DEVELOPMENT OF INCREMENTAL STRATEGIES FOR WIRELESS SENSOR NETWORK

A Thesis submitted in partial fulfillment of the Requirements for the degree of

Master of Technology

In

Electronics and Communication Engineering

Specialization: signal and image processing

By

SIBA PRASAD MISHRA

Roll No. : 212EC6191

Under the Guidance of

Prof. A.K.Sahoo

Department of Electronics and Communication Engineering

National Institute of Technology Rourkela

Rourkela, Odisha, 769 008, India

May 2014

DEVELOPMENT OF INCREMENTAL STRATEGIES FOR WIRELESS SENSOR NETWORK

A Thesis submitted in partial fulfillment of the Requirements for the degree of

Master of Technology

In

Electronics and Communication Engineering

Specialization: Signal and Image Processing

By

Siba Prasad Mishra

Roll No. : 212EC6191

Under the Guidance of

Prof. A.K.Sahoo

Department of Electronics and Communication Engineering

National Institute of Technology Rourkela

Rourkela, Odisha, 769 008, India

May 2014

Dedicated to…,

My parents and my elder brother and sisters

DEPT. OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA
ROURKELA – 769008, ODISHA, INDIA

Certificate

This is to certify that the work in the thesis entitled **DEVELOPMENT OF INCREMENTAL STRATEGIES FOR WIRELESS SENSOR NETWORK** by Siba Prasad Mishra is a record of an original research work carried out by him during 2013 - 2014 under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of Master of Technology in Electronics and Communication Engineering (Signal and Image Processing), National Institute of Technology, Rourkela. Neither this thesis nor any part of it, to the best of my knowledge, has been submitted for any degree or diploma elsewhere.

Place: NIT Rourkela

Date: 25th May 2014

Prof.A.K.Sahoo

Professor

DEPT. OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA
ROURKELA – 769008, ODISHA, INDIA

Declaration

I certify that

a) The work contained in the thesis is original and has been done by myself under the general supervision of my supervisor.

b) The work has not been submitted to any other Institute for any degree or diploma.

c) I have followed the guidelines provided by the Institute in writing the thesis.

d) Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

e) Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Siba Prasad Mishra

25th May 2014

Acknowledgements

It is my immense pleasure to avail this opportunity to express my gratitude, regards and heartfelt respect to Prof. Ajit Kumar Sahoo, Department of Electronics and Communication Engineering, NIT Rourkela for his endless and valuable guidance prior to, during and beyond the tenure of the project work. His priceless advices have always lighted up my path whenever I have struck a dead end in my work. It has been a rewarding experience working under his supervision as he has always delivered the correct proportion of appreciation and criticism to help me excel in my field of research.

I would like to express my gratitude and respect to HOD Prof.S Meher , Prof. P. Singh, Prof. S. Maiti ,Prof. L.P.Roy and Prof. S. Ari for their support, feedback and guidance throughout my M. Tech course duration. I would also like to thank all the faculty and staff of ECE department, NIT Rourkela for their support and help during the two years of my student life in the department.

I would like to make a special mention of the selfless support and guidance I received from PhD Scholar Mr.Sanand Kumar and Mr.Nihar Ranjan Panda, Department of Electronics and Communication Engineering, NIT Rourkela during my project work. Also I would like to thank Trilochan Behera , Chandan Kumar, Prateek Mishra and Avinash Giri for making my hours of work in the laboratory enjoyable with their endless companionship and help as well; along with all other friends like Sumit kumar, Dhunish Kumar, Nilay Pandey, Anurag Patra ,Manu Thomas, and many more who made my life in NIT Rourkela a memorable experience all together.

Last but not the least; I would like to express my love, respect and gratitude to my parents and my elder brother Durga Prasad Mishra, my two sisters, my sir Surjyo Narayan Panigrahy, Sudhansu Nayak ,Jagadananda Purohit and my best friend Guddi Rani Panda who have always supported me in every

decision I have made, guided me in every turn of my life, believed in me and my potential and without whom I would have never been able to achieve whatsoever I could have till date.

SIBA PRASAD MISHRA

Mishra.sibaprasad11@gmail.com

ABSTRACT

Adaptive filter plays an important role in the field of digital signal processing and wireless communication. It incorporates LMS algorithm in real time environment because of its low computational complexity and simplicity. The LMS algorithm encompasses RLS (recursive least square), GN (Gaussian Newton), LMF (least mean fourth) and XE-NLMF algorithms, which provides faster convergence rate and low steady state error when compared to LMS.

The adaptive distributed strategy is based on the incremental mode of co-operation between different nodes, which are distributed in the geographical area. These nodes perform local computation and share the result with the predefined nodes. The resulting algorithm is distributed, co-operative and able to respond to the real time change in environment. By using incremental method, algorithms such as RLS,GN, DCT-LMS and DFT-LMS produces faster convergence and better steady state performance than that of the LMS when simulated in the presence of Gaussian noise. Higher Order error algorithm like LMF, XE-NLMF and variable XE-NLMF algorithm produce better convergence and steady state performance under Gaussian and non-Gaussian noise. A spatial-temporal energy conservation argument is used to evaluate the steady state performance of the entire network.

A topology named as CLMS (convex LMS) was presented which combined the effect of both fast and accurate filtering at the same time. Initially CLMS have parallel independent connection, the proposed topology consists of series convex connection of adaptive filters, which achieves similar result with reduced time of operation. Computer simulations corroborate the results.

Keywords: Incremental, Adaptive, CLMS,INC DCT-LMS,INC DFT-LMS,QWDILMS,XE-NLMF,LMF,LMS

Table of Contents

Acknowledgements 1

Chapter 1 INTRODUCTION 9

1.1 PROBLEM STATEMENT 10

1.2 THESIS LAYOUT 13

Chapter 2 INCREMENTAL ADAPTIVE STRATEGIES OVER DISTRIBUTED NETWORK 15

2.1 Applications 16

2.2 Modes of cooperation 17

2.3 Consensus strategy 18

2.4 Contribution 19

2.5 ESTIMATION PROBLEM AND ADAPTIVE DISTRIBUTED SOLUTION 20

2.5.1 Steepest Descent Solution 21

2.5.2 Incremental Steepest Descent Solution 23

2.5.3 Incremental Adaptive Solution 23

2.6 PERFORMANCE ANALYSIS 24

2.6.1 Data Model and Assumption 25

2.6.2 Weighted Energy Conservation Relation 25

2.6.3 Gaussian Data 28

2.6.4 Steady State Behavior 30

2.6.5 Simulation Results 32

2.7 QUALITY AWARE INCREMENTAL LMS ALGORITHM FOR DISTRIBUTED ADAPTIVE ESTIMATION 37

2.7.1 Effect Of Noisy Nodes 37

2.7.2 QWDILMS Algorithm 39

2.7.3 Conclusion 45

Chapter 3 FREQUENCY DOMAIN INCREMENTAL STRATEGIES OVER DISTRIBUTED NETWORK 46

3.1 PREWHITENING FILTERS 47

3.2 UNITARY TRANSFORMATION 50

3.2.1 General Transform Domain LMS Algorithm 53
3.2.2 DFT Domain LMS Algorithm 54
3.2.3 DCT LMS Algorithm 54
3.3 RLS ALGORITHM 60
3.4 Conclusion 66
Chapter 4 CONVERGENCE ANALYSIS OF VARRIABLE XE-NLMF ALGORITHM 67
4.1 LMF Algorithm 67
4.2 OPTIMIZED NORMALIZED ALGORITHM FOR SUBGAUSSIAN NOISE 70
4.3 VARIABLE NORMALIZED XE-NLMF ALGORITHM 72
4.3.1 Convergence Analysis 73
4.4 Simulation Results 74
Chapter 5 CONVEX COMBINATION OF ADAPTIVE FILTER 77
5.1 PARALLEL INDEPENDENT STRUCTURE 77
5.2 SERIES COOPERATIVE STRUCTURE 80
5.3 SWITCHING ALGORITHM 80
5.3.1 Deterministic Design of the Combining Parameter 81
5.3.2 A Simple Design for the Mixing Parameter 83
5.4 Conclusion 86
Chapter 6 CONCLUSION AND FUTURE WORK 87
6.1 Conclusion 87
6.2 Future work 88
Bibliography 89

LIST OF FIGURE

Fig. 1 Distributed network 15
Fig. 2 monitoring a diffusion phenomenon by a network of sensors 16
Fig. 3 three mode of cooperation (a) incremental (b) diffusion (c) probabilistic diffusion 17
Fig. 4 Distributed network with N nodes accessing space time data 20
Fig. 5 Data processing in adaptive distributed structure 22
Fig. 6 Regressor power profile 33
Fig. 7 Correlation index per node 33
Fig. 8 Noise power profile 34
Fig. 9 Transient MSE performance at node 1for both incremental adaptive solution and stochastic steepest descent solution 34
Fig. 10 Transient EMSE performance at node 1for both incremental adaptive solution and stochastic steepest descent solution 35
Fig. 11 Transient MSD performance at node 1for both incremental adaptive solution and stochastic steepest descent solution 35
Fig. 12 MSE performance node wise 36
Fig. 13 EMSE performance node wise 36
Fig. 14 MSD performance node wise 37
Fig. 15 The node profile of $\boldsymbol{\sigma u, k2}$ 38
Fig. 16 The node profile of $\boldsymbol{TrRu, k}$ 39
Fig. 17 The global average EMSE for DILMS algorithm in different condition 39
Fig. 18 Block diagram of proposed algorithm 43
Fig. 19 The EMSE performance of DILMS algorithm with and without noisy nodes and QWDILMS Algorithm 44
Fig. 20 The MSD performance of DILMS algorithm with and without noisy nodes and QWDILMS Algorithm 44
Fig. 21 filtering of wide sense stationary random process {x(i)} by a stable linear system H(z) 49
Fig. 22 Prewhitening of $\boldsymbol{ui}$ by using the inverse of the spectral factor of $\boldsymbol{Suz}$ 49
Fig. 23 Adaptive filter implementation with a prewhitening filter 50
Fig. 24 Transform domain adaptive filter implementation, where T is a unitary transformation 52
Fig. 25 comparison between LMS, DFT-LMS, DCT-LMS and DCT-LMS 56
Fig. 26 Block diagram of frequency domain incremental LMS algorithm 57
Fig. 27 Transient MSE performance at node 1 for incremental adaptive solution, stochastic steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS 58
Fig. 28 EMSE performance at node 1 for incremental adaptive solution, incremental steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS 58
Fig. 29 MSD performance at node 1 for incremental adaptive solution, incremental steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS 59
Fig. 30 MSE performance with respect to Node for all algorithm 59
Fig. 31 EMSE with respect to node for all the algorithm 60
Fig. 32 MSE comparison between RLS and LMS algorithm 63

Fig. 33 MSE comparison between GN and LMS algorithm 63
Fig. 34 MSE comparison of RLS, LMS and GN algorithm 64
Fig. 35 MSE performance comparison of RLS, LMS and GN algorithm using incremental strategies ... 64
Fig. 36 EMSE performance comparison of RLS, LMS and GN algorithm using incremental strategies. 65
Fig. 37 MSD performance comparison of RLS, LMS and GN algorithm using incremental strategies... 65
Fig. 38 Performance comparison of LMS, NLMS and LMF 68
Fig. 39 Performance comparison of LMS, NLMS and LMF algorithm 68
Fig. 40 performance comparison between LMF and LMS for different noise condition 69
Fig. 41 comparison of LMF and LMS algorithm for different noise condition using incremental method 70
Fig. 42 convergence performance of XENLMF, LMF, LMS and NLMS for $\boldsymbol{\lambda = 0.9}$ 72
Fig. 43 Effect of lambda in the fixed XE-NLMF 74
Fig. 44 convergence performance for the variable XE-NLMF algorithm, the XE-NLMF algorithm (λ=0.9) and the NLMS algorithm in White Gaussian noise using incremental adaptive algorithm 75
Fig. 45 convergence performance of NLMS, XE-NLMF and variable XE-NLMF under Binary additive noise case using incremental adaptive algorithm 76
Fig. 46 MSE performance of NLMS, XE-NLMF and variable XE-NLMF under AWGN case using incremental adaptive algorithm 76
Fig. 47 convex combination of two adaptive filter 78
Fig. 48 EMSE of the LMS filter and their convex combination averaged over 200 realization 79
Fig. 49 EMSE of the LMS filter and their convex combination averaged over 200 realization using incremental adaptive algorithm 79
Fig. 50 series topology 80
Fig. 51 Time revolution curve of λ(i) for both CLMS and INC-COOP 82
Fig. 52 EMSE curve for the fast filter, accurate filter, CLMS, INCCOOP1 and INC-COOP2 using incremental adaptive algorithm 83
Fig. 53 time evaluation of $\boldsymbol{\lambda s(i)}$ using the simple design technique for both CLMS and INC-COOP algorithm 84
Fig. 54 EMSE performance for fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 algorithm using simple design technique 84
Fig. 55 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=10 dB using incremental adaptive algorithm 85
Fig. 56 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=5 dB using incremental adaptive algorithm 85
Fig. 57 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=3 dB using incremental adaptive algorithm 86

ABBREVIATIONS

LMS	Least Mean Square
NLMS	Normalized LMS
RLS	Recursive Least Square
LMF	Least Mean fourth
NLMF	Normalized LMF
XE-NLMF	Variable NLMF
CLMS	Convex LMS
DCT-LMS	Discrete cosine transform LMS
DFT-LMS	Discrete Fourier transform LMS
MSE	Mean Square Error
MSD	Mean Square Deviation
EMSE	Excess Mean Square Error
DILMS	Distributed LMS
QWLMS	Quality aware LMS

Chapter 1 INTRODUCTION

Wireless Sensor Networks (WSNs) is networks composed of tiny embedded devices. Each device is capable of sensing, processing and communicating the local information. The networks can be made up of hundreds or thousands of devices that work together to communicate the information that they obtain [1]. In distributed signal processing Number of nodes are distributed in a geographical area, it collects the information or data which is present in the node. Each node assembles some noisy information related to a certain parameter of interest and performing local estimation, then share the data to the other nodes by some defined rule. The main object behind this is to reach the parameter of interest, which really outcomes from the node after share in the network. In traditional centralized solution the nodes collect the data then send it to the central processor for processing, the central processor process the data then finally again give back the estimated data to all the node. For this a powerful central processor required and a huge amount of communication between node and central processor required. But in case of distributed solution, the nodes only depends on their immediate neighbor [2]. Hence in case of distributed solution the amount of processing and communication reduced ([1], [3]).

Distributed solution has large number of application including tracking of target trajectory, monitoring concentration of chemical in air or water, also having application in agriculture, environment monitoring, disaster relief management, medical ([1], [4]) etc. There are three mode of cooperation namely incremental, diffusion and probabilistic diffusion will discuss in chapter 2. Here we use only the incremental mode of cooperation. This chapter describes about the central distributed algorithm, non-distributed algorithm and the advantage of distributed over non distributed solution. The comparison is done on the basis of convergence rate, steady state performance and computational complexity. There are two type of algorithm used one is incremental steepest descent solution and other is incremental adaptive solution, comparing both on the basis convergence rate and steady state performance the adaptive solution perform better than steepest descent solution. The more explanation will found in the chapter 2.each case we consider the variance of noise is small i.e. Less than one, but sometime case arises where the noise

variance is more than that of one, than a quality aware algorithm is used in the incremental method to maintain the steady state performance.

The convergence performance of LMS (least mean square) algorithm depends on the correlation of the input data and the Eigen value spread of the covariance matrix of the regressor data. The smaller Eigen value of auto-correlation matrix results in slower convergence and larger Eigen value limit the range of the allowed step size and thereby limit the learning abilities of the filter. Best convergence result when all the Eigen value equal i.e. having unit Eigen spread, this is possible only when auto correlation matrix is constant multiplication of identity matrix. This can be achieved by pre-whiten the data by passing it through pre-whiten filter which is practically not possible. Hence same thing will achieve by unitary transformation of data, such as DFT (discrete Fourier transform), and DCT (discrete cosine transform) [5].

Adaptive algorithms based on the higher order moments of the error signal found performs better than that of LMS algorithm in some important application. The practical use of such type application is not considerable because of its lack of accuracy in the model to predict the behavior. One of such type of algorithm is LMF (least mean fourth) algorithm, which minimize the mean fourth error. It is found that the LMF algorithm outperforms than the LMS algorithm in non-Gaussian noise case [6]. We will find the family of LMF algorithm and its performance in both Gaussian and non Gaussian noise case in the chapter 4.

Generally fast filter gives higher convergence rate and accurate filter gives better steady state performance. An algorithm developed named CLMS (convex LMS) algorithm which consists of two adaptive filters connected parallel. The CLMS algorithm track initially the faster convergence respond, then followed the accurate response. It has advantage that it achieve both at the same time. It is very difficult to develop a filter which provides both at same time. Hence this algorithm has number of application in the distributed signal processing.

1.1 PROBLEM STATEMENT

Adaptive digital filtering self-adjusts its transfer function to get an optimal model for the unknown system based on some function of error based on the output of the adaptive filter and the unknown system. To get an optimal model of the unknown system it depends on the structure, adaptive algorithm and the nature of the input signal. System Identification estimates models of dynamic

systems by observing their input output response when it is difficult to obtain the mathematical model of the system.

Mathematical analysis has also been extended to the transform domain adaptive filter, CLMS algorithm, XE-NLMF algorithm and variable XE-NLMF algorithm. This work has examined the convergence conditions, steady-state performance, and tracking performance. The theoretical performance is confirmed by computer simulations. The performance is compared between the original adaptive filter algorithms and different other algorithm like incremental adaptive solution, incremental RLS, incremental GN, incremental CLMS, XE-NLMF and incremental variable XE-NLMF algorithm. Since a specific method mention previously in one adaptive filter algorithm may achieves good performance, but may not perform well in another adaptive filter algorithm, hence we will examine the number of methods in adaptive filter to find the better one.

In wireless sensor network the fusion center provides a central point to estimate parameters for optimization. Energy efficiency i.e. low power consumption, low latency, high estimation accuracy and fast convergence are important goals in estimation algorithms in sensor network. Depending on application and the resources, many algorithms are developed to solve parameter estimation problem. One approach is the centralized approach in which the most information to be present when making inference. However, the main drawback is the drainage of energy resources to transmit all observation to fusion center at every iteration. So this is wasting energy at idle time interval. Hence there was a need to find an approach that avoids the fusion center all together and allows the sensors to collaboratively make inference. This approach is called as the distributed scheme. Distributed computation of algorithms among sensors reduces energy consumption of the overall network, by tradeoff between communication cost and computational cost. In order to make the inference procedure robust to nodal failure and impulsive noise, robust estimation procedure should be used. Optimization of sensor locations in a network is essential to provide communication for a longer duration. In most cases sensor placement needs to be done in hostile areas without human involvement, e.g. by air deployment. The aircraft carrying the sensors has a limited payload, so it is impracticable to randomly drop thousands of sensors over the ROI. Thus, the objective must be performed with a fixed number of sensors. The air deployment may introduce uncertainty in the final sensor positions. These limitations motivate the establishment of a planning system that optimizes the WSN deployment process.

In the field of signal processing and communication Adaptive Filtering has a tremendous application such as non-linear system identification, forecasting of time-series, linear prediction, channel equalization, and noise cancellation. Adaptive digital filtering self-adjusts its transfer function to get an optimal model for the unknown system based on some function of error based on the output of the adaptive filter and the unknown system. To get an optimal model of the unknown system it depends on the structure, adaptive algorithm strategy and the nature of input signal.

System Identification estimates models of dynamic systems by observing their input output response when it is difficult obtain the mathematical model of the system.

DSP-based equalizer systems have become ubiquitous in many diverse applications including voice, data, and video communications via various transmission media. Typical applications range from acoustic echo cancellers for full-duplex speakerphones to video de-ghosting systems for terrestrial television broadcasts to signal conditioners for wire line modems and wireless telephony. The effect of an equalization system is to compensate for transmission-channel impairments such as frequency-dependent phase and amplitude distortion. Rather for correcting for channel frequency-response ambiguity, cancel the effects of Multipath signal and to reduce the inter-symbol interference. So, construction of Equalizer to work for the above specifications is always a challenge and an active field of research.

On-line system identification or identification of complex systems is a major area of research from last several years. To abstract a new solution to some long standing necessities of automatic control and to work with more and more complex system to satisfy stricter design criteria and to fulfill previous points with less and less a priori knowledge of the unknown system. In this context a great effort is being made within the system identification towards the development of nonlinear models of real processes with less no of mathematical complexity, less no of input sample, faster matching and better convergence. This has been verified by MATLAB simulation version 2013.

1.2 THESIS LAYOUT

Chapter 2 describes the fundamental of incremental adaptive strategies in distributed network with practical application. it describes the number of algorithm like incremental adaptive solution, incremental steepest descent solution, quality aware incremental adaptive solution etc.It also provides the mathematical analysis to estimate the parameter of interest and effect of noisy nodes on the performance of incremental adaptive algorithm. Number of simulation results carried out individually to compare the performance of incremental adaptive solution with steepest descent solution by considering the both case (noisy node and non-noisy node).Some cases where the variance of noise in node is more than that of one on that case a quality aware DILMS (distributed incremental LMS algorithm) is applicable to improve the steady state performance of the algorithm. Hence this chapter provides a brief idea of effect of noisy node on the performance and perform a simulation to show how the Quality incremental LMS algorithm improves the performance with noisy node.

In chapter 3, the transform domain incremental adaptive strategy is describe and also focus on the RLS (recursive least square algorithm), GN (Gaussian Newton) algorithm. The convergence of LMS algorithm totally depends on the Eigen value and Eigen value spread of the auto correlation matrix. Small Eigen value slower the convergence rate and large value effects on the stability, hence for better convergence all the Eigen value of the autocorrelation matrix of input regressor should be same [5]. To make it we should design a pre-whiten filter which is not possible practically. Hence how we will achieve same without using the pre-whiten filter is describe in chapter 3. It gives the brief idea about the unitary transformation and its effect on the performance.

Chapter 4 describes how higher error order algorithm like LMF,NLMF,XE-NLMF and variable XE-NLMF algorithm outperforms than that of LMS algorithm under both Gaussian and Sub-Gaussian noise case. It also provide few mathematically analysis for the convergence analysis of the algorithm. Simulations are performed to compare the higher order error algorithm with the standard LMS algorithm using incremental method of cooperation.

Chapter 5 describes the CLMS (convex LMS) algorithm using incremental method. Generally fast filter gives faster convergence and accurate filter gives better steady state error performance. It is very difficult to design a filter which gives both. CLMS algorithm designs which consists of both the filter connected either in series or parallel to track the both response for different SNR case.

Chapter 2 INCREMENTAL ADAPTIVE STRATEGIES OVER DISTRIBUTED NETWORK

In Distributed processing number of nodes are distributed in a geographical area, it extract the information from data collected at nodes. For example nodes distributed in a geographical area collects some noisy information related to a certain parameter, than share it with their neighbor by some defined network topology, the aim is to reach the required parameter of interest. The objective is to reach the exact parameter of interest and it should same as it outcome from the nodes estimation in the geographical area. In a comparison Distributed solution is better than that of centralized solution because in centralized solution a central processor is required, nodes collect noisy information than send it to the central processor for process, central processor process the data than send back to all nodes. Hence for this a heavy communication between node and central processor required and a powerful central processor also required, but in distributed solution, the nodes only depends upon their local data and an interaction with the immediate neighbors [2]. Distributed solution reduces the amount of processing and communication ([1], [3]).

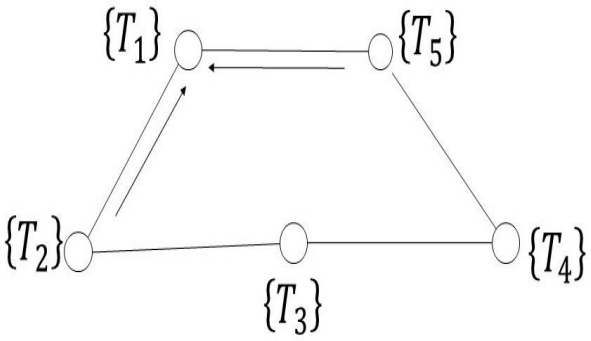

Fig. 1 Distributed network

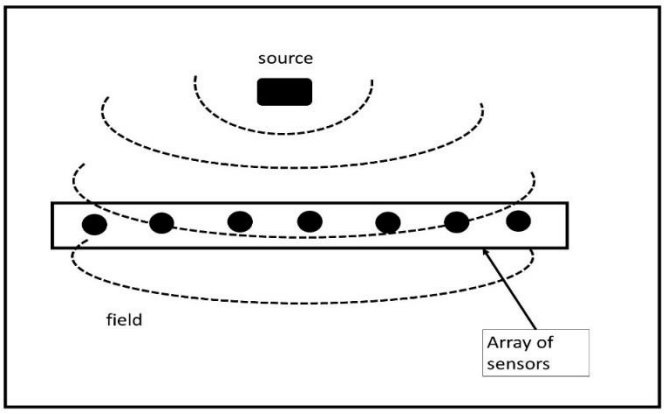

Fig. 2 monitoring a diffusion phenomenon by a network of sensors

2.1 Applications

Consider there are N number of nodes are distributed in a geographical area as shown in Fig.1. Each node collect some noisy temperature measurementsT_i. The main goal is to give all the node information about the average temperature $\bar{T}$. This can be possible by using the distributed solution known as consensus implementation, which states that one node measurement combines with the measurement of the immediate neighbor node and the outcome become the nodes new measurement.i.e. For node 1 we can write that

$$x_1(i) \leftarrow \alpha_1 x_1(i-1) + \alpha_2 x_2(i-1) + \alpha_5 x_5(i-1)(node\ 1)$$

Where $x_1(i)$ update measurement for node 1and α's are appropriately chosen coefficients. Similarly we can apply the same update process to other nodes and repeat the process. By suitably choosing α and network topology all the node finally converge to desired average temperature $\bar{T}$.

