Performance Evaluation of Adaptive Polynomial Filtering Algorithms for Time-Varying Parameter Estimation

Thesis submitted in partial fulfillment of the requirement for the award of the degree of

MASTER OF ENGINEERING

In

ELECTRONICS & COMMUNICATION ENGINEERING

Submitted by

Ankur Gupta

Roll no. 800961002

Under the Guidance of

Dr. Amit Kumar Kohli

Assistant Professor

Electronics and Communication Engineering Department

Thapar University, Patiala-147004 (PUNJAB)

June 2011

CERTIFICATE

I, **Ankur Gupta**, hereby certify that the work which is being presented in this thesis entitled **"Performance evaluation of adaptive polynomial filtering algorithms for time-varying parameter estimation"** by me in partial fulfillment of the requirements of the award of the degree of Masters of Engineering in Electronics and Communication Engineering from Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Amit Kumar Kohli**.

The matter presented in this thesis has not been submitted in any other University/Institute for the award of the degree of Masters of Engineering.

Ankur Gupta

Ankur Gupta

Date...24/6/2011.....

This is certified that the above statement made by the candidate is correct to the best of my knowledge.

Amit Kohli

Dr. Amit Kumar Kohli

Assistant Professor, ECED

Thapar University, Patiala

Date...24/06/2011...

Countersigned by:

Professor and Head, ECED,

Thapar University, Patiala

Date...24/6/11.....

Dr. S. K. Mohapatra

Dean of Academic Affairs

Thapar University, Patiala

Date.....

ACKNOWLEDGEMENT

No volume of words is enough to express my gratitude towards my guide, **Dr. Amit Kumar Kohli**, Assistant. Professor, Electronics and Communication Engineering Department, Thapar University, who has been very concerned and has aided for all the material essential for the preparation of this thesis report. He has helped me to explore this vast topic in an organized manner and provided me with all the ideas on how to work towards a research-oriented venture.

I am also thankful to **Dr. A. K. Chatterjee**, Head of Department, ECED and **Ms. Alpana Aggarwal**, P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there in the need of the hour and provided with all the help and facilities, which I required, for the completion of my thesis.

Most importantly, I would like to thank my parents and the Almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

Ankur Gupta

Roll No. – 800961002

ABSTRACT

The current trend in the telecommunication systems design is the identification and compensation of unwanted nonlinearities. It is known that unwanted nonlinearities in the system will have a determinant effect on its performance. The use of nonlinear models considered in this thesis is to characterize and compensate harmful nonlinearities offer a possible solution. There are many applications in communication system where time varying nature of Volterra system is required therefore Gauss Morkol model is used to represent time varying systems. The time varying Volterra system has been widely applied as nonlinear system modeling technique with considerable success. When the nonlinear system is unknown, adaptive methods and algorithms are widely used for the Volterra kernel estimation. The accuracy of the Volterra kernels will determine the accuracy of the system model and the accuracy of the inverse system used for compensation. Parameter estimation of Volterra systems is a very important part of the adaptive algorithm when it comes to controlling noisy systems. This thesis proposes some adaptive algorithms, which is used to track and estimate the time varying nonlinear systems. Parameter estimation is used in tracking of objects like face, missiles, hand, head etc.

Firstly we proposes Kalman filter which is used to recursively estimate and track the time variation of the first and second order Volterra kernels. It produces estimates of the true values of measurements and their associated calculated values by predicting a value, estimating the uncertainty of the predicted value, and computing a weighted average of the predicted value and the measured value. The estimates produced by the method tend to be closer to the true values than the original measurements because the weighted average has a better estimated uncertainty than either of the values that went into the weighted average. Then there is another adaptive algorithm we propose which belongs Kalman filter family that is Recursive least squares (RLS) algorithm which recursively finds the filter coefficients that minimize a weighted linear least squares cost function relating to the input signals. The tracking performance and convergence rate of RLS algorithm depends upon forgetting factor. RLS algorithm having fixed forgetting factor has to make some adjustment for previous performance. If the forgetting factor which is closed to unity RLS algorithm gives good stability but at the

cost of low misadjustment and worse tracking performance. If forgetting factor is very low then it gives good tracking performance but with bad stability.

So there is a need of an adaptive algorithm which is used to estimate and track the time varying Volterra system having variable forgetting factor. Then we proposes Dynamic forgetting factor recursive least square (DFFRLS) adaptive algorithm for first and second order Volterra system. The DFFRLS is adapted to a time varying signal by an extended prediction error criterion which accounts for the nonstationarity of the signal. This method has good adaptability in the nonstationary situation and low variance in the stationary situation.

Keywords: **Gauss Morkov model, DFFRLS, The Kalman filter, RLS algorithm, Volterra kernels**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AF	adaptive filter
AR1	autoregressive model of first order
AWGN	additive white Gaussian noise
DFFRLS	dynamic forgetting factor recursive least square
EWRLS	exponentially windowed recursive least square
FFF	fixed forgetting factor
FRLS	fast recursive least square
LMS	least mean square
LS	least square
MSE	mean square
MMSE	minimum mean square error
NVFF	new variable forgetting factor
RLS	recursive least square
TV	time varying
VS	Volterra series
VFF	variable forgetting factor
VFFRLS	variable forgetting factor recursive least square

CHAPTER 1

INTRODUCTION

1.1 Adaptive Filters

The development of nonlinear adaptive filtering is motivated by increasing requirement in a wide range of signal processing applications include echo cancellation, adaptive equalization, adaptive noise cancellation, and adaptive beamforming. These applications involve processing of signals that are generated by systems whose characteristics are not known a priori. Under this condition, a significant improvement in performance can be achieved by using adaptive rather than fixed filters [1]. However, in many applications such as communication channels modelling, speech processing, and biological systems, time variation of the system characteristics is required and that system has time-varying parameters, this parameter can be linear or nonlinear. There are certain nonlinear models which is used to represent nonlinear systems, among them Volterra model, which states that any continuous real-valued function can be approximated by a polynomial function within an arbitrary small error. We adapt the filtering technique to track the dynamics of the Volterra system or learn the unknown parameters. Nonlinear effect has been reduced by certain algorithm. Parameter estimation using adaptive algorithm plays an important role in day to day signal processing applications. When operating in a time varying environment, the adaptive filter has the additional task of tracking the statistical variations in environmental conditions [2],[3]. It is well recognized that the convergence behaviour of an adaptive filter is a transient phenomenon, whereas its tracking behaviour is a steady-state phenomenon. This means that, in general, the good convergence behaviour does not necessarily translate into a good tracking behaviour.

An adaptive filter is a self-designing filter that uses a recursive algorithm (known as adaptation algorithm or adaptive filtering algorithm) to “design itself.” The algorithm starts from an initial guess, chosen based on the a priori knowledge available to the system, then refines the guess in successive iterations, and converges, eventually, to the optimal solution in some statistical sense. It is a computational device that attempts to model the relationship between two signals in real time in an iterative manner [4]. Adaptive filters are often realized either as a set of program instructions running on an arithmetical processing device such as a microprocessor or DSP chip, or as a set of logic

operations implemented in a field-programmable gate array (FPGA) or in a semicustom or custom VLSI integrated circuit. However, ignoring any errors introduced by numerical precision effects in these implementations, the fundamental operation of an adaptive filter can be characterized independently of the specific physical realization that it takes. For this reason, we shall focus on the mathematical forms of adaptive filters as opposed to their specific realizations in software or hardware.

An adaptive filter is defined by four aspects:

1. The signals being processed by the filter
2. The structure that defines how the output signal of the filter is computed from its input signal.
3. The parameters within this structure that can be iteratively changed to alter the filter's input-output relationship.
4. The adaptive algorithm that describes how the parameters are adjusted from the time instant to the next

By choosing a particular adaptive filter structure, one specifies the number and type of parameters that can be adjusted. The adaptive algorithm used to update the parameter values of the system can take on a myriad of forms and is often derived as a form of optimization procedure that minimizes an error criterion that is useful for the task at hand. In this section, we present the general adaptive filtering problem and introduce the mathematical notation for representing the form and operation of the adaptive filter.

Adaptive Filtering Problem

Figure 1.1 shows a block diagram in which a sample from a digital input signal $x(n)$ is fed into a device, called an adaptive filter, that computes a corresponding output signal sample $y(n)$ at time n. For the moment, the structure of the adaptive filter is not important, except for the fact that it contains adjustable parameters whose values affect how $y(n)$ is computed. The output signal is compared to a second signal $d(n)$ called the desired response signal, by subtracting the two samples at time n. This difference signal, given by

$$e(n) = d(n) - y(n) \tag{1.1}$$

is known as the error signal. The error signal is fed into a procedure which alters or adapts the parameters of the filter from time n to time $(n+1)$ in a well-defined manner. This process of adaptation is represented by the oblique arrow that pierces the adaptive filter block in the figure. As the time index n is incremented, it is hoped that the output of the

adaptive filter becomes a better and better match to the desired response signal through this adaptation process, such that the magnitude of $e(n)$ decreases over time. In this context, what is meant by "better" is specified by the form of the adaptive algorithm used to adjust the parameters of the adaptive filter. In the adaptive filtering task, adaptation refers to the method by which the parameters of the system are changed from time index n to time index $(n+1)$ The number and types of parameters within this system depend on the computational structure chosen for the system[5]-[7].

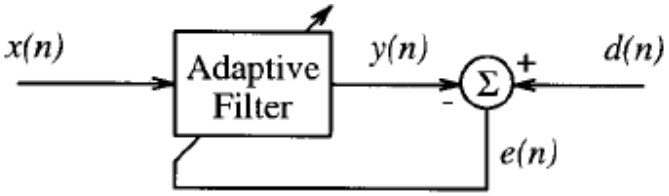

Fig.1.1. The general adaptive filtering problem [1].

Task of an Adaptive Filter

When considering the adaptive filter problem as illustrated in Fig. 1.1 for the first time, one can ask, "If we already have the desired response signal, what is the point of trying to match it using an adaptive filter?" In fact, the concept of "matching" $y(n)$ to $d(n)$ with some system obscures the subtlety of the adaptive filtering task. Consider the following issues that pertain to many adaptive filtering problems:

1. The quantity of interest is not always $d(n)$. Our desire may be to represent in $y(n)$ a certain component of $d(n)$ that is contained in $x(n)$, or it may be to isolate a component of $d(n)$. within the error $e(n)$ that is not contained in $x(n)$, Alternatively, we may be solely interested in the values of the parameters in $w(n)$ and have no concern about $x(n)$, $y(n)$, or $d(n)$ themselves.
2. There are situations in which $d(n)$ is not available at all times. In such situations, adaptation typically occurs only when $d(n)$ is available. When $d(n)$ is unavailable, we typically use our most-recent parameter estimates to compute $y(n)$ in an attempt to estimate the desired response signal $d(n)$.
3. There are real-world situations in which $d(n)$ is never available. In such cases, one can use additional information about the characteristics of a "hypothetical"

$d(n)$, such as its predicted statistical behaviour or amplitude characteristics, to form suitable estimates of $d(n)$ from the signals available to the adaptive filter. Such methods are collectively called blind adaptation algorithms. The fact that such schemes even work is a tribute both to the ingenuity of the developers of the algorithms and to the technological maturity of the adaptive filtering field.

It should also be recognized that the relationship between $x(n)$ and $d(n)$ can vary with time. In such situations, the adaptive filter attempts to alter its parameter values to follow the changes in this relationship as "encoded" by the two sequences $x(n)$ and $d(n)$ This behaviour is commonly referred to as tracking.

The Characteristics of an adaptive filtering algorithm is evaluated based on one or more of the following factors:

Rate of Convergence

It determines the number of iterations required to get to the vicinity of a steady-state solution. This quantity describes the transient behaviour of the algorithm. This is defined as the number of iterations required for the algorithm, under stationary conditions, to converge "close enough" to the optimum Wiener solution in the mean square sense. It is desirable to achieve the highest rate possible. Since in many applications the system has to meet stringent deadlines, the convergence must be fast enough to meet the deadlines and not to affect the performance.

Misadjustment

This quantity describes steady-state behaviour of the algorithm. This is a quantitative measure of the amount by which the ensemble averaged final value of the mean-squared error exceeds the minimum mean-squared error produced by the optimal Wiener filter.

Computational Complexity

The computational complexity is the determining factor of algorithm's demand of resources. By resources we usually mean the time of processing and the memory for data Storage. Adaptive algorithms are iterative methods that repeat their job all rounds. Thus, to estimate the computational power needed, we have to count the number of operations in a Single iteration. For block-based algorithms this value determines the number of operations needed to process N consecutive samples whereas for sample-based

algorithm's it is only for a single sample. Therefore, when comparing different algorithms together, the strategy is to count the number of operations for a block of N samples.

Robustness

This may be viewed as a combined requirement of maximum immunity against internal errors, such as quantization and round-of errors and insensitivity to external errors. Sometimes, however, it is better for the algorithm to be sensitive to certain changes of the environment, such as non-stationary of speech signals and noise processes. The trade-off between sufficient sensitivity and relative robustness is often a difficult task to solve.

Application of Adaptive Filters

An adaptive filter is useful whenever the statistics of the input signals to the filter are unknown or time-varying. Since its parameters are changing according to the conditions of an ambient environment, little or no a-priori information is required about the input signals. Hence, it is also of interest to note that adaptive systems are capable of working with non stationary signals, provided they can be considered stationary at least in a short interval. Numerous applications of adaptive filters have been proposed in different literatures. Two possible application scenarios of adaptive filters are given in fig 1.3, system identification and inverse filtering. For system identification the adaptive filter is used to approximate an unknown system. Both the unknown system and the adaptive filter are driven by the same input signal and the adaptive filter coefficients are adjusted in a way, that the output signal resembles the output of the unknown system, i.e the adaptive filter is used to approximate the unknown system. For inverse modelling or equalization the adaptive filter is used in series with the unknown system and the learning algorithm tries to compensate the influence of the unknown system on the test signal $u[n]$ by minimizing the (squared) difference between the adaptive filters output and the delayed test signal.

Other applications of adaptive filters include:

1. Channel equalization and interference suppression
2. Teleconferencing and videoconferencing
3. Hands-free telephony
4. Voice control

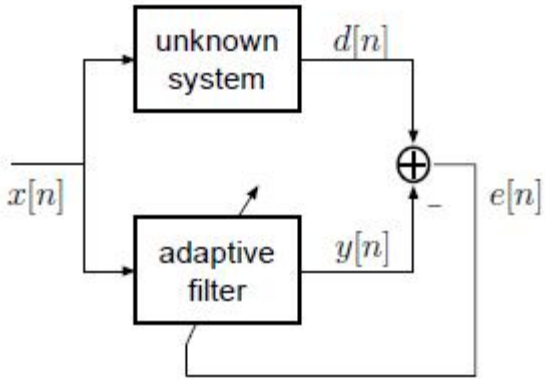

Fig. 1.2. System identification [4].

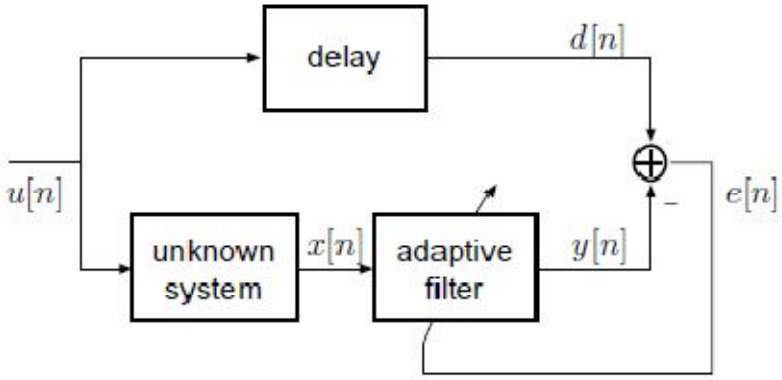

Fig.1.3. Inverse modelling [4].

1.2 Markov Model for Time-Varying Systems

The nature of time varying systems can be defined by either a stochastic process or a deterministic one. For a deterministic model, periodic time-varying characteristics are reported to be an appropriate one. Here we choose a stochastic model for the time varying kernels, one often finds useful to model such process by a first-order Markov model. Any system may become non stationary in one of the following two basic ways:

1. The frame of reference provided by the desired response may be time varying. Such a situation arises, for example, in system identification when an adaptive traversal filter is used to model a time varying systems. In this case, the correlation matrix of the tap inputs of the adaptive filter remain fixed, whereas the cross correlation vector between the tap inputs and the desired response assumes a time varying form.

2. The stochastic process supplying the tap inputs of the filter is non stationary. This situation arises , for examle when an adaptive filter is used to equalize a time varying channel. In that case, both the correlation matrix of the tap inputs in the adaptive traversal filter and the cross correlation of the tap inputs and the desired responce asume time varying forms.

Thus the traking details of a time varying systems not only are dependent on the type of adaptive filter employed but also are problem specific.

Here we will focus on a popular time varying model for a non stationary systems. The model is governed by two basis processess.

First Order Markov Process- The unknown dynamic equations of the environment are modeled by a traversal filter whose tap weight vector $h(n)$ undergoes a first order markov process, written in vector form as

$$h(n) = \phi h(n-1) + w(n) \tag{1.2}$$

where ϕ is a fixed parameter of the model and $w(n)$ is the process vector assumed to be of zero mean and correlation matrix R.

Multiple Regression- The Observable Feature of the Systems –The desired response, denoted by $d(n)$-is governed by the multiple linear regression model.

$$d(n) = h^T(n)x(n) + v(n) \tag{1.3}$$

Where, $v(n)$ is the measurement noise, assumed to be white with zero mean and varience σ_v^2. The model described above is a special case of the linear dynamical model. The system under consideration is the linear dynamical model of a nonstationary environment.

1.3 Overview of Parameter Estimation Algorithms

As the increasing application of digital techniques in the area of mobile, indoor and personal radio communications, it becomes increasingly important to analyze the existing adaptive algorithms and develop new adaptive algorithms for identifying time-varying system. It is obvious that a more generalized system model is necessary if we wish to develop a more general means square error (MSE) analysis method for system tracking problems. A widely used model is wide sense stationary uncorrelated scattering model [8] [9]. With this model, the different paths of the system are assumed to be uncorrelated.

The coefficient of each path, which is represented as a random variable, is assumed to be wide sense stationary.

An adaptive filter (AF) operating in a non-stationary environment needs to be able to track variations in input statistics. If implemented with infinite precision arithmetic, an adaptive filter has four sources of Misadjustment error.

(1) First, the error comes from the noise, i.e. measurement noise.

(2) Second is the estimation error which arises from basing estimates on windowed data Sequences. This causes the estimates of signal statistics to never converge in the mean Square sense.

(3) Lag noise, the second source of Misadjustment, is caused by the reaction time of the adaptation algorithm to changes in its environment. Lag error is minimized by rapidly discounting the past and basing estimates predominantly on recent data. These two errors place conflicting demands on the adaptation algorithm.

(4) Finally, model error could be contained in prediction error.

Different parameter estimation algorithms had been proposed. In the field of adaptive signal processing, least mean square (LMS) is extensively explored algorithm and has found wide applications for the stationary environments due to its implementation simplicity. However, the performance of LMS substantially degrades in the time-varying environment (non-stationary cases) [10]-[13]. Machhi et al. have evaluated the performance of the LMS algorithm in the non-stationary environment [14], where the non-stationary is introduced using the complex chirp exponential signal buried in additive white Gaussian noise (AWGN). For AR1 channel model, the LMS algorithm performs better than the RLS algorithm under some typical conditions.

The optimum algorithm for the non-stationary environment is Kalman filtering algorithm. A distinctive feature of the Kalman filter is that its mathematical formulation is described in terms of the state-space concepts. Another novel feature of the Kalman filter is that its solution is computed recursively, applying without modification to stationary as well as nonstationary environments. However, the computation complexity and requirement of the knowledge of system model may often preclude the above Kalman filter based approaches. And Kalman filter is suitable for both stationary and non-stationary environment, but it has too much computation complexity. The RLS algorithm is the algorithm which lies between LMS algorithm and Kalman filtering algorithm, i.e. it has less complexity than Kalman filtering algorithm and performs better than in LMS in non-stationary environment. For this reason, in this thesis work, we have chosen RLS

algorithm with some modification, to track the channels in non-stationary environment. In [15], Sayed et al. have presented the RLS algorithm as a special case of Kalman algorithm. In non-stationary environment, the substantial degradation in the tracking performance of the RLS algorithms observed due to the design constraints. Subsequently the extended RLS algorithm has been proposed in [16], which provides the better tracking performance than LMS algorithm.

Under time varying environment, the weighing factor in RLS algorithm is used to ensure that data in distant past are forgotten in order to afford the possibility of following the statistical variation of the observable data. A different form of weighting that is used is the exponential weighting factor or forgetting factor defined by, $\beta=\lambda^{(n-i)}$ .For time varying environment $0<\lambda<1$, for time invariant environment $\lambda=1$

The main dynamic force behind recursive estimation of adaptive signal processing is to track the parameter changes. The famous standard RLS algorithm, which is known to have the optimal properties in stationary environments, is unsuitable for non stationary environments. Thus many attempts have been directed to the development of modified versions of the RLS algorithm to include tracking capability in time-varying environments. Among these modified RLS algorithms, the best known is an exponential data weighting RLS algorithm using a forgetting factor. However, in certain situations, this algorithm can lead to a problem often referred to as the blow-up problem. Also lower the value of the forgetting factor, higher the tracking velocity but the higher the influence of the noise, that is, the larger the parametric errors. Recursive least squares algorithms have been used extensively in adaptive filtering, self-tuning control systems and system identification. RLS is well known for its good convergence property and small mean square error when the system is time-invariant However RLS is not effective for tracking time-varying parameters because it is difficult to find a suitable forgetting factor to provide good tracking in dealing with large model variations. To avoid these difficulties, the idea of a variable forgetting factor was introduced [17]. The general strategy for the control of variable forgetting factor (VFF) can be described as follows. Large forgetting factor (effectively large memory of data) is used when the learning is in the steady state and also there is no obvious model variation, while small one (to fade away the very old data) is applied when the model error is large. In time-varying environment, the control should be able to sense the change of the model and reduce the disturbance from the noise. In the environment with impulsive noise, at the incident of large error signal, there could be two possibilities. The error is due to either large model variation or impulse

noise. In case the former one occurred, the forgetting factor should be adjusted to make the filter response to the change; otherwise, the forgetting factor should remain large to neglect the effect of the impulse noise. In order to avoid the disturbance by the impulse noise, we can use the autocorrelations of nonzero lags to measure the model error and control the forgetting factor for those model errors not very large [18].

1.4 Non Linear Systems

Linear filters are useful in a large number of applications arid relatively simple from conceptual and implementation view points, there are many practical Situations there require nonlinear processing of the signals involved. This article explains adaptive nonlinear filters equipped with polynomial models of nonlinearity. The polynomial systems considered are those nonlinear systems whose output signals can be related to the input signals through a truncated Volterra series expansion, or a recursive nonlinear difference equation. The Volterra series expansion can model a large class of nonlinear systems and is beneficial in adaptive filtering applications because the expansion is a linear combination of nonlinear functions of the input signal. Such systems are attractive because they may be able to approximate many nonlinear systems with great parsimony in the use of coefficients.

Linear filters have played a very crucial role in the development of various signals processing techniques. The obvious advantage of linear filters is their inherent simplicity. Design, analysis, and implementation of such filters are relatively straightforward tasks in many applications. However, there are several situations in which the performance of linear filters is unacceptable. A simple but highly pervasive type of nonlinearity is the saturation- type nonlinearity. Trying to identify these types of systems using linear models can often give misleading results. Another situation where nonlinear models will do well when linear models will fail miserably is that of trying to relate two signals with non overlapping spectral components.

System analysis using nonlinear structures has several applications. High-speed communications channels often need nonlinear equalizers for acceptable performance. Although channel equalization using linear, tap delay line structures is adequate in many applications, there are several other situations when they will not work at all. For example, Lucky [19] has conjectured that error probability performance of data

transmission systems operating at rates better than 4800 bits/s is due almost entirely to nonlinear distortion.

In telephone transmission, nonlinearities arise principally from inaccuracies in signal companding. In digital satellite links, the satellite amplifiers are usually driven to near the saturation point and they exhibit highly nonlinear characteristics. Several researchers have used Volterra series representation of nonlinear systems to implement nonlinear channel equalizer. Other applications of nonlinear models and filtering in communication problems include echo cancellation, performance analysis of data transmission systems, adaptive noise cancellation and detection of nonlinear functions of Gaussian processes. Nonlinear filters are very useful in modelling biological phenomena, myoelectric signal processing, and characterization of semiconductor devices, image processing, modelling drift oscillations in random seas and several other areas.

Unlike the case of linear systems which are completely characterized by the system's unit impulse response function; it is impossible to find a unified framework for describing arbitrary nonlinear systems. Consequently, the researchers working on nonlinear filters are forced to restrict themselves to certain nonlinear system models that are less general. Nonlinear filters developed using such models include order statistics filters, homomorphic filters morphological filters and filters based on Volterra and other polynomial descriptions of the nonlinearities involved. Order statistics filters are attractive because of their robustness and computational simplicity. As the name suggests, they are based on the order statistics (i.e., the location of any given data sample in a rearrangement of the samples under consideration in the ascending or descending order of magnitude) of the input signal to the filter. A very widely used order statistic filter is the median filter. Such filters have good edge preserving properties and are very useful in removing additive impulse noise (in general, noise belonging to long-tailed distributions) from the input signals, and have found applications especially in image processing. Homomorphic filters are among the oldest types of nonlinear filters and have applications in image enhancement, seismic signal processing, and removal of multiplicative noise from input signals. Models of human visual systems based on homomorphic filters have been extensively used in image coding applications. Morphological filters utilize geometric features of the input signals and are employed in applications involving shape recognition, edge detection, and others. A good description of time invariant nonlinear filters belonging to all of the above classes may be found in many articles.

