CHAPTER 1

INTRODUCTION

Artificial neural networks or neural networks for short are defined as a system consisting of a set of processing elements (neurons) that are connected by connections known as synapses or weights to form fully connected networks. Neural networks are nonlinear in nature as the neurons usually consist of nonlinear functions. Hence, they are capable of learning and identifying nonlinear relationships. These attributes make neural networks an ideal nonlinear modeling tool in many areas of science, including control engineering.

1.1 Background

A Radial Basis Function Neural Network (RBF network) is one of feed forward neural networks architecture besides commonly used Multilayer Layer Preceptor (MLP), which has good generalization performance especially in the nonlinear system identification (Sundarajan, N et al., 1999, Fung, C.F et al., 1996, Huang, G.B et al., 2004). RBF networks have simpler networks interpretation compared to MLP (Sundarajan, N et al., 1999; Fung, C.F et al., 1996; Huang, G.B et al., 2004). Thus, the learning process can be revealed explicitly.

RBF networks have become popular among scientists and engineers since it has the capability of addressing and representing many statistical techniques (Paggio and Girosi, 1990). It is really helpful in many purposes especially when mathematics or statistics understanding are needed in representing complex nonlinear system. RBF networks also have been used successfully in a number of applications such as time series prediction, speech recognition, and adaptive control (Sundarajan, N et al., 1999). Thus, many literatures in application of RBF networks and also some

fundamental research regarding RBF networks have been reported recently (e.g., Jun, Y., and Meng J. Er., 2008). The term, learning method, is one of the most important components in any form of neural network architecture, including RBF networks. The learning or training of neural network (also known as neural computing in many research and development communities) is mostly left unnoticed by many researchers who always prefer to stick to the slow gradient decent based back-propagation which is prone to stuck in local minima. The learning theory researchers try to find the most effective learning method to train a particular network (which includes RBF networks). Learning method is actually a process which tunes the parameters inside the networks so that it could represent a black-box system which the process is commonly known as the system identification process in control research communities hence it has become one of the most important issue to be discussed in this thesis.

1.2 Problem Statement

An effective learning method is also needed to train RBF networks as the RBF networks are characterized by Radial Basis Function type, a real valued function whose value depends on the distance from the origin, as the activation function. Thus, the initialization center and width as the nonlinear parameters become crucial in constructing learning algorithm for RBF networks (Roy, A et al., 1997).

Moreover, the updating process (updating the RBF network weights) is also an important issue when comes to learning processes thus, need to be taken into consideration to develop an effective learning algorithm for RBF networks in system identification.

System identification process is not always with constantly sample data. Therefore, learning methods on system with lost packets needs a derivative free algorithm.

1.3 Research Objective

This research will look into more on fundamental research on learning behaviour of nonlinear network such as RBF network. This thesis aims to introduce a new form of learning or training approach for RBF Networks using derivative free methods. Some mathematical reasons also will be presented to support fundamental theory which is related to the simulation results. The whole research objectives in this thesis can be summarized into as:

1. To present the importance of initialization on RBF network learning process.
2. To apply Finite Difference on Nonlinear parameter update.
3. To apply the decomposed Network to reduce the complexity of learning method.
4. To present extensive simulation results on fixed size RBF learning techniques using finite difference for both regular sample data and irregular sample data with the influence of lost packets
5. To present some mathematics fundamental theory to support the simulation.

1.4 Scope of Research

The research narrows its scope in proposing learning methods which can be summarized as:

1. Proposing learning algorithm on fix RBF networks
2. Proposing learning algorithm in recursive manner.
3. Focusing on SISO identification application either with regular sample data or with irregular sample data.
4. Gaussian is the only considered activation function.

1.5 Research Methodology

The research work flow representations of the phases that have evolved during this investigation are as follows:

First Phase:

- Reviewing literature regarding RBF network, learning method for RBF, the effectiveness of Extreme Learning Machine (ELM) methods – a recent proposed technique which only update the output neuron connected weights with linear optimization techniques. The ELM technique will be used as the comparison to all proposed techniques discussed in this research work.

- Investigating the effect of Gaussian centre placement for RBF networks on general to system identification. The ELM approach uses random Gaussian kernel placement thus other techniques such as K-means and Fuzzy C-means clustering are investigated on ELM to evaluate the influence of parameter initialization process to the overall performance.

Second Phase:

- Reviewing recent work on derivative free gradient estimate based learning using central finite difference approach. Adopting the finite difference for estimating the gradient using second order Recursive Prediction Error (RPE) on nonlinear neuron associated parameters (centers and width) update.

- Applying finite difference RPE by decomposing the RBF network to neuron level (decomposed RBF), thus reduces the memory requirement for online weight update. Test the viability of decomposed RBF network using derivative free gradient estimate.

Third Phase:

- Introduce the concept of lost packets in which the single recursive (online) input data to the nonlinear network (RBF Network) assumed to be irregularly sampled rather than with constant sample time. The derivative gradient estimate is viewed as the only near optimal solution for updating weights for

system with lost packets (or irregularly sampled). The term lost packets meant a packets consist of the current input (say at time *t*) and current output with time stamp will be bundled to packets and transmit to the identifier of the system with probability of dropping.

1.6 Simulation Set up

There are five nonlinear test benchmarks problems used to evaluate some of the proposed techniques discussed in this thesis. The RBF network training approaches are evaluated using fixed size prior set Gaussian basis function with recursive (online) strategies being employed. The term recursive strategies means the data feed to the network one by one rather than a batch form.

System Identification involves in this research work identifying the dynamic nonlinear system online. The work involves real-time simulation using Matlab® , Simulink™ using recursive S-function for identifying nonlinear functions.

1.7 Research Contribution

The objective of this research work is to propose new learning algorithm strategies using derivative free technique by adopting finite difference approach on RBF networks for system identification. The contribution of thesis with relevant research articles can be summarized as follows.

1. Presenting the important of weights initialization on fixed size RBF network prior to learning process.

    - Nur Afny C. Andryani, Vijanth Sagayan Asirvadam, "Modified Extreme Learning Machine for Fixed RBF Network on Time Series Problem," *Proceedings of 2nd International Conference on Science and Technology, Pulau Pinang, Malaysia,* 12th-13th December 2008.

    - Nur Afny C. Andryani, Vijanth S. Asirvadam, N.H Hamid, "Some Modifications on Online Extreme Learning Machine for Fixed Radial

Basis Function Network," *Proceedings of National Postgraduate Conference, Universiti Teknologi PETRONAS,* 25th-26th March 2009

2. Applying derivative free techniques using finite difference for nonlinear parameter update on RBF network. Simulation results shows the derivative free gradient estimate can perform comparably to physically derived gradient estimate which not only time consuming but applicable to only one-step ahead prediction

    - Nur Afny C. Andryani, Vijanth S. Asirvadam, N.H Hamid, "Finite Difference Approach for Nonlinear Parameter Update on Fixed Radial Basis Function Network's Learning Method," *Proceedings of International Graduate Conference on Engineering and Sciences, Johor, Malaysia,* 23th-24th December 2008.

3. Presenting extensive simulation results on fixed size RBF learning techniques using derivative free technique for irregular sample data (with the influence of loss packets).

    - Nur Afny C. Andryani, Vijanth S. Asirvadam, N.H Hamid, "Finite Difference Approach on RBF Networks for On-line System Identification with Lost packets," *Proceeding of 2009 International Conference on Electrical Engineering and Informatics, Selangor, Malaysia,* 5th-7th August 2009.

4. Applying Decomposed Network to reduce the memory and computation cost for training RBF network with larger hidden neurons for system identification with irregular sample time.

    - Nur Afny C. Andryani, Vijanth S. Asirvadam, N.H Hamid, "Finite Difference Recursive Update on Decomposed RBF Networks for System Identification with Lost packets," *Proceeding of International Conference on Soft Computing and Pattern Recognition 2009, Melaka,* 4th-7th December 2009.

5. Presenting that Finite Difference on Decomposed RBF Networks can be the solution for the whole case in terms of accuracy, stability and convergence rate

1.8 Thesis Organization

The thesis is organized into six chapters where the Chapter 1 describes the introductory text about this research work. This chapter also describes the research objective and the methodology taken for the completion of this report.

Chapter 2 gives a brief description about Neural Network which focuses on the Radial Basis Function (RBF) with supporting mathematical theory on its universal approximation capability. In depth understanding of RBF network is being described as the thesis focus only on the training of RBF recursively.

Chapter 3 of this thesis will look into the influence of initialization Gaussian kernel parameters (centers and width) to the RBF network training and final convergence. An approach known as *Extreme Learning Machine* (ELM) which randomly assigned Gaussian kernels is used as benchmark comparison to the other methods which assigned Gaussian centers using intelligence clustering techniques.

Chapter 4 will elaborate on the adaptation of gradient free estimate using finite difference approaches in deducing the recursive learning algorithms for RBF networks. This chapter also will study a new variant of recursive learning which decomposed to neuron level. The finite difference based gradient estimate involves weights which are nonlinear-in-parameter which is the centre and the width of Gaussian kernel associated to each neuron. The influence of weight initialization also looks into recursive learning using simulation studies.

Chapter 5 will be looking at the feasibility of gradient free estimate using finite difference in estimating the response of benchmark problems based on constantly time sampled or irregular time sampled system. A thorough evaluation studies being made in order to look into the effectiveness of finite difference on system with lost packets problems.

Chapter 6 concludes this report by revisiting some of the chapters and its distinct contributions. Some future direction also being shown for possible further research work on the neural network based recursive learning for system with irregular sample input or termed as ‘lost packets’ in this report. Appendix in the end included to describe the test benchmark problems used in this thesis.

CHAPTER 2

RBF NETWORKS: AN OVERVIEW

In this chapter literature review regarding the RBF Network and its learning capabilities and some mathematics background of learning and optimization process are elaborated.

2.1 Artificial Neural Network

Humans tried to develop artificial intelligence by imitating biological neuron network then it is called as Artificial Neural Network (ANN). It represents an interconnection among neuron which consists of several adjustable parameters which are tuned using a set of learning examples to obtain the desired function representing the actual system (Du, K.L et al., 2006; Billings, S.A et al., 1992). Similar to the biological neuron, it provides adaptability to change its environment by learning which can be supervised, reinforcement and unsupervised.

Basically the core points of ANN are its architecture and its learning algorithm. Based on its architecture, ANN can be divided as feed forward neural network, recurrent neural network, and its combination. The supervised learning, reinforcement learning, and unsupervised learning are types of its learning algorithm. There are two kinds of ANN which are most popular in feed forward neural network. They are MLP (Multi Layer Preceptor) and RBF (Radial Basis Function) Networks.

2.2 Radial Basis Function Networks

Radial Basis Function is one of special function as its response decreases or increases monotonically with distance from the origin (center). An RBF variants or

types can be any continuous function which forms the basis function in RBF networks structure. There are several basis functions or kernels of Radial Basis Function which may guarantee the nonlinear mapping that is needed in the RBF networks learning process (Gupta, M et al., 2003) as:

- Gaussian Radial Basis Function

$$\varphi(x) = \exp\left(-\frac{(x-c)^2}{2\sigma^2}\right) \tag{2.1}$$

- Multi quadratic Radial Basis Function

$$\varphi(x) = (\sigma^2 + (x-c)^2)^{\beta}, 0 < \beta < 1 \tag{2.2}$$

- inverse multi quadratic Radial Basis Function

$$\varphi(x) = \frac{1}{\left(\sigma^2 + (x-c)^2\right)^{\alpha}}, \quad \alpha > 0 \tag{2.3}$$

- Thin plate splines Radial Basis Function

$$\varphi(x) = (x-c)^2 \log(x-c) \tag{2.4}$$

- Cubic splines Radial Basis Function

$$\varphi(x) = (x-c)^3 \tag{2.5}$$

- linear splines Radial Basis Function

$$\varphi(x) = x - c \tag{2.6}$$

A useful property of Radial Basis Functions which characterize the activation of RBF networks and it's capability to guarantee the non-singularity is stated in the following Lemma (Powell, 1992).

Definition 2.1 *A matrix which is* $A \in \Re^{nxn}$ *is positive definite if* $x^T Ax > 0$ *for all nonzero* $x \in \Re^n$

Lemma 2.1 *The nxn matrix whose elements have the values* $\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$, *i, j=1,2,..n is positive definite for all choice of different interpolation points in* $\Re^d$, *d being any positive integer.*

RBF networks are one of the feed forward neural networks architecture widely used besides multi-layer preceptor as its good capability in approximation especially in nonlinear system identification (Sundarajan, N et al., 1999; Fung, C.F et al., 1996; Huang, G.B et al, 2004; Roy, A et al., 1997; Ahmad, R et al.,2002; Asirvadam, V.J.,2002; and Finan, R.A et al.,1996).

RBF networks have simpler architecture compared to MLP networks. It consists of three layers. First layer is the input layer, which is made up of source nodes that connect the networks to its environment. The second layer is the hidden layer which applies a nonlinear transformation from the input layer to the last layer in RBF networks – termed as output layer. The output layer is linear in parameter, supplying the response of the network to the activation pattern feed through the input layer. Hidden layer for RBF networks are characterized of nonlinear parameters which form the centre and width of the basis functions. It can be one of the Radial Basis Functions as stated in 2.2.1 previously.

Its simpler topology makes RBF networks can reveal its learning process explicitly compared to MLP networks (Sundarajan, N et al., 1999; Fung, C.F et al., 1996; Huang, G.B, et al, 2004). The other benefit of its simpler architecture is in terms of computing speed compared to MLP (Brown Kenneth, M et al., 1972; Bishop, C.M, 1995). Besides those benefits mentioned, its topology which consist only one hidden layer has rigorously proved that it can uniformly approximate any smooth continuous function (Du, K.L et al., 2006).

Mathematically, RBF networks with Gaussian as the activation function and single output can be described as:

Given *N* arbitrary distinct samples (x_i, y_i) where $x_i = [x_{i1}, x_{i2}, ..., x_{in}]^T \in \Re^n$, $y_i \in \Re^1$.

RBF network with $\tilde{N}$ hidden nodes can be mathematically modeled as (Brown Kenneth, M et al., 1972).

$$F_{\tilde{N}}(x) = \sum_{j=1}^{\tilde{N}} \beta_j G(c_i, \sigma_i, x) \tag{2.7}$$

where

$$G(c_i, \sigma_i, x) = \exp\left(- \sum_{i=1}^{n} \frac{\left(x_i - c_{ji}\right)^2}{2\sigma^2{}_{ij}} \right) \tag{2.8}$$

2.3 Function Approximation Capability of RBF Networks with Gaussian

The Gaussian kernel function is normally chosen as the activation function for the RBF network due to its ease in interpretation (Surandrajan, N et al., 1999). It is found to be suitable not only in generalizing a global mapping but also filtering local features without much altering the already learned input features (Du, K.L. et.al, 2006; Surandrajan, N et al., 1999). The Gaussian function tends to be local in response compared to other types of Radial Basis Functions, such as multiquadratic function, thin-plate-spline function. Both position and shape of Gaussian function are more flexible to be adjusted (the nonlinear weight are easy to update) (Surandrajan, N et al., 1999). RBF networks with Gaussian as their activation function has many well defined mathematical features which can be used in learning some complex input output mapping and dynamics nonlinear system identification (Gupta, M et al., 2003; Young Sin, Mi, 1998).

The function approximation capability of RBF networks with Gaussian kernel can be proven to satisfy Stone-Weierstrass theorem (refers to Appendix B). The approximation capabilities of RBF networks are also highly related to its localization properties. Thus, RBF network with Gaussian kernel can be shown to have good approximation capability in overall with transparent interoperability. For those

reasons, Gaussian function is the best candidate of the activation function in the hidden neuron for RBF network.

Figure 2.1 shows the description of RBF Network as one of feed forward neural with single output.

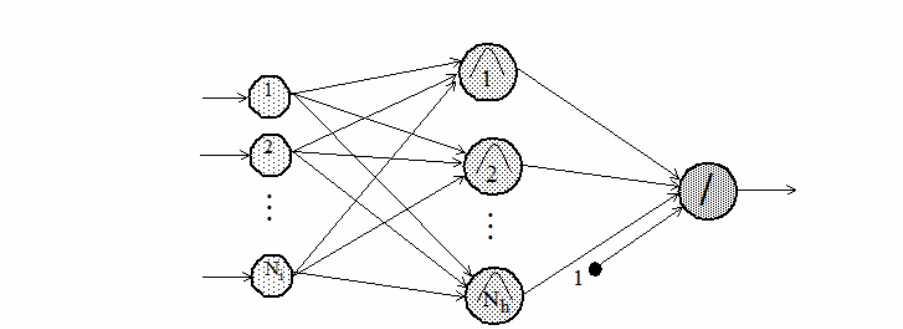


Figure 2.1. RBF Network with Gaussian Kernel and One Output Neuron

It consists of three layers. They are Input layer, hidden layer and output layer sequentially.

2.4 Learning in RBF networks

RBF network with Gaussian has capability to approximate arbitrary continuous functions uniformly defined on a compact set to satisfy a given approximation error (Yuan, X.B et al., 2005) as proven previously. This approximation process is usually carried out by learning phase where numbers of hidden nodes and the network parameters are appropriately adjusted so that the approximation error is minimized.

Learning of a neural network can be viewed as a nonlinear optimization problem in which the goal is to find a set of network parameters minimizing the cost function (error function) for given examples (Chen, S et al., 1990). In another way learning is an optimization process that produces an output that is as close as possible to the desired output by adjusting network parameters such as center, width and weight.

Radial Basis Function learning with Gaussian consists of two kinds of learning process. The first is learning of center and the second is learning of weight. Mathematically, learning RBF network can be described as:

Let a vector valued nonlinear function as the activation as stated at equation (2.8) and the weight vector is

$$\beta = [\beta_1, \beta_2, ..., \beta_{\tilde{N}}] \tag{2.9}$$

The overall input-output relationship of *n* input and 1 output can be described by this following nonlinear relationship

$$\hat{y}(t, c, \sigma, \beta, x) = \sum_{j=1}^{\tilde{N}} \beta_j G(c_i, \sigma_i, x) = f(G, \beta, t) \tag{2.10}$$

where $\hat{y}(x)$ is the network's output or prediction output, c is the center of Gaussian kernels and σ is the width, β is the linear in parameter weights connection hidden and output nodes and x is the input to the network.

Training or Learning module of RBF Network involves supplying the network with the input, determining and updating the parameter such as centre, width and weight so that the networks output $\hat{y}(t, c, \sigma, \beta, x)$ approximate the desired output $y(t, x)$ or the difference between the network's output and the desired output are kept minimum which leads to optimization problem (Sundarajan, N et al., 1999; Hawlett Robert, J et al., 2001 and Roy, A et al., 1997).

$$e = (\hat{y} - y) \tag{2.11}$$

2.4.1 Online and Offline Learning

There are two forms of learning algorithm in Neural Network in terms of terminology which categories by the way the data fed to the network during the training procedure. First is offline learning as when all data are assumed to be available on the initially or at priori then it is called as off-line learning. Besides off-line learning, there has been a considerable interest in on-line algorithm recently as the other alternative learning algorithm for ANN where the data are processed

sequentially as they measured (Pintelon, R et al., 2000; Asirvadam, V.S and Musab, J.O. Elamin., 2009; Bomberger, J.D et al., 1995; Junge, T.F and Unbehaeun, H., 1997; Marino, M and Scarpetta, S., 2000).

In another literature it is mentioned that the term 'on-line' refers to methods where the learning parameters are updated based only on information from one single pattern, while 'off-line' refers to learning method where the learning parameters are updated based on information from the whole training data (Isakson, Alf J., 1993). Some researchers also use the terms stochastic and batch as an alternative names for on-line and off-line. The focus of this thesis will be on on-line learning algorithm which is also termed recursive which trains the RBF networks adaptively.

2.4.2. Supervised and Unsupervised Learning

There are many learning modes that are conventionally used for Neural Networks training. They are supervised, unsupervised, reinforcement, and evolutionary learning. Supervised and unsupervised learning become the most commonly used in system identification, pattern recognition, control modeling, signal processing and approximation (Du, K.L et al, 2006).

Unsupervised learning does not need the target response. It is only based on the correlation among the input data and is used to find major features in input data without the need of output data (or the teacher).

Unsupervised learning is usually characterized by numerous clustering algorithms e.g. K-means clustering, which has been widely used due to its simplicity and ability to produce good results.

Supervised learning can be said as learning with a teacher which needs the actual output to compare with network response. The learning process involves nonlinear optimization process which minimizes the error between network approximation and real output of the plant or system. The learning parameters are adjusted by a combination of the learning pattern set and the corresponding error between the desired output and the network output.

2.4.3 Some Learning Principles for Neural Network

There are some learning principles of neural network that are commonly used as indicator for building learning methods. The learning principles, or prior settings before a neural network training took place, is produced based on the following criteria commonly (Roy, A et al., 1997)

1. Perform Network design task

    - Proper network architecture is set which will be optimal for a given set of problems.

2. Robustness in Learning

    - The learning method must be robust so as not to have the local minima problem. This can be done by simulating average of multiple initializations.

3. Quickness in Learning

    - The learning method must be quick in its learning and rapidly from only a few of examples as it means fast in learning the underlying functions.

4. Efficiency in Learning

    - The learning method must be computationally efficient in the learning when provided with a finite number of training examples.

5. Generalization in Learning

    - The learning method must be able to generalize reasonably well so that only small amount of network recourses is used. It must be tried to design smallest possible net, although it might not be able to do so every time.

2.5 System Identification

Generally, system Identification problem is to infer a mathematical relationship between past input-output data and future outputs (Sjöberg, J et al., 1994; Lennart Ljung., 1987).

