues

Features	GMDH	Neural Network
Optimizations	- fast -only include relevant terms -irrelevant terms are eliminated	- Global search of a highly multimodal surface - result depends on initial solutions - slow and require the user to set various algorithmic parameters by trial and error - time consuming
Computing	Efficient: for ordinary computers and for massively parallel computation	-Efficient: for implementation in hardware using massively parallel computation -insufficient : for ordinary computers
A priori information knowledge	used directly to select the reference functions and criteria	Not usable if they aren't transformed in the world of neural networks
Feature	General-purpose, flexible linear or non-linear, static or dynamic parametric models	General-purpose, flexible, nonlinear (especially linear) static or dynamic nonparametric models

4 APPLICATION TO MR DAMPER MODELING

In this chapter, an application to MR damper modeling is provided. It introduces the MR damper, its construction and characteristics. Then, it provides some applications of MR devices. Finally, various methods for identifying and modeling the non-linear behavior of MR dampers are described.

4.1 Introduction to MR Dampers

The MR damper typically consists of a hydraulic cylinder that contains a solution called MR fluid. Figure 14 shows an example of a large-scale MR damper. The solution consists of a micrometer-sized magnetically polarizable particles dispersed in a carrier medium such as water, mineral or synthetic oil. In the presence of magnetic field, the particles of the MR fluid form chains and increase the fluid viscosity, until it becomes a semi-solid as shown in Figure 15 (Wilson, 2005)

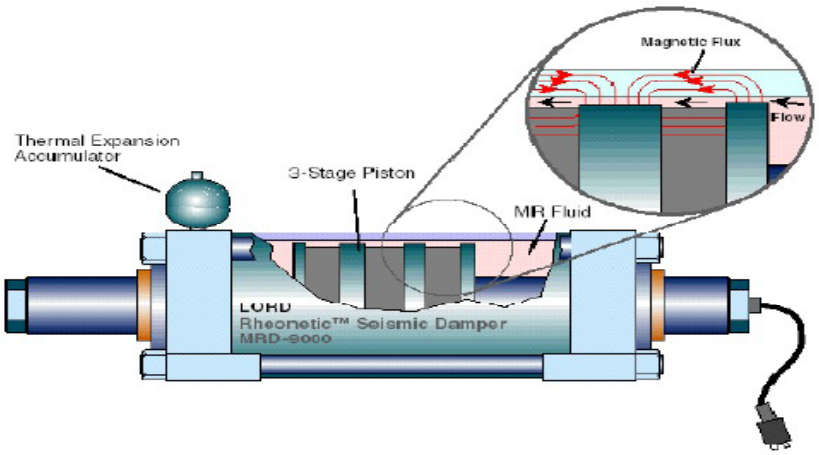
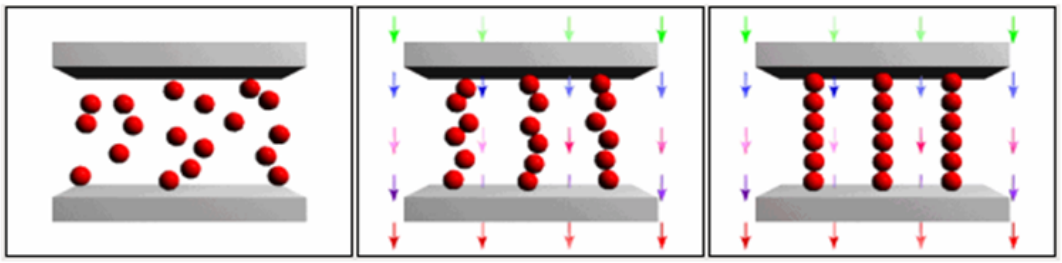


Figure 14: Large-Scale MR Damper (Wilson, 2005)



(a) No magnetic field (b) and (c) with increasing magnetic field

Figure 15: MR Fluid Behavior (Wilson, 2005)

The Magnetorheological (MR) Dampers have unique characteristics such as low power requirement, fast response rate, mechanical simplicity, low manufacturing and maintenance cost, compactness, and environmental robustness (Atray, V., & Roschke, P. N, 2003). Such characteristics allow the MR dampers to be used in various applications.

4.2 MR Damper Application

MR dampers are involved in various applications. For example, MR dampers are used in shock absorbers in NASCAR racing cars to provide faster, quieter and better vibration control than previously used shockers. Furthermore, MR dampers are used to decrease noise and vibration in washing machines and other appliances. MR fluid brakes were also developed to vary the stiffness of exercise equipment. Finally, MR dampers are used to minimize the damage to civil engineering structures due to seismic motion such as earthquakes (Wilson, 2005). Figure 16 illustrates some applications of MR dampers.

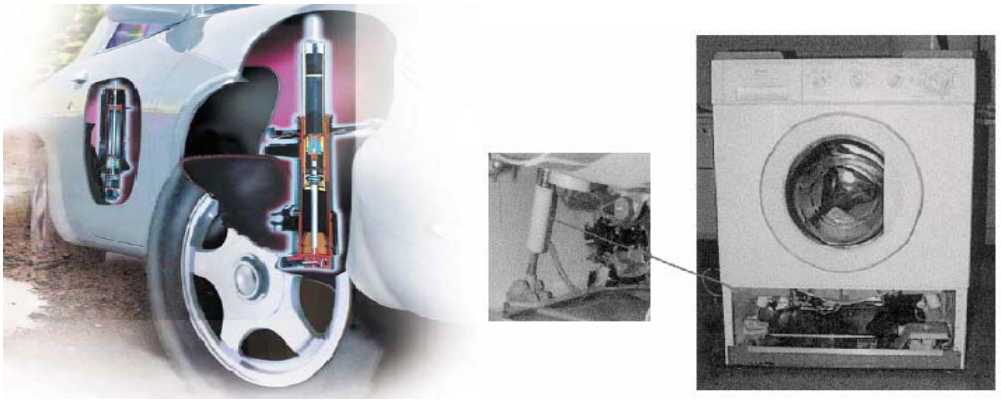


Figure 16: Applications of MR Dampers

However, because of the inherent non-linear nature of MR dampers, it is challenging to model and control MR dampers(Chang, 1998). Therefore, several studies were proposed in the literature to model the behavior of MR dampers.

4.3 MR Damper Models

Several models have been developed for describing the behavior of MR dampers. MR models are divided in two categories: non-parametric and parametric models, both of which are described in the following subsections.

4.3.1 Parametric Models

MR dampers have been previously tested in a laboratory to determine its dynamic properties and modeled by a system of non-linear differential equations(Chang, 1998). Bingham model, proposed by Stanway et al, is one of the first mathematical models developed to describe the behavior of MR dampers. It consists of a coulomb friction element in parallel with a viscous damping element. Figure 17 shows the schematic of Bingham model. The force generated from this model is described in Equation 4.1:

$$F = f_c \, \mathrm{sgn}(\dot{x}) + c_0 \dot{x} + f_0 \qquad 4.1$$

Where:

f_c : Friction force

c_0: Viscous damping coeffecient

$\dot{x}$: Piston velocity

f_0: Offset force that accounts for non-zero mean

Although Bingham model is simple, the damping force isn't unique at zero velocity. For small velocity range (order of cm/s), the behavior of MR damper is hysteretic rather than jump type non-linear function (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997). Figure 18 compares between the predicted and experimentally obtained responses for the Bingham Model.

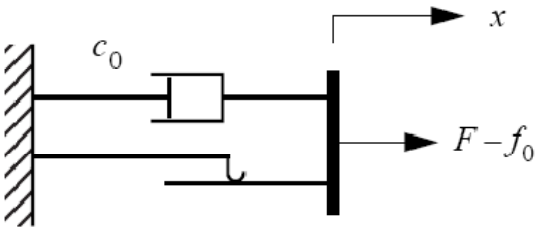


Figure 17: Bingham Model (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

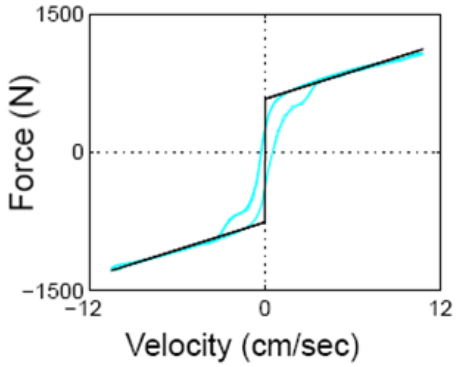


Figure 18: Comparison between the Predicted and Experimentally Obtained Responses for the Bingham Model (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

Due to the limitations of Bingham model, an extended Bingham model is proposed by Gamota and Filisko (1991) to improve the prediction of damper responses. The extended Bingham model consists of the original Bingham model in series with a standard linear solid model as shown in Figure 19. The force generated from this model is described in Equation 4.2:

$$\left.\begin{aligned} F &= k_1(x_2 - x_1) + c_1(\dot{x_2} - \dot{x_1}) + f_0 \\ &= c_0\dot{x_1} + f_c \operatorname{sgn}(\dot{x_1}) + f_0 \\ &= k_2(x_3 - x_2) + f_0 \end{aligned}\right\}, \quad |F| > f_c$$

$$\left.\begin{aligned} F &= k_1(x_2 - x_1) + c_1\dot{x_2} + f_0 \\ &= k_2(x_3 - x_2) + f_0 \end{aligned}\right\}, \quad |F| \leq f_c \qquad 4.2$$

Where:

c_0: Viscous damping coefficient

k_1, k_2 and c_1: Linear solid model parameters

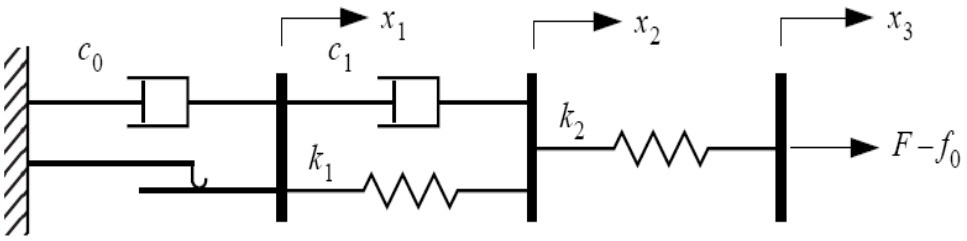


Figure 19: Extended Bingham Model (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

The extended Bingham model was found to more precisely represent the MR damper behavior. However, the fluid behavior around zero velocity is still not well produced. Furthermore, the force equations in the extended Bingham model are numerically challenging and require the use of very small time steps (around 10^{-6} s) during system simulations (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997). Figure 20 compares the force-velocity response of an MR damper using extended Bingham model and real experimental data.

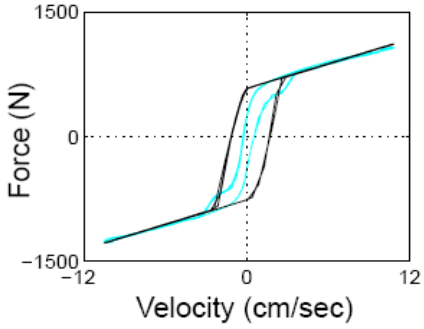


Figure 20: Force-Velocity Response of an MR Damper Using Extended Bingham Model and Real Experimental Data (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997).

Since the Bingham model and its extended version fail to represent the MR behavior around zero velocity and suffer from numerical problems, other parametric models, which are numerically traceable, were developed. For example, the Bouc-Wen model, developed by Wen, was used by Spencer Jr. et al to model the behavior of the MR dampers. The diagram of the Bouc-Wen model is shown in Figure 21. The equations that govern the Bouc-Wen model are described in Equations 4.3- 4.4.

$$F = c_0\dot{x} + k_0(x - x_0) + \alpha z \quad 4.3$$

$$\dot{z} = -\gamma|\dot{x}|z|z|^{n-1} - \beta\dot{x}|z|^n + A\dot{x} \quad 4.4$$

Where:

c_0: Viscous damping coefficient

k_0: Linear spring parameter

x_0: Initial displacement of spring k_0

α: Bouce-Wen parameter associated with the yield stress of the MR fluid

γ, β and A: Determine the linearity and transition smoothness of the non-linear function

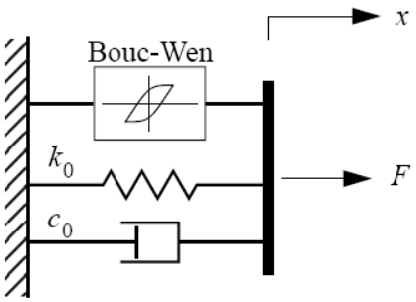


Figure 21: Bouc-Wen Model (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

The velocity-force behavior of the MR damper, generated by the Bouc-Wen model was similar to the behavior generated by the extended Bingham model. In other words, the fluid behavior around zero velocity is still not well produced. Figure 22 compares between the predicted and experimentally obtained responses for the Bouc-Wen model (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997).