Here, we will concentrate on polynomial models of nonlinearity. Such models are more general than most of the other models used in research work. Two specific cases will be considered in some detail - adaptive filters employing truncated Volterra series representation of nonlinear systems and those using recursive nonlinear difference equations to relate the input and output signals of the system. Even though it is possible to treat the truncated Volterra series representation as a special case of the recursive nonlinear system representation and consider a unified framework for polynomial system representations, we will discuss the two cases separately. The Volterra system model is extremely popular in adaptive nonlinear filtering and has developed an identity of its own in the last few years. The theory of adaptive nonlinear filters employing nonlinear feedback models, on the other hand, is very much in its infancy: and while such systems are very attractive from an implementation point of view, there are several problems for which effective solutions have not yet been found. Discussing the two cases separately will enable us to treat such problems in a better manner. Adaptive order statistic filters are available.

1.4.1 Volterra Series

The Spanish mathematician Vito Volterra first introduced the notion of what is now known as a Volterra series in his “Theory of Functionals”. The first major application of Volterra’s work to nonlinear circuit analysis was done by the mathematician Norbert Wiener at M.I.T., who used them in a general way to analyze a number of problems including the spectrum of an FM system with a Gaussian noise input. Since then, Volterra series have found a great deal of use in calculating small, but nevertheless troublesome, distortion terms in transistor amplifiers and other systems. The Volterra series is a model for non-linear behaviour similar to the Taylor series. It differs from the Taylor series in its ability to capture 'memory' effects. The Taylor series can be used to approximate the response of a nonlinear system to a given input if the output of this system depends strictly on the input at that particular time. In the Volterra series the output of the nonlinear system depends on the input to the system at all other times. This provides the ability to capture the 'memory' effect of devices such as capacitors and inductors.

The common method in the communication systems design is the identification and compensation of unwanted nonlinearities. It was demonstrated that unwanted nonlinearities in the system will have a determinant effect on his performance [20]. There

are various ways of reducing the effects of undesired nonlinearities [21]. The Volterra series have been widely applied as nonlinear system modelling technique with considerable success [22],[23]. However, at present, none general method exists to calculate the Volterra kernels for nonlinear systems, although they can be calculated for systems whose order is known and finite. When the nonlinear system order is unknown, adaptive methods and algorithms are widely used for the Volterra kernel estimation. The accuracy of the Volterra kernels will determine the accuracy of the system model and the accuracy of the inverse system used for compensation. The speed of kernel estimation process is also important. A fast kernel estimation method may allow the user to construct a higher order model that gives an even better system representation. In this thesis we present the estimation of first and second order Volterra kernel using different adaptive algorithm.

1.4.2 Volterra Model

A linear, causal system with memory can be described by the convolution representation

$$y(t) = \int_{-\infty}^{\infty} h(\partial)x(t-\partial)d\partial \quad (1.4)$$

Where $x(t)$ is the input, $y(t)$ the output, and $h(t)$ the impulse response of the system. A Nonlinear system without memory can be described with a Taylor series

$$y(t) = \sum_{n=1}^{\infty} a_n\ [x(t)]^n \quad (1.5)$$

Where $x(t)$ is the input, $y(t)$ the output and a_n are the taylor series coefficient.

A Volterra series combines the above two representations to describe a nonlinear system with memory

$$y(t) = \sum_{n=1}^{\infty} 1/n! \int_{-\infty}^{\infty} du1 \dots\dots\dots\dots \int_{-\infty}^{\infty} dun\ \ gn\ (u1 \dots\dots\dots\dots un) \prod_{r=1}^{n} x(t-ur) \quad (1.6)$$

$x(t)$ is the input, $y(t)$ is the output, and the $gn\ (u1, \dots\dots\dots un)$ are called the Volterra kernels of the system or simply the kernel. The ui are time variables and are labelled ui instead of ti to distinguish them better from t. For n=1, $g1(u1)$ will be recognized as the familiar impulse response $h(t)$ in equation (1); thus, gn for $n > 1$ are rather like "higher-order impulse responses." These serve to characterize the various orders of nonlinearity.

1.4.3 Order of Volterra Model

Zeroth and First-Order Volterra Model

The zeroth-order Volterra model is just a constant defined as

$$Y_0[x(n)]=h_0 \tag{1.7}$$

Where, x(n) is the input signal and h_0 is a constant

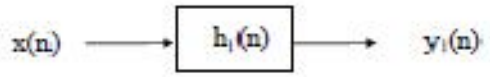

Fig.1.4. First order linear system block diagram [6].

The first-order Volterra system is basically the same as the linear system. In other words, the linear system is a subclass of the Volterra system. Consider a general isolated linear system as shown in figure 1.4 Where the $h_1(n)$ represents the linear filter coefficients. The output $y_1(n)$ can be expressed by input x(n) as:

$$y_1(n) = x(n)*h_1(n) = \sum_{I=0}^{\infty} h_1(k)x(n-k) \tag{1.8}$$

Where, * means linear convolution. If all the components in $h_1(n)$ can be represented by some linear combination of orthonormal basis $b_m(n)$. This means that the first-order Volterra kernel $h_1(k)$ in equation(1.8) can be represented by:

$$h_i(k) = \sum_{M=0}^{\infty} a_1(m)\, b_m(k) \tag{1.9}$$

Where, $a_l(m)$ is a proper constant. Note that $\{b_m(n), 0 \le m \le \infty\}$ is the set of orthonormal basis, which means $[bi(n), b_m(n)] = \delta(l - m)$

Where [,] denotes the inner product and $\delta(l - m)$ is the Dirac delta function. Substituting equation we can define the first order Volterra functional as:

$$y_1(n) = \sum_{I=0}^{\infty} \sum_{M=0}^{\infty} a_1(m) b_m(k)\, x(n-k) \tag{1.10}$$

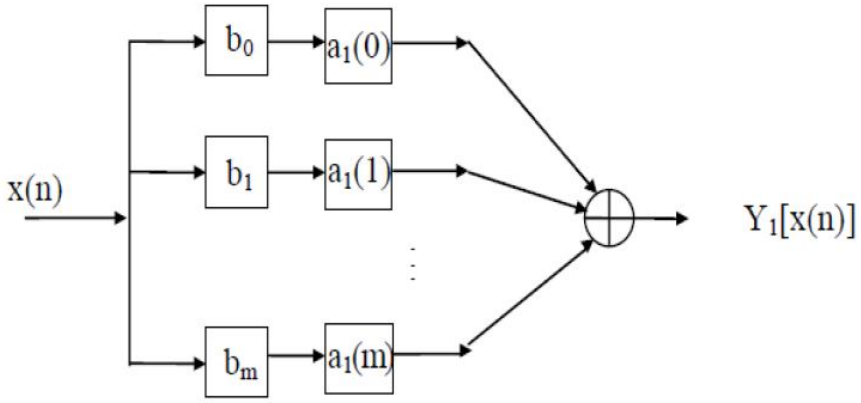

Fig.1.5. First order Volterra model [6].

For the most general form of first-order Volterra system, we should include the DC term in equation 1.8, which can be expressed in terms of the Volterra functional Y_0 [x(n)] and Y_1[x(n)] as:

$$Y(n) = h_0 + x(n) * h_1(n) = Y_0[x(n)] + Y_1[x(n)] \quad (1.11)$$

From equation 1.10, we conclude that a general first-order Volterra system with DC term is one for which the response to a linear combination of inputs is the same as the linear combination of the response of each individual input.

Second-Order Volterra Model

The linear combination concept described above can be extended to the second-order case, which is one for which the response to a second-order Volterra system is a linear combination of the individual input signals.

Fig.1.6. Second order Volterra model block diagram[6].

$$y_i(n) = \sum_{i=0}^{\infty} h_2(k_1, k_2)\, x(n - k_1) x(n - k_2) \quad (1.12)$$

Where h_2 (k_1, k_2) is defined as the second-order Volterra kernel. As in the literature, for simplicity and without loss of generality, we assume the kernel to be symmetric, which implies that h_2(k1, k2) = h_2(k2, k1). If h_2(k1, k2) can be expressed by the linear combination of orthonormal basis set b_k (n), then h_2 (k1, k2) can be written as

$h_2(k_1, k_2) = \sum_{m1=0}^{\infty} \sum_{m2=0}^{\infty} a_2(m_1, m_2) b_{m1}(k_1) b_{m2}(k2)$ (1.13)

Substituting equation 1.13 in equation 1.12, we can obtain the output as

$$Y_2(n) = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} a_2(0,0) b_0(k_1) b_0(k_2) x(n-k_1) x(n-k_2) +$$
$$\sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} a_2(0,1) b_0(k_1) b_1(k_2) x(n-k_1) x(n-k_2) +$$
$$\sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} a_2(1,0) b_1(k_1) b_0(k_2) x(n-k_1) x(n-k_2) + \ldots\ldots \quad (1.14)$$

$= a_2(0,0)[b_0(n)*x(n)]_2 + a_2(1,1)[b_1(n)*b_2(n)]_2 + \ldots\ldots$

$+ [a2(0,1)] + a_2(1,0)][b_0(n)*x(n)][b_1(n)*x(n)] + \ldots\ldots\ldots$ (1.15)

This equation, which is defined as the second-order Volterra functional

$y_2(n) = Y_2[x(n)]$ (1.16)

In equation 1.12, we recognize that all the operations are two-dimensional convolutions, therefore equation 1.16 is a second-order homogeneous functional; i.e.

$y_2[cx(n)] = c_2 Y_2[x(n)]$ (1.17)

Consider a special case such that the linear orthonormal set contains two orthonormal bases $\{b_m(n), 0 \leq m \leq 1\}$. From equation 1.13, the second-order Volterra functional $y_2(n)$ can be expressed as

$y_2(n) = a_2(0,0)\,[b_0(n)*x(n)]2 + a_2(1,1)\,[b_1(n)*x(n)]2$

$+ [a_2(0,1) + a_2(1,0)]\,[b_0(n)*x(n)]\,[b_1(n)*x(n)]$ (1.18)

The block diagram of equation1.18 is shown in figure 1.6

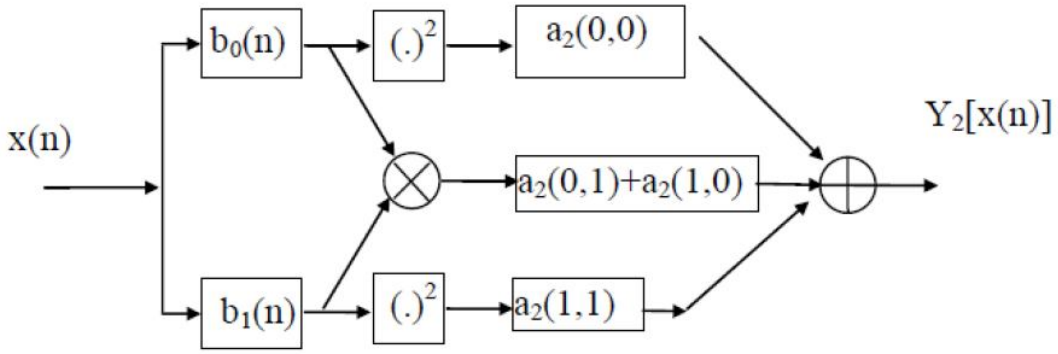

Fig.1.7. Second order Volterra system with orthonormal basis set $\{b_0, b_1\}$ [6].

Based on the description above, the general second-order Volterra system can be represented in terms of $Y_0[x(n)]$, $Y_1[x(n)]$ and $Y_2[x(n)]$ which is

$Y(n) = h_0 + \sum_{k_1}^{\infty} h_1(k) x(n-k) + \sum_{k_1}^{\infty} \sum_{k_2}^{\infty} h_2(k_1, k_2) x(n-k_1) x(n-k_2)$

$= Y_0[x(n)] + Y_1[x(n)] + Y_2[x(n)]$ (1.19)

To make the VS filter adaptive is straight forward, because adaptive least square algorithm from linear adaptive models can be used. Two well known adaptive algorithms are the LMS and the RLS algorithm. The LMS algorithm has low computational complexity and slow convergence in least square sense and the RLS algorithm has high computational complexity and fast convergence. It is pointed out that the linear expansion in the input vector x_N will cause the eigen value spread to increase, even with a white input signal. Therefore, It is important to use algorithm whose convergence speed is on dependent or less dependent on the statistics of the input signal. It is well known that the LMS algorithm suffers in convergence speed when the Eigen value spread of the autocorrelation matrix is large. One approach to circulate this dilemma is to use the RLS algorithm at the expense of high computational complexity.

The VS filter will fail if it has to control discontinuities, for instance a saturation type of nonlinearity. This is because the VS expansion is taylor series with memory which only fits the data well when the function are smooth ,in the sense that they are at least once differentiable .

1.5 Truncated Volterra Series Expansion for Nonlinear Systems

Let x[nl and y[n] represent the input and output signals, respectively, of a discrete-time and causal nonlinear system. The Volterra series expansion for y[nl using x[n] is given by

$$y(n) = h_0 + \sum_{m1=0}^{\infty} h1(m_1) + \sum_{m1=0}^{\infty} \dots\dots \sum_{m2}^{\infty} hp(m_1, m_2 \dots mp)\ \text{x(n-}m_1\text{)}\ \text{x(n-}m_2\text{)....x(n-}m_p\text{)} + ... \quad (1.20)$$

Where, h_p[ml, m2. ..., mp] is known as the p-th order Volterra kernel of the system. Without any loss of generality, one can assume that the Volterra kernels are symmetric, i.e., h [m, m2. ...mp] is left unchanged for any of the possible permutations of the indices m, m2, ..., mp systems involving this type of nonlinearity. Even though clearly not applicable in all situations, Volterra system models have been successfully employed in a wide variety of applications, and such models continue to be popular with researchers in this area. Among the early works on nonlinear system analysis is a very important contribution by Wiener [24]. Following his work, several researchers have employed Volterra series expansion and related representations for estimation and time-invariant nonlinear system identification. His analysis technique involved white Gaussian input signals and used "G-functionals" to characterize nonlinear system behaviour. Since an

infinite series expansion like in equation 1.20 is not useful in filtering applications, one must work with truncated Volterra series expansions in the following form:

$y(n) = h_0 + \sum_{m1=0}^{N-1} h1(m_1) + \sum_{m1=0}^{N-1} \ldots\ldots \sum_{m2}^{N-1} hp(m_1, m_2 \ldots mp)$ x(n-m_1) x(n-m_2)....x(n-m_p) + ... (1.21)

From the Fig 1.8 A truncated Volterra system of order P = 2 and N - 1 = 2 delay elements. (h_0 can often be estimated outside the basic adaptive filter structure. Therefore, we will, without loss of generality, assume that h_0 = 0.)Note that this system **is** linear in the input signal to each coefficient. This fact highly simplest the design problems involving Volterra series representations. On the other hand, even for moderately large values of N and P, the number of coefficients becomes very large. Consequently, the truncated Volterra series representation **is** most useful in applications where the values of N and P are relatively small.

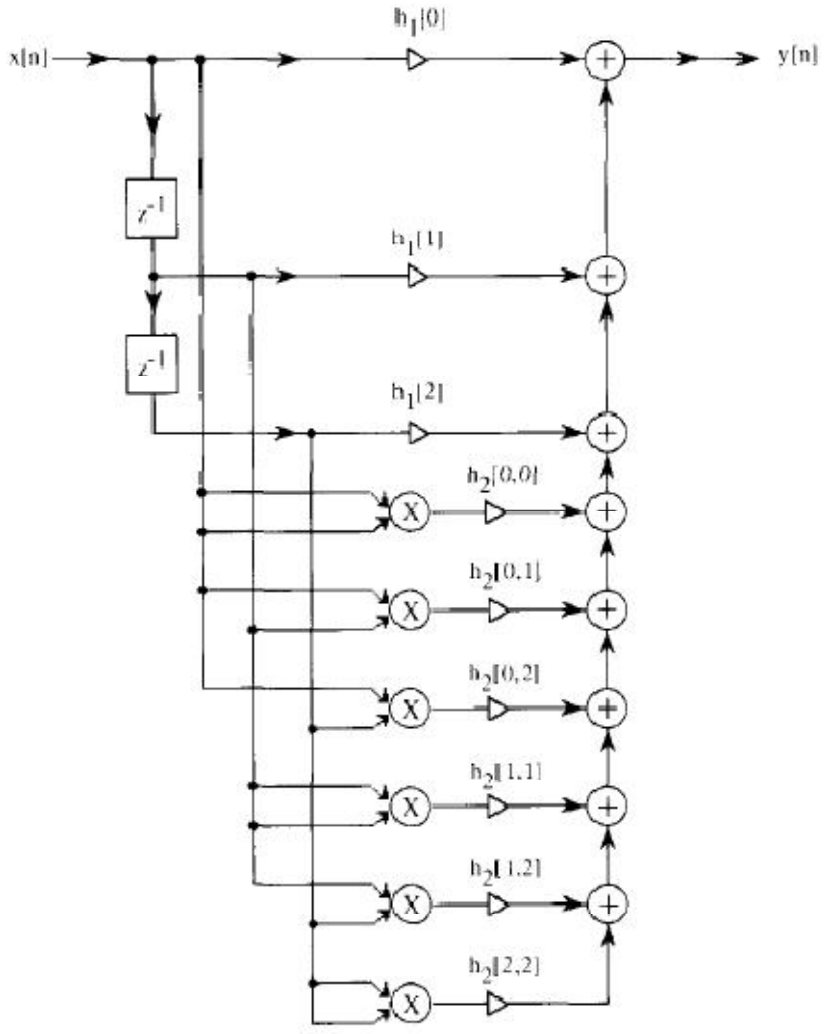

Fig.1.8. Truncated Volterra system of order two and N-1=2 delay element [25].

One big disadvantage for the Volterra system model is that the complexity of implementing filters using this model can be very large even for moderately large values of N and P. Consequently, most of the practical applications of systems employing Volterra series expansions involve low-order models.

1.6 Problem Statement

This thesis presents the following work

1) Firstly, the parameter estimation of first and second order time varying Volterra systems using Kalman filter and RLS algorithm with fixed forgetting factor is presented.

2) Then we will estimate the time varying Volterra kernel using dynamic forgetting factor recursive least square algorithm (DFFRLS), and we will observe it's tracking performance in comparison to other adaptive algorithm.

1.7 Organization of Thesis

This thesis organised in four chapters:

Chapter 1 Summarize the adaptive filters, filtering problem, nonlinear systems, Morkov model which represent time varying systems, then the expansion of Volterra series for first and second order.

Chaprer 2 describes the literature review of different work related to parameter estimation using different adaptive algorithms.

Chapter 3 presents the three adaptive algorithms i.e Kalman, RLS and dynamaic forgetting factor recursive least square algorithm.

Chapter 4 Presents the simulation results of estimation of all time varying Volterra kernels for first and second order using Kalman, RLS and DFFRLS algorithm.

Finally we conclude our work by comparing different parameter estimation algorithms for time varying Volterra system.

CHAPTER 2

LITERATURE REVIEW

In this chapter, the technical background of this thesis is reviewed. Initially the tracking performances of Kalman and RLS algorithm have been proposed in this thesis. The nonlinear model under consideration is Volterra model. Then, another algorithm named dynamic forgetting factor recursive least square algorithm have been presented. Now, we will go through different literature work and research work which is required to understand and implement new ideas.

V. J. Mathews [25] proposed the nonlinear Volterra model with different adaptive algorithms. His article explains that adaptive nonlinear filters equipped with polynomial models of nonlinearity. The polynomial systems considered here are those nonlinear systems whose output signals can be related to the input signals through a truncated Volterra series expansion, or a recursive nonlinear difference equation. Such systems are attractive because they may be able to approximate many nonlinear systems with great parsimony in the use of coefficients.

M. schetzen [26] gives an introduction to linear and nonlinear systems. In this text the results are developed by considering the basic concepts of linear systems and generalising these to higher-order Volterra systems. a study of the auto- and cross-correlation of system responses the wiener g-functionals and the decomposition of the wiener characterisation known as the wiener model is developed and analysed. Finally, the gate-function model is introduced as a generalisation of the Wiener model for arbitrary input.

Various new techniques related to the second-order Volterra filtering problem have been presented by T. Koh and E. J. Powers [27]. The results presented in this paper are intended mainly to reduce the complexity involved with the Volterra filtering problem. Iterative factorization and the adaptive algorithm are used to implement Volterra filters. A simple minimum mean-square error solution for the Volterra filter is derived, based on the assumption that the filter input is Gaussian. Also they proposed an iterative factorization technique to design a subclass of the Volterra filters, which can alleviate the complexity of the filtering operations considerably. Finally, an adaptive algorithm for the

Volterra filter is investigated along with its mean convergence and asymptotic excess mean-square error

J. B. MacNeil etal. [28] has presented a new method for the identification of time-varying systems from ensemble data using singular value decomposition. This technique, which produces a series of nonparametric impulse response function representations of the system as it varies with time, is capable of tracking rapid changes in dynamics with no a priori assumptions regarding the system's dynamic structure, or the nature of its time-varying behaviour.

M. Green and A. M. Zoubir [29] addressed the task of selecting basis sequences for a time-varying Volterra model in which the weighted sum of the basis sequences represents the time-variation. To enable identification, the system's time variation is approximated by a weighted sum of known basis sequences. Using wavelet packet basis sequences increases the flexibility of the model, allowing a suitable basis to be selected. A basis selection procedure is formulated using the Best Basis algorithm to choose the minimum entropy wavelet packet basis.

The problem of identification and tracking of time varying nonlinear systems is addressed by A. E. Nordsjo and L. H. Zetterberg [30]. the Wiener system that consists of a dynamic time-varying linear part followed by a fixed nonlinearity and the Hammerstein system in which the order of these two blocks is reversed are studied. The extended Kalman filter (EKF) algorithm is applied. It is also shown in this paper that this algorithm can be reformulated in terms of a nonlinear minimization problem with a quadratic inequality constraint in order to ensure exponential stability, resulting in the algorithm CEKF.

Binwei Weng and Kenneth E. Barner [31] proposed a proposed a novel method for time-varying Volterra system identification. First, the TV kernels are assumed to obey a Gauss-Markov stochastic difference equation. The TVVS is then reformulated into a state-space model. In the state-space equations, there are three unknowns:

1. The state transition matrix
2. The noise covariance matrix of process noise
3. The variance of measurement noise are all assumed to be unknown.

Then he form the input and output autocorrelation to exploit the dependence of the correlation matrix of the kernel vector on them. Then he use the estimated state transition matrix and noise covariance in the Kalman filtering algorithm to recursively estimate the TV Volterra kernels.

The techniques presented in this paper have allowed the variation of the forgetting factor during the operation of FRLS adaptive Filters structures. B. Toplis and *S* . Pasupathy [32] presented prewindowed and growing memory covariance algorithms in transversal and lattice structures. Each of the algorithms simplifies to their original form when the FF is not undergoing transition. The VFF algorithms allow the user greater flexibility in controlling the tradeoff between lag and estimation noise. This translates into improved tracking performance when the AF is operating in a nonstationary environment.

In this paper, Y.S. Cho etal. [33] proposed an AR method of estimating time-frequency representations of nonstationary signals using variable forgetting factors . The error arising from finite data increases proportionally to the difference between X and unity. However, in a nonstationary situation, the error caused by the nonstationarity is reduced by choosing a small forgetting factor. To decrease the combination of these errors, first he define the extended prediction error which is used to estimate the nonstationarity of the signal. Then, the time-varying forgetting factor is adapted to the signal via the criterion of the extended prediction error. Finally, the time-varying spectrum is estimated by the RLS with the time-varying forgetting factor.

As in above correspondence, in this paper Y. S. Cho etal. [34] has presented an AR method of estimating time-varying spectra of nonstationary signals using VFF's which are adapted to the signals via the criterion of the extended prediction error. This method has good adaptability in the nonstationary situation and low variance in the stationary situation. The feasibility of the approach is demonstrated with both simulation and experimental data.

In this paper Ning Zhou Nils Holte [35] shows how the tracking properties of Least Squares (LS) estimation may be improved for a time variant channel estimation (system identification) problem, by introducing a parametric time variant model instead of the conventional model which is a constant impulse response in each iteration. A recursive LS algorithm is developed for the case that the model varies linearly with time.

Simulation results are given for examples where the unknown channel has linear, sinusoidal, and stochastic variations with time. A significant improvement of the tracking properties compared to LMS and conventional LS algorithms is demonstrated for cases with high signal to noise ratio and 'smooth" variations of the unknown channel.

In this paper, this frequency domain analysis method has been developed to a systematic approach to perform the MSE analysis for adaptive tracking algorithms. Jingdong Lin, and John G. Proakis [36] has performed this analysis with the aid of two new concepts

1. The tracking filter
2. The tracking error filter.

By employing the frequency domain method, they analyzed the existing adaptive tracking algorithms, developed a new optimal windowed RLS algorithm and studied the robust design of optimal windows.

Deva K. Borah and Brian D. Hart has proposed [37] the rectangular-windowed recursive least squares algorithm in conjunction with a polynomial time-varying channel model for estimating a time-varying frequency-selective channel. This method achieves better performance in tracking time-varying channels than the conventional LMS and the exponentially windowed RLS algorithms. The performance of the algorithm is dependent on the choice of the window size and the polynomial model order used, which in turn depend on the fading rate and the signal-to-noise ratio.

In this correspondence, the variable forgetting factor linear least squares algorithm is presented by Seongwook Song [38] to improve the tracking capability of channel estimation. To reduce estimation error due to model mismatch, he incorporate the modified variable forgetting factor into the proposed algorithm. Compared to the existing algorithms—exponentially windowed recursive least squares algorithm with the optimal forgetting factor and linear least squares algorithm—the proposed method makes a remarkable improvement in a fast fading environment.