Let

$$\varphi(t) = \left[y(t-1)...y(t-n_a) \quad u(t-1)...u(t-n_b) \right]^{\mathrm{T}} \tag{2.12}$$

As a finite number of input $u(k)$ and outputs $y(k)$ into vector $\varphi(t)$ for $d = n_a + n_b$, then $\varphi(t) \in \Re^d$.

Mathematically the system identification problem is the process to understand the relationship between the output of the system/plant, $y(t)$ and the input of the system, $\varphi(t)$. To obtain the understanding, a set of observation data is needed which is termed as the training set.

Let

$$Z^N = \left\{ \left[y(t), \varphi(t) \right] t = 1,...N \right\} \tag{2.13}$$

Thus, from these data we infer the relationship

$$\hat{y}(t) = \hat{g}_N\left(\varphi(t)\right) \tag{2.14}$$

$\hat{y}$ is the prediction for the real system by the model, in this case is the RBF Network, in the system identification process based on information that we get from $\varphi(t)$.

2.5.1 Nonlinear Black Box Modeling Using RBF

Describing the nonlinear and dynamic condition of the system will generally lead to become one of the challenging segments in nonlinear black box modeling. However, the basic idea is almost the same as black box modeling as the difference here is the identification process involves nonlinear network that is the RBF network.

From the previous fundamental researches (Sjöberg, J et al., 1994), there are several assumptions made for nonlinear black box modeling which are listed as follows:

$$\hat{y}(t) = g_0(\varphi(t)) \tag{2.15}$$

For given any function $g_0(\varphi): \Re^d \rightarrow \Re$.

The requirements are:

- $\{g_k(\varphi)\}$ is a basis for such function , i.e. which can be written as

$$g_0(\varphi) = \sum_{k=1}^{\infty} \beta(k) g_k(\varphi) \tag{2.16}$$

- For any reasonable function g_0 using suitable coefficients $\beta(k)$
- It produces 'good' approximations for finite sums.
- In optimization term minimizes the error function below:

$$e = \left\| g_0(\varphi) - \sum_{k=1}^{n} \beta(k) g_k(\varphi) \right\| \tag{2.17}$$

2.5.2. System Identification using RBF network

There are many ways to implement system identification; one of them is by using (ANN) technique. ANN techniques are widely used in identification especially for nonlinear system identification. The universal approximation property of ANN makes them suitable for identifying of black box nonlinear process plant with complex dynamics.

There are some reasons why ANN has attained so much interest in system identification circle. Its function expansion has good properties regarding the nonlinear function approximation and its localized features for RBF networks. Moreover, its capability can replace many multi dimensional regression and statistical techniques.

ANN can be seen as the predictor or modeler in system identification term. The output of the network is the prediction for the output of the underlying system. The prediction's output is computed based on past input and also previous prediction (output) of the system which is known as the autoregressive exogenous (ARX) input. The thesis will only focus on nonlinear ARX (NARX) using RBF network as the function to predictor or modeler.

There are several steps of system identification process using ANN. Similar to mathematical modeling; the identification process starts by building some assumptions underlying the systems. Secondly, one of the most important elements is to determine the model order which could be different for different problem of investigation. The third is determining the neural network architecture which determines the number of input and the number of hidden neurons. The last part is estimating the free or user defined parameter such as learning rates and the stopping criteria. The estimation process will be minimizing the error between the actual output and the prediction's output iteratively during the training.

System Identification by RBF network means, RBF network is chosen as the network architecture in the identification process thus, the network specifies the model parameterization. By estimating its learning parameters, RBF networks are the predictor in order to identify the system. As in (2.7), the free parameters of the RBF network are the weights (linear parameter) and the centers and also its widths of the basis functions (nonlinear learning parameters). There are some learning methods proposed by many researchers to determine priori the user defined (free parameters) (Li, B et al., 2007; Fung, C.F et al., 1996; Huang, G.B et al., 2004; Jun, Li et al., 2007; Chen J-Y, et al., 2006; Li, B et al., 2006; Zhu, Q.Y et al., 2005; Asirvadam, V.S et al., 2004; Bishop, C.M., 1998; Neruda, R and Kudova, P., 2004; Ngia Lester, S.H and Sjoberg, J., 2000; Lightbody, G and Irwing, G.W., 1992) before a training process takes place.

There is another concept known as system identification with lost packets for where the modeler will identify a nonlinear system online when the input data is presented with irregular sample instant. This is introduced and discussed thoroughly in Chapter 5.

2.6 Conclusion

This chapter has reviewed some basic concepts regarding RBF networks. It is started by some brief explanation about artificial neural networks and some overviews regarding RBF networks are also presented.

Since the objective of this thesis is to propose learning methods of RBF networks for nonlinear system identification, thus, the chapter also presents brief explanation about learning method on RBF networks and some basic theory of system identification. Some mathematics analytical theorems are also presented in the Appendix B to support the research methodology.

This chapter gives some basic terminology which will be used in the later chapters. It describes the research objectives in trying to find out the alternative effective learning methods of RBF networks with Gaussian kernel especially for system with lost packets.

The next chapter deals with the affect nonlinear parameter initialization to the overall RBF learning. It is started by some brief explanation on recently proposed Extreme Learning Machine (ELM) which is used as the benchmark comparison to the others techniques which uses smart clustering approach for initialization of Gaussian kernel parameters.

CHAPTER 3

INITIALIZATION OF RBF NETWORK

This chapter is talking about the importance of initialization on learning RBF networks. Since the benchmark algorithm, ELM, only use simple random initialization, some unsupervised learning method is used to improve the accuracy performance.

3.1 Parameter Initialization

Parameter's initialization is an important part in learning process of RBF networks. Theoretically an appropriate parameter's initialization method will give significant affect on the performance of the learning (Roy, A et al., 1997). Since the output of the RBF networks is the weighted sum of the local basis functions, the networks response or output is local in nature. This means that the output predict tends to zero as the data moves farther away from the center of the Gaussian kernel. Thus, determining center of Gaussian kernels in an important part of RBF initialization process based on density of the data.

There are several ways to initialize the RBF networks nonlinear parameters. It can be done using unsupervised learning technique such as K-means clustering and Fuzzy C-means clustering or just simply by random initialization. The main contribution of this chapter is to look into the influence of initialization methods for RBF networks to its overall performance on nonlinear system identification.

3.2 Extreme Learning Machine (ELM)

Extreme Learning Machine (ELM), which has been developed by Huang, (Huang, G.B et al., 2004; Liang, N.Y et al., 2006; Li, M.B and Er, M.J., 2006; Minh-Tuan, T et al., 2007; Singh, R and Balasundaran, S., 2007) has been proved to perform better compared to other learning methods such as stochastic back-propagation, resource allocation networks (RAN) (Platt, J., 1990) and minimal RAN (MRAN) (Yingwei et al., 1997,2000) in some classification, regression, and system identification purposes. ELM uses simply random initialization to set the nonlinear learning's parameters (centre and the width) and pseudo-inverse technique and linear optimization to adjust and update the linear learning's parameter. It does not perform any update procedure on the nonlinear learning parameters. ELM has been developed as batch learning method initially then it was extended for online learning method with RLS algorithm to update the linear weights (Liang, N.Y et al., 2006). The idea of ELM is not new as similar method can be found in literature which is termed as hybrid or separable learning techniques (Mcloone(a) et al., 1998; McLoone(b) et al., 1998).

The ELM has two learning phases, prior initialization and online output neuron connected weights update phase which is described next.

Phase I. Initialization

Initialize the parameters using small batch of initial learning data $N_0 = \{(x_i, y_i)\}_{i=1}^{N_0}$ from the given learning set

$$N = \{(x_i, y_i) / x_i \in \Re^n, y_i \in R^1, i = 1,2,...\} \tag{3.1}$$

$N_0 \geq \tilde{N}$

a). assign random the center and impact width of the hidden neuron $(c_i, \sigma_i, i = 1,..., \tilde{N})$

b). Calculate the initial hidden layer output H_0

$$H_0 = \begin{bmatrix} G(c_1, \sigma_1, x_1) & .... & G(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_1) \\ . & & \\ . & & \\ . & & \\ G(c_1, \sigma_1, x_{N_0}) & & G(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_{N_0}) \end{bmatrix}_{N_0 x \tilde{N}} \tag{3.2}$$

c). Estimate the initial output weight

$$\beta^{(0)} = P_0 \; H_0{}^T Y_0 \tag{3.3}$$

where

$$P_0 = (H_0{}^T H_0)^{-1} \tag{3.4}$$

d).set t=0

Phase II. Update learning phase

Given the $(t+1)$ -th chunk of new observation

$$N_{t+1} = \{(x_i, y_i)\} \quad \begin{matrix} \sum_{j=0}^{t+1} N_j \\ i = \left(\sum_{j=0}^{t} N_j \right) + 1 \end{matrix} \tag{3.5}$$

Where N_{t+1} denote the number of observation in the $(t+1)$ th sample instant.

a). Calculate the partial hidden layer output matrix H_{t+1} for the $(t+1)$ th chunk of data N_{t+1}

b). set $$Y_{k+1} = \left[y_{\left(\sum_{j=0}^{t} N_j\right)+1}, ..., y_{\left(\sum_{j=0}^{t+1} N_j\right)} \right]^T \tag{3.6}$$

c). Calculate the output weight $\beta^{(k+1)}$

$$P_{t+1} = P_t \; - P_t \; H_{t+1}^T (I + H_{t+1} P_t \; H_{t+1}^T)^{-1} H_{t+1} P_t \tag{3.7}$$

$$\beta^{(t+1)} = \beta^{(t)} + P_{t+1}\, H_{t+1}{}^{T}\left[Y_{t+1} - H_{t+1}\beta^{(t)}\right] \tag{3.8}$$

d). set $t = t+1$. Go to step 2).

If the data is not arrive chunk by chunk but one by one (N_{t+1}=1), then by using Sherman-Morrison Formula we get

$$P_{t+1} = P_t - \frac{P_t\, h_{t+1} h_{t+1}^T P_t}{1 + h_{t+1}^T P_t h_{t+1}} \tag{3.9}$$

$$\beta^{(t+1)} = \beta^{(t)} + P_{t+1}\, h_{t+1}\left[y_{t+1}^T - h_{t+1}^T \beta^{(t)}\right] \tag{3.10}$$

where

$$h_{t+1} = \left[G\left(c_1, \sigma_1, x_{(t+1)}\right)..G\left(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_{(t+1)}\right)\right] \tag{3.11}$$

ELM approach discussed in the thesis looks into only the online variant as all the proposed techniques throughout the thesis which deal with recursive RBF learning techniques use the ELM as the comparison due to its simplicity and ease of interpretation. This chapter looks into the influence of nonlinear parameter initialization using ELM as a framework, as ELM use only random initialization to RBF overall performance.

3.3 Smart Clustering Method for Learning's Initialization

As stated previously, center placement plays crucial part in RBF networks learning process. Efficient center placement prior to learning process gives very significant effect on the accuracy of the RBF networks model. Thus, modifying the initialization methods especially for the nonlinear parameters become core issues.

3.3.1 K-means clustering

K-means Clustering is one of the most popular unsupervised learning methods used for grouping data points. The K-means has been widely accepted because of its simplicity and ability to produce good results (Kim, K.J., and Ahn, H., 2008). The basic idea of this algorithm is clustering or grouping the learning data (input data) into some cluster based on the measurement between the input data and the attracting centers which is usually Euclidian. For RBF case the each cluster will be associated as one hidden node.

K-means will group learning data into one cluster with certain degree of similarity. K-means clustering is categorized as hard clustering which assign each data into exactly one cluster with degree of membership is set to one. Thus, the boundaries among clusters are well defined. The center of each Gaussian function will be updated until there are not any movements or assigning data on the next iteration.

Mathematically, K-means clustering problem has been formulated as a combinational optimization problem to divide a data set (Bishop, C.M., 1998).

Let a data set

$$X = \left\{ x_i \in \Omega : \Omega \subset \Re^d, 1 \le i \le n \right\} \tag{3.12}$$

into *k* different cluster.

$$\mathrm{Y} = \left\{ y_v \in \Omega : 1 \le v \le k \right\} \tag{3.13}$$

and a mapping $m : \Re^d \to \{1,...,k\}$ which as signs a cluster label $\alpha \in \{1,...,k\}$ to each data vector x_i. K-means clustering applies the nearest neighbor rule such that $x_i \to y_\alpha$ if

$$\left\| x_i - y_\alpha \right\| \le \left\| x_i - y_v \right\|, \forall v \in \{1,...,k\} \tag{3.14}$$

The next section looks into K-means clustering technique implemented on ELM RBF structure. A slight modification being made on Gaussian width setting on top of

the K-means clustering technique in order to see improvement to basic random initialization on RBF learning.

Initialization Phase Using K-means Clustering

(a) Given the input data.

(b) Determine k-number of cluster.

(c) Choose the initial center for Gaussian kernels. It can be random from the input data or out of the input data.

(d) Calculate the distance between the data (object) and the center.

(e) Group them based on minimal distance set with user defined threshold.

(f) Update the centre based on the displacement of the average cluster value and the new data.

(g) Determine new center for hidden neuron and go to (c).

The width of the clustering can also be set after the K-means clustering for the centers. The width of the Gaussian kernel controls the amount of overlap of the RBF which influence the networks generalizations. Although they can be determined in a variety of ways, the most common one is to make them equal to the average of sum of the Euclidian distance between the center and its neighboring.

$$\sigma_i^2 = \frac{1}{M_i} \sum_{x \in z_i} \|x - c_i\|^2 \tag{3.15}$$

z_i = the i-th cluster of the learning data and M_i = the number of data in the z_i.

In this research, another form mathematical formulation is proposed as an alternative to determine the width as one over the root square of sum of the Euclidian distance between each center and all input data, which leads to normalization of Gaussian function.

$$\sigma_i = \frac{1}{\sqrt{\sum_{n=1}^{M}(m_n - c_i)^2}} \tag{3.16}$$

It can be shown in that K-means clustering prior initialization of RBF centers can perform better compared to random initialization especially in terms of accuracy (computed as an average mean squared error of learning curves). Almost by the all benchmarks shows that K-means technique can improve the performance of the RBF networks learning compared to basic ELM technique as presented in Table 3.1.

Table 3.1 Performance comparison of ELM and ELM K-means

	ELM	ELM {K-means}
SELS, Nh=5		
Training	3.40E-02	1.86E-03
Testing	3.68E-02	1.93E-03
Mackey Glass, Nh=10		
Training	4.62E-03	7.46E-05
Testing	5.89E-03	9.76E-05
Narendra Time Serise, Nh=30		
Training	2.25E-03	1.36E-04
Testing	1.20E-02	1.99E-03
Henon Map, Nh=10		
Training	5.94E-02	4.99E-03
Testing	6.94E-02	6.50E-03
Chen Series, Nh=10		
Training	5.21E-02	1.80E-02
Testing	6.03E-02	2.23E-02

One example of the performance improvement can be seen on Mackey Glass Time Series in Table 3.1. K-means clustering outperforms random initialization. Taking the same number of hidden nodes, it can decrease the MSE of the training and testing. It means that it has better accuracy.

3.3.2 Fuzzy C-means Clustering

The second initialization method uses the Fuzzy C-means clustering technique. Fuzzy C-means clustering is actually an improvement of the K-means clustering (Kim, K.J et al., 2008). The difference between them will be more on the membership degree as set by the fuzzy logic rules. The membership degree for K-means clustering is crisp based which is 0 or 1, but the membership degree for Fuzzy C-means clustering can be value within interval from 0 until 1. Thus, Fuzzy C-means clustering is also well known as soft K-means clustering approach as opposed to the original hard K-means clustering.

Initialization Phase Fuzzy C-Mean Clustering

(a) Initialize Unitary matrix $U = [u_{ij}] = U^0$

(b) At t-step calculate the centers vectors

(c) $C^{(t)} = [c_j]$ with $U^{(t)}$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}{}^m x_i}{\sum_{i=1}^{N} a_{ij}{}^m} \tag{3.17}$$

(d) Update $U^{(t)}, U^{(t+1)}$

$$U_{ij} = \frac{1}{\sum_{t=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_t\|} \right)^{\frac{2}{m-1}}} \tag{3.18}$$

(e) if $\left\| U^{(t+1)} - U^{(t)} \right\| < e$ then stop, otherwise return to step 2.

x_i = n dimensional measured data and c_j = center of the cluster while m =any real number that greater than 1 and u_{ij} = degree of membership of x_i in the cluster j.

The impact width will be determined as one over the root square of sum of the Euclidian distance between each center and all input data.

$$\sigma_i = \frac{1}{\sqrt{\sum_{n=1}^{M}(x_n - c_i)^2}} \tag{3.19}$$

Table 3.2 presents Fuzzy C-means clustering based initialization. The Fuzzy C-means clustering can perform better compared to random initialization especially in terms of average mean square error for the whole benchmark problems.

Table 3.2 Performance comparison of ELM and ELM Fuzzy C- means

	ELM	ELM {Fuzzy C-means}
SELS, Nh=5		
Training	3.40E-02	1.78E-03
Testing	3.68E-02	1.88E-03
Mackey Glass, Nh=10		
Training	4.62E-03	9.00E-05
Testing	5.89E-03	1.57E-04
Narendra Time Serise, Nh=30		
Training	2.25E-03	1.17E-04
Testing	1.20E-02	1.59E-03
Henon Map, Nh=10		
Training	5.94E-02	4.80E-03
Testing	6.94E-02	6.11E-03
Chen Series, Nh=10		
Training	5.21E-02	3.40E-03
Testing	6.03E-02	2.12E-02

One example of the performance improvement can be seen on Mackey Glass Time Series in Table 3.2. Fuzzy C-means clustering outperforms random initialization. Taking the same number of hidden nodes, it can decrease the MSE of the training and testing, means that it has better accuracy.

3.4 Comparison of Convergence rate

Table 3.3 – 3.6 show the convergence rate for the benchmark problem which depict the unsupervised learning of K-mean and Fuzzy C-means outperform the

traditional random initialization method. The simulations for each benchmark are measured based on various numbers of training data to see effectiveness of its final value. One main observation is the prior clustering of Gaussian center of RBF Networks did improve initial part of learning capability (using less number of training patterns) for all benchmarks. In overall the K-means and the Fuzzy C-means, show the best performance in RBF networks learning.

Table 3.3 Convergence rate of SELS

Number of train data	Method		
	ELM	ELM {K-means}	ELM {Fuzzy C-means}
300	4.17E+00	9.97E+01	9.93E+00
500	1.44E-01	1.33E+00	1.07E-01
1000	3.93E-02	1.62E-01	2.79E-02
2000	3.48E-02	9.17E-03	2.05E-03
3000	3.40E-02	1.86E-03	1.78E-03

Table 3.3 performs the capability of each learning method to reach convergence for SELS. It can be seen for SELS, ELM with smart clustering has almost the same performance in terms of convergence rate as ELM. On average, It converges after 500 training data.

Table 3.4 Convergence rate of Mackey Glass

Number of train data	Method		
	ELM	ELM {K-means}	ELM {Fuzzy C-means}
300	1.74E+04	1.86E+06	8.67E+03
500	1.37E-01	5.74E+01	9.33E-02
1000	2.27E-02	1.18E-01	1.33E+00
2000	5.84E-03	9.48E-05	9.99E-05
3000	4.62E-03	7.46E-05	9.00E-05

Table 3.4 performs the capability of each learning method to reach convergence for Mackey Glass. It can be seen for Mackey Glass, ELM with smart clustering has almost the same performance in terms of convergence rate as ELM. It converges after 500 training data.

Table 3.5 Convergence rate of Narendra

Number of train data	Method		
	ELM	ELM {K-means}	ELM {Fuzzy C-means}
300	2.10E+06	3.44E+01	2.24E+06
500	1.36E+01	7.95E+03	5.07E+00
1000	4.16E+00	1.13E+01	1.54E-02
2000	2.25E-03	7.37E-03	2.14E-04
3000	2.25E-03	1.36E-04	1.17E-04

Table 3.5 performs the capability of each learning method to reach convergence for Narendra Time Series. It can be seen for Narendra Time Series, ELM with smart clustering has almost the same performance in terms of convergence as ELM. On average, it converges after 1000 training data.

Table 3.6 Convergence rate of Henon Map

Number of train data	Method		
	ELM	ELM {K-means}	ELM {Fuzzy C-means}
300	3.10E+04	5.99E+05	8.06E+03
500	2.54E+00	5.07E-01	7.20E-02
1000	6.25E-02	1.68E+04	6.22E-03
2000	6.22E-02	5.47E-03	3.89E-02
3000	5.94E-02	4.99E-03	4.80E-03

Table 3.6 performs the capability of each learning method to reach convergence for Henon Map. It can be seen for Henon Map, ELM with smart clustering has almost the same performance in terms of convergence rate compared to ELM. On average, they converge after 1000 training data. However ELM with Fuzzy C-means clustering initialization has faster convergence rate for this benchmark. It converges after 300 training data.

Table 3.7 Convergence rate of Chen Series

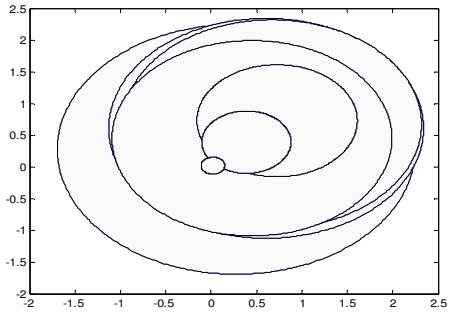
Number of train data	Method		
	ELM	ELM {K-means}	ELM {Fuzzy C-means}
300	1.09E+02	1.10E+03	2.78E-01
500	5.72E-02	1.85E-02	9.43E-03
1000	4.75E-02	1.81E-02	3.76E-03
2000	5.23E-02	1.95E-02	2.47E-03
3000	5.21E-02	1.80E-02	3.40E-03

Table 3.7 above performs the capability of each learning method to reach convergence for Chen Series problem. It can be seen for Chen Series problem, ELM with smart clustering has almost the

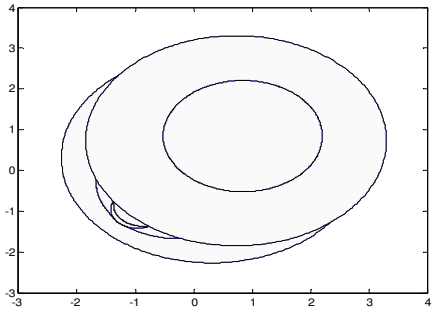
same performance in terms of convergence rate as ELM. On average, it converges after 300 training data.