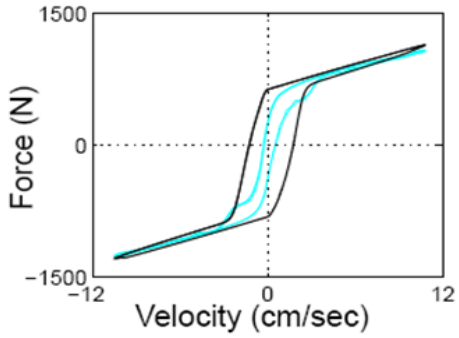


Figure 22: Force-Velocity Response of an MR Damper Using Bouc-Wen Model and Real Experimental Data (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

It is shown that all the previous parametric models weren't sufficient enough to produce the MR damper behavior near zero velocity. Therefore, further studies were carried out by various researches to produce an optimal model for MR dampers. Spencer Jr. et al. (1997) have successfully developed a phenomenological model (modified Bouc-Wen model) to describe the dynamic behavior of an MR damper. The model was capable of predicting the response of an MR damper over a wide range of loading under a constant or binary type of voltage (Chang, 1998). The phenomenological model is governed by Equations 4.5 :

$$f = c_1\dot{y} + k_1(x - x_0)$$

$$\dot{y} = \frac{1}{c_0 + c_1}\left[\alpha z + c_0\dot{x} + k_0\dot{(x - y)}\right]$$

$$\dot{z} = -\gamma|\dot{x} - \dot{y}|z|z|^{n-1} - \beta(\dot{x} - \dot{y})\,|z|^n + A(\dot{x} - \dot{y})$$

$$\alpha = \alpha_a + \alpha_b u$$

$$c_1 = c_{1a} + c_{1b} u$$

$$c_0 = c_{0a} + c_{0b} u$$

$$\dot{u} = -\eta(u - v) \qquad 4.5$$

Where: x and f are the displacement and the force generated by the MR damper, respectively; y in an internal displacement of the MR damper; u in the output of a first order filter and v is the commanded voltage sent to the current driver. In this model, the accumulator stiffness is represented by k_1, the viscous damping observed at large and low velocities are denoted by c_0 and c_1 respectively; k_0 control the stiffness and large velocities, x_0 is the initial displacement of spring k_1 associated with the nominal damper force due to the accumulator; γ, B and A are hysteresis parameters for the yield element and α is the evolutionary coefficient.

The schematic of the phenomenological model, proposed by Spencer Jr. et al., is shown in Figure 23. Figure 24 shows that this model accurately reproduces the force-velocity responses obtained experimentally (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

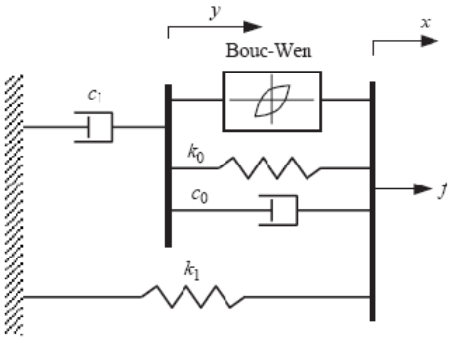


Figure 23: Phenomenological Model, Proposed by Spencer Jr. et al (1997)

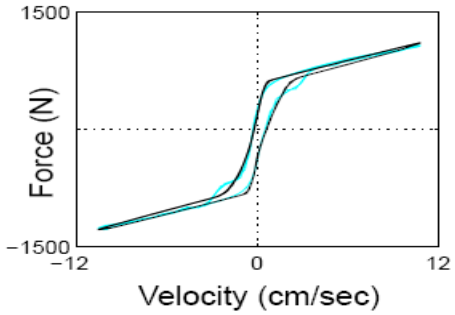


Figure 24: Force-Velocity Response of an MR Damper Using Modified Bouc-Wen Model and Real Experimental Data (Spencer, Jr., Dyke, S., Sain, M. & Carlson, J., 1997)

Although parametric models effectively characterize the MR damper, they are difficult to solve numerically and didn't count for magnetic field. Furthermore, parametric identification methods require assumptions for the structure of the mechanical model that will simulate the behavior of a system. Therefore, several non-parametric techniques were proposed and discussed in the following sections.

4.3.2 None Parametric Models

Due to the limitations of parametric models, several non-parametric models for modeling and identifying the MR damper were developed such as neural network, Neuro-fuzzy and Chebyshev polynomials.

Neural-networks have been able to accurately emulate the non-linear MR fluid behavior. For example, Chang and Roschke (1998) proposed a multi-layer perception (MLP) neural network for modeling the non-linear behavior of MR dampers. The neural network model consists of six input neurons, one output neuron and twelve neurons in the hidden layer. The sets of input-output data were generated using a parametric model proposed by Spencer Jr. et al. (1997). The training method selected was the Gauss-Newton based Levenberg-Marquardt training algorithm. After the network has been trained, the optimal brain surgeon (OBS) strategy was employed to remove unnecessary weights and optimize the network. Finally, they found that multi-layer Perceptron neural network (MLP) was able to produce a satisfactory representation of the dynamic behavior of the MR damper. However, the training of the network was time consuming and further studies are still required to validate this neural network model under laboratory or practical situations. Furthermore, Wang and Liao (2001) proposed the use of recurrent neural networks to identify two models for MR dampers; forward and inverse models. In the forward model, the damper force is predicted from the applied voltage while in the inverse model, the voltage is predicted from the output force. The designed recurrent neural network consists of three layers with fifteen input neurons, one output neuron and

fifteen neurons in the hidden layer. The sets of input-output data were generated using a parametric model proposed by Spencer Jr. et al. (1997). Then, the network is trained for both the forward and inverse models. The results showed the ability of the recurrent neural network to reasonably estimate and predict the non-linear behavior of the forward and inverse models of MR dampers. Another recurrent neural network was proposed by Yu.et.al (2005) to emulate the behavior of MR dampers. The proposed network consists of three layers, three input neurons, one output neuron and seven recurrent neurons in the hidden layer. The network was trained by means of recursive prediction error algorithm and using data generated from tests on an actual MR damper. Finally, the recurrent neural network model was compared with the experimental results. It was shown that the recurrent neural networks are efficient and reliable to emulate the behavior of MR dampers over a wide range of operating conditions.

Neuro-fuzzy models are another example of non-parametric models proposed for describing MR dampers' behavior. Schurter and Roschke (2000) developed a Neuro-fuzzy model based on the behavior of a set of non-linear differential equations of an MR damper. They used the Adaptive Neuro-Fuzzy Inference System (ANFIS) from the Matlab Fuzzy Logic Toolbox to determine the parameters required for the modeling of the damper. ANFIS combines both the back-propagation gradient descent and the least squares methods to train the data set. The proposed Neuro-fuzzy model was trained and validated using data generated from numerical simulation of non-linear differential equations. The resulting fuzzy interface model was shown to accurately represent the MR

damper behavior while reducing computational requirements. In addition, a fully trained fuzzy model is approximately 1000 time faster than the mathematical model. Furthermore, Takagi-Sugeno (T-S) fuzzy models are becoming powerful engineering tools for modeling complex dynamic systems. The T-S fuzzy model is described by fuzzy if-then rules that give local linear representation of the non-linear system. Du and Zhang (2006) and Askari and Davaie-Markazi (2008) applied (T-S) fuzzy models to approximate the dynamic behavior of an MR damper. The rule structure and the membership (MF) parameters were simultaneously developed by Genetic Algorithm (GA) to reduce the sum of square errors (SSE) between the predicted and target forces. The training and testing data were obtained from the phenomenological model proposed by Spencer Jr. *et al.* (1997). Using training and validation data, it was shown that the predicted force matches quite well with the target force. Therefore, the proposed T-S fuzzy model can satisfactory emulate the behavior of an MR damper.

Other example of non-parametric approaches is the use of Chebyshev polynomials to describe an MR damper. Ehrgott and Masri (1992) presented a nonparametric approach employing two dimensional orthogonal Chebychev polynomials to estimate the damper force using the damper displacement, velocity and acceleration information. They found that the predicted damper force is better described as a function of velocity and acceleration rather than a function of displacement and velocity. Gavin et al. (1996) also used Chebyshev polynomials to approximate the MR damper characteristics. However, the damper force was assumed to be a function of three

dimensional orthogonal Chebyshev polynomials; displacement, velocity and electric field strength. It was shown that the predicted MR force resembles the corresponding experimental data (Wilson, 2005). However, using Chebyshev polynomials to estimate the MR damper force requires a huge computational effort to determine the large number of Chebyshev coefficients. The force plot may also exhibit oscillatory behavior which is frequently observed for polynomial interpolation (ZAMM, Z., Butz, T. & Von Stryk, O., 1998)

In this thesis, an alternative presentation of MR damper behavior using GMDH networks is proposed. The GMDH network is applied to the problem of identification the forward and inverse models of MR dampers. The results of the proposed methodology are illustrated in the following chapter.

5 GMDH IDENTIFICATION OF MR DAMPERS

In this chapter, an application of GMDH to the modeling of the dynamic behavior of MR damper is presented. Both the forward and inverse models of a 2000N MR damper are obtained using the GMDH algorithm. Furthermore, the chapter outlines the steps for the development of the GMDH model. These steps involve the collection of sample training and testing data as produced by the target model, training the model using the implemented GMDH algorithm to relate the displacement, velocity and voltage signals to the force of the MR damper and validating of the new model through comparison of its output to the output of the target model, given identical inputs.

5.1 Data Collection and Description

Data for training and testing are obtained from the mathematical model of the MR damper proposed by Spencer Jr. et al. A set of typical parameters of the 2000N MR damper is presented in Table2.

Table2: 2000N MR Damper Parameters (Askari,M. & Davaie-Markazi, A., 2008)

Parameter	Value
c_{0a}(N.s/cm)	21.0
c_{0b}(N.s/cmV)	3.50
k_0(N/cm)	46.9
c_{1a}(N.s/cm)	283
c_{1b}(N.s/cmV)	2.95
k_1(N/cm)	5.00
x_0(cm)	14.3
α_a(N/cm)	140
α_b(N/cm V)	695
γ(cm^{-2})	363
B(cm^{-2})	363
A	301

table continues

Parameter	Value
η (sec^{-1})	190
N	2

A Matlab/Simulik model, obtained from (Askari,M. & Davaie-Markazi, A., 2008), is build to solve the MR damper non-linear equations. Figure 25 shows the Matlab/Simulink Model of a 2000N MR damper. The inputs of the model are the displacement, velocity and the commended voltage while the output is the force generated by the damper.

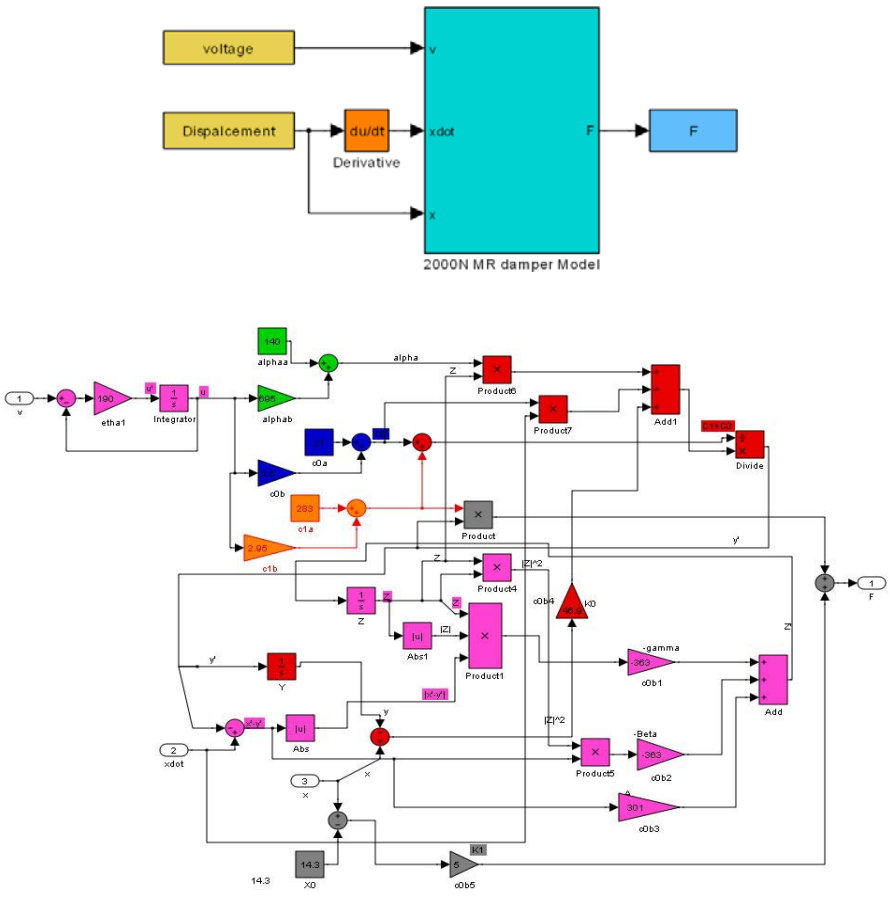


Figure 25: Matlab/Simulink Model for 2000N MR Damper (Askari,M. & Davaie-Markazi, A., 2008)

To produce a meaningful model of an MR damper, the input data must include information in the entire operating range of the system and cover the spectrum of operation in which the damper will function. Usually, the limits of the input signals are based on the characteristics and applications of the MR damper. Advance knowledge of the input signals enables the creation of more useful training data. In this thesis, the displacement, velocity and voltage data are obtained from (Askari,M. & Davaie-Markazi, A., 2008). The range of the voltage signal is between [0 2.5V] where 2.5 V represents the saturation voltage of the damper and is obtained experimentally. The saturation voltage (2.5 V) implies a voltage level at which no further increase in the yield strength of the damper is exhibited. Similarly, the displacement signal range is between ±2 cm. The velocity input is generated by taking the time derivative of the displacement. Matlab is used to solve the system of differential equations for a simulation time of 4 seconds. A time step of 0.0005 seconds is used to produce a total of 8000 sets of data. Figure 26 shows time histories of displacement, voltage and damper force.