Amit Kumar Kohli etal. [39] has presented variable forgetting factor (VFF) least squares (LS) algorithm for polynomial channel paradigm for improved tracking performance under nonstationary environment. The main focus is on updating VFF when each time-varying fading channel is considered to be a first-order Markov process. Firstly the time-varying frequency-selective wireless system model has been presented and Details about

the presented second-order polynomial model-based least squares algorithm using VFF (LSn2-VFF) are shown. Mathematical formulation of NVFF based on the extended estimation error criterion has also been discussed, which shows that The LSn-NVFF algorithm not only performs better than LSn-VFF algorithm, but also precludes the need of LMS algorithm in variable forgetting factor updating at each iteration, which in turn reduces computational burden

CHAPTER 3

ADAPTIVE TRACKING ALGORITHMS FOR TIME-VARYING ENVIRONMENT

As we have seen in previous chapter the Volterra series have been widely applied as nonlinear system modelling technique with considerable success. However, at present, none general method exists to calculate the Volterra kernels for nonlinear systems, although they can be calculated for systems whose order is known and finite. When the nonlinear system order is unknown, adaptive methods and algorithms are widely used for the Volterra kernel estimation. In this chapter we use second order volterra model and adaptive algorithms to estimate the time varying Volterra kernel. Gauss Morkov model is use to represent the time varying environment.

3.1 An Introduction to Kalman Filter

The Kalman filter is named after Rudolph E. Kalman [40], who in 1960 published his famous paper describing a recursive solution to the discrete-data linear filtering problem (Kalman 1960). Its purpose is to use measurements observed over time, containing noise (random variations) and other inaccuracies, and produce values that tend to be closer to the true values of the measurements and their associated calculated values. The Kalman filter has many applications in technology, and is an essential part of space and military technology development. A very simple example and perhaps the most commonly used type of Kalman filter is the phase-locked loop, which is now ubiquitous in FM radios and most electronic communications equipment. Extensions and generalizations to the method have also been developed.

The Kalman filter is essentially a set of mathematical equations that implement a predictor-corrector type estimator that is optimal in the sense that it minimizes the estimated error covariance—when some presumed conditions are met. Since the time of its introduction, the Kalman filter has been the subject of extensive research and application, particularly in the area of autonomous or assisted navigation. This is likely due in large part to advances in digital computing that made the use of the filter practical, but also to the relative simplicity and robust nature of the filter itself. Rarely do the conditions necessary for optimality actually exist, and yet the filter apparently works well for many applications in spite of this situation. Of particular note here, the Kalman filter

has been used extensively for tracking in interactive computer graphics. We use a single-constraint-at-a-time Kalman filter in HiBall Tracking System [41]. It has also been used for motion prediction and it is used for multi-sensor (inertial-acoustic) fusion in the commercial Constellation wide area tracking system.

Suppose we have a linear system model. We want to use the available measurements *y* to estimate the state of the system *x*. We know how the system behaves according to the state equation, and we have measurements of the position, so how can we determine the best estimate of the state *x*? We want an estimator that gives an accurate estimate of the true state even though we cannot directly measure it. What criteria should our estimator satisfy? Two obvious requirements come to mind. First, we want the average value of our state estimate to be equal to the average value of the true state. That is, we don't want our estimate to be biased one way or another. Mathematically, we would say that the expected value of the estimate should be equal to the expected value of the state.

Second, we want a state estimate that varies from the true state as little as possible. That is, not only do we want the average of the state estimate to be equal to the average of the true state, but we also want an estimator that results in the smallest possible variation of the state estimate. Mathematically, we would say that we want to find the estimator with the smallest possible error variance. It so happens that the Kalman filter is the estimator that satisfies these two criteria. But the Kalman filter solution does not apply unless we can satisfy certain assumptions about the noise that affects our system.

3.2 The Discrete Kalman Filter

This section describes the filter in its original formulation (Kalman 1960) where the measurements occur and the state is estimated at discrete points in time.

The Process to be estimated

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{3.11}$$

with a measurement $z \in \Re^m$ that is

$$z_k = Hx_k + v_k \tag{3.12}$$

The random variables w_k and v_k represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions

$$p(w) < N(0, Q) \tag{3.13}$$

$$p(v) < N(0, R) \tag{3.14}$$

In practice, the process noise covariance and measurement noise covariance matrices might change with each time step or measurement, however here we assume they are constant.

The n*n matrix A in the difference equation equation (3.11) relates the state at the previous time step k-1 to the state at the current step , in the absence of either a driving function or process noise. Note that in practice A might change with each time step, but here we assume it is constant. The n*l matrix B relates the optional control input u € R to the state *x*. The m*n matrix H in the measurement equation equation (3.12) relates the state to the measurement z_k. In practice H might change with each time step or measurement, but here we assume it is constant.

The Computational Origins of the Filter

We define $\hat{x}_k^- = \Re^n$

(the "super minus") to be our a priori state estimate at step k given knowledge of the process prior to step k, and $\hat{x}_k = \Re^n$ to be our a posterior state estimate at step k given measurement . We can then define a priori and a posterior estimate error as

$$e_k^- = x_k - \hat{x}_k^-$$

$$e_k = x_k - \hat{x}_k$$

The *a priori* estimate error covariance is then

$$P_k^- = E[e_k^- e_k^{-T}] \tag{3.15}$$

and the *a posteriori* estimate error covariance is

$$P_k = E[e_k e_k^T] \tag{3.16}$$

In deriving the equations for the Kalman filter, we begin with the goal of finding an equation that computes an *a* posteriori state estimate $\hat{x}_k$ as a linear combination of an *a priori* estimate $\hat{x}_k^-$ and a weighted difference between an actual measurement z_k and a measurement prediction as shown below in equation (3.17). Some justification for equation (3.11) is given in "The Probabilistic Origins of the Filter" found below.

$$\hat{x}_k = \hat{x}_k^- + k(z_k - H\hat{x}_k^-) \tag{3.17}$$

The difference $(z_k - H\hat{x}_k^-)$ in equation (3.17) is called the measurement innovation, or the *residual*. The residual reflects the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. A residual of zero means that the two are in complete agreement.

The n*m matrix K in equation (3.17) is chosen to be the gain or blending factor that minimizes the *a* posteriori error covariance equation (3.16). This minimization can be accomplished by first substituting equation (3.17) into the above definition for e_k, substituting that into equation (3.16), performing the indicated expectations, taking the derivative of the trace of the result with respect to K, setting that result equal to zero, and then solving for K.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{3.18}$$

Looking at equation (3.18) we see that as the measurement error covariance R approaches zero, the gain K weights the residual more heavily. Specifically

$$\lim_{R_k \to 0} K_K = H^- \tag{3.19}$$

On the other hand, as the a priori estimate error covariance P_k^- approaches zero, the gain K weights the residual less heavily. Specifically,

$$\lim_{P_k^- \to 0} K_K = 0 \tag{3.20}$$

Another way of thinking about the weighting by K is that as the measurement error covariance R approaches zero, the actual measurement is z_k "trusted" more and more, while the predicted measurement $H\hat{x}_k^-$ is trusted less and less. On the other hand, as the *a* priori estimate error covariance P_k^- approaches zero the actual measurement z_k is trusted less and less, while the predicted measurement $H\hat{x}_k^-$ is trusted more and more.

The Probabilistic Origins of the Filter

The justification for equation (3.17) is rooted in the probability of the *a priori* estimate $\hat{x}_k^-$ conditioned on all prior measurements z_k (Bayes' rule). For now let it suffice to point out that the Kalman filter maintains the first two moments of the state distribution

$$E[x_k] = \hat{x}_k \tag{3.21}$$

$$E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] = P_k \tag{3.22}$$

The a posteriori state estimate equation (3.17) reflects the mean (the first moment) of the state distribution— it is normally distributed if the conditions of equation (3.13) and equation (3.14) are met. The a posteriori estimate error covariance equation (3.16) reflects the variance of the state distribution (the second non-central moment). In other words,

$$p(x_k / z_k) \sim N(E[x_k], E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T])$$

$$= N(\hat{x}_k, p_k) \tag{3.23}$$

3.3 The Discrete Kalman Filter Algorithm

We will begin this section with a broad overview, covering the "high-level" operation of one form of the discrete Kalman filter (see the previous footnote). After presenting this high-level view, we will narrow the focus to the specific equations and their use in this version of the filter. The Kalman filter estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of noisy measurements. As such, the equations for the Kalman filter fall into two groups time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimates for the next time step. The measurement update equations are responsible for the feedback—i.e. for incorporating a new measurement into the a priori estimate to obtain an improved a posteriori estimate. The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations. Indeed the final estimation algorithm resembles that of a predictor-corrector algorithm for solving numerical problems as shown in Fig 3.1

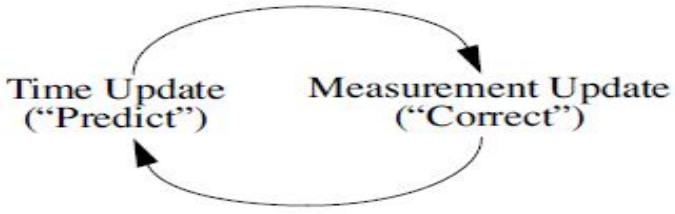

Fig.3.1.The discrete Kalman filter cycle [50].

The figure 3.1 shows the time update and measurement update cycle. The time update projects the current state estimate ahead in time. The measurement update adjusts the

projected estimate by an actual measurement at that time. The specific equations for the time and measurement updates are presented below

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$$

$$P_k^- = AP_{k-1}A^T + Q \tag{3.24}$$

The above equation is Discrete Kalman filter time update equations.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + k(z_k - H\hat{x}_k^-)$$

$$P_k = (I - K_k H)P_k^- \tag{3.25}$$

The above equation is Discrete Kalman filter measurement update equations .The first task during the measurement update is to compute the Kalman gain. Notice that the equation given here as equation (3.21) is the same as equation (3.18). The next step is to actually measure the process to obtain, and then to generate an a posteriori state estimate by incorporating the measurement as in equation (3.20). Again equation (3.20) is simply equation (3.17) repeated here for completeness. The final step is to obtain an a posteriori error covariance estimate via equation (3.20). After each time and measurement update pair, the process is repeated with the previous a posteriori estimates used to project or predict the new a priori estimates. This recursive nature is one of the very appealing features of the Kalman filter—it makes practical implementations much more feasible than (for example) an implementation of a Wiener filter (Brown and Hwang 1996) which is designed to operate on all of the data directly for each estimate. The Kalman filter instead recursively conditions the current estimate on all of the past measurements.

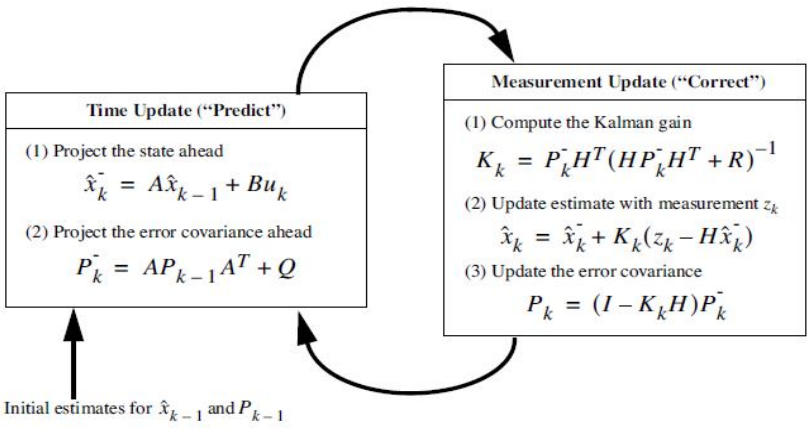

Fig.3.2. A complete picture of the operation of the Kalman filter [50].

3.4 Estimation of Time Varying Volterra System Using Kalman Filter

Now considering the concept of Kalman filter in previous part, we will apply this algorithm to estimate time varying Volterra kernel. Consider the state space representation of the time varying Volterra system. Let $h(n)$ be the time varying Volterra kernel that is to be estimated and it is represented by Morkov model. The nonlinear channel model presented here is based on second order time varying Volterra series and it is used to describe the input-output relationship.

$$y(n) = h_0 + \sum_{k_1=0}^{M-1} h_1(n;k_1)x(n-k_1) + \sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1} h_2(n;k_1,k_2)x(n-k_1)x(n-k_2) + e(n) \tag{3.26}$$

$e(n)$ is the measurement noise with variance σ_e^2. The state space equation of Volterra system is represented by

$$\begin{aligned} h(n) &= \phi h(n-1) + w(n) \\ y(n) &= x^T(n)h(n) + e(n) \end{aligned} \tag{3.27}$$

Where

$$h(n) = [h_1(o;n), h_1(1;n), \ldots\ldots h_1(M-1;n), h_2(0,0;n), h_2(0,1;n), \ldots\ldots$$
$$h_2(0,M-1;n), h_2(1,1;n), \ldots\ldots h_2(M-1,M-1;n)]^T$$

And

$$x(n) = [x_1(n), x_1(n-1), \ldots\ldots x_1(n-M+1), x_1^{\,2}(n), x_1(n)x_1(n-1), \ldots\ldots x_1(n)x_1(n-M+1),$$
$$x_1^{\,2}(n-1), \ldots\ldots x_1^{\,2}(n-M+1)]^T$$

$w(n)$ is the process noise with variance σ_w^2. If we want to find the MMSE estimate of $h(n)$ given the measured data $y(n)$ and $x(n)$, the optimal estimate can be obtained by the Kalman filtering algorithm. Since (3.27) is in linear form, standard Kalman filtering can be applied to yield the following recursive estimation equations.

$$h'(0|0) = 0,\ P(0|0) = \mathrm{I} \tag{3.27a}$$

$$h'(n+1|n) = \Phi\, h'(n|n) \tag{3.27b}$$

$$P(n+1|n) = \Phi P(n|n)\Phi^T + Q_w \tag{3.27c}$$

$$K(n) = P(n+1|n)x(n)[x^T(n)P(n+1|n)x(n) + \sigma_e^2]^{-1} \quad (3.27d)$$

$$h'(n+1|n+1) = h'(n+1|n) + K(n)[y(n) - x^T(n)\, h'(n+1|n)] \quad (3.27e)$$

$$P(n+1|n+1) = [I - K(n)x^T(n)]P(n+1|n). \quad (3.27f)$$

Eq. (3.27a) initializes the kernel vector $h'(n|n)$ and its covariance matrix

$$P(n|n) = E\{[h'(n|n) - h(n)][h'(n|n) - h(n)]^T\}. \quad (3.27g)$$

Eq. (3.27b) calculates the prior estimate of $h(n)$ while (3.27c) gives the prior covariance matrix. Eq. (3.27d) calculates the Kalman gain and (3.27e) and (3.27f) update the posterior estimates for $h(n)$ and its covariance

3.5 Parameter Estimation of Second Order Time Varying Volterra System Using RLS Adaptive Algorithm

The Volterra filter of a fixed order and a fixed memory adapts to the unknown nonlinear system using one of the various adaptive algorithms. A simple and commonly used algorithm is based on the LMS adaptation criterion. Adaptive Volterra filters based on the LMS adaptation algorithm are computational simple but suffer from slow and input signal dependant convergence behaviour and hence are not useful in many applications. Here we discuss 2nd order case using RLS algorithm [42]. A typical adaptive technique is shown in Fig 3.3.

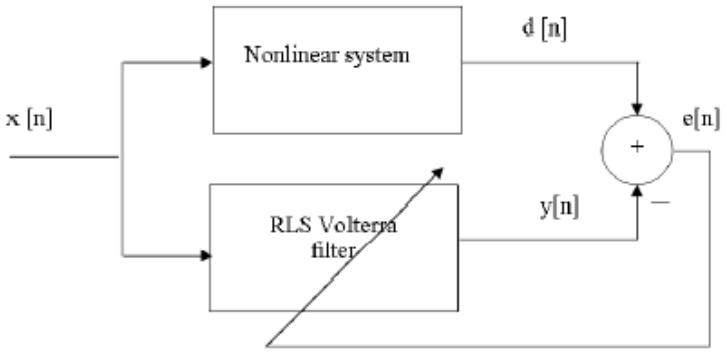

Fig.3.3.RLS Volterra identifier [44].

As in the case of second order LMS algorithm, the adaptive algorithm structure is different from that of the linear case only in the way in which the vectors are defined. It is relatively straightforward under some simplifying assumptions to show that the mean values of the Coefficients converge (for stationary environments) to their optimal values

if the convergence constant is chosen such that $0 < \mu_1, \mu_2 < 2/\lambda_{max}$ where λ_{max} is the maximum eigenvalue of the autocorrelation matrix of the input vector *x*[n]. The problem, as is for the linear case, is that the eigenvalue of the autocorrelation matrix control the speed of convergence. In general, the larger the eigenvalue spread (the ratio of the maximum and minimum eigenvalue), the slower is the convergence speed. This is particularly troublesome in the nonlinear filtering case, since the eigenvalue spreads are in general very large. Even when the input signal is white, the presence of the nonlinear entries in the input vector will cause the eigenvalue spread to be more than one. Consequently, it is important to seek alternate algorithms and structures that have convergence behaviours that are independent of or less dependent on the statistics of the input signal. One approach is to use recursive least squares (RLS) algorithms in place of the LMS adaptive filter. Another alternative is to use lattice (or other orthogonalized) structures to implement the nonlinear filters. The kernel estimation accuracy becomes the major problem in the practical applications. It is known that the Volterra operators are homogeneous and generally not orthogonal. As a consequence, the Volterra kernels cannot be measured using the cross correlation techniques and the values of the Volterra kernels will depend on the order of the Volterra representation being used. If the order of the Volterra model is changed the Volterra kernels will change and they must be recalculated. However, for an input having a symmetric amplitude density function, such as the Gaussian noise, the odd order Volterra functionals are orthogonal to the even order Volterra functionals. It follows that for this type of input, a 2nd order Volterra model, with zero DC component, is an orthogonal model. This leads to direct Volterra kernel measurement by the cross-correlation methods

The LMS adaptive filter can be considered as an approximate solution to the statistical optimization problem that tries to minimize the mean squared value of the estimation error at each time. RLS adaptive filters, on the other hand, yield the exact solution to an optimization problem formulated in a deterministic fashion. One such formulation gives rise to the exponentially weighted RLS adaptive filter and in the case of the second-order Volterra filter, such adaptive systems minimize the following cost function at each time.

$$J(n) = \sum_{k=o}^{n} \lambda^{n-k} (d(k) - h^{'}(n)^T x(k))^2 \qquad (3.28)$$

Where

$$h'(n) = [h_1'(o;n), h_1'(1;n), ...h_1'(M-1;n), h_1'(0,0;n), h_1'(0,1;n), ...$$
$$h_1'(0,M-1;n), h_1'(1,1;n), ...h_1'(M-1,M-1;n)]^T$$

The state space equation of time varying Volterra system is represented by

$$h(n) = \phi h(n-1) + w(n)$$
$$y(n) = x^T(n)h(n) + e(n) \qquad (3.27)$$

Where

$$h(n) = [h_1(o;n), h_1(1;n), ...h_1(M-1;n), h_2(0,0;n), h_2(0,1;n), ...$$
$$h_2(0,M-1;n), h_2(1,1;n), ...h_2(M-1,M-1;n)]^T$$

And

$$x(n) = [x_1(n), x_1(n-1), ...x_1(n-M+1), x_1^2(n), x_1(n)x_1(n-1), ...x_1(n)x_1(n-M+1),$$
$$x_1^2(n-1), ...x_1^2(n-M+1)]^T$$

are the second order time varying Volterra coefficient and input vector, respectively. h is a factor that controls the memory span of the adaptive filter. The solution to this problem at each time can be easily found by differentiating $J(n)$ with respect to $h'(n)$, setting the derivative to zero, and solving for $h(n)$. The optimal solution at time n is given by

$$h'(n) = P^{-1}(n)C(n) \qquad (3.29)$$

Where

$$P(n) = \sum_{k=o}^{n} \lambda^{n-k} x(k)x^T(k) \qquad (3.30)$$

And

$$P(n) = \sum_{k=o}^{n} \lambda^{n-k} x(k)x^T(k) \qquad (3.31)$$

$h'(n)$ can be recursively updated by realizing that

$$P(n) = \lambda P(n-1) + x(n)x^T(n) \qquad (3.32)$$

And

$$C(n) = \lambda C(n-1) + d(n)x(n)$$

One can simplify the computational complexity a little bit by making use of the matrix inversion lemma for inverting $P(n)$.

Now, We summarize the algorithm

Initialization:

$$h^{'}(0) = [0, 0, 0.....,0]^T \tag{3.33}$$

$$P^{-1}(0) = \delta^{-1} I \tag{3.34}$$

δ a small positive constant

Algorithm:

$$k(n) = \frac{\lambda^{-1} P^{-1}(n-1) x(n)}{1 + \lambda^{-1} x^T(n) P^{-1}(n-1) x(n)} \tag{3.35}$$

$$\varepsilon(n) = d(n) - h^{'}(n-1)^T x(n) \tag{3.36}$$

$d(n)$ is the desired response

$$h^{'}(n) = h^{'}(n-1) + \mu k(n) \varepsilon(n) \tag{3.37}$$

$$P^{-1}(n) = \lambda^{-1} P^{-1}(n-1) - \lambda^{-1} k(n) x^T(n) P^{-1}(n-1) \tag{3.38}$$

$$e(n) = d(n) - h^{'}(n)^T x(n). \tag{3.39}$$

3.6 Dynamic Forgetting Factor Recursive Least Square algorithm (DFF RLS)

The parameter estimation of time varying Volterra system using recursive least-squares (RLS) algorithm is one of the most popular adaptive algorithm [43]. As compared to the least-mean-square (LMS) algorithm, the RLS offers a superior convergence rate, especially for highly correlated input signals. The price to pay for this is an increase in the computational complexity. The RLS algorithm belongs to the Kalman filters family [44], and many adaptive algorithms (including the LMS) can be seen as approximations of it. The performance of RLS-type algorithms in terms of convergence rate, tracking, misadjustment, and stability depends on the forgetting factor. The classical RLS algorithm uses a constant forgetting factor ($0<\lambda\leq1$) and needs to compromise between the previous performance criteria. When the forgetting factor is very close to one, the algorithm achieves low misadjustment and good stability, but its tracking capabilities are reduced. A smaller value of the forgetting factor improves the tracking but increases the misadjustment, and it could affect the stability of the algorithm. Motivated by these aspects, a number of variable forgetting factor RLS (VFF-RLS) algorithms have been developed [45], [46]. The performance and the applicability of these methods for system

identification depend on several factors such as the ability of detecting the changes of the system, the level and the character of the noise that usually corrupts the output of the unknown system, and complexity and stability issues.

It should be mentioned that in the system identification context, when the output of the unknown system is corrupted by another signal (which is usually an additive noise), the goal of the adaptive filter is not to make the error signal goes to zero, because this will introduce noise in the adaptive filter. The objective instead is to recover the "corrupting signal" in the error signal of the adaptive filter after this one converges to the true solution. The mechanism that controls the forgetting factor is very simple and not expensive in terms of complexity. Moreover, the proposed algorithm has good tracking capabilities, a stable behaviour, and it is very robust against different variations of the "corrupting signal."

An important requirement of recursive estimators for adaptive control and adaptive signal processing lies in their ability to track parameter changes. From this viewpoint, the famous standard RLS algorithm which has been discussed in previous part, is known to have the optimal properties in stationary environments is unsuitable for nonstationary environments. Thus many attempts have been directed to the development of modified versions of the RLS algorithm to include tracking capability in time varying environments. Among these modified RLS algorithms, the best known is an exponential data weighting RLS algorithm using a forgetting factor. However, in certain situations, this algorithm can lead to a problem often referred to as the blow-up problem. Also the lower the value of the forgetting factor the higher the tracking velocity but the higher the influence of the noise, that is, the larger the parametric errors. To avoid these difficulties, the idea of a variable forgetting factor was introduced. We present a new exponentially weighted RLS (EWRLS) algorithm using a novel form of variable forgetting factor. The method presented has an excellent tracking adaptability with a low forgetting factor in the nonstationary situation and less error variance than other algorithms with a unity forgetting factor in stationary environments. It is the algorithm of which the control of the forgetting factor is much less sensitive to impulse noise but can response well to model variations. Unlike other algorithms, the control is based on the autocorrelation values of the error of nonzero lags and constrained by a sigmoidal function. This approach can reduce the affect of the impulse noise and make the change of the forgetting factor directly response to the model variations. Based on the mean square analysis, a control scheme is devised. In this thesis, we apply the new VFF scheme to the nonlinear RLS in

[8]. The additional computational complexity is negligible as compared to standard RLS by using simple operation.

This correspondence presents a method of estimating time-varying spectra of Volterra kernels using dynamic forgetting factors (DFF's). The concept of DFF was introduced in self-tuning control [47] to avoid a "blowing-up'' of the covariance matrix of the estimates and subsequent unstable control. A similar scheme has been used in adaptive filtering techniques. The method presented in this correspondence has better adaptability than the conventional algorithm with high fixed forgetting factor (FFF) in the nonstationary situation, and has lower variance than the conventional one with low FFF in the stationary situation. When the signal experiences nonstationarity, the method presented in next section can estimate the global trend quickly by decreasing the forgetting factor (FF) which is automatically accomplished by the proposed extended prediction error criterion. When the signal experiences stationarity, the actual memory length increases by increasing the FF, resulting in frequency estimation of the signal with high accuracy. A different adaptation scheme [48] has been proposed by varying the memory length.