Another Application is it is also very useful to monitor the concentration of a chemical in air or water by collecting the measurements in time and space by number of sensors as shown in Fig.2. The measurements collected from number of sensors used to estimate the parameter $\{\theta_1, \theta_2, \theta_3\}$ that calculate the concentration of chemical in the environment by some diffusion equation with some boundary condition. e.g.,

$$\frac{\partial c(x,t)}{\partial t} = \theta_1 \frac{\partial^2 c(x,t)}{\partial x^2} + \theta_2 \frac{\partial c(x,t)}{\partial x} + \theta_3 c(x,t) + u(x,t)$$

Where $c(x,t)$ indicates the concentration at location x at time t [7]. Another Application of distributed processing is to monitoring the moving target by collecting the signal from different sensors, with the help of the sensors we can find the presence of the target and we can also track its trajectory [4].

Distributed network links to pc, laptop, cell phones and sensors forms backbone for future data communication and Network.

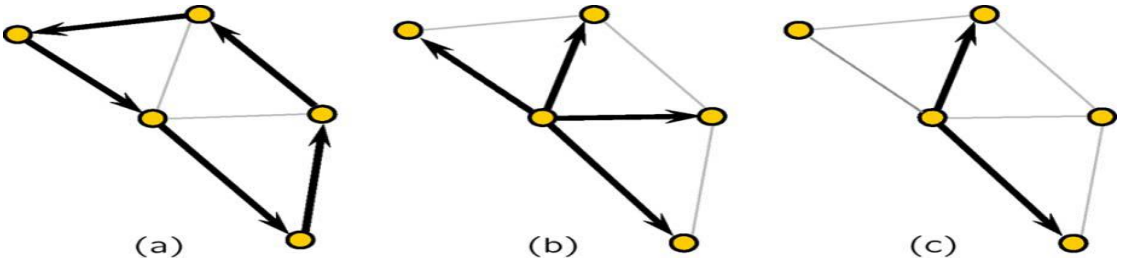

Fig. 3 three mode of cooperation (a) incremental (b) diffusion (c) probabilistic diffusion

2.2 Modes of cooperation

The successes of any Distributed Network depends upon the mode of cooperation that used among the nodes. There are three mode of cooperation as shown in Fig.3. In an incremental mode of cooperation the information flows in one direction from one node to adjacent node. Incremental mode of cooperation follows a cyclic pattern among the nodes, and it requires least amount of power and communication [8], [9], [10]. In diffusion mode of communication the information flows to all the nodes connected to that node where information starts to communicate, it requires more power and communication than that of Incremental mode of cooperation. It is complex than that of incremental mode of cooperation. In case of incremental mode of cooperation if one node is failed than we cannot get the information that is the network fails to transmit the information, which is one of the disadvantage of incremental mode of cooperation but this problem can be solved in diffusion mode of cooperation because if one node failed than we can collect information from any of its connected node, since the information flows to all the connected node in case of diffusion mode of cooperation. But the design of Diffusion mode of cooperation is more complex

than that of incremental mode of cooperation and also it requires more power and communication than that of incremental mode of cooperation. In case probabilistic mode of cooperation the information flows to subset of number of nodes that is connected to a particular node .It also require more power and communication than that of incremental mode of cooperation. Here I used Incremental mode of cooperation for all my work.

2.3 Consensus strategy

The temperature example explain in section 2.2 represents the consensus strategy. Consensus strategy states that first every node collects noisy information and update itself to reach an individual decision about a parameter of interest. During updating period each node act as an individual agent i.e. there is no interaction with the other node, then according to consensus strategy all the node combines their estimates to converge asymptotically to the desired global parameter of interest [2].

Let consider another example to understand the consensus strategy properly. Let each node has a data vector y_k and a data matrixH_k. For some unknown vector w^0 the noisy and distorted measurement y_k is given by

$$y_k = H_k w^0 + v_k$$

Each node estimate for w^0 by using its local data $\{y_k, H_k\}$ .for estimate, the node should evaluate the local cross correlation vector $\theta_k = {H_k}^* y_k$ and its autocorrelation matrix$R_k = {H_k}^* H_k$. Then, the local estimate for w^0 can be found from $\widehat{w}_k = {R_k}^{-1}\theta_k$ .similarly each node should estimate its local estimation, then a consensus iteration apply to all node to calculate $\hat{R}$ and $\hat{\theta}$ defined by as follows

$$\hat{R} = \frac{1}{N}\sum_{k=1}^{N} R_k \quad \text{And} \quad \hat{\theta} = \frac{1}{N}\sum_{k=1}^{N} \theta_k$$

A global estimate of w^0 is given by$\widehat{w} = \hat{R}^{-1}\hat{\theta}$. For all practical proposes, a least square implementation is an offline or non-recursive solution. A difficulty is come when one particular node collect one more data and updating for the optimal solution w^0 without repeating the prior process and iteration. The offline averaging limits the consensus solution, especially when the network having limited communication resources [2].

2.4 Contribution

When consider the forgoing issues (real time adaption with environment, low computation and communication complexity), we consider a Distributed LMS (least mean square) algorithm, since the computational complexity is less for both computation and communication. This algorithm solves the problem of new entry of data, it responds the data and also update it. The advantage of distributed algorithm than that of consensus strategy is it does not require of intermediate averaging as is done in consensus strategy. It also not required two different time scales. The distributed adaptive solution is the advance version or extension of adaptive filter, it is totally model independent i.e. it can be used without any knowledge of statistics of data. Generally adaptive filter responds to real time data and varies with statistical properties of data, distributed algorithm just extend this property to network domain [2]. The main purpose of this algorithm is:

1) Using distributed adaptive algorithm optimization technique to inspire the family of incremental adaptive algorithm [11].
2) Using incremental algorithm develop an interconnected network such that it is able to respond the real time data and also shows adaptive nature in variation with the statistical properties of the data as follow:
    a) Each time node receives a new information and that information is used by node to update its local estimation parameter of interest.
    b) After local estimation finished, the estimated parameter share with the immediate neighbors of node and repeat the same process to the other node in the network.
3) Distributed processing task is challenging, since it contain “system of systems” ,that process the data cooperatively manner both in time and space. In distributed algorithm different nodes converge at different MSE (mean square error) levels, which reflects the statistical diversity of data and the different noise levels [2].

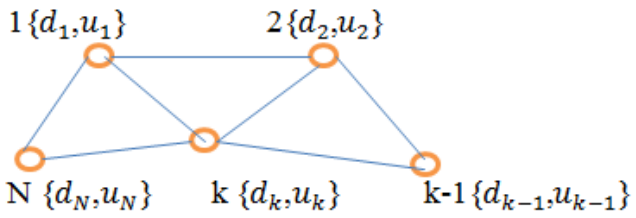

Fig. 4 Distributed network with N nodes accessing space time data

2.5 ESTIMATION PROBLEM AND ADAPTIVE DISTRIBUTED SOLUTION

There has been lots of work we can found in the literature solving distributed optimization problem using incremental method. In distributed algorithm a cost function can be decomposes into sum of individual cost functions using incremental procedure. The procedure can be explained below in the context of MSE.

Consider a network with N nodes as shown in Fig.4. Each node has access to time realizations $\{d_k(i), u_{k,i}\}$ of zero mean spatial data$\{d_k, u_k\}$,$k = 1,2,\cdots,N$, where d_k is a scalar and u_k is a row regression vector of size $1\times M$.

$$U \triangleq col\{u_1, u_2, \dots, u_N\}(N \times M) \tag{2.5.1}$$

$$d \triangleq col\{d_1, d_2, \dots, d_N\}(N \times 1) \tag{2.5.2}$$

The above quantities collect data from all N nodes. The main objective is to estimate the vector w of size M× 1 that solves

$$\min_{w} \quad J(w) \tag{2.5.3}$$

Where $J(w)$ represents the cost function denotes the MSE, given as follows:

$$\text{J (w)} = \text{E}\|d - Uw\|^2$$

(2.5.4)

Where E is the expectation operator .The optimal solution w^0 can be found by using the othogonality condition given by

$$E\|d - Uw\|^2 = 0 \tag{2.5.5}$$

The solution to the above normal equation given by

$$R_{du} = R_u w^0 \tag{2.5.6}$$

Where R_u=E$U^* U$ $(M \times M)$, R_{du} =E$U^* d$=$\sum_{k=1}^{N} R_{du,k}$ (2.5.7)

But the solution obtained from equation (2.5.6) is not distributed in nature since for this solution we required to access the global information $\{R_u, R_{du}\}$ One way to do this is process it centrally than pass the information to all the nodes, but for this we require a heavy communication betweet node and central processor , also require huge amount of power.It also not adaptive in nature with respect to the environment. This is the reason why we go for the distributed solution, which reduces the communication burden and the amount of power required for communication [1].In this project we totally focus on the incremental mode of coperation, where each node produces its local estimation and share it with the immdeate neighbor node at a time.

2.5.1 Steepest Descent Solution

To work out distributed solution, the first fundamental knowledge of steepest descent required. Then apply it in the incremental solution. The cost function can be decomposes for each nodes given by:

$$\text{J}\ (w) = \sum_{k=1}^{N} J_k(w) \tag{2.5.8}$$

Where $J_k(w)$ is given by

$$J_k(w) \triangleq E|d_k - u_k w|^2 \tag{2.5.9}$$

$$= {\sigma_{d,k}}^2 - R_{ud,k} w - w^* R_{du,k} + w^* R_{u,k} w \tag{2.510}$$

And the second order quantities are defined by

$${\sigma_{d,k}}^2 = E|d_k|^2,\ R_{u,k} = E{u_k}^* u_k, \text{ and } R_{du,k} = E d_k {u_k}^* \tag{2.5.11}$$

The above explanation represents that J (w) can be expressed as sum of N different cost functions $J_k(w)$, one for each node k. the weight update equation used in the steepest descent solution for determining w^0 given by;

$$w_i = w_{i-1} - \mu[\nabla J(w_{i-1})]^* \, , w_{-1} = initial\ condition$$

$$= w_{i-1} - \mu \sum_{k=1}^{N} [\nabla J_k(w_{i-1})]^*$$

$$= w_{i-1} + \mu \sum_{k=1}^{N} \left(R_{du,k} - R_{u,k} w_{i-1}\right) \qquad (2.5.12)$$

Where $\mu > 0$ is properly chosen step size parameter, w_i is used to estimate w^0 at iteration i, and $\nabla J(w_{i-1})$ represents the gradient vector of $J(w)$ with respect to w calculated at w_{i-1} .For small value of μ ,$w_i \to w^0$ as $i \to \infty$ for using any initial condition.

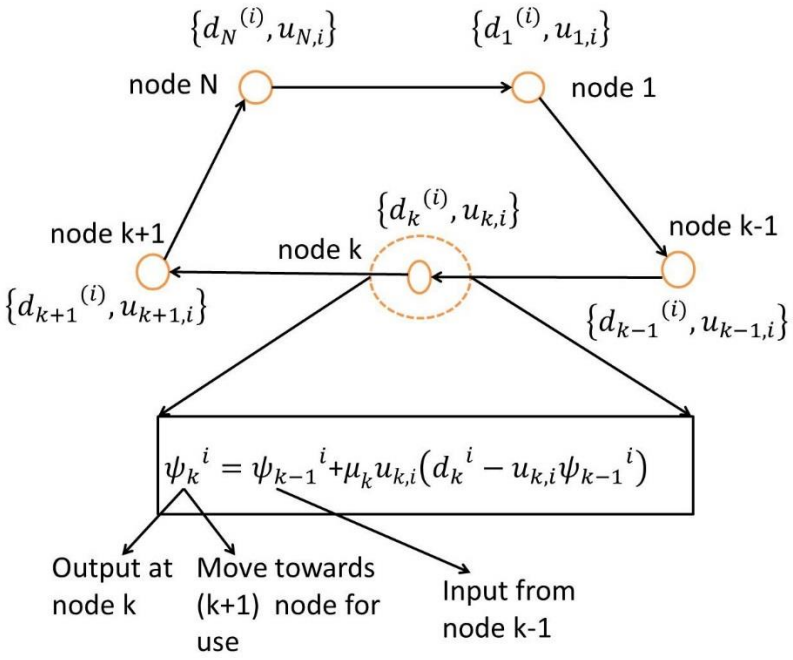

Fig. 5 Data processing in adaptive distributed structure

Consider a cycle define among nodes in such a way such that it visit every node once over the network topology and only access to its immediate neighbor as shown in Fig.5. Let $\psi_k{}^{(i)}$ represents the local estimate of w^0 at node k and at time i. Let assume that node k access data $\psi_{k-1}{}^{(i)}$, which is estimate of of w^0 at node k-1 and time i in the defined cycle, at each time instant i we start with initial condition $\psi_0{}^{(i)} = w_{i-1}$ at node 1(i.e. recent global estimate w_{i-1} for w^0), and process cyclically across the nodes, then at the end of the process we found that the local estimate at node

N will coincide at w_i from (2.5.12).i.e., ${\psi_N}^{(i)} = w_i$.In the other words the above implementation equivalent to:

$$\begin{cases} {\psi_0}^{(i)} = w_{i-1} \\ {\psi_k}^{(i)} = {\psi_{k-1}}^{(i)} - \mu_k[\nabla J_k(w_{i-1})]^*, \;\; k = 1,2,\cdots,N \\ w_i = {\psi_N}^{(i)} \end{cases} \tag{2.5.13}$$

In the steepest descent solution the iteration for ${\psi_k}^{(i)}$ over the spatial index k.

2.5.2 Incremental Steepest Descent Solution

The equation mentioned in equation (2.5.13) is cooperative in nature, since here each node k using information from immediate neighbor node for estimation process, still it is not pure cooperative in nature, because still each node require a global information w_{i-1} to calculate$\nabla J_k(w_{i-1})$. In order to make it totally cooperative in nature we have to use the incremental gradient algorithm. In incremental gradient algorithm each node uses the local estimate ${\psi_{k-1}}^{(i)}$ from node k-1 to find the partial gradient$\nabla J_k(\cdot)$, as opposite to w_{i-1}. Then by using the incremental adaptive algorithm we can rewrite the equation (2.5.13) *as:*

$$\begin{cases} {\psi_0}^{(\mathrm{i})} = \mathrm{w_{i-1}} \\ {\psi_k}^{(\mathrm{i})} = {\psi_{k-1}}^{(\mathrm{i})} - \mu_k\left[\nabla J_k\left({\psi_{k-1}}^{(\mathrm{i})}\right)\right]^*, \;\; \mathrm{k} = 1,2,\cdots,\mathrm{N} \\ \mathrm{w_i} = {\psi_N}^{(\mathrm{i})} \end{cases} \tag{2.5.14}$$

The above cooperative scheme represents a total distributed solution [2]. The above scheme shows that each node truly depends only upon its immediate neighbor for communication purpose, there is no global information required. That's why it saves both communication and energy resources.

2.5.3 Incremental Adaptive Solution

The incremental adaptive solution as shown in equation (2.5.14) depends on the cross correlation matrix and autocorrelation matrix $R_{du,k}$ and $R_{u,k}$,which is used to calculate the local gradients∇J_k. An adaptive incremental solution (2.5.14) can be used to replacing the second order moments $\{R_{du,k}, R_{u,k}\}$ by some approximation as follows [2]:

$$R_{du,k} \approx d_k(i){u_{k,i}}^*, R_{u,k} \approx {u_{k,i}}^* u_{k,i} \tag{2.5.15}$$

By using the data $\{d_k(i), u_{k,i}\}$ at time i, the equation given in (2.5.15) lead to an adaptive distributed incremental algorithm, or simply a distributed incremental LMS algorithm of the following form:

For each time $i \geq 0$, repeat:

K=1,$\cdots, N$

$$\begin{cases} \psi_0{}^{(i)} = w_{i-1} \\ \psi_k{}^{(i)} = \psi_{k-1}{}^{(i)} - \mu_k\left[\nabla J_k\left(\psi_{k-1}{}^{(i)}\right)\right]^*, \ k = 1,2,\cdots,N \\ w_i = \psi_N{}^{(i)} \end{cases} \tag{2.5.16}$$

The operation of algorithm given in (2.5.16) well explained in the Fig.5. At each time i the node uses its local data $\{d_k(i), u_{k,i}\}$ and the estimated weight $\psi_{k-1}{}^{(i)}$ taken from its adjacent node to perform the following three tasks:

1) Calculate the local error quantity:$e_k(i) = d_k(i) - u_{k,i}\psi_{k-1}{}^{(i)}$;
2) Update the weight by using the equation:$\psi_k{}^{(i)} = \psi_{k-1}{}^{(i)} + \mu_k e_k(i)$;
3) Pass the update weight information of node k to the neighbor node k+1.

2.6 PERFORMANCE ANALYSIS

It is important to know how the incremental adaptive solution works. The study of interconnected node is very challenging because of the following reasons:

1) Each node distributed in the geographical area must influence by statistics of its local data $\{R_{du,k}, R_{u,k}\}$.
2) Each node distributed in the geographical area influence by their neighbor through the incremental mode of cooperation.
3) Each node distributed in the geographical area also influence by the local noise with variance$\sigma_{v,k}{}^2$.

In steady state number of nodes distributed in the geographical area affected by the whole network and also somewhere affected by the local statistics of the data. When the step size decreases asymptotically than both the quantities MSD (mean square deviation), EMSE (excess mean square error) approach zero for every node in the network [2].

In order to perform the performance analysis we should go through the energy conservation relation. We have to apply the energy conservation relation for space dimension, since distributed adaptive algorithm involves both space and time variable. In the network number of nodes are distributed and each node can stabilize at individual MSE value, hence energy conservation relation can flow across interconnected filters. In order to calculate the individual node performance, weighting will be used to decouple the equation and calculate the estimated quantity of interest in steady states [2].

2.6.1 Data Model and Assumption

To do the performance analysis the data model and assumption is needed for adaptive algorithm. The data model and assumption for the data model $\{d_k(i), u_{k,i}\}$ is given by

1) The desired unknown vector w^0 relates $\{d_k(i), u_{k,i}\}$ as

   $$d_k(i) = u_{k,i}w^0 + v_k(i) \quad (2.6.1)$$

   Where $v_k(i)$ is white noise sequence with variance ${\sigma_{v,k}}^2$ and independent of$\{d_l(j), u_{l,j}\}$;
2) $u_{k,i}$ is independent of $u_{l,i}$ for k$\neq l(spatial\ independence)$;
3) $u_{k,i}$ is independent of $u_{k,j}$ for $i \neq j$ $(time\ independence)$.

The model given in (2.6.1) used in different application and here it is used to estimate the unknown vector w^0, this is referred to as the stationary model. Here we can study for only the stationary case, the distributed adaptive algorithm (2.5.16) can also useful to study for the non-stationary case. For simplification purpose we assume the regressor as spatially and temporal independent.

2.6.2 Weighted Energy Conservation Relation

Weight error vector at time i ${\tilde{\psi}_k}^{(i)} \triangleq w^0 - {\psi_k}^{(i)}$ (2.6.2)

A priori error $e_{a,k}(i) \triangleq u_{k,i}{\tilde{\psi}_{k-1}}^{(i)}$

(2.6.3)

A posterior error $e_{p,k}(i) \triangleq u_{k,i}{\tilde{\psi}_k}^{(i)}$ (2.6.4)

Output error $e_k(i) \triangleq d_k(i) - u_{k,i}{\psi_{k-1}}^{(i)}$ (2.6.5)

The vector $\tilde{\psi}_k^{(i)}$ measures the difference between the weight estimated at node k and the optimum weight w^0. The signal $e_k(i)$ represents the estimation error, the estimation error related to the a priori error by using the data model (2.6.1) as

$$e_k(i) = d_k(i) - u_{k,i}\psi_{k-1}{}^{(i)} = u_{k,i}w^0 + v_k(i) - u_{k,i}\psi_{k-1}{}^{(i)}$$

$$= e_{a,k}(i) + v_k(i) \tag{2.6.6}$$

Now $E|e_k(i)|^2 = E\left|e_{a,k}(i)\right|^2 + \sigma_{v,k}{}^2$ (2.6.7)

The interested parameter such as MSD (mean square deviation), MSE (mean square error) and the EMSE (excess mean square error) can be evaluate at steady state as follows:

$$\eta_k \triangleq E\left\|\tilde{\psi}_{k-1}{}^{\infty}\right\|^2 (MSD) \tag{2.6.8}$$

$$\varsigma_k \triangleq E\left|e_{a,k}(\infty)\right|^2 (EMSE) \tag{2.6.9}$$

$$\xi_k \triangleq E|e_k(\infty)|^2 = \varsigma_k + \sigma_{v,k}{}^2 (MSE)$$
(2.6.10)

The weight norm for a vector x and a Hermitian positive definite matrix $\Sigma > 0$ is given by $\|x\|_\Sigma{}^2 \triangleq x^*\Sigma x$. Then, under the assumed data condition we have

$$\eta_k = E\left\|\tilde{\psi}_{k-1}{}^{(\infty)}\right\|_I^2, \varsigma_k = E\left\|\tilde{\psi}_{k-1}{}^{(\infty)}\right\|_{R_{u,k}}^2 \tag{2.6.11}$$

The weighted a priori and a posteriori local error signal for each node k given by:

$$e_{a,k}{}^{\Sigma}(i) \triangleq u_{k,i}\Sigma\tilde{\psi}_{k-1}{}^{(i)} \; and \; e_{p,k}{}^{\Sigma}(i) \triangleq u_{k,i}\Sigma\tilde{\psi}_k{}^{(i)} \tag{2.6.12}$$

Where Σ Hermitian positive definite matrix, can be chosen freely. Using algorithm (2.5.16) subtracting w^0 On both side we get;

$$\tilde{\psi}_k{}^{(i)} = \tilde{\psi}_{k-1}{}^{(i)} - \mu_k u_{k,i}{}^* e_k(i) \tag{2.6.13}$$

Multiplying (2.6.13) both side from left by $u_{k,i}\Sigma$ then we get;

$$u_{k,i}\Sigma\ \tilde{\psi}_k^{(i)} = u_{k,i}\Sigma\ \tilde{\psi}_{k-1}^{(i)} - \mu_k \left\|u_{k,i}\right\|_\Sigma^{\ 2} e_k(i) \tag{2.6.14}$$

From (2.6.12) we get

$$e_{p,k}{}^{\Sigma}(i) = e_{a,k}{}^{\Sigma}(i) - \mu_k \left\|u_{k,i}\right\|_\Sigma^{\ 2} e_k(i) \tag{2.6.15}$$

From (2.6.15) we get

$$e_k(i) = \frac{1}{\mu_k \left\|u_{k,i}\right\|_\Sigma^{\ 2}} \left(e_{a,k}{}^{\Sigma}(i) - e_{p,k}{}^{\Sigma}(i)\right) \tag{2.6.16}$$

Substituting (2.6.16) into (2.6.13) and rearranging terms, we get

$$\tilde{\psi}_k^{(i)} + \frac{u_{k,i}{}^{*} e_{a,k}{}^{\Sigma}(i)}{\left\|u_{k,i}\right\|_\Sigma^{\ 2}} = \tilde{\psi}_{k-1}^{(i)} + \frac{u_{k,i}{}^{*} e_{p,k}{}^{\Sigma}(i)}{\left\|u_{k,i}\right\|_\Sigma^{\ 2}} \tag{2.6.17}$$

Equating the weighted norms of both side, the cross terms are cancelled out and the energy terms are

$$\left\|\tilde{\psi}_k^{(i)}\right\|_\Sigma^{\ 2} + \frac{\left|e_{a,k}{}^{\Sigma}(i)\right|^2}{\left\|u_{k,i}\right\|_\Sigma^{\ 2}} = \left\|\tilde{\psi}_{k-1}^{(i)}\right\|_\Sigma^{\ 2} + \frac{\left|e_{p,k}{}^{\Sigma}(i)\right|^2}{\left\|u_{k,i}\right\|_\Sigma^{\ 2}} \tag{2.6.18}$$

The above equation represents the space-time weighted energy conservation relation, which shows how energies of several variable related to each other in space and time.