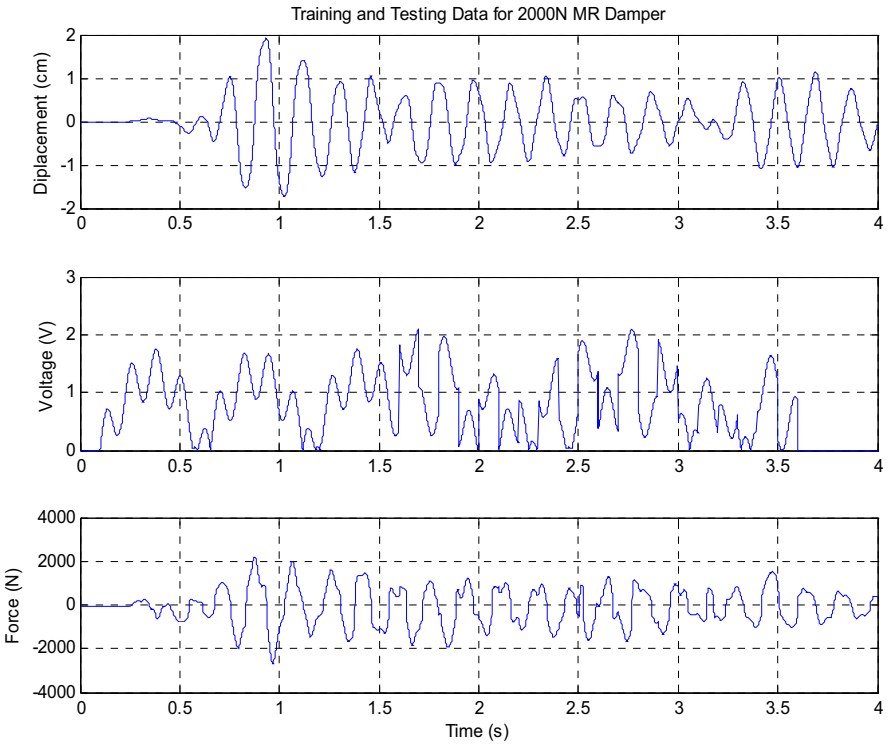


Figure 26: Training and Testing Data for 2000N MR Damper

5.2 Training of the Model

With establishing of the training and testing data, GMDH network is used to emulate the forward and inverse behaviors of the MR damper. The GMDH algorithm is implemented in Matlab. The corresponding GMDH codes are fully described in Appendix A

Before training can occur, a pre-defined threshold that is used as a criterion for selecting optimal network architecture should be defined. This threshold is defined by Equation 5.1

$$\text{Threshold} = (\min(R)*(1-\alpha) + (\alpha)*\max(R)) \quad 5.1$$

Where:

α is a user defined variable that ranges from 0 to 1

R is the regularity criteria described previously in Equation 3.2

The training of both the forward and inverse models of a 2000N MR damper is described in the following subsections.

5.2.1 Forward MR Model

The GMDH interface system is designed with 11 inputs and one single output. The inputs of the model are three displacements, three velocities, three voltages and two forces. More explicitly, to predict the force of the MR damper at time (t), the inputs are taken to be the displacements, the velocities and the voltages at times (t), (t-1) and (t-2) , and the forces at time (t-1) and(t-2). The output of the model is the damping force at time (t). Out of the generated 8000 data, 4000 data points are randomly selected for the training set while the remaining 4000 points are selected for the testing set. After several trials with various α's, α = 0.15 attains the best results of forward MR model. Table3 illustrates the effect of changing α on the accuracy of the GMDH network.

Table3: RMSE with Different Threshold Values

α	RMSE (N)
0.05	6.6402
0.07	6.5057
0.09	6.5057
0.1	6.5057
0.15	6.4312

The total computational time is found to be 40 seconds. After completing the GMDH training, the eleven input variables are reduced to 6 inputs and the irrelevant

input variables are automatically eliminated. The relevant inputs of the network are found to be the displacements at times (t) and (t-2), the voltages at times (t) and (t-2) and the forces and time (t-1) and (t-2). The final structure of forward MR damper consists of 6 inputs, 5 layers and one single output. The first and the second layers consist of 4 neurons, the third layer consists of 3 neurons while the last two layers consist of 2 and one neurons, respectively. The distribution of the minimum of the regularity criteria at each layer is shown in Figure 27. At the sixth layer, the minimum regularity criteria is larger than the fifth layer, therefore, training is stopped at the fifth layer. The RMSE of the obtained model was 6.4312 N. A comparison between the predicted and actual forces along with the prediction error is shown in Figure 28. It can be seen that the implemented GMDH network is able to predict the forward model of the MR damper.

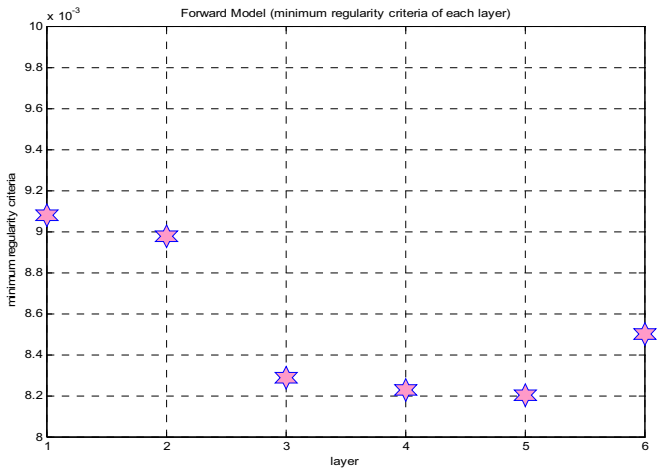


Figure 27: Distribution of Regularity Criteria of Each Layer

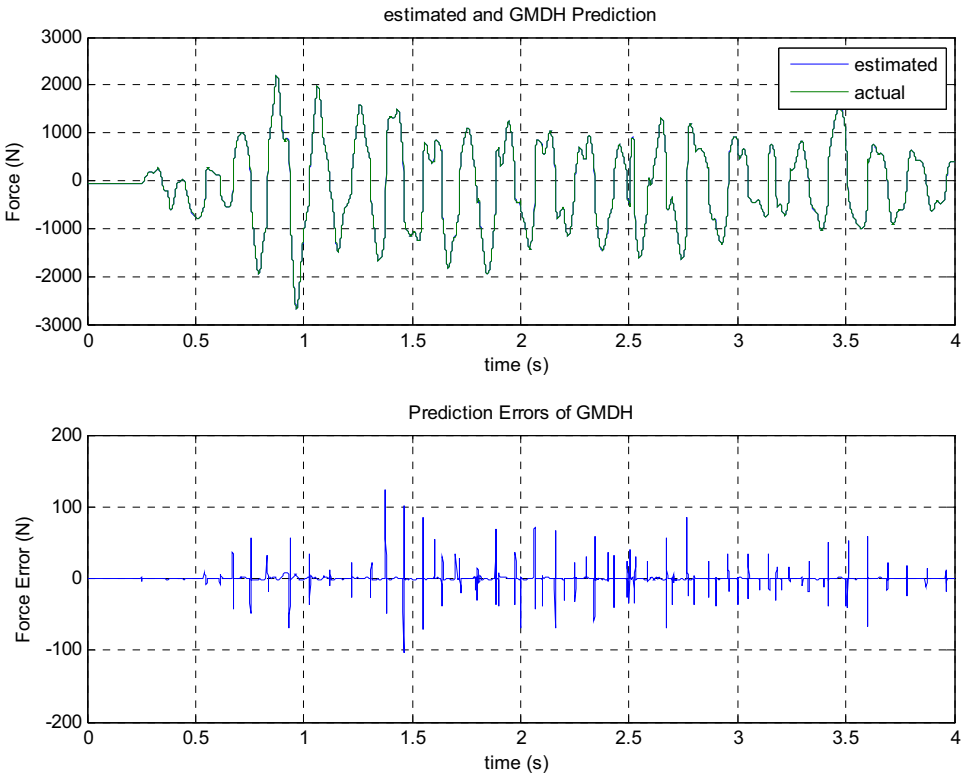


Figure 28: Target and Predicted Forces from GMDH and Prediction Error

5.2.2 Inverse MR Model

To predict the inverse MR model, another GMDH network with 11 inputs and one single output is implemented. The inputs consist of the displacements, the velocities and the forces at times (t), (t-1) and (t-2) and the voltages at times (t-1) and (t-2). The output of the model is the command voltage at time (t). After several trials with various α's, α = 0.05 attains the best results of inverse MR model. The total computational time is found to be 45 seconds. After completing the GMDH training, the eleven input variables are reduced to 6 input variables and the unnecessary input variables are automatically

eliminated. The relevant input variables are found to be the displacements at times (t) and (t-2), the velocity at time (t-1), the force at time (t-1) and the voltages at times (t-1) and (t-2). The final structure of the inverse model contains three layers; the first layer has 4 neurons, the second layer has 2 neurons while the last layer has one neuron. The RMSE of the inverse model was found to be 0.03V. A comparison between the predicted and actual voltages along with the prediction error is shown in Figure 29. It can be seen that the implemented GMDH network is able to predict the inverse model of the MR damper.

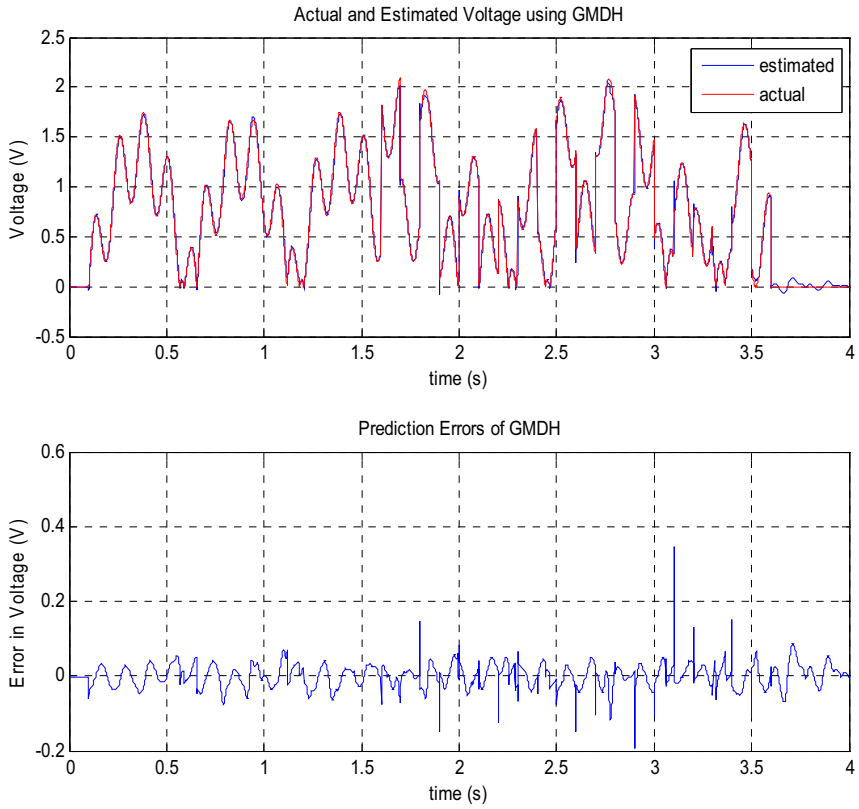


Figure 29: Target and Predicted Voltages from GMDH and Prediction Error

6 RESULTS AND DISCUSSIONS

In this chapter, the relative efficiency of GMDH modeling for a non-linear system is determined. The identification results of the GMDH are compared with other system identification approaches such as neural network and polynomial classifiers. Finally, a two step identification procedure is introduced to enhance the obtained results.

6.1 Neural Network Identification of MR Models

As discussed previously in Section 2.1.3 , system identification in neural network involves four steps which are data collection, selection of neural network architecture, model training and model validation. The data used for training the neural network are the same data used to train the GMDH network. For further details about the data collection and description, refer to Section 5.1. The selection of neural network architecture, model training and model validation for predicting MR damper forward and inverse models are discussed in the following subsections.

6.1.1 Forward MR Model

A feedforward neural network with 11 neurons in the input layer, 22 neurons in hidden layer and one neuron in the output layer is designed to identify the forward model of an MR damper. The 11 inputs of the neural network are the displacements, the velocities and the voltages at times (t), (t-1) and (t-2) and the forces at times (t-1) and (t-2). The output of the neural network is the damper force at time (t). The Matlab Neural Network toolbox is used to train the network. The training method was the Gauss-

Newton based Levenberg-Marquardt training algorithm. After the neural network is trained, the network is validated by calculating the root mean square error (RMSE) between the target and estimated forces. The obtained RMSE using the designed network was 17.0568 N while the training time of the network was around 3 minutes. Figure 30 shows the target and predicted force from the designed neural network and the prediction error between the target and estimated forces. From Figure 30, both the target and the predicted forces show a very similar trend. However, there are several peaks that correspond to large errors. The peaks imply that the network model under the designed conditions cannot effectively emulate the MR behavior. On the other hand, there are several regions where the error is very small. Hence, in such regions the MR damper behavior is quite well represented.