Sources of Error in RLS Algorithm

The RLS algorithm may have four sources of errors. The first of these comes from noise, e.g., measurement noise. The second, estimation error, arises from finite data. This error has zero bias and a variance which decreases with window length. The third error, caused by nonstationarity, has a variance which increases with window length. This lag error could be minimized by rapidly discounting the past and basing estimates predominantly on recent data. Finally, model error could be contained in the prediction error.

Effect of Variation in the Value of Forgetting Factor

In the stationary case we can estimate the parameters with the Forgetting factor $\lambda = 1$. If each parameter of filter is calculated at every sample of the process using the same $\lambda < 1$ in the nonstationary case, the spectral estimation would not be statistically optimal. Given a small value of λ we can estimate the global trend of a nonstationary signal quickly at the expense of higher variance due to a smaller amount of available data. With a value of λ close to unity, it takes a relatively long time to estimate the correct parameter but eventually the parameters are estimated with high accuracy when the signal experiences stationarity. The speed of adaption is determined by the asymptotic memory length [49].

$$N = \frac{1}{1-\lambda} \tag{3.40}$$

which implies that the information dies away with N memory length. In the stationary case, the spectrum is estimated from an N memory length of the signal which is supposed to be stationary. If a stationary signal is composed of sub signals with different memory lengths varying between a minimum memory length, N_{min} and a maximum memory length N_{max}. the optimal time-varying spectrum of the signal is estimated by assigning the corresponding FF in (3.40) varying between $\lambda_{\max}$ and $\lambda_{\min}$ to each subsignal. However, it is not realistic in practice to know the memory lengths and starting points of the subsignal beforehand. Thus, we should estimate the degree of the nonstationarity of the signal to calculate the next value of the FF. The extended prediction error is defined by

$$Q_t = \frac{1}{M}\sum_{i=0}^{M-1} e_{t-i}^2 \tag{3.41}$$

Since the error coming from the noise is a random process, an appropriate averaging M of the error in (3.41) is helpful to minimize the effect of a spurious large additive noise error. However, M should be a small number compared with the minimum asymptotic memory length, N_{min} , so that the averaging does not obscure the nonstationary of the signal. The prediction error, which is the difference between the signal and model output, may have periodicity since the signal under consideration has periodicity. M can also be used to cancel out periodicity of the error.

A strategy for choosing the FF may now be defined by letting

$$\lambda_t = 1 - \frac{1}{N_t} \tag{3.42}$$

Where

$$N_t = \frac{\sigma_e^2 N_{\max}}{Q_t}$$

where σ_e^2 is the expected noise variance based on real knowledge of the process. The maximum asymptotic memory length N_{max}, then, will control the speed of adaptation. For a stationary process, the extended prediction error will approach the noise variance, which results in an N_{max} asymptotic memory length. Since this FF adaptation scheme does not guarantee that the λ_t does not become negative, a reasonable lower limit $\lambda_{\min}$ is placed on the FF.

3.6.1 Parameter Estimation of Second Order Time Varying Volterra System Using Dynamic Forgetting Factor Recursive Least Square (DFFRLS) Algorithm

In this thesis the nonlinear channel model is based on second order Volterra series and it is used to describe the input-output relationship

$$y(n) = h_0 + \sum_{k_1=0}^{M-1} h_1(n;k_1)x(n-k_1) + \sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1} h_2(n;k_1,k_2)x(n-k_1)x(n-k_2) + e(n) \quad (3.43)$$

the state space equation of Volterra system is represented by

$$h(n) = \phi h(n-1) + w(n) \quad (3.44)$$

and, from equation 3.43, for second order Volterra system, we have

$$y(n) = x^T(n)h(n) + e(n) \quad (3.45)$$

Where

$$h(n) = [h_1(o;n), h_1(1;n), \ldots\ldots h_1(M-1;n), h_2(0,0;n), h_2(0,1;n), \ldots\ldots$$
$$h_2(0,M-1;n), h_2(1,1;n), \ldots\ldots h_2(M-1,M-1;n)]^T$$

And

$$x(n) = [x_1(n), x_1(n-1), \ldots\ldots x_1(n-M+1), x_1{}^2(n), x_1(n)x_1(n-1), \ldots\ldots x_1(n)x_1(n-M+1),$$
$$x_1{}^2(n-1), \ldots\ldots x_1{}^2(n-M+1)]^T$$

Equation (3.43) and (3.44) represent the time varying Volterra system in terms of stochastic dynamic system which is governed by Gauss Morkov model. The estimated Volterra kernel is

$$h'(n) = [h'_1(o;n), h'_1(1;n), \ldots\ldots h'_1(M-1;n), h'_2(0,0;n), h'_2(0,1;n), \ldots\ldots$$
$$h'_2(0,M-1;n), h'_2(1,1;n), \ldots\ldots h'_2(M-1,M-1;n)]^T$$

Therefore the estimated signal is

$$y'(n) = x^T(n)h'(n) \quad (3.46)$$

The estimation error is $v(n) = y(n) - y'(n)$ with zero mean and variance σ_v^2.

$$v(n) = (h(n) - h'(n))^T x(n) + e(n) \quad (3.47)$$

Then,

$$v(n) = (\phi h(n-1) + w(n) - h^{'}(n))^{T} x(n) + e(n) \tag{3.48}$$

Under optimum condition

$$h^{'}(n) = \phi h(n-1) \tag{3.49}$$

Therefore, the equation leads to

$$v(n) \cong w^{T}(n) x(n) + e(n) \tag{3.50}$$

The estimation error variance is

$$\sigma_v^2 = \sigma_e^2 (1 + \frac{\sigma_w^2}{\sigma_e^2}) \tag{3.51}$$

For $\frac{\sigma_w^2}{\sigma_e^2} << 1$,

We have

$$\sigma_v^2 \cong \sigma_e^2 \tag{3.52}$$

From equation (3.52) process noise is almost equal to measurement noise. The polynomial channel model-based Dynamic forgetting factor RLS algorithm uses forgetting factor $\lambda(n)$ to update the channel state after each sample point. The proposed algorithm for second order Volterra filter is implemented as follows

Initialize the algorithm by setting

$$h^{'}(0) = 0 \tag{3.53}$$

$$P(0) = \delta^{-1} I \tag{3.54}$$

The extended estimation error is determined by

$$Z(n) = \frac{1}{K} \sum_{m=0}^{K-1} |v(n-m)|^2 \tag{3.55}$$

K must be kept smaller than minimum asymptotic memory length. If process noise σ_w^2 is very small in comparison to measurement noise that is $\sigma_w^2 < \sigma_e^2$, then using extended estimation error ,we have,

$$N(n) = \frac{\sigma_v^2 N_{\max}}{Z(n)} = \frac{\sigma_e^2 N_{\max}}{Z(n)} \tag{3.56}$$

Therefore the dynamic forgetting factor is

$$\lambda(n) = 1 - (N(n))^{-1} \tag{3.57}$$

Smaller value of forgetting factor in nonstationary environment seems more beneficial which is bounded by

$$\lambda_{\min} < \lambda(n) < \lambda_{\max}$$

$$\Pi(n) = P(n-1)x(n) \tag{3.58}$$

$$k(n) = \frac{\Pi(n)}{\lambda(n) + x^H(n)\Pi(n)} \tag{3.59}$$

$$\xi(n) = v(n) = y(n) - h^{'}(n-1)x(n) \tag{3.60}$$

$$h^{'}(n) = h^{'}(n-1) + k(n)\xi^*(n) \tag{3.61}$$

$$P(n) = \lambda^{-1}(n)P(n-1) - \lambda^{-1}(n)k(n)x^H(n)P(n-1) \tag{3.62}$$

Equation (3.61) and (3.62) gives the posterior estimate of $h(n)$ and its covariance . The value of forgetting factor vary between λ_{max} and λ_{min} . Another possible area where we can improve the estimation of Volterra kernel using variable forgetting factor when we increase the value of *K*.

CHAPTER 4

SIMULATION RESULTS

Simulation results shown below are verified using Matlab. This chapter is organized as follows.

Section 4.1 shows the tracking performance and MMSE for the first order Volterra filter using Kalman filter.

Section 4.2 shows the tracking performance and MMSE for the second order Volterra filter using Kalman filter.

Section 4.3 shows the tracking performance and MMSE for the first order Volterra filter using RLS algorithm.

Section 4.4 shows the tracking performance and MMSE for the second order Volterra filter using RLS algorithm

Section 4.5 shows the tracking performance and MMSE for the first order Volterra filter using Dynamic forgetting factor RLS.(DFFRLS) algorithm

Section 4.6 shows the tracking performance and MMSE for the second order Volterra filter using Dynamic forgetting factor RLS.(DFFRLS) algorithm .

In the simulation results mentioned below ,the system under consideration is second order time varying Volterra system that consist of linear and quadratic kernel for first and second order Volterra system respectively with memory length N=3. Volterra filter based on the decomposition of the input signal and on symmetric kernels assumption, for that purpose we have introduced the concepts of “second order input vector”. The nonlinear adaptive filter performances were evaluated in a typical system identification application and compared with the performances other adaptive algorithms. The costs of these performances are paid by the computational complexity required by the nonlinear adaptive Volterra filter implementation .

The simulation results shown here will present the tracking performance and MMSE of all the three Volterra kernels for first order Volterra system and six Volterra kernels for second order Volterra system. The results presented here are ensemble average of 1000 independent simulation runs. The actual Volterra kernel is denoted as “True” in Figures.

Time varying Volterra kernel follows the Gauss Morkov model, the state transition matrix of the linear kernel is assumed to be

$$\phi = \begin{matrix} -1.31 & -0.43 & 2.25 \\ -0.88 & -0.53 & 2.03 \\ -0.53 & -1.43 & 2.65 \end{matrix}$$

The state transition matrix of quadratic kernel is assumed to be

$$\phi = \begin{matrix} 2.71 & -7.19 & -0.04 & 0.72 & -0.82 & 4.27 \\ 1.6 & -4.19 & -0.09 & 0.36 & -0.52 & 2.67 \\ -0.39 & 1.27 & 0.17 & -0.57 & -0.47 & 0.25 \\ 1.99 & -5.67 & -0.19 & 0.84 & -0.11 & 2.81 \\ 2.38 & -5.27 & -0.26 & 0.77 & -0.91 & 3.02 \\ 0.42 & -0.46 & -0.15 & -0.26 & -0.58 & 1.14 \end{matrix}$$

The choice of ϕ makes stable matrices ensuring the stationarity of $h(n)$,and input should be random signal. The process noise covariance matrix is assumed to be $Q_w = \sigma_w^2 I$ where σ_w^2=0.01. The measurement noise is σ_e^2=10^{-5} . The minimum mean square error (MMSE) is given by

$\varepsilon = [| h(n) - h^{'}(n) |^2]$

In section 4.3 and 4.4 , the fixed forgetting factor λ and δ is chosen to be 0.975 and 10^{-5} respectively.

In section 4.5 and 4.6 the polynomial channel model-based dynamic forgetting factor RLS algorithm uses forgetting factor λ (n) to update the channel state after each sample point. The time varying nature of the signal is estimated by using variable forgetting factor between λ_{min} and λ_{max}. Where λ_{min} and λ_{max} are empirically chosen as 0.975 and 0.75 respectively. In the same section the value of averaging factor K is chosen to be 5. The value of averaging factor K can be increased for performance.

In the subsequent section, the tracking performance and MMSE performance of adaptive algorithms for first and second order Volterra filter are shown. All the simulations are performed for time varying environment.

4.1 Performance of Kalman Filter for First Order Time-Varying Volterra System

4.1.1. Tracking Performance and MMSE of First Coefficient of First Order Time-Varying Volterra System Using Kalman Filter

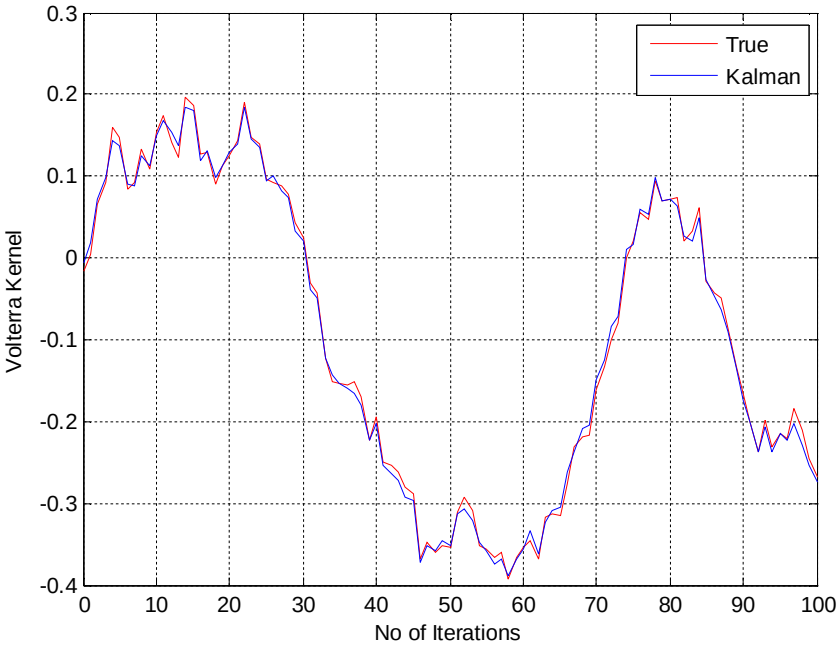

Fig.4.1.1.1. Tracking Performance.

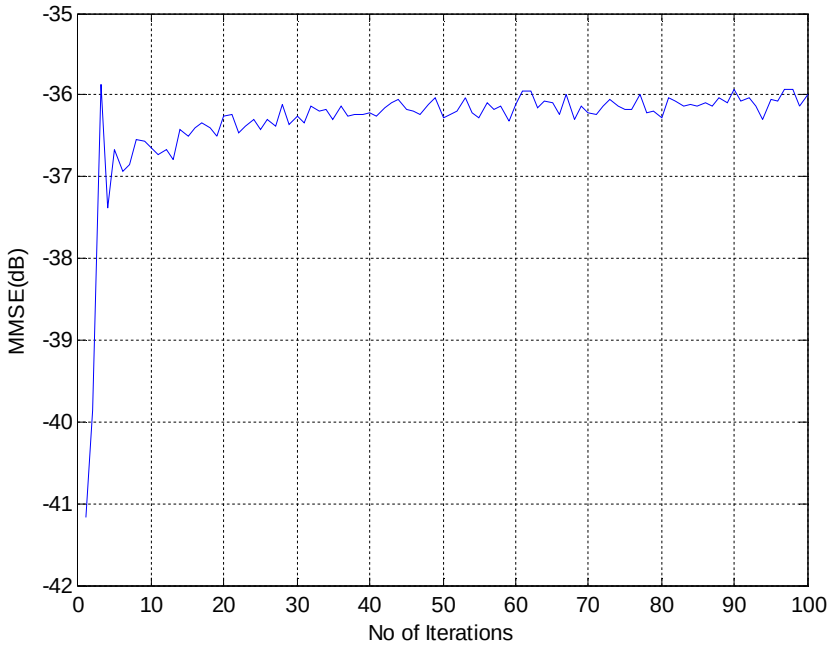

Fig. 4.1.1.2. MMSE Performance.

4.1.2. Tracking Performance and MMSE of Second Coefficient of First Order Time-Varying Volterra System Using Kalman Filter

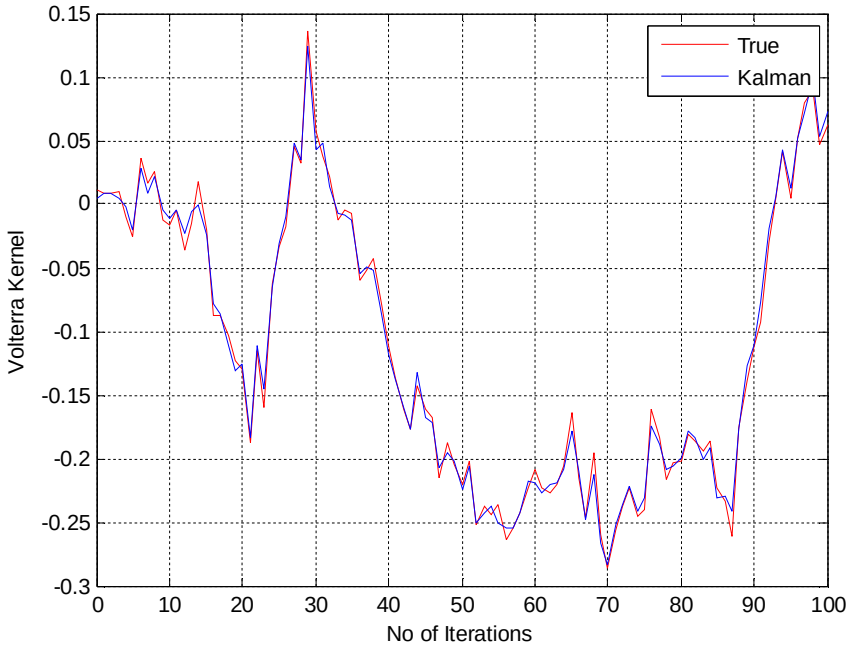

Fig.4.1.2.3. Tracking Performance.

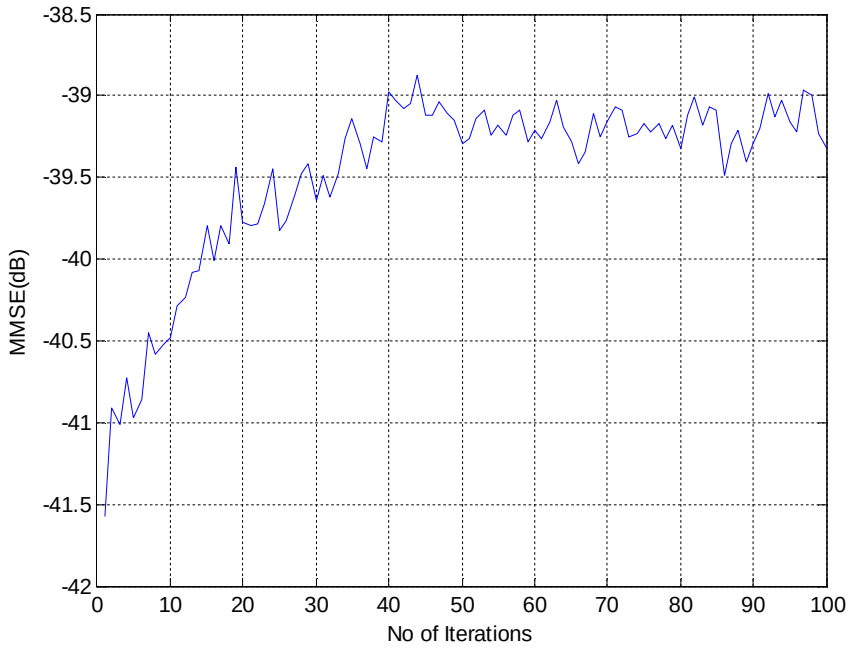

Fig.4.1.2.4. MMSE Performance.

4.1.3. Tracking Performance and MMSE of Third Coefficient of First Order Time-Varying Volterra System Using Kalman Filter

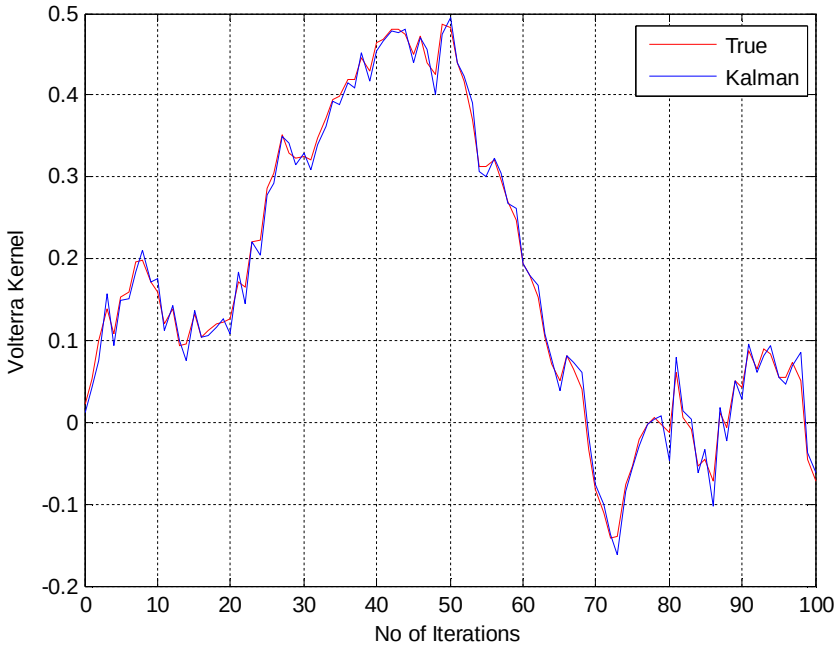

Fig.4.1.3.5. Tracking Performance.

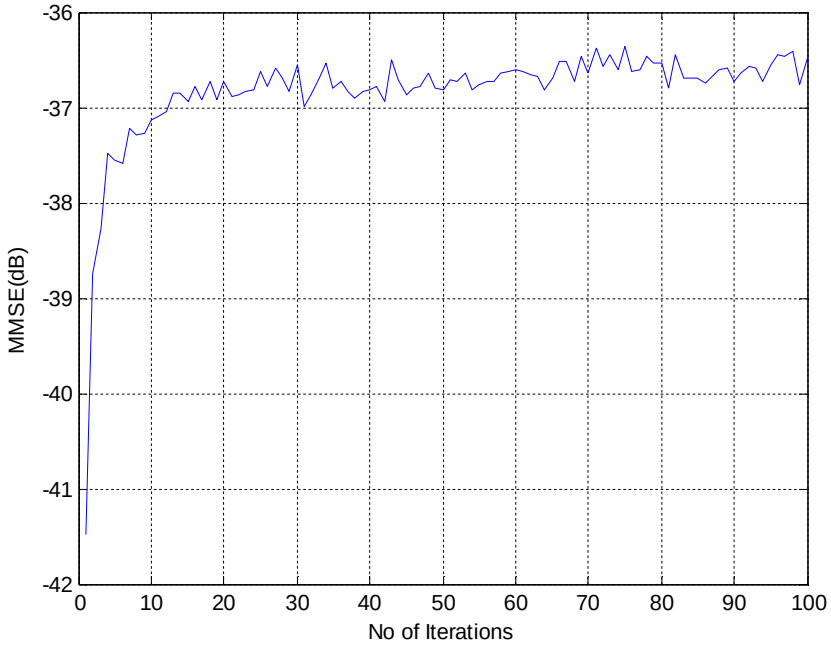

Fig.4.1.3.6.MMSE Performance.

4.2. Performance of Kalman Filter for Second Order Volterra System

4.2.1. Tracking Performance and MMSE of First Coefficient of Second Order Time-Varying Volterra System Using Kalman Filter

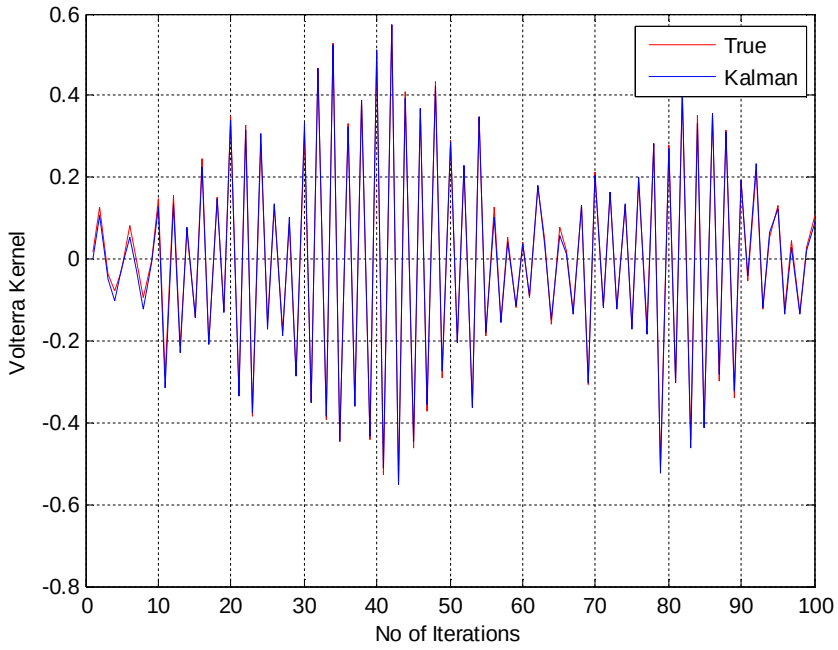

Fig.4.2.1.7. Tracking Performance.

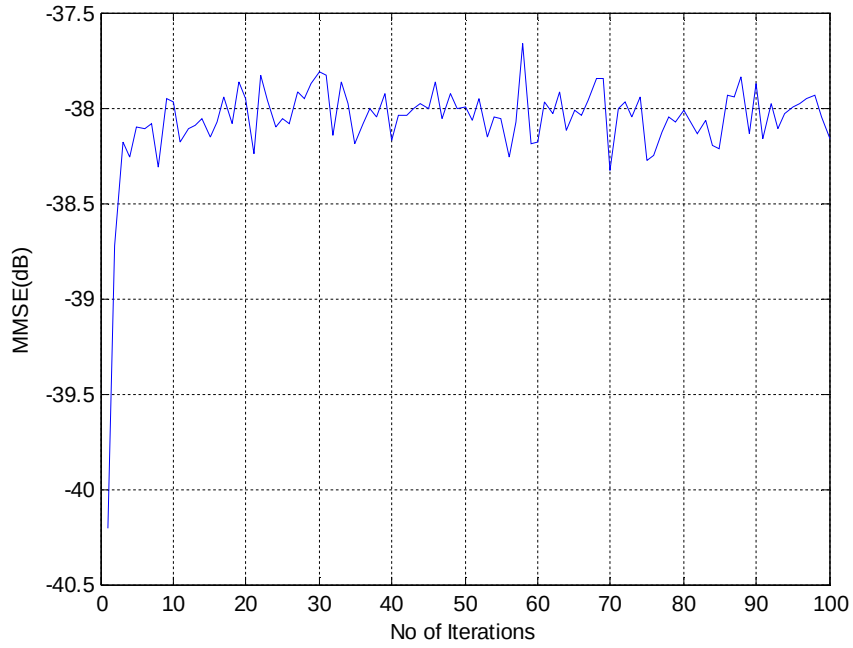

Fig.4.2.1.8. MMSE Performance.