Now by substituting (2.6.15) into (2.6.18) and rearranging terms we get;

$$\left\|\tilde{\psi}_k^{(i)}\right\|_\Sigma^{\ 2} = \left\|\tilde{\psi}_{k-1}^{(i)}\right\|_\Sigma^{\ 2} - \mu_k e_{a,k}{}^{\Sigma *} e_k - \mu_k e_k{}^{*} e_{a,k}{}^{\Sigma} + \mu_k{}^2 |u_k|_\Sigma{}^2 |e_k|^2 \tag{2.6.19}$$

Using (2.6.6) and taking expectation of both side we get

$$E\left\|\tilde{\psi}_k^{(i)}\right\|_\Sigma^{\ 2} = E\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_\Sigma^{\ 2} - \mu_k E e_{a,k}{}^{\Sigma *} e_k - \mu_k e_k{}^{*} E e_{a,k}{}^{\Sigma} + \mu_k{}^2 E |u_k|_\Sigma{}^2 \left|e_{a,k}\right|^2 \tag{2.6.20}$$

Using (2.6.12) and weighted error norm definition, we can expand the (2.6.20) in terms of regressor data and weighted error vector as follows:

$$E\left\|\tilde{\psi}_k^{(i)}\right\|_{\Sigma}^2 = E\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_{\Sigma}^2 - \mu_k E\tilde{\psi}_{k-1}^{*}\Sigma u_k^{*}u_k\tilde{\psi}_{k-1} - \mu_k E\tilde{\psi}_{k-1}^{*}u_k^{*}u_k\Sigma\tilde{\psi}_{k-1} +$$
$$\mu_k^2 E\tilde{\psi}_{k-1}^{*}u_k^{*}u_k\Sigma u_k^{*}u_k\tilde{\psi}_{k-1} + \mu_k^2\sigma_{v,k}^2 E\|u_k\|_{\Sigma}^2 \quad (2.6.21)$$

We know that $\|x\|^2_A + \|x\|^2_B = \|x\|^2_{A+B}$, by using this (2.6.21) can be rewritten as

$$E\left\|\tilde{\psi}_k^{(i)}\right\|_{\Sigma}^2 = E\left(\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_{\Sigma'}^2\right) + \mu_k^2\sigma_{v,k}^2 E|u_k|_{\Sigma}^2 \quad (2.6.22)$$

Where the term Σ' represents the stochastic weighted matrix given by

$$\Sigma' = \Sigma - \mu_k\left(u_k^{*}u_k\Sigma + \Sigma u_k^{*}u_k + \mu_k^2\|u_k\|_{\Sigma}^2 u_k^{*}u_k\right) \quad (2.6.23)$$

Since u_k is the independence regressor data we can write as

$$E\left(\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_{\Sigma'}^2\right) = E\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_{\mathrm{E}\Sigma'}^2 \quad (2.6.24)$$

Again rewrite (2.6.22) and (2.6.23) as

$$E\left\|\tilde{\psi}_k^{(i)}\right\|_{\Sigma}^2 = E\left(\left\|\tilde{\psi}_{k-1}^{(i)}\right\|_{\Sigma'}^2\right) + \mu_k^2\sigma_{v,k}^2 E|u_k|_{\Sigma}^2 \quad (2.6.25)$$

$$\Sigma' = \Sigma - \mu_k\left(u_k^{*}u_k\Sigma + \Sigma u_k^{*}u_k + \mu_k^2\|u_k\|_{\Sigma}^2 u_k^{*}u_k\right) \quad (2.6.26)$$

Where Σ' is now a deterministic matrix.

2.6.3 Gaussian Data

Equation (2.6.25) is represent as a spatial variance relation which allows as to perform the steady state performance for every node k. From (2.6.42) it is clear that Σ' totally regressor dependent, hence the study of the behavior of network depend on the following three parameter:

$$R_{u,k} = Eu_k^{*}u_k, \quad \mathrm{E}\|u_k\|_{\Sigma}^2 = Tr\left(R_{u,k}\Sigma\right), \text{ and } \mathrm{E}\|u_k\|_{\Sigma}^2 u_k^{*}u_k \quad (2.6.27)$$

But for the evaluation of $\mathrm{E}\|u_k\|_{\Sigma}^2 u_k^{*}u_k$ the input regressor should be non-Gaussian data, but initially we assume Gaussian data for simplicity, hence assume that $\{u_k\}$ arise from circular Gaussian distribution and introduce the Eigen decomposition $R_{u,k} = U_k\Lambda_k U_k^{*}$, where U_k is a unitary matrix and Λ_k is a diagonal matrix with eigen value of $R_{u,k}$.

Now let the transformed quantities are

$$\bar{\psi}_k \triangleq {U_k}^* \tilde{\psi}_k,\ \bar{\psi}_{k-1} \triangleq {U_k}^* \tilde{\psi}_{k-1},\ \bar{u}_k \triangleq u_{kU_k},\ \bar{\Sigma} \triangleq {U_k}^* \Sigma U_k,\ \bar{\Sigma}' \triangleq {U_k}^* \Sigma' U_k \tag{2.6.28}$$

We know that U_k is unitary and $\left\|\tilde{\psi}_{k-1}\right\|_{\Sigma}^{\ 2} = \left\|\bar{\psi}_{k-1}\right\|_{\bar{\Sigma}}^{\ 2}\ and\ \|u_k\|_{\Sigma}^{\ 2} = \|\bar{u}_k\|_{\bar{\Sigma}}^{\ 2}$,by using this (2.6.25) and (2.6.26) can be written in the form

$$E\left\|\bar{\psi}_k\right\|_{\bar{\Sigma}}^{\ 2} = E\left\|\bar{\psi}_{k-1}\right\|_{\bar{\Sigma}'}^{\ 2} + {\mu_k}^2 {\sigma_{v,k}}^2 E\|\bar{u}_k\|_{\bar{\Sigma}}^{\ 2} \tag{2.6.29}$$

$$\bar{\Sigma}' = \bar{\Sigma} - \mu_k E(\bar{u}_k{}^* \bar{u}_k \bar{\Sigma} + \bar{\Sigma} \bar{u}_k{}^* \bar{u}_k) + {\mu_k}^2 E\|\bar{u}_k\|_{\bar{\Sigma}}^{\ 2} \bar{u}_k{}^* \bar{u}_k \tag{2.6.30}$$

$$E\left\|\bar{\psi}_k\right\|_{\bar{\Sigma}}^{\ 2} = Tr(\Lambda_k \bar{\Sigma}) \text{ and } E\bar{u}_k{}^* \bar{u}_k = \Lambda_k \tag{2.6.31}$$

$$E\|\bar{u}_k\|_{\bar{\Sigma}}^{\ 2} \bar{u}_k{}^* \bar{u}_k = \Lambda_k Tr(\bar{\Sigma}\Lambda_k) + \gamma \Lambda_k \bar{\Sigma} \Lambda_k \tag{2.6.32}$$

$\gamma = 1$ For circular complex data and $\gamma = 2$ for real data. Now putting (2.6.31) and (2.6.32) into (2.6.29) and (2.6.30) we get

$$E\left\|\bar{\psi}_k\right\|_{\bar{\Sigma}}^{\ 2} = E\left\|\bar{\psi}_{k-1}\right\|_{\bar{\Sigma}'}^{\ 2} + {\mu_k}^2 {\sigma_{v,k}}^2 Tr(\Lambda_k \bar{\Sigma}) \tag{2.6.33}$$

$$\bar{\Sigma}' = \bar{\Sigma} - \mu_k(\Lambda_k \bar{\Sigma} + \bar{\Sigma}\Lambda_k) + {\mu_k}^2(\Lambda_k Tr(\bar{\Sigma}\Lambda_k) + \gamma \Lambda_k \bar{\Sigma} \Lambda_k) \tag{2.6.34}$$

We can choose $\bar{\Sigma}' and\ \bar{\Sigma}$ according to our wish, hence we can chose in such a way, such that both are become diagonal. Let we introduce the $M \times 1$ column vectors

$$\bar{\sigma} \triangleq diag\{\bar{\Sigma}\},\ \bar{\sigma}' \triangleq diag\{\bar{\Sigma}'\},\ \ \lambda_k \triangleq diag\{\Lambda_k\} \tag{2.6.35}$$

Where the $diag\{\ \ \}$ notation will be used in two ways first a diagonal matrix whose entries are the vector of λ and a vector containing main diagonal of Λ.

Using this concept (2.6.34) can be rewritten as

$$\bar{\sigma} = \left(I - 2\mu_k \Lambda_k + \gamma {\mu_k}^2 {\Lambda_k}^2\right)\bar{\sigma} + {\mu_k}^2 \left({\lambda_k}^T \bar{\sigma}\right)\lambda_k = \bar{F}_k \bar{\sigma} \tag{2.6.36}$$

Where the coefficient matrix $\bar{F}_k$ is defined by

$$\bar{F}_k \triangleq I - 2\mu_k \Lambda_k + \gamma {\mu_k}^2 {\Lambda_k}^2 + {\mu_k}^2 \lambda_k {\lambda_k}^T \tag{2.6.37}$$

The expression (2.6.33) becomes

$$E\left\|\bar{\psi}_k\right\|_{diag\{\bar{\sigma}\}}^{\ 2} = E\left\|\bar{\psi}_{k-1}\right\|_{diag\{\bar{F}_k\bar{\sigma}\}}^{\ 2} + {\mu_k}^2{\sigma_{v,k}}^2\left({\lambda_k}^T\bar{\sigma}\right) \tag{2.6.38}$$

$$E\left\|\bar{\psi}_k^{\ (i)}\right\|_{\bar{\sigma}_k}^{\ 2} = E\left\|\bar{\psi}_{k-1}^{\ (i)}\right\|_{\bar{F}_k\bar{\sigma}_k}^{\ 2} + {\mu_k}^2{\sigma_{v,k}}^2\left({\lambda_k}^T\bar{\sigma}_k\right) \tag{2.6.39}$$

2.6.4 Steady State Behavior

Let $\bar{p}_k \triangleq \bar{\psi}_k^{\ (\infty)}\ and\ g_k \triangleq {\mu_k}^2{\sigma_{v,k}}^2{\lambda_k}^T (a\ row\ vector)$. Then for $i \to \infty$, the equation (2.6.55) can be written as

$$E\|\bar{p}_k\|_{\bar{\sigma}_k}^{\ 2} = E\|\bar{p}_{k-1}\|_{\bar{F}_k\bar{\sigma}_k}^{\ 2} + g_k\bar{\sigma}_k,\ \ k = 1,2,\cdots,N \tag{2.6.40}$$

Now the performance measurement quantities are as follows:

$$\eta_k = E\|\bar{p}_{k-1}\|_q^{\ 2},\ \ q \triangleq diag\{I\}(MSD) \tag{2.6.41}$$

$$\zeta_k = E\|\bar{p}_{k-1}\|_{\lambda_k}^{\ 2},\ \ \lambda_k = diag\{\Lambda_k\}(EMSE) \tag{2.6.42}$$

$$\xi_K = \zeta_k + {\sigma_{v,k}}^2(MSE) \tag{2.6.43}$$

Now by iterating the (2.6.40) we can get a set of N coupled equalities

$$E\|\bar{p}_1\|_{\bar{\sigma}_1}^{\ 2} = E\|\bar{p}_1\|_{\bar{F}_1\bar{\sigma}_1}^{\ 2} + g_1\bar{\sigma}_1$$

$$E\|\bar{p}_2\|_{\bar{\sigma}_2}^{\ 2} = E\|\bar{p}_2\|_{\bar{F}_2\bar{\sigma}_2}^{\ 2} + g_2\bar{\sigma}_2$$

$$\vdots$$

$$E\|\bar{p}_{k-2}\|_{\bar{\sigma}_{k-2}}^{\ 2} = E\|\bar{p}_{k-3}\|_{\bar{F}_{k-2}\bar{\sigma}_{k-2}}^{\ 2} + g_{k-2}\bar{\sigma}_{k-2} \tag{2.6.44}$$

$$E\|\bar{p}_{k-1}\|_{\bar{\sigma}_{k-1}}^{\ 2} = E\|\bar{p}_{k-2}\|_{\bar{F}_{k-1}\bar{\sigma}_{k-1}}^{\ 2} + g_{k-1}\bar{\sigma}_{k-1}$$

$$\vdots$$

$$E\|\bar{p}_N\|_{\bar{\sigma}_N}^{\ 2} = E\|\bar{p}_{N-1}\|_{\bar{F}_N\bar{\sigma}_N}^{\ 2} + g_N\bar{\sigma}_N \tag{2.6.45}$$

By choosing $\bar{\sigma}_{k-2} = \bar{F}_{k-1}\bar{\sigma}_{k-1}$ and use in (2.6.44) we get

$$E\|\bar{p}_{k-2}\|^2_{\bar{F}_{k-1}\bar{\sigma}_{k-1}} = E\|\bar{p}_{k-3}\|^2_{\bar{F}_{k-2}\bar{F}_{k-1}\bar{\sigma}_{k-1}} + g_{k-2}\bar{F}_{k-1}\bar{\sigma}_{k-1} \quad (2.6.46)$$

$$E\|\bar{p}_{k-1}\|^2_{\bar{\sigma}_{k-1}} = E\|\bar{p}_{k-3}\|^2_{\bar{F}_{k-2}\bar{F}_{k-1}\bar{\sigma}_{k-1}} + g_{k-2}\bar{F}_{k-1}\bar{\sigma}_{k-1} + g_{k-1}\bar{\sigma}_{k-1} \quad (2.6.47)$$

By iterating in this way finally we get

$$E\|\bar{p}_{k-1}\|^2_{\bar{\sigma}_{k-1}} = E\|\bar{p}_{k-1}\|^2_{\bar{F}_k\cdots\bar{F}_N\bar{F}_1\cdots\bar{F}_{k-1}\bar{\sigma}_{k-1}} + g_k\bar{F}_{k+1}\cdots\bar{F}_N\bar{F}_1\cdots\bar{F}_{k-1}\bar{\sigma}_{k-1} +$$
$$g_{k+2}\bar{F}_{k+2}\cdots\bar{F}_N\bar{F}_1\cdots\bar{F}_{k-1}\bar{\sigma}_{k-1} + \cdots + g_{k-2}\bar{F}_{k-1}\bar{\sigma}_{k-1} + g_{k-1}\bar{\sigma}_{k-1} \quad (2.6.48)$$

We can define set of N matrix in the terms of product of $\bar{F}$ matrices

$$\Pi_{k,l} \triangleq \bar{F}_{k+l-1}\bar{F}_{k+l}\cdots\bar{F}_N\bar{F}_1\cdots\bar{F}_{k-1}, \quad l = 1,2,\cdots,N \quad (2.6.49)$$

Here the subscripts are all mod N. $\Pi_{k,l}$ work as an transition matrix for the weighting vector $\bar{\sigma}_{k-1}$ to reach node k cyclically through k-1, k-2, N-1,…, k. by using this we can rewrite equation(2.6.48) as

$$E\|\bar{p}_{k-1}\|^2_{(I-\Pi_{k,1})\bar{\sigma}_{k-1}} = a_k\bar{\sigma}_{k-1} \quad (2.6.50)$$

Where the row vector a_k is defined as

$$a_k \triangleq g_k\Pi_{k,2} + g_{k+1}\Pi_{k,3}\cdots + g_{k-2}\Pi_{k,N} + g_{k-1} \quad (2.6.51)$$

By choosing the weight vector $\bar{\sigma}_{k-1}$ as the solution of the linear equation$\left(I - \Pi_{k,1}\right)\bar{\sigma}_{k-1} = q$, we arrive at the desired expression for MSD

$$\eta_k = a_k\left(I - \Pi_{k,l}\right)^{-1}q \quad (MSD) \quad (2.6.52)$$

Similarly by choosing $\bar{\sigma}_{k-1}$ as the solution of $\left(I - \Pi_{k,1}\right)\bar{\sigma}_{\zeta,k-1} = \lambda_k$ we arrive at the desired response of EMSE

$$\zeta_k = a_k\left(I - \Pi_{k,l}\right)^{-1}\lambda_k \quad (EMSE) \quad (2.6.53)$$

$$\xi_k = \zeta_k + {\sigma_{v,k}}^2 \quad (MSE) \quad (2.6.54)$$

2.6.5 Simulation Results

Here for simulation we take N=20 number of nodes with M=10 taps of each local filter. We take 1000 iterations and perform 500 independent experiment to get the simulation result. The measurement data $d_k{}^{(i)}$ can be generated by using the data model (2.6.1) at each node and the vector w^0=col {1, 1,…,1}/$\sqrt{M}$, of size M×1.the background noise is white and Gaussian with $\sigma_v{}^2$=10^{-3}. The EMSE (Excess Mean square error), MSE (Mean square error) and MSD (Mean square deviation) can be plot by using $\left|u_{k,i}(\psi_k{}^{(i)}-\overline{w}^0)\right|^2$,$\left|d_k(i)-u_{k,i}\psi_{k-1}{}^{(i)}\right|^2$, $\left|(\psi_k{}^{(i)}-\overline{w}^0)\right|^2$. Here we consider each regressor of size (1×M) collecting data by observing a time-correlated sequence $\{u_k{}^{(i)}\}$, generated as

$u_k{}^{(i)}$=$\propto_k u_k{}^{(i-1)}$+$\beta_k z_k{}^{(i)}$, i>-∞

Here $\propto_k \in [0,1)$, is the correlation index and $z_k{}^{(i)}$ is a spatially Gaussian independent process with unit variance and β_k =$\sqrt{\sigma_{u,k}{}^2(1-\alpha_k{}^2)}$. The resulting regressor have Toeplitz covariance matrices $R_{u,k}$, with correlation sequence r_k(i) = $\sigma_{u,k}{}^2(\propto_k)^{|i|}$, $i=0,1,\cdots,M-1$. The input regressor power profile $\sigma_{u,k}{}^2 \in (0,1]$, the correlation index $\propto_k \in (0,1]$ and the Gaussian noise variance $\sigma_{v,k}{}^2 \in (0,0.1]$ chosen at random. The node power profile, correlation index, node noise power profile and SNR as shown in Fig.6, Fig.7 and Fig.8. The transient performance of MSE (mean square error), EMSE (excess mean square error) and MSD (mean square deviation) as shown in Fig.9.Fig.10, and Fig.11. Fig.12, Fig.13 and Fig.14 represents the MSE, EMSE and MSD performance node wise by taking average of last 300 experiments. The simulation results clear the fact that the convergence rate and steady state performance of incremental adaptive solution is better than that of steepest descent solution.

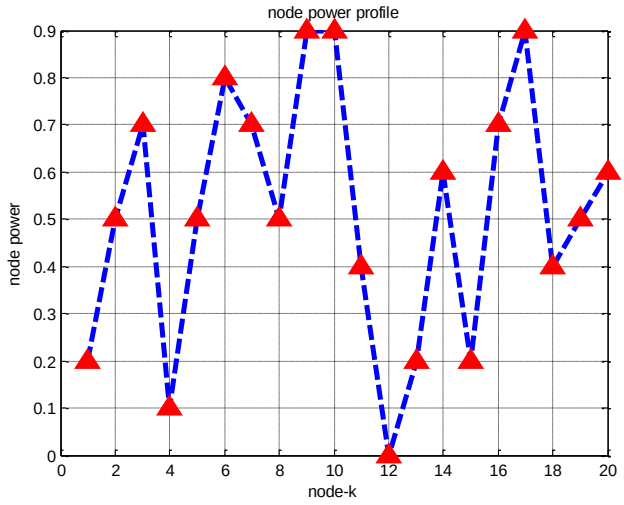

Fig. 6 Regressor power profile

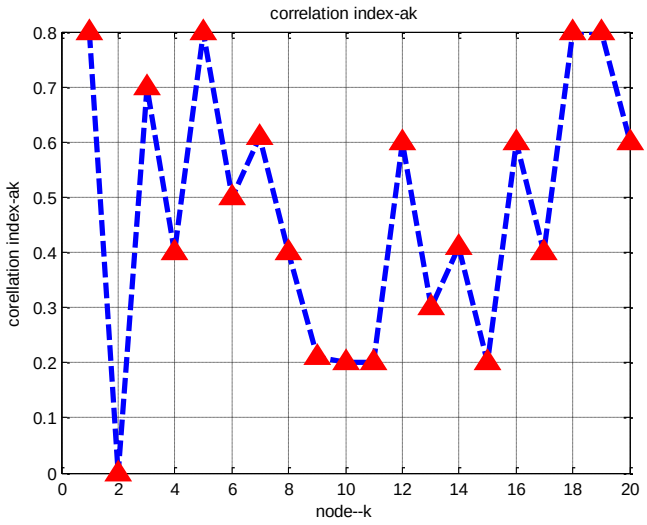

Fig. 7 Correlation index per node

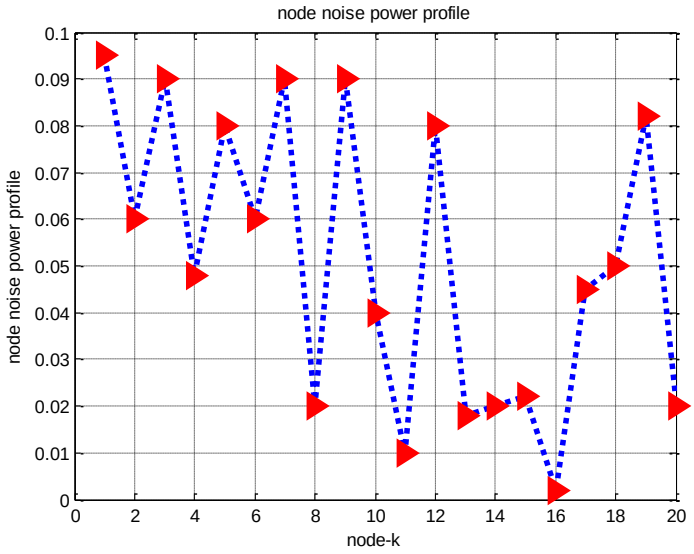

Fig. 8 Noise power profile

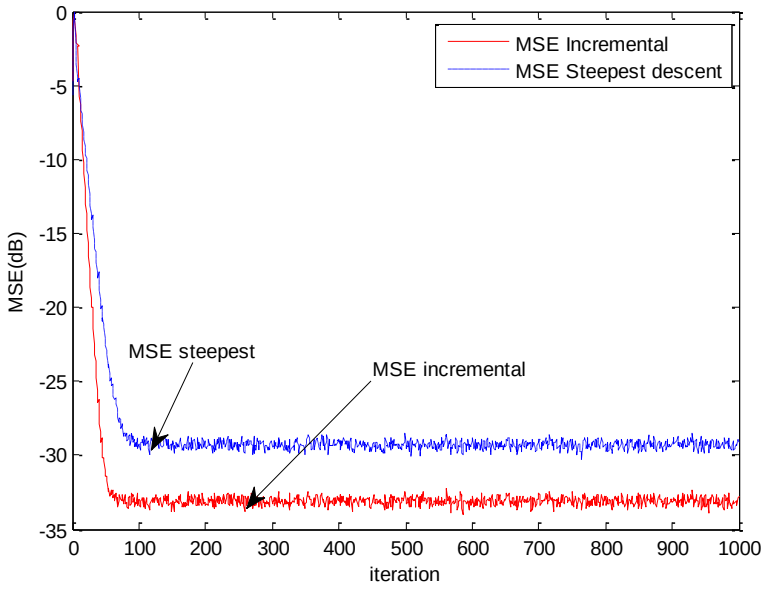

Fig. 9 Transient MSE performance at node 1for both incremental adaptive solution and stochastic steepest descent solution

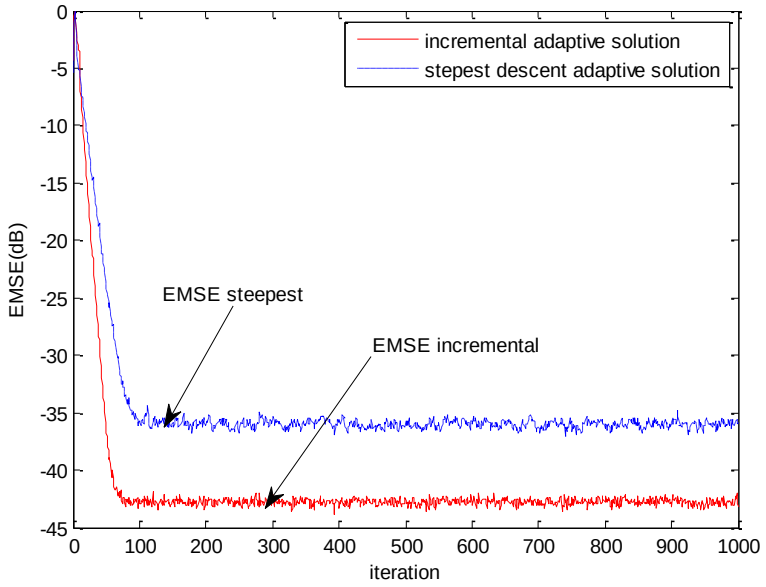

Fig. 10 Transient EMSE performance at node 1for both incremental adaptive solution and stochastic steepest descent solution

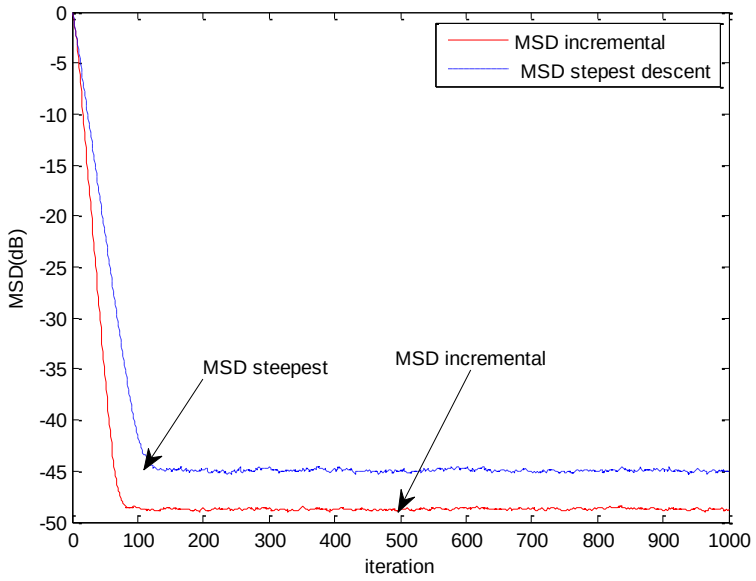

Fig. 11 Transient MSD performance at node 1for both incremental adaptive solution and stochastic steepest descent solution

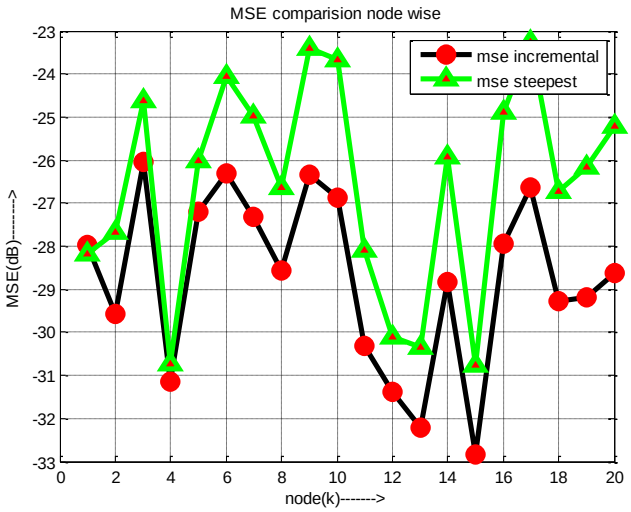

Fig. 12 MSE performance node wise

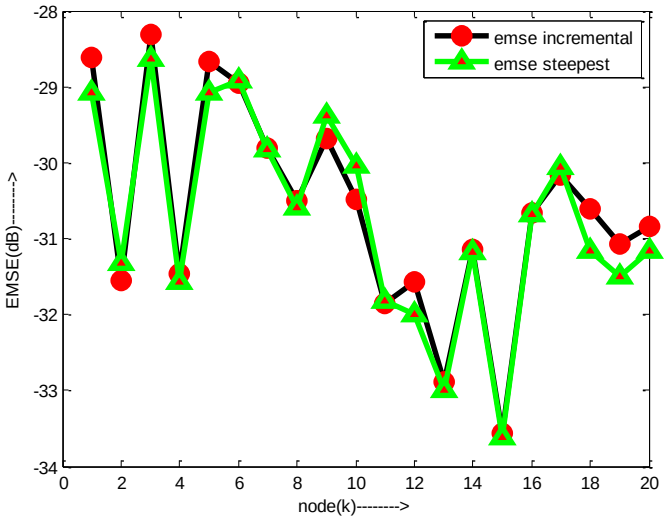

Fig. 13 EMSE performance node wise

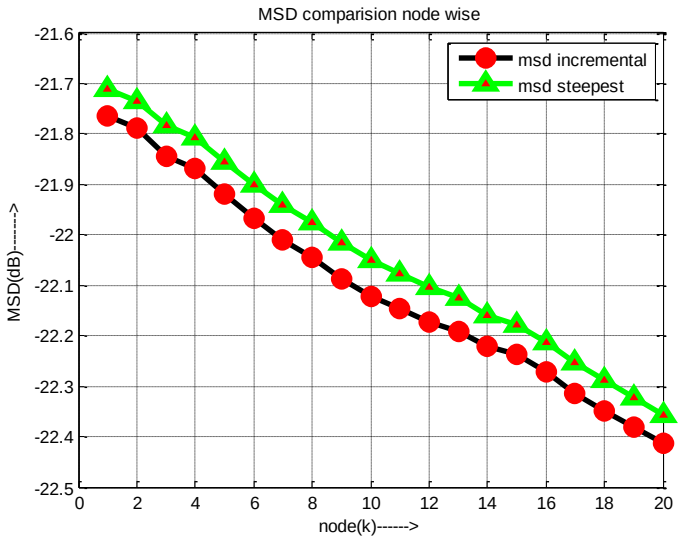

Fig. 14 MSD performance node wise

2.7 QUALITY AWARE INCREMENTAL LMS ALGORITHM FOR DISTRIBUTED ADAPTIVE ESTIMATION

In the incremental adaptive algorithm the node profile not considered, the performance can be detoriates if the SNR of some node is lower than the other node. To overcome this problem we can use the efficient step size for each node in the incremental least mean square adaptive algorithm. The aim behind this is to improve the robustness of the algorithm against the spatial variation of noise in the network. The algorithm provides improved steady state performance in comparison with the incremental LMS algorithm. Another method to achieve the requirement is assigning a suitable weight according to reliability of measurement. Initially we formulate the weight assigning as a constrained optimization problem, then recast it into distributed form and finally applied to adaptive solution problem [12]. Simulation result provide the performance of proposed algorithm.