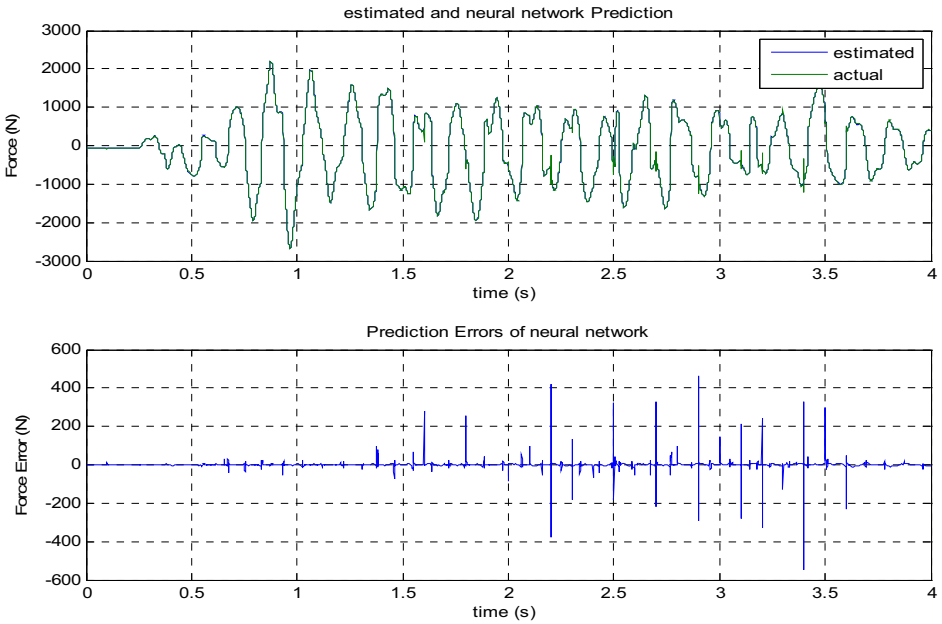


Figure 30: Target and Predicted Forces from Neural Network and Prediction Error

6.1.2 Inverse MR Model

To predict the inverse MR model, another feedforward neural network with 11 neurons in the input layer, 22 neurons in hidden layer and one neuron in the output layer is designed. The 11 inputs of the neural network are three displacements, three velocities, three Forces and two voltages. More explicitly, to predict the voltage of the MR damper at time (t) , the inputs are taken to be the displacements, the velocities and the forces at times (t), (t-1) and (t-2) and the voltages at times (t-1) and (t-2). The training method was the Gauss-Newton based Levenberg-Marquardt training algorithm. After the neural network is trained, the network is validated by calculating the RMSE between the target and estimated voltages. The obtained RMSE using the designed network was 0.08V while the training time of the network was around 6 minutes. Figure 31 shows the target and predicted voltages from the designed neural network and the prediction error between the target and estimated voltage.

From Figure 31, it is clearly shown that the target and the predicted voltages are similar to each other except at some edges where the prediction error is large. At such edges, the inverse model of the MR damper isn't well described.

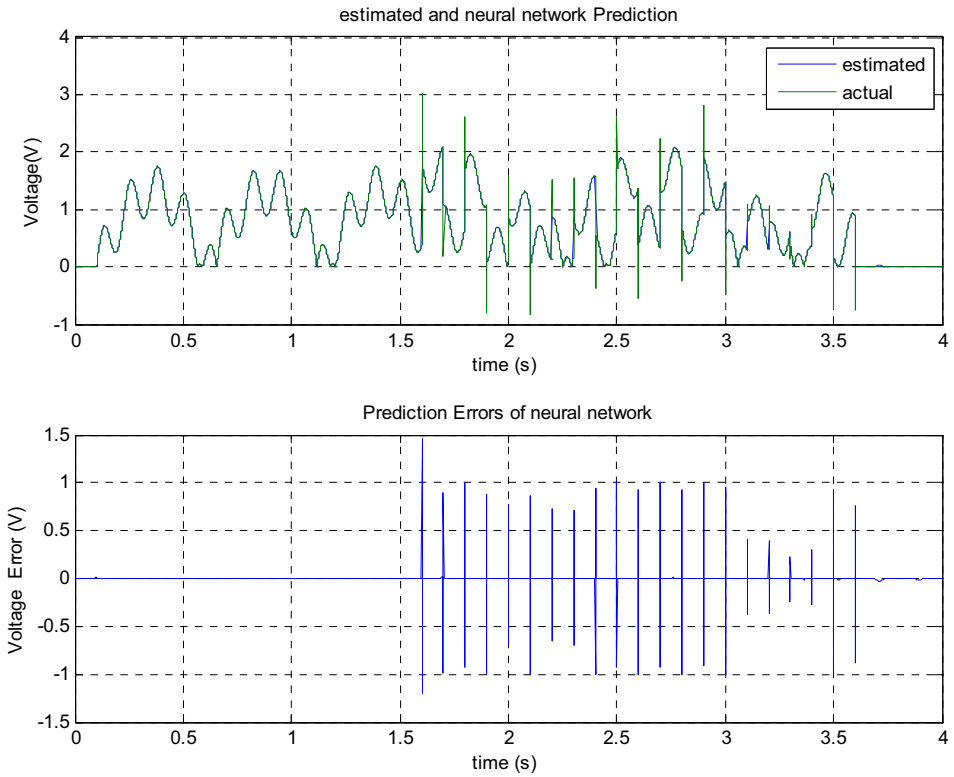


Figure 31: Target and Predicted Voltages from Neural Network and Prediction Error

6.2 Polynomial Classifiers Identification of MR Models

6.2.1 Forward MR Model

Polynomial classifiers are used to identify the forward model of the MR damper. The designed polynomial networks have 11 inputs and one output. The inputs are the displacements, the velocities and the voltages at times (t), (t-1) and (t-2) and the forces at times (t-1) and (t-2) while the output is the force at time (t). The polynomial classifier model is programmed using Matlab and the implemented code is described in Appendix

B. The polynomial classifier models were constructed using 1st, 2nd and 3rd polynomial functions. Then, each polynomial network was evaluated against the testing set. Table4 summarizes the results of modeling the MR damper using polynomial classifiers in terms of modeling accuracy and training time. Figure 32, Figure 33 and Figure 34 show the target and predicted forces and the prediction error between the target and estimated forces from polynomial classifiers of 1st, 2nd and 3rd orders, respectively.

Table4: Polynomial Classifiers Results (Forward Model)

Polynomial Order	RMSE (N)	Training Time (s)
1	23.7636	0.005955
2	22.5286	0.094768
3	335.5272	0.912515

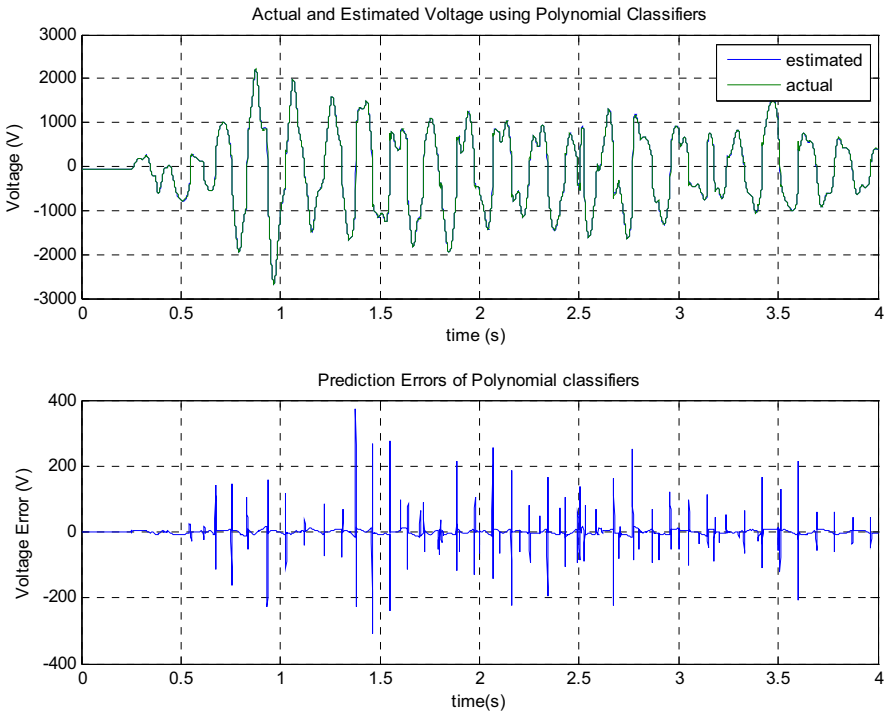


Figure 32: Target and Predicted Forces from 1st Order Polynomial and Prediction Error

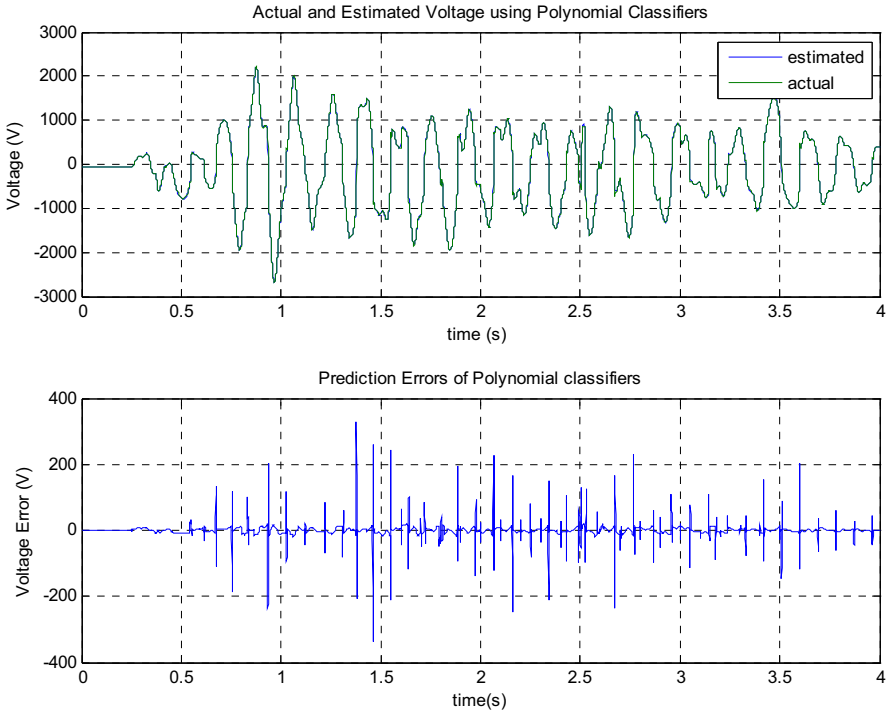


Figure 33: Target and Predicted Forces from 2nd Order Polynomial and Prediction Error

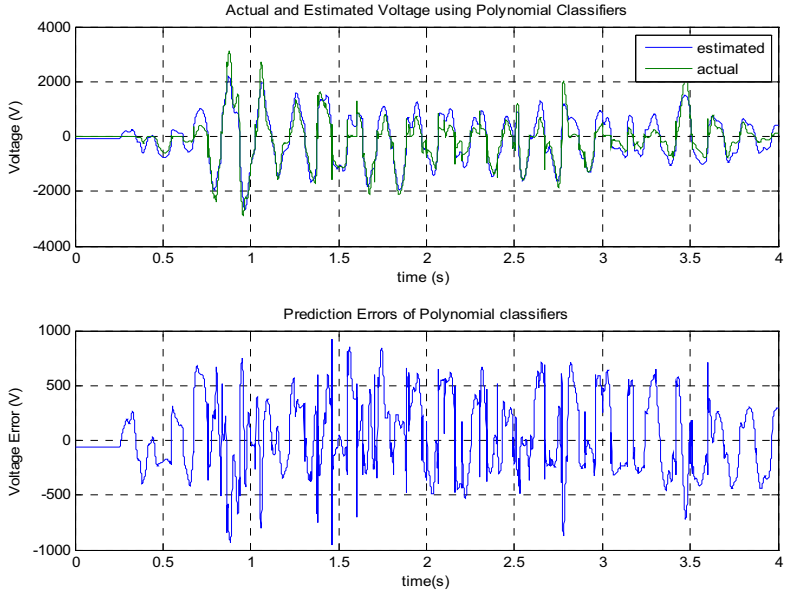


Figure 34: Target and Predicted Forces from 3rd Order Polynomial and Prediction Error

6.2.2 Inverse MR Model

Polynomial classifiers are used to obtain the inverse model of the MR damper. The designed polynomial networks have 11 inputs and one output. The inputs are the displacements, the velocities and the forces at times (t), (t-1) and (t-2) and the voltages at times (t-1) and (t-2) while the output is the voltage at time (t). The polynomial networks are constructed using 1^{st}, 2^{nd} and 3^{rd} polynomial functions. Then, each polynomial network was evaluated against the testing set. Table5 summarizes the results of modeling the MR damper using polynomial classifiers in terms of modeling accuracy and training time. Figure 35, Figure 36 and Figure 37 show the target and predicted voltages and the prediction error between the target and estimated voltages from polynomial classifiers of 1^{st}, 2^{nd} and 3^{rd} orders, respectively.