From Tables 3.3- Table 3.7 it can be concluded that all the initialization method does not give any significance into the learning performance in terms of convergence rate. They are comparable

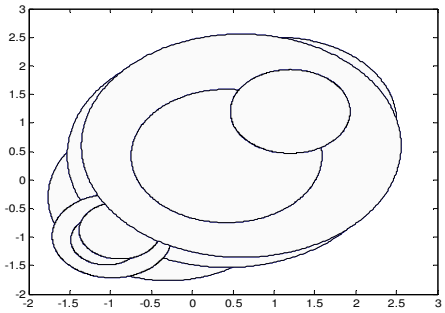
Figure 3.1(a)-(c) shows the center placement comparison among random initialization and the unsupervised learning method for Chen Series benchmark problem.



(a) Center placement using random methods



(b) Center placement using K-means



(c) Center placement using fuzzy c-means

Figure 3.1 Initialization comparisons of Chen Series

The degree of overlap is due to vast random nature of initialization before clustering technique take place.

3.5 Conclusion

This chapter has discussed the smart clustering for RBF initialization compared to random initialization technique. The main contribution stated is on examining how important the initialization of RBF networks parameter to the learning accuracy. The ELM technique being introduced which is used as benchmark for other proposed techniques. The smart clustering technique such as K-means clustering, and Fuzzy C-means are applied on ELM algorithm for observing the difference in performance compared to basic random initialization. Simulation results show the prior clustering approach do affect in the final value of error convergence. Thus, prior initialization of nonlinear learning parameter as one of the presetting has been done before the RBF networks is subjected to training recursively or online.

The next chapter deals with finite difference or derivative free gradient estimate on RBF recursive nonlinear parameter update. The influence of recursive update method for nonlinear learning parameter with various initialization strategies discussed in this chapter will also be evaluated.

CHAPTER 4

RECURSIVE LEARNING WITH FINITE DIFFERENCE

Recursive learning algorithm can be viewed as optimization problem of RBF Networks parameters (Barwis, J et al., 1992; Sudkam, Thomas A., 2006). It needs the availability of gradients estimate online for adaptive update of weight. Finite difference is applied on Recursive Prediction Error (RPE) algorithm for online RBF learning. It is used on nonlinear parameter update.

4.1 Finite Difference

Finite Difference is one of indirect gradient estimation methods which will estimate the approximation of the real gradient of any nonlinear networks underlying dynamics system (Michael C. Fu., 2005; Simandl, M and Dunika, J., 2009). Figure 4.1 shows the derivative estimate using finite difference.

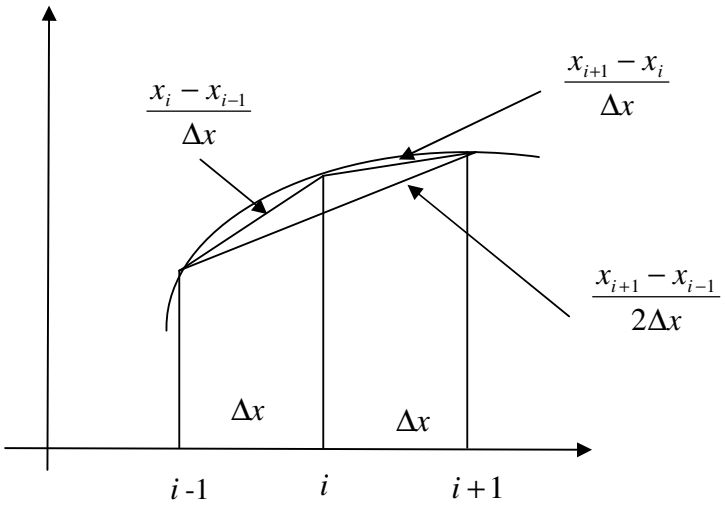


Figure 4.1 Derivative free Gradient using Finite Difference.

Referring to Figure 4.1, if the function curve is described by the function $f(x)$, then the slope or the gradient is simply defined as

$$f'(c) = \frac{f(x_{i+1}) - f(x_{i-1})}{2\Delta x} \tag{4.1}$$

That is, it is the distance between the values of the function evaluated at the two points divided by the distance between the two points as depicted in Figure 4.1.

This basic definition of derivative lays over the two points which makes equation (4.1) undefined since it may be divided by zero. This problem can be solved by taking the limit as the convergence ratio can be seen during the limit process.

$$\frac{\partial f(x)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x_i + \Delta x) - f(x_i)}{\Delta x} \tag{4.2}$$

This form of definition is difficult in the programming point of view. Thus, a finite difference form of representation needed to as an alternative.

Finite difference approximates gradient of certain known function by using the function value at only a set of discrete point. It can be generated for any other derivative's order with any number of points (Chung, T.J., 2002).

The derivation of gradient's approximation through finite difference will be presented as follows:

Lets function $f(x)$ and its derivative at point x as (4.2) thus if $f(x + \Delta x)$ can be expanded using Taylor Series about $f(x)$,

$$f(x + \Delta x) = f(x) + \Delta x \frac{\partial f(x)}{\partial x} + \frac{(\Delta x)^2}{2} \frac{\partial^2 f(x)}{\partial x^2} + \frac{(\Delta x)^3}{3!} \frac{\partial f(x)}{\partial x^3} + \ldots \tag{4.3}$$

Substituting (4.3) into (4.2) yields

$$\frac{\partial f(x)}{\partial x} = \lim_{\Delta x \to 0} \left(\frac{\partial f(x)}{\partial x} + \frac{(\Delta x)}{2} \frac{\partial^2 f(x)}{\partial x^2} + \frac{(\Delta x)^2}{3!} \frac{\partial f(x)}{\partial x^3} + \ldots \right) \tag{4.4}$$

Or it can be seen from (4.3) that

$$\frac{f(x+\Delta x)-f(x)}{\Delta x}=\frac{\partial f(x)}{\partial x}+\frac{(\Delta x)}{2}\frac{\partial^2 f(x)}{\partial x^2}+\frac{(\Delta x)^2}{3!}\frac{\partial f(x)}{\partial x^3}+\ldots=\frac{\partial f(x)}{\partial x}+O(\Delta x) \tag{4.5}$$

The derivative $\frac{\partial f(x)}{\partial x}$ in (4.5) is of first order in $\Delta(x)$, indicating that the truncation error $O(\Delta x)$ goes to zero like the first power in $\Delta(x)$, with first order accuracy. Referring to Figure 4.1, some finite difference will be derived as shown below. Based on the Figure 4.1, f may be written in Taylor Series at $x+\Delta x$ and $x-\Delta x$

$$f(x+\Delta x)=f(x)+\Delta x\frac{\partial f(x)}{\partial x}+\frac{(\Delta x)^2}{2}\frac{\partial^2 f(x)}{\partial x^2}+\frac{(\Delta x)^3}{3!}\frac{\partial f(x)}{\partial x^3}+\ldots \tag{4.6}$$

$$f(x-\Delta x)=f(x)-\Delta x\frac{\partial f(x)}{\partial x}+\frac{(\Delta x)^2}{2}\frac{\partial^2 f(x)}{\partial x^2}-\frac{(\Delta x)^3}{3!}\frac{\partial f(x)}{\partial x^3}+\ldots \tag{4.7}$$

Rearrange (4.6), it can be derived the forward finite difference

$$\frac{\partial f(x)}{\partial x}=\frac{f(x+\Delta x)-f(x)}{\Delta x}+O(\Delta x) \tag{4.8}$$

Rearrange (4.7), the backward finite difference is derived into

$$\frac{\partial f(x)}{\partial x}=\frac{f(x)-f(x+\Delta x)}{\Delta x}+O(\Delta x) \tag{4.9}$$

Thus, a central finite difference is obtained by subtracting (4.8) from (4.9)

$$\frac{\partial f(x)}{\partial x}=\frac{f(x+\Delta x)-f(x-\Delta x)}{2\Delta x}+O(\Delta x^2) \tag{4.10}$$

Assumption made that the function f is known at the points $x+h,$ and $x-h$. Hence, in this work, the computation of any derivative free gradient is based on its approximation of $f'(x)$. The approximation of $f'(x)$

$$\frac{\partial f(x)}{\partial x}=f'(x)\cong D_0(h)=\frac{f(x+h)-f(x-h)}{2h} \tag{4.11}$$

The equation is ***central difference approximation*** and this approximation is known to be more accurate compared with forward or backward finite difference approximation. Thus, the central difference approximation is preferred to be used for gradient estimate approximation for this research work.

4.2 Recursive Prediction Error (RPE) algorithm

Recursive Predictive Error algorithm uses Gauss Newton algorithm in determining the search direction in RBF parameter space (Chen, S., 1990; Chen, S and Billings, S.A., 1989). This technique improves the efficiency of the minimization process online by increasing the convergence rate using second order approximation. The derivation of search direction of RPE which leads to nonlinear weight update for RBF networks derived below:

The nonlinear parameters which are the centers, c_i, and the width, σ_t are grouped as

$$\Theta_t = [c_t; \sigma_t] \tag{4.12}$$

Hessian matrix is subject to

$$H(\Theta) = \frac{1}{2N_0} \sum_{i=1}^{N_0} \nabla\hat{y}(\Theta) \nabla\hat{y}^{\mathrm{T}}(\Theta) \tag{4.13}$$

With the gradient is derived as

$$\nabla\hat{y}(\Theta) = \left[\frac{d\hat{y}(t, c, \sigma, \beta, x)}{d\Theta} \right] \tag{4.14}$$

In case $H(\Theta)$ is near singular, (4.13) can be modified into

$$H(\Theta) = \frac{1}{2N_0} \sum_{i=1}^{N_0} \nabla\hat{y}(\Theta) \nabla\hat{y}^{\mathrm{T}}(\Theta) + \rho I, \quad \rho > 0 \tag{4.15}$$

for the small value of ρ and I is the identity matrix with appropriate dimension.

The search direction for RPE algorithm is

$$\nabla P = \left[\frac{1}{N_0} \sum_{i=1}^{N_0} \nabla \hat{y}(\Theta) \nabla \hat{y}^{\mathrm{T}}(\Theta) + \rho I \right]^{-1} \frac{1}{N_0} \sum_{i=1}^{N_0} \nabla \hat{y}(\Theta) \cdot e, \quad \rho > 0 \tag{4.16}$$

4.3 Finite Difference based Recursive update on RBF Networks

Finite difference based gradient estimate is used to find the search direction for updating online the RBF nonlinear parameters. This approach is adopted as a part of proposed nonlinear learning parameter's updating procedure in this thesis. The linear learning parameter update of RBF uses the linear optimization technique such RLS which has been discussed in Chapter 3. Thus, the RBF learning involved uses the hybrid training with derivate free estimate which lead to main contribution of the thesis.

Let ∇c and $\nabla \sigma$ is the partial derivation of the nonlinear parameters as in (2.10) subject to c and σ. Central finite difference will be used to determine it.

$$\begin{aligned} \nabla c &= \left[\frac{\partial \hat{y}(t, c, \sigma, \beta, x)}{\partial c} \right] \\ &= \frac{\hat{y}(c+h) - \hat{y}(c-h)}{2h} \\ &= g_c(G(c), t) \end{aligned} \tag{4.17}$$

$$\begin{aligned} \nabla \sigma &= \left[\frac{\partial \hat{y}(t, c, \sigma, \beta, x)}{\partial \sigma} \right] \\ &= \frac{\hat{y}(\sigma+h) - \hat{y}(\sigma-h)}{2h} \\ &= g_c(G(\sigma), t) \end{aligned} \tag{4.18}$$

In this research work, the step size h is set to dynamic value not a small values constant as in many literature. This make the derivates free estimates for nonlinear terms, the center and the width more robust dynamic changes to underlying function. The term h is set to dynamic value as

$$h = 0.0001 \cdot \|\sigma\| \tag{4.19}$$

which takes 0.01% of the width of the Gaussian kernel.

Denote $\hat{c}(t)$, $\hat{\sigma}(t)$ as the estimate of c and σ at t and let a time-varying networks model as followed

$$\begin{bmatrix} \hat{y}_t \\ grad\_c_t \end{bmatrix} = \begin{bmatrix} f(G(c_{t-1})) \\ g_c(G(c_{t-1})) \end{bmatrix} \tag{4.20}$$

$$\begin{bmatrix} \hat{y}_t \\ grad\_\sigma_t \end{bmatrix} = \begin{bmatrix} f(G(\sigma_{t-1})) \\ g_c(G(\sigma_{t-1})) \end{bmatrix} \tag{4.21}$$

The recursive algorithm that adopted from the conventional Recursive Prediction Error algorithm, can be summarized as follow

$$\nabla p_t = \gamma_m \nabla p_{t-1} + \gamma_g \nabla p_t e \tag{4.22}$$

$$P_t = \frac{1}{\lambda}\{P_{t-1} - [P_{t-1} x D' K D P_{t-1}]\} \tag{4.23}$$

$$\Theta_t = \Theta_{t-1} + P_t \nabla P \tag{4.24}$$

Where

$$D = [\nabla c; \nabla \sigma] \tag{4.25}$$

$$K = [\lambda I + D' P_{t-1} D]^{-1} \tag{4.26}$$

$$\Theta_t = [c_t; \sigma_t] \tag{4.27}$$

∇P is recursive form of the gradient search also known as smoothed stochastic gradient (Chen, S., 1990). The forgetting factor, λ, momentum and adaptive gain γ_m and γ_g, are the user defined parameters set prior to learning. P_t determines the asymptotically accuracy of the estimation and therefore is referred to as covariance matrix.

Basic form of covariance stated in (4.23) may lead to c*ovariance wind up* which would further lead to P_t become explosive. Constant trace adjustment is one of the recommended techniques to overcome it.

$$P_t = P_{t-1} - P_{t-1} DKD' P_{t-1} \tag{4.28}$$

$$P_t = \frac{K_0}{trace[P_t]} P_t, K_0 > 0 \tag{4.29}$$

This new definition of P_t sets an upper bound for the eigen values of P_t matrix.

Finite Difference RPE learning's algorithm is divided into two phases, initialization and update phase. Besides random initialization, some unsupervised learning methods discussed in the previous chapters will be evaluated to observe the influence of weight initialization to the final value convergence.

4.3.1 Finite Difference RPE with Random Initialization (FD-RPE Rand)

The finite difference RPE for RBF Networks learning with random weight initialization procedure can be summarized as follows:

Phase I. Initialization

Initialize the learning using a pair of learning data $N_0 = \{(x_i, y_i)\}$ from the given learning set

(a). Assign randomly the center and impact width of the hidden neuron $\left(c_i, \sigma_i, i = 1,..., \tilde{N}\right)$

(b). Calculate the initial hidden layer output H_0

$$H_0 = \begin{bmatrix} G(c_1, \sigma_1, x_1) & .... & G(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_1) \\ . & & \\ . & & \\ . & & \\ G(c_1, \sigma_1, x_{N_0}) & & G(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_{N_0}) \end{bmatrix}_{N_0 x \tilde{N}} \tag{4.30}$$

(c). Estimate the initial output weight

$$\beta^{(0)} = (H_0{}^T H_0) H_0{}^T Y_0 \tag{4.31}$$

where

$$P_{NL} = 1000 \times rand(2xNi \cdot Nh) \tag{4.32}$$

$$P_L = 1000 \times rand(Nh + 1) \tag{4.33}$$

(d) set t=0

Phase II. Learning/Training phase

Given the $(t+1)$ -th chunk of new observation

$$N_{t+1} = \{(x_i, y_i)\} \tag{4.34}$$

Where N_{t+1} denote the observation in the $(t+1)$ th iteration

Nonlinear Parameter update

a) Calculate error as

$$e = y(t) - \beta(t) x H(t) \tag{4.35}$$

b) Calculate gradient of cost function (2.11) by Finite difference

$$\nabla J = \frac{J(\Theta_t + h) - J(\Theta - h)}{2h} \tag{4.36}$$

c) Calculate gradient of $\hat{y}$ subject to Θ as

$$\nabla \hat{y} = -\nabla J / err \tag{4.37}$$

d) Calculate Search Direction of RPE algorithm

$$\nabla P = \left[\frac{1}{N_0} \sum_{i=1}^{N_0} \nabla \hat{y}(\Theta) \nabla \hat{y}^{\mathrm{T}}(\Theta) + \rho I \right]^{-1} \frac{1}{N_0} \sum_{i=1}^{N_0} \nabla \hat{y}(\Theta) \times e, \quad \rho > 0 \tag{4.38}$$

e) Update the nonlinear learning parameter as

$$P_{NL} = P_{NL} - \frac{\left[P_{NL} \times \nabla \hat{y} \times (fgfc + \nabla \hat{y}' \times P_{NL} \times \nabla \hat{y})^{-1} \times \nabla \hat{y}' \times P_{NL} \right]}{fgfc} \tag{4.39}$$

$$P_{NL} = {}^{1}\!\!\diagup\!\!{}_{trace(P_{Nl})} \times P_{NL} \tag{4.40}$$

$$\nabla P = \gamma \times \nabla P \times \lambda \times \nabla \hat{y} \times err \tag{4.41}$$

$$\Theta_t = \Theta_{t-1} + P_t \nabla P \tag{4.42}$$

Linear Parameter update

a). Calculate the partial hidden layer output matrix h_{t+1}

$$h_t = \left[G(c_1, \sigma_1, x_{(t)}) ... G(c_{\tilde{N}}, \sigma_{\tilde{N}}, x_{(t)}) \right] \tag{4.43}$$

b). Calculate the output weight $\beta^{(k+1)}$

$$P_{L(t)} = P_{L(t-1)} - \frac{P_{L(t-1)} h_t h_t^T P_{L(t-1)}}{1 + h_t^T P_{L(t-1)} h_t} \tag{4.44}$$

$$\beta^{(t)} = \beta^{(t-1)} + P_{L(t)} \times \ h_t x err \tag{4.45}$$

Table 4.1 presents simulation results obtained for derivate free (finite difference based) weight update. As a comparison, the real gradient estimate result is presented too. For a wired system or system identification with constant sample time, the derivative free and real gradient estimate is expected to perform similarly and Table 4.1 shows exactly the same as the expectation. In addition, based on simulation on the benchmark problems, both nonlinear weight update methods outperform the ELM technique. It can be observed that the finite difference based gradient estimate with dynamic step size (h) output outperforms the real gradient estimate for case studies where underlying function is highly nonlinear in nature by comparing its MSE (e.g. Mackey-Glass and Narendra).

Table 4.1 Performance comparison (by MSE) of ELM and RPE for Random Initialization

	ELM	FD-RPE {Rand}	Grad-RPE {Rand}
SELS, Nh=5			
Training	3.40E-02	3.39E-02	4.30E-02
Testing	3.68E-02	3.64E-02	2.43E-02
Mackey Glass, Nh=10			
Training	4.62E-03	3.70E-03	7.57E-02
Testing	5.89E-03	5.04E-03	1.15E-02
Narendra Time Series, Nh=30			
Training	2.25E-03	2.14E-03	7.24E-03
Testing	1.20E-02	8.87E-03	3.10E-02
Henon Map, Nh=10			
Training	5.94E-02	7.66E-03	8.11E-02
Testing	6.94E-02	1.03E-02	8.01E-02
Chen Series, Nh=10			
Training	5.21E-02	8.10E-03	7.60E-02
Testing	6.03E-02	1.32E-02	7.64E-02

One example of the performance improvement can be seen on Narendra Time Series Problem in Table 4.1. Applying Finite Difference on nonlinear learning parameter update increases the accuracy. Its MSE is lower compared to ELM. Moreover, Finite Difference also outperforms the real gradient. It means that Finite Difference can estimate the gradient well. However, in some other benchmarks (e.g. SELS, Henon Map, Chen Series), both of them are performing almost the same.

4.3.2 Finite Difference RPE with Smart Clustering

The finite difference RPE for RBF networks is adopted with prior smart clustering initialization method such as K-means and Fuzzy C-means clustering technique. Two finite difference RPE methods introduced in this section which will be describe next.

Phase I. Initialization

All initialization steps are the same as **FD-RPE Rand** except the initialization methods uses the smart clustering technique such as K-means (**FD-RPE K-means**) and Fuzzy C-means (**FD-RPE Fuzzy C-Means**) clustering algorithm as stated in previous chapter.

Phase II. Learning/ Training phase

Update learning phase for **FD-RPE K-means** and FD**-RPE Fuzzy C-means** is similar to FD-RPE Rand as discussed in the previous section will be conducted as update learning phase on FD-RPE Random, refer to equations (4.34)-(4.45).

Some parts of simulation results are presented in Table 4.2. It presents the performance comparison among ELM, FD RPE, and Grad RPE with K-means clustering initialization in terms of accuracy (MSE comparison). In addition, Table 4.3 presents the performance comparison among ELM, FD RPE, and Grad RPE with Fuzzy C-means clustering initialization in terms of accuracy (MSE comparison).

