Adaptive Signal Processing Algorithms for bilinear sparse systems

Thesis to be submitted in partial fulfillment of the requirements for the degree

of

B.Tech

by

Battu SriCharan
17EC10009

Under the guidance of

Dr. Mrityunjoy Chakraborty

ELECTRONICS AND ELECTRICAL COMMUNICATION ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Department of Electronics and Electrical
Communication Engineering
Indian Institute of Technology,
Kharagpur
India - 721302

CERTIFICATE

This is to certify that we have examined the thesis entitled **Adaptive Signal Processing Algorithms for bilinear sparse systems**, submitted by **Battu SriCharan**(Roll Number: *17EC10009*) an undergraduate student of **Department of Electronics and Electrical Communication Engineering** in partial fulfillment for the award of B.Tech degree. We hereby accord our approval of it as a bona fide study carried out and presented in a manner required for its acceptance in partial fulfillment for the undergraduate Degree for which it has been submitted. The thesis has fulfilled all the requirements as per the regulations of the Institute and has reached the standard needed for submission.

Supervisor

Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology,
Kharagpur

Place: Kharagpur
Date:

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof. Mrityunjoy Chakraborty for his constant help and support in this project.I hope to continue this work further. I would also like to thank Dr.Rajib Lochan Das for continuously monitoring my progress and helping with every aspect of this project.

Battu SriCharan
IIT Kharagpur
Date:

ABSTRACT

In this report, motivation for the adoptive algorithms is developed and these algorithms are thoroughly discussed. In practice, many filters are sparse in nature. These algorithms are sparsity agnostic and do not use this knowledge in their method. Algorithms such as ZA-LMS,PNLMS etc. use this knowledge better and achieve improvised results. Improvised versions of these algorithms are discussed to further improve the results and to generalise them for non sparse systems. Bilinear systems are defined and appropriately formulated. Adaptive algorithms for these are discussed.Sparsity of bilinear system is characterised by the sparsity in its temporal part. Most of the bilinear systems are also sparse in nature. Algorithms which assume sparsity perform better than the sparsity agnostic algorithms. These are also thoroughly discussed. Finally simulations on linear and bilinear systems are presented to see how theoretical results work in practice.

Keywords: Adaptive algorithms,Weiner filter, Steepest gradient descent, LMS, NLMS,ZA-LMS,RZA-LMS,PNLMS,IPNLMS,normalized mialignment,bilinear systems,kronecker product.

Contents

List of Figures

Chapter 1

Adaptive Signal Processing Algorithms for linear filters

1.1 Introduction

The traditional filter design is based on the requirements on the frequency response of the output signal desired. Many problems in practice require identification of the parameters of an unknown filter. If the input signal and the corresponding output characteristics are known, it is possible to estimate the filter parameters. We deal with statistical signals in practice. It might not be possible to estimate the exact values of the filter(because of corruption of output due to noise).It could be possible to estimate the filter parameters as closely, depending on the noise power. More the noise power,more is the error in estimation. The correlation matrix of the input vector and the correlation vector between input and output are the statistical quantities, which if known, can estimate the filter values which decrease certain loss functions(that could be errors between estimated output and true output).

However, often these statistical parameters are largely unknown. Even if known, computational cost for the optimal filter estimation is very large. Here is where adaptive filtering comes. Though the filter parameters are unavailable, filters itself will be available and hence,the output to the given input will be known. If we have filter parameters which we assume to be true momentarily, we can calculate the output from this dummy filter(called adaptive filter) and estimate how far the assumed parameters are from the true filter parameters(from the difference between actual output and the dummy output). Based on this knowledge, it is possible to update

our filter(adapt) and reach close to the true parameters. Many methods are proposed based on this approach like Least mean squares(LMS) algorithms, Recursive least squares(RLS) etc.

Bilinear filters are another class of filters, where instead of one input at a time, multiple inputs are given. Thus these have linearity in time as well as space. Adaptive algorithms can be extended to identification of bilinear filters, but the challenges in these methods will be quite different from linear filters. These filters also appear in many practical applications like modelling non linear filters,channel equalization, target detection,chaotic communications, echo cancellation,neural networks etc.