4.2.2. Tracking Performance and MMSE of Second Coefficient of Second Order Time-Varying Volterra system Using Kalman Filter

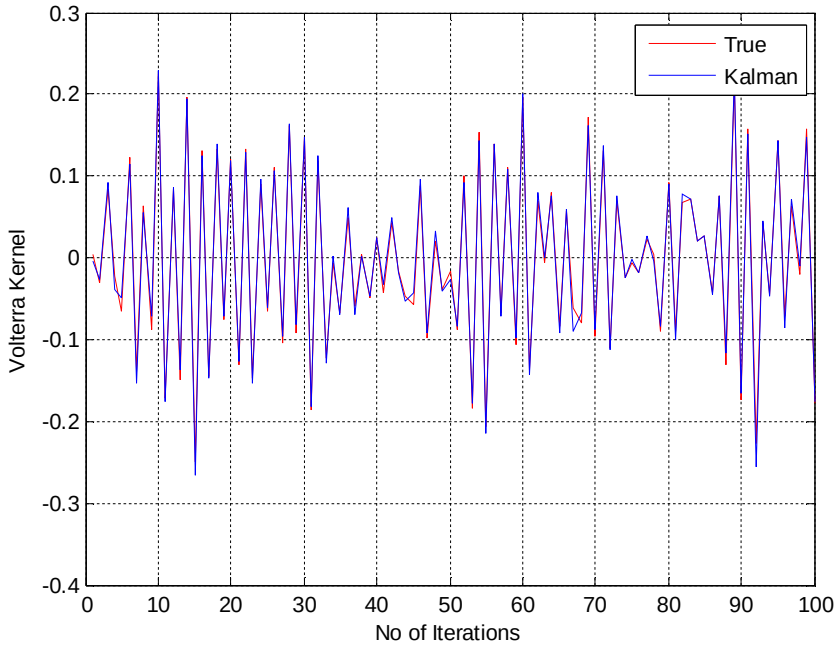

Fig 4.2.2.9 Tracking Performance

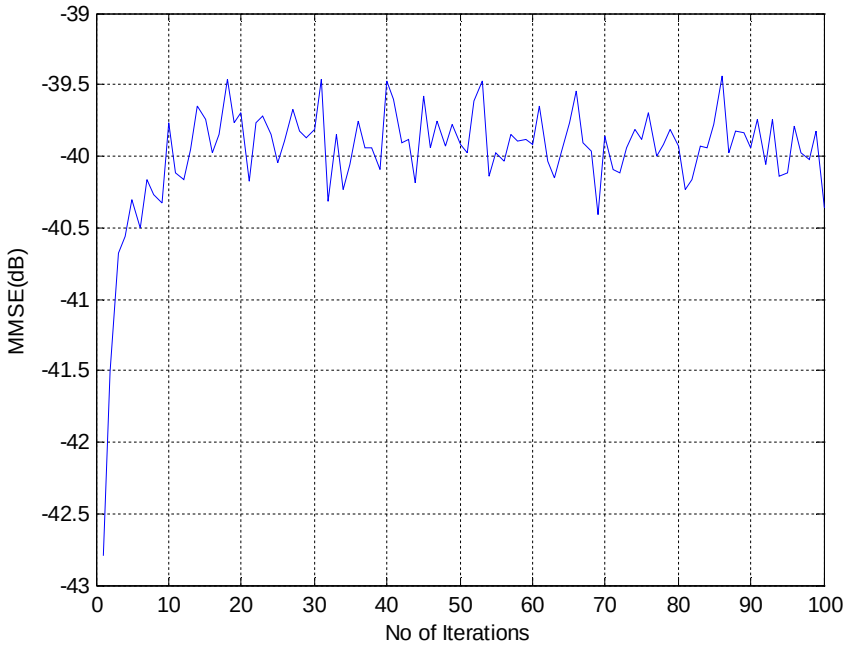

Fig.4.2.2.10. MMSE Performance.

4.2.3. Tracking Performance and MMSE of Third Coefficient of Second Order Time-Varying Volterra System Using Kalman Filter

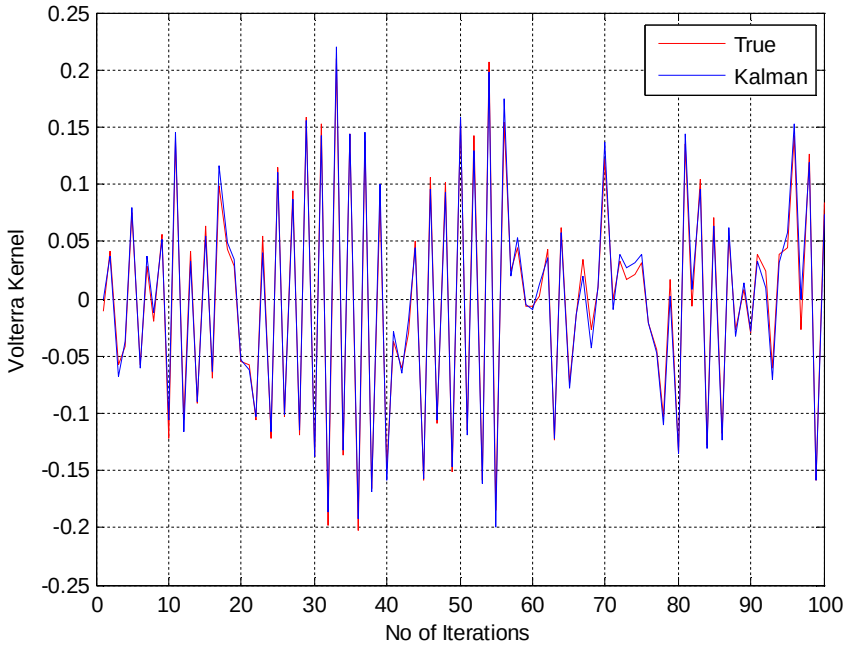

Fig 4.2.3.11 Tracking Performance

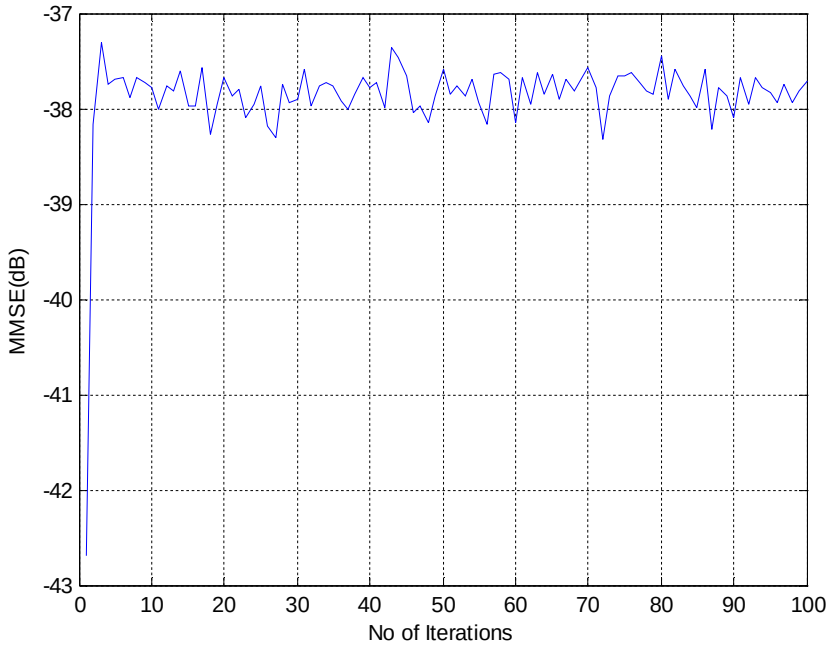

Fig.4.2.3.12. MMSE Performance.

4.2.4. Tracking Performance and MMSE af Fourth Coefficient of Second Order Time-Varying Volterra System Using Kalman Filter

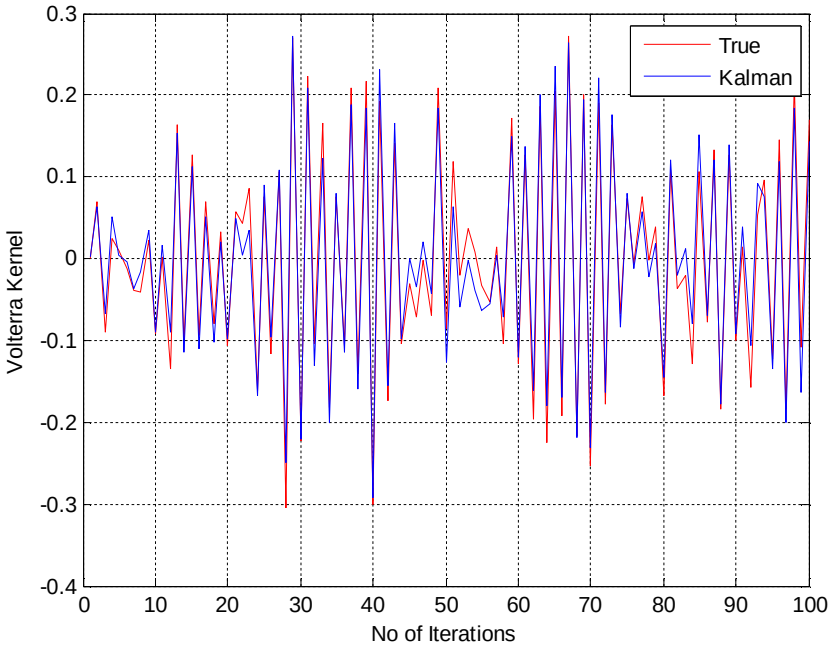

Fig.4.2.4.13. Tracking Performance.

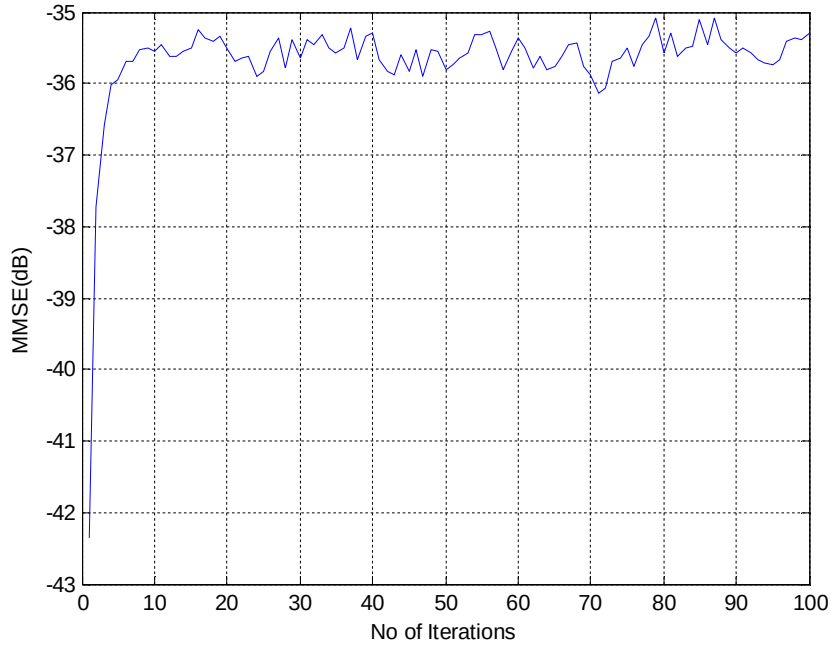

Fig.4.2.4.14. MMSE Performance.

4.2.5. Tracking Performance and MMSE of Fifth Coefficient of Second Order Time-Varying Volterra System Using Kalman Filter

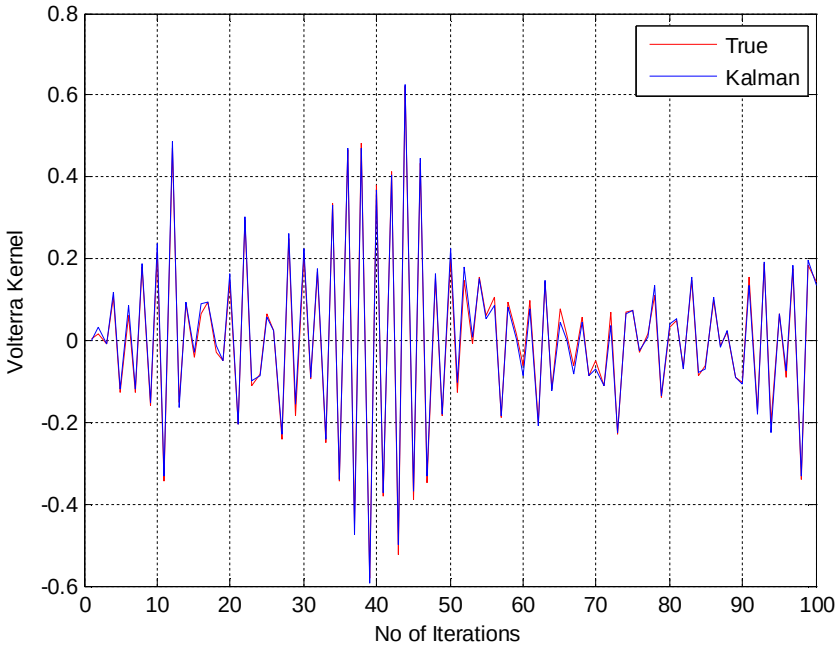

Fig.4.2.5.15. Tracking Performance.

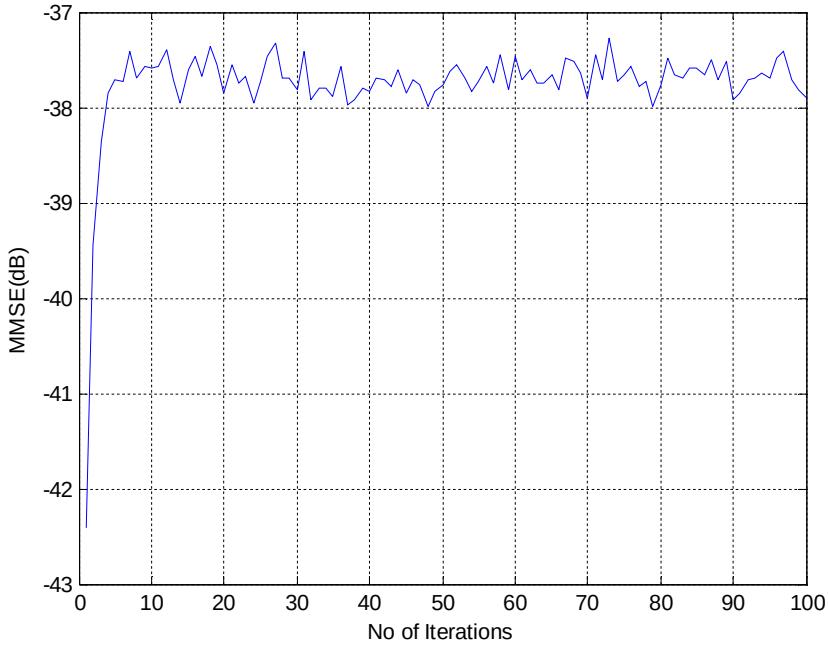

Fig 4.2.5.16 MMSE Performance

4.2.6. Tracking Performance and MMSE of Sixth Coefficient of Second Order Time-Varying Volterra System Using Kalman Filter

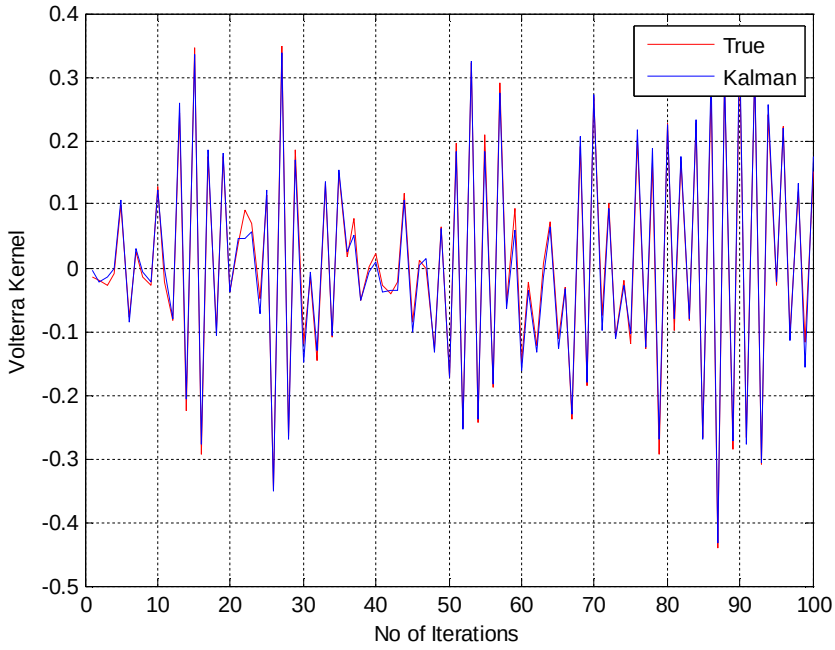

Fig.4.2.6.17. Tracking Performance.

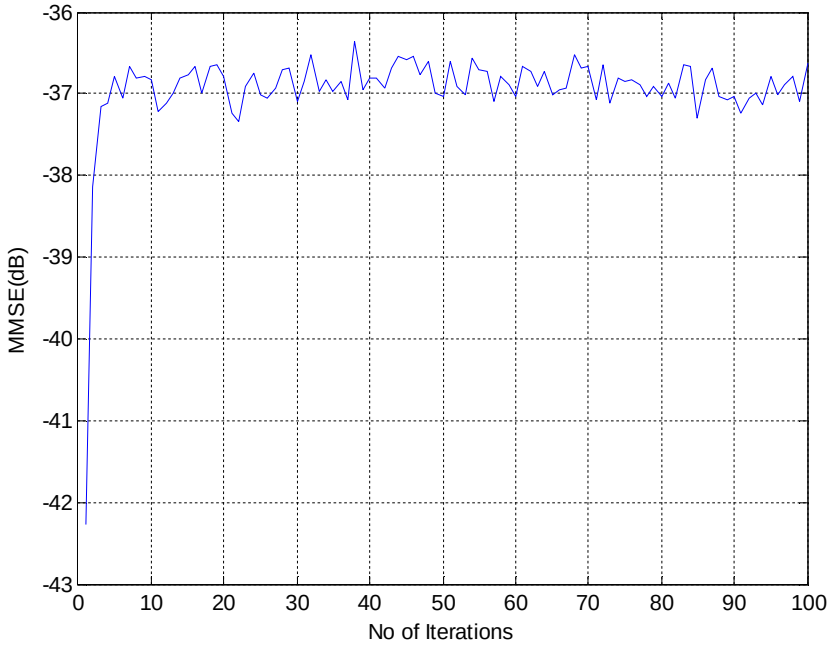

Fig.4.2.6.18. MMSE Performance.

4.3. Performance of RLS Algorithm for First Order Volterra System

4.3.1. Tracking Performance and MMSE of First Coefficient of First Order Time-Varying Volterra System Using RLS Algorithm

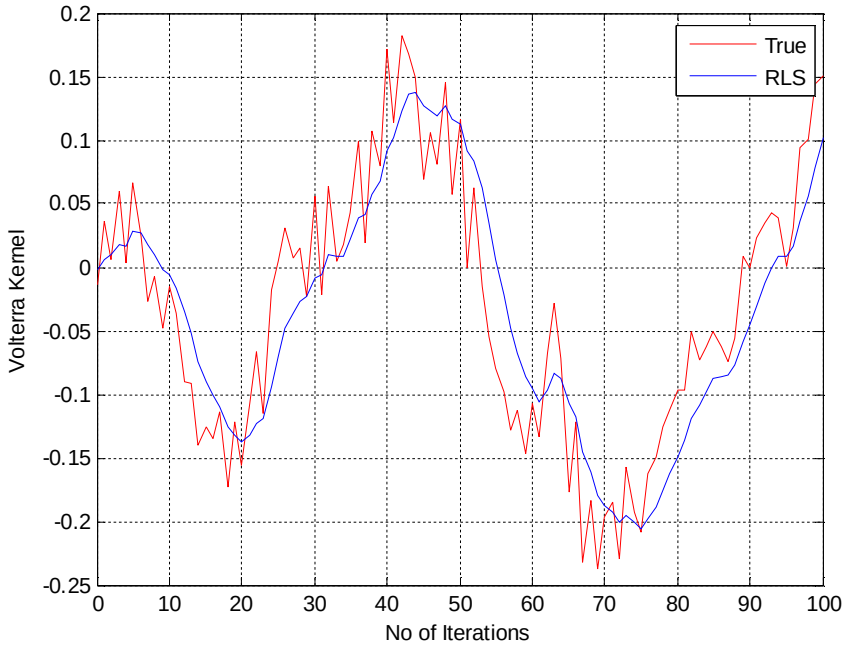

Fig.4.3.1.19. Tracking Performance.

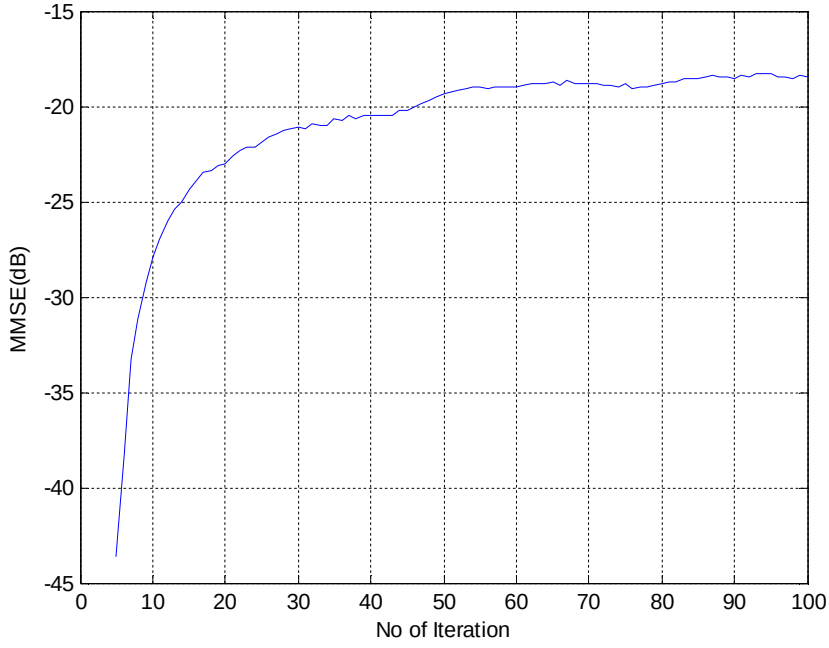

Fig.4.3.1.20. MMSE Performance.

4.3.2. Tracking Performance and MMSE of Second Coefficient of First Order Time-Varying Volterra System Using RLS Algorithm

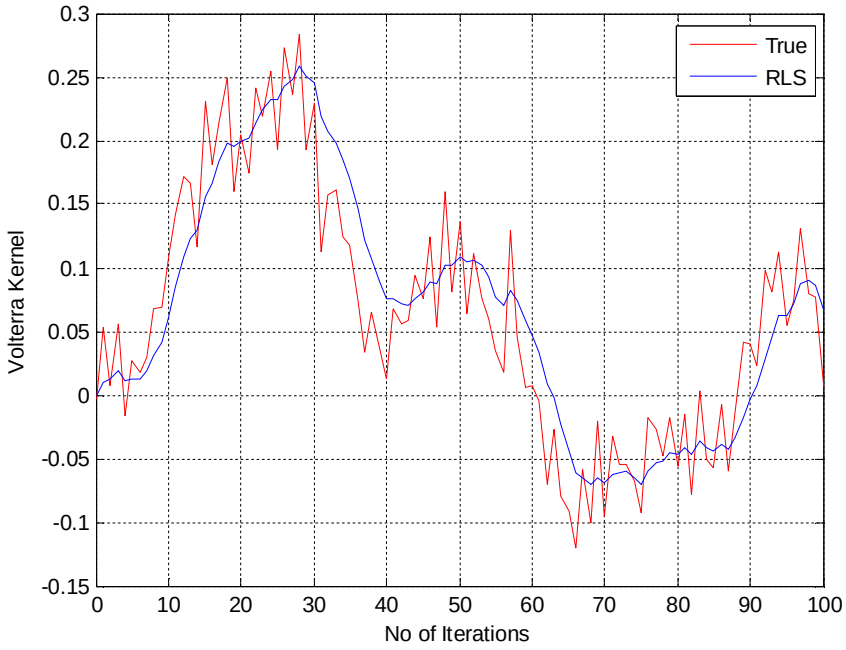

Fig.4.3.2.21. Tracking Performance.