Table 4.2 Performance comparison (by MSE) of ELM and RPE using K-means clustering

	ELM	ELM {K-means}	FD-RPE {K-means}	Grad-RPE {K-means}
SELS, Nh=5				
Training	3.40E-02	1.86E-03	1.66E-03	2.17E-02
Testing	3.68E-02	1.93E-03	1.72E-03	3.03E-02
Mackey Glass, Nh=10				
Training	4.62E-03	7.46E-05	7.48E-05	2.64E-03
Testing	5.89E-03	9.76E-05	9.73E-05	4.17E-03
Narendra Time Series, Nh=30				
Training	2.25E-03	1.36E-04	1.32E-04	7.70E-02
Testing	1.20E-02	1.99E-03	1.94E-03	6.85E-02
Henon Map, Nh=10				
Training	5.94E-02	4.99E-03	5.30E-03	6.54E-03
Testing	6.94E-02	6.50E-03	6.76E-03	3.81E-03
Chen Series, Nh=10				
Training	5.21E-02	1.80E-02	3.85E-03	7.02E-02
Testing	6.03E-02	2.23E-02	1.78E-02	6.27E-02

Table 4.3 Performance comparison (by MSE) of ELM and RPE using Fuzzy C-means Clustering

	ELM	ELM {Fuzzy C-means}	FD-RPE {Fuzzy C-means}	Grad-RPE {Fuzzy C-means}
SELS, Nh=5				
Training	3.40E-02	1.78E-03	1.80E-03	2.50E-02
Testing	3.68E-02	1.88E-03	1.90E-03	3.10E-02
Mackey Glass, Nh=10				
Training	4.62E-03	9.00E-05	1.40E-04	6.49E-03
Testing	5.89E-03	1.57E-04	1.53E-04	7.48E-03
Narendra Time Series, Nh=30				
Training	2.25E-03	1.17E-04	1.01E-04	2.62E-03
Testing	1.20E-02	1.59E-03	1.42E-03	1.79E-02
Henon Map, Nh=10				
Training	5.94E-02	4.80E-03	4.78E-03	4.63E-03
Testing	6.94E-02	6.11E-03	6.10E-03	4.80E-03
Chen Series, Nh=10				
Training	5.21E-02	3.40E-03	6.12E-03	7.44E-02
Testing	6.03E-02	2.12E-02	2.85E-02	8.23E-02

Table 4.2 and Table 4.3 show the performance comparison of finite difference based on smart clustering technique. The simulation results also include the real gradient estimate as a control benchmark. Compared to Table 4.1, there is slight improvement when clustering technique being performed to weight initialization of RBF.

Another observation is that finite difference gradient estimate outperforms the real gradient of RBF for complex nonlinear case studies even for constant time sample data. This leads to look into the feasibility of using derivative free estimate to tediously defined mathematical derived option of real gradient. In addition, the real gradient approaches only valid for one-step-ahead prediction which is not the case for finite difference RPE for RBF.

Finally both smart clustering techniques performed fairly comparable to each other but outperform the random initialization technique and the ELM. The next section will look at decomposed recursive prediction error (RPE) by taking advantage of parallel structure of neural networks (McLoone et al., 1997 and Asirvadam, V.S et al., 2002).

4.4. Finite Difference Recursive Update on Decomposed RBF Networks

One of the drawbacks of finite difference gradient approach learning is the high dimension of the nonlinear parameter vector that consist of all nonlinear parameters of RBF networks which needs higher computational and storage requirements consequently. Decomposing the nonlinear parameter vector seems to be an efficient approach for them. By discarding the inter-neuron nonlinear parameter interaction, the vector is decomposed neuron by neuron (Asirvadam Vijanth S., 2004, 2005, 2006).

P_t in (4.28) can be decomposed on a neuron by neuron as stated below (Asirvadam Vijanth S., 2004).

$$P_t \cong \begin{bmatrix} [P_{t(1)}]_{nwxnw} & \dots & 0 & 0 \\ \dots & [P_{t(2)}]_{nwxnw} & \dots & \dots \\ 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & [P_{t(m)}]_{nwxnw} \end{bmatrix}_{NLxNL} \tag{4.46}$$

Where $P_{t(i)}$ represent correlation parameters for the i^{th} neuron and nw represents the number of the nonlinear parameters.

The gradient vector also can be decomposed as followed

$$D(c_t, \sigma_t) = \left[D(c_{t(1)}, \sigma_{t(2)}), ..., D(c_{t(m)}, \sigma_{t(m)})\right]^T_{nwx1} \tag{4.47}$$

The decomposed recursive Finite Difference on RPE is obtained as follows;

$$\nabla p_{t(i)} = \gamma_m \nabla p_{t-1(i)} + \gamma_g \nabla p_{t(i)} error \tag{4.48}$$

$$P_{t(i)} = \frac{1}{\lambda}\left\{P_{t-1(i)} - [P_{t-1(i)} \times D' KDP_{t-1(i)}]\right\} \tag{4.49}$$

$$\Theta_{t(i)} = \Theta_{t-1(i)} + P_{t(i)} \nabla P_{t(i)} \tag{4.50}$$

The decomposed weight update technique differs to full update only on the nonlinear update part by decomposing the weights to each neuron. Thus, reduce the memory requirement.

4.4.1 Finite Difference RPE on Decomposed RBF Networks (FD-dRPE Rand)

Similar to previously proposed learning method on this chapter, **FD-dRPE Rand** consist of two phase which is initialization phase and learning or training phase. The difference on the learning phase of nonlinear weights update where the decomposed strategies are used here.

Table 4.4 presents MSE comparison to show the accuracy comparison between ELM and Finite Difference RPE on Decomposed RBF Networks (FD-dRPE) using random initialization.

Table 4.4 Performance Comparison (by MSE) of ELM and RPE on Decomposed RBF using Random Initialization.

	ELM	FD-dRPE-{Rand}
SELS, Nh=5		
Training	3.40E-02	2.10E-02
Testing	3.68E-02	2.16E-02
Mackey Glass, Nh=10		
Training	4.62E-03	6.77E-03
Testing	5.89E-03	8.18E-03
Narendra Time Series, Nh=30		
Training	2.25E-03	1.27E-04
Testing	1.20E-02	1.45E-03
Henon Map, Nh=10		
Training	5.94E-02	7.06E-03
Testing	6.94E-02	9.65E-03
Chen Series, Nh=10		
Training	5.21E-02	1.33E-02
Testing	6.03E-02	2.59E-02

Table 4.4 shows the performance of decomposed RPE to ELM which depict the superiority of the proposed techniques. One of the examples that can be seen from five benchmarks is on Henon Map. FD-dRPE{Rand} succeeds to reduce the value of MSE which means has better accuracy compared to ELM.

Tables 4.5-4.8 show the conclusive results simulation of four benchmark problems with various training data feed online. For most cases, the decomposed version converges fast with less number of training data and best performance in accuracy. Overall, the finite difference (both full and decomposed) techniques performed better compared to ELM.

Table 4.5 Convergence rate among ELM RPE and Decomposed RPE using Random Initialization for SELS Case Study

Number of train data	Method		
	ELM	FD-RPE {Rand}	FD- dRPE {Rand}
300	4.17E+00	4.13E+00	2.34E-02
500	1.44E-01	1.67E-01	2.18E-02
1000	3.93E-02	3.88E-02	2.10E-02
2000	3.48E-02	3.43E-02	2.08E-02
3000	3.40E-02	3.39E-02	2.10E-02

Table 4.5 performs the capability of each learning method to reach convergence for SELS. It can be seen for SELS, FD-dRPE{Rand} outperforms the other method

including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 4.6 Convergence rate among ELM RPE and Decomposed RPE using Random Initialization for Mackey-Glass Test Case

Number of train data	Method		
	ELM	FD-RPE {RAND}	FD- DRPE {RAND}
300	1.74E+04	1.83E+05	1.82E-01
500	1.37E-01	1.47E-01	1.48E-02
1000	2.27E-02	1.82E-02	8.36E-03
2000	5.84E-03	4.45E-03	7.00E-03
3000	4.62E-03	3.70E-03	6.77E-03

Table 4.6 performs the capability of each learning method to reach convergence for Mackey Glass. It can be seen for Mackey Glass, FD-dRPE{Rand} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 4.7 Convergence rate among ELM RPE and Decomposed RPE using Random Initialization for Narendra's Dynamic System

Number of train data	Method		
	ELM	FD-RPE {Rand}	FD- dRPE {Rand}
300	2.10E+06	4.28E+03	3.87E-04
500	1.36E+01	8.17E+00	2.33E-04
1000	4.16E+00	2.15E-03	1.44E-04
2000	2.25E-03	1.36E-03	1.27E-04
3000	2.25E-03	2.14E-03	1.27E-04

Table 4.7 performs the capability of each learning method to reach convergence for Narendra's Dynamic System. It can be seen for Narendra's Dynamic System, FD-dRPE{Rand} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{Rand} convergences very fast. It starts to converge in the first 300

training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 4.8 Convergence rate among ELM RPE and Decomposed RPE using Random Initialization for Henon's Chaotic Map

Number of train data	Method		
	ELM	FD-RPE {Rand}	FD- dRPE {Rand}
300	3.10E+04	4.18E+04	1.91E-02
500	2.54E+00	6.48E-01	9.31E-03
1000	6.25E-02	4.02E-01	7.57E-03
2000	6.22E-02	8.43E-03	7.64E-03
3000	5.94E-02	7.66E-03	7.06E-03

Table 4.8 performs the capability of each learning method to reach convergence for Henon's Chaotic Map. It can be seen for Henon's Chaotic Map, FD-dRPE{Rand} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 4.9 Convergence rate among ELM RPE and Decomposed RPE using Random Initialization for Chen Nonlinear Time Series

Number of train data	Method		
	ELM	FD-RPE {Rand}	FD- dRPE {Rand}
300	1.09E+02	1.26E-01	1.50E-02
500	5.72E-02	1.68E-02	1.44E-02
1000	4.75E-02	9.54E-03	1.29E-02
2000	5.23E-02	7.55E-03	1.44E-02
3000	5.21E-02	8.10E-03	1.33E-02

Table 4.9 performs the capability of each learning method to reach convergence for Chen Nonlinear Time Series. It can be seen for Chen Nonlinear Time Series, FD-dRPE{Rand} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300

training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Finite difference recursive on decomposed RBF update procedure has contributed in making fast convergence which means decreasing the number of training data. In addition, it also decreases the storage requirement since we just concern on the diagonal part of the covariance matrix.

4.4.2 Finite Difference Recursive Update on Decomposed RBF Networks with Smart Clustering Initialization

Similar to the proposed learning method on previous sections, **FD-dRPE Rand** as the decomposed variant of RPE for RBF is tested with smart clustering techniques for nonlinear initialization. They are the finite difference RPE with K- Means (**FD-RPE K-means**) and the finite difference RPE with Fuzzy C-means (**FD-RPE Fuzzy C-Means)**. The difference is on the initialization stage. The nonlinear weights will be updated using decomposed strategies.

Table 4.10 and Table 4.16 show the performance of decomposed techniques using K-means and Fuzzy C-means respectively for all cases. The decomposed finite difference approach performed better than the ELM techniques. Compared to the results in Table 4.4, the adopting K-means and Fuzzy C-means with decomposed finite difference do improve slight on the end results.

Table 4.10 Performance (by MSE) comparison of ELM, and RPE on Decomposed RBF using K-means

	ELM	ELM {K-means}	FD-dRPE {K-means}
SELS, Nh=5			
Training	3.40E-02	1.86E-03	1.66E-03
Testing	3.68E-02	1.93E-03	1.74E-03
Mackey Glass, Nh=10			
Training	4.62E-03	7.46E-05	7.42E-05
Testing	5.89E-03	9.76E-05	1.77E-04
Narendra Time Series, Nh=30			
Training	2.25E-03	1.36E-04	1.74E-04
Testing	1.20E-02	1.99E-03	2.07E-03
Henon Map, Nh=10			
Training	5.94E-02	4.99E-03	5.30E-03
Testing	6.94E-02	6.50E-03	6.88E-03
Chen Series, Nh=10			
Training	5.21E-02	1.80E-02	7.62E-03
Testing	6.03E-02	2.23E-02	1.21E-02

All MSE comparisons in Table 4.10 show that FD-dRPE{K-means} has the same performance as ELM{K-means} in terms of accuracy. Both methods outperform ELM by having lower MSE.

Tables 4.11-4.15 conclude the performance of decomposed finite difference technique when applied K-means clustering technique on its weight initialization process. The tables present the comparison of the convergence rate among the decomposed technique using K-means clustering initialization, Finite Difference RPE using K-means clustering and ELM.

Table 4.11 Performance comparison (by MSE) of ELM, and RPE on Decomposed RBF using K-means for SELS system

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE { K-means}
300	4.17E+00	9.97E+01	2.50E+02	4.90E-03
500	1.44E-01	1.33E+00	2.02E+00	1.72E-03
1000	3.93E-02	1.62E-01	1.42E-01	1.59E-03
2000	3.48E-02	9.17E-03	2.35E-03	1.64E-03
3000	3.40E-02	1.86E-03	1.66E-03	1.66E-03

Table 4.11 performs the capability of each learning method to reach convergence for SELS. It can be seen for SELS, FD-dRPE{K-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ K-means } converges very fast. It starts to converge

in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.12 Performance comparison (by MSE) of ELM, and RPE on Decomposed RBF using K-means for Mackey Glass

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE { K-means}
300	1.74E+04	1.86E+06	1.58E+02	5.78E-02
500	1.37E-01	5.74E+01	1.07E+00	7.18E-04
1000	2.27E-02	1.18E-01	6.38E-01	2.38E-04
2000	5.84E-03	9.48E-05	9.44E-05	9.46E-05
3000	4.62E-03	7.46E-05	7.48E-05	7.42E-05

Table 4.12 performs the capability of each learning method to reach convergence for Mackey Glass. It can be seen for Mackey Glass, FD-dRPE{K-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ K-means } converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.13 Performance comparison (by MSE) of ELM, and RPE on Decomposed RBF using K-means for Narendra Dynamic System

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE { K-means}
300	2.10E+06	3.44E+01	3.40E+01	3.89E-04
500	1.36E+01	7.95E+03	3.73E+00	2.33E-04
1000	4.16E+00	1.13E+01	5.08E-02	2.15E-04
2000	2.25E-03	7.37E-03	4.42E-03	1.74E-04
3000	2.25E-03	1.36E-04	1.32E-04	1.74E-04

Table 4.13 performs the capability of each learning method to reach convergence for Narendra Dynamic System. It can be seen for Narendra Dynamic System, FD-dRPE{K-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ K-means } converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.14 Performance comparison (by MSE) of ELM, and RPE on Decomposed RBF for system using K-means for Henon Map Series.

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE { K-means}
300	3.10E+04	5.99E+05	6.56E+05	1.01E-02
500	2.54E+00	5.07E-01	5.84E-01	7.37E-03
1000	6.25E-02	1.68E+04	7.56E-03	6.20E-03
2000	6.22E-02	5.47E-03	6.33E-03	5.79E-03
3000	5.94E-02	4.99E-03	5.30E-03	5.30E-03

Table 4.14 performs the capability of each learning method to reach convergence for Henon Map Series. It can be seen for Henon Map Series, FD-dRPE{K-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ K-means } converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.15 Performance comparison of ELM, and RPE on Decomposed RBF using K-means for Chen Nonlinear Time Series.

num of learning data	methods			
	ELM	ELM K-means	FD RPE K-means	Dec FD RPE K-means
300	1.09E+02	1.10E+03	2.47E-01	9.69E-03
500	5.72E-02	1.85E-02	1.33E-02	8.21E-03
1000	4.75E-02	1.81E-02	4.05E-03	8.31E-03
2000	5.23E-02	1.95E-02	5.65E-03	8.01E-03
3000	5.21E-02	1.80E-02	3.85E-03	7.62E-03

Table 4.15 performs the capability of each learning method to reach convergence for Chen Nonlinear Time Series. It can be seen for Chen Nonlinear Time Series, FD-dRPE{K-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ K-means }converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to go convergence very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.16 Performance comparison of ELM, and RPE on Decomposed RBF for system using Fuzzy C-means

	ELM	ELM Fuzzy C-means	Dec FD RPE Fuzzy C-means
SELS, Nh=5			
Training	3.40E-02	1.78E-03	1.85E-03
Testing	3.68E-02	1.88E-03	1.95E-03
Mackey Glass, Nh=10			
Training	4.62E-03	9.00E-05	8.39E-05
Testing	5.89E-03	1.57E-04	1.90E-04
Narendra Time Serise, Nh=30			
Training	2.25E-03	1.17E-04	1.27E-04
Testing	1.20E-02	1.59E-03	1.45E-03
Henon Map, Nh=10			
Training	5.94E-02	4.80E-03	4.75E-03
Testing	6.94E-02	6.11E-03	6.04E-03
Chen Series, Nh=10			
Training	5.21E-02	3.40E-03	7.50E-03
Testing	6.03E-02	2.12E-02	1.22E-02

All MSE comparisons in Table 4.16 show that FD-dRPE{ Fuzzy C-means} has the same performance as ELM{ Fuzzy C-means} in terms of accuracy. Both methods outperform ELM by having lower MSE.

Table 4.17-4.20 conclude the performance of decomposed finite difference technique when applied K-means clustering technique on its weight initialization process. The tables present the comparison of the convergence rate among the decomposed technique using Fuzzy C-means clustering initialization, Finite Difference RPE using Fuzzy C-means clustering and ELM.

Table 4.17 Performance comparison of ELM, and RPE on Decomposed RBF for system using Fuzzy C-means for SELS system

Number of train data	Methods			
	ELM	ELM { Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE { Fuzzy C-means}
300	1.74E+04	8.67E+03	1.21E+03	5.45E-02
500	1.37E-01	9.33E-02	2.34E-02	5.57E-04
1000	2.27E-02	1.33E+00	1.42E-01	2.29E-04
2000	5.84E-03	9.99E-05	1.05E-04	9.28E-05
3000	4.62E-03	9.00E-05	1.40E-04	8.39E-05

Table 4.17 performs the capability of each learning method to reach convergence for SELS. It can be seen for SELS, FD-dRPE{Fuzzy C-means} outperforms the other method including

ELM in terms of convergence rate. FD-dRPE{ Fuzzy C-means} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to go converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.18 Performance comparison of ELM, and RPE on Decomposed RBF for system using Fuzzy C-means for Mackey Glass

Number of train data	Methods			
	ELM	ELM { Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE { Fuzzy C-means}
300	4.17E+00	9.93E+00	7.31E+00	5.07E-03
500	1.44E-01	1.07E-01	5.92E-02	1.92E-03
1000	3.93E-02	2.79E-02	2.18E-02	1.79E-03
2000	3.48E-02	2.05E-03	1.97E-03	1.84E-03
3000	3.40E-02	1.78E-03	1.80E-03	1.85E-03

Table 4.18 performs the capability of each learning method to reach convergence for Mackey Glass. It can be seen for Mackey Glass, FD-dRPE{Fuzzy C-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ Fuzzy C-means} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to go convergence very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.19 Performance comparison of ELM, and RPE on Decomposed RBF for system using Fuzzy C-means for Narendra's dynamic system

Number of train data	Methods			
	ELM	ELM { Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE { Fuzzy C-means}
300	2.10E+06	2.24E+06	5.06E+04	3.87E-04
500	1.36E+01	5.07E+00	4.51E+03	2.33E-04
1000	4.16E+00	1.54E-02	1.31E-02	1.44E-04
2000	2.25E-03	2.14E-04	2.50E-04	1.27E-04
3000	2.25E-03	1.17E-04	1.01E-04	1.27E-04

Table 4.19 performs the capability of each learning method to reach convergence for Narendra's dynamic system. It can be seen for Narendra's dynamic system, FD-dRPE{Fuzzy C-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE{ Fuzzy C-means} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

Table 4.20 Performance comparison of ELM, and RPE on Decomposed RBF for system using Fuzzy C-means for Henon Map Series

Number of train data	Methods			
	ELM	ELM { Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE { Fuzzy C-means}
300	3.10E+04	8.06E+03	8.93E+03	8.57E-03
500	2.54E+00	7.20E-02	8.13E-02	6.16E-03
1000	6.25E-02	6.22E-03	6.23E-03	5.88E-03
2000	6.22E-02	3.89E-02	8.83E-03	5.20E-03
3000	5.94E-02	4.80E-03	4.78E-03	4.75E-03

Table 4.20 performs the capability of each learning method to reach convergence for Narendra's dynamic system. It can be seen for Narendra's dynamic system, FD-dRPE{Fuzzy C-means} outperforms the other method including ELM in terms of convergence rate. FD-dRPE
{Fuzzy C-means} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data. In addition, this method succeeds to increase the accuracy compared to ELM by having lower MSE.

The tables summarize that the decomposed technique convergence fast. With only 300 training data, its MSE is near to the optimal values. The smart clustering, both K-means and Fuzzy C-means techniques, did contribute to good convergence of mean squared error compared to random initialization. The decomposed techniques also lead to reduce memory requirement for nonlinear parameter update with good performance accuracy in overall.

Figures 4.2-4.5 depict clearly the performance of all the techniques discussed to four benchmark problems. For all cases it can be seen the decomposed techniques convergence fast with less number of iterations. But as the iteration increase the other method started to converge to a saturation value.