2.7.1 Effect Of Noisy Nodes

Let there are some noisy nodes are present in the network having SNR at some node lower than that of the others. In this case the performance detoriates since in the incremental mode cooperation the nodes are connected in one direction i.e. the energy flow in one direction only, hence if one node found to be noisy then it effects the performance of other. Let consider there are N=20 number of nodes present in the network and assume that M=5, step size $\mu = 0.01$, $w^0 = {1_M}/{\sqrt{M}}$

.let we assume that there are five nodes are there having noise variance $\sigma_{v,k}{}^2 \epsilon(0,2)$ in the network and other node have noise variance$\sigma_{v,k}{}^2 \epsilon(0,0.1)$. Fig.15 and Fig.16 node profile of $\sigma_{v,k}{}^2$ and $Tr\{R_{u,k}\}$. Fig.17 shows the global average EMSE in both condition i.e. with present and absent of noisy node. The simulation result illustrate that the convergence rate and performance of DILMS algorithm without noisy nodes outperforms than that of DILMS algorithm with noisy nodes.

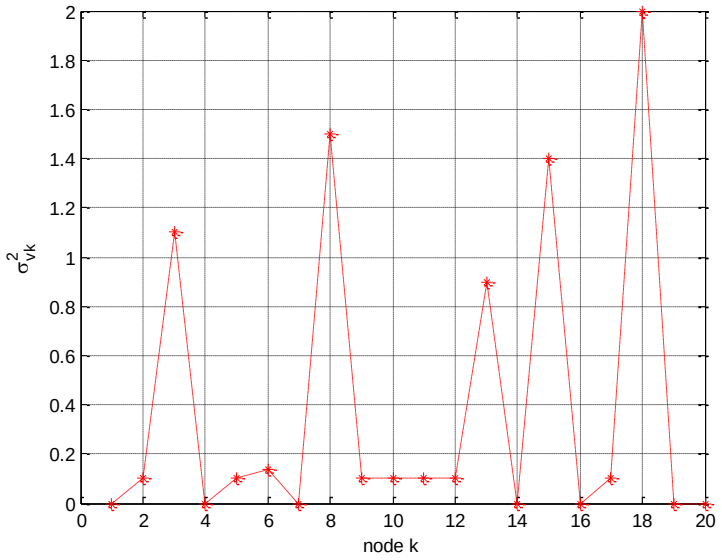

Fig. 15 The node profile of $\sigma_{u,k}{}^2$

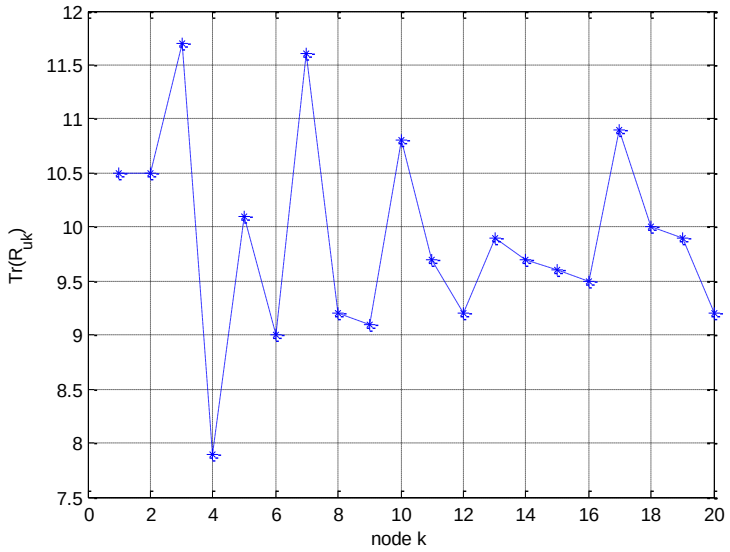

Fig. 16 The node profile of $Tr\{R_{u,k}\}$

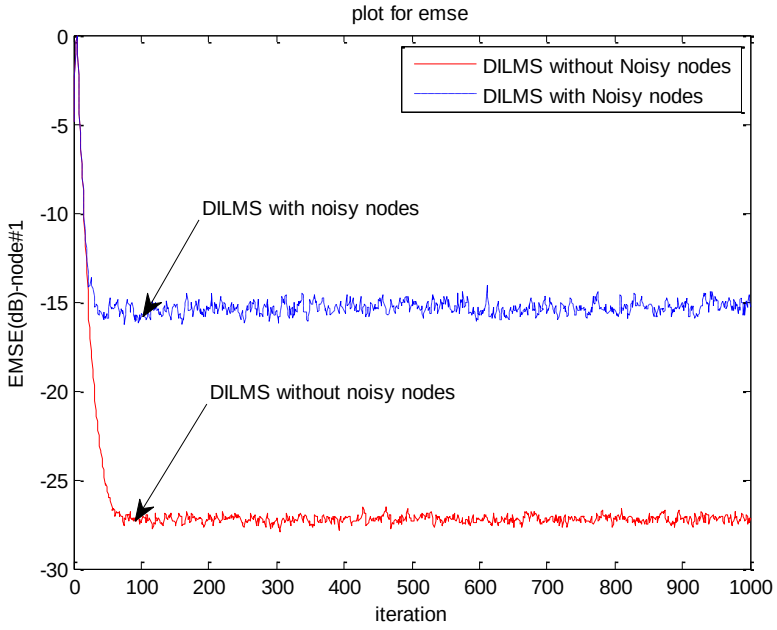

Fig. 17 The global average EMSE for DILMS algorithm in different condition

2.7.2 QWDILMS Algorithm

The block diagram of the QWDILMS algorithm as shown in Fig.18. The first step is to modify the DILMS algorithm mention at (2.5.16) as follows

$$\begin{cases} \phi_{k,i} = c_{k,k-1}\psi_{k-1,i} + c_{k,k}\psi_{k,i-1} \\ \psi_{k,i} = \phi_{k,i} + \mu_k u_{k,i}{}^{*}\left(d_k(i) - u_{k,i}\phi_{k,i}\right) \end{cases} \tag{2.7.1}$$

Where $\{c_{k,k-1}, c_{k,k}\}\epsilon\mathbb{R}$ are combination coefficients at node k. thus in modified version of DILMS algorithm each node updates with its local estimate from previous node i.e. $\psi_{k-1,i}$ and estimate from previous time i.e.$\psi_{k,i-1}$. Again it should be noted that for node k=1 we have $c_{1,0} = c_{1,N}$, for incremental mode of cooperation.

Now to formulate the problem of finding combination coefficients, let we define the $N \times 1$ vector $c_k = \left[c_{k,1}\ ,\ c_{k,2}, \cdots, c_{k,N}\right]^T \epsilon\mathbb{R}^N$ for every node k and$c_{k,l} = 0, l \neq \{k-1, k\}$. Now the local estimates for each node k is $\{\psi_{k,i},\ i = 0,1, \cdots\}$ realizable for some random vector ψ_k. Now we should found the coefficient vector $c_k \epsilon \mathbb{R}^N$, by solving the equation given below;

$$\begin{cases} \underset{\{c_1, c_2, \cdots, c_N\}\epsilon R^N}{minimize} \quad \sum_{k=1}^{N} E\left\{\left\|\psi_{c_k} - w^0\right\|^2\right\}, subject\ to\ c_{k,l} = 0\ \ for\ l \notin \mathcal{N}_k, \mathcal{N}_k = \{k-1, k\} \end{cases}$$

(2.7.2)

Where $\psi = [\psi_1, \psi_2, \cdots, \psi_N]\ is\ an\ M \times N$ random row matrix. Now we can write or decompose (2.7.2) at each node then the equation given by:

$$\begin{cases} \underset{c_k \in R^N}{minimize} J(c_k) = E\left\{\left\|\psi_{c_k} - w^0\right\|^2\right\} \\ subject\ to\ c_{k,l} = 0\ \ for\ l \notin \mathcal{N}_k, \mathcal{N}_k = \{k-1, k\} \end{cases} \tag{2.7.3}$$

It is difficult to solve the optimization problem directly, hence let we take an assume that every local estimate ψ_k is unbiased i.e. $E\{\psi_k\} = w^0$, hence we can say that $E\{\psi\} = w^0 1_N{}^T$.by using the bias variance decomposition we can write (2.7.3) as

$$J(c_k) = c_k{}^T Q_\psi c_k + \left\|\left(1_N{}^T c_k - 1\right)w^0\right\|^2 \tag{2.7.4}$$

Where Q_ψ is an $N \times N$ matrix defined by

$$Q_\psi = E\left\{(\psi - E\{\psi\})^{*}\left((\psi - E\{\psi\})\right)\right\} \tag{2.7.5}$$

Now by considering $1_N{}^T c_k = 1$ the second term of (2.7.4) totally eliminated and we can write (2.7.3) as

$$\underset{c_k \in R^N}{minimize}\ {c_k}^T Q_\psi c_k, subject\ to\ {1_N}^T c_k = 1\ and\ c_{k,l} = 0\ \ for\ l \notin \mathcal{N}_k \tag{2.7.6}$$

If we consider the DILMS algorithm mention in (2.5.16), then the dimension of the problem (2.7.1) reduced from N unknown to the cardinality of $\mathcal{N}_k$, say 2, by introducing the $N \times 2$ auxiliary variable

$$p_k = [\mathcal{T}_{k-1}, \mathcal{T}_k]_{N\times 2} \tag{2.7.7}$$

Where $\mathcal{T}_k$ is an $N \times 1$ vector whose all component zero except k, for example$\mathcal{T}_2 = [0\ \ 1\ \ 0\ \ \cdots\ \ 0]^T$. Since here we use the incremental method cooperation hence $P_1 = [\mathcal{T}_1, \mathcal{T}_N]$. Now any vector that satisfy $c_{kl} = 0\ for\ l \in \mathcal{N}_k$ can be represented as ([13], [14]);

$$c_k = P_k a_k\ , with\ a_k\ \in R^2 \tag{2.7.8}$$

Now using (2.7.8) in (2.7.6) we get

$$\begin{cases} \underset{c_k \in R^N}{minimize} f_k(\alpha_k) = {\alpha_k}^T Q_{\psi,k} \alpha_k \\ subject\ to\ a_k \in V_N = \{a \in R^2, {1_2}^T a = 1\} \end{cases} \tag{2.7.9}$$

Where $Q_{\psi,k}$ is the 2×2 matrix defined by

$$Q_{\psi,k} = {P_k}^T Q_\psi P_k \tag{2.7.10}$$

And $1_2 = {P_k}^T 1_N$ is a vector whose all components are 1. The solution to (2.7.9) well defined([13], [14]) given by

$${\alpha_k}^0 = \frac{{Q_{\psi,k}}^{-1} 1_2}{{1_2}^T {Q_{\psi,k}}^{-1} 1_2} \tag{2.7.11}$$

By applying similar techniques introduced in [13], we finally reach at the iterative solution

$$\begin{cases} b_{k,i} = b_{k,i-1} - \lambda_k(i) \Lambda Q_{\psi,k} b_{k,i-1} \\ c_k = P_k b_{k,i} \end{cases} \tag{2.7.12}$$

Where $\lambda_k(i)$ is the step-size and Λ is a 2×2 matrix given by

$$\Lambda = \begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \tag{2.7.13}$$

To derive an adaptive solution we can approximate $Q_{\psi,k}$ as follows

$$Q_{\psi,k} \approx (\Delta\emptyset)^*(\Delta\emptyset) \tag{2.7.14}$$

Where $(\Delta\emptyset)$ is a $M \times 2$ matrix given by

$$(\Delta\emptyset) = \begin{bmatrix} \psi_{k-1,i} - \psi_{k-1,i-1} & \psi_{k,i-1} - \psi_{k,i-2} \end{bmatrix} \tag{2.7.15}$$

Now replacing (2.7.14) in (2.7.12) and using the modified DILMS algorithm, our QWDILMS algorithm represent by

$$\begin{cases} g_{k,i} = \Lambda(\Delta\phi)^*(\Delta\phi)b_{k,i-1} \\ b_{k,i} = b_{k,i-1} - \lambda_k(i)g_{k,i} \\ c_k = P_k b_{k,i} \\ \phi_{k,i} = c_{k,k-1}\psi_{k-1,i} + c_{k,k}(i)\psi_{k,i-1} \\ \psi_{k,i} = \phi_{k,i} + \mu_k {u_{k,i}}^* \left(d_k(i) - u_{k,i}\phi_{k,i}\right) \end{cases} \tag{2.7.16}$$

Where $\lambda_k(i)$ is the normalized step size given by:

$$\lambda_k(i) = \gamma \frac{min\{b_{k,i-1}(1), b_{k,i-1}(2)\}}{\|g_{k,i}\|_\infty + \varepsilon} \tag{2.7.17}$$

$\gamma \in (0,1)$ and ε are constants, $\|\cdot\|_\infty$ represents the maximum norm and $b_{k,i-1}(m)$ m^{th} component of $b_{k,i-1}$.

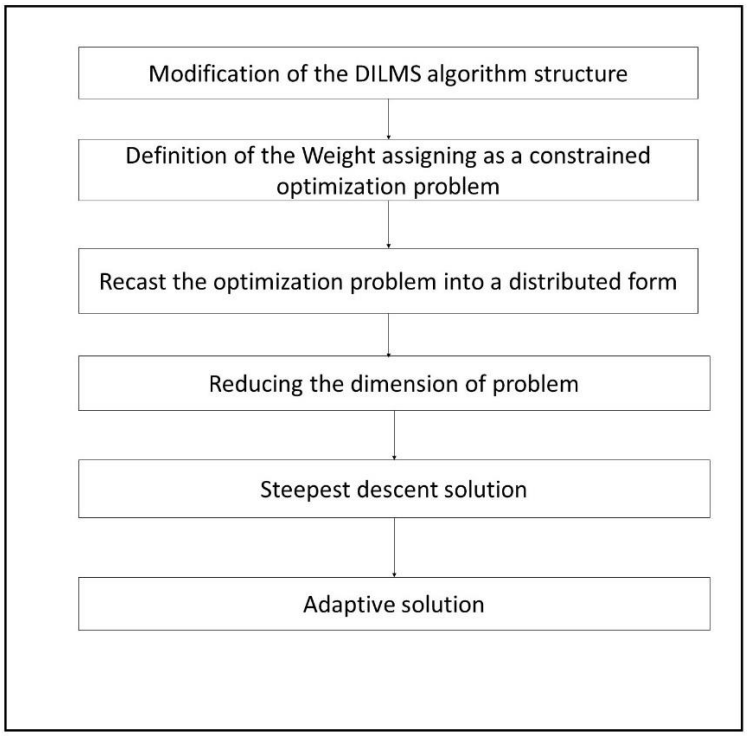

Fig. 18 Block diagram of proposed algorithm

Now we can perform a simulation result to study the MSD (mean square deviation) and EMSE (excess mean square) performance of the DILMS algorithm with noisy nodes, without noisy nodes and the QWDILMS algorithm. The average MSD can be defined as

$$MSD = \frac{1}{N}\sum_{k=1}^{N} E\left\{\left\|w^0 - \psi_{k-1,i}\right\|^2\right\}$$

Let $\gamma = 0.01\ and\ \varepsilon = 10^{-5}$. Here we formed 100 individual experiments and averaged. The EMSE and MSD performance of all the three algorithm shown in Fig.19 and Fig.20.

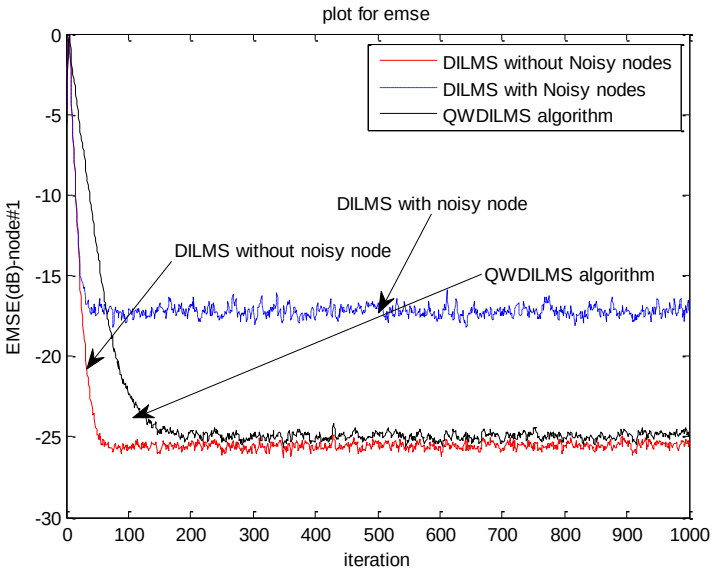

Fig. 19 The EMSE performance of DILMS algorithm with and without noisy nodes and QWDILMS Algorithm

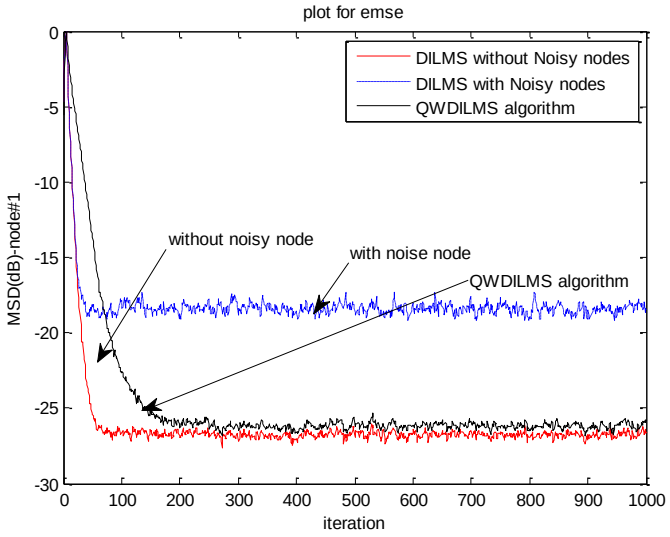

Fig. 20 The MSD performance of DILMS algorithm with and without noisy nodes and QWDILMS Algorithm

The simulation results reveals that the quality aware incremental LMS algorithm outperforms the DILMS algorithm in a sense of steady state performance and the performance of QWDILMS

algorithm increased about 7dB in comparison with the DILMS algorithm with noisy nodes. The disadvantage of the QWDILMS algorithm is its convergence rate. The advantage is the QWDILMS algorithm improves the robustness against the spatial variation SNR at different node.

2.7.3 Conclusion

Here we can draw two conclusion first the convergence rate and steady state performance of the incremental adaptive solution outperforms than that of the incremental steepest descent solution. Second one is in the case presence of noisy nodes having variance more than one, by applying quality aware incremental algorithm we can improve the steady state performance up to the level reached by the incremental algorithm without noisy nodes. The simulation result shows the effectiveness of the both DILMS and QWDILMS algorithm. We can also modify the QWDILMS algorithm structure to improve the convergence rate.

Chapter 3 FREQUENCY DOMAIN INCREMENTAL STRATEGIES OVER DISTRIBUTED NETWORK

The convergence rate of LMS (Least mean square) type filter is dependent on the Autocorrelation matrix of the input data and on the eigen value spread of the covariance matrix of the regressor data [5]. The mean square error (MSE) of an adaptive filter using LMS algorithm decreases with time as sum of the exponentials, whose time constants are inversely proportional to the eigen value of the auto correlation matrix of input data [15]. The smaller eigen value of autocorrelation matrix of the input results slower convergence mode and larger eigen values limit on the maximum learning rate that can be chosen without encountering stability problem. Best convergence and learning rate results when all the eigen values of the input autocorrelation matrix are equal i.e. Autocorrelation matrix should be in the form of a constant multiplication with the identity matrix [5].

Practically the input data's are colored and the Eigen values of autocorrelation matrix vary from smallest to the largest. The filter response can be improved by prewhitening the data, but for this the autocorrelation of the input data should be known. It is difficult to know the autocorrelation of the input data. It can be achievable by using unitary transformation, such as discrete cosine transform (DCT), discrete Fourier transform (DFT) etc. These transformation have de-correlation properties that improves the convergence performance of LMS for correlated input data [5].

Transform domain (which is also called frequency domain) can be applied in two ways one is block wise frequency domain algorithm other is non-block wise frequency domain algorithm [16]. In block wise frequency domain algorithm a block of input data is first transformed then input to the incremental LMS algorithm and in non-block or real time algorithm the data are continuously transformed by a fixed data-independent transform to de-correlate the input data [15]. DFT-LMS algorithm was first introduced by Narayan belongs to a simplest algorithm family because of the

exponential nature [17]. But in many practical situation it was found that DCT-LMS performs better than that of DFT-LMS and other transform domain [15].In this paper we interpret the incremental LMS using DCT/DFT algorithm and found that it produce better convergence and performance than previous

3.1 PREWHITENING FILTERS

The statistics of input data required for design of a Prewhiten filter. Let assume that the input sequence $\{u(i)\}$ is a zero mean and wide sense stationary, with autocorrelation function

$$r(k) = Eu(i)u^*(i-k), \quad k = 0, \pm 1, \pm 2, \cdots \tag{3.1.1}$$

To determine the prewhiten filter the knowledge of power spectrum and spectral factorization required. The Z-spectrum of wide sense stationary process $\{u(i)\}$ denoted by $S_u(z)$ and it is given by

$$S_u(z) = \sum_{k=-\infty}^{\infty} r(k)z^{-k} \tag{3.1.2}$$

For convergence r(k) should be exponentially bounded i.e.

$$|r(k)| \leq \beta a^{|k|} \tag{3.1.3}$$

For $\beta > 0 \; and \; 0 < |z| < 1$ the series (3.1.2) absolutely converge in the ROC: $a < |z| < a^{-1}$ i.e. it satisfies

$$\sum_{k=-\infty}^{\infty} |r(k)||z^{-k}| < \infty \; for \; all \;\; a < |z| < a^{-1} \tag{3.1.4}$$

Since the ROC includes the unit circle, so we can say that $S_u(z)$ cannot have poles on the unit circle. Now the power spectrum of the input regressor $\{u(i)\}$ is given by:

$$S_u\left(e^{jw}\right) = \sum_{k=-\infty}^{\infty} r(k)e^{-jwk} \tag{3.1.5}$$

We know that the power spectrum has two important property, first it is hermitian symmetry and second is it is nonnegative on the unit circle i.e. $S_u\left(e^{jw}\right) \geq 0 \; for \; 0 \leq w \leq 2\pi$.