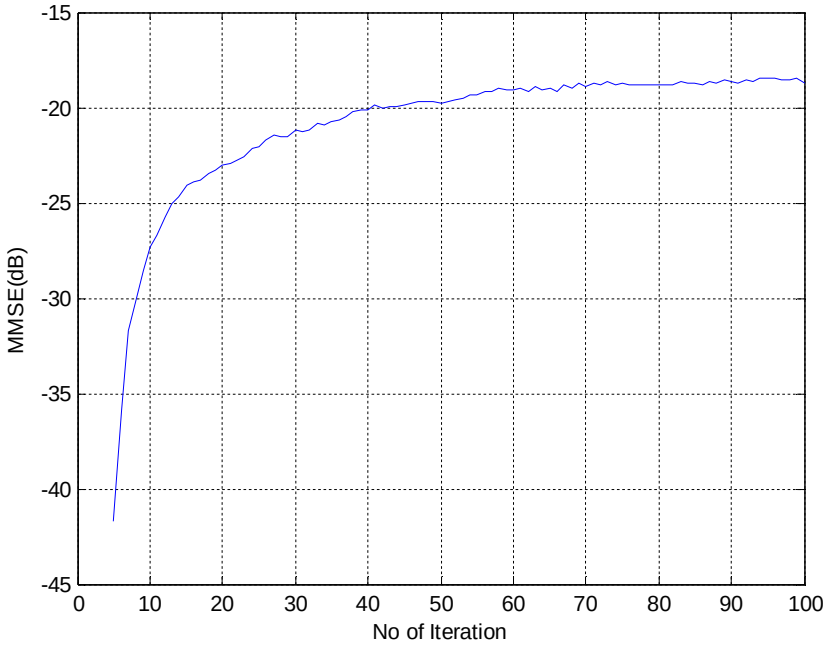

Fig.4.3.2.22. MMSE Performance.

4.3.3. Tracking Performance and MMSE of Third Coefficient of First Order Time-Varying Volterra System Using RLS Algorithm

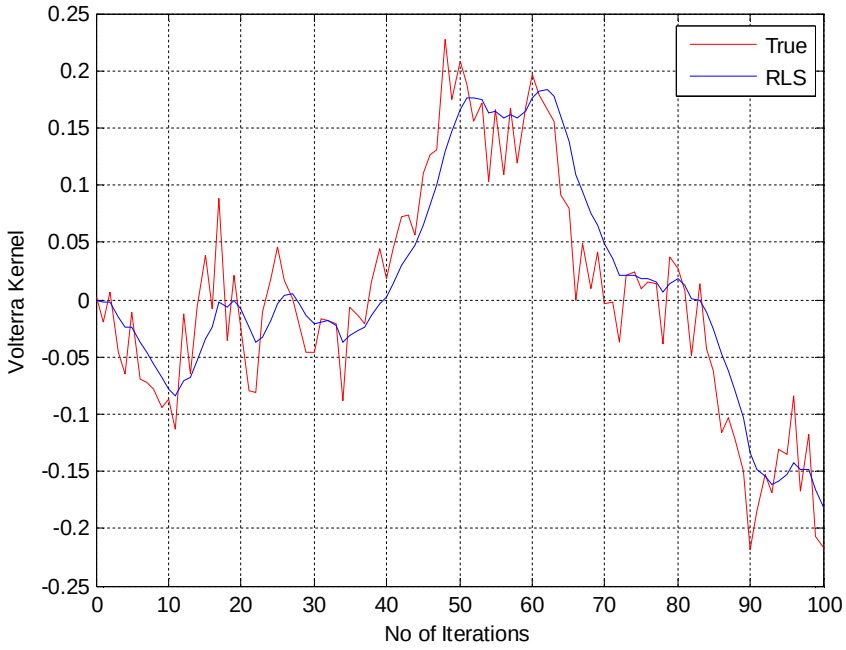

Fig.4.3.3.23. Tracking Performance.

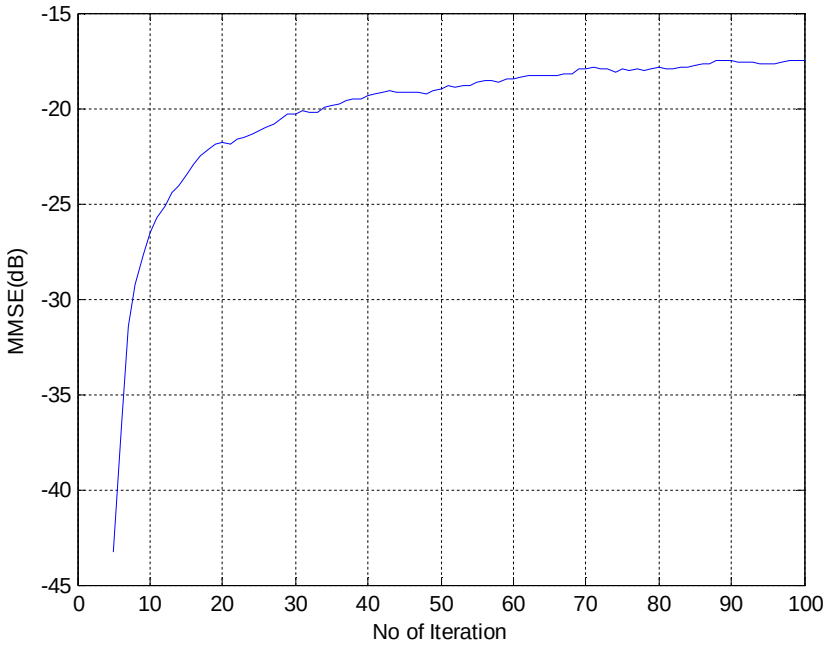

Fig.4.3.3.24. MMSE Performance.

4.4. Performance of RLS Algorithm for Second Order Volterra System

4.4.1. Tracking Performance and MMSE of First Coefficient of Second Order Time-Varying Volterra System Using RLS Algorithm

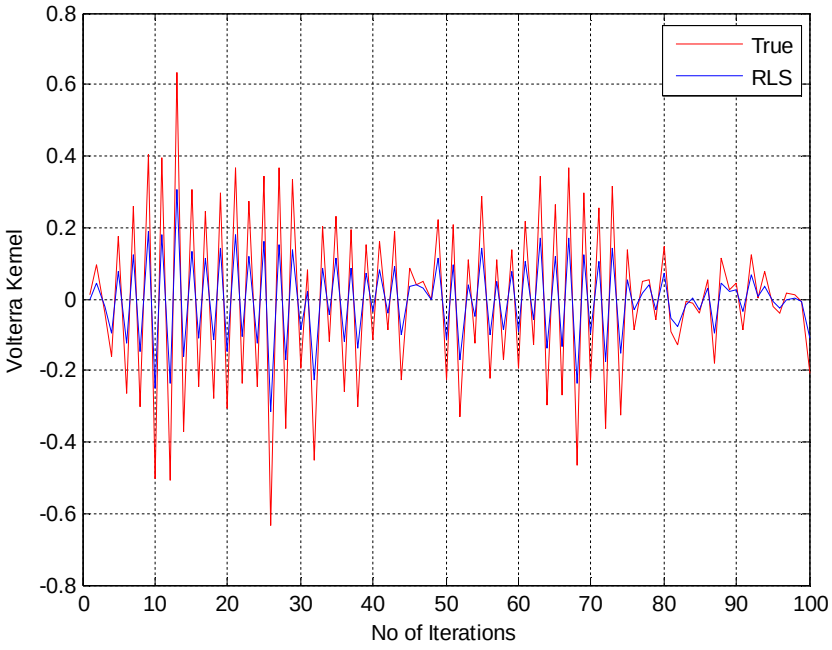

Fig.4.4.1.25. Tracking Performance.

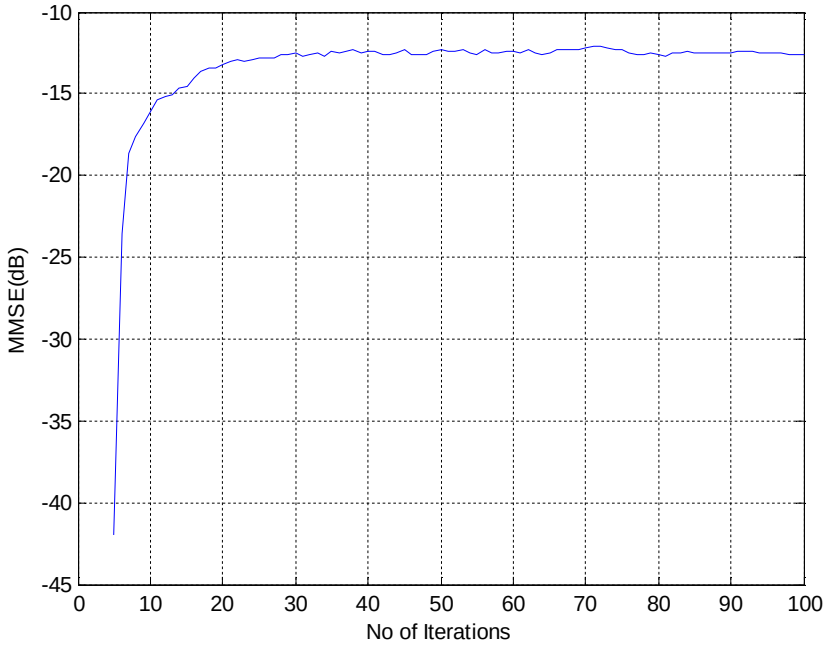

Fig.4.4.1.26. MMSE Performance.

4.4.2. Tracking Performance and MMSE of Second Coefficient of Second Order Time-Varying Volterra System Using RLS Algorithm

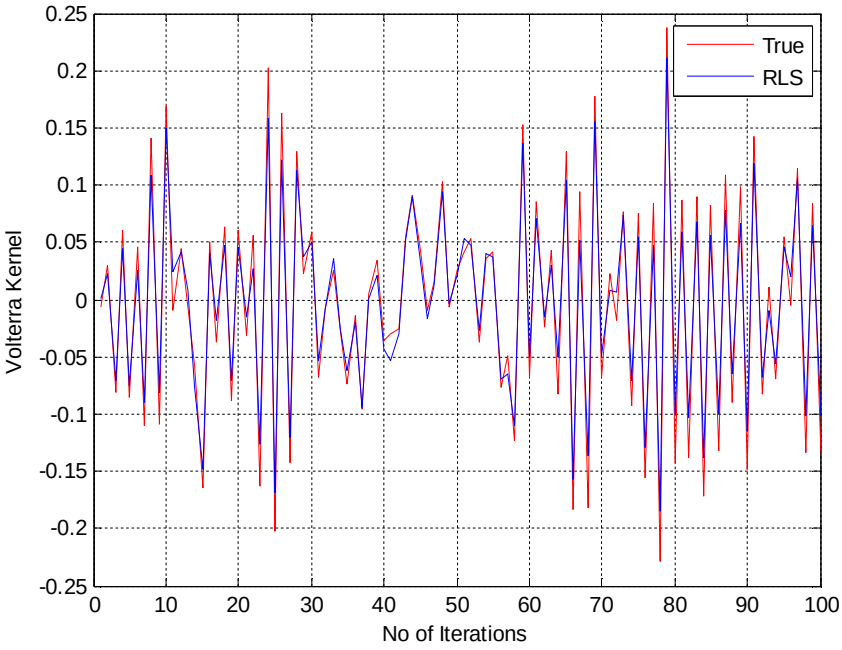

Fig.4.4.2.27. Tracking Performance.

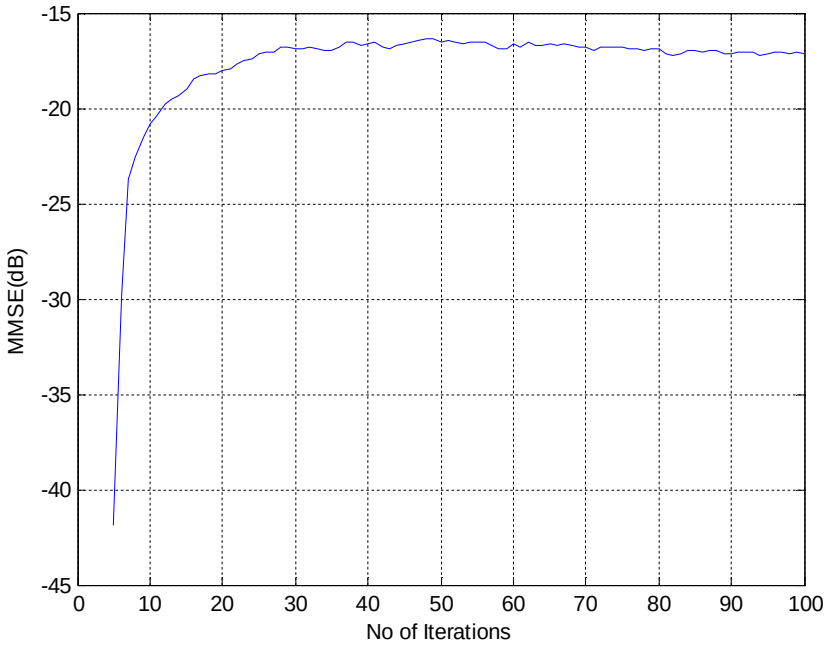

Fig 4.4.2.28 MMSE Performance

4.4.3. Tracking Performance and MMSE of Third Coefficient of Second Order Time-Varying Volterra System Using RLS Algorithm

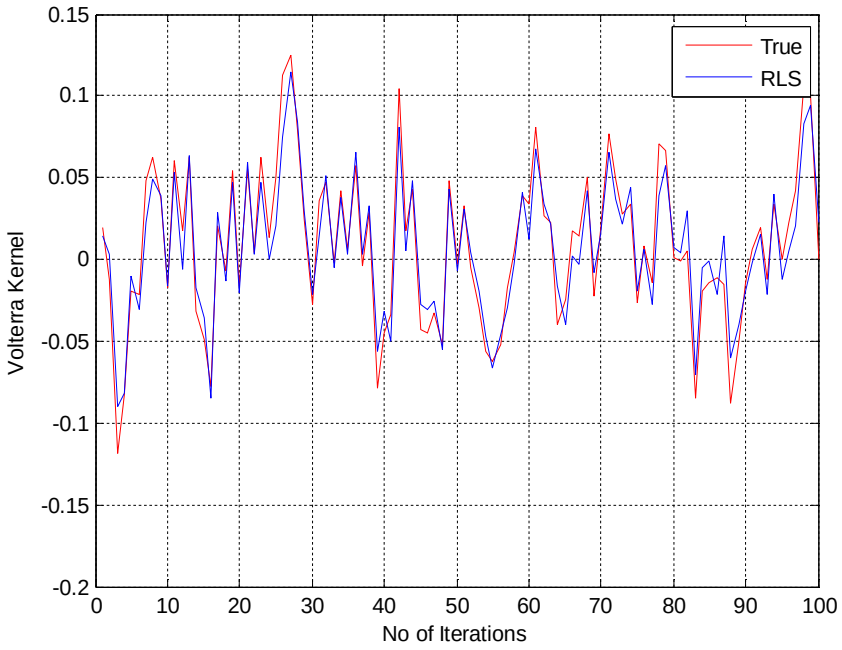

Fig.4.4.3.29. Tracking Performance.

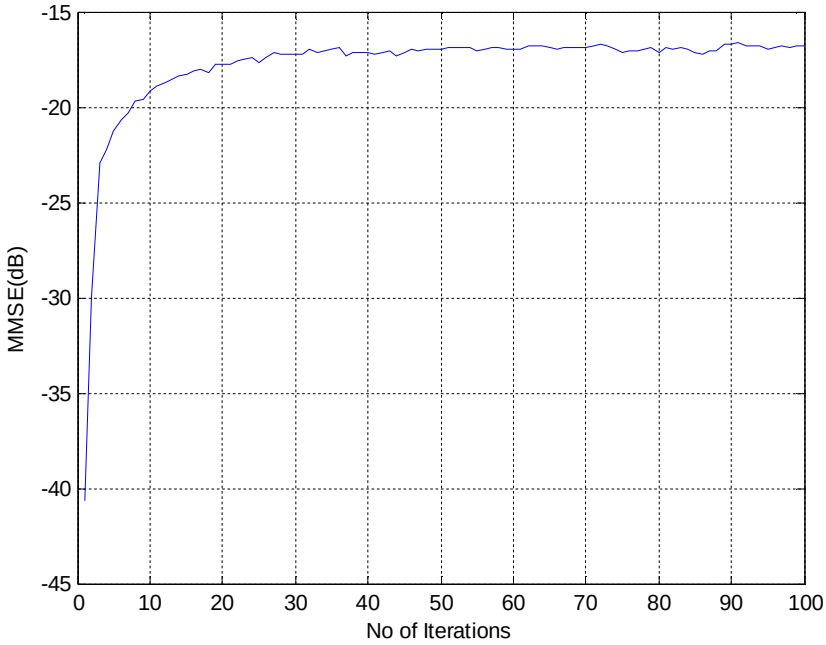

Fig.4.4.3.30. MMSE Performance.

4.4.4. Tracking Performance and MMSE of Fourth Coefficient of Second Order Time-Varying Volterra System Using RLS Algorithm

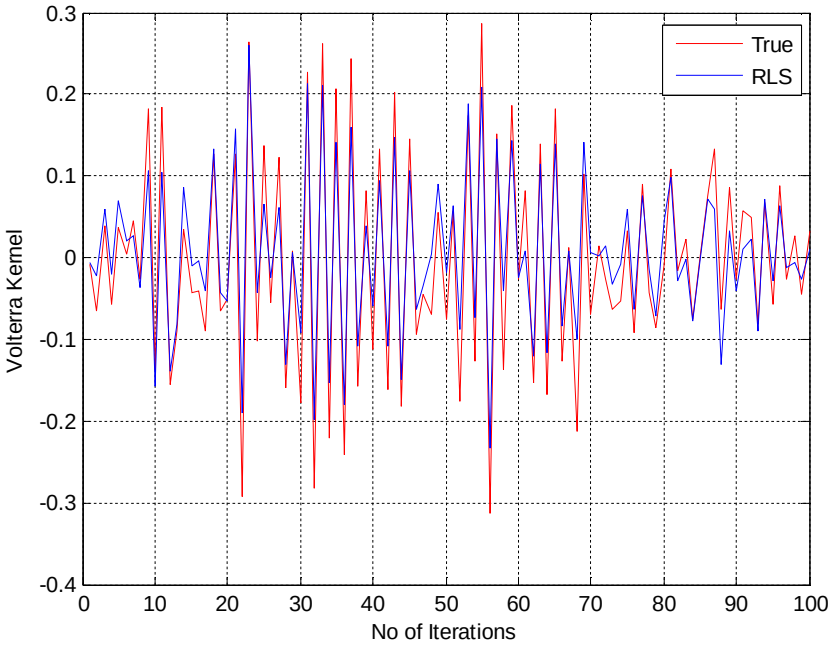

Fig.4.4.4.31. Tracking Performance.

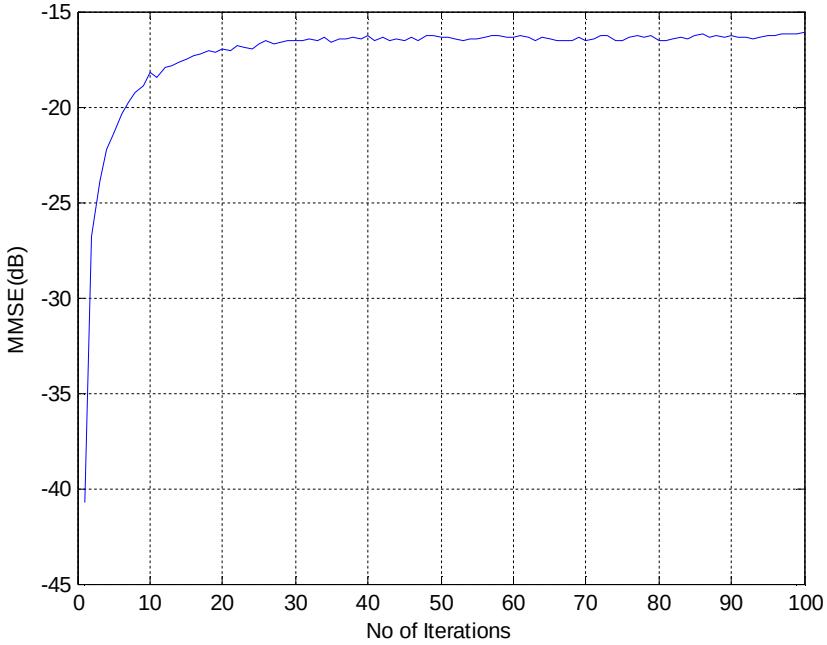

Fig.4.4.4.32. MMSE Performance.

4.4.5. Tracking Performance and MMSE of Fifth Coefficient of Second Order Time-Varying Volterra Filter System Using RLS Algorithm

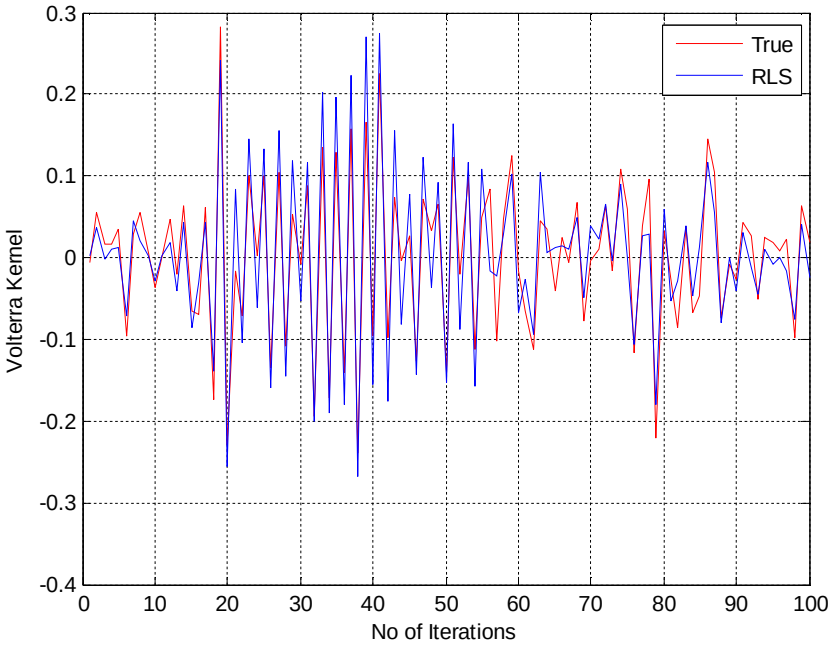

Fig.4.4.5.33. Tracking Performance.

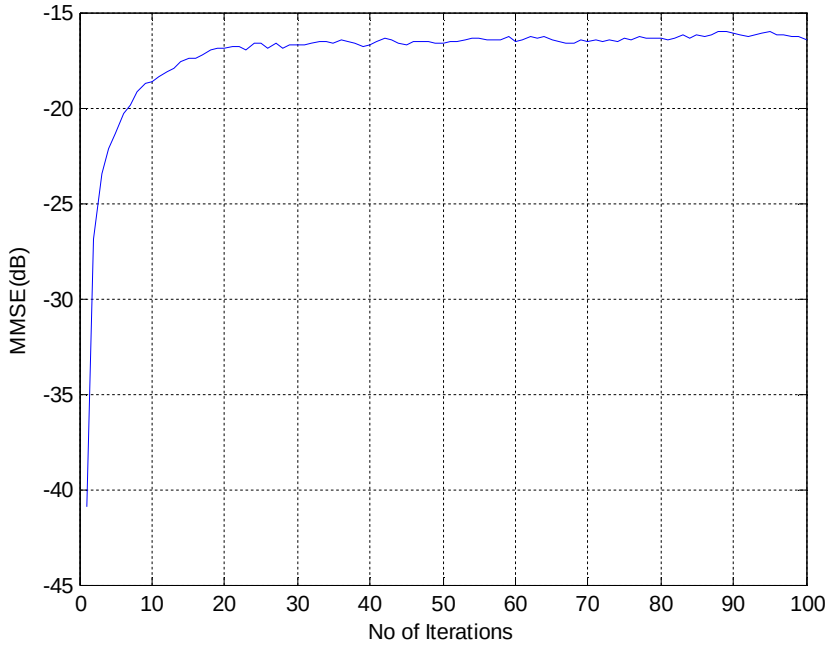

Fig.4.4.5.34. MMSE Performance.

4.4.6. Tracking Performance and MMSE of Sixth Coefficient of Second Order Time-Varying Volterra System Using RLS Algorithm

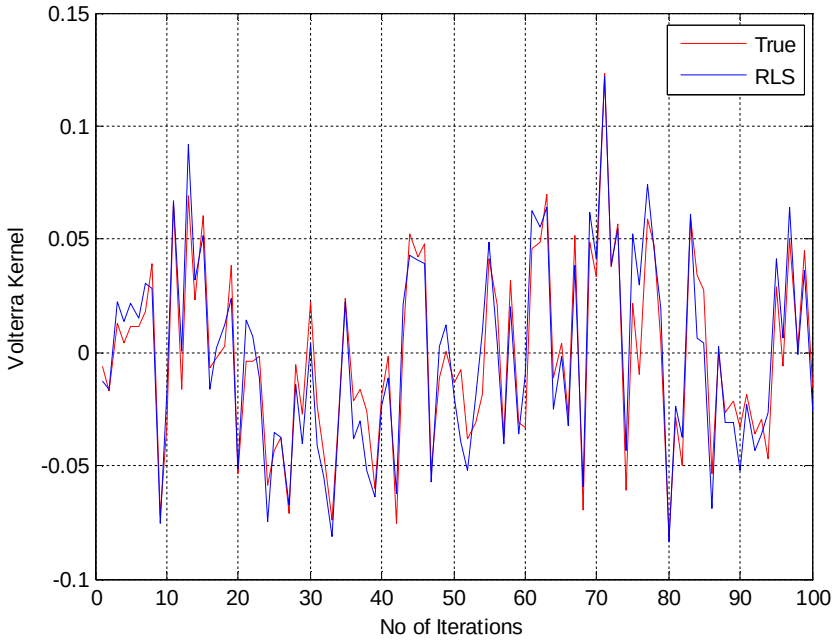

Fig.4.4.6.35. Tracking Performance.