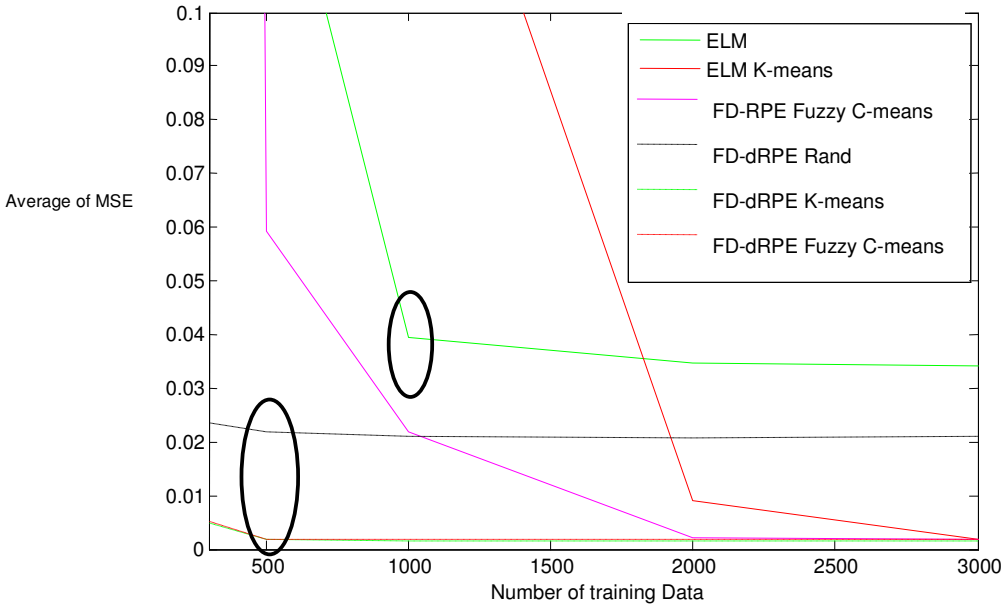


Figure 4.2 Convergence rates of SELS

Convergence performance of SELS can be seen on Figure 4.2. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge after 1000 training data but getting stable after 2000 training data. In addition, the figure shows that ELM cannot reach the smallest average of MSE as small as Finite Difference RPE on decomposed RBF.

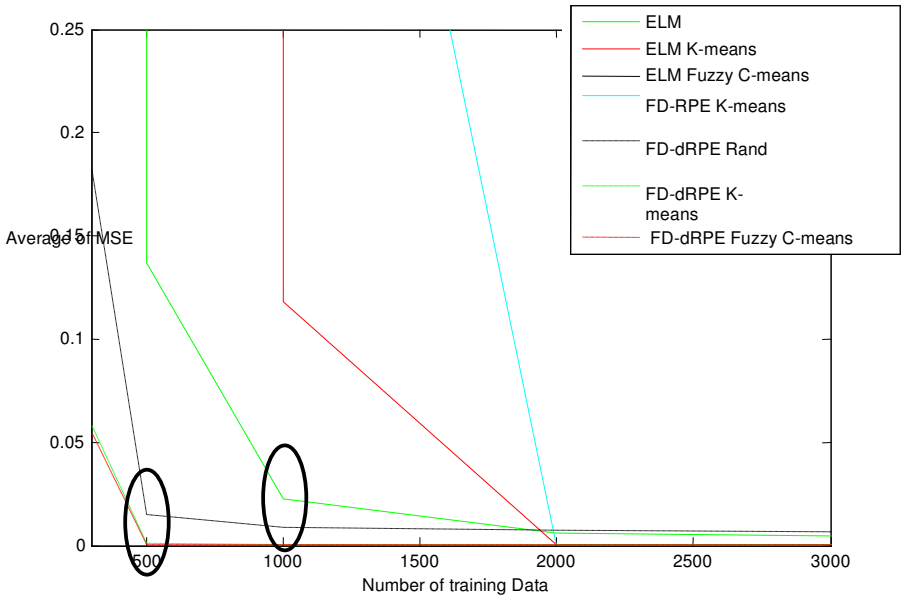


Figure 4.3 Convergence rate of Mackey Glass

Convergence performance of Mackey Glass can be seen on Figure 4.3. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). Especially for Finite Difference RPE on decomposed RBF which is getting stable after 1000 training data. On the other hand, other methods do not. ELM which is represented by green solid line start converge after 1000 training data but getting stable after 2000 training data.

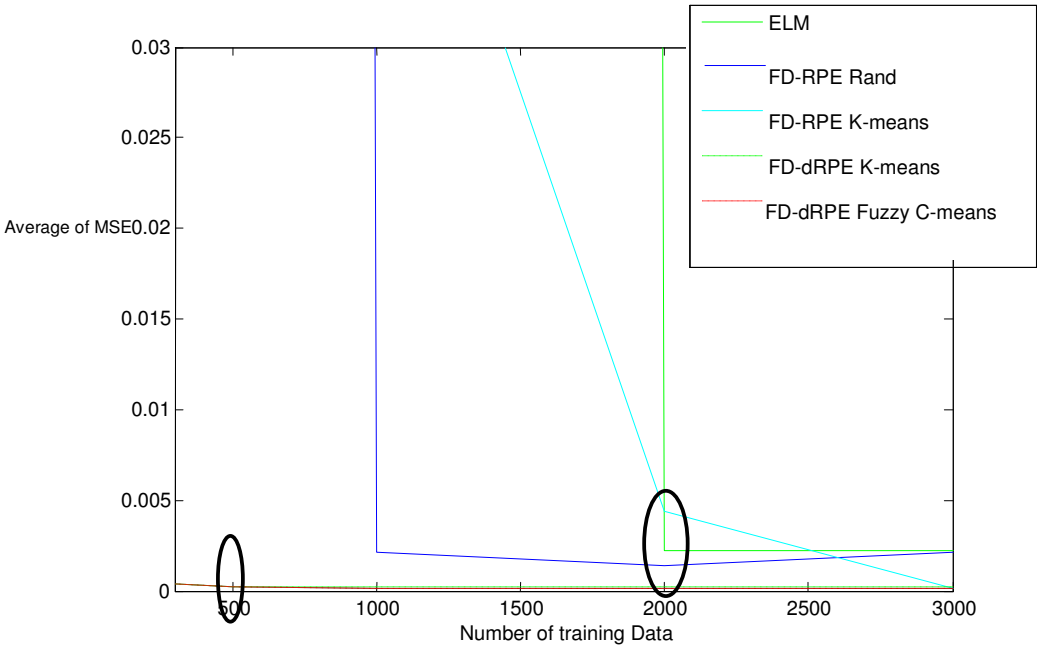


Figure 4.4 Convergence rate of Narendra Dynamic System

Convergence performance of Narendra Dynamic System can be seen on Figure 4.4. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge and getting stable after 2000 training data. Through this comparison, it can be observed that by using Finite Difference RPE on decomposed RBF, 1500 training data can be saved. In addition, the figure shows that ELM cannot reach the smallest average of MSE as small as Finite Difference RPE on decomposed RBF.

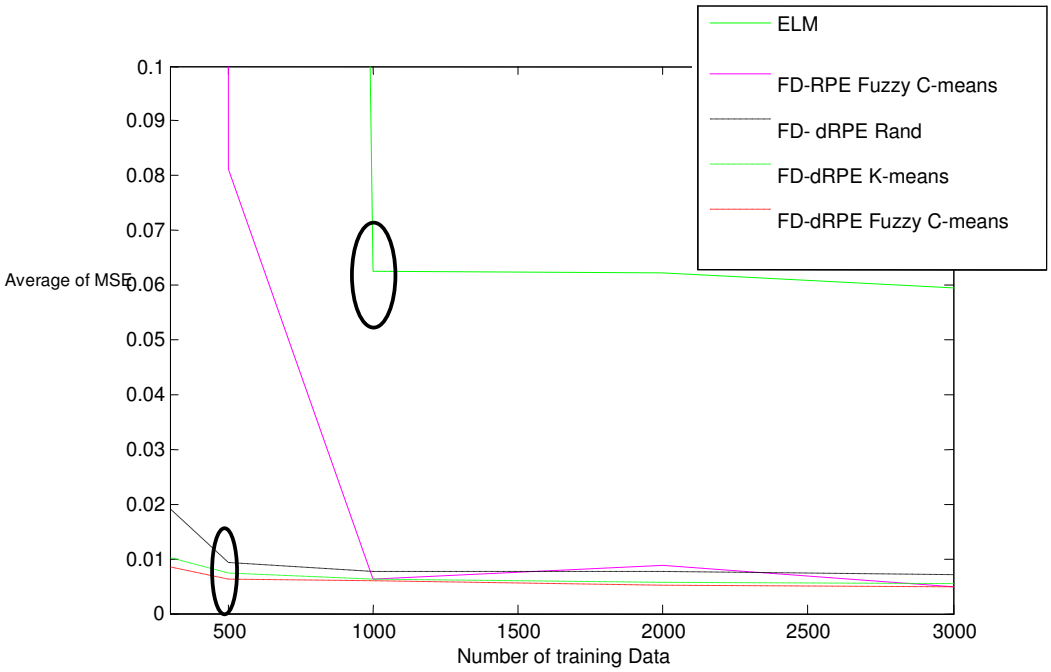


Figure 4.5 Convergence rate of Henon Map

Convergence performance of Henon Map System can be seen on Figure 4.5. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge and getting stable after 1000 training data. Through this comparison, it can be observed that by using Finite Difference RPE on decomposed RBF, 500 training data can be saved. Besides that, the figure shows that ELM cannot reach the smallest average of MSE as small as Finite Difference RPE on decomposed RBF.

4.5 Conclusion

This chapter discusses the adaptation of finite difference recursive RPE weight update for the RBF networks. The improved version of finite difference recursive update proposed uses time varying step sizes in this chapter is comparable or in some cases better than the real gradient. This leads to the adaptation finite difference or derivative techniques for the system identification application.

In this chapter also, the influence of smart clustering being look into when the nonlinear weight being updated and from extensive simulation studies the clustering did show improvement in the final value of error convergence.

Finally a decomposed version of finite difference RPE being proposed shows promising results where it convergence fast and gives good final value in performance accuracy. The smart clustering weight initialization of decomposed RPE also has been investigated for improvement.

The next chapter will look into the nonlinear system identification for lost packets where the feasibility of finite difference learning is at most important.

CHAPTER 5

RBF RECURSIVE LEARNING WITH LOST PACKETS

Chapter 3 and Chapter 4 have discussed the implementation of the training algorithm on system identification with regular time. The training data is always available on every single simulation time or known as constant sample time. This chapter deals with implementing the training algorithm on system identification with lost packets. The term packet means encapsulation of the input and output of the system with the time stamp.

5.1 System Identification with Lost packets

In practice, there are many applications in which observation of the discrete system are not available at every sample time, especially in the case of wireless system which getting large popularity now (Musab, J.O. Elamin et al., 2007). Thus, some observations may be missing at periodic manner or even randomly.

Sensor failure, non response in statistical surveys and outlier are some examples of randomly missing data, whereas periodically missing data may appear in astronomical experiments, radar scans, time sharing of sensors and multi-rate sampling (Pintelon R et al., 2000; Isakson Alf J et al., 1993; Wallin, R and Isakson Alf J., 2002).

The termed lost packets in the chapter mean there is lost input feed in random in an online learning framework. The chapter will look into the feasibility studies on the performance of finite difference RPE and the decomposed version to lost packets.

The real gradient derivation is only valid for system with one-step-ahead prediction. Thus, for system with lost packets it should be viewed as multi-step-ahead prediction problem. Therefore, finite difference gradient estimate is needed.

5.1.1 Simulating the Lost packets Recursive Environment.

The system Input/output observations are assumed to be time stamped, encapsulated into packets and transmitted over congested environment to the modeler (RBF networks) where the packet is considered lost if it does not been received after a certain time. Assumptions made that the lost observations are not retransmitted which is in accordance to transport layer protocol in UDP-like and the modeler either receives the observation in full or none in a single packet (both input and output). Thus the arrival of the packets at the identifier side is modeled as a Bernoulli process (β_k) which has two possible values 0 and 1 at any instant time (2), where:

$$\beta = \begin{cases} 1 & \text{Indicates the packet has been received successfully.} \\ 0 & \text{Indicates the packet has been completely lost during the transmission.} \end{cases}$$

The probability of β is random with time and known as the Packet Dropping Probability (PDP) (Asirvadam and Musab., 2009, Shcenato et al., 2007). The modeling of the observations arrival via Bernoulli process is illustrated in Figure 5.1.

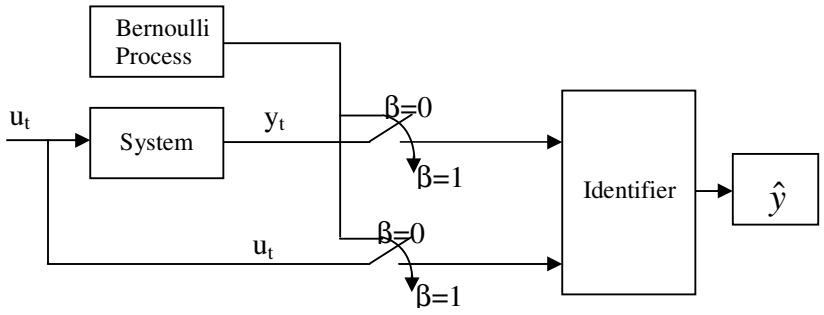


Figure 5.1: Modeling the arrival of the Input/Output using Bernoulli process.

The system identification approach to lost packets is also known as identifying a system with irregular sample time. In this chapter the probability of dropping packets

is set to 0.5 for all simulations which mean 50% of the input-output which for the input structure of NARX fail to feed in to the RBF networks during its recursive training.

5.2 Finite Difference Recursive update for RBF Networks with Lost packets

This section investigates the performance of finite difference recursive update for RBF networks for all benchmarks in the case of with lost packets. Table 5.1 presents the performance of finite difference (derivative free) output. It also performs the true gradient based as anticipated due to irregular sampled input feed.

Table 5.1 Performance comparison using Random Initialization for the System with Lost packets (PDP set to 0.5)

	ELM	Grad -RPE {Rand}	FD-RPE {Rand}	FD-dRPE-{Rand}
SELS, Nh=5				
Training	2.62E-01	6.56E-02	3.90E-02	1.22E-02
Testing	2.62E-01	5.76E-02	4.66E-02	2.08E-02
Mackey Glass, Nh=10				
Training	1.04E-01	1.83E-02	9.70E-03	1.00E-02
Testing	9.51E-01	1.49E-02	9.22E-03	1.38E-02
Narendra Time Serise, Nh=20				
Training	1.52E-02	1.93E-02	1.54E-02	2.12E-02
Testing	3.09E-02	5.40E-02	2.80E-02	3.36E-02
Henon Map, Nh=20				
Training	5.86E-01	5.97E-01	5.83E-01	5.65E-01
Testing	5.72E-01	5.89E-01	5.73E-01	5.85E-01
Chen Series, Nh=20				
Training	5.79E-01	5.72E-01	5.74E-01	5.75E-01
Testing	6.01E-01	6.19E-01	6.17E-01	6.13E-01

Accuracy performance based on MSE value for each benchmarks has been compared in Table 5.1. Through the simulation results, Finite Difference applied on RBF networks (FD-RPE{Rand}) and decomposed RBF networks (FD-dRPE{Rand}) able to perform better by having lower MSE value compared to ELM. In addition, Finite Difference and the actual gradient are comparable. However, Finite Difference sometime performs better compared to the actual gradient as on Mackey Glass Time series problem.

Tables 5.2-5.5 present results of convergence in the case of lost packets with variable number of training data in the case of random initialization. The tables record the performance of the MSE values for every 300, 500, 1000, 2000 and 3000 training data to examine the convergence performance for each benchmark with the whole method by random initialization.

Table 5.2 Performance comparison using Random Initialization for the System with Lost packets (PDP set to 0.5) for SELS system

Number of train data	Methods		
	ELM	FD-RPE {Rand}	FD-dRPE { Rand}
300	2.46E+02	1.44E+05	7.80E-01
500	2.06E-01	6.13E-02	2.16E-02
1000	1.98E-01	1.11E-01	1.28E-02
2000	1.47E-01	1.82E-02	1.11E-02
3000	1.04E-01	9.70E-03	1.00E-02

Table 5.2 performs the capability of each learning method in case of random initialization to reach convergence for SELS. It can be seen for SELS, FD-dRPE{Rand} outperforms the other methods including ELM in terms of convergence rate. It converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 5.3 Performance comparison using Random Initialization for the System with Lost packets (PDP set to 0.5) for Mackey Glass system

Number of train data	Methods		
	ELM	FD-RPE {Rand}	FD-dRPE { Rand}
300	3.53E+02	7.36E+05	2.01E-02
500	1.08E+00	6.23E-02	2.18E-02
1000	3.31E-01	2.13E-02	2.10E-02
2000	2.82E-01	1.64E-02	2.08E-02
3000	2.62E-01	1.21E-02	2.10E-02

Table 5.3 performs the capability of each learning method in case of random initialization to reach convergence for Mackey Glass system. It can be seen for Mackey Glass Time Series system, FD-dRPE{Rand} outperforms the other method including ELM in terms of convergence rate. It converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 5.4 Performance comparison using Random Initialization for the System with Lost packets (PDP set to 0.5) for Henon Map system

Number of train data	Methods		
	ELM	FD-RPE {Rand}	FD-dRPE { Rand}
300	1.73E+05	1.05E+08	5.29E-01
500	2.25E+01	2.27E+01	6.62E-01
1000	4.17E+02	1.19E+00	7.16E-01
2000	5.71E-01	5.79E-01	6.72E-01
3000	5.86E-01	5.83E-01	5.65E-01

Table 5.2 performs the capability of each learning method in case of random initialization to reach convergence for Henon Map system. It can be seen for Henon Map system, FD-dRPE{Rand} outperforms the other methods including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

Table 5.5 Performance comparison using Random Initialization for the System with Lost packets (PDP set to 0.5) for Chen Series

Number of train data	Methods		
	ELM	FD-RPE {Rand}	FD-dRPE { Rand}
300	7.99E+00	5.71E+01	5.48E-01
500	1.98E+01	2.64E+01	6.80E-01
1000	3.42E+01	9.10E+01	5.65E-01
2000	1.42E+01	5.80E-01	5.83E-01
3000	5.79E-01	5.74E-01	5.75E-01

Table 5.5 performs the capability of each learning method in case of random initialization to reach convergence for Nonlinear Chen Series Problem. It can be seen for Nonlinear Chen Series problem, FD-dRPE{Rand} outperforms the other methods including ELM in terms of convergence rate. FD-dRPE{Rand} converges very fast. It starts to converge in the first 300 training data while the others do not. Its capability to converge very fast is worth to decrease the number of training data.

5.2.1 Adopting K-means Initialization to Lost packets Problems

In this section the influence of K-means clustering to RBF parameter initialization is investigated for the case of system with lost packets. The prior initial

of weight with unsupervised learning technique do show good performance measure and RBF generalization on testing data.

Tables 5.6 presents the performance of finite difference or derivate free gradient estimate for RBF weight update with K-means weight initialization. Compared to the performance in Table 5.1, the training and testing error has improved using clustering techniques.

Table 5.6 Performance comparison using K-means Initialization for the System with Lost packets (PDP=0.5)

	ELM	ELM {K-means}	FD-dRPE {K-means}	FD-dRPE {K-means}
SELS, Nh=5				
Training	2.62E-01	1.22E-02	1.22E-02	1.21E-02
Testing	2.62E-01	2.08E-02	2.08E-02	2.07E-02
Mackey Glass, Nh=10				
Training	1.04E-01	5.13E-02	2.66E-03	2.64E-03
Testing	9.51E-01	5.09E-02	2.60E-03	2.60E-03
Narendra Time Serise, Nh=20				
Training	1.52E-02	4.33E-03	5.39E-03	4.27E-03
Testing	3.09E-02	6.76E-03	8.12E-03	6.49E-03
Henon Map, Nh=20				
Training	5.86E-01	5.95E-01	6.05E-01	5.65E-01
Testing	5.72E-01	5.74E-01	5.61E-01	5.79E-01
Chen Series, Nh=20				
Training	5.79E-01	5.79E-01	5.79E-01	5.79E-01
Testing	6.01E-01	6.01E-01	6.01E-01	6.01E-01

Accuracy performance based on MSE value for each benchmarks has been compared in Table 5.6. Through the simulation's results in Table 5.6, Finite Difference applied on RBF networks (FD-RPE{K-means}) and decomposed RBF networks (FD-dRPE{K-means}) able to perform better by having lower MSE value compared to ELM and ELM{K-means}. It means that Finite Difference on the nonlinear update combining with K-means clustering on the initialization has significance effect to increase the accuracy.

Tables 5.7-5.9 present results of the convergence in the case of lost packets with variable number of training data for random initialization. The tables record the performance of the MSE values for every 300, 500, 1000, 2000 and 3000 training data

to examine the convergence performance for each benchmark with the whole method by K-means clustering initialization.

Table 5.7 Performance comparison using K-means for the System with Lost packets (PDP set to 0.5) for SELS system

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE {K-means}
300	3.53E+02	7.36E+05	3.53E+02	2.28E-02
500	1.08E+00	6.23E-02	7.26E+04	1.25E-02
1000	3.31E-01	2.11E-02	2.47E-01	2.09E-02
2000	2.82E-01	1.71E-02	4.27E-02	1.65E-02
3000	2.62E-01	1.28E-02	3.90E-02	1.22E-02

Movement of MSE among ELM, ELM{K-means}, FD-RPE{K-means}, and FD-dRPE{K-means} on SELS has been compared in Table 5.7. Through the simulation results, it can be concluded that decomposed algorithm really works to increase the convergence rate. FD-dRPE{K-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{K-means} outperforms ELM in terms of accuracy.

Table 5.8 Performance comparison using K-means for the System with Lost packets (PDP set to 0.5) for Mackey Glass system

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE {K-means}
300	2.46E+02	8.28E+00	3.12E+02	2.54E-01
500	2.06E-01	2.17E+01	4.70E-01	3.61E-03
1000	1.98E-01	3.46E-01	7.17E+00	2.68E-03
2000	1.47E-01	2.16E+00	2.49E-01	3.06E-03
3000	1.04E-01	5.13E-02	2.66E-03	2.64E-03

Movement of MSE among ELM, ELM{K-means}, FD-RPE{K-means}, and FD-dRPE{K-means} on Mackey Glass system has been compared in Table 5.8. Through the simulation results, it can be concluded that decomposed algorithm really works to increase the convergence rate. FD-dRPE{K-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{K-means} outperforms ELM and ELM{K-means} in terms of accuracy.