The z transform satisfies the para Hermitian property i.e.

$$S_u(z) = \left[S_u\left(1/_{z^*}\right)\right]^* \tag{3.1.6}$$

If we replace z by $1/_{z^*}$ then again we get $S_u(z)$, which is nothing but the para Hermitian property. Now let $S_u(z)$ is a proper rational function and it does not have zeros on the unit circle so that

$$S_u\left(e^{jw}\right) > 0 \; for \; all - \pi \leq w \leq \pi \tag{3.1.7}$$

Than by using the para Hermitian property , we can say that for every pole(or zero) at a point ξ, there must exist a pole (or zero) at point $1/_{\xi^*}$, and there is no poles and zero's on unit circle [5]. Hence the rational number $S_u(z)$ can be expressed as

$$S_u(z) = {\sigma_u}^2 \frac{\prod_{l=1}^{m}(z-z_l)(z^{-1}-{z_l}^*)}{\prod_{l=1}^{n}(z-p_l)(z^{-1}-{p_l}^*)} \tag{3.1.8}$$

The $S_u(z)$ can be factorized in the form

$$S_u(z) = {\sigma_u}^2 A(z)\left[A\left(1/_{z^*}\right)\right]^* \tag{3.1.9}$$

Where $\{{\sigma_u}^2, A(z)\}$ satisfies the following condition

1. ${\sigma_u}^2$ is a positive scalar.
2. $A(z)$ is normalized to unity at infinity i.e. $A(\infty) = 1$.
3. $A(z)$ is a rational minimum phase function.

In order to meet the normalized condition $A(\infty) = 1$, we can take A(z) as

$$A(z) = z^{n-m} \frac{\prod_{l=1}^{m}(z-z_l)}{\prod_{l=1}^{n}(z-p_l)} \tag{3.1.10}$$

Let $\{x(i)\}$ be a widesense random process with z-transform $S_x(z)$ and assume that it feed to a stable system of transfer function H(z) shown in Fig.21 than the output in the z transform can be written as

$$S_y(z) = H(z)S_x(z)\left[H\left(1/_{z^*}\right)\right]^* \tag{3.1.11}$$

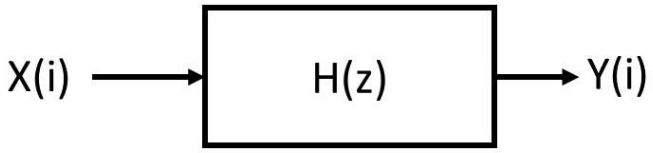

Fig. 21 filtering of wide sense stationary random process {x(i)} by a stable linear system H(z)

Now let we pass the input regressor $\{u(i)\}$ through a filter $1/[\sigma_u A(z)]$ as shown in Fig.22 then the output process denoted by $\{\bar{u}(\cdot)\}$, and the z spectrum of the output process given by

$$S_{\bar{u}}(z) = \frac{1}{\sigma_u A(z)}[\sigma_u{}^2 A(z)A^*(z^{-*})]\frac{1}{\sigma_u A^*(z^{-*})} = 1 \tag{3.1.12}$$

The autocorrelation of the sequence given by

$$\bar{r}(k) = E\bar{u}(i)u^*(i-k) = \begin{cases} 1\ if\ k = 0 \\ 0\ otherwise \end{cases} \tag{3.1.13}$$

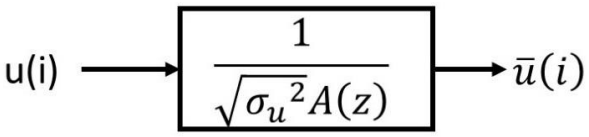

Fig. 22 Prewhitening of $u(i)$ by using the inverse of the spectral factor of $S_u(z)$

Consider a LMS filter with input is $\{\bar{u}(i)\}$ instead of $\{u(i)\}$ as shown in Fig.23. Let the reference sequence d(i) also filter with $\frac{1}{\sigma_u A(z)}$ then the weight updating is given by

$$\bar{w}_i = \bar{w}_{i-1} + \mu\bar{u}_i{}^*\bar{e}(i),\ \ \bar{e}(i) = \bar{d}(i) - \bar{u}_i\bar{w}_{i-1},\ \ \bar{w}_{-1} = 0 \tag{3.1.14}$$

Where $\overline{w}_i$ represents the resulting weight vector, $\{d(i), u_i\}$ satisfies the regression model $d(i) = u_i w^0 + v(i)$, for some unknown vector w^0. The covariance matrix of the transformed regressor given by

$$R_{\overline{u}} = E\overline{u}_i{}^* u_i = I$$

With an Eigen value spread of unity. Hence the convergence performance of the filter improves relative to LMS implementation that depend on the $\{d(i), u(i)\}$.

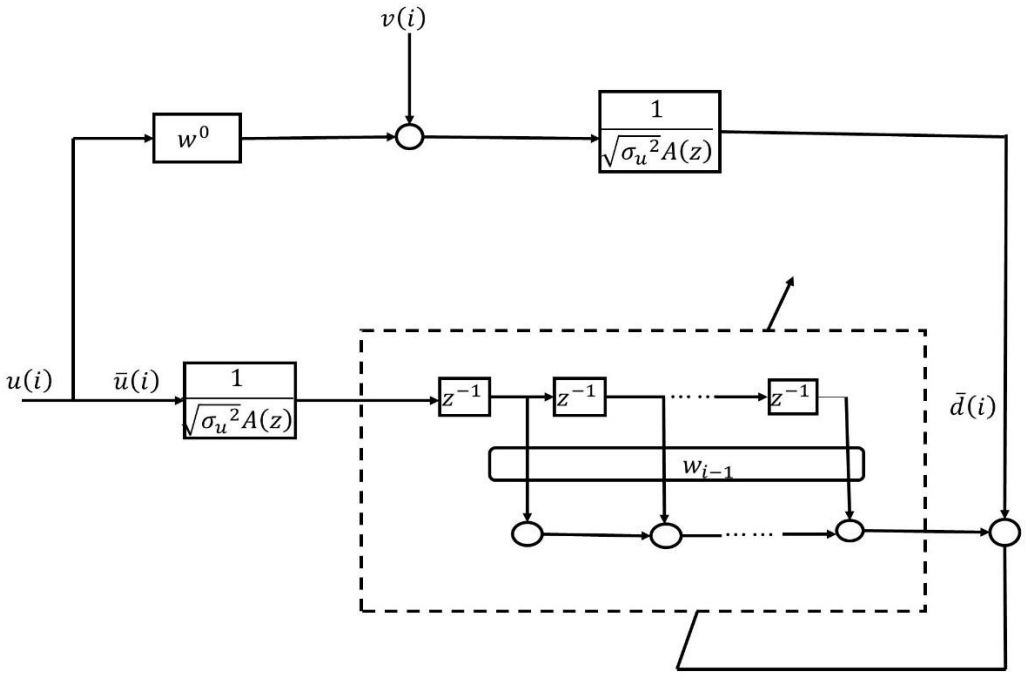

Fig. 23 Adaptive filter implementation with a prewhitening filter

3.2 UNITARY TRANSFORMATION

The statistics of input data is very difficult to know, since the data itself not even stationary, hence it is not possible to design a prewhitening filter $\frac{1}{A(z)}$. There are another way to prewhiten the data , transform the regressor by some pre-selected unitary transformation, such as DCT (Discrete Cosine Transform)or the DFT(Discrete Fourier Transform) [5].

Consider the standard LMS implementation

$$w_i = w_{i-1} + \mu u_i^{*}[d(i) - u_i w_{i-1}] \tag{3.2.1}$$

Let T be an arbitrary unitary matrix of size $M \times M$, $T \times T = I$, for example T could be chosen as DFT or DCT. Once T is selected then the transformed regressor can be written as

$$\bar{u}_i = u_i T \tag{3.2.2}$$

The covariance matrix related to R_u is given by

$$R_{\bar{u}} = E\bar{u}_i^{*}\bar{u}_i = T^{*}R_u T \tag{3.2.3}$$

Let multiplying T^{*} on both side of (3.2.1) we get

$$\bar{w}_i = \bar{w}_{i-1} + \mu \bar{u}^{*}{}_i[d(i) - \bar{u}_i \bar{w}_{i-1}],\ w_{-1} = 0 \tag{3.2.4}$$

The reference sequence $d(i)$ remains same, since $u_i w_{i-1} = \bar{u}_i \bar{w}_{i-1}$. Since T is a unitary therefore $T^{*} = T^{-1}$, the relation between between $\{R_u\ ,\ R_{\bar{u}}\}$ has the similar transformation known as preserve eigen values(two matrices A and B are said to be similar if they related via $B = T^{-1}AT$ for some invertible matrix T. This type similarity matrix preserves eigen values and both have same eigen values).This means that both $\{R_u\ ,\ R_{\bar{u}}\}$ have same eigen values and same eigen value spread. Finally we can say that this type of implementation faces same problem as previous.

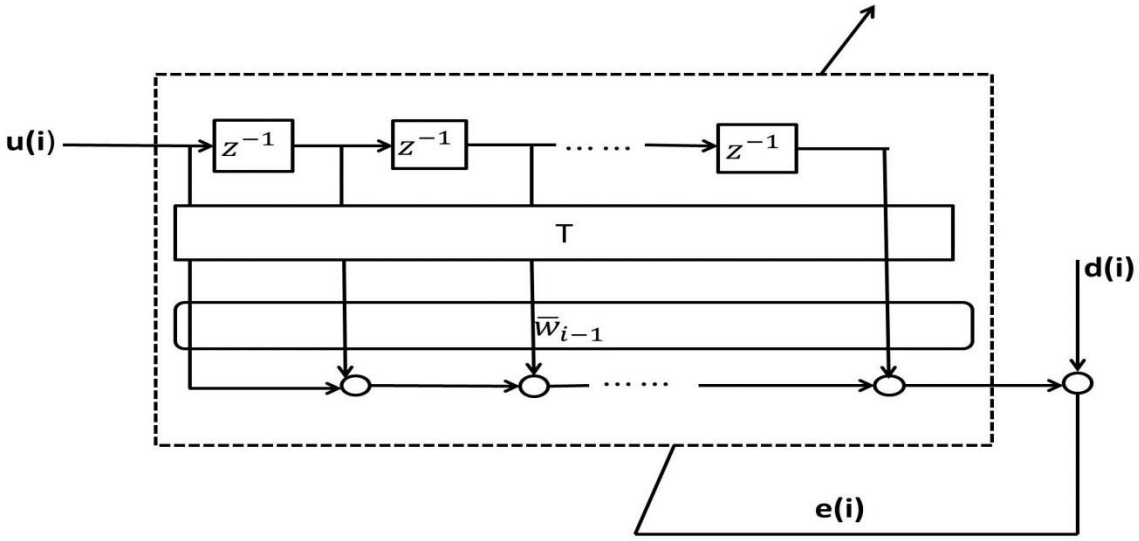

Fig. 24 Transform domain adaptive filter implementation, where T is a unitary transformation

Hence to make better convergence and performance let the (3.2.4) can be written as

$$\bar{w}_i = \bar{w}_{i-1} + \mu D^{-1}\bar{u}^*{}_i[d(i) - \bar{u}_i\bar{w}_{i-1}],\ w_{-1} = 0 \tag{3.2.5}$$

Here D is a diagonal normalization matrix introduced, now the new step size become μD^{-1}. Now let use a diagonal matrix $D^{1/2}$ whose entries are positive square root of entries of D, multiply it both side of (3.2.5) we get

$$w_i' = w_{i-1}' + \mu u_i'^*[d(i) - u_i' w_{i-1}'],\ w_{-1}' = 0 \tag{3.2.6}$$

Where $w_i' = D^{1/2}\bar{w}_i, \quad and\ u_i' = \bar{u}_i D^{-1/2} = u_i T D^{-1/2}$ (3.2.7)

Now the regression covariance matrix given by

$$R_{u'} = E u_i'^* u_i' = D^{-1/2} T^* R_u T D^{-1/2} \tag{3.2.8}$$

Now the relation between$\{R_u\ ,\ R_{u'}\}$, not have the same transformation, hence the Eigen value and Eigen value spread of both are different. By suitable choice of D make $R_{u'}$ become the identity matrix or a multiple of identity.

Let $R_u = U\Lambda U^*$ denote the eigen decomposition of R_u, and choose T and D as

T=U and D=Λ (3.2.9)

Then$R_{\bar{u}} = \Lambda \ and \ R_{u'} = I$, i.e the choice of T and D decorrelate the entries of $\bar{u}_i$ and the variances of the individual entries of $\bar{u}_i$ are the $\{\lambda_k\}$ i.e. the eigen value of R_u. This choice of T is known as KLT (karhunen loeve transform), which is not practical, since it requires the knowledge of R_u. Hence another method is by choosing T as DFT or DCT matrices, which not give exactly a diagonal covariance matrix of $R_{\bar{u}}$, but close to diagonal i.e.

$$R_{\bar{u}} = T^* R_u T \approx diagonal \quad (3.2.10)$$

3.2.1 General Transform Domain LMS Algorithm

Consider a zero mean random variable d with realizations $\{d(0), d(1), \cdots\}$ and a zero mean random row vector u with realizations $\{u_0, u_1, \cdots\}$.the optimal weight vector w^0 , that solves

$$\min_w \ E|d - uw|^2$$

Can be approximated iteratively via $w_i = T\bar{w}_i$, where T is some preselected unitary transformation and $\bar{w}_i$ is updated as follows.

$$\lambda_k(-1) = \varepsilon(a \ small \ positive \ number) \ \ and \ repeat \ for \ i \geq 0 \quad (3.2.11)$$

$$\bar{u}_i = T\bar{w}_i = [\bar{u}_i(0) \ \ \bar{u}_i(1) \quad \cdots \quad \bar{u}_i(M-1) \quad] \quad (3.2.12)$$

$$\bar{u}_i(k) = kth \ entry \ of \ \ \bar{u}_i$$

$$\lambda_k(i) = \beta\lambda_k(i-1) + (1-\beta)|\bar{u}_i(k)|^2, \ \ k = 0,1,\cdots,M-1 \quad (3.2.13)$$

$$D_i = diag\{\lambda_k(i)\} \quad (3.2.14)$$

$$e(i) = d(i) - \bar{u}_i\bar{w}_{i-1} \quad (3.2.15)$$

$$\bar{w}_i = \bar{w}_{i-1} + \mu D^{-1}\bar{u}_i{}^* e(i) \quad (3.2.16)$$

Where μ is a step size (usually small) and $0 \ll \beta < 1$

Practically it is found that the DCT transformation is more successful in transforming R_u to close to diagonal matrix.

3.2.2 DFT Domain LMS Algorithm

Consider a zero mean random variable d with realizations $\{d(0), d(1), \cdots\}$ and a zero mean random row vector u with realizations $\{u_0, u_1, \cdots\}$.the optimal weight vector w^0 that solves

$$\min_w E|d - uw|^2$$

Can be approximated iteratively via$w_i = T\overline{w}_i$, where T is the unitary DFT matrix given by

$$[F_{mk}] = \frac{1}{\sqrt{M}} e^{-\frac{j2\pi mk}{M}}, m, k = 0,1, \cdots, M-1 \quad (3.2.17)$$

$\overline{w}_i$ is updated as follows. The $M \times M$ diagonal matrix given by

$$S = diag\left\{1, e^{\frac{-j2\pi}{M}}, \cdots, e^{\frac{-j2\pi(M-1)}{M}}\right\} \quad (3.2.18)$$

Start with $\lambda_k(-1) = \epsilon(a\ small\ positive\ number), w_{-1} = 0, \bar{u}_{-1} = 0, and\ repet\ for\ i \geq 0$

$$\bar{u}_i = \bar{u}_{i-1} S + \frac{1}{\sqrt{M}}\{u(i) - u(i-M)\}[1\ \ 1\ \ \cdots\ \ 1] \quad (3.2.19)$$

$\bar{u}_i(k) = kth\ entry\ of\ \bar{u}_i$

$$\lambda_k(i) = \beta\lambda_k(i-1) + (1-\beta)|\bar{u}_i(k)|^2, \ \ k = 0,1, \cdots, M-1 \quad (3.2.20)$$

$$D_i = diag\{\lambda_k(i)\} \quad (3.2.21)$$

$$e(i) = d(i) - \bar{u}_i \overline{w}_{i-1} \quad (3.2.22)$$

$$\overline{w}_i = \overline{w}_{i-1} + \mu D^{-1} \bar{u}_i^* e(i) \quad (3.2.23)$$

Where μ is a step size (usually small) and $0 \ll \beta < 1$

3.2.3 DCT LMS Algorithm

Consider a zero mean random variable d with realizations $\{d(0), d(1), \cdots\}$ and a zero mean random row vector u with realizations $\{u_0, u_1, \cdots\}$.the optimal weight vector w^0 that solves

$$\min_w E|d - uw|^2$$

Can be approximated iteratively via$w_i = T\bar{w}_i$, where $T = C^T$ and C is the DCT matrix

$$[C]_{km} = \alpha(k)\cos\left(\frac{k(2m+1)\pi}{2M}\right),\ \ k,m = 0,1,\cdots,M-1 \tag{3.2.24}$$

$$\alpha(0) = {}^{1}/_{\sqrt{M}}, \alpha(k) = \sqrt{\frac{2}{M}},\quad for\ k \neq 0 \tag{3.2.25}$$

$\bar{w}_i$ is updated as follows. The $M \times M$ diagonal matrix given by

$$S = diag\{2\cos(k\pi/M)\},\ k = 0,1,\cdots,M-1 \tag{3.2.26}$$

Start with $\lambda_k(-1) = \epsilon(a\ small\ positive\ number), w_{-1} = 0, \bar{u}_{-1} = 0, and\ repet\ for\ i \geq 0$

$$a(k) = [u(i) - u(i-1)]\cos\left(\frac{k\pi}{2M}\right),\ k = 0,1,\cdots,M-1 \tag{3.2.27}$$

$$b(k) = (-1)^k[u(i-M) - u(i-M-1)]\cos\left(\frac{k\pi}{2M}\right),\ k = 0,1,\cdots,M-1 \tag{3.2.28}$$

$$\phi(k) = \alpha(k)[a(k) - b(k)], k = 0,1,\cdots,M-1 \tag{3.2.29}$$

$$\bar{u}_i = \bar{u}_{i-1}S - \bar{u}_{i-2} + [\phi(0)\quad \phi(1)\quad \cdots\quad \phi(M-1)\ ] \tag{3.2.30}$$

$\bar{u}_i(k) = kth\ entry\ of\ \bar{u}_i$

$$\lambda_k(i) = \beta\lambda_k(i-1) + (1-\beta)|\bar{u}_i(k)|^2,\ \ k = 0,1,\cdots,M-1 \tag{3.2.31}$$

$$D_i = diag\{\lambda_k(i)\} \tag{3.2.32}$$

$$e(i) = d(i) - \bar{u}_i\bar{w}_{i-1} \tag{3.2.33}$$

$$\bar{w}_i = \bar{w}_{i-1} + \mu D^{-1}\bar{u}_i{}^* e(i) \tag{3.2.34}$$

By performing a simulation to compare with the convergence and performance of LMS, DCT-LMS, DFT-LMS and Prewhitening method, it is found that the prewhitening filter gives better result than that of rest as shown in Fig.25

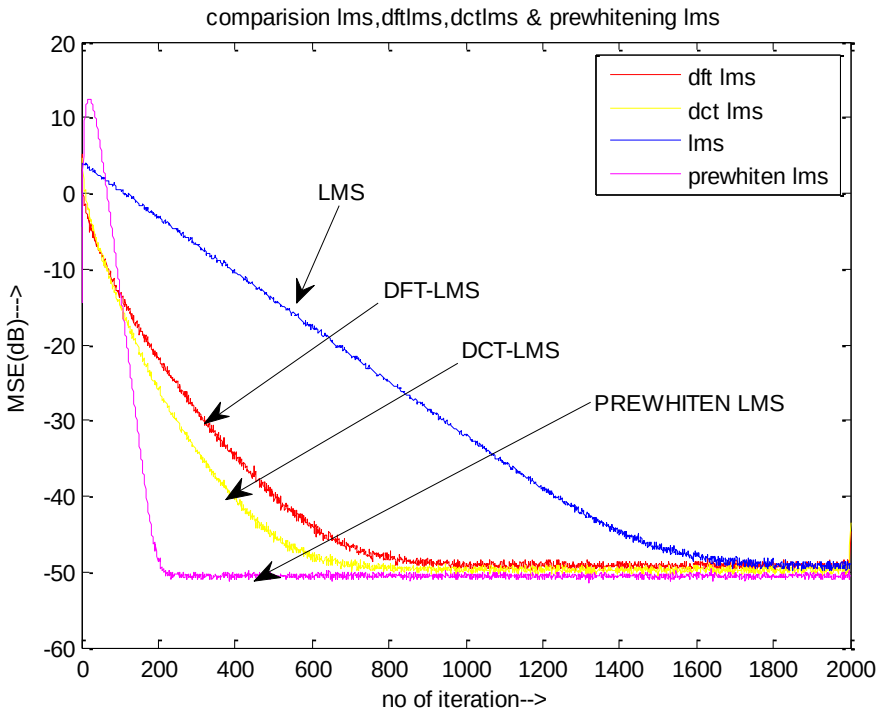

Fig. 25 comparison between LMS, DFT-LMS, DCT-LMS and DCT-LMS

Now we apply DCT-LMS and DFT-LMS algorithm in incremental method, then we found that the DCT-LMS not only gives better convergence but also gives better performance than that of rest. In in frequency domain incremental, the process is same as the incremental method, only the difference is instead taking all the parameter time domain here we will take in frequency domain and introduce a term D in the weight updation equation. That is here first the data transform to frequency domain then apply to incremental strategies using the respective algorithm. The block diagram of frequency domain incremental strategy shown in Fig.26.