Table5: Polynomial Classifiers Results (Inverse Model)

Polynomial Order	RMSE (V)	Training Time (s)
1	0.0666	0.0507848
2	0.1271	0.081487
3	0.3607	0.919898

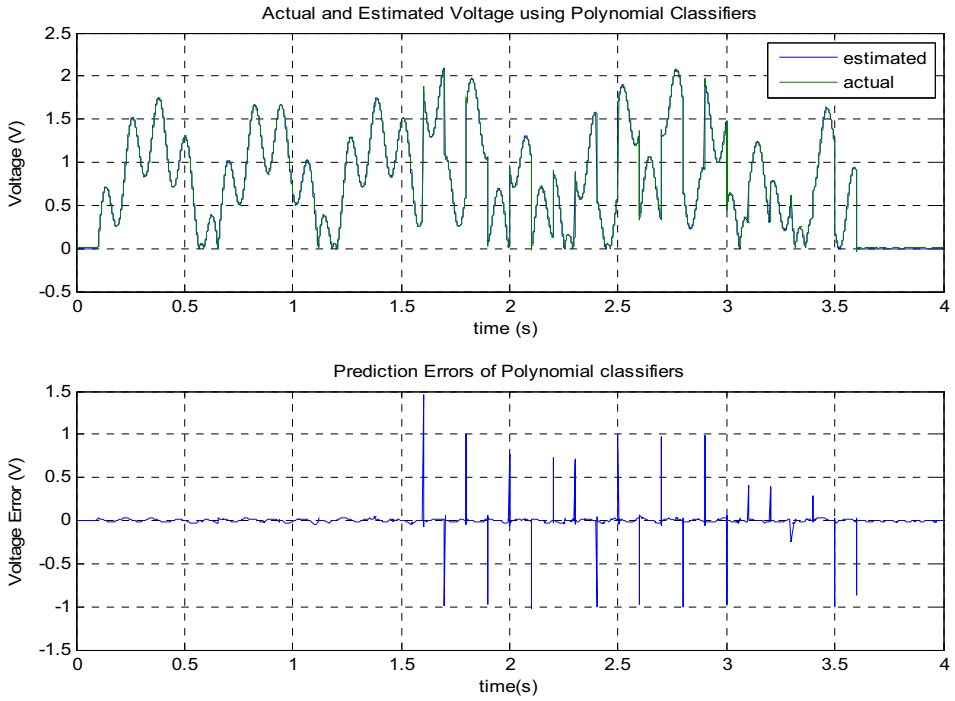


Figure 35: Target and Predicted Voltages from 1st Order Polynomial and Prediction Error

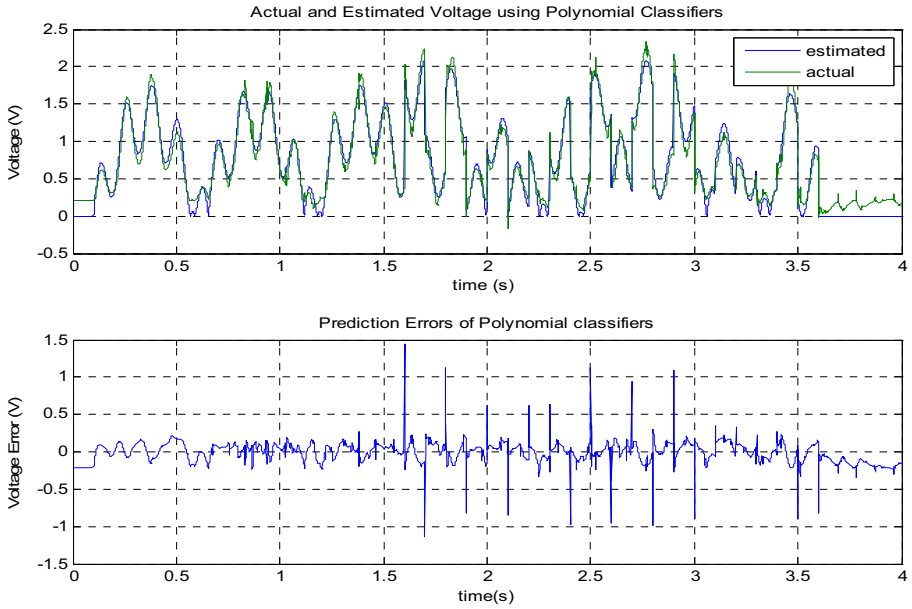


Figure 36: Target and Predicted Voltages from 2nd Order Polynomial and Prediction Error

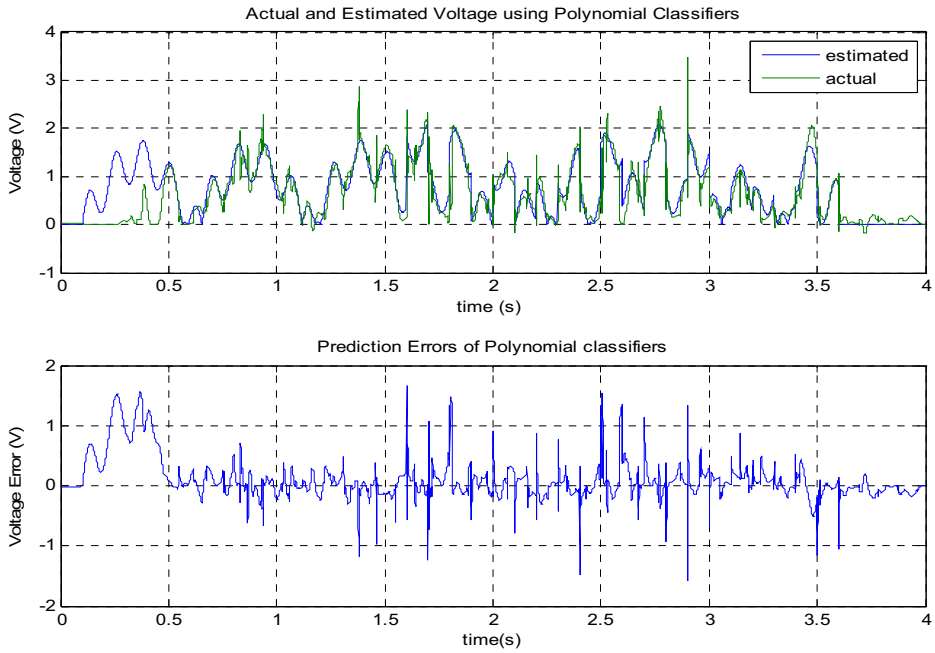


Figure 37: Target and Predicted Voltages from 3rd Order Polynomial and Prediction Error

6.3 Two Step Identification Procedure

To improve the results obtained from GMDH networks and reduce the training time, two-step identification procedure is implemented. In the first step, the best features of the input data were selected using forward and backward stepwise regression features selection. Then, in the second step, the best features are trained using GMDH network. The block diagram of the proposed two-step identification procedure is shown in Figure 38.

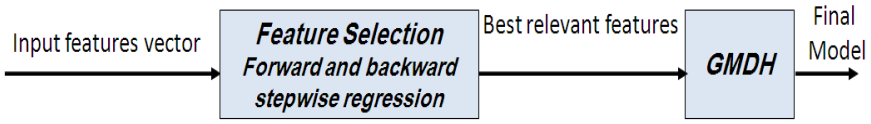


Figure 38: Two-Step Identification Block Diagram

The two step identification procedure was applied to the identification of the forward model of the MR damper. The best relevant feature vectors were found using *stepwisefit* matlab function that implements the forward and backward stepwise regression. Then, the obtained features were trained and tested using the GMDH networks. The RMSE of the obtained GMDH was found to 4.74 N instead of 6.43 N in the case of GMDH network alone. Figure 39 shows the target and predicted forces using two step identification procedures.

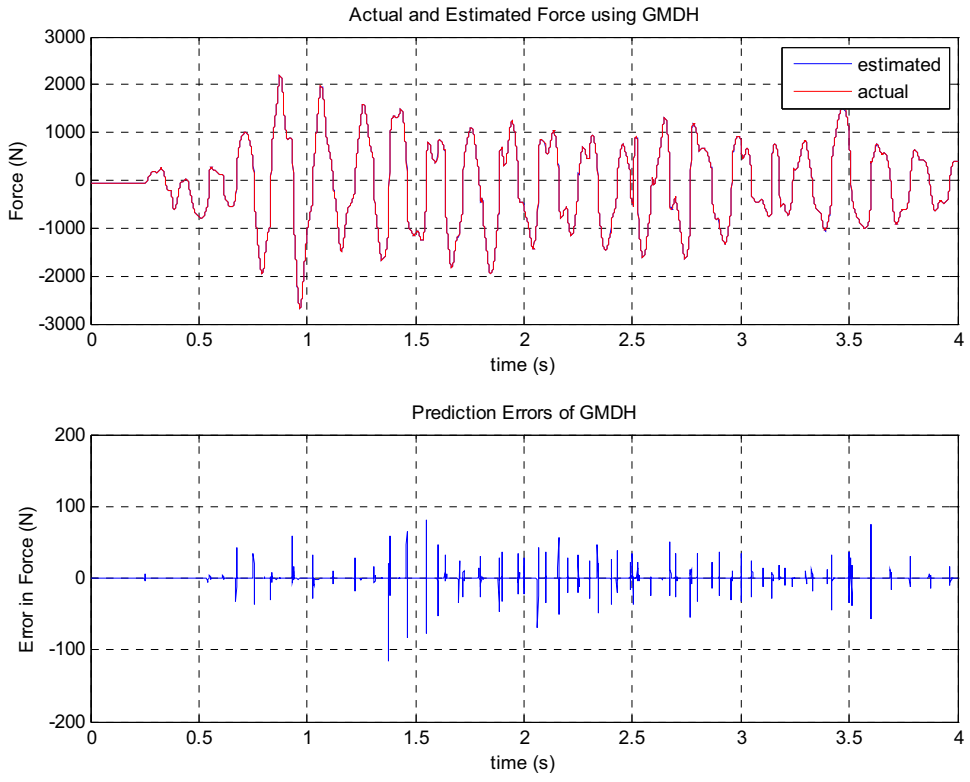


Figure 39: Target and Predicted Forces Using Two-Step Identification Procedure

However, when the two step identification procedure was applied to model the inverse behavior of the MR damper, the RMSE between the predicted and actual voltages was found to be 0.045 V.

Another way of implementing the two step identification procedure is to select the best features from the output of the best performing neurons from each layer in the GMDH network. Figure 40 shows a flow chart of two-step identification procedure within the GMDH network.

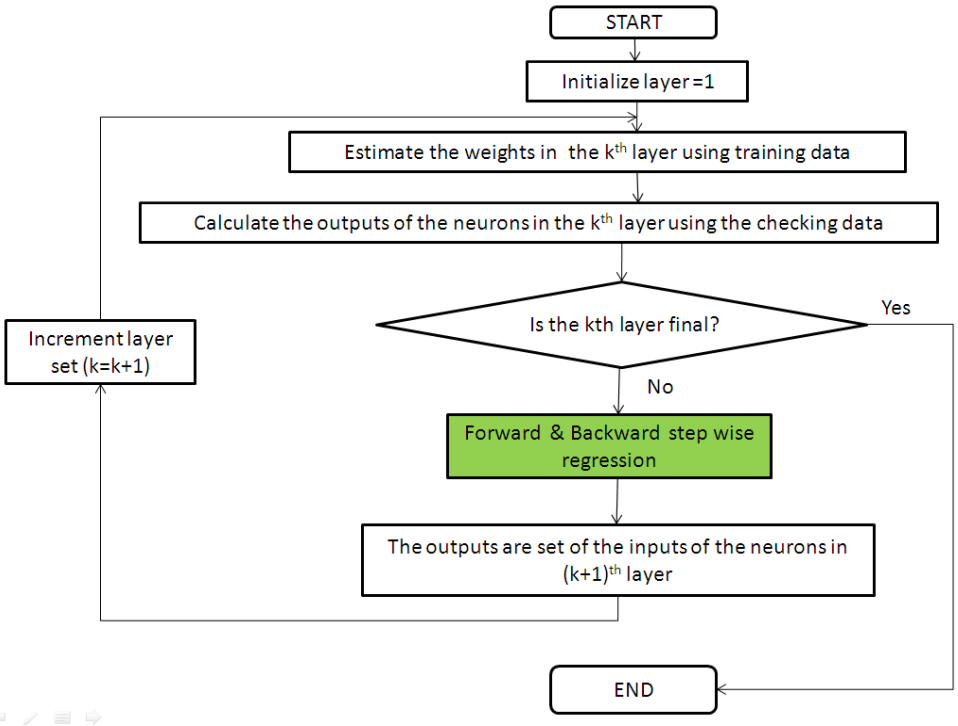


Figure 40: Two-Step Identification Procedure within the GMDH Network

Both the forward and inverse models of the MR dampers are estimated using two-step identification procedure within the GMDH network. The RMSE between the actual

and predicted forces and voltages are found to be 5.14 N and 0.05 V, respectively. Figure 41 and Figure 42 show the target and predicted forces and voltages using the two-step identification within the GMDH network, respectively.