Table 5.9 Performance comparison using K-means for the System with Lost packets (PDP set to 0.5) for Narendra Dynamic System

Number of train data	Methods			
	ELM	ELM {K-means}	FD-RPE {K-means}	FD-dRPE {K-means}
300	4.17E+04	3.29E+03	2.19E+02	5.09E-03
500	5.95E+00	1.04E+02	4.94E+00	3.98E-03
1000	4.62E+03	8.16E+00	3.76E-02	4.40E-03
2000	1.57E-02	1.02E-01	1.06E-02	4.89E-03
3000	1.52E-02	4.33E-03	5.39E-03	4.27E-03

Movement of MSE among ELM, ELM{K-means}, FD-RPE{K-means}, and FD-dRPE{K-means} on Narendra Dynamic System has been compared in Table 5.9. Through the simulation results, it can be concluded that decomposed algorithm really works to increase convergence rate. FD-dRPE{K-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{K-means} outperforms ELM in terms of accuracy.

Tables 5.7-5.9 show the performance of K-means with various training data measure. From the table of results, conclusion can be made that the performance of finite difference and the ELM vastly improve due prior initialization using clustering techniques. In overall, decomposed technique convergence fast with less training data and with inclusion of clustering technique the performance gets better.

5.2.2 Adopting Fuzzy C-means Initialization to Lost packets Problems

This section looks into the option of using Fuzzy C-means for prior setting the RBF weights before being trained adopted on lost packets environment. Table 5.10 illustrates the performance of RBF with fuzzy C-means. It performs better than random initialization.

Table 5.10 Performance comparison using Fuzzy C-meansInitialization for the System with Lost packets (PDP=0.5)

	ELM	ELM {Fuzzy C-means}	FD-RPE {Fuzzy C-means}	FD-dRPE {Fuzzy C-means}
SELS, Nh=5				
Training	2.62E-01	1.21E-02	2.10E-02	1.85E-03
Testing	2.62E-01	2.06E-02	2.16E-02	1.95E-03
Mackey Glass, Nh=10				
Training	1.04E-01	5.22E-02	2.75E-03	2.65E-03
Testing	9.51E-01	5.14E-02	2.63E-03	2.61E-03
Narendra Time Serise, Nh=20				
Training	1.52E-02	4.10E-03	5.18E-03	4.25E-03
Testing	3.09E-02	6.03E-03	6.52E-03	6.13E-03
Henon Map, Nh=20				
Training	5.86E-01	5.88E-01	5.87E-01	5.65E-01
Testing	5.72E-01	5.70E-01	5.69E-01	5.78E-01
Chen Series, Nh=20				
Training	5.79E-01	5.79E-01	6.21E-01	5.79E-01
Testing	6.01E-01	6.01E-01	6.02E-01	6.01E-01

Mackey Glass Time series problem can be one of the examples that can be observed from Table 5.10. It can be seen that Finite Difference on decomposed RBF networks can improve the accuracy by having lower MSE value compared to ELM and also ELM{Fuzzy C-means}. It can be concluded that Finite Difference on the nonlinear update combining with Fuzzy C-means clustering on the initialization has significance effect to increase the accuracy.

Tables 5.11 –Table 5.13 shows the performance of fuzzy C-mean clustering initialization technique by using various numbers of training data.

Table 5.11 Performance comparison using Fuzzy C-meansfor the System with Lost packets (PDP set to 0.5) for SELS system

Number of train data			Methods	
	ELM	ELM {Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE {Fuzzy C-Means}
300	3.53E+02	1.20E+05	1.20E+05	2.28E-02
500	1.08E+00	1.25E-02	1.25E-02	1.24E-02
1000	3.31E-01	2.05E-02	2.09E-02	2.08E-02
2000	2.82E-01	1.62E-02	1.65E-02	1.64E-02
3000	2.62E-01	1.22E-02	1.22E-02	1.21E-02

Movement of MSE among ELM, ELM{Fuzzy C-means}, FD-RPE{ Fuzzy C-means}, and

FD-dRPE{ Fuzzy C-means} on SELS system has been compared in Table 5.11. Through the simulation results, it can be concluded that decomposed algorithm really works to increase convergence rate. FD-dRPE{ Fuzzy C-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{ Fuzzy C-means} outperform ELM in terms of accuracy.

Table 5.12 Performance comparison using Fuzzy C-meansfor the System with Lost packets (PDP set to 0.5) for Mackey-Glass

Number of train data			Methods	
	ELM	ELM {Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE {Fuzzy C-Means}
300	2.46E+02	2.70E+01	7.57E+03	2.53E-01
500	2.06E-01	2.64E-01	8.57E-01	3.44E-03
1000	1.98E-01	1.26E-01	1.70E+02	2.60E-03
2000	1.47E-01	7.44E-01	3.16E-01	3.08E-03
3000	1.04E-01	5.22E-02	2.75E-03	2.65E-03

Movement of MSE among ELM, ELM{Fuzzy C-means}, FD-RPE{ Fuzzy C-means}, and FD-dRPE{ Fuzzy C-means} on Mackey-Glass system has been compared in Table 5.12. Through the simulation results, it can be concluded that decomposed algorithm really works to increase convergence rate. FD-dRPE{ Fuzzy C-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{ Fuzzy C-means} outperform ELM and ELM{Fuzzy C-means}in terms of accuracy.

Table 5.13 Performance comparison using Fuzzy C-meansfor the System with Lost packets (PDP set to 0.5) for Narendra Systems

Number of train data			Methods	
	ELM	ELM {Fuzzy C-Means}	FD-RPE {Fuzzy C-Means}	FD-dRPE {Fuzzy C-Means}
300	4.17E+04	1.41E+03	1.60E+04	5.22E-03
500	5.95E+00	1.25E-01	1.55E+00	3.83E-03
1000	4.62E+03	1.88E+01	9.76E-01	4.27E-03
2000	1.57E-02	2.91E-02	2.65E-02	4.75E-03
3000	1.52E-02	4.10E-03	5.18E-03	4.25E-03

Movement of MSE among ELM, ELM{Fuzzy C-means}, FD-RPE{ Fuzzy C-means}, and FD-dRPE{ Fuzzy C-means} on Narendra Systems has been compared in Table 5.13. Through the simulation results, it can be concluded that decomposed algorithm really works to increase convergence rate. FD-dRPE{ Fuzzy C-means} starts to converge since the first 300 training data while others do not. In addition, FD-dRPE{ Fuzzy C-means} outperform ELM in terms of accuracy.

Figures 5.2-5.4 depict the performance of various training algorithms for system with lost packets. The decomposed training variant show fast convergent and most of the training technique set to a saturation level when the iteration proceeds with large number of training data.

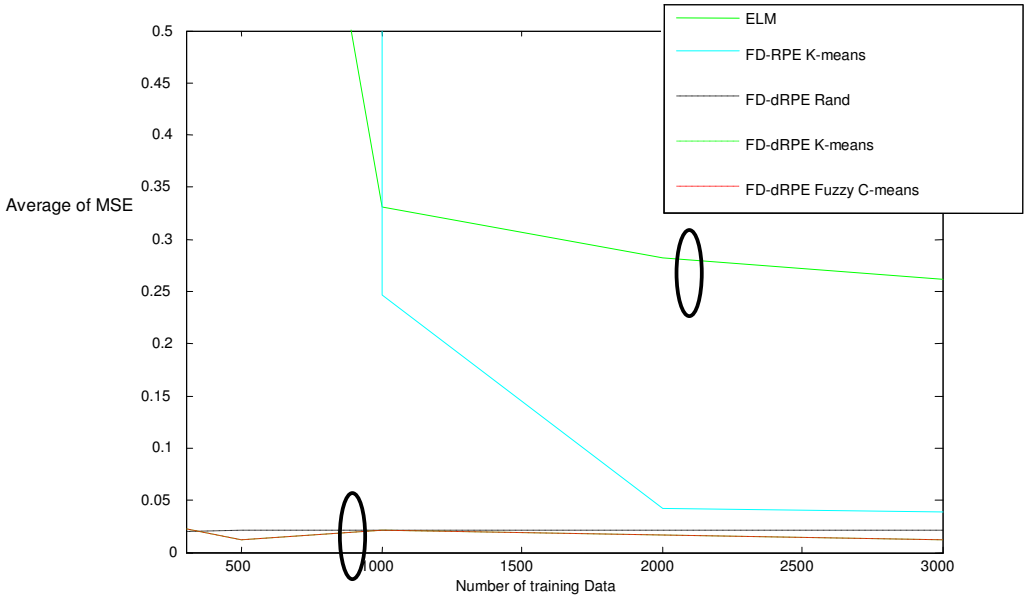


Figure 5.2 Convergence rate for SELS problem with Lost packets

Convergence performance of SELS System on system identification with lost packets can be seen on Figure 5.3. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (less than 1000 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge after 1000 training data and getting stable after 2000 training data. Through this comparison, it can be concluded that by using Finite Difference RPE on decomposed RBF, 1000 training data can be saved. In addition, through this figure, the accuracy performance also can be examined. After getting stable, ELM cannot reach the smallest MSE as small as Finite Difference RPE on decomposed RBF (black, green and red dash line).

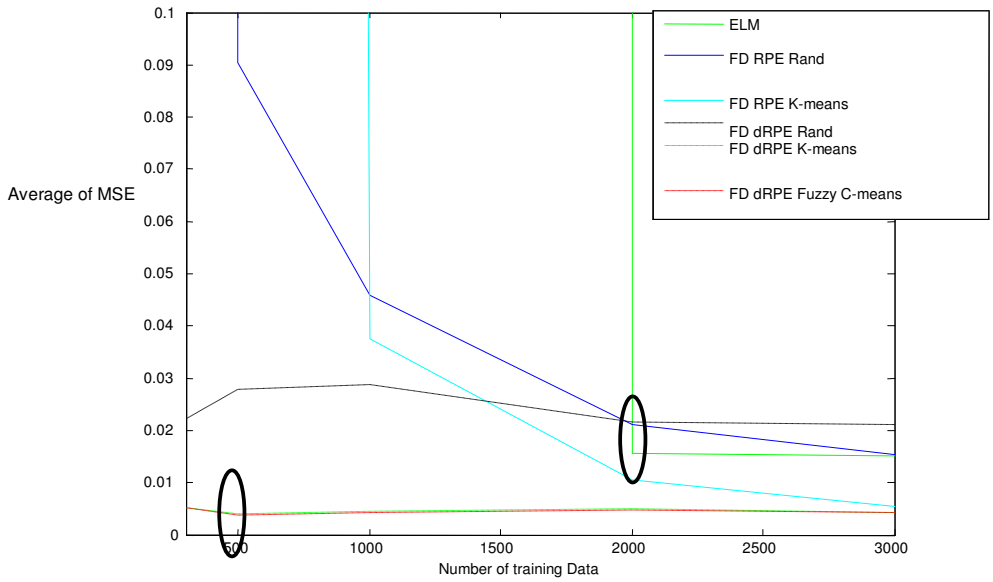


Figure 5.3 Convergence rate for Narendra System problem with Lost packets

Convergence performance of Narendra Dynamic Time Series System for system identification with lost packets can be seen on Figure 5.3. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge and getting stable after 2000 training data. Through this comparison, it can be concluded that by using Finite Difference RPE on decomposed RBF, 1500 training data can be saved. Besides that, through this figure, the accuracy performance also can be examined. After getting stable, ELM cannot reach the smallest MSE as small as Finite Difference RPE on decomposed RBF with smart clustering initialization (green and red dash line).

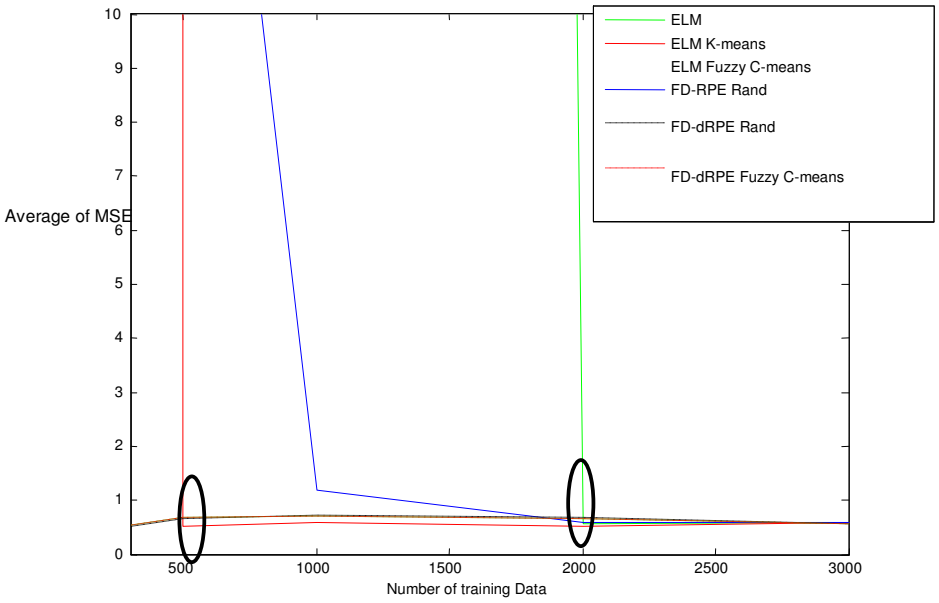


Figure 5.4 Convergence rates for Henon Map with Lost packets

Convergence performance of Henon Map System for system identification with lost packets can be seen on Figure 5.4. Through this figure, the powerful of Finite Difference RPE on decomposed RBF can be observed. The last three dash line (black, green and red) representing Finite Difference RPE on decomposed RBF start to converge and stable since the early training data (500 training data). On the other hand, other methods do not. ELM which is represented by green solid line start to converge and getting stable after 2000 training data. Through this comparison, it can be concluded that by using Finite Difference RPE on decomposed RBF, 1500 training data can be saved.

5.3 Conclusion

Chapter five has discussed simulation results of all proposed training algorithms on system identification application with lost packets. Five benchmarks are used to train the algorithms.

Overall, for system identification with lost packets, the finite difference recursive update is a viable technique to increase the accuracy compared the real gradient approach. Thus, this leads to the proposed finite difference technique for nonlinear weight update when system is irregularly sampled. Simulation results show the finite difference approach improves the convergence rate compared to the real gradient.

The prior setting of weight is still an important issue in online learning especially for RBF networks adopted with lost packets. Simulation results show that again initialization becomes the crucial part on the training process and the unsupervised learning method such as K-means clustering and Fuzzy C-means clustering performs better to random initialization technique for system with lost packets.

Applying the finite difference recursive update on decomposed RBF seems to be promising alternative to identify system with lost packets with much better and faster convergence rate compared to all counter parts.

There is an interesting issue regarding simulation result of system identification with lost packets, all training algorithms perform almost the same on Henon Map and Chen Series Problem. Only finite difference recursive update on decomposed RBF can improve the performance but only in terms of convergence rate.

The next chapter concludes the research writing on overall proposed RBF recursive learning approaches with influence of prior setting using clustering techniques.

CHAPTER 6

CONCLUSION

This chapter summarizes the content, the discussion and mainly the contribution of this research. It is started by looking at motivation on doing this piece of research through the background of the research which is RBF networks itself. Then the thesis chapters are revisited in brief to look at the main discussion and the contributions. Finally, the chapter looks into the future direction which could lead from the research work.

6.1 Revisiting the Motivation of this Research

RBF networks are one of the feed forward neural networks besides the popular MLP. It is widely used among engineer and scientist because of its good capability especially in nonlinear system identification (Surandrajan N et al., 1999; Fung Chi F., 1996; Huang G-B et al., 2004). Its capability can replace many statistical techniques and compared to MLP, it has simpler topology hence can reveal the learning process explicitly.

Learning or training methods for RBF networks is one of the most important parts in any neural networks implementation. Many researches have done on applying RBF networks for various forms of purposes. However, developing an efficient learning method for RBF networks is also an important issue in RBF networks research area which this thesis focuses on. Extreme Learning Machine (ELM) is the latest recent learning methods for RBF networks developed by Huang (Huang, G.B et al., 2004; Liang, N.Y et al., 2006) with utmost simplicity.

It has the advantage on the efficient of running time because it does not update the nonlinear learning parameters (uses random initialization for determining initial nonlinear learning parameters) and only trains its linear weights using linear optimization techniques. This learning method is quite similar to hybrid training techniques (McLoone et al., 1998; Asirvadam et al., 2002). Thus, it just needs less computation cost over time in training compared to many gradients based on the learning methods developed before. However, the drawbacks is the lacking of nonlinear learning parameter update and random initialization which make it is not really robust in terms of accuracy and convergence rate especially for identifying nonlinear system with higher order moreover with lost packets. This leads to the investigation based on finite difference technique which recently shown to have the property of particle filter approach which is based on importance sampling (Šimandl M. and Duníka., 2009).

Another issue is on the centers initialization of the basis function which gives significant effect on the properties of the RBF networks. Due to its locality, the center placement will be crucial in the RBF networks performance. Thus, the thesis looks into replacing the random initialization with some unsupervised learning methods.

Besides the adaptation of RBF with finite difference, another improvement is also being looked into by decomposing the networks to neuron level. Applying finite difference won decomposed RBF networks not only performs well but also show good convergence compared to full update.

Finally the thesis looks into the feasibility of finite difference technique on RBF and decomposed RBF to system with irregular sample time. Recently, some researches (Wang J et al., 2009; Asirvadam and Musab., 2009), look at the system identification with lost packets on linear networks. This thesis looks into nonlinear networks using RBF and finite difference weight update for the case of online lost packets system.

6.2 Revisiting the Thesis Chapters Contents

Chapter 1 starts the thesis by presenting some backgrounds of the research. Some research's objective, methodology and contributions are also presented.

Chapter 2 gives the meaning to RBF and the theoretical framework on its interpolation and generalization.

Chapter 3 introduces the smart clustering techniques and the adaptation to ELM which is basic of a RBF to see the performance gain. The chapter introduces both K-means and Fuzzy C-means for improving prior setting of RBF. Both clustering methods perform better compared to random initialization especially in terms of the final error value. Only RBF initialization with the clustering methods can improve the networks performance but it is not sufficient in improving convergence rate. As shown in some case studies, the clustering based initialization methods perform almost the same or case worse compared to random initialization. The first contribution by presenting the important of learning parameters initialization on fixed size RBF networks prior to learning process has been completed. The smart clustering shows better performance compared to random initialization as ELM.

Chapter 4 looks at finite difference technique as a powerful numerical method to estimate gradient, a derivative estimates, of the underlying dynamics system. The chapter proposes a time varying finite difference, central difference, technique rather using an ad hoc constant step size. Simulation results present that the derivative free estimate performs better than the real gradient of the nonlinear networks which leads to the feasibility in applying it for the case of irregular sample systems.

In this chapter also, the finite difference technique is used on recursive prediction error (RPE) to update the nonlinear parameters part. The finite difference RPE is combined with various types of initialization methods on nonlinear learning parameters to observe the influence of prior setting of RBF weights. Simulations results show the initial setting of RBF weight do influence in final convergence value with smart clustering techniques give the best results in overall when applied to finite

difference technique for nonlinear update. The second contribution is presented by applying finite difference on the nonlinear update.

This Chapter also discusses and proposes decomposed finite difference RPE which is an efficient approach for reducing the computational and storage requirements. By applying the finite difference RPE on decomposed RBF networks, a much better and faster convergence rate is obtained. It can be seen through the whole simulation results in the tables on chapter 4 for decomposed method. Finite difference on decomposed RBF networks starts converge since the first 300 training data while other methods do not. In another words it can be said that by applying finite difference on decomposed RBF networks, the convergence rate is obtained with minimal training data.

Applying Finite Difference technique on updating nonlinear learning parameters combining with decomposed technique on the networks become third contributions which is really worth to decrease the number of training data since it can converge much faster compared to ELM. In addition, the decomposed technique contribute to reduce the memory and computation cost for training RBF networks with larger hidden neurons without decreasing accuracy performance. It becomes forth contribution.

Chapter 5 elaborates all proposed methods for the system identification applied on lost packets. The finite difference presents the best fit compared to the real gradient estimate as the latter only applicable for one-step-ahead prediction. The influence of prior weights setting for RBF and the decomposition of training strategies on the system with irregular time interval are also being investigated in this chapter. Presenting extensive simulation results on fixed RBF learning using the proposed technique in chapter 3 and chapter 4 on system identification with irregular sample data has been completed through the whole discussion on chapter 5 which becomes fifth contribution.

6.3 Future Work and Discussion.

The system studied is based on ELM but the nonlinear weights are updated using finite difference techniques. Future work is looking at constructive networks or rather fixed RBF networks. This leads to growth in networks and the possible application of pruning strategies.

Another option is the finite difference gradient estimate technique can further be refined by adopting Richardson Extrapolation to smoothen the central difference performance.

A thorough study on the time varying step size for finite difference is another option too. Rather than just looking time varying width of Gaussian kernel of RBF, there is possibility of looking at normalized input displacement, center displacement of Gaussian and many other worthy ideas.

REFERENCES

Ahmad, R., & Jamaluddin, H. (2002). Radial Basis Function for Nonlinear Dynamic System Identification. *Journal Teknologi Universiti Teknologi Malaysia, 36(A)*, 39-54.

Asirvadam, Vijanth Sagayan. (2002). On-Line Learning and Construction for Neural Networks. *PhD Thesis School of Electrical and Electronic Engineering, Queen's University Belfast*.

Asirvadam, V S., McLoone S.F., & Irwin G.W. (2002). Minimal update sequential Learning Algorithms for RBF networks. *Digest UKACC Control 2002 Postgraduate Symposium, University of Sheffield, UK*, 71-76.