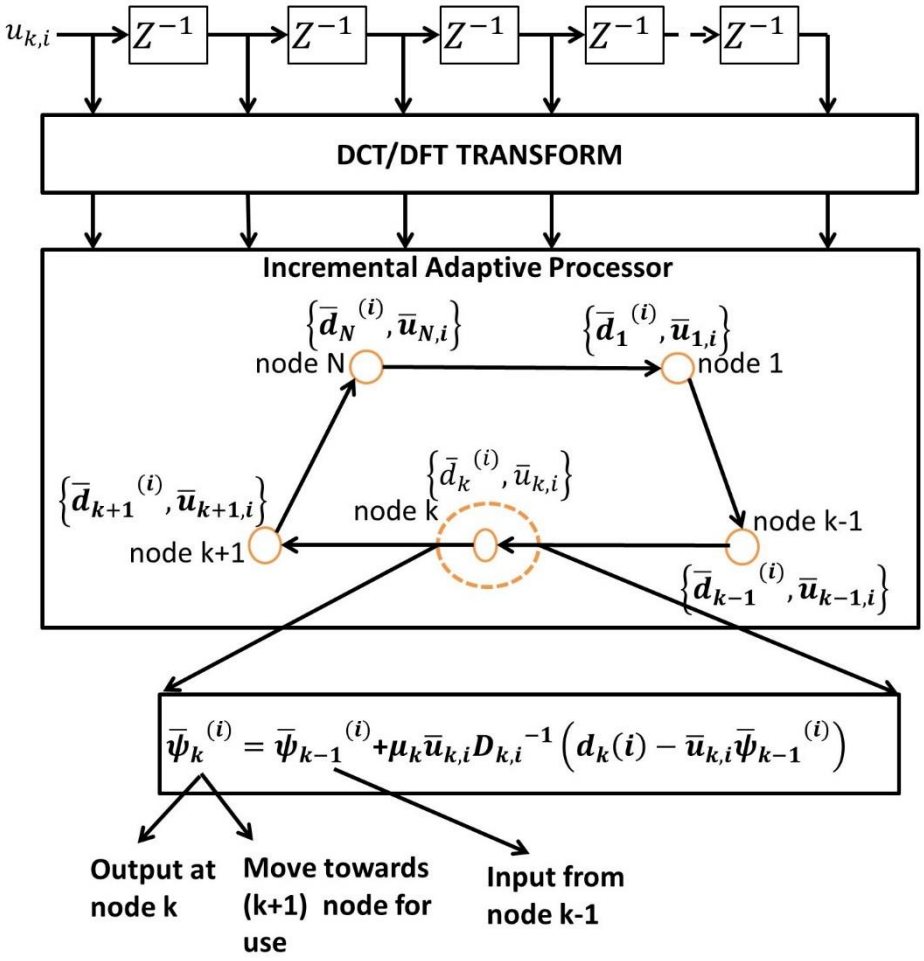

Fig. 26 Block diagram of frequency domain incremental LMS algorithm

Now we will perform a simulation to compare the convergence rate and performance of all the algorithm i.e. adaptive incremental method, incremental steepest descent solution, and DCT-LMS and DFT-LMS algorithm using incremental method. It is found that the convergence rate and performance of DCT-LMS algorithm using incremental method gives better result than that of rest algorithm. The MSE (mean square error), EMSE (excess mean square error) and MSD (mean square deviation) comparison of all the algorithm as shown in Fig.27, Fig.28 and Fig.29. The MSE and EMSE of all the algorithm with respect to node shown in Fig.30 and Fig.31

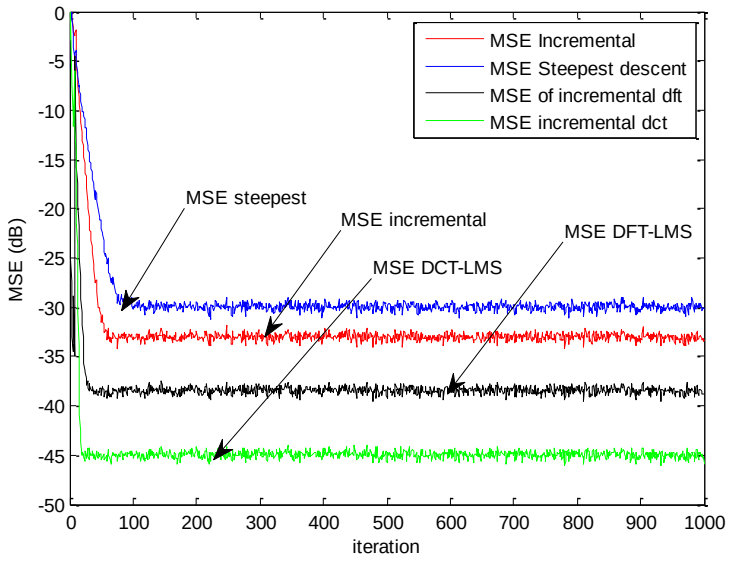

Fig. 27 Transient MSE performance at node 1 for incremental adaptive solution, stochastic steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS

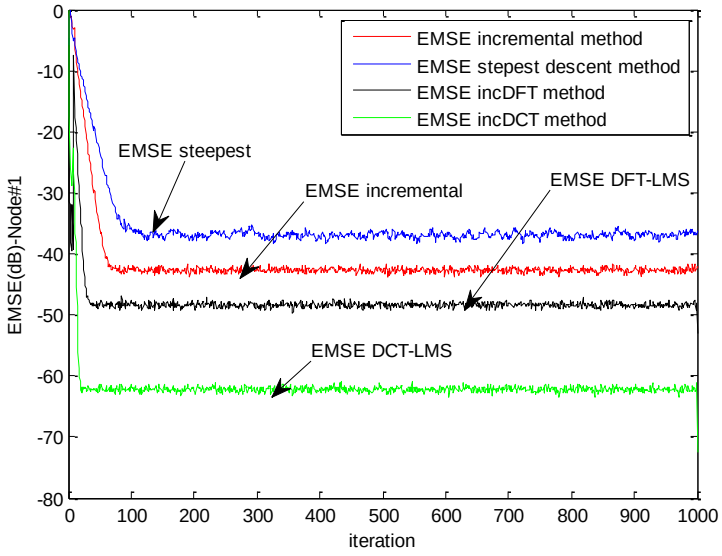

Fig. 28 EMSE performance at node 1 for incremental adaptive solution, incremental steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS

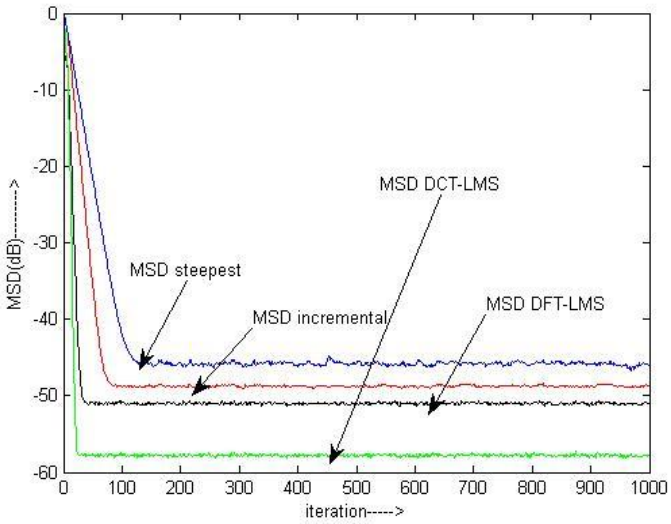

Fig. 29 MSD performance at node 1 for incremental adaptive solution, incremental steepest descent solution. Incremental DCT-LMS and incremental DFT-LMS

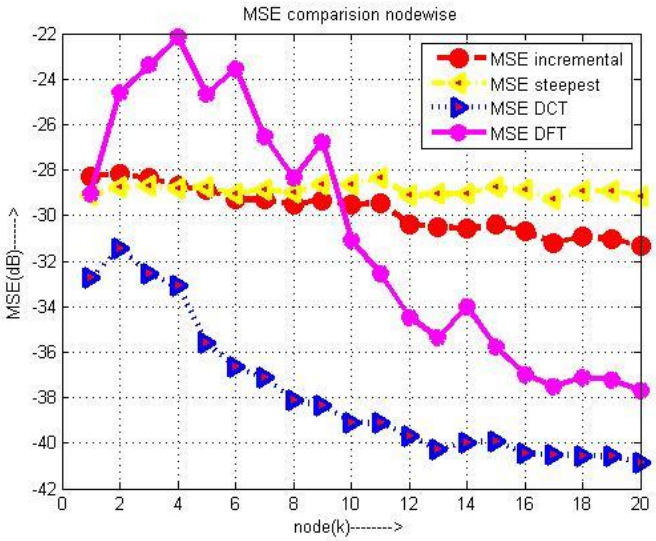

Fig. 30 MSE performance with respect to Node for all algorithm

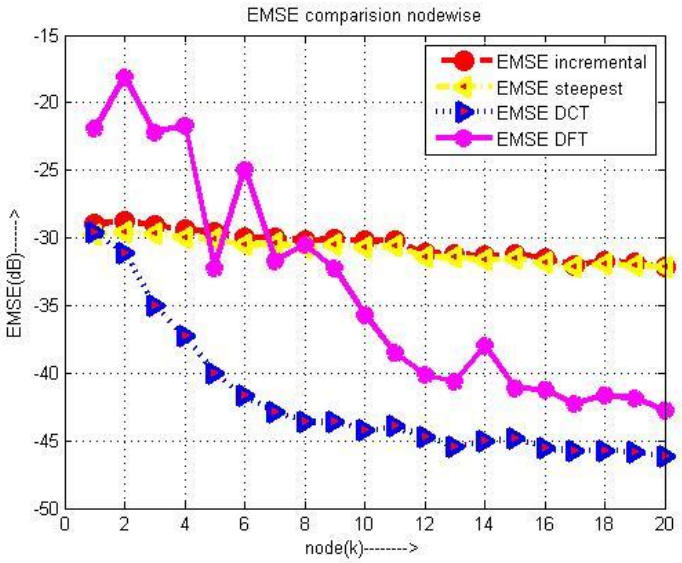

Fig. 31 EMSE with respect to node for all the algorithm

3.3 RLS ALGORITHM

The RLS stands for recursive least square, RLS algorithm gives a more sophisticated way to approximate the input covariance matrixR_u [5]. Here we are not going through details of RLS algorithm, just proceed it to reach the algorithm using stochastic gradient method. Like LMS algorithm to find RLS algorithm we just start with the Newton's recursion method [5]given by

$$w_i = w_{i-1} + \mu(i)[\epsilon(i)I + R_u]^{-1}[R_{du} - R_u w_{i-1}] \tag{3.3.1}$$

Let replace R_u by some exponentially weighted average sample, given by;

$$\hat{R}_u = \frac{1}{i+1}\sum_{j=0}^{i} \lambda^{i-j} u_j{}^* u_j \tag{3.3.2}$$

Where λ lies between, $0 \ll \lambda \leq 1$. Let assume that the value of $\lambda = 1$, then the above expression becomes

$$\hat{R}_u = \frac{1}{i+1}\sum_{j=0}^{i} u_j{}^* u_j \tag{3.3.3}$$

Choosing the value of λ is very important, by choose it less than one it requires more memory into the estimation of $\hat{R}_u$. Because by choosing less than one λ would assign more weight to recent regressor and less weight to regressor in the remote past. By this the filter track the data in the

remote past and give more relevance to recent data so that changes in R_u can be better tracked. Now let choose step size $\mu(i)$ as

$$\mu(i) = {}^{1}/_{(i+1)} \tag{3.3.4}$$

And choosing regularization factor as

$$\varepsilon(i) = {}^{\lambda^{i+1}\epsilon}/_{(i+1)}, i \geq 0 \tag{3.3.5}$$

By using the above data now the Newton's recursion (2.6.140) become

$$w_i = w_{i-1} + \left[\lambda^{i+1}\varepsilon I + \sum_{j=0}^{i} \lambda^{i-j} {u_j}^* u_j\right]^{-1} {u_i}^* [d(i) - u_i w_{i-1}] \tag{3.3.6}$$

Now let we form a matrix ϕ_i that forms by combining the present and past data, then it can be write as

$$\phi_i = \left(\lambda^{i+1}\varepsilon I + \sum_{j=0}^{i} \lambda^{i-j} {u_j}^* u_j\right) \tag{3.3.7}$$

Now from the above definition ofϕ_i, we can write the ϕ_i in the form given by:

$$\phi_i = \lambda\phi_{i-1} + {u_i}^* u_i, \ \phi_{-1} = \varepsilon I \tag{3.3.8}$$

Let $P_i = {\phi_i}^{-1}$, then applying the matrix inversion formula [5] we get

$$P_i = \lambda^{-1}\left[P_{i-1} - \frac{\lambda^{-1}P_{i-1}{u_i}^* u_i P_{i-1}}{1+\lambda^{-1}u_i P_{i-1}{u_i}^*}\right] \tag{3.3.9}$$

From the above recursion it is clear that for update form P_{i-1} to P_i the knowledge of recent regressor u_i is required. The RLS algorithm can be summarize in this way given by:

RLS algorithm: consider a zero mean random variable d with realizations$\{d(0), d(1), \cdots\}$, and a zero mean random row vector u with realizations$\{u_0, u_1, \cdots\}$. The optimal weight vector w^0 that solves

$$\min_w E|d - uw|^2$$

Can be approximated iteratively via the recursion

$$P_i = \lambda^{-1}\left[P_{i-1} - \frac{\lambda^{-1}P_{i-1}{u_i}^* u_i P_{i-1}}{1+\lambda^{-1}u_i P_{i-1}{u_i}^*}\right]$$

$$w_i = w_{i-1} + P_i {u_i}^* [d(i) - u_i w_{i-1}], \qquad i \geq 0$$

With initial condition $P_{-1} = \varepsilon^{-1} I \ and \ 0 \ll \lambda \leq 1$

Similarly another algorithm is there known as GN (Gauss Newton) algorithm. Here this algorithm not explained fully only gives the algorithm, so that we can compare it with LMS and RLS algorithm.

GN algorithm: consider a zero mean random variable d with realizations$\{d(0), d(1), \cdots\}$, and a zero mean random row vector u with realizations$\{u_0, u_1, \cdots\}$. The optimal weight vector w^0 that solves

$$\min_{w} E|d - uw|^2$$

Can be approximated iteratively via the recursion

$$P_i = \frac{\lambda^{-1}}{1-\alpha}\left[P_{i-1} - \frac{\lambda^{-1}P_{i-1}u_i{}^*u_iP_{i-1}}{\frac{1-\alpha}{\alpha} + \lambda^{-1}u_iP_{i-1}u_i{}^*}\right]$$

$$w_i = w_{i-1} + P_iu_i{}^*[d(i) - u_iw_{i-1}], \qquad i \geq 0$$

With initial condition $P_{-1} = \varepsilon^{-1}I \ \ and\ 0 \ll \lambda \leq 1$ and $0 < \alpha \leq 0.1$

Now we can perform a simulation to compare the RLS algorithm with LMS and GN algorithm. The Fig.32 provides the MSE comparison of RLS algorithm with the LMS algorithm, the Fig clearly discloses that the RLS algorithm has better convergence rate and performance than that of LMS algorithm.Fig.33 represents the MSE comparison of GN algorithm with the LMS algorithm. The simulation result clears the fact that the GN algorithm has better convergence rate than that of the LMS algorithm.Fig.34 represents the MSE performance comparison of RLS, GN and LMS algorithm. The result represent that the performance of RLS algorithm is better than that of GN and LMS algorithm.Fig.35 represents the MSE comparison of RLS,LMS and GN algorithm using incremental mode of cooperation and Fig.36 and Fig.37 represents the EMSE and MSD comparison of all the three algorithm, form all the simulation result we get same conclusion that the performance of RLS algorithm is better than that of LMS and GN algorithm.

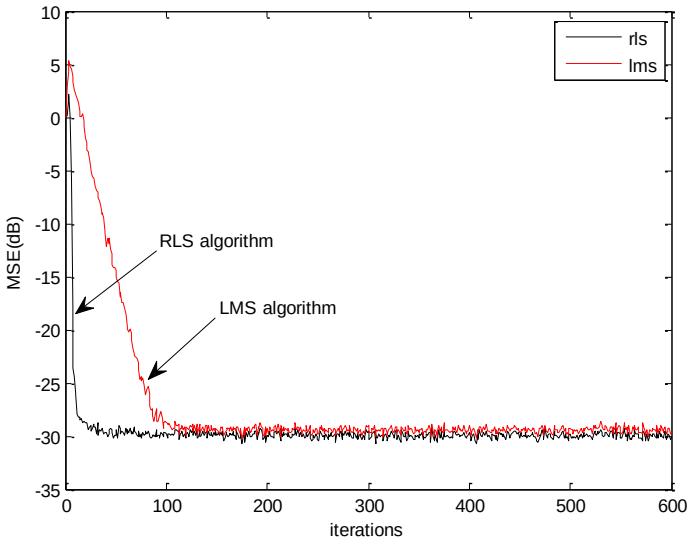

Fig. 32 MSE comparison between RLS and LMS algorithm

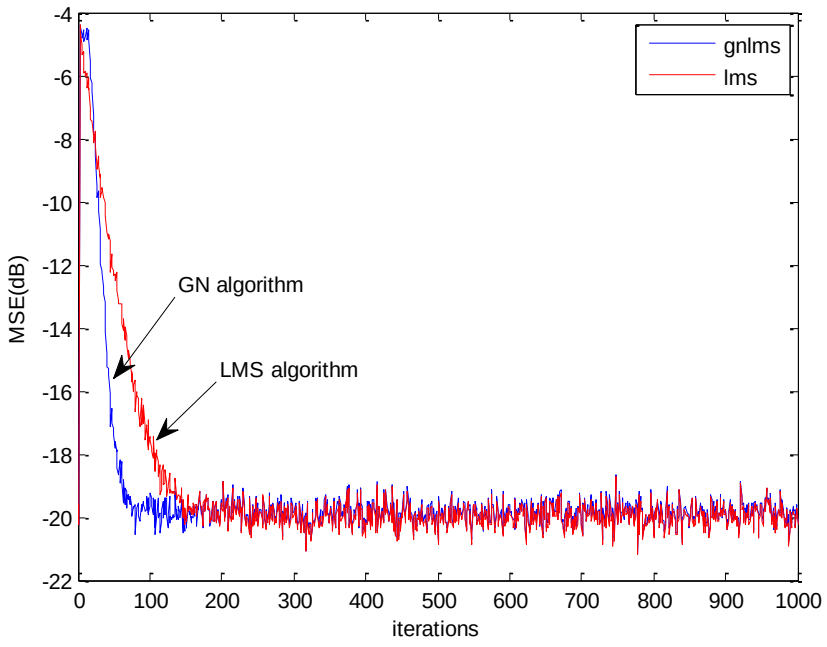

Fig. 33 MSE comparison between GN and LMS algorithm

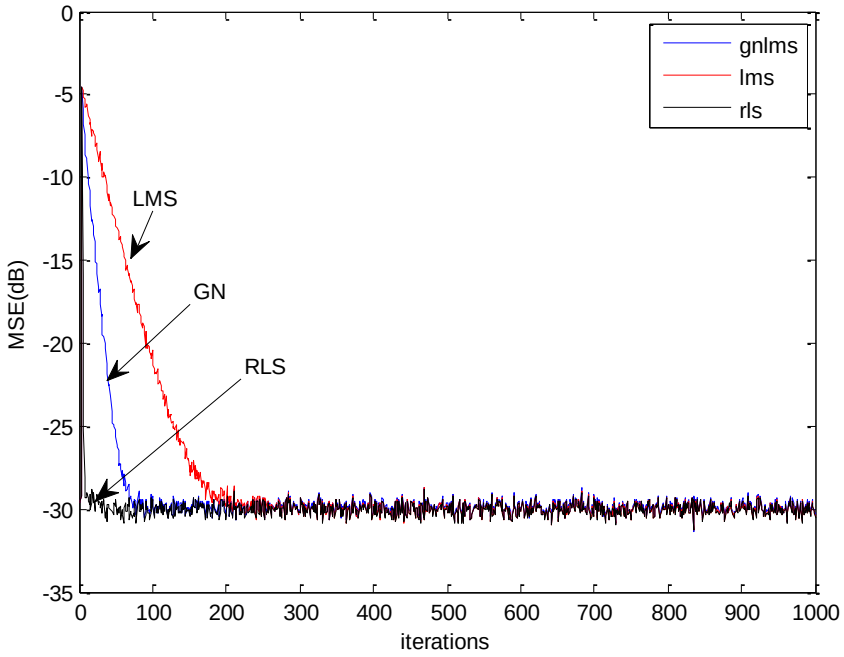

Fig. 34 MSE comparison of RLS, LMS and GN algorithm

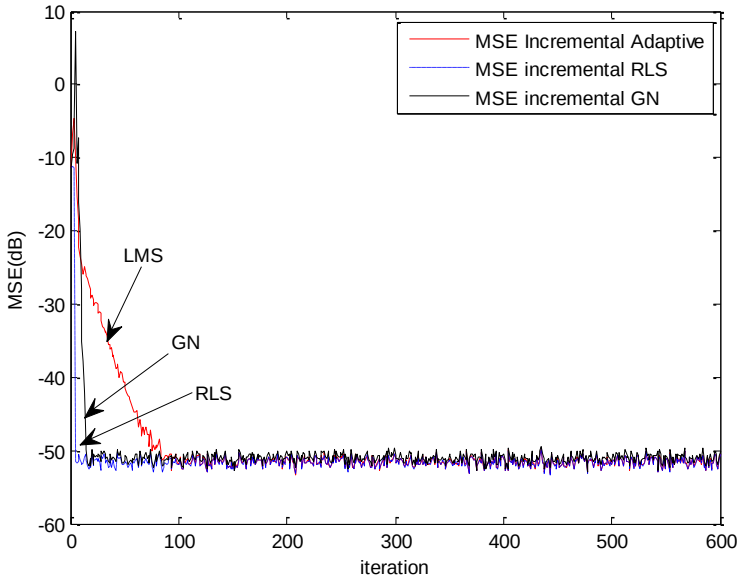

Fig. 35 MSE performance comparison of RLS, LMS and GN algorithm using incremental strategies

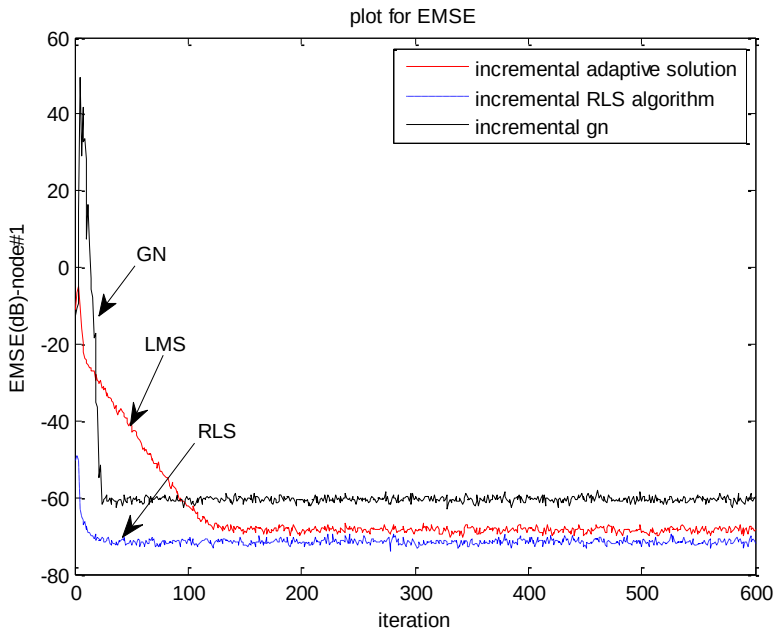

Fig. 36 EMSE performance comparison of RLS, LMS and GN algorithm using incremental strategies

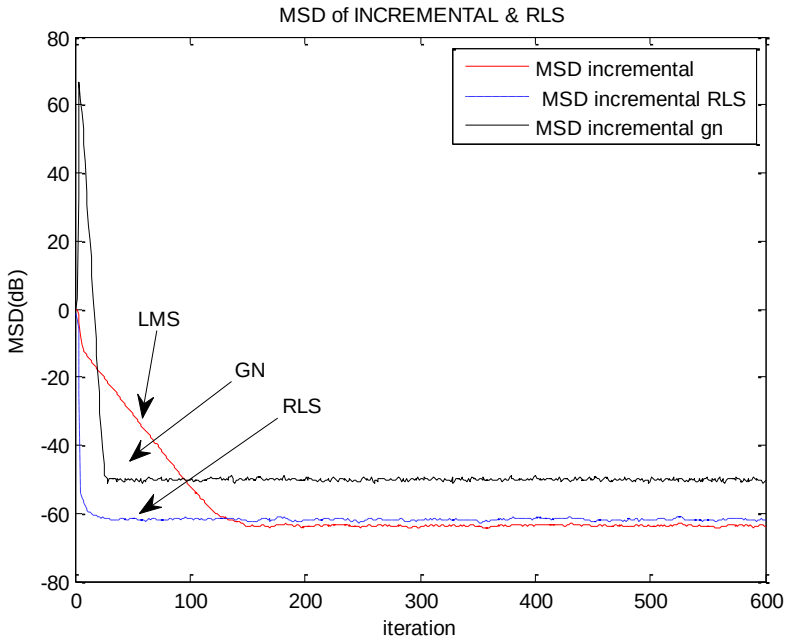

Fig. 37 MSD performance comparison of RLS, LMS and GN algorithm using incremental strategies

3.4 Conclusion

Transform domain algorithm like DCT-LMS and DFT-LMS algorithm provides a platform to achieve faster convergence and better steady state performance without practically implementation of prewhiten filter. Most practical purpose it is found that the DCT-LMS algorithm gives better result than that of DFT-LMS algorithm. Since it provide more accurate autocorrelation matrix of the input regressor of the required form mention earlier than that of other transform domain algorithm. Similarly the RLS algorithm and GN algorithm also provide better performance both in steady state and convergence than that of LMS algorithm. But the complexity of this type of algorithm is more than that of LMS algorithm.

Chapter 4 CONVERGENCE ANALYSIS OF VARRIABLE XE-NLMF ALGORITHM

The LMS (least mean square) algorithm has variety of practical application, since it is very simple to implement. It is found that for some application the adaptive algorithm with higher order error signal Performs better result than that of LMS algorithm, one of the such algorithm is LMF(least mean forth) algorithm. It is found that this algorithm (LMF algorithm) performs better than that of LMS algorithm in the case of non-Gaussian additive noise. The performance of LMF algorithm also improves in Gaussian case by adding a square wave noise to the existing noise [6].

Let we can perform one experiment to study the performance of LMF and LMS algorithm under different noise condition. First we will perform a simple simulation to study the performance of LMS, NLMS and LMF algorithm.

4.1 LMF Algorithm

Consider a zero mean random variable d with realizations $\{d(0), d(1), \cdots\}$ and a zero-mean random row vector u with realizations$\{u_0, u_1, \cdots\}$. The optimal weight vector that solves [5]

$$\min_{w} E|d - uw|^4 \tag{4.1.1}$$

Can be approximated iteratively via the recursion

$$w_i = w_{i-1} + \mu u_i^* e(i)|e(i)|^2, \quad i \geq 0 \tag{4.1.2}$$

Where μ is a positive step size (usually small)

A simulation perform to compare the performance of LMS, NLMS and LMF algorithm. Taking the step size for LMS, NLMS and LMF is 0.007, 0.02 and 0.06, the result shown in Fig.38.Now

another simulation perform to compare the same only the change is let taking step size of LMF is 0.02, then the output as shown inn Fig.39

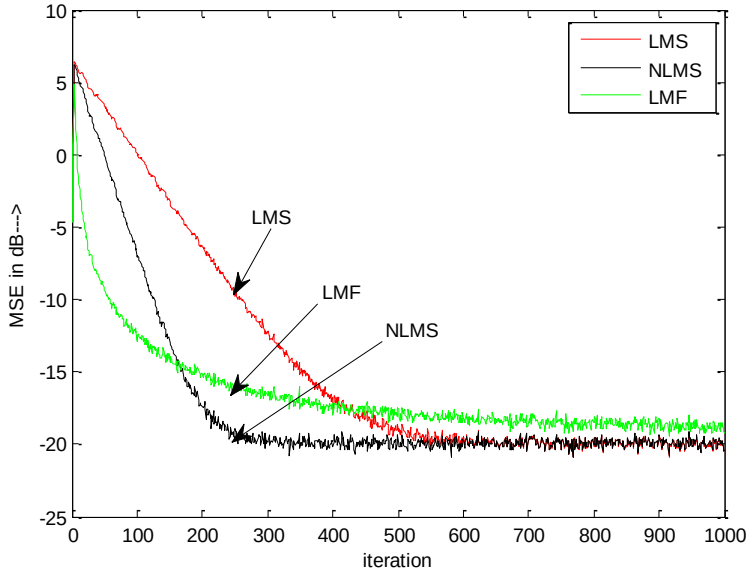

Fig. 38 Performance comparison of LMS, NLMS and LMF

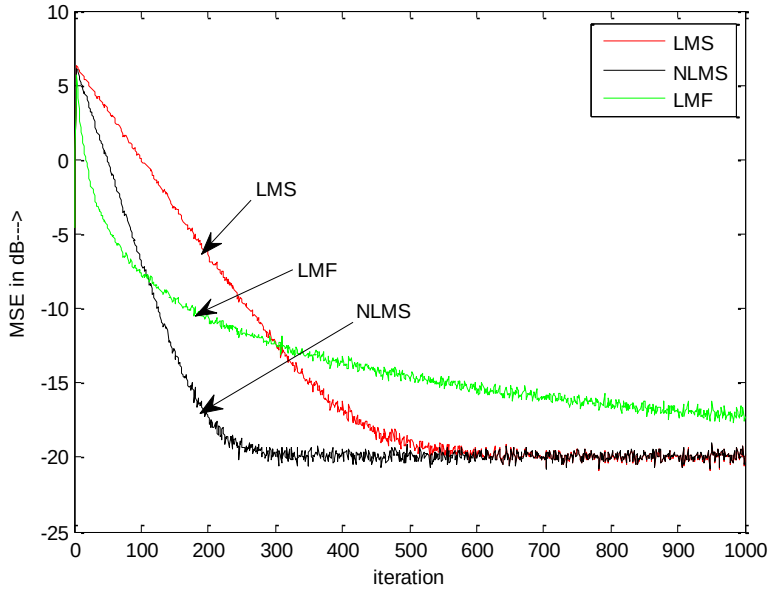

Fig. 39 Performance comparison of LMS, NLMS and LMF algorithm

The simulation results show that the performance NLMS is better than that of LMS and LMF algorithm under Gaussian noise case. Now let we perform another simulation by taking nongaussian noise and also Gaussian to compare the performance of LMS, NLMS and LMF algorithm. Let the input is an AR (auto regressor model), input $x(n)$ obtain from

$$x(n) = 0.4x(n-1) + y(n) \tag{4.1.3}$$

Where $y(n)$ is a Gaussian white process with unit variance. Let the system simulate for Gaussian, uniform and sinusoidal noise distribution, for all the case take$SNR = 20.75dB$. The step size for all case is given by $\mu_{LMS} = 2.571 \times 10^{-5}$, $\mu_{LMFuni} = 2 \times 10^{-3}$, $\mu_{LMFgau} = 5.13 \times 10^{-4}$ $and\ \mu_{LMFsin} = 3.05 \times 10^{-3}$. The LMS MSE behavior same for all the noise distribution, since it depends only the noise variance. The LMF performance for different noise as shown in Fig.40. It clears that from simulation result the LMF algorithm performs better result than that of LMS algorithm. Same comparison can also do by performing a simulation using the incremental method cooperation as mentioned in (2.5.16).the simulation result shown in Fig.41. It also clearly reveals that the LMF algorithm performs better than that of LMS algorithm not only in Gaussian noise case but also nongaussian noise case.