We will start from how LMS algorithm is motivated from the Steepest gradient descent and study various algorithms, which are mostly derivatives of the LMS algorithm

1.1.1 LMS Algorithm:

Least Mean Square(LMS) algorithm is perhaps the most celebrated algorithm in adaptive signal processing. Unlike Weiner filter and Steepest gradient descent, it does not requite any prior information on the stochastic properties of the signals. Such a set up is what is seen in the practical systems. This makes LMS algorithm to be useful in the practical applications.

Steepest gradient descent can be tweaked a little to remove the statistical parameters from the algorithm and introduce the parameters available at hand.

$$w(i+1) = w(i) + \mu(p - R.w(i)) \tag{1.1}$$

where R is the correlation matrix of the input data vector and p is the correlation vector between input and output.

The statistical parameters p and R are not known and hence should be removed to incorporate available data. Since these are expectations, they tend to be the average of the quantities in long term. This averaging is carried over only one sample of the data available at the particular iteration step. Albeit this is not the optimal strategy

to replace the expectation, this gives a justification over replacing the expectations with sample values.

$$R = \mathbf{E}[\underline{x}(i).\underline{x}^T(i)] \approx \underline{x}(i).\underline{x}^T(i) \tag{1.2}$$

$$p = \mathbf{E}[\underline{\mathrm{x}}(i).d(i)] \approx \underline{\mathrm{x}}(i).d(i) \tag{1.3}$$

By replacing the expectations with these terms, Steepest gradient descent can be modified as

$$\begin{aligned}\underline{\mathrm{w}}(i+1) &= \underline{\mathrm{w}}(i) + \mu\{\underline{\mathrm{x}}(i)d(i) - \underline{\mathrm{x}}(i).\underline{\mathrm{x}}^T(i)\underline{\mathrm{w}}(i)\} \\ &= \underline{\mathrm{w}}(i) + \mu\{\underline{\mathrm{x}}(i)[d(i) - \underline{\mathrm{x}}^T(i)\underline{\mathrm{w}}(i)]\} \\ &= \underline{\mathrm{w}}(i) + \mu\{\underline{\mathrm{x}}(i)e(i)\}\end{aligned}$$

The above represented equation is the LMS algorithm, which requires no prior knowledge of the signal properties.

Since the actual expectations are not taken, LMS algorithm may not yield the actual filter values. But LMS algorithm will converge in mean to the actual filter values. The performance of the algorithm is determined by the variance of the filter coefficients given by this algorithm and how they are spread around the actual coefficients.

The convergence of LMS algorithm is heavily influenced by the value of the step size parameter μ. Lower values of μ decreases the misadjustment but also decrease the speed of convergence. Higher values of μ tend to increase the speed of convergence at the cost of misadjustment. Hence, one must carefully tune the value of μ based on the requirements of the application. The convergence analysis of LMS algorithm gives the bounds for the step size parameter μ for the proper convergence of the filter coefficients.

$$0 < \mu < \frac{2}{\lambda_{max}}$$

where λ_{max} is the maximum eigenvalue of R.
A more practical bound is given to be

$$0 < \mu < \frac{2}{Trace(R)}$$

Many algorithms have been derived from LMS algorithm, to establish a higher convergence rate(speed) and to decrease the misalignment.

1.1.2 NLMS Algorithm

One of the main drawbacks of LMS algorithm is it is indifferent to the scaling of the input. The large values of inputs often tend to sensitise the LMS algorithm and it often becomes difficult for the choosing the learning rate. To correct this drawback, normalised values of the input signals are taken instead of the actual values. This algorithm is known as Normalised Least Mean Squares(NLMS) algorithm.

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i+1) + \frac{\mu \underline{\mathrm{x}}(i) e(i)}{\underline{\mathrm{x}}^T(i)\underline{\mathrm{x}}(i)}$$

It is possible that a 0 division error may occur sometimes if the input is a 0 vector. To avoid this, a little tweaking is done in the denominator. A small offset δ is added to the denominator to remove the possibility of 0 division error.

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i+1) + \frac{\mu \underline{\mathrm{x}}(i) e(i)}{\underline{\mathrm{x}}^T(i)\underline{\mathrm{x}}(i) + \delta}$$

The NLMS algorithm is known to perform better than LMS in terms of misadjustment at the cost of convergence rate.