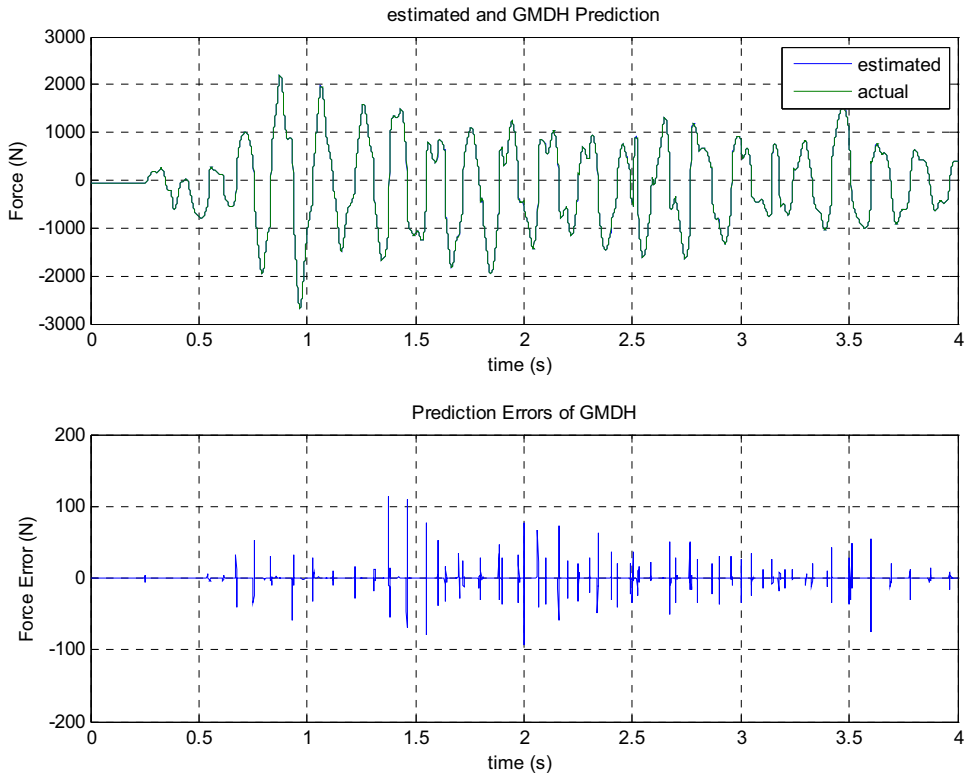


Figure 41: Target and Predicted Forces Using Two-Step Identification Procedure within the GMDH Network

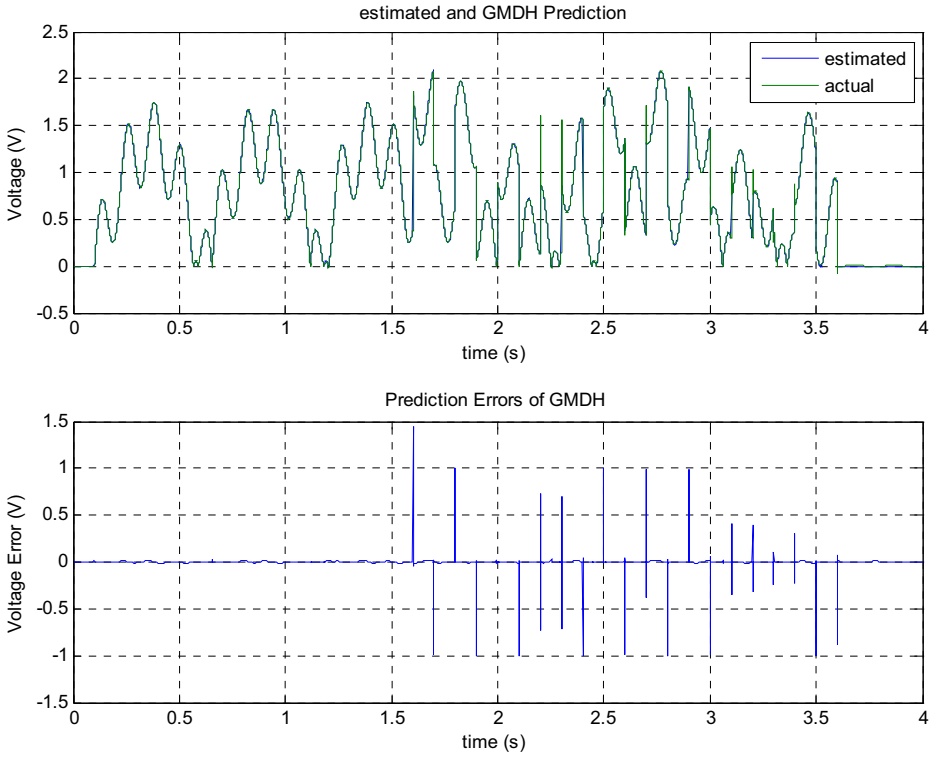


Figure 42: Target and Predicted Voltages Using Two-Step Identification Procedure within the GMDH Network

6.4 Two- Tier Identification

A two- tier identification procedure is also implemented to identify the behavior of MR damper. In the first tier, the model parameters are estimated using the GMDH network and then used to evaluate the estimated output in both the training and testing sets. In the second tier, the estimated output on the training and testing sets are concatenated with the training and testing sets, respectively. Then, the parameters of the GMDH model are re-estimated to obtain a final GMDH model of both the forward and

inverse models of MR dampers. Figure 43 shows a block diagram of two-tier identification.

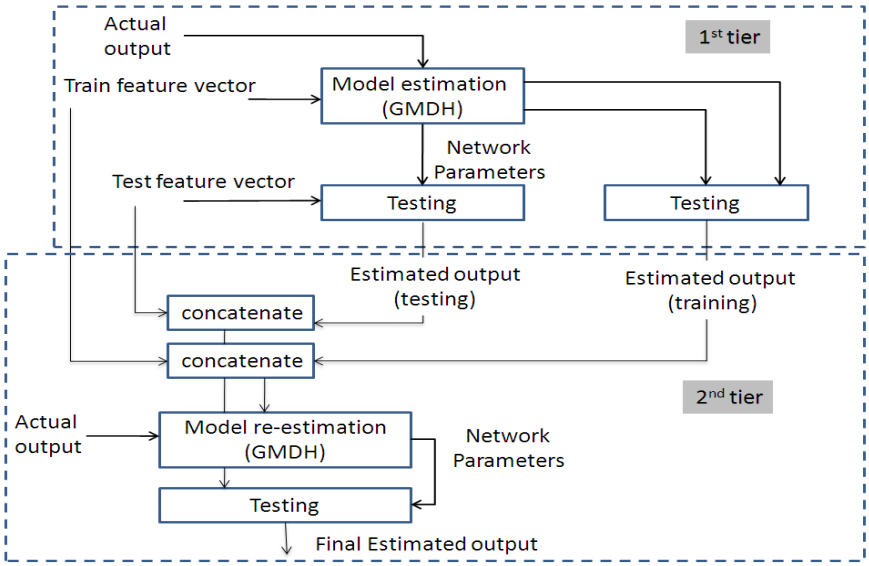


Figure 43: Block Diagram of Two-Tier Identification

The forward and inverse MR dampers are estimated using two-tier identification. In the forward model, the RMSE between the estimated and actual forces was found to be 4.81 N while in the inverse model, the RMSE between the estimated and actual voltages was found to be 0.016 V. Figure 44 and Figure 45 shows the targeted and predicted forces and voltages, respectively.

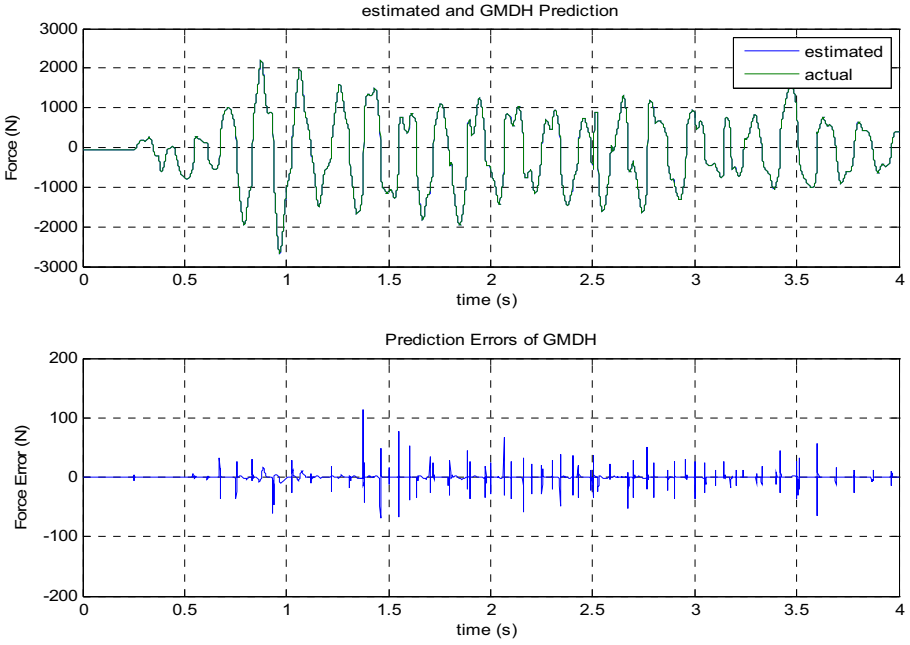


Figure 44: Target and Predicted Forces Using Two-Tier Identification

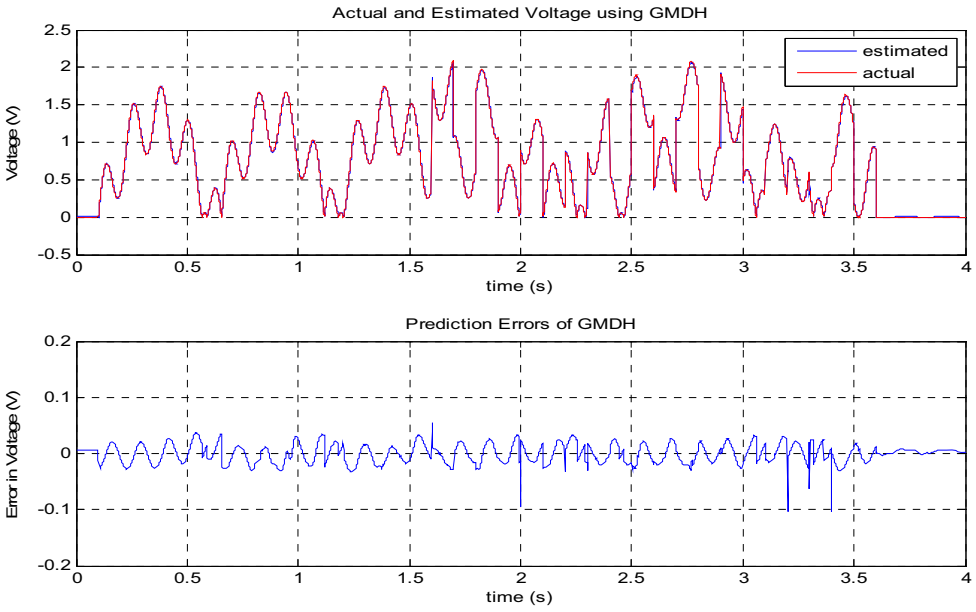


Figure 45: Target and Predicted Voltages Using Two-Tier Identification

6.5 Two-Step Identification and Two- Tier Identification (Combined Approach)

To combine the advantages of both two-step identification and two-tier identification, both techniques are applied to estimate the forward and inverse models of MR dampers. The resulting forward model was capable of estimating the MR damper force with a RMSE of 4.63 N. Figure 46 show the actual and estimated forces from the implemented combined approach.

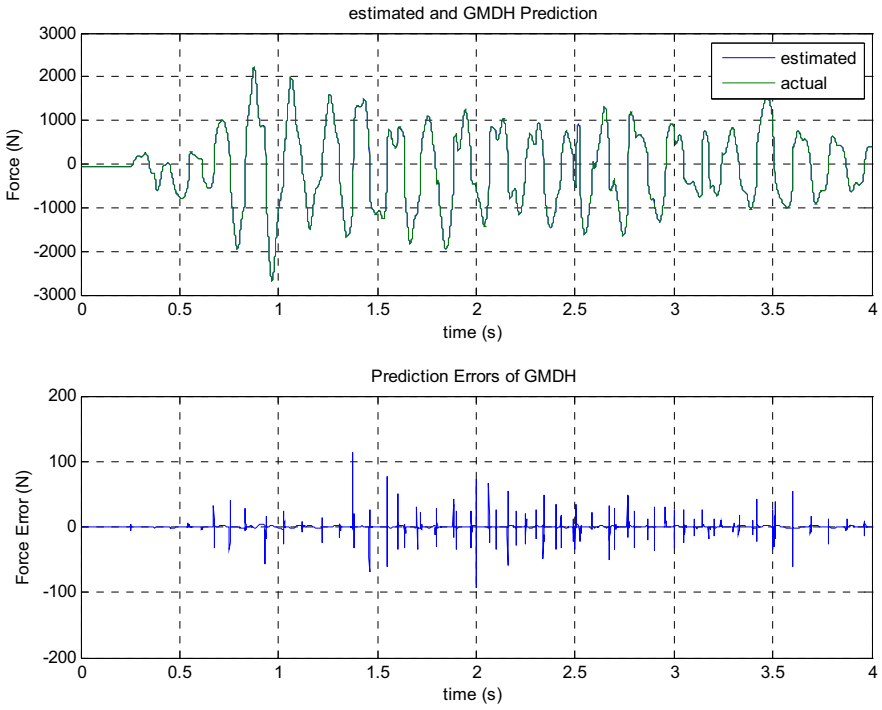


Figure 46: Target and Predicted Forces Using Two-Step Identification and Two-Tier Identification (Combined Approach)

On the other hand, when applying the combined approach in identifying the inverse model of the MR damper, the proposed identification technique failed to predict the voltage of the MR damper. The RMSE between the actual and predicted voltages was found to be 0.06 V. Figure 47 shows the actual and estimated voltages from the implemented combined approach.