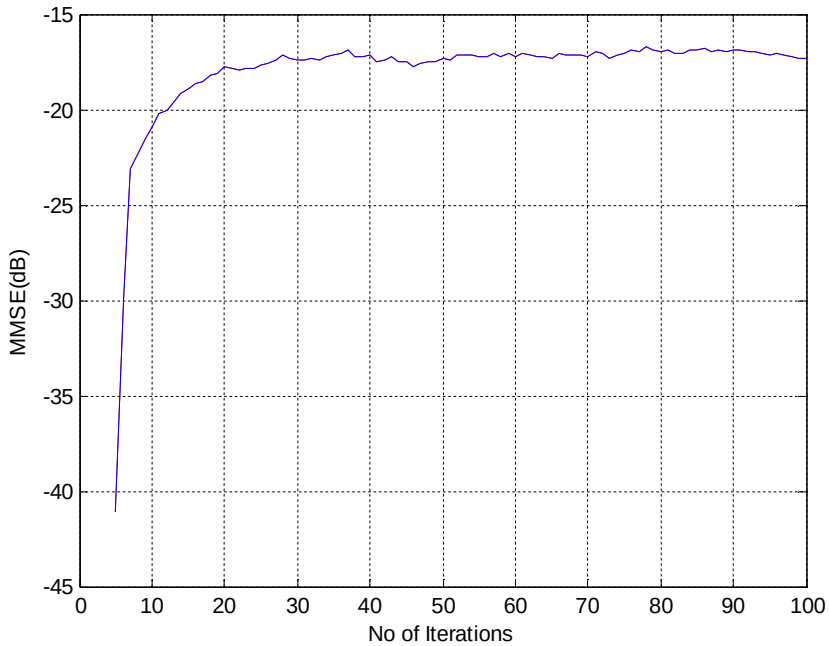

Fig.4.4.6.36. MMSE Performance.

4.5. Performance of DFF RLS Algorithm for First Order Volterra System.

4.5.1. Tracking Performance and MMSE of First Coefficient of Volterra System Using DFFRLS Algorithm

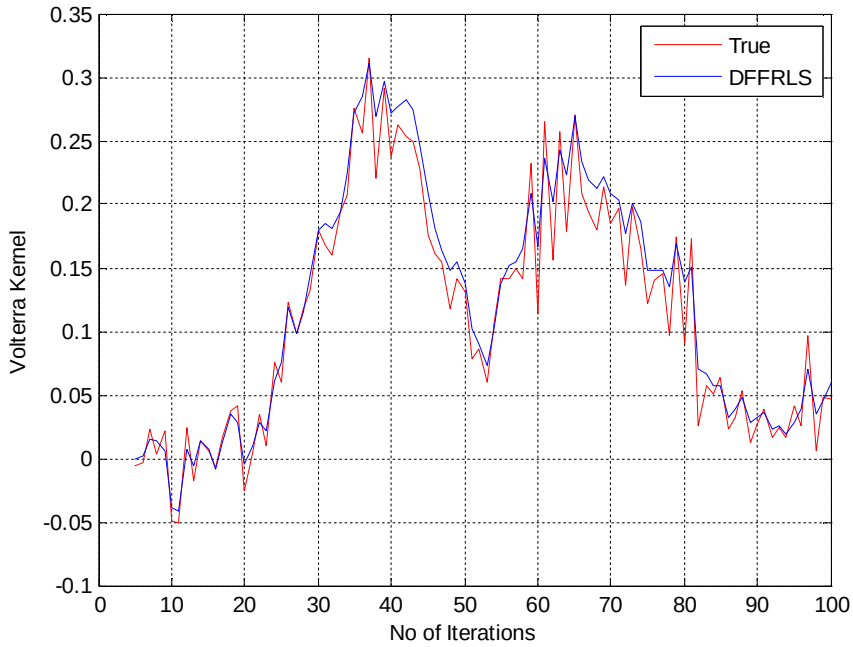

Fig.4.5.1.37. Tracking Performance.

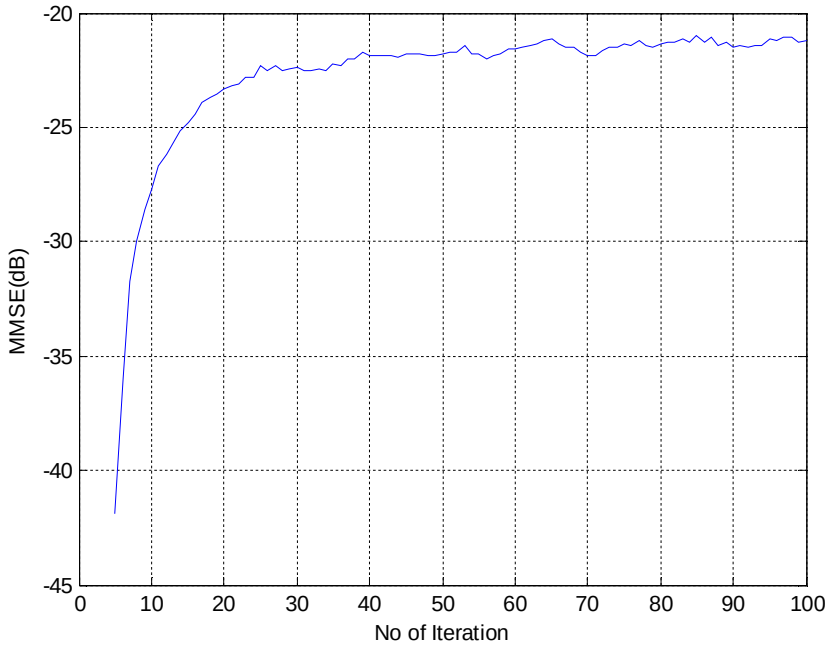

Fig. 4.5.1.38. MMSE Performance.

4.5.2. Tracking Performance and MMSE of Second Coefficient of First Order Time-Varying Volterra System Using DFFRLS Algorithm

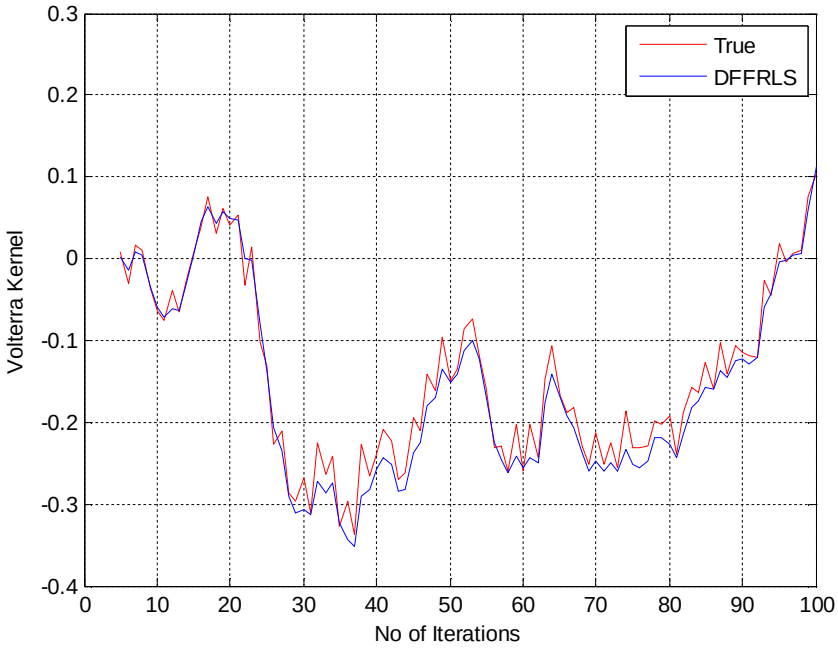

Fig.4.5.2.39. Tracking Performance.

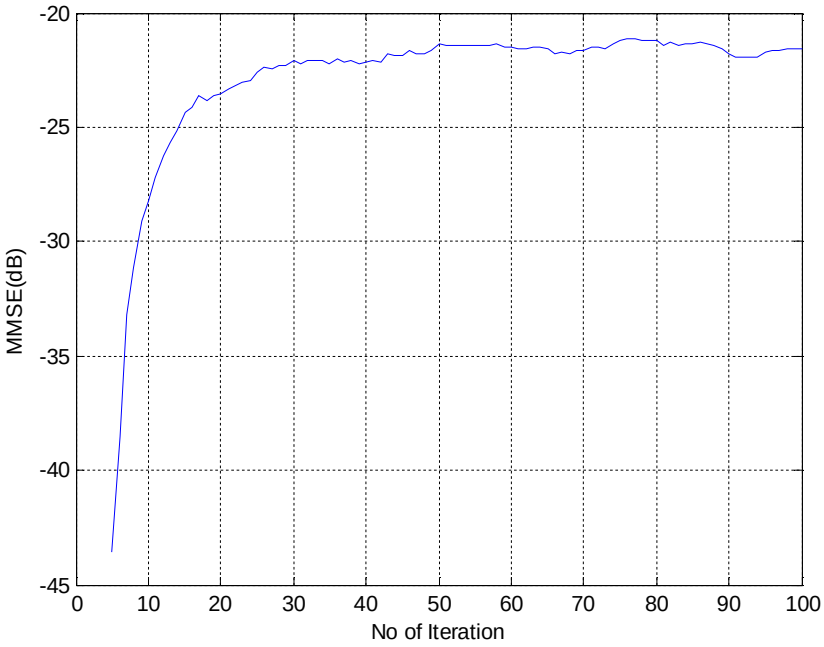

Fig.4.5.2.40. MMSE Performance.

4.5.3. Tracking Performance and MMSE of Third Coefficient of First Order Time-Varying Volterra System using DFFRLS Algorithm

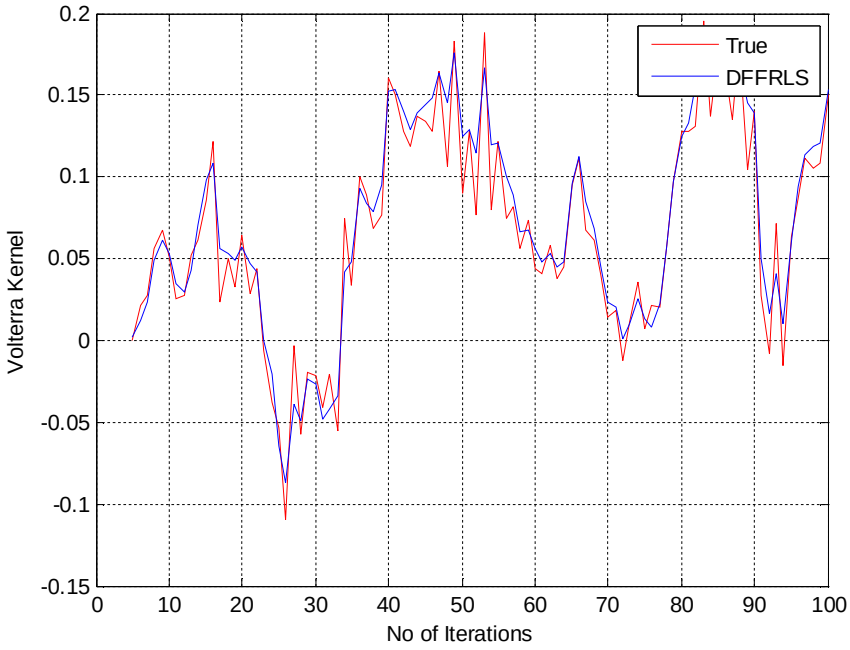

Fig.4.5.3.4.Tracking Performance.

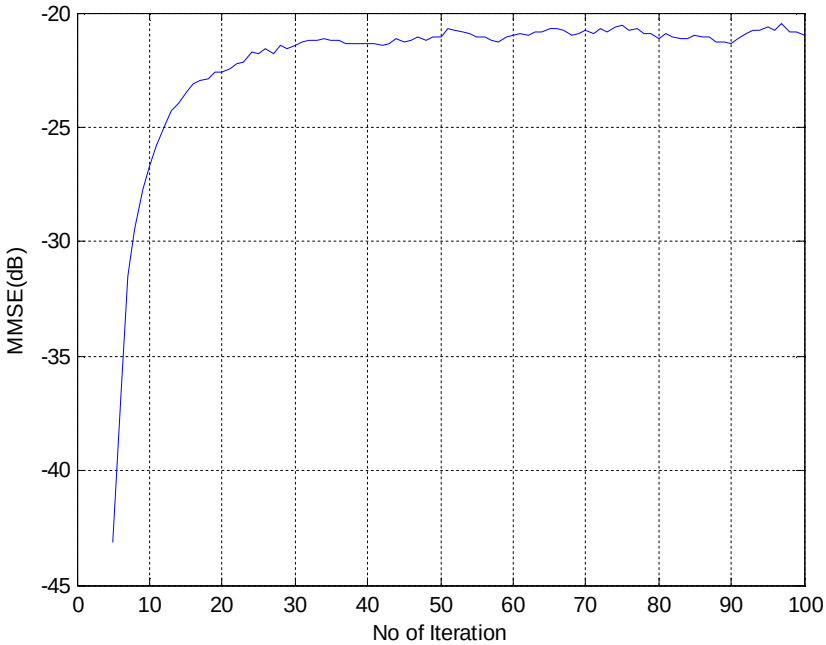

Fig.4.5.3.42. MMSE Performance.

4.6. Performance of DFF RLS Algorithm For Second Order Volterra System

4.6.1. Tracking Performance and MMSE of First Coefficient of Volterra System Using DFFRLS Algorithm

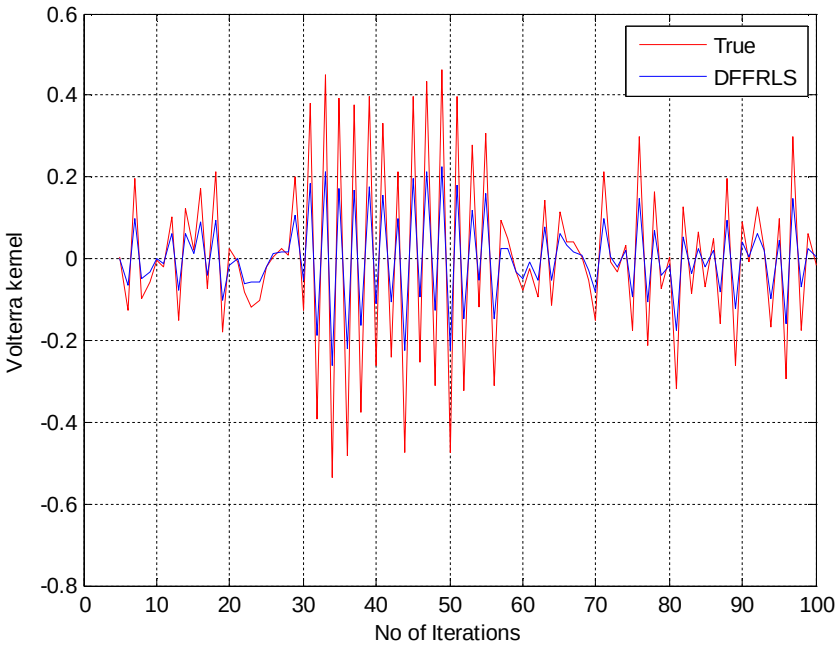

Fig.4.5.3.43. Tracking Performance.

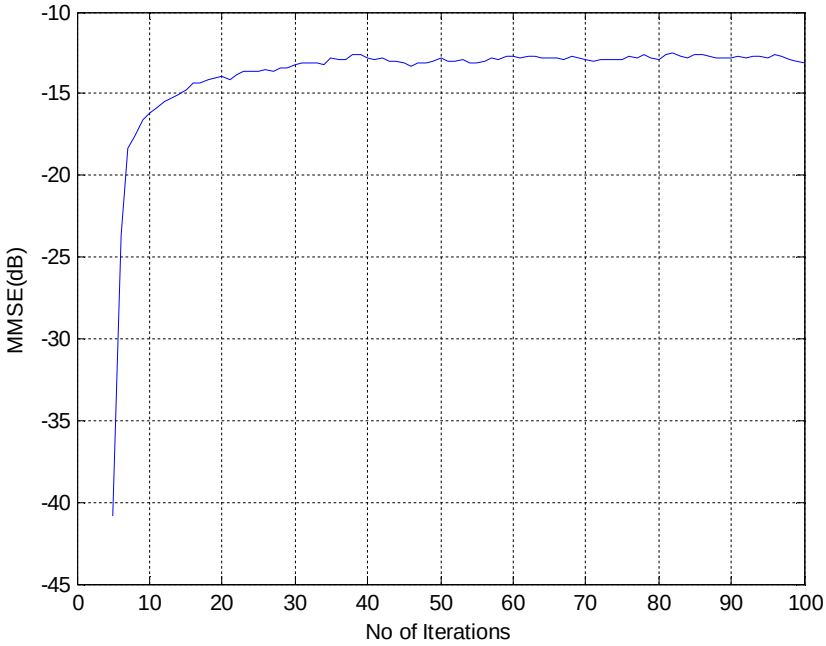

Fig.4.5.3.44.MMSE Performance.

4.6.2. Tracking Performance and MMSE of Second Coefficient of Second Order Time-Varying Volterra System Using DFFRLS Algorithm

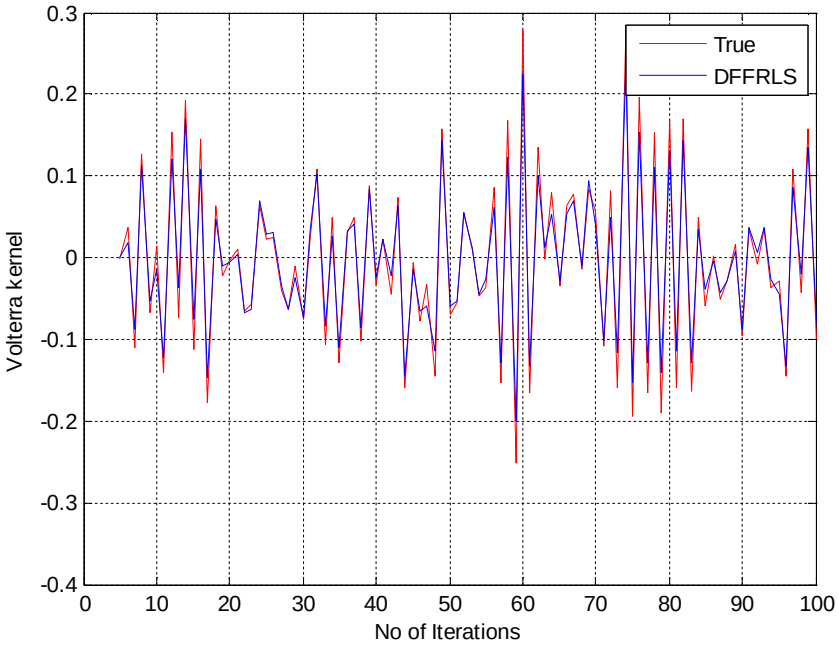

Fig.4.6.2.45. Tracking Performance.

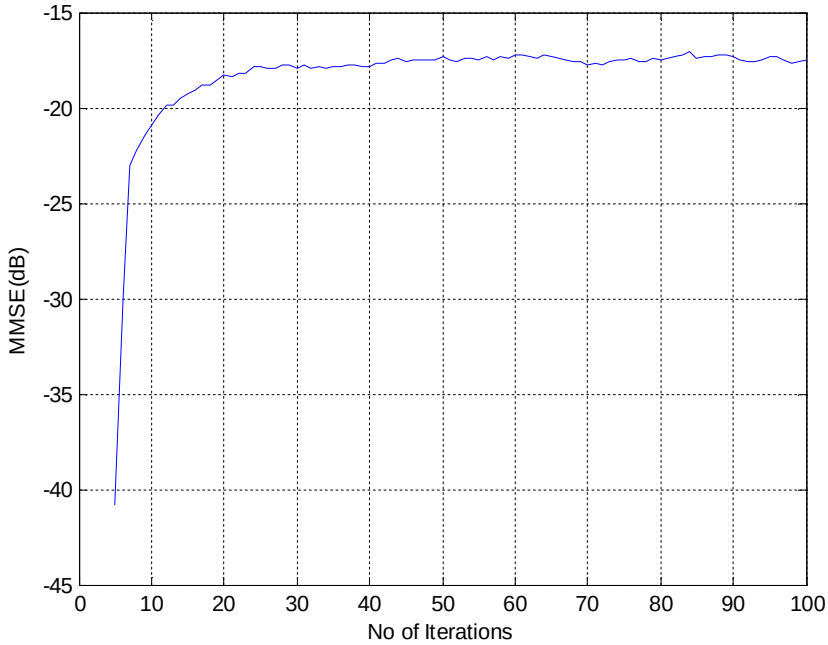

Fig.4.6.2.46. MMSE Performance.

4.6.3. Tracking Performance and MMSE of Third Coefficient of Second Order Time-Varying Volterra System Using DFFRLS Algorithm

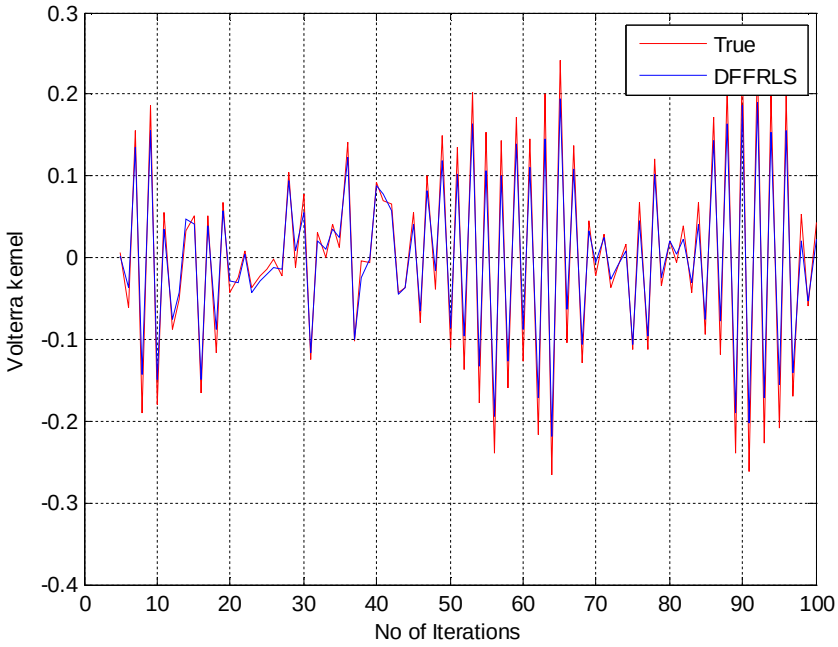

Fig.4.6.3.47. Tracking Performance.

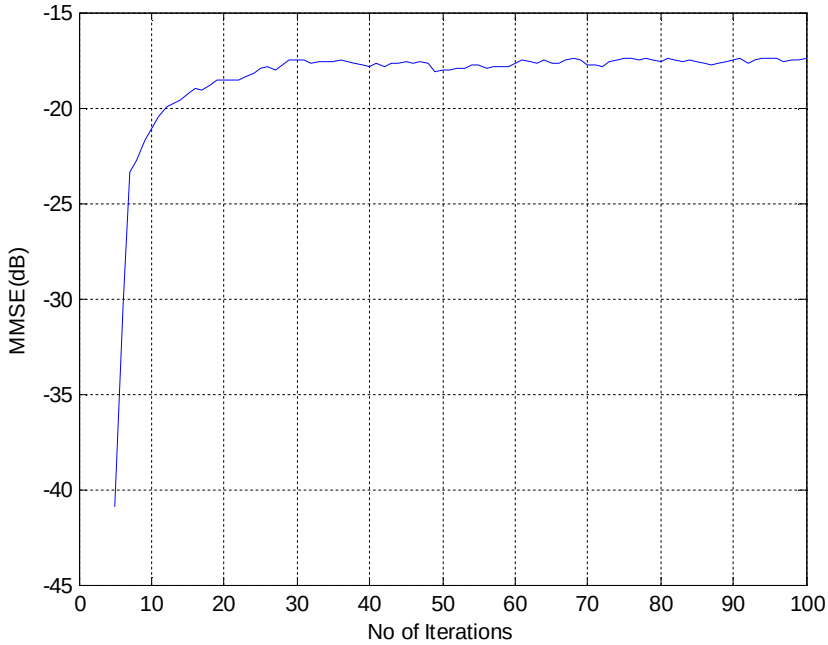

Fig.4.6.3.48. MMSE Performance.