1.2 Sparse Adaptive Algorithms

Many practical systems can be modelled as sparse filters, where most of the coefficients are 0. The well known LMS and NLMS algorithms do not incorporate this aspect in them. These are sparsity agnostic algorithms. LMS algorithm can be tweaked a bit to incorporate this knowledge. There are many algorithms that consider sparsity and can increase both convergence rate and misadjustment.There are two classes of algorithms well-studied in the literature. Zero attractive class and Proportionate normalised class. The zero attractive class enhances the convergence of the 0 coefficients while proportionate class algorithms enhance the convergence of non-zero coefficients.

1.2.1 ZA-LMS Algorithm

If we tweak the algorithm so that the coefficients tend to move toward 0,as most of the coefficients are already known to be 0, it can significantly enhance the overall

convergence of LMS algorithm. This algorithm is known as Zero attractive LMS algorithm(ZA LMS). This introduces extra terms,known as zero attractors, which make the coefficients move toward 0.

While the Loss function for LMS algorithm is mean squared error, the loss function for ZA LMS has an extra penalty on the magnitude of the coefficients.

$$L = \frac{1}{2}e^2 + \gamma||w||_1$$

where ||.|| denotes vector 1-norm
The gradient descent updating step gives the following equations

$$\begin{aligned}\underline{\mathrm{w}}(i+1) &= \underline{\mathrm{w}}(i) - \mu\frac{\partial L}{\partial \underline{\mathrm{w}}} \\ &= \underline{\mathrm{w}}(i) + \mu\underline{\mathrm{x}}(i)e(i) - \rho.sgn(\underline{\mathrm{w}}(i))\end{aligned}$$

where $\rho = \mu\gamma\mathrm{sgn}(.)$ is a component wise sign function defined as follows:

$$\begin{aligned}sgn(x) &= 1 \;\; if \;\; x > 0 \\ &= -1 \;\; if \;\; x < 0 \\ &= 0 \;\; if \;\; x = 0\end{aligned}$$

The convergence of ZA-LMS is studied in literature and the bounds for the step size for low misalignment are well known

The major drawback of ZA-LMS algorithm is that it does not distinguish between zero coefficients and non zero coefficients. Hence although the convergence of zero taps is enhanced, the misalignment of non zero taps is severely affected. For this reason, LMS algorithm outperforms ZA-LMS in terms of misalignment in case of non sparse filters. This problem can be solved using Reweighted ZA-LMS algorithm.

1.2.2 RZA-LMS Algorithm

As stated, the convergence of non zero taps is severely inhibited by zero attractors. If the zero attractors are selectively chosen only for zero coefficients, then this version of ZA-LMS algorithm can outperform LMS algorithm even in case of non sparse

systems.The cost function for RZA LMS algorithm is chosen as follows:

$$L = \frac{1}{2}e^2 + \gamma' \sum_{i=1}^{n} \ln(1 + \frac{|w_i|}{\epsilon'})$$

The zero attractors are reweighted such that they are shrinked for coefficients which tend to be larger in magnitude.

$$w_j(i+1) = w_j(i) + \mu.x_j(i)e(i) - \rho\frac{sgn(w_j(i)}{1 + \epsilon.|w_j(i)|} \quad \forall\ j = 1\ to\ n$$

In vector form, this is written as

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i) + \mu.\underline{\mathrm{x}}(i)e(i) - \rho.\frac{sgn(\underline{\mathrm{w}}(i))}{1 + \epsilon.|\underline{\mathrm{w}}(i)|}$$

where $\rho = \mu.\gamma'/\epsilon'$,$\epsilon = 1/\epsilon'$

The zero attractor effects only those taps whose size is comparable to $1/\epsilon'$. It does not effect those taps whose size is $>> 1/\epsilon'$.

RZA-LMS performs similar to ZA-LMS for sparse system. It performs better than both LMS and ZA-LMS for semi-sparse system and performs similar to LMS algorithm for dispersive(non-sparse) systems in terms of misalignment or Mean square deviation. Hence, RZA-LMS shows better performance than LMS or ZA-LMS algorithms on an average.

Now we will look into the class of algorithms known as proportionate class of algorithms for sparse systems. Though Both ZA class and proportionate class are specialised algorithms for sparse systems, their behaviour is quite opposite. While the ZA-LMS tend to enhance the convergence of zero coefficients, the proportionate class tend to enhance the convergence of non zero coefficients.