Asirvadam, Vijanth S., & McLoone, Sean F. (2006). Sequential Learning Methods on RBF Networks with Novel Approach of Minimal Weight Update. *Nonlinear Statistical Signal Processing Workshop IEEE*, 189-192.

Asirvadam, Vijanth S., McLoone, Sean F., & Irwin, George W. (2005). Computationally Efficient Sequential Learning Algorithms for Direct Link Resource-Allocating Networks. *Science Direct, Neurocomputing, 69*, 142-157.

Asirvadam, Vijanth S., McLoone, Sean F., & Irwin, George W. (2004). Fast Sequential Learning Methods on RBF Network Using Decomposed Training Algorithms. *Proceeding of the 2004 IEEE, International Symposium on Intelligent Control, Taipei, Taiwan*, 84-89.

Asirvadam, Vijanth S., & Musab J.O. Elamin. (2009). Wireless System Identification for Linear Networks. *The 5th International Colloquium on Signal Processing and its Applications CSPA, Kuala Lumpur Malaysia*, 118-124.

Barwise, J., Keiler, H.J., Suppes P., & Troelstra, A.S. (1992). Classical Recursion Theory. *ELSEVIER*.

Billings, S.A., Jamaluddin, H.B., and Chen, S. (1992). Properties of Neural Networks with Applications to Modelling Nonlinear Dynamical System. *Int. Journal of Control, 55* (1), 193-224.

Bishop, C.M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press.

Bishop, C M. (1998). Neural Networks and Machine Learning. *NATO ASI Series, Series F: Computer and System Sciences, 168*, Spinger-Verlag Berlin Heidelberg.

Bomberger, J.D., & Seborg, D.E. (1995). On-line Updating of Radial Basis Function Netowrk Model. *IFAC Nonlinear Control Systems Design, Tahoe City, California.*

Brown, Kenneth M., & J.E. Dennis, Jr. (1972). Derivative Free Analogues of The Lavenberg-Marquardt and Gauss Algorithm for Nonlinear Least Squares Approximation. *Journal Numerische Mathematik*, 289-297.Springer-Verlag.

Chen, J.Y., & Qin, Z. (2006). Training RBF Networks with PSO and Improved Subtractive Clustering Algorithms. *International Conference on Neural Information Processing*, 1148-1155. Spinger-Verlag Berlin Heidelberg.

Chen, S., Cowan, C-F.N., Billings, S.A., & Grant, P.M. (1990). Parallel Recursive Prediction Error Algorithm for Training Layered Neural Networks. *International Journal of control*, *51*, 1215-1228.

Chen S., & Billings S.A. (1989). Recursive Prediction Error Parameter Estimator for Nonlinear model. *Int. Journal of control, 49(2)*, 569 – 594.

Chen S., & Billings S.A. (1990). Neural Networks for Nonlinear Dynamic System Modeling and Identification. *Int. Journal of Control, 56(2),* 319-346.

Chen S., Billings S.A., & Grant P.M. (1990). Nonlinear system Identification using Neural Networks. *Int. Journal of control, 51.(6)*, 1191-1214.

Chen S., Billings S.A., & Grant P.M. (1992). Recursive Hybrid Algorithm for Nonlinear System Identification using Radial Basis Function Networks. *Int. Journal Control, 55(5),*1051-1070.

Chung, T.J. (2002). Computational Fluid Dynamics. *Cambridge: Cambridge University Press.*

Cohen, S., & Intrator, N. (1995). *Global Optimization of RBF Networks.* IEEE TNN.

Du, K.L., & M N S. Swammy.(2006). *Neural Networks in a Soft computing Framework.* London: Spinger-Verlag.

Finan, R.A., Sapeluk, A.T., & Dampir, R.I. (1996). Comparison of Multilayer and Radial Basis Function Neural Networks for Text-Dependent Speaker Recognition. *International Conference on Neural Networks (ICNN'96*).

Fung, Chi F., Billings, Steve A., & Luo, W. (1996). On-Line Supervised Adaptive Training Using Radial Basis Function Networks. *Neural Network, 19,* 1597-1617, Elsevier Science.

Gupta, M., Jin, L., & Humma, N. (2003). *Static and Dynamic Neural Networks.* Willey-Interscience.

Howlett, Robert J., & Jain, Lakhmi C. (2001). *Radial Basis Function Networks 1, Recent Development in Theory and Application.* New York: Physica-Verlag Heidelberg.

Huang, G.B., & Siew, C.K. (2004). Extreme Learning Machine with Randomly Assigned RBF Kernels. *International Journal of Information Technology*, *11(1)*, 16-24.

Isaksson, Alf J. (1993). Identification of ARX-Models Subject to Missing Data. *IEEE Transaction on Automatic Control, 35(5),* 813-819.

Junge T.F., & Unbehauen H. (1997). On-line Identification of Nonlinear Time-Variant Systems Using Structurally Adaptive Radial Basis Function Networks. *Proc. American Control Conference, Albuquerque, New Mexico*,1037-1041.

Jun, Y., & Meng J. Er. (2008). An Enhanced On-line Sequential Extreme Learning Machine. *IEEE Control and Decision Conference*, *IEEE*, 2902-2907.

Jun-Li., Zhang, Y-P. (2007). *Modeling of Dynamic Systems Using Generalized RBF Network Based on Kalman Filter Method,*" Spinger-Verlag Berlin Heidelberg, 676-684.

Kim, K.J., & Ahn, H. (2008). A Recomender System Using GA K-means Clustering in a On-line Shopping Market. *Science Direct, Expert System with Application, 34*, 1200-1209.

Lennart Ljung. (1987). *System Identification: Theory for the user.* New Jercey: Prentice Halls.,Englewood Cliffs.

Li M.B., & Er M. J. (2006). Nonlinear System Identification Using Extreme Learning Machine. *The ninth International Conference on Control, Automation, Robotics and Vision 2006, IEEE*, 1-4.

Lightbody, G., & Irwin G.W. (1992). A Parallel Algorithm for Training Neural Network Based Nonlinear Models. *Proc. 2nd IFAC Workshop on Algorithms and Architectures for Real-Time Control*, 99-104.

Li, B., Wang, J., Li, Y., & Sung, Y. (2007). An Improved On-Line Sequential Learning Algorithm for Extreme Learning Machine. *LNCS 4491, Berlin Heidelberg : Springer-Verlag,* 1087-1093 .

Liang, N.Y., Huang, G.B., & Saratchandran, P. (2006). A Fast and Accurate On-line Sequential Learning. *IEEE Transaction on Neural Networks, 17(6)*, 1345-1358,

Marino, M., & Scarpetta, S. (2000). On-Line in RBF Neural Networks a Stochastic Approach. *Elsevier Science Ltd, Neural Networks 13*, 719-729.

McLoone, S.F., & Irwin, G.W. (1997). Fast Parallel Off-Line Training of Multilayer Perceptrons. *IEEE Trans. Neural Networks*, *8*(3), 646-653.

McLoone(a), S.F., & Irwin, G.W. (1998). Nonlinear Optimization of RBF Networks. *International Journal of System Science, 29(2),*179-189.

McLoone(b), S.F., Brown M., Irwin G.W., & Lightbody G. (1998). A Hybrid Linear/Nonlinear Training Algorithm for Feedforward Neural Networks. *IEEE Transaction on Neural Networks, 9(4)*, 669-684.

Musab, J.O. Elamin., Vijanth, S. Asirvadam., & Nordin, Saad. (2007). Systems Modeling with Lost Information Packets Using ARMAX Model. *International Conference in Information and Advanced System, ICIAS*.

Michael, C. Fu. (2005). *Gradient Estimation.* Handbooks in Operation Research and Management Science: Simulation, S.G Henderson & B.L Neson, eds, Elsevier, a pre-print version, chapter 19, 2.

Minh-Tuan, T., Huang, Hieu T., Huynh, Nguyen H. Vo., & Yunggwan won. (2007). A Robust On-Line Sequential Extreme Learning Machine. *ISNN, Part1, LNCS, 4491, Berlin Heidelberg: Spinger-Verlag*, 1077-1086.

Neruda, R., & Kudova, P. (2004). "Learning Methods for Radial Basis Function Networks. *Science Future Generation, Computer System, 21*, Direct, Elsevier, 1131-1142.

Ngia Lester S.H., & Sjöberg J. (2000). Efficient Training of Neural Nets for Nonlinear Adaptive Filtering using a Recursive Levenberg- Marquardt Algorithm. *IEEE Transactions on Signal Processing*, *48(7),* 1915-1926.

Pintelon, R., & Schoukens, J. Frequency Domain System Identification with Missing Data. *IEEE Transaction on Automatic Control, 45(2)*, 364-369.

Paggio T., & Girosi F. (1990). Network for Approximation and Learning. *Proc. IEEE 78(9),* 1481-1497.

Platt J. (1991). A Resource-Allocating Network for Function Interpolation. *Neural Computation, 3,* 213-225.

Powel, M.J.D. (1992). The Theory of Radial Basis Function Approximation in 1990. *Advances in Numerical Analysis, Wavelet, Subdivision Algorithms and Radial Basis Function, Oxford University Press*, *2,* 105-210.

Roy, A., Govil, S., & Miranda, R. (1997). A Neural Network Learning Theory and a Polynomial Time RBF Algorithm. *IEEE Transaction on Neural Networks, 8(6),* 1301-1313.

Schenato, L., Sinopoli B., Franceschetti M., Poolla K., & Sastry S. (2007). Foundations of Control and Estimation over Lossy Networks. *Proceedings of the IEEE on Special issue on emerging technologies of networked control systems, 95(1),*163-187.

Šimandl, M., & Duníka J. (2009). Derivative-free estimation methods: New results and performance analysis. *Automatica, 45, Issue 7*, 1749-1757.

Singh, R., & Balasundaran, S. (2007). Application of Extreme Learning Machine Method for Time Series Analysis. *Proceedings of World Academy of Science, Engineering and Technology, 26*, 81-87.

Sjöberg, J., Hjalmarsson H., & Ljung L. (1994). Neural Network in System Identification. *IFAC System Identification, 1, Denmark,* 359-382.

Sudkamp. Thomas, A. (2006). *An Introduction to Theory of Computer Science, Languages and Machine.* Third edition, Addison Weley.

Surandrajan, N., Saratchandran, P., & Wei, Lu Ying. (1999). Radial Basis Function Neural Networks with Sequential Learning, MRAN and Its Application. *Progress in Neural Processing, 11, World Scientific Publishing Co. Pte. Ltd.*

Wallin, R., & Isaksson, Alf J. (2002). Multiple Optima in Identification of ARX Models Subject to Missing Data. *EURASIP Journal on Applied Signal Processing,* 30-37.

Wang, J., Zheng, W.X., & Chen T. (2009). Identification of Linear Dynamic Systems Operating in a Networked Environment. *Automatica, 45, Issue 12*, 2763-2772.

Yingwei, L., Sundararajan N., & Saratchandran P. (1997). Identification of Time-Varying Nonlinear Systems Using Minimal Radial Basis Function Neural Networks. *IEEE Proceedings Control Theory Application,144 (2).*

Yingwei, L., Sundararajan, N., & Saratchandran, P. (2000). Analysis of Minimal Radial Function Basis Function Network Algorithm for Real-Time Identification of Nonlinear Dynamic Systems. *IEEE Proceedings Control Theory Application, 147(4).*

Young Sin, Mi. (1998). *Design and Evaluation of Radial Basis Function Model for Approximation.* A dissertation for PhD in Computer and Informatics Science in Graduate School of Syracuse University.

Yuan, X.B., Wang, Z., Yu, S.H., & Li, Y-J. (2005). Time Series Forecasting with RBF Neural Network. *Proceeding of the Fourth International Conference on Machine Learning and Cybernetic, Guangzhou, 18-21 August,* 4680-4638.

Zhu, Qin-Yu., Qin, A.K., Suganthan, P.N., & Huang, G.B. (2005). Evolutionary Extreme Learning Machine. *Science Direct, Pattern Recognition, 38,*1759-1763.

APPENDIX A:

BENCHMARKS TEST PROBLEMS

This section provides a brief description of the various benchmarks modeling problem used in the evaluation of the training algorithms in this thesis. In all cases training performance summary is measured by average of mean square error for 25 simulations per each algorithm on each benchmarks and test case.

$$MSE = J(\Theta_k) = \frac{1}{N_v}\sum_{n=1}^{N_v}\left(f(x_n - \Theta_k) - y_n\right)^2$$

$$Average\ \ of\ \ MSE = \frac{\frac{1}{N_v}\sum_{n=1}^{N_v}\left(f(x_n - \Theta_k) - y_n\right)^2}{25}$$

Computed over a representative training data set (x_n, y_n). N_v is of the training data set.

Test Problem I: Slow Excitation Linear System (SELS)

SELS involves the estimation of first order discrete-time system from correlated input-output data. The system is defined by the equation

$$y_t = 0.8y_{t-1} + 0.2u_{t-1}$$

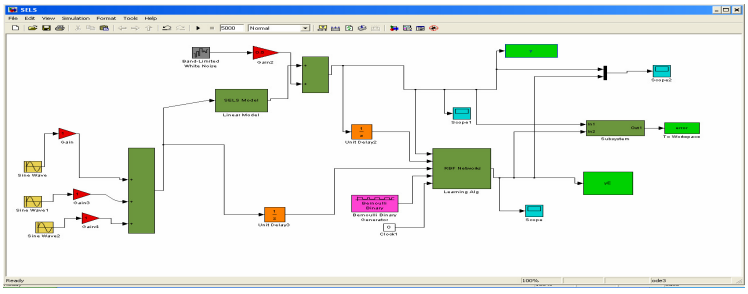


Figure A.1 Simulink Model for SELS simulastion

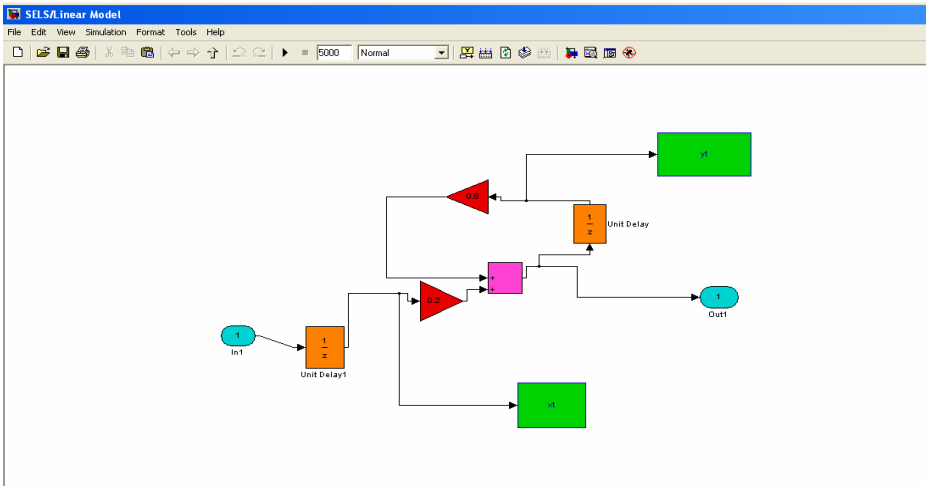


Figure A.2 Simulink model for SELS system

The input data is generated by summation of three sinusoidal functions with various amplitude and frequency

A data set contains 5000 data. Three thousands data is used as training data and 2000 others data is used as testing data.

Test Problem II. Mackey Glass Chaotic Time Series Problem

Benchmark problem II involves the identification of nonlinear auto regressive (NAR) model of chaotic Mackey-Glass time series defined by the differential delay equation below

$$\frac{dy_t}{dt} = -by_t + \frac{ay_{t-\tau}}{1+y_{t-\tau}^{10}}$$

With a=0.2, b=0.1, τ=17 and the sample rate is one second. The NAR Model is implemented with network input vector given by,

$$y_t = [y_{t-1}\ y_{t-6}\ y_{t-12}\ y_{t-18}]$$

A data set contains 5000 data. Three thousands data is used as training data and 2000 others data is used as testing data.

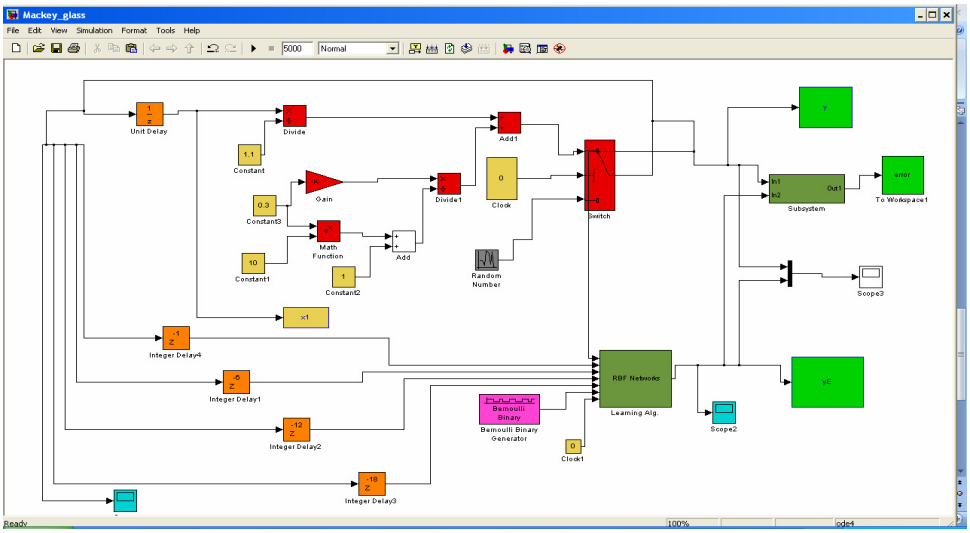


Figure A.3 Simulink Model for Mackey Glass Chaotic simulation

Test Problem III. Narendra's Dynamics System (NDYS)

This benchmarks problem involves the identification of nonlinear dynamics system suggested by Narendra and Parthasarathy (1990). This governed by the equation

$$y_t = 0.3y_{t-1} + 0.6\sin(\pi u_k) + 0.3\sin(3\pi u_k) + 0.1\sin(5\pi u_{k-1})$$

The identification process is run over 5000 training data (iteration) with the plant input defined as

$$u_t = \sin\left(\frac{2\pi t}{250}\right) + \sin\left(\frac{2\pi t}{200}\right)$$

The input vector for the RBF Networks model for this test problem is given by

$$x_t = [y_{t-1}, y_{t-2}, u_{t-1}]$$

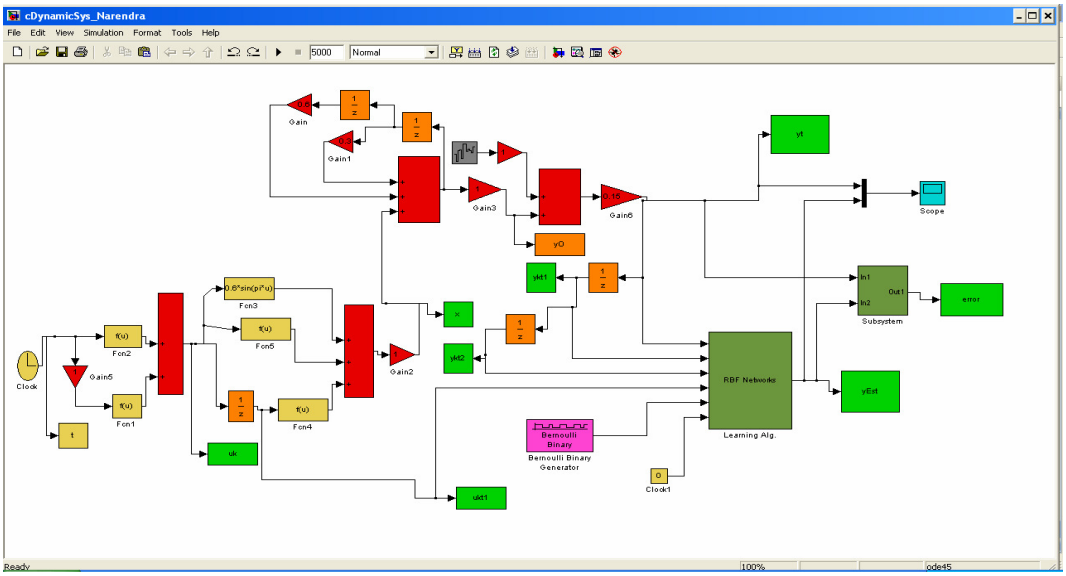


Figure A.4 Simulink model for NDYS problem simulation

Test Problem IV. Chen's Nonlinear System (CNLS)

A simulated nonlinear time series (Chen et al.,*1 992)* was employed to test the capabilities of the learning algorithm. The time series is described by the following nonlinear difference equation

$$y_t = \left[0.8 - 0.5\exp\left(-y_{t-1}^2\right)\right]y_{t-1} + \left[0.3 + 0.9\exp\left(-y_{t-1}^2\right)\right]y_{t-2}^2 + 0.1\sin\left(\pi y_{t-1}\right) + \eta_t$$

where the noise η_t is a Gaussian white noise sequence with zero mean and variance 0.01. The RBF model is based on the nonlinear auto regressive (NAR) model structure, that is,

$$y_t = f\left(y_{t-1}, y_{t-2}\right) + \eta_t$$

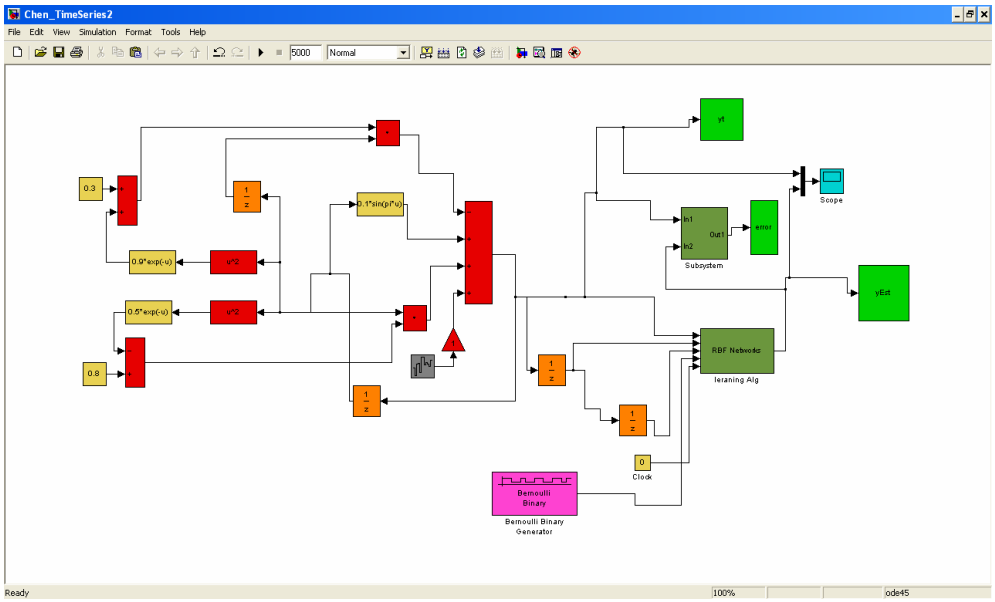


Figure A.5 Simulink model for Chen Series problem simulation

Test Problem V. Henon Map Problem

The fifth test problem will be on Henon Map Problem. It is a discrete time dynamical system proposed by French astronomer, Michael Henon. Henon Map is described as a nonlinear dynamics system below

$$x_{t+1} = 1 + ax_t^2 + by_t$$

$$y_{t+1} = x_t$$

With a=*1.4*, b=0.3.