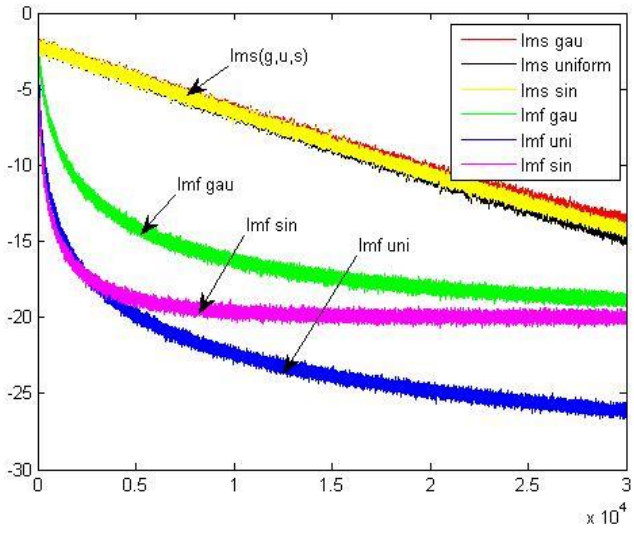

Fig. 40 performance comparison between LMF and LMS for different noise condition

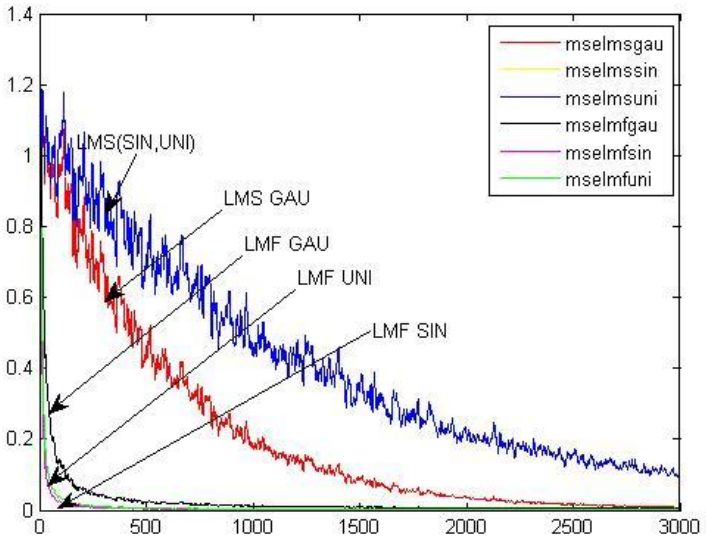

Fig. 41 comparison of LMF and LMS algorithm for different noise condition using incremental method

4.2 OPTIMIZED NORMALIZED ALGORITHM FOR SUBGAUSSIAN NOISE

The LMF algorithm comes under the class of stochastic gradient based algorithm. The power of LMF lies in its faster initial convergence and lower steady state error relative to the LMS algorithm [18]. From previous result it is clear that it performs better in the case of sub Gaussian noise. Generally higher order algorithm requires a very small step size for stability, but in case of LMF algorithm it has of order three, hence it destroy the initial stability condition, the solution for this is to normalize the step size [19].

Here we can use two type normalized technique, in one case the step size normalized by the signal power known as XE-NLMF algorithm, in other case the step size normalized by combination of signal power and noise power. The XE-NLMF algorithm [20] represented by

$$w(n+1) = w(n) + \frac{\gamma_{xe} e^3(n) x(n)}{\delta + (1-\lambda)|x(n)|^2 + \lambda \|e(n)\|^2} \tag{4.2.1}$$

The variable XE-NLMF algorithm [18] represented by

$$w(n+1) = w(n) + \frac{\gamma_{xe} e^3(n) x(n)}{\delta + (1-\lambda(n))|x(n)|^2 + \lambda(n)|e(n)|^2} \tag{4.2.2}$$

Where γ_{xe} represents the step size. $w(n)$ Represents the filter coefficient vector of the adaptive filter. $x(n)$ is the input vector and $e(n)$ is the error vector. In XE-NLMF algorithm the LMF is normalized by signal power and error power balanced by the power parameter λ.it has advantage that the signal power normalize the signal, while the error power reduces the outlier estimation error.in general both (4.2.1) and (4.2.2) can be represent by

$$w(n+1) = w(n) + \gamma_{xe} f(e(n))x(n) \qquad (4.2.3)$$

Where $f(e(n))$ represent the scalar function of the output estimation error.

Some example of $f(e(n))$ given in the table

Algorithm	$f(e(n))$
LMS	e(n)
LMF	$e(n)$
NLMS	$e^3(n)/\|x(n)\|^2$
NLMF	$e^3(n)/\|x(n)\|^2$
XE-NLMF	$e^3(n)/\delta + (1-\lambda)\|x(n)\|^2 + \lambda\|x(n)\|^2$
Sign-LMS	sign$[e(n)]$

Table 1 example of $f(e(n))$

This algorithm has great application in the dynamic channel, where the time variations of mixing parameter allow the algorithm changes in the channel opposed to the same algorithm with fixed mixing parameter.

A simulation result can be perform to simple study the performance of fixed XE-NLMF algorithm, LMF algorithm, LMS and NLMS algorithm. Let the step size for LMS algorithm$\mu_{lms} = 0.007$, the step size for other algorithms are$\mu_{NLMS} = 0.02$, $\mu_{LMF} = 0.06$, $\mu_{XE-NLMF} = 0.5$.The convergence performance of LMF, LMS, NLMS and XE-NLMF are shown in Fig.42. The result clears the fact that the convergence of fixed XE-NLMF better than that of rest. Here the value of λ is set to 0.9.

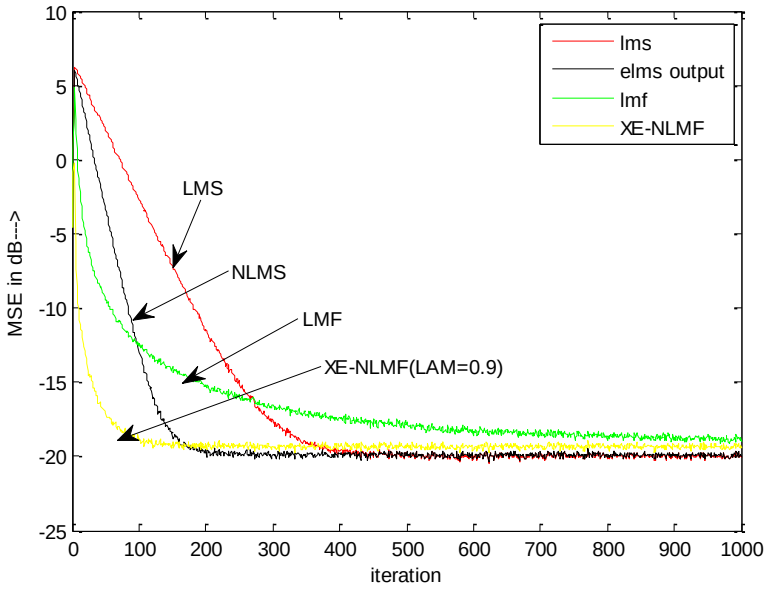

Fig. 42 convergence performance of XENLMF, LMF, LMS and NLMS for $\lambda = 0.9$

4.3 VARIABLE NORMALIZED XE-NLMF ALGORITHM

The mixing parameter λ lies in the interval $[0,1]$ and weighted recursively to adjust the signal power and noise power. The error is defined as;

$$e(n) = d(n) - w^T(n)x(n) \tag{4.3.1}$$

The error feedback quantity $\mu(n)$ updated according to the variable step size parameter

$$\mu(n+1) = v\mu(n) + p(n)|e(n)e(n-1)| \tag{4.3.2}$$

Where the quantity $e(n)e(n-1)$ determines the distance of $w(n)$to the optimum weights, $|\cdot|$ denotes the absolute value operation, $p(n)$ updated according to sum of past three samples of $\lambda(n)$ according to following way:

$$p(n) = [\lambda(n-2) + \lambda(n-1) + \lambda(n)]a \tag{4.3.3}$$

$a\ and\ v\ are\ constants$.

Here it is important to choose$\lambda(n)$, since it controls the signal power and error power.

$$\lambda(n) = erf\{\mu(n)\} \tag{4.3.4}$$

Where $erf\{\cdot\}$ is refers to error function, the main aim to take this is to restrict $\mu(n)$in the interval[0,1]. The parameter v and a also restricted to interval[0,1]. To avoid zero feedback we have to take the initial value of p is 0.5.

This will provide automatic adjustment for $\lambda(n)$ based on the estimation error. If the error is large than λ approaches to unity and provide faster adaption. While when error function is small, λ is adjusted to smaller value for lower steady state error. Based on this motivation the variable XE-NLMF [18] algorithm can be express as

$$w(n+1) = w(n) + \frac{\gamma_{xe} e^3(n)x(n)}{\delta + (1-\lambda(n))|x(n)|^2 + \lambda(n)|e(n)|^2} \tag{4.3.5}$$

4.3.1 Convergence Analysis

The mean convergence analysis can be done by taking the mean of weight error deviation,$v(n) = w(n) - w^*$. Considering the LMF convergence analysis in [21] and [20]. The difference equation for the weight error defined by:

$$E\{v(n+1)\} = [I - 3\gamma E\{\gamma^2(n)\}R]E\{v(n)\} \tag{4.3.6}$$

Mean convergence of XE-NLMF can be verified by replacing γ with normalized step size as follows:

$$E\{v(n+1)\} = \left[I - \frac{3\gamma_{xe} E(\eta^2(n))R}{(1-\lambda(n))\sigma_x{}^2 + \lambda(n)\sigma_e{}^2}\right] E\{v(n)\} \tag{4.3.7}$$

Where$\|x(n)\|^2 = \sigma_x{}^2 \ and \ \|e(n)\|^2 = \sigma_e{}^2 . for \ tr[R] = N\sigma_x{}^2$, now we can write a general condition for equation (4.3.7) by:

$$\gamma_{xe} < \frac{1}{3N}\left(\frac{(1-\lambda(n))}{\sigma_\eta{}^2} + \frac{\lambda(n)\sigma_e{}^2}{\sigma_\eta{}^2\sigma_x{}^2}\right) \tag{4.3.8}$$

Here we observe two conditions for stability first for $\lambda(n) = 1$ and second for$\lambda(n) = 0$. Now the effect of λ on XE-NLMF can be shown in Fig.43.We know that the error is large during initial condition and then gradually decreases towards the minimum. Hence the signal power$\|x(n)\|^2$, act as a threshold to reduce the step size when the error convergence to minimum. The combination $(1-\lambda(n))|x(n)|^2 + \lambda(n)|e(n)|^2$ has advantage that the normalizing the signal power improves

stability and $|e(n)|^2$ will decreases the outlier distribution of $e^3(n)$ in the recursive updating equation of XE-NLMF algorithm [18].

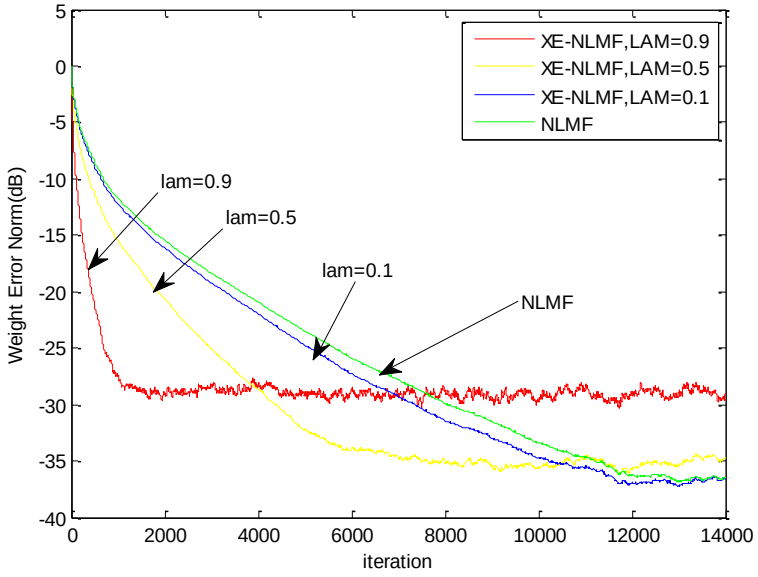

Fig. 43 Effect of lambda in the fixed XE-NLMF

4.4 Simulation Results

Let an unknown system is modelled by N=10 time invariant filter with weights given by:

$$w^* = [0.035 - 0.068\ 0.12 - 0.258\ 0.9 - 0.25\ 0.10 - 0.07\ 0.067 - 0.067]^T$$

The input signal $x(n)$ is obtained by passing the white Gaussian noise $u(n)$ through an AR (auto regressor) model,$x(n) = x(n-1) + 0.6u(n)$. The signal to noise ratio for whole experiment set to 20dB. Two types of noise used to perform the result one the Gaussian noise and second one is the binary additive noise (or nongaussian noise).here we take the average of 300 experiments to plot the weight error norm, $10log10(\|w^* - w(n)\|^2/\|w^*\|^2)$.The step size for XE-NLMF and variable XE-NLMF algorithm set to 0.1 and the step size for the NLMS algorithm set to be 0.2. Other parameters are v=0.98 and a=0.9.the convergence performance of variable XE-NLMF algorithm, NLMF and NLMS algorithm as shown in Fig.44. The simulation result clears that the variable XE-NLMF algorithm adapts faster than that of the XE-NLMF algorithm and NLMF algorithm. The variable XE-NLMF algorithm not only adapts faster but also producing lower steady state weight error norm of more than 15dB. Hence this the advantage of using variable

mixed power parameter for further improvement of the XE-NLMF algorithm.in the second case we will take binary additive noise instead of additive Gaussian noise. The Fig.45 represents the convergence performance of NLMS algorithm, XE-NLMF algorithm and variable XE-NLMF algorithm under binary additive noise case. Fig.46 represents the MSE performance of XE-NLMF,variable XE-NLMF and NLMS algorithm using incremental adaptive solution algorithm.The simulation results show that for binary additive noise case the variable normalized XE-NLMF algorithm fast convergence and with lower steady state error. It is found that the weight error norm improves about 25dB over NLMS algorithm.

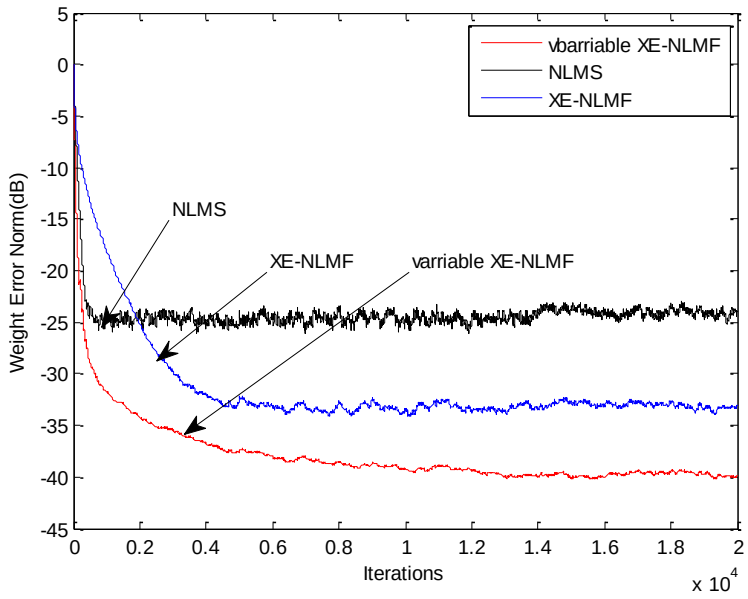

Fig. 44 convergence performance for the variable XE-NLMF algorithm, the XE-NLMF algorithm (λ=0.9) and the NLMS algorithm in White Gaussian noise using incremental adaptive algorithm

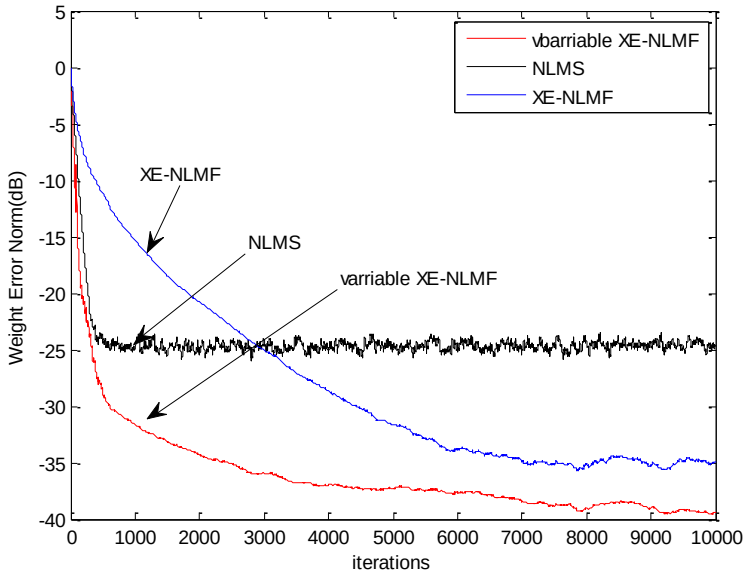

Fig. 45 convergence performance of NLMS, XE-NLMF and variable XE-NLMF under Binary additive noise case using incremental adaptive algorithm

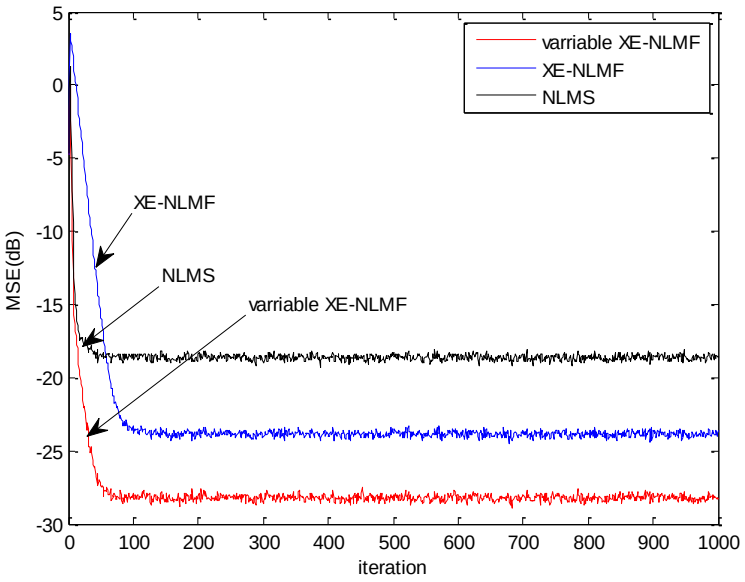

Fig. 46 MSE performance of NLMS, XE-NLMF and variable XE-NLMF under AWGN case using incremental adaptive algorithm

Chapter 5 CONVEX COMBINATION OF ADAPTIVE FILTER

The convex combination of adaptive filter comes into picture to improve the performance of filter which is not possible by a single designed filter.in such type combination a set of filters connected via a supervisor to achieve a universal combination and improves the filter performance in the MSE(mean square error) sense. The filters take part in the combination may have different order, different step size and different adaptive algorithm. In such type of combination the adaptive parameter aggregates the components of filters via convex combination, so that the resulting combination achieves faster convergence and higher accuracy in steady state, and also shows the better tracking property if the combined parameter are properly adapted [22].

5.1 PARALLEL INDEPENDENT STRUCTURE

In parallel independent structure two filters are connected in parallel, having independent adaptive filter operation. Convex combination of two parallel independent filter [10] shown in Fig.47. Here two type filter are used to form the convex combination. One is the fast filter (LMS1,having large step size) and another is accurate filter (LMS2, having small step size). Two filters are convexly combined through a combining parameter$\lambda(i)$. The overall weight vector for this type of combination is given by

$$w_i = \lambda(i)w_{1,i} + [1 - \lambda(i)]w_{2,i} \tag{5.1.1}$$

Where $w_{1,i}$ and $w_{2,i}$ are individual LMS filters updates independently according the LMS updation method [5]given by

$$w_{k,i} = w_{k,i-1} + \mu_k u_i^* \left[d(i) - u_i w_{k,i-1}\right], \quad k = 1,2 \tag{5.1.2}$$

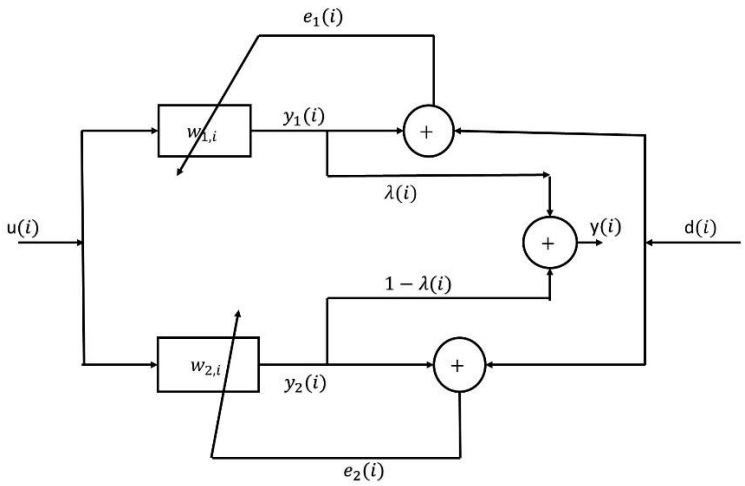

Fig. 47 convex combination of two adaptive filter

Here u_i is an input regressor vector of size $1 \times M$, variance ${\sigma_u}^2$ and μ_k is an filter step size. The plant output model is given by $d(i) = u_i w^0 + v(i)$. Where $v(i)$ a gaussian is noise with variance ${\sigma_v}^2$ and w^0 is a $M \times 1$ column vector that models the unknown plant.

The combining parameter $\lambda(i)$ plays an important role in the convex combination of the filter, it is also known as activation function [22] given by:

$$\lambda(i) = \frac{1}{1+e^{-a(i)}} \tag{5.1.3}$$

Where the parameter $a(i)$ is used to minimize the error $e(i) = d(i) - u_i e_{i-1}$ and the updated equation for $a(i)$ given by

$$a(i) = a(i-1) + \mu_a e(i)[y_1(i) - y_2(i)]\lambda(i)[1-\lambda(i)] \tag{5.1.4}$$

Where $y_k(i) = u_i w_{k,i-1}, k = 1,2$ and μ_a is the step size. The resulting algorithm is known as convex LMS algorithm. Let we perform a simulation to study how the convex algorithm works, for this purpose take$\mu_1 = 0.07, \mu_2 = 0.007, \mu_a = 1000, {\sigma_u}^2 = 1 \; and \; {\sigma_v}^2 = 10^{-3}$. The EMSE (excess mean square error) given by$EMSE = E|u_i(w^0 - w_{i-1})|^2$. Fig.2-41 represents the EMSE curve for convex LMS algorithm, fast filter (LMS1) and accurate filter (LMS2). The Fig.48 clearly

indicates that the convex combination algorithm track the transient response of the faster filter (LMS1) and reach the steady state performance of the more accurate filter(LMS2). Fig.49 represents the convex combination algorithm using incremental adaptive algorithm.