1.2.3 PNLMS Algorithm

The proportionate normalized least mean square(PNLMS) algorithm was originally developed for use in network echo cancellers. In all the algorithms previously discussed, the step size is fixed and constant for all the coefficients. In Proportionate class algorithms, the step sizes are different for each coefficient and vary with each

iteration. The step size is proportional to the magnitude of the filter coefficient. PNLMS algorithm is characterised by the following equations.

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i) + \frac{\mu G(i)\underline{\mathrm{x}}(i)e(i)}{\underline{\mathrm{x}}^T(i)G(i)\underline{\mathrm{x}}(i) + \delta_{PNLMS}}$$

where $G(i) = diag\{g_0(i), g_1(i), ..., g_n(i)\}$ Here,

$$g_l(i) = \frac{\gamma_l(i)}{\sum_{j=1}^{n}\gamma_j(i)} \quad 0 \leq l \leq n$$
$$\gamma_l(i) = max\{\rho.max[\delta_p, |w_0(i)|, |w_1(i)|, ..., |w_n(i)|], |w_l(i)|\} \quad 0 \leq l \leq n$$

Parameters δ_p and ρ are positive numbers with typical values δ_p=0.01 and $\rho = 5/\mathrm{n}$. The term ρ prevents w_l(i) from stalling when it is much smaller than the largest coefficient and δ_p regularizes the updating when all coefficients are zero at initialization.

The PNLMS shows very fast convergence initially but slows down eventually. The convergence of PNLMS is not that better even for relatively sparse systems. PNLMS gives a large convergence rate for large coefficients boosting the convergence rate but at the cost of misalignment for these coefficients. Since a low step size is provided for zero coefficients, they don't update much fast, remaining near 0. Hence there will be an initial fast convergence. Hence for sparse systems, this looks like a better algorithm since the non zero coefficients are very few in number.Hence, their misalignment might not impact the overall misalignment that much.It should be noted that the misalignment remains more or less same for PNLMS and NLMS for any type of systems. The overall convergence rate will be faster for PNLMS for sparse case. However, as the number of non zero coefficients increase, this will impact the overall performance much severely in terms of rate of convergence, but the misalignment remains same.

1.2.4 IPNLMS Algorithm

Instead of taking naive magnitude to exploit the proportionate idea, a better measure can be used to consider the proportional value for the step sizes. The improved proportionate normalized least mean squares(IPNLMS) algorithm one such algorithm

that makes a better use of proportionate idea.The algorithm is described by the following equations:

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i) + \frac{\mu.K(i).\underline{\mathrm{x}}(i).e(i)}{\underline{\mathrm{x}}^T(i).K(i).\underline{\mathrm{x}}(i) + \delta_{IPNLMS}}$$

where K(i)=diag$\{k_1(i), k_2(i), ..., k_n(i)\}$.
Here,

$$k_l(i) = \frac{1-\alpha}{2n} + (1-\alpha)\frac{|w_l(i)|}{2||\underline{\mathrm{w}}(i)||}$$

However in practice, to avoid the 0-division error,a small value ϵ is introduced in the denominator:

$$k_l(i) = \frac{1-\alpha}{2n} + (1-\alpha)\frac{|w_l(i)|}{2||\underline{\mathrm{w}}(i)|| + \epsilon}$$

Here α is a value taken between -1 and 1. It can be seen that by taking α as 1, IPNLMS behaves similar to PNLMS. If α is taken to be -1, it behaves similar to NLMS. A middle ground is to take α as 0 or 0.5. With these IPNLMS can be used for any filter without any loss.

There are other algorithms like MPNLMS , filters based on convex combination of LMS and ZA-LMS filter, ZA-PNLMS algorithms etc. which can give better misalignment and convergence rates.

Chapter 2

Bilinear systems and adaptive algorithms for them

2.1 Bilinear Systems

Bilinear systems find their applications in various domains. Among these, we can mention prediction problems, channel equalization,echo cancellation, chaotic communication,active noise control,neural networks etc. However, in almost all these frameworks, bilinear term is defined with respect to the data i.e in terms of input output relation.

Here, we focus on a different approach by defining the term bilinear with respect to impulse responses of a spatio-temporal model, in the context of multi input single output(MISO) systems.This problem had been addressed to form a system identification perspective and two forms of Weiner filter were developed in this context. However, Weiner filters are not feasible to implement because of several reasons(matrix inversion, statistical feature estimation etc.). Naturally, the next step is to analyze this framework in adaptive filtering approach.