4.6.4. Tracking Performance and MMSE of Fourth Coefficient of Second Order Time-Varying Volterra System Using DFFRLS Algorithm

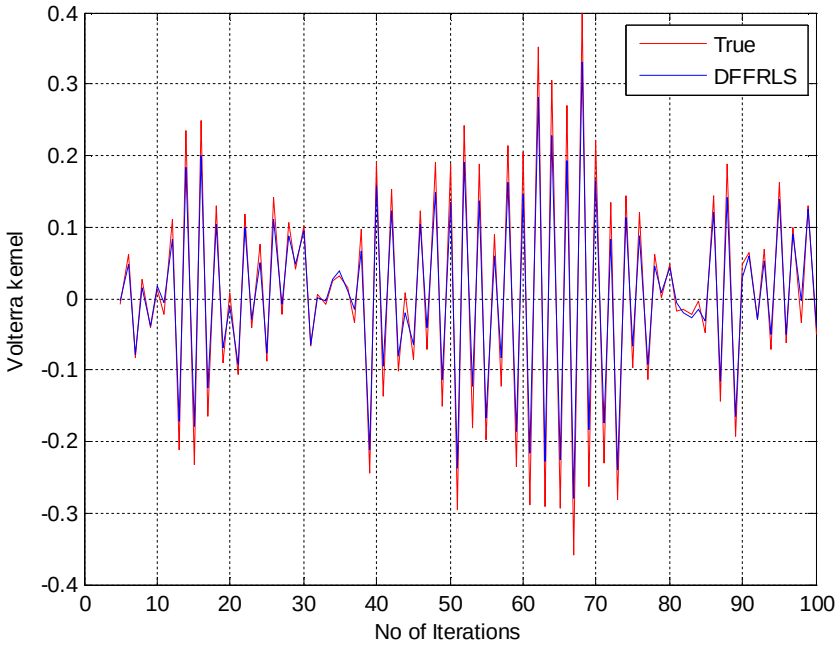

Fig.4.6.4.49. Tracking Performance.

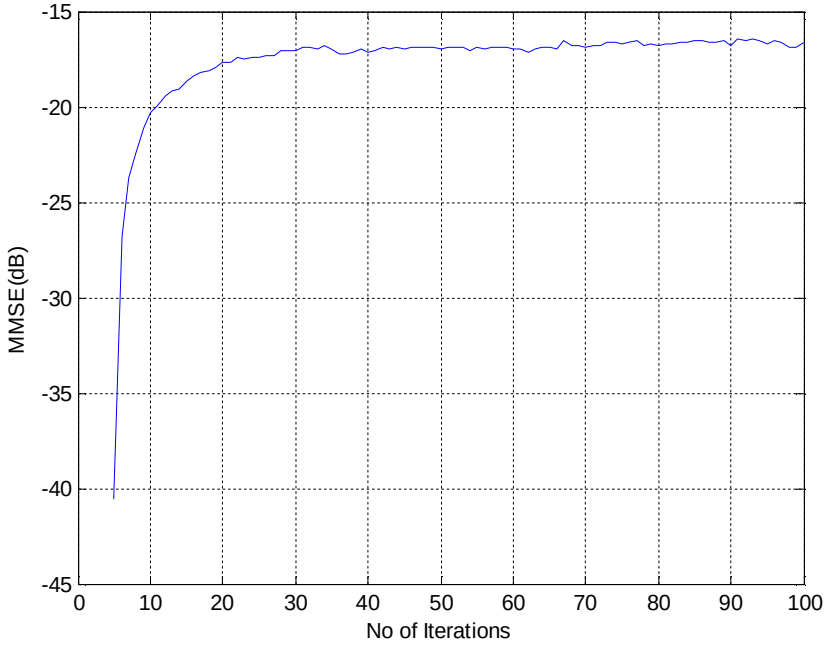

Fig.4.6.4.50. MMSE Performance.

4.6.5. Tracking Performance and MMSE of Fifth Coefficient of Second Order Time-Varying Volterra System Using DFFRLS Algorithm

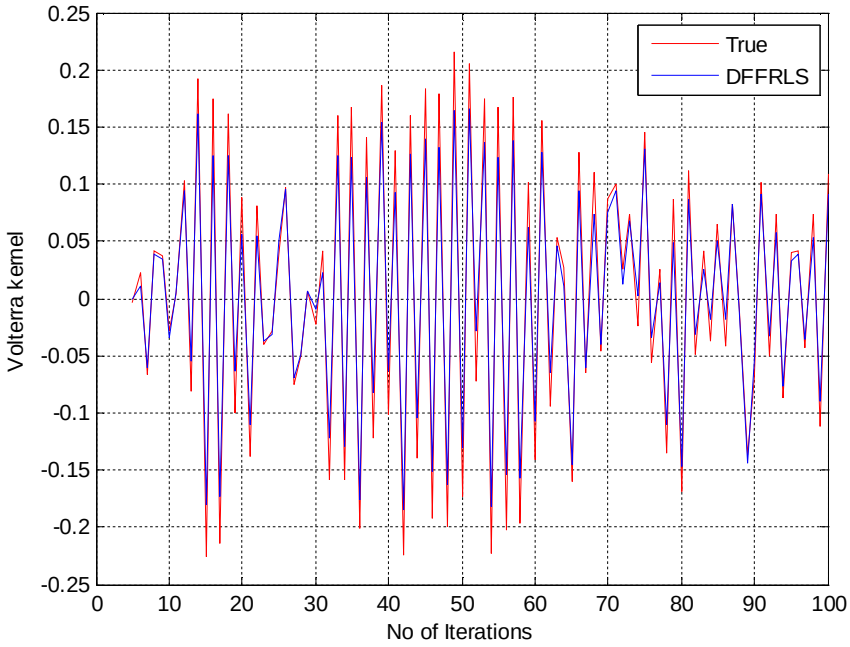

Fig.4.6.5.51. Tracking Performance.

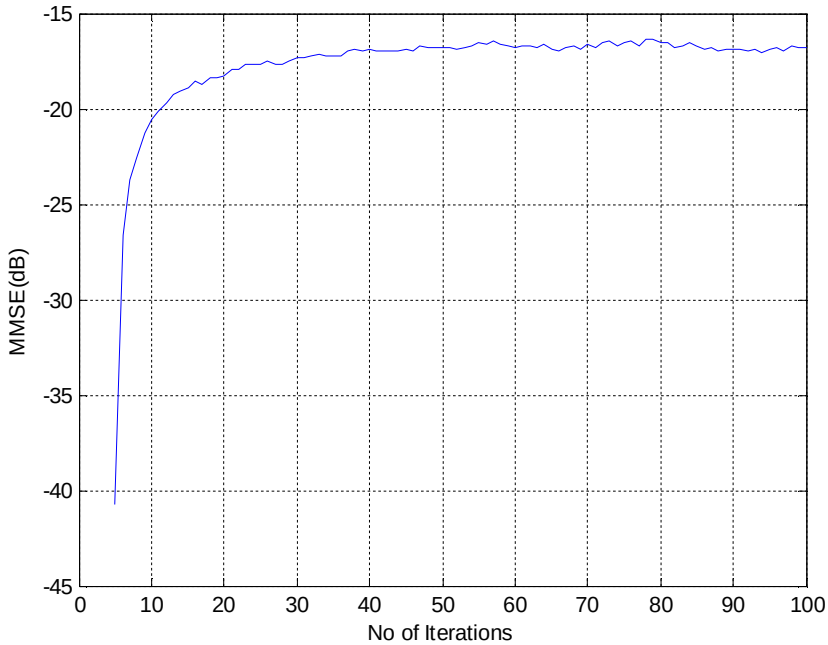

Fig.4.6.5.52. MMSE Performance.

4.6.6. Tracking Performance and MMSE of Sixth Coefficient of Second Order Time-Varying Volterra System Using DFFRLS Algorithm

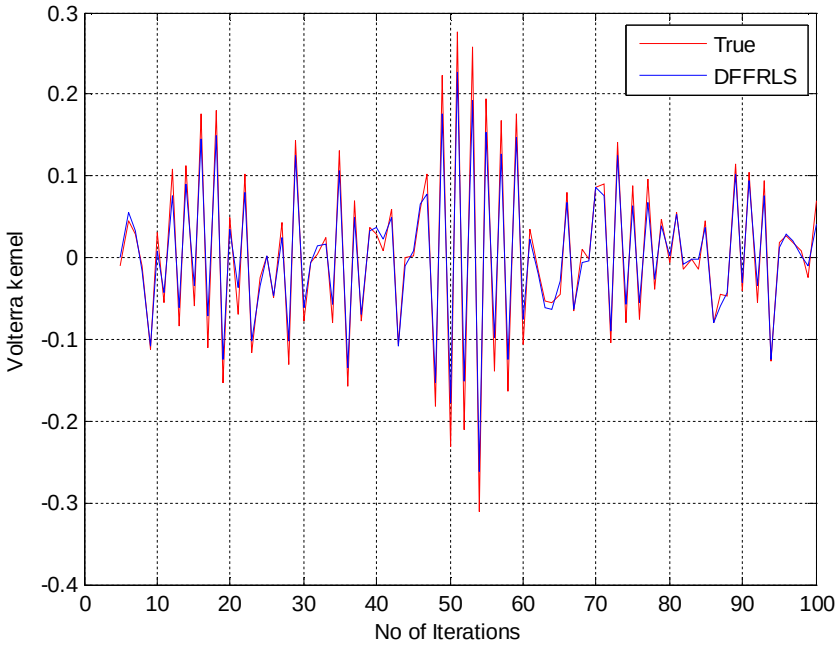

Fig.4.6.5.53. Tracking Performance.

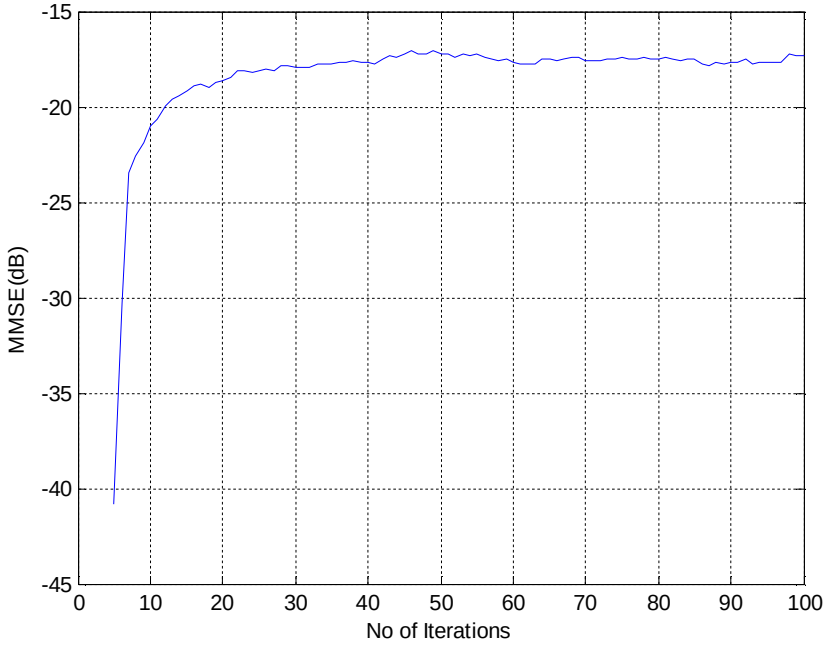

Fig.4.6.5.54. MMSE Performance.

Simulation Performance Discussion

In this thesis, first we have proposed the implementation of kalman algorithm and recursive lesst square (RLS) algorithm first and second order time varying volterra system. The performance of RLS-type algorithms depends upon convergence rate, tracking, misadjustment, and stability and all these factor vary according to the forgetting factor, and RLS do give complete performance satisfaction because fixed forgetting factor. So, we have proposed another adaptive algorithm for time varying volterra system having variable forgetting factor named dynamic forgetting factor recursive least square algorithm (DFFRLS). We will explain its MMSE performance in subsequent paragraphs and we will discuss its performance with respect to other algorithm.

From section 4.1, the tracking performance of first order Volterra filter using Kalman algorithm are shown. The results are presented for all the three Volterra coefficients. Minimum mean square estimate (MMSE) of above is presented. MMSE of first, second and third coefficient of Volterra filter approximately saturates at -36dB, -39 dB and -36.5dB respectively.

From section 4.2, the tracking performance and MMSE of second order Volterra system using Kalman algorithm are shown. Here the results are presented for all the six Volterra coefficients. Minimum mean square estimate (MMSE) of above is presented. Here MMSE for six coefficient saturates at-38dB, -40dB, 37.8dB, -35.5dB, -37.5dB and -37 dB.

From section 4.3, the tracking performance and MMSE of first order Volterra system using recursive least square algorithm (RLS) are shown. MMSE for three Volterra coefficients are -18.8 dB, -18.9 dB and -17.5 dB respectively. For second order Volterra system using RLS algorithm, MMSE of six Volterra coefficient is -12 dB ,17 dB , 16.5 dB, 16 dB ,16 dB and 16.8 dB.

The tracking performance of first order Volterra filter using Dynamic forgetting factor recursive least square (DFFRLS) algorithm are shown in section 4.5. The results are presented for all the three Volterra coefficients. Minimum mean square estimate (MMSE) of three Volterra coefficients saturates at -21.5 dB, -21.7 dB and -20.8 dB. For second order Volterra system using Dynamic forgetting factor recursive least square(DFFRLS)

algorithm, MMSE of six Volterra coefficient is -12.6 dB , 17.5 dB, 17 dB ,16.5 dB ,16.6 dB and -17.5 dB.

From the above performance figures, if we compare the performance of first order Volterra system for Kalman, RLS and DFFRLS algorithm in terms of MMSE, we find that Kalman has lowest MMSE and its tracking performance is much better than RLS and DFFRLS. The difference between the MMSE of RLS and DFFRLS algorithm of three Volterra kernels for first order Volterra system is -2.7 dB, -2.8 dB and -3.3dB. it means the performance of DFFRLS is better than RLS algorithm as DFFRLS has around 3dB less minimum mean square error.

For second order Volterra system, the Kalman filter still able to track the true value but the performance of RLS and DFFRLS has slightly degraded from first order Volterra system because convergence of higher-order kernels for both RLS and DFFRLS is slower than that of the linear kernel. but the difference between the MMSE of RLS and DFFRLS is approximately -0.6 dB,- 0.5 dB -0.5 dB,-0.5 dB,- 0.6 dB and -0.7 dB. The minimum mean square error of DFFRLS is approximately 0.5 dB less than RLS Volterra system. Therefore for second order Volterra system, the tracking performance of DFFRLS is slightly better than RLS Volterra system.

From the above discussion, a new algorithm DFFRLS for first and second order Volterra system is proposed for estimating time-varying spectra of nonstationary signals using variable forgetting factor which are adapted to the signals via the criterion of the extended prediction error. With respect to the simulation results, this method is shown to have better adaptability than the one with high FFF (close to unity) in the nonstationary situation, and to have a lower variance than the one with low FFF in the stationary situation. The extra computation time for the forgetting factor adaptation is almost negligible. It is inferred from simulation results that the higher order polynomial model-based RLS algorithms in conjunction with VFF are not providing any additional advantage.

CHAPTER 5

CONCLUDING REMARKS AND FUTURE SCOPE

In this thesis, tracking performance and MMSE performance of Kalman, RLSand dynamic forgetting factor recursive least square (DFFRLS) algorithms are analyzed and compared for parameter estimation under time-varying environment. The nonlinear system under consideration is Volterra system. Firstly we simulate the tracking performance for Kalman and RLS algorithm for first and second order Volterra system. Then we have performed simulations for DFF RLS for first and second Volterra filter of all Volterra coefficients. From the simulations, following results are inferred for the channel estimation algorithms.

- Tracking performance of Kalman filter for first order time varying Volterra system is far better than RLS and DFFRLS algorithm.
- Tracking performance of Kalman filter for second order time varying Volterra system is slightly better than first order Volterra system using Kalman filter.
- Tracking performance of Kalman filter for second order time varying Volterra system is far better than RLS and DFFRLS algorithm.
- Tracking performance of DFFRLS algorithm for first order time varying Volterra system is better than RLS algorithm.
- Tracking performance of RLS for second order time varying Volterra system is much worse than first Volterra system under RLS algorithm
- Tracking performance of DFFRLS for second order time varying Volterra system is much worse than first Volterra system under DFFRLS algorithm.
- Tracking performance of DFFRLS algorithm for second order time varying Volterra system is slightly better than second order Volterra system under RLS algorithm.
- Due to slow convergence and increase in computation complexity, Second order RLS and DFFRLS do not have performance as good as first order RLS and DFFRLS respectively.

Future Scope

As the application of signal processing increases, the complexity of nonlinear systems also increases. Here we have performed simulations of adaptive algorithm for first

and second order Volterra system, but as with the increase in the sources of nonlinearity, we have to look upon certain adaptive algorithm or other refined adaptive algorithm that is used to overcome the nonlinear effects. So, we can apply these adptive algorithms for parameter estimation of higher Volterra filter. We can also apply these adaptive algorithms in channel estimation for latest wireless technologies like BLAST, 3G, 4G etc.

Another possible way to improve the tracking performance of DFFRLS algorithm by increasing the value of averaging factor that is K in extended estimation formula.

REFERENCES

[1]. S. Haykin, *Adaptive Filter Theory*, 4th ed. New Jersey: Prentice Hall 2002.

[2]. A. Benveniste, "Design of adaptive algorithms for the tracking of time varying systems," Int J. Adaptive Contr. Signal Processing, vol. 1, pp.3–29, 1987.

[3]. O. Macchi, "Optimization of adaptive identification for time-varying filters," *IEEE Trans. Automat. Contr*, vol. AC-31, pp. 283–287, Mar.1986.

[4]. B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.

[5]. Lucky, R.W., Techniques for adaptive equalization of digital communication systems, *Bell Sys. Tech. J.*, 45, 255–286, Feb. 1966.

[6]. M. Rupp, "A Family of Adaptive Filter Algorithms with Decorrelating Properties", *IEEE Transactions on Signal Processing*, Vol. 46, No. 3, pp. 771 – 775, Sept. 1993.

[7]. Qureshi, S.U.H., Adaptive equalization, *Proc. IEEE,* 73(9), 1349–1387, Sept. 1985.

[8]. J. G. Proakis, *Digital communication*. New York: McGraw-Hill, 1989.

[9]. F. Ling and J. G. Proakis, "Nonstationary learning characteristics of least square adaptive estimation algorithm," *in Proc. IEEE Trans. Acoust., Speech, Signal Processing*, vol. 34, no. 5, pp. 1097-1110, Oct. 1986.

[10]. S. Gazor, "Prediction in LMS type adaptive algorithms for smoothly time-varying environments," *IEEE Trans. Signal Process.*, vol. 47, no. 7, pp. 1735- 1739, Jun. 1999.

[11]. B. Widrow and E. Walach, "On the statistical efficiency of the LMS algorithm with nonsatationary inputs," *IEEE Trans. Inf. Theory (Special Issue on Adaptive Filtering)*, vol. IT-30, no. 2, Mar. 1984.

[12]. B. Farhang-Borojeny and S. Gazor, "Performance of LMS-based adaptive filter in tracking a time-varying plant," *IEEE Trans. Signal Process.*, vol. 44, no. 11, pp. 2868-2871, Nov.1996.

[13]. Y. Xue and X. Zhu, "Second order LMS based wireless channel tracking: Implementation under imperfect carrier synchronization," *IEEE Trans. SignalProcess*, vol. 83, pp. 199-212, Jan.2003.

[14]. O. M. Macchi, Adaptive Processing: The LMS approach with applications in Transmission. New York: Wiley, 1995.

[15]. A. H. Sayed and T. Kailath, " A state-space approach to adaptive RLS filtering," *IEEE Signal Process. Mag.*, vol. 11, no. 3, pp. 18-60, Jul. 1994.

[16]. S. Haykin, A. H. Sayed, J.R. Zeidler, P. Yee and P. C. Wei, "Adaptive tracking of linear Time-variant systems by extended RLS algorithms," *IEEE Trans. Signal Process.*, vol. 45, no. 5, pp. 1118-1128, May 1997.

[17]. D. J. Park, , B. E. Jun and J. H. Kim, "Fast tracking RLS algorithm using novel variable forgetting factor with unity zone," *Electronics Letters*, vol. 27, no. 3, pp. 2150-2151, 7 November 1991.

[18]. S. H. Lengue and C. F. So, "Non linear RLS algorithm using variable forgetting factor in mixture noise," in *proc.ICASSP'01*,vol.6, 2001, pp. 3777-3780.

[19]. Lucky. R. W. "Modulation and detection for data transmission on the telephone channel." In New Directions in Signal Processing in Communication and Control, J. K. Skwirzynski, ed., Leiden, Holland. Noordhoff, 1975.

[20]. J. Katzenelson and L. A. Gould, "The design of nonlinear filters and control systems," *Inform. Contr.*, vol. *5*, pp. 108-143, 1962.

[21]. J. F. Barret, "The use of functionals in the analysis of nonlinear system," J. Electron. Contr., vol. 15, pp. 567-615, 1963.

[22]. P. Eykhoff, "Some fundamental aspects of process-parameter estimation," *IEEE Trans. Automat. Contr.*, vol. AC-8, pp. 347-357, Oct. 1963.

[23]. A. V. Barakrishnan, "A general theory of nonlinear estimation problems in control systems," J. Math. Anal. Appl., vol. 8, pp. 4-30, Feb. 1964.

[24]. Wi581Wiener. N., *Nonlinear Problems in Random Theory*, Wiley and Sons, New York, 1958.

[25]. V. J. Mathews, "Adaptive polynomial filters," *IEEE, Signal Processing Magazine*, pp. 10–25, July 1991.

[26]. M. Schetzen, The Volterra and Wiener theories of nonlinear systems. New York: John Wiley and Sons, 1980.

[27]. T. Koh and E. J. Powers, "Second-order Volterra filtering and its application to nonlinear system identification," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, no. 6, pp. 1445–1455, Dec. 1985

[28]. J. B. MacNeil, R. E. Kearney, and I. W. Hunter, "Identification of time varying biological systems from ensemble data," *IEEE Trans. Biomed. Eng.*, vol. 39, no. 12, pp. 1213–1225, Dec. 1992.

[29]. M. Green and A. M. Zoubir, "Selection of a time-varying quadratic Volterra model using a wavelet packet basis expansion," *IEEE Trans. Signal Processing*, vol. 52, no. 10, pp. 2721–2728, Oct. 2004.

[30]. A. E. Nordsj¨o and L. H. Zetterberg, "Identification of certain time varying nonlinear Wiener and Hammerstein systems," *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 577–592, Mar. 2001.

[31]. Weng, B and Barner. K.E, "Time-varying Volterra system identification using Kalman Filtering", Information Sciences and System, 2006 40th Annual Conference, Page(s): 1617 – 1622, 2006.

[32]. B. Toplis and *S* . Pasupathy, "Tracking improvements in fast RLS algorithm using a variable forgetting factor," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 206-227, Feb. 1988.

[33]. Y. *S* . Cho, *S* . B. Kim, and E. J. Powers, "Time-frequency analysis using AR models with variable forgetting factors," in *Proc. ICASSP'90*, 1990, pp. 2479-2482.

[34]. Y. S. Cho, S. B. Kim, and E. J. Powers, "Time-varying spectral estimation using AR models with variable forgetting factors," *IEEE Transactions on Signal Processing*, vol. 39, no. 6, pp. 1422– 1426, 1991.

[35]. N. Zhou and N. Holte, "Least squares channel estimation for a channel with fast time variations," in *Proc. ICASSP'92*, 1992, pp. v165–v168.

[36]. J. Lin *and* John G. Proakis, "Optimal tracking of time-varying channel: A frequency domain approach for known and new algorithms," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 141–154, Jan. 1995.

[37]. D. K. Borah and B. D. Hart, "Frequency-selective fading channel estimation with a polynomial time-varying channel model*," IEEE Trans. Commun.*, vol. 47, pp. 862–873, June 1999.

[38]. S. Song, J. S. Lim, S. J. Baek, and K. M. Sung, "Variable forgetting factor linear least squares algorithm for frequency selective fading channel estimation," *IEEE Transactions on Vehicular Technology*, vol. 51, no. 3, pp. 613–616, 2002.

[39]. Amit Kumar Kohli, Amrita Rai and Meher Krishna Patel "Variable Forgetting Factor LS Algorithm for Polynomial Channel Model" ISRN Signal Processing, Volume 2011, 4 pages.

[40]. Q. T. Zhang and S. Haykin, "Tracking characteristics of the Kalman filter in a nonstationary environment for adaptive filter applications," *Proc. ICASSP*, Boston, MA, 1983, pp. 671–674.

[41]. Kalman. R, "On the general theory of control systems*", IRE Transactions on Automatic Control*, vol-4, pp. 110 – 110, 1959

[42]. E. Eleftheriou and D. D. Falconer, "Tracking properties and steady-state performance of RLS adaptive filter algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 1097–1110, 1986.

[43]. Adaptive Signal Processing-Applications to Real-World Problems, J. Benesty and Y. Huang, Eds. Berlin, Germany: Springer-Verlag, 2003.

[44]. A. H. Sayed and T. Kailath, "A state-space approach to adaptive RLS filtering," *IEEE Signal Process. Mag.*, vol. 11, no. 3, pp. 18–60, Jul. 1994.

[45]. S. Song, J. S. Lim, S. J. Baek, and K. M. Sung, "Gauss Newton variable forgetting factor recursive least squares for time varying parameter tracking," *Electron. Lett.*, vol. 36, no. 11, pp. 988–990, May 2000.

[46]. S.-H. Leung and C. F. So, "Gradient-based variable forgetting factor RLS algorithm in time-varying environments," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3141–3150, Aug. 2005.

[47]. T. R. Fortescue, L. S. Kershenbaum, and B. E. Ydstie, "Implementation of self-tuning regulators with variable forgetting factors," *Automatica*, vol. 17, pp. 831-835, 1981

[48]. N. Martin, "An AR spectral analysis of nonstationary signals," *Signal Processing*, vol. 10, pp. 61-74, 1986.

[49]. D. Clarke, and P. J. Gawthrop, "Self-tuning controller," Proc. *Inst. Elec. Eng.*, vol. 122, pp. 929-934, 1975.

[50]. Greg Welch and Gary Bishop, "An Introduction to the Kalman Filter", Course 8, SIGGRAPH 2001.