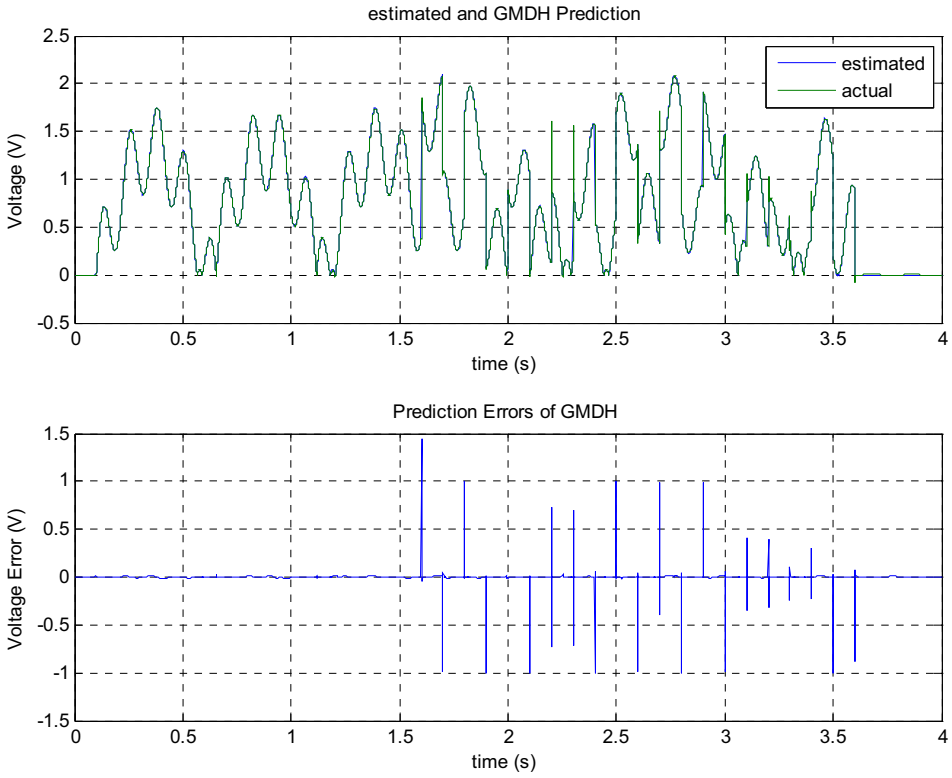


Figure 47: Target and Predicted Voltages Using Two-Step Identification and Two- Tier Identification (Combined Approach)

6.6 Summary of Results

The criterion, for the comparisons of different modeling approaches is the root mean square (RMS) error of the predicted outputs from each model. Figure 48 and Figure 49 show the comparison of the results of the different modeling methods applied to identification of forward and inverse models of a 2000N MR damper, respectively.

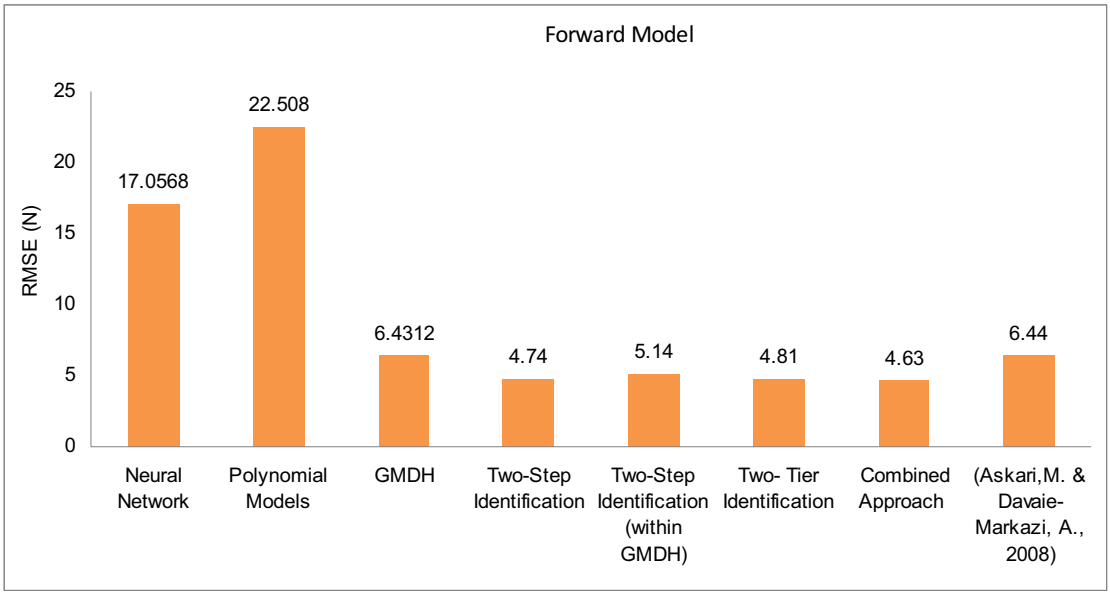


Figure 48: Comparison of the Results of the Different Modeling Methods Applied to Identification of a Forward MR Model

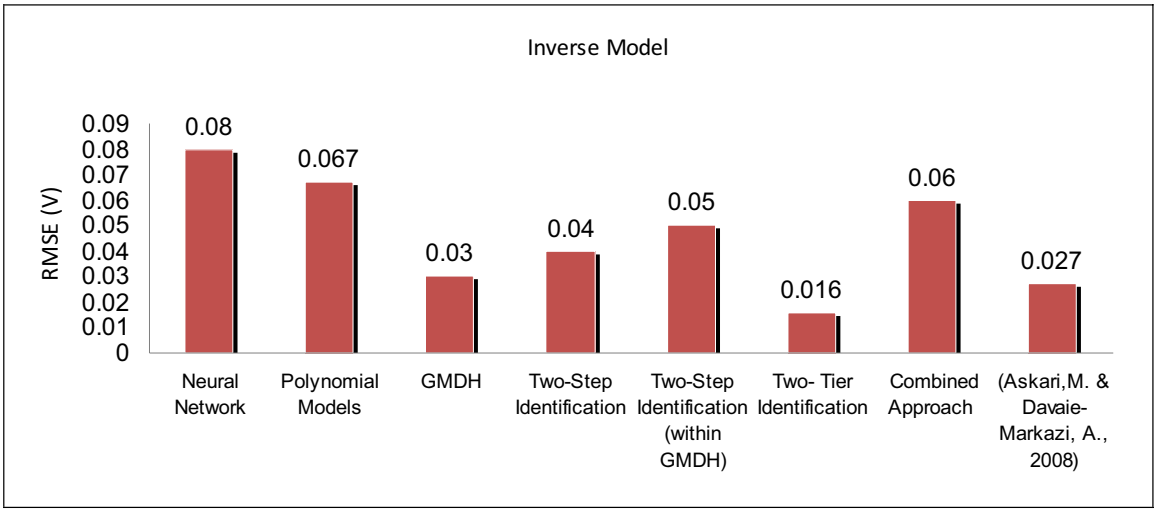


Figure 49: Comparison of the Results of the Different Modeling Methods Applied to Identification of an Inverse MR Model

As indicated in Figure 48, the best identification technique for modeling the forward behavior of the MR damper was the combined approach of two-step identification and two-tier identification while the worst identification technique was polynomial models. Furthermore, Figure 49 shows that the best identification technique for modeling the inverse behavior of the MR damper was the two-tier identification while the worse identification technique was neural networks.

7 CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

In this thesis, the effectiveness of GMDH networks in modeling complex systems is shown. The GMDH algorithm is implemented and applied to non-linear system identification. The key feature of the GMDH network is that its structure isn't predefined as in most of the existing neural networks. Instead, it is automatically adjusted during model development. Furthermore, GMDH networks can model complex systems without having specific knowledge of the systems. The trained GMDH networks decide which of the possible inputs are actually relevant to a certain output. Furthermore, the trained GMDH networks give a polynomial relation between system output and relevant input variables. The GMDH algorithm has been tested on the identification of the forward and inverse models of MR dampers. Results of identifying MR damper models show an excellent performance of the network in comparison to the existing neural network and polynomial models. They also clarify that the GMDH networks can precisely model the forward and inverse models of MR dampers.

7.2 Future Work

The following recommendations are made for future research and development as follow-on to the work reported in this thesis:

- Apply hybrid modeling approaches where the GMDH network is combined with other modeling techniques such as fuzzy logic and neural network (GMDH-NF) to the identification of MR damper models.

- Apply evolutionary methods for designing generalized GMDH-type networks to identify MR damper models. Such generalization of network's topology provides optimal networks in terms of number of layers and/or number of neurons and their connectivity.
- Apply GMDH algorithm on real laboratory experiments and practical situations.
- Study the effect of noise with various signals to noise ratios (SNRs) on modeling the MR damper behavior.

REFERENCE LIST

Anastasakis, L. &. (2001). *The development of self-organization techniques in modeling: a review of the group method of data handling (GMDH).* ACSE Research Report No 813, University of Sheffield, UK.

Askari,M. & Davaie-Markazi, A. (2008). APPLICATION OF A NEW COPMACT OPTIMIZED T-S FUZZY MODEL TO NONLINEAR SYSTEM IDENTIFICATION. *Proceeding of the 5th International Symposium on Mechatronics and its Applications (ISMA08).* Amman, Jordan.

Atray, V., & Roschke, P. N. (2003). Design, Fabrication, Testing, and Fuzzy Modeling of a Large Magnetorheological Damper for Vibration Control in a Railcar. *Proceedings of the Joint ASME/IEEE Conference on Railroad Technology, .* Chicago, Illinois, .

Belle, R. U. , Wesley, H., Baofu, L., Xuedong, H. , Rosani, L. P., Sergio, R. P., & Ke, Z. (2004). *On-Line Monitoring and Diagnostics of the Integrity of Nuclear Plant Steam Generators and Heat Exchangers.* research report No. DE-FG07-01ID14114/UTNE-07, The University of Tennessee, Nuclear Engineering Department.

Chang, C. &. (1998). Neural Network Modeling of a Magnetorheological Damper. *Journal of Intelligent Material Systems and Structures , 9* (9), 755-764.

Dai-Hua,W., & Wei-Hsin, L. (2001). Neural network modeling and controllers for magnetorheologicalfluid dampers. *The 10th IEEE International Conference on Fuzzy Systems*, *3*, pp. 1323-1326. Melbourne, Vic., Australia.

DJAJAKESUKMA, S. , SAMALI, B., LI, J. & NGUYEN, H. (2002). Modeling and identification of MR damper for semi-active stiffness damper. *Processing International Conference in Motion Vibration Control*, *6*, pp. 1197-1202.

Ehsan, N., Saeed, B. & Vesal, H. (2004). Evolutionary GMDH-based Identification of Building Blocks for Binary-Coded Systems. *2nd IEEE International Conference on Information & Communication Technologies (ICTTA'06)*, (pp. 1900-1904). Damascus, Syria.

Galina, S. (2008). THE FUZZY-NEURO CLASSIFIER FOR DECISION SUPPORT. *International Journal of Information Theories & Applications , 15*.

Haiping, D., & Nong, Z. (2006). Evolutionary Takagi-Sugeno Fuzzy Modelling for MR Damper. *Proceedings of the Sixth International Conference on Hybrid Intelligent Systems, 2006, HIS*, (p. 6:69).

Ivakhnenko, A. G. (1968). The Group Method of Data Handling-A rival of the Method of Stochastic Approximation. *Soviet Automatic Control , 13* (3), 43-55.

Kondo, T. (1998). GMDH neural network algorithm using the heuristic self-organization method and its application to the pattern identification problem. *Proc. of the 37th SICE Annual Conference, International Session Paper*, (pp. 1143-1148).

Kondo, T. (2004). Revised GMDH-type neural networks using PSS criterion. *The 2004 47th Midwest Symposium on Circuits and Systems (MWSCAS '04)*, *3*, pp. iii- 81-4.

Kondo, T. (1998). The learning algorithms of the GMDH neural network and their application to the medical image recognition. *Proc. of the 37th SICE Annual Conference, International Session Paper*, (pp. 1109 – 1114).

Kondo, T., Ueno, J. & Kondo, K. (2005). Revised GMDH-Type Neural Networks Using AIC or PSS Criterion and their Application to Medical Image Recognition. *Journal of Advanced Computational Intelligence and Intelligent Informatics , 9* (3), 257-267.

Korbicz, J. & Mrugalski, M. (2008). Confidence estimation of GMDH neural networks and its application in fault detection systems. *International Journal of Systems Science , 39*, 783-800.

Kung, S. (1993). *Digital Neural Networks.* NewJersy: Prentic-Hal.

Liao, C., Wang, K.,Yu, M.,& Weimin, C. (2005). Modeling of magneto-rheological fluid damper employing recurrent neural networks. *Proceedings of 2005 International Conference on Neural Networks and Brain*, *2*, pp. 616-620.

Liu, G. P. (2001). *Nonlinear identification and control: a neural network approach.* New York: Springer.

Ljung, L. (1987). *System Identification: Theory for the User.* New Jersey: Prentice-Hall.

Mohannad, K. & Shaik.K. (2001, October). fuzzy and neuro-fuzzy modeling and control of non-linear systems . *Proceeding for second international conference on Electrical & Electronics*, (pp. 197-201). Bursa, Turkey.

Mrugalski, M., Witczaka,M. & Korbicza, J. (2008). Confidence estimation of the multi-layer Perceptron and its application in fault detection systems. *Engineering Applications of Artificial Intelligence , 21*, 895-906.

Müller, J. A. & Lemke, F. (1999). SELF-ORGANIZING DATA MINING BASED ON GMDH PRINCIPLE. *ESIT ´99, European Symposium on Intelligent Techniques.* Greece.