2.1.1 Signal model with Bilinear forms

The bilinear term is defined with respect to the impulse responses of spatio-temporal model,in the context of MISO systems.Consequently, the signal model is:

$$\begin{aligned} d(i) &= \underline{\mathrm{h}}^T X(i) \underline{\mathrm{g}} + s(i) \\ &= y(i) + s(i) \end{aligned}$$

where d(n) is the zero mean desired signal at the discrete time index n, $\underline{h}$and $\underline{g}$are two impulse responses of the system of lengths L and M respectively.

$$X(i) = [\underline{x}_1(i), \underline{x}_2(i), ..., \underline{x}_M(i)]$$

is the zero-mean multi input signal matrix of size LxM

$$\underline{x}_m(i) = [x_m(i), x_m(i-1), ..., x_m(i-L+1)]$$

is a vector containing the L most recent samples of the mth(m=1,2,...,M) input signal,$y(i) = \underline{h}^T X(i)\underline{g}$ is the bilinear form and s(i) is the zero mean additive noise.It is assumed that X(i) and s(i) are uncorrelated.

The two impulse responses $\underline{h}$ and $\underline{g}$ correspond to the temporal and spatial parts of the system respectively.It is easy to verify that for every fixed $\underline{h}$, y(i) is a linear function of $\underline{g}$ and for every fixed $\underline{g}$, y(i) is a linear function of $\underline{h}$.Therefore, y(i) is bilinear in $\underline{h}$ and $\underline{g}$.

Based on the vectorization operation (i.e., conversion of a matrix into a vector), the matrix X(n)of size L×M can be rewritten as a vector of length ML

$$\begin{aligned} vec[X(i)] &= [\ \underline{x}_1^T(i), \underline{x}_2^T(i), ..., \underline{x}_m^T(i)\ ]^T \\ &= \widetilde{\underline{x}}(i) \end{aligned}$$

It can be noticed that the output can be expressed as

$$\begin{aligned} y(i) &= \underline{h}^T X(i)\underline{g} \\ &= Tr[\underline{h}^T X(i)\underline{g}] \\ &= Tr[\underline{g}\underline{h}^T X(i)] \\ &= Tr[(\underline{h}\underline{g}^T)^T X(i)] \\ &= vec^T(\underline{h}\underline{g}^T).vec[X(i)] \\ &= [\underline{g} \otimes \underline{h}]^T \widetilde{\underline{x}}(i) \\ &= \underline{f}^T \widetilde{\underline{x}}(i) \end{aligned}$$

where Tr[.] denotes trace, $\otimes$ is the kronecker product and $\underline{f} = \underline{g} \otimes \underline{h}$ is the spatio-temporal response of length ML. Hence the signal model results in :

$$d(i) = \underline{f}^T \widetilde{\underline{x}}(i) + s(i)$$

Clearly, the output depends on $\underline{f}$.It is possible that normalised versions of $\underline{h}$ and $\underline{g}$ can produce the same out as shown belowe:

$$\begin{aligned}\underline{f} &= \underline{g} \otimes \underline{h} \\ &= (\eta \underline{g}) \otimes (\frac{1}{\eta}\underline{h})\end{aligned}$$

Though the adaptive algorithms for bilinear systems can be viewed in the light of single input single output systems,the convergence of these algorithms take too long as the spatio-temporal response length is much greater than either of spatial filter and temporal filter. The convergence speed will be more for the filters which have fewer coefficients. For this very reason, adaptive algorithms should be analysed separately for bilinear systems.

We use the normalized misalignment as a performance measure for $\underline{f}$

$$NM(\underline{f}, \hat{\underline{f}}) = \frac{||\underline{f} - \hat{\underline{f}}||^2}{||\hat{\underline{f}}||^2}$$

where $\hat{\underline{f}}$ is the adaptive spatiotemporal filter that is being updated.