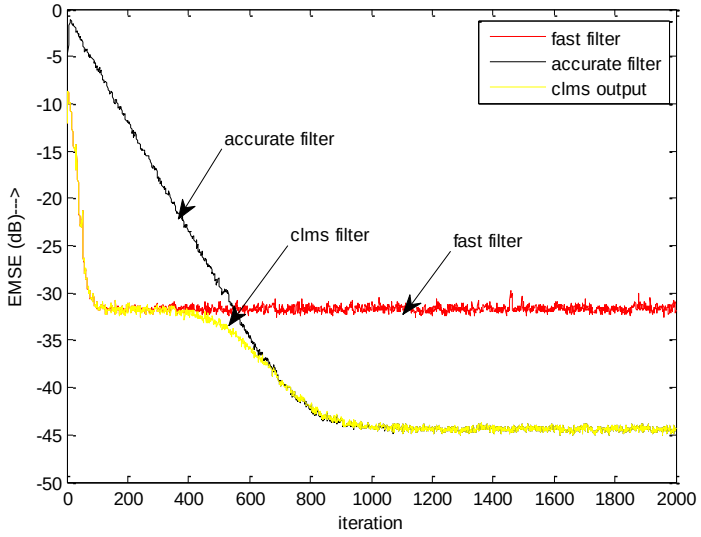

Fig. 48 EMSE of the LMS filter and their convex combination averaged over 200 realization

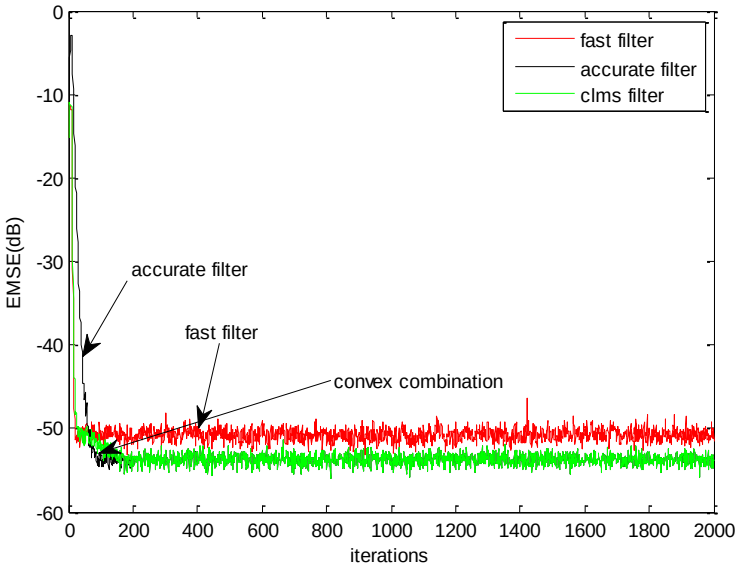

Fig. 49 EMSE of the LMS filter and their convex combination averaged over 200 realization using incremental adaptive algorithm

5.2 SERIES COOPERATIVE STRUCTURE

The CLMS algorithm having advantage over the component of the filter, but it is clear that the transient performance of the accurate filter is not relevant and the steady state of the fast filter is wasted i.e. maximum time performance of one filter totally destroyed by the combiner λ. This is caused because of the parallel filter combination and independent operating nature of the filter. In this operation it awaits the accurate filter to catch up the order to quick compute. λ Can be operated as a switching mechanism in stationary environment. The advantage of parallel combination is simple design of the combiner.

The switching time can be reduced by ad-hoc weight transfer$(w_1 \rightarrow w_2)$. Since the accurate filter are corrupted by the higher gradient noise arising from the fast filter, a control mechanism required.

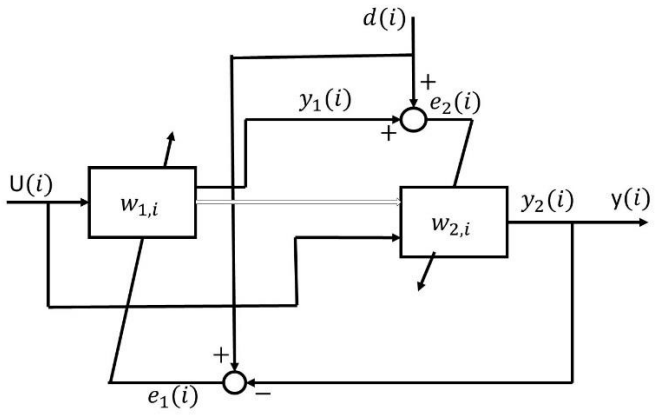

Fig. 50 series topology

5.3 SWITCHING ALGORITHM

The ad-hoc weight transfer process motivated and also implemented without utilizing the control mechanisms. Fig.50 represents topologically series connection of the filter and now λ can be continuously transfer weights. The resulting algorithm is simple , inspired by incremental cooperative structure(INC-COOP1) [10]and can be summarized as follows:

$$\begin{cases} w_{1,i} = w_{i-1} + \mu_1 \lambda(i) u_i^* [d(i) - u_i w_{i-1}] \\ w_{2,i} = w_{1,i} + \mu_2 [1 - \lambda(i)] u_i^* \left[d(i) - u_i w_{1,i} \right] \\ w_i \leftarrow w_{2,i} \end{cases} \quad (5.3.1)$$

The Fig.50 shows that the $\lambda(i) and\ [1 - \lambda(i)]$ are placed inside the block $w_{1,i}\ and\ w_{2,i}$. The incremental step indicated by the dashed arrow. Now the filters are no more independent, they are cooperative and balanced by the factorλ. Here λ not only work as a combiner but also at the same time decreasing the step size.

The potential of the series combination can be improved if we implement the simultaneous operation. For simultaneous operation the λ should be more efficient. The algorithm for the simultaneous operation, inspired by incremental cooperative structure (INC-COOP2) is given by

$$\begin{cases} w_{1,i} = w_{i-1} + \mu_1 \lambda(i) u_i^* [d(i) - u_i w_{i-1}] \\ w_{2,i} = w_{1,i} + \left[\frac{\mu_1 \lambda(i) + [1 - \lambda(i) \mu_2]}{\frac{1}{\gamma}} \right] \\ w_i \leftarrow w_{2,i} \end{cases} \quad (5.3.2)$$

Where $\gamma \in (0,1]$ is a step size contracting factor, it is used to improve the steady state performance, at the same time maintaining the transient performance. λ Should be design in such a way such that adaptive filters are operate simultaneously.

5.3.1 Deterministic Design of the Combining Parameter

Now we get a two type of algorithm mentioned in (5.3.1) and (5.3.2), hence we can make a fair comparison between this two algorithms. For this design of λ is very important, it can be chosen as in the case of parallel structure, as given by:

$$\lambda(i) = \frac{1}{1 + e^{s(i-n)}} \quad (5.3.3)$$

Where 'n' is an activation instant and's' controls the curve smoothness. The choice of $[s, n]$ is very important for the better performance and for better meaningful comparison between two algorithm mentioned in (5.3.1) and (5.3.2).

Consider $w^0 = \frac{1}{\sqrt{M}}[1\ 1\ \cdots\ 1]\ and\ M = 10\ with\ signal\ variance\ {\sigma_u}^2 = 1\ and\ {\sigma_v}^2 = 10^{-3}$.let the step size for fast filter is $\mu_1 = 0.07\ and\ for\ accurate\ filter\ is\ \mu_2 = 0.007$. for CLMS we have $[s = 0.012, n = 550\ ]\ and\ for\ both$ INC-COOP the value will be $[s = 0.015, n = 120\ ]$. for INC-COOP2 the value of γ taken is $\gamma = 0.1$.

Note that in Fig.52 it is clear that the algorithm (5.3.1) i.e. INC-COOP1 it follow the CLMS algorithm as mentioned in the parallel case and reproducing the steady state performance of the accurate filter. At the same time the simultaneous performance algorithm mentioned in (5.3.2) i.e. INC-COOP2 produce faster convergence and small steady state error for same λ value. Fig.51 represents the time revolution curve for $\lambda(i)$ for both the algorithm.

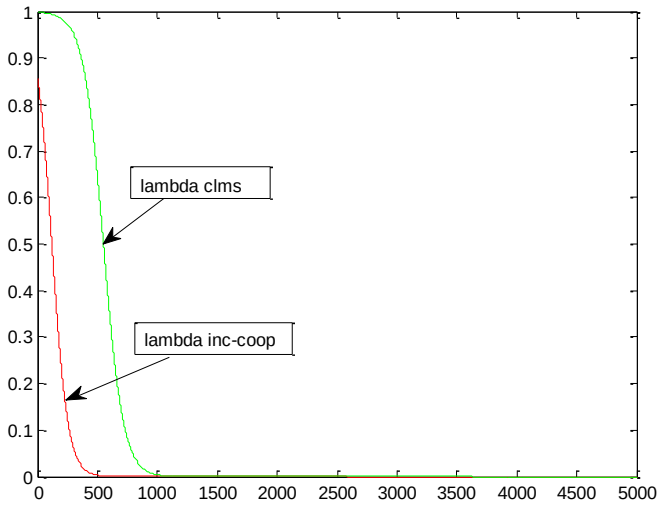

Fig. 51 Time revolution curve of λ(i) for both CLMS and INC-COOP

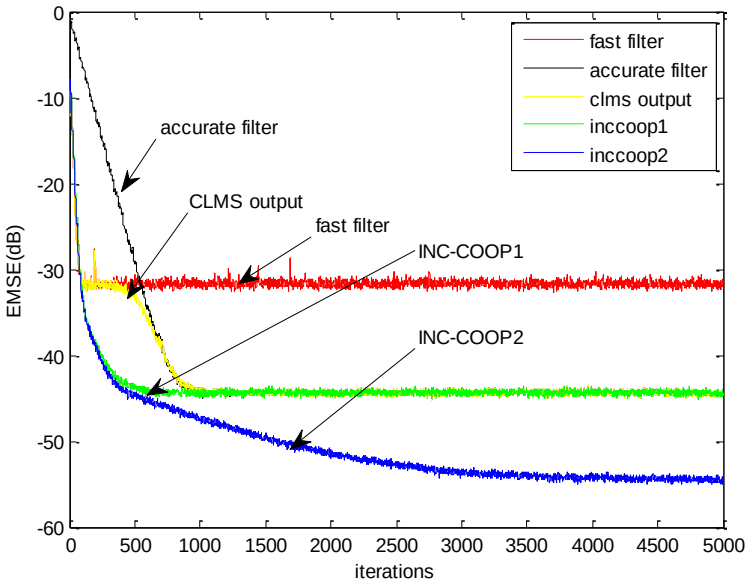

Fig. 52 EMSE curve for the fast filter, accurate filter, CLMS, INCCOOP1 and INC-COOP2 using incremental adaptive algorithm

5.3.2 A Simple Design for the Mixing Parameter

We know that the simple rule in case of adaptive filter is to low pass a quantity let $q(i)$ and feed back to it to the adaptive process, i.e. it is like this

$$a(i) = \alpha a(i-1) + \beta q(i) \tag{5.3.4}$$

Where $\alpha\ lies\ betwwen$ [0,1]. Such type of concepts used in the several application of the adaptive filter such as step-size design, robust filtering and regularization control etc. practically it was found that the value of $\alpha\ lies\ betwwen$ [0.95,0.99] to perform good learning for a wide Signal to Noise ratio(SNR) range. The values of β can be taken as $\beta = 1-\alpha\ or\ < 1-\alpha$. For low signal to noise ratio the value of β taken as $\beta = 0.1(1-\alpha)$. Here $q(i)$ is taken as $e^2(i)$, where $e(i) = d(i) - u_i w_{i-1}$. The value of $\lambda_s(i)$ taken in this case as

$$\lambda_s(i) = \frac{2}{1+e^{-a(i)}} - 1 \tag{5.3.5}$$

λ_s Should be lies in between [0,1]

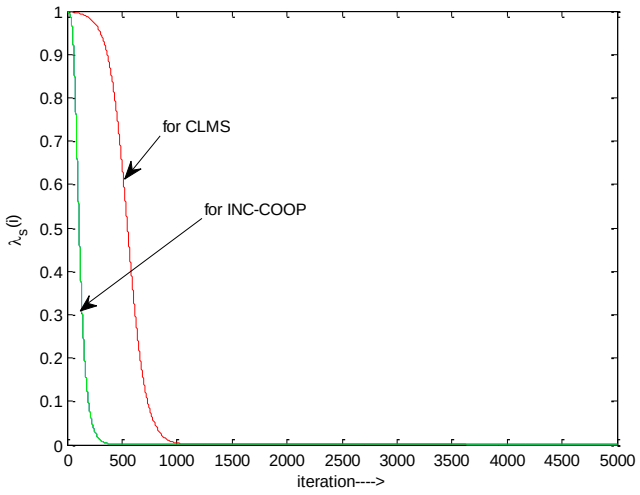

Fig. 53 time evaluation of $\lambda_s(i)$ using the simple design technique for both CLMS and INC-COOP algorithm

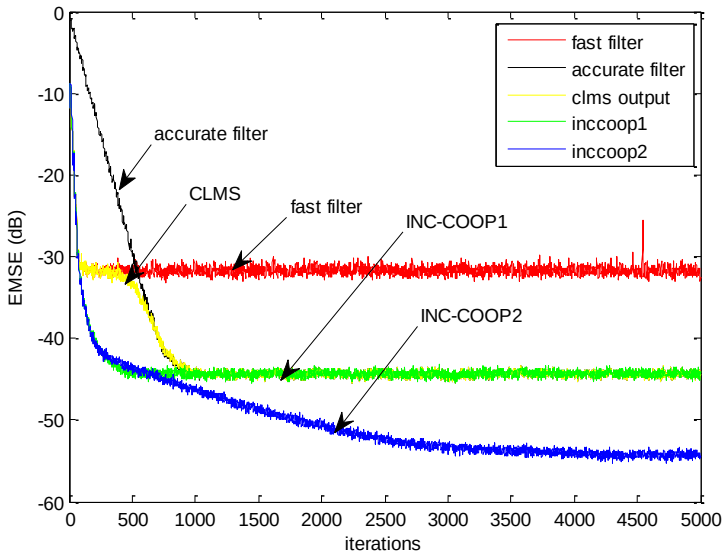

Fig. 54 EMSE performance for fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 algorithm using simple design technique

In all the above case the simulation result carried out for$SNR = 30dB$. Now we will do some simulation to study the performance of all the algorithm for low SNR i.e at SNR=10dB, 5dB and

3dB. Take $\beta = 1 - \alpha$ in the case of SNR=10dB and for the rest case take $\beta = 0.1(1 - \alpha)$. Fig.55, Fig.56 and Fig.57 clearly spectacles the fact that the INC-COOP algorithm always performs better than that of all filter i.e than that of accurate filter, fast filter and CLMS combination.

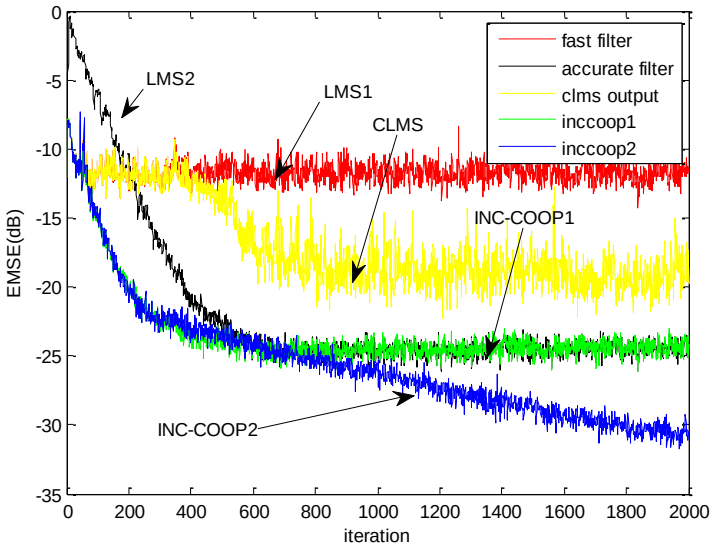

Fig. 55 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=10 dB using incremental adaptive algorithm

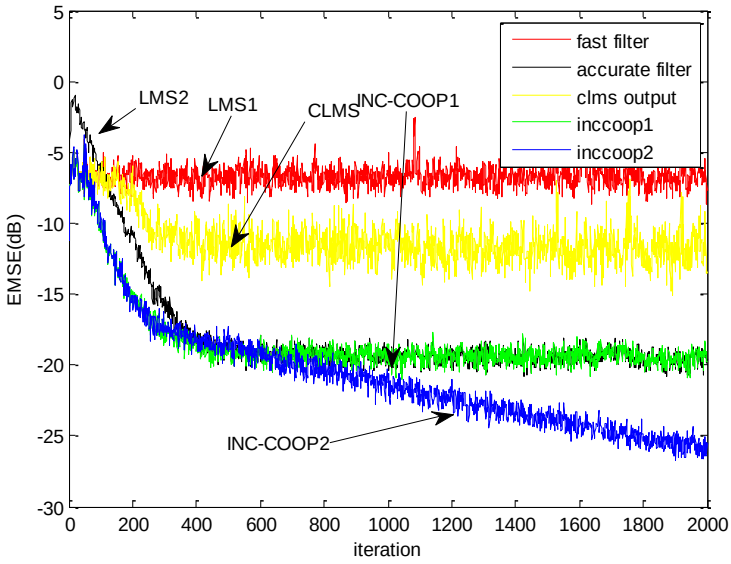

Fig. 56 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=5 dB using incremental adaptive algorithm

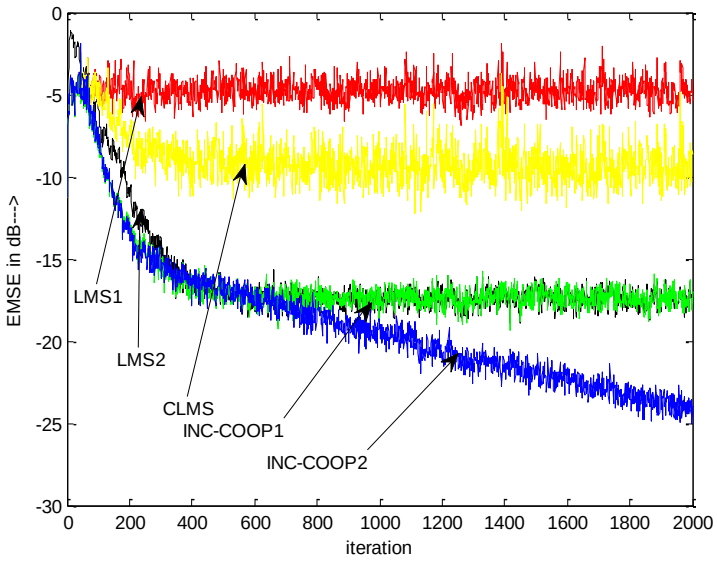

Fig. 57 EMSE performance of fast filter, accurate filter, CLMS, INC-COOP1 and INC-COOP2 for SNR=3 dB using incremental adaptive algorithm

5.4 Conclusion

The CLMS algorithm introduce a new background for combining adaptive filters. Motivated by simple though meaningful scenario, the new technique is able to naturally circumvent the stagnation effect without sacrificing the steady state performance. This is achieved with no extra complexity. The same effect in the parallel-independent case is alleviated only partially and relies on extra weight transfer control mechanisms.

Chapter 6 CONCLUSION AND FUTURE WORK

6.1 Conclusion

Distributed signal processing has wide number of application in the field of signal processing. Day to day number of algorithms are developed to improve the convergence rate, steady state performance and to reduce the computational complexity. Here in this thesis number of algorithms like incremental steepest descent algorithm, incremental adaptive solution, INC RLS, INC GN, INC LMF, INC XE-NLMF, INC variable XE-NLMF, INC CLMS, QWDILMS, INC DCT-LMS, INC DFT-LMS algorithms are tested to achieve the same. In case of INC RLS, INC-GN algorithm it achieve the goal but the computational complexity is more than that of previous. The algorithms are tested under different noise condition and at different SNR case it is found that the lower order error algorithms like INC RLS,INC GN,INC DCT-LMS,INC GN and INC DFT-LMS perform better than that of LMS algorithm under Gaussian noise case, but it fails to achieve the same under non Gaussian noise case like under binary noise , sinusoidal noise and uniform noise. By experiment it is found that the higher order noise algorithm like LMF algorithm, XE-NLMF and variable XE-NLMF algorithm performs better than that of LMS algorithm under non Gaussian noise case.

In all case we consider the SNR is uniform i.e. the variance of noise in all the node present in the network is less than that of one. But it not happens always practically. It is found that in number practical application the SNR of one or more node is less than that of other on that case the algorithms are fails to give better performance by using incremental adaptive strategies. To improve the performance the algorithms like QWDILMS developed which improves the steady state performance under noisy node condition by assigning special weights to each node. But the disadvantage of this algorithm is it only improves the steady state performance but not effects on the convergence rate. But by proper design the convergence rate of the QWDILMS algorithm also will improve.

It is found that the frequency domain incremental algorithm such that INC DCT-LMS and INC DFT-LMS achieves both i.e. the faster convergence rate and better steady state performance than that of INC LMS algorithm with same computational complexity i.e O(M).but the design is little bit complex than that of LMS algorithm.

In distributed signal processing one of the better developments is the CLMS algorithm. Generally the fast filter provides better convergence rate and the accurate filter provides better steady state performance, but it is very difficult to achieve the both in a single filter. Since it is very difficult to design and also very complex, which cannot be reliable. The CLMS algorithm achieves both, by connecting the two filters in parallel or in series. The simulation results provide in the chapter 5 clarifies that the series connection achieves both in better way than that of parallel. CLMS algorithm provides a new platform in the field of adaptive filter to improve the performance with a simple connection.

6.2 Future work

All the algorithms are studied till now by using incremental adaptive strategies .The disadvantage of incremental strategy is that if one the node is failed because of any reason then we cannot recover back the information i.e. the process is stop there , since in incremental strategy the information flow is unidirectional. Hence to overcome this the study and implement of all the algorithm using other mode of cooperation is essential, which comes under my future work. Also study of kalman filter and its application to the real signal processing field, study of blind algorithm and its implementation using different mode of cooperation comes under in my future work.

Bibliography

[1] D. Estrin, L. Girod, G. Pottie and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, 2001.

[2] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *Signal Processing, IEEE Transactions on,* vol. 55, no. 8, pp. 4064-4077, 2007.

[3] M. Wax and T. Kailath, "Decentralized processing in sensor arrays," *Acoustics, Speech and Signal Processing, IEEE Transactions on,* vol. 33, no. 5, pp. 1123-1129, 1985.

[4] D. Li, K. D. Wong, Y. H. Hu and A. M. Sayeed, "Detection, classification, and tracking of targets," *Signal Processing Magazine, IEEE,* vol. 19, no. 2, pp. 17-29, 2002.

[5] A. H. Sayed, Fundamentals of adaptive filtering, John Wiley \& Sons, 2003.

[6] P. I. Hubscher and J. C. M. Bermudez, "An improved statistical analysis of the least mean fourth (LMF) adaptive algorithm," *Signal Processing, IEEE Transactions on,* vol. 51, no. 3, pp. 664-671, 2003.

[7] L. A. Rossi, B. Krishnamachari and C.-C. Kuo, "Distributed parameter estimation for monitoring diffusion phenomena using physical models," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, 2004.

[8] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *Selected Areas in Communications, IEEE Journal on,* vol. 23, no. 4, pp. 798-808, 2005.

[9] M. G. Rabbat and R. D. Nowak, "Decentralized source localization and tracking [wireless sensor networks]," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, 2004.

[10] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: formulation and performance analysis," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, 2006.

[11] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM Journal on Optimization,* vol. 7, no. 4, pp. 913-926, 1997.

[12] W. M. Bazzi, A. RastegarniaR and A. Khalili, "A Quality Aware Incremental LMS Algorithm for Distributed Adaptive Estimation".

[13] T. Panigrahi and G. Panda, "Robust Distributed Learning in Wireless Sensor Network using Efficient Step Size," *Procedia Technology,* vol. 6, pp. 864-871, 2012.

[14] N. Takahashi, I. Yamada and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, 2009.

[15] F. Beaufays, "Transform-domain adaptive filters: an analytical approach," *Signal Processing, IEEE Transactions on,* vol. 43, no. 2, pp. 422-431, 1995.

[16] J. J. Shynk and others, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Processing Magazine,* vol. 9, no. 1, pp. 14-37, 1992.

[17] S. Narayan, A. M. Peterson and M. J. Narasimha, "Transform domain LMS algorithm," *Acoustics, Speech and Signal Processing, IEEE Transactions on,* vol. 31, no. 3, pp. 609-615, 1983.

[18] M. Chan, A. Zerguine and C. Cowan, "An optimised normalised LMF algorithm for sub-Gaussian noise," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, 2003.

[19] A. Zerguine, "Convergence behavior of the normalized least mean fourth algorithm," in *Signals, Systems and Computers, 2000. Conference Record of the Thirty-Fourth Asilomar Conference on*, 2000.

[20] M. Chan and C. Cowan, "Using a normalised LMF algorithm for channel equalisation with co-channel interference," *EUSIPCO-2002, Toulouse, France,* pp. 48-51, 2002.

[21] E. Walach and B. Widrow, "The least mean fourth (LMF) adaptive algorithm and its family," *Information Theory, IEEE Transactions on,* vol. 30, no. 2, pp. 275-283, 1984.

[22] W. B. Lopes and C. G. Lopes, "Incremental-cooperative strategies in combination of adaptive filters.," in *ICASSP*, 2011.

[23] A. Zerguine, "Convergence and steady-state analysis of the normalized least mean fourth algorithm," *Digital Signal Processing,* vol. 17, no. 1, pp. 17-31, 2007.