Murphy, S. *Development of Neural Networks for System Identification.* B.ENG. IN ELECTRONIC ENGINEERING, , Dublin City University, Dublin.

Nelles, O. (2001). *Non-Linear System Identification.* New York: Springer.

Onwubolu, G. C. (2007). Design of Hybrid Differential Evolution and Group Method of Data Handling for Inductive Modeling . *Proceedings of International Workshop on Inductive Modeling*, (pp. 87-95). Prague, Czech.

Park, H. O. (August,1999). A study on multi- layer fuzzy polynomial inference system based on an extended GMDH algorithm. *Proceedings for IEEE International Fuzzy Systems Conference*, *1*, pp. 354 – 359.

Rahim, N.A.,Taib,M.N, Adom, A.H & Mashor, M.Y. (n.d.). Retrieved DECEMBER 2008, from http://publicweb.unimap.edu.my/~norasmadi/conference/ICoMMS2006.pdf

Roger, J. (1993). ANFIS: Adaptive Network -Based fuzzy inference systems. *IEEE Transactions on Systems, Man, and Cybernetics , 23* (03), 665-685.

Sakaguchi, A., Yamamoto, T., Fujii, K., & Monden, Y. (2004). Evolutionary GMDH-based identification for nonlinear systems. *2004 IEEE International Conference on Systems, Man and Cybernetics*, *6*, pp. 5812 – 5817.

Schurter, K., & Roschke, P. N. (2000). Fuzzy Modeling of a Magnetorheological Damper Using ANFIS. *Proceedings of the IEEE Fuzzy 2000 Conference.* San Antonio, Texas.

Spencer, Jr., Dyke, S., Sain, M. & Carlson, J. (1997). Phenomenological Model of a Magnetorheological Damper. *ASCE J. Eng. Mech* , 230-238.

Toh, K. , Tran, Q. & Srinivasan, D. (2004). Benchmarking a Reduced Multivariate Polynomial Pattern Classifier. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE , 26* (6).

Vlachos, D. & Tolias, Y.A. (2003). Neuro-Fuzzy modeling in bankruptcy prediction. *Yugoslav Journal of Operations Research , 13* (02), 165-174.

Water, P.R, Kerckhoffs,J.H, & Welden,D.V. (2000). GMDH-based dependency modeling in the identification of dynamic systems. *European Simulation Multi-conference (ESM)*, (pp. 211-218).

Wilson, C. (2005). *Fuzzy Control of Magnetorheological Dampers for Vibration Reduction of Seismically Excited Structures.* PhD research, Florida State University, Department of Civil and Environmental Engineering.

ZAMM, Z., Butz, T. & Von Stryk, O. (1998). Modelling and Simulation of Electro- and Magnetorheological Fluid Dampers. *Math. Mech , 78*, 1–22 .

Zhao, X.M, Song, Z. & Ping,L. (October, 2002). A novel NF-GMDH-IFL and its application to identification and prediction of non-linear systems. *Proceedings for 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, *3*, pp. 1286- 1289.

Appendix A

A. GMDH MATLAB CODES

Appendix A GMDH Matlab Codes

Appendix A1 gmdh Main code

```
clear
clc
close all

% load training and testing features and targets and time samples
load MR.mat

% MR.mat includes:
% inputs --> training feature vector
% targets --> targets feature vector (training)
% tests --> testing feature vector
% Ytests --> targets feature vector (testing)
% t --> time samples

% Initialization:
% Wf --> final weights
% poly_coff_f --> best selected polynomial in each layer
% alpha --> threshold value
Wf=[];poly_coff_f=[];
alpha = input('Enter your threshold value (alpha)');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%% GMDH Implementation%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% R_min --> minimum measure in each layer
% Ym --> output from each neuron
% iter+1 --> number of layers
[R_min,poly_coff,W,Ym, Ym2]= gmdh_proposed(inputs, tests, targets, Ytests, alpha);

iter=0;
i=1;
R_min=[R_min 0];
[a,b]=size(Ym);
Wf=[Wf W];
poly_coff_f=[poly_coff_f ;poly_coff];

%%%% obtaining all adalines for other layers
while (R_min(i+1)<R_min(i) && ~(isempty(Ym))&& b>1 ) %--> stopping criteria

[R_min(iter+2),poly_coff,W,Ym,Ym2]= gmdh_proposed(Ym,Ym2, targets,Ytests, alpha);
   Wf=[Wf W];
   poly_coff_f=[poly_coff_f ;zeros(1,2);poly_coff];

   iter=iter+1
   [a,b]=size(Ym);
   if ( iter >1)
```

```
        i=i+1;
    end
end

if (poly_coff_f(end)==0)
    cof = find(poly_coff_f(:,1)==0);
    if ~isempty(cof==1)
        poly_coff_f=poly_coff_f(1,:);
        Wf= Wf(:,1);
    else

        poly_coff_f=poly_coff_f(:cof(end-1)+,:);
        B=size(cof);
        C=size(poly_coff_f);
        Wf = Wf(:,1:C(1)-B(1)+1);
    end
end

Yf= net_eval(poly_coff_f,Wf,iter,inputs);
Yf2=net_eval(poly_coff_f,Wf,iter,tests);


RMSE1 = sqrt(sum((Yf-targets).^2)/length(targets))
RMSE2 = sqrt(sum((Yf2-Ytests).^2)/length(Ytests))

t=0:0.0005:4;

figure
subplot(211),plot(t,Yf2); hold on,plot(t,Ytests,'r')

xlabel('time (s)'),ylabel('Voltage (V)'),title('Actual and Estimated Voltage using GMDH');
legend('estimated','actual')
grid
e1=Yf-targets;
e2=Yf2-Ytests;

subplot(212),plot(t,e2);
hold on,
xlabel('time (s)'),ylabel('Error in Voltage (V)'), title('Prediction Errors of GMDH);
grid
```

Appendix A2 gmdh algorthim

```
function [R_min,k,W,Ym, Ym2]= gmdh_proposed(data, datatest,Ytrain, Ytest, alpha)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%This code is the implementation of the algorithm
for GMDH %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

b=size(data);

% Generate the indexes that corresponds to all two inputs combination of
% the features

k=nchoosek(1:b(2),2);
c = size(k);
j=1;
R=zeros(1,c(1));

 for i=1:c(1)
   % obtain the 2 inputs combination data
   com = data(:,k(i,:)); com2 = datatest(:,k(i,:));

   % expand the feature vector into a 2nd order polynomial
   expand_comb = feat_mat2poly_mat(com,2);
   expand_comb2 = feat_mat2poly_mat(com2,2);

   % estimate the weighing coefficient using least square method
   % (training set)
   W= pinv(expand_comb'*expand_comb)*expand_comb'*Ytrain;

   %compute the measure
   R(i)= sqrt(sum((Ytest-expand_comb2*W).^2)/sum(Ytest.^2));
 end

clear com com2 expand_comb expand_comb2 W

% obtain the best features
k=k(find(R<=(min(R)*(1-alpha) + (alpha)*max(R))),:);

if (isempty(k)==0)
   c=size(k);
   j=1;
   C=repmat({zeros(1,b(2))},c(1),4);
   for i=1:c(1)
      com=data(:,k(i,:));
      com2 = datatest(:,k(i,:));
      expand_comb = feat_mat2poly_mat(com,2);
```

```
        expand_comb2 = feat_mat2poly_mat(com2,2);
        W= pinv(expand_comb'*expand_comb)*expand_comb'*Ytrain;
        Ym2=expand_comb2*W;
        Ym=expand_comb*W;
        R= sqrt(sum((Ytest-Ym2).^2)/sum(Ytest.^2));
        C(i,:)={W Ym2 Ym R};
        clear com expand_comb com2 expand_comb2 W R Ym Ym2
    end
end
clear data Ytrain alpha
W=cell2mat({C{:,1}}); Ym2=cell2mat({C{:,2}});
Ym=cell2mat({C{:,3}}); R=cell2mat({C{:,4}});
R_min = min(R);
clear C
```

Appendix A3 gmdh network evaluation

```
function Yf= net_eval(poly_coff_f,Wf,iter,test)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%perform the network and test it on the testing data%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

t=1;
i=0;
l=size(poly_coff_f);

if (l(1)~=1)
    % Perform the gmdh_network
    [g,t]= gmdh_net(test, poly_coff_f,Wf,t,i);

    while ( iter >1)
        i=i+1;
        [g,t]= gmdh_net(g, poly_coff_f,Wf,t,i);
        iter = iter-1;
    end

    % Final output from the formed gmdh_network
    Yf= feat_mat2poly_mat(g(:,poly_coff_f(t+i+1,:)),2)*Wf(:,t);

else
    Yf = feat_mat2poly_mat(test(:,poly_coff_f),2)*Wf(:,1);
end

clear g t poly_coff_f Wf iter test
```

Appendix A4 gmdh network construction

```
function [g,t]= gmdh_net(data, poly_coff_f,Wf,t,i)
m=1;
class = size(poly_coff_f);

while((poly_coff_f(t+i,:)~=[0 0]))
   g(:,m)= feat_mat2poly_mat(data(:,poly_coff_f(t+i,:)),2)*Wf(:,t);
   t=t+1;
   m=m+1;
   if ((t+i)> class(1))
     poly_coff_f(t+i,:)= [0 0];
   end
end
clear data poly_coff_f Wf i
```

Appendix A5 gmdh Feature selection main code

```
function [features_1, features_2] = gmdh_best_features(training, testing, targets, tests, alpha )
% inputs
   % training: training feature vectors
   % testing:  testing feature vectors
   % targets:  targets for training data
   % tests :   targets for testing data
   % alpha :   threshold
%outputs
   %features_1: best features in the training data
   %features_2: best features in the testing data

poly_coff_f=[];
[R_min,poly_coff,W,Ym, Ym2]= gmdh_proposed(training, testing, targets, tests, alpha);
iter=0;
i=1;
R_min=[R_min 0];
[a,b]=size(Ym);
poly_coff_f=[poly_coff_f ;poly_coff];

while (R_min(i+1)<R_min(i) && ~(isempty(Ym))&& b>1 )
   [R_min(iter+2),poly_coff,W,Ym,Ym2]= gmdh_proposed(Ym,Ym2, targets,tests, alpha);
   poly_coff_f=[poly_coff_f ;zeros(1,2);poly_coff];
   iter=iter+1;
   [a,b]=size(Ym);
   if ( iter >1)
      i=i+1;
   end
end

if (poly_coff_f(end)==0)
   cof = find(poly_coff_f(:,1)==0);
   if ~isempty(cof==1)
```

```
      poly_coff_f=poly_coff_f(1,:);
   else

      poly_coff_f=poly_coff_f(1:cof(end-1)+1,:);
   end
end

index = gmdh_feature_select(poly_coff_f);
features_1 = training(:, index);
features_2 = testing(:,index);
```

Appendix A6 gmdh Feature selection function

```
function index = gmdh_feature_select(poly)

z=find (poly(:,1) == 0);
iter = length(z);
location = [z(1); diff(z)]-1;

polyn=poly(end-location(end)-1:end-2,:);
index=polyn(poly(end,:),:);
index=unique(index(:));

for i=2:iter-1
   polyn= poly(z(end-i)+1 : z(end-(i-1))-1,:);
   index=polyn(index,:);
   index=unique(index(:));
end

polyn = poly(1: z(end-(iter-1))-1,:);
index=polyn(index,:);
index=unique(index(:));
```

APPENDIX B

B. POLYNOMIAL MODEL MATLAB CODES

Appendix B Polynomial Model Matlab codes

```
clear
clc
close all

% load training and testing features and targets and time samples
load MR.mat

% MR.mat includes:
% inputs --> training feature vector
% targets --> targets feature vector (training)
% tests --> testing feature vector
% Ytests --> targets feature vector (testing)
% t --> time samples

% Initialization
order = input ('Enter the order of the polynomial');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Polynomial Classifiers%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  expand_input = feat_mat2poly_mat(inputs,order);
  W = pinv(expand_input'*expand_input)*expand_input'*targets;
  expand_output = feat_mat2poly_mat(tests,order);
  Yf2=expand_output*W;
  Yf=expand_input*W;
  RMSE1 = sqrt(sum((Yf-targets).*(Yf-targets))/length(targets))
  RMSE2 = sqrt(sum((Yf2-Ytests).*(Yf2-Ytests))/length(Ytests))

figure
subplot(211),plot(t,Yf2); hold on,plot(t,Ytests,'r')

xlabel('time (s)'),ylabel('Voltage (V)'),title('Actual and Estimated Voltage using polynomial networks');
legend('estimated','actual')
grid
e1=Yf-targets;
e2=Yf2-Ytests;

subplot(212),plot(t,e2);
hold on,
xlabel('time (s)'),ylabel('Error in Voltage (V)'), title('Prediction Errors of Polynomial networks');
grid
```

VITA

Yasmeen Abu Kheil was born on December 7, 1982, in Ras Al Khaimah (RAK), UAE. She was educated in local public schools and graduated from high school in 2001. She received a scholarship from his Highness Sheikh Dr. Sultan Bin Mohammed Al-Qassimi to, Ruler of Sharjah, to study at American University of Sharjah, from which she graduated with a bachelor degree of electrical engineering with a magna cum laude, in 2005. Then, she worked as an assistant Engineer at the Gulf Pharmaceutical Industries for one year. In 2006, Ms. Yasmeen Abu Kheil began a master's program in Mechatronics Engineering at the American University of Sharjah where she served as a graduate research assistant till 2008. Currently, she is working at Institute of applied technology as a lab engineer.