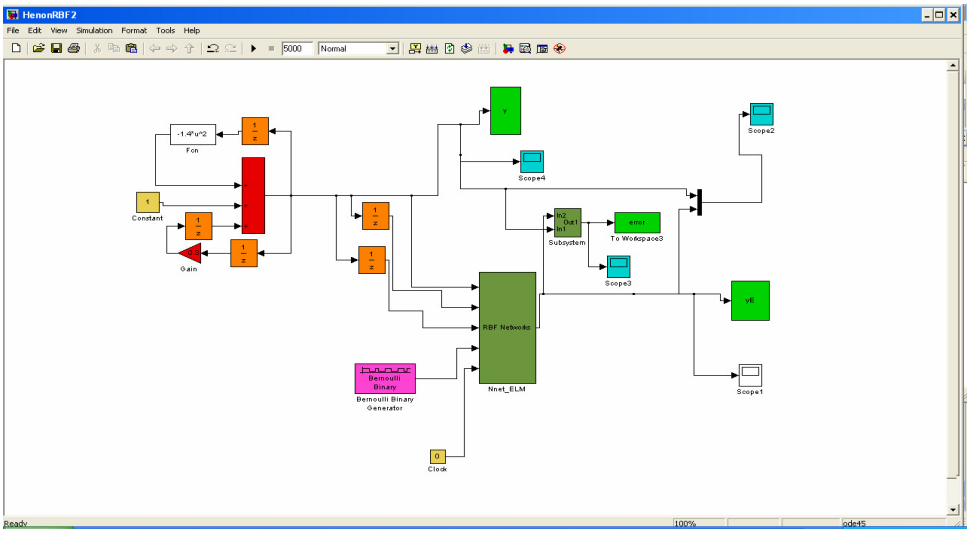


Figure A.6 Simulink model for Henon Map problem simulation

APPENDIX B

RBF FUNCTION APPROXIMATION

RBF networks with Gaussian kernels satisfy *Stone-Weierstrass Theorem* which stages necessary condition for function approximation (Gupta M et al., 2003; Young, Sin Mi, 1998). Some mathematical analysis regarding the function approximation capability of RBF network with Gaussian can be summarized below.

Definition B.1

Let a set Ω,

- ***a.*** ***int (*** Ω ***)*** *is collection all interior point of* Ω
- ***b.*** ***ext (*** Ω ***)*** *is collection all exterior point of* Ω
- ***c.*** Ω' *is collection all limit points of* Ω, *it is said as derived set.*

Definition B.2. Closure Definition

If Ω *is a set of elements, then by the closure* $[\Omega]$ *of* Ω, *it is meant that the set of all points in* Ω *together with the s*et al*l limit points of* Ω, $[\Omega] = \Omega \cup \Omega'$

Definition B.3. Open Set

Ω *is open set if every elements of* Ω *is an interior point*

Definition B.4. Closed Set

Ω *is closed set if* Ω^c *is open set*

Definition B.5. Dense set Definition

Let V *be a subset of the set* Ω, V *is dense in* Ω *if* $[V] = \Omega$

Further discussion in approximation theory, if V *is* dense in Ω, then each element of Ω can be approximated arbitrary well by elements of V. This denseness will play a key role in later discussion on the approximation capabilities of RBF networks with Gaussian.

Definition B.6. Compact set

A set is said to be compact if every infinite subset of the set contains at least one limit point.

By the above definitions, the Stone Weierstrass Theorem is discussed below to show the function approximation capability of RBF networks with Gaussian.

Theorem B.1. Stone Weierstrass II

Let S be a compact set with n dimension, and $\Omega \supset C[S]$ *be a set of continuous real valued functions on S satisfying the conditions:*

i. *Identity function: The constant function* $f(x) = 1$ *is in* Ω*;*

ii. *Separability : for any two points* $x_1, x_2 \in S$ *and* $x_1 \neq x_2$*, there exists a* $f \in \Omega$ *such that* $f(x_1) \neq f(x_2)$*;*

iii. *Algebraic Closure: for any* $f, g \in \Omega$ *and* $\alpha, \beta \in \Re$*, the function fg and* $(\alpha f + \beta g)$ *are in* Ω

Then Ω *is dense in* $C[S]$*. In other words, for any* $\varepsilon > 0$ *and any function* $f \in \Omega$ *such that* $|g(x) - f(x)| < \varepsilon$ *for all* $x \in S$*.*

Conditions, *(i) to (iii)* are used to ensure the function approximation capability of RBF networks with Gaussian. An exponential function such as Gaussian function can process the multiplication into addition as follow:

$$e^{x} \bullet e^{y} = e^{x+y} \tag{B.1}$$

Hence, it can be verified that RBF networks with Gaussian satisfies the whole condition of Weierstrass Theorem which mean the function approximation capability is fulfilled.

Theorem B.2

Let Ω *be the set of all function that can be computed by RBF networks with Gaussian on a compact set* $S \supset \Re^n$ *:*

$$\Omega_n = \left\{ f(x) = \sum_{i=1}^{N} \beta_i \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}}{\sigma_{ik}} \right]^2 \right) : \beta_i, c_{ik}, \sigma_{ik} \in \Re, x \in S \right\} \tag{B.2}$$

$$\Omega = \bigcup_{N=1}^{\infty} \Omega_N \tag{B.3}$$

Then Ω *is dense in* $C[S]$

Proof:

Trough this theorem, it will be proven that RBF networks with Gaussian satisfies the Weierstrass Theorem with its three necessary conditions hence it can be proven the function approximation capability of RBF networks with Gaussian.

i. Identity function

   $f(x) = 1$ belongs to Ω, since it can be considered as a Gaussian function with infinite variance σ.

ii. Separability

   $\forall x, y \in S$, $x \neq y$, it can be obviously verified that $f(x) \neq f(y)$ because the exponential function is strictly monotic.

iii. Algebraic closure

Let $f, g \in \Omega$ which are represented by the Gaussian function as

$$f(x_1,...,x_n) = \sum_{i=1}^{N_f} \beta_i^f \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^f}{\sigma_{ik}^f} \right]^2 \right) \tag{B.4}$$

$$g(x_1,...,x_n) = \sum_{i=1}^{N_g} \beta_i^g \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^g}{\sigma_{ik}^g} \right]^2 \right) \tag{B.5}$$

Since

$$\begin{aligned} fg &= \sum_{i=1}^{N_f} \sum_{j=1}^{N_g} \beta_i^f \beta_j^g \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^f}{\sigma_{ik}^f} \right]^2 \right) \times \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^g}{\sigma_{ik}^g} \right]^2 \right) \\ &= \sum_{i=1}^{N_f} \sum_{j=1}^{N_g} \beta_i^f \beta_j^g \exp\left(-\frac{1}{2} \left\{ \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^f}{\sigma_{ik}^f} \right]^2 + \sum_{k=1}^{n} \left[\frac{x_k - c_{ik}^g}{\sigma_{ik}^g} \right]^2 \right\} \right) \\ &= \sum_{i=1}^{N_f} \sum_{j=1}^{N_g} \beta_{ij} \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{x_k - c_{ijk}^f}{\sigma_{ijk}^f} \right]^2 \right) \end{aligned} \tag{B.6}$$

where

$$\beta_{ij} = \beta_i^f \beta_j^g \exp\left(-\frac{1}{2} \sum_{k=1}^{n} \left[\frac{a_{ijk} - c_{ijk}^2}{\sigma_{ijk}^2} \right] \right) \tag{B.7}$$

$$c_{ijk} = \frac{\left(\sigma_{ik}^f\right)^2 \left[c_{ik}^f + c_{jk}^g \right]}{\left(\sigma_{ik}^f\right)^2 + \left(\sigma_{jk}^g\right)^2} \tag{B.8}$$

$$\sigma_{ijk} = \frac{\sigma_{ik}^{f}\sigma_{jk}^{g}}{\left[\left(\sigma_{ik}^{f}\right)+\left(\sigma_{jk}^{g}\right)\right]^{\frac{1}{2}}} \tag{B.9}$$

$$a_{ijk} = \frac{\left(c_{ik}^{f}\right)^{2}\left(\sigma_{jk}^{f}\right)^{2}+\left(c_{ik}^{g}\right)^{2}\left(\sigma_{jk}^{g}\right)^{2}}{\left(\sigma_{jk}^{f}\right)^{2}+\left(\sigma_{jk}^{g}\right)^{2}} \tag{B.10}$$

Hence, the product fg is in Ω so that the RBF networks with Gaussian satisfy all properties of Stone-Weierstrass theorem. Thus Ω is dense in $C[S]$.

APPENDIX C

PROOF OF FINITE DIFFERENCE CONVERGENCE

As stated on Chapter 2, learning is an optimization process which traces the learning parameters that will minimize the error function. Thus, some mathematics analysis will be presented to guarantee that FD RPE will converge to the learning parameters which optimize the objective function.

Definition C.1. *Convex Set*

A set D_0 *is said to be convex if the line segment between any two points in* D_0 *also lies in* D_0 ,i.e, *if for any* $x, y \in D_0$ *and any* θ *with* $0 \le \theta \le 1$, $\theta x + (1-\theta) y \in D_0$

Let

$$F(\Theta^*) = (\hat{y}(c_i, \sigma_i, \beta) - y)^2 \tag{C.1}$$

$$\Theta^* = [c^*; \sigma^*] \tag{C.2}$$

$$J(\Theta) = \frac{F(\Theta + h) - F(\Theta - h)}{2h} = \frac{\Delta F}{2h} \tag{C.3}$$

Lemma C.1. *Let* D_0 be an open convex set and $K \ge 0$. If $\forall x, y \in D_0$, $\left\| J(x) - J(y) \right\|_2 \le K \left\| x - y \right\|_2$, then the following inequalities hold for every $x, y \in D_0$,

i. $\left\|J(x)^T - J(y)^T\right\|_2 \le K\left\|x - y\right\|_2$ (C. 4)

ii. there exist non negative constant C and C' such that

$$\left\|J(x) - J(y)\right\|_1 \le C.K\left\|x - y\right\|_1 \quad \text{(C.5)}$$

and

$$\left\|A\right\|_2 \le C'\left\|A\right\|_2 \quad \text{(C.6)}$$

iii. For any real rectangular matrix *A*

$$\left\|F(x) - F(y) - J(y)(x, y)\right\|_1 \le C.K / 2\left\|x - y\right\|_1^2 \quad \text{(C.7)}$$

Theorem C.1

Let

$$J(\Theta^*)^T F(\Theta^*) = 0 \quad \text{(C.8)}$$

for some $\Theta^* \in D_0$, and let

$$\left\|J(\Theta)^T - J(y)^T\right\|_2 \le K\left\|\Theta - y\right\|_2, \forall \Theta, y \in D_0 \quad \text{(C.9)}$$

If $K\left\|F(\Theta^*)\right\|_2$ is a strict lower bound for the spectrum of $J\left(\Theta^*\right)^T J\left(\Theta^*\right)$ then $\exists \eta > 0 \ni \forall |h| < 0 \;\&\; h \to 0$, FD RPE method converges locally to Θ^*

Proof

For each $\Theta \in D_0$, let $\lambda(\Theta, h)$ denote the least Eigen value of

$$\frac{\Delta F(\Theta, h)^T \Delta F(\Theta, h)}{2h} \quad \text{(C.10)}$$

For $|h|$ sufficiently small and Θ in some smaller neighborhood (than D_0) of Θ^*, λ is non negative jointly continuous function of $\Theta \;\&\; h$ since by hypothesis that

$$\lambda(\Theta^*,0) > K\left\|F(\Theta^*)\right\|_2 \geq 0 \tag{C.11}$$

It can be found $\eta' > 0$ and D_1, an open convex neighborhood of Θ^* such that $\overline{D}_1 \subset D_0$ and for $\Theta \in \overline{D}_1$, $|h| \leq \eta'$,

$$\lambda(\Theta,h) > \lambda' = K\left\|F(\Theta^*)\right\|_2 + \left(\lambda(\Theta^*,0) - K\left\|F(\Theta^*)\right\|_2\right)/2 \tag{C.12}$$

Let *B* denote the uniform bound on $\left\|\frac{\Delta F(\Theta,h)^T}{2h}\right\|_2$ for $\Theta \in \overline{D}_1$, $h \in [-\eta',\eta]$

(*B* exists by continuity). Now set

$$e_t = \left\|\Theta_t - \Theta^*\right\|_2 \tag{C.13}$$

and

$$L_n \equiv \frac{\Delta F(\Theta,h)^T \Delta F(\Theta,h)}{4h^2} \tag{C.14}$$

And notice that the smallest Eigen value of L_n is $(\lambda')^{-1} \geq (\lambda') \geq 0$, hence L_n^{-1} exist and its l_2 norm is bounded by $(\lambda')^{-1}$.

For some $\gamma \in (0,1)$ we can write

$$\begin{aligned}
e_t &\equiv \left\|\Theta_{t-1} - \Theta^*\right\|_2 \\
&\leq (\lambda')^{-1}\left\{\left\|L_t(\Theta) - \frac{\Delta F(\Theta_t,h_t)^T}{2h} J\left(\Theta_t + \gamma(\Theta^* - \Theta_t)\right)\right\|_2 \left\|\Theta_t - \Theta^*\right\|_2 + \left\|J(\Theta^*) - \frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2 \left\|F(\Theta^*)\right\|_2\right\} \\
&\leq (\lambda')^{-1}\left\{\left[B\left\|\frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2 + B\left\|J(\Theta_t) - J\left(\Theta_t + \gamma(\Theta^* - \Theta)\right)\right\|_2\right].e + \left[\left\|J(\Theta^*) - J(\Theta_t)\right\|_2 + \left\|J(\Theta_t) - \frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2\right]\left\|F(\Theta^*)\right\|\right\}
\end{aligned} \tag{C.15}$$

it is shown in (*Dennis J.E.,Jr, 1970*) that

$$\left\|\frac{\Delta F(\Theta_t,h_t)}{2h} - J(\Theta_t)\right\|_1 \leq CK\frac{|h|}{2} \tag{C.16}$$

Hence by *lemma 1*

$$e_t \le (\lambda')^{-1}\left(BKe_t + BC'CK\frac{|h|}{2} + K\left\|F\left(\Theta^*\right)\right\|_2 \right)e + (\lambda')^{-1}C'CK\left\|F\left(\Theta^*\right)\right\|_2\frac{|h|}{2} \tag{C.17}$$

Now choose $r > 0$ and $\eta'' > 0$ so small that $N\left(\Theta^*, r\right) \subset D_1, \eta'' < \eta'$ and

$$\lambda\left(\Theta^*, 0\right) - K\left\|F\left(\Theta^*\right)\right\|_2 > 2BK_r + BCC'K\eta'' \tag{C.18}$$

This insures that $\gamma < 1$ where

$$\gamma \equiv \sup_t (\lambda')^{-1}\left(BK_r + BC'CK\frac{\eta''}{2} + K\left\|F\left(\Theta^*\right)\right\|_2 \right) \tag{C.19}$$

Finally select $\eta' < \eta''$ such that

$$\eta \le \frac{2(\lambda' - BKr)}{BC'CK}, \text{ for } \left\|F\left(\Theta^*\right)\right\|_2 = 0 \tag{C.20}$$

Now let us assume that for $t \ge 0$, we have $e \le r$ and $|h| \le \gamma$. From (1) we obtain

$$e_{t+1} \le \gamma e_t + (1-\gamma)r \le r \tag{C.21}$$

This shows that FD RPE iteration is locally well defined. It can be shown that as long as $h \to 0$, the method is locally converge to Θ^*

Teorema C.2

Let F satisfies the differentiability assumption of theorem 4.1 in a neighborhood D_0 of Θ^* of F.

If $|h|$ is $O\left(\left\|F\left(\Theta_t\right)\right\| any\right)$ then FD RPE is quadratically convergent.

Proof

For each $\Theta \in D_0$, let $\lambda(\Theta, h)$ denote the least Eigen value of

$$\frac{\Delta F(\Theta,h)^T \Delta F(\Theta,h)}{2h} \tag{C.22}$$

For $|h|$ sufficiently small and Θ in some smaller neighborhood (than D_0) of Θ^*, λ is non negative jointly continuous function of Θ & h since by hypothesis that

$$\lambda(\Theta^*,0) > K\left\|F(\Theta^*)\right\|_2 \geq 0 \tag{C.23}$$

It can be found $\eta' > 0$ and D_1, an open convex neighborhood of Θ^* such that $\overline{D}_1 \subset D_0$ and for $\Theta \in \overline{D}_1$, $|h| \leq \eta'$,

$$\lambda(\Theta,h) > \lambda' = K\left\|F(\Theta^*)\right\|_2 + \left(\lambda(\Theta^*,0) - K\left\|F(\Theta^*)\right\|_2\right)/2 \tag{C.24}$$

Let *B* denote the uniform bound on $\left\|\frac{\Delta F(\Theta,h)^T}{2h}\right\|_2$ for $\Theta \in \overline{D}_1$, $h \in [-\eta',\eta]$

(*B* exists by continuity). Now set

$$e_t = \left\|\Theta_t - \Theta^*\right\|_2, \tag{C.25}$$

$$L_n \equiv \frac{\Delta F(\Theta,h)^T \Delta F(\Theta,h)}{4h^2}, \tag{C.26}$$

And notice that the smallest eigen value of L_n is $(\lambda')^{-1} \geq (\lambda') \geq 0$, hence L_n^{-1} exist and its l_2 norm is bounded by $(\lambda')^{-1}$.

For some $\gamma \in (0,1)$ we can write

$$\begin{aligned}
e_t &\equiv \left\|\Theta_{t-1} - \Theta^*\right\|_2 \\
&\leq (\lambda')^{-1}\left\{\left\|L_t(\Theta) - \frac{\Delta F(\Theta_t,h_t)^T}{2h} J\left(\Theta_t + \gamma(\Theta^* - \Theta_t)\right)\right\|_2 \left\|\Theta_t - \Theta^*\right\|_2 + \left\|J(\Theta^*) - \frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2 \left\|F(\Theta^*)\right\|_2\right\} \\
&\leq (\lambda')^{-1}\left\{\left[B\left\|\frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2 + B\left\|J(\Theta_t) - J\left(\Theta_t + \gamma(\Theta^* - \Theta)\right)\right\|_2\right].e + \left[\left\|J(\Theta^*) - J(\Theta_t)\right\|_2 + \left\|J(\Theta_t) - \frac{\Delta F(\Theta_t,h_t)}{2h}\right\|_2\right]\left\|F(\Theta\right.\right.
\end{aligned}$$

it is shown in (Dennis J.E.,Jr, 1970) that

(C.27)

$$\left\| \frac{\Delta F(\Theta_t, h_t)}{2h} - J(\Theta_t) \right\|_1 \leq CK \frac{|h|}{2}, \tag{C.28}$$

From lemma 4.1 we get

$$e_t \leq (\lambda')^{-1} \left(BKe_t + BC'CK \frac{|h|}{2} \right) e, \tag{C.29}$$

To see that the convergence is quadratic, note that since there is a uniform upper bound on

$$\left\| J(\Theta \ ) \right\|_2 \text{ for } \left\| \Theta - \Theta^* \right\|_2 \leq r, \tag{C.30}$$

$$\begin{aligned} \left\| F(\Theta_t) \right\|_2 &= \left\| F(\Theta_t) - F(\Theta^*) \right\|_2 \\ &\leq \sup_{\Theta} \left\| J(\Theta) \right\|_2 \left\| \Theta_t - \Theta^* \right\|_2, \end{aligned} \tag{C.31}$$

Where $\Theta \in [\Theta_t, \Theta^*] \subset \overline{N}(\Theta^*, r)$, hence using the above and the order equivalence of norms, $|h|$

$$e_t \leq (\lambda')^{-1} \left(BKe_t + BC'CK \frac{|h|}{2} \right) e, \ e_{t+1} = O(e_t^2). \tag{C.32}$$