However since the estimation of $\underline{f}$ and $\underline{g}$ has ambiguity with a scaling factor, we use the normalized projection misalignment(NPM) for $\underline{f}$ and$\underline{g}$

$$NPM(\underline{g}, \hat{\underline{g}}) = 1 - (\frac{< \underline{g}^T, \hat{\underline{g}} >}{||\underline{g}||||\hat{\underline{g}}||})^2$$

$$NPM(\underline{h}, \hat{\underline{h}}) = 1 - (\frac{< \underline{h}^T, \hat{\underline{h}} >}{||\underline{h}||||\hat{\underline{h}}||})^2$$

where $||.||$ denotes euclidean norm and $< ., . >$ denotes vector inner product, $\hat{\underline{g}}$ is the adaptive spatial filter that is being updated and $\hat{\underline{h}}$ is the adaptive temporal filter that is being updated

2.2 Adaptive Algorithms for Bilinear systems

We will now analyse bilinear systems in the framework of adaptive filtering.Although two forms of weiner filter were proposed for bilinear systems,namely direct and iterative,the practical limitations of them make them difficult to implement. Hence,

it can be expected that the LMS algorithm and its variations will overcome these limitations and perform well in practice.

Let us consider two adaptive filters $\hat{\underline{h}}$(i) and$\hat{\underline{g}}$(i) and the estimated signal $\hat{y}(i+1) = \hat{\underline{h}}^T(i)X(i)\hat{\underline{g}}(i)$,where X(i) is the data matrix available at the iteration i. We have the following definitions for errors

$$\begin{aligned} e(i+1) &= d(i+1) - \hat{y}(i+1) \\ &= d(i+1) - \hat{\underline{h}}^T(i)X(i+1)\hat{\underline{g}}(i) \\ &= d(i+1) - [\hat{\underline{g}}(i) \otimes \hat{\underline{h}}(i)]^T\widetilde{\underline{x}}(i+1) \\ &= d(i+1) - \hat{\underline{f}}^T(i)\widetilde{\underline{x}}(i+1) \end{aligned}$$

where $d(i) = \hat{\underline{h}}^T X(i)\hat{\underline{g}} + s(i)$.
Alternatively, we may define

$$\begin{aligned} e_{\hat{\underline{g}}}(i+1) &= d(i+1) - \hat{\underline{h}}^T(i)\widetilde{\underline{x}}_{\hat{\underline{g}}}(i+1) \\ e_{\hat{\underline{h}}}(i+1) &= d(i+1) - \hat{\underline{g}}^T(i)\widetilde{\underline{x}}_{\hat{\underline{h}}}(i+1) \end{aligned}$$

where

$$\begin{aligned} \widetilde{\underline{x}}_{\hat{\underline{g}}}(i+1) &= [\hat{\underline{g}}(i) \otimes I_L]^T\widetilde{\underline{x}}(i+1) \\ \widetilde{\underline{x}}_{\hat{\underline{h}}}(i+1) &= [I_M \otimes \hat{\underline{h}}(i)]^T\widetilde{\underline{x}}(i+1) \end{aligned}$$

where I_L and I_M are identity matrices of size LxL and MxM respectively.It can be verified that $e_{\hat{\underline{g}}}(i+1) = e_{\hat{\underline{h}}}(i+1) = e(i+1)$. However for clarity, we keep $e_{\hat{\underline{g}}}(i+1)$ and $e_{\hat{\underline{h}}}(i+1)$.

2.2.1 LMS-BF algorithm

Now we formulate the LMS algorithm for bilinear filters,namely (LMS-BF)

$$\begin{aligned} \hat{\underline{h}}(i+1) &= \hat{\underline{h}}(i) - \frac{\mu_{\hat{\underline{h}}}}{2}\frac{\partial e^2_{\hat{\underline{g}}}(i+1)}{\partial\hat{\underline{h}}(i)} \\ &= \hat{\underline{h}}(i) + \mu_{\hat{\underline{h}}}.\widetilde{\underline{x}}_{\hat{\underline{g}}}(i+1).e_{\hat{\underline{g}}}(i+1) \end{aligned}$$

and

$$\begin{aligned} \hat{\underline{g}}(i+1) &= \hat{\underline{g}}(i) - \frac{\mu_{\hat{\underline{g}}}}{2}\frac{\partial e^2_{\hat{\underline{h}}}(i+1)}{\partial\hat{\underline{g}}(i)} \\ &= \hat{\underline{g}}(i) + \mu_{\hat{\underline{g}}}.\widetilde{\underline{x}}_{\hat{\underline{h}}}(i+1).e_{\hat{\underline{h}}}(i+1) \end{aligned}$$

where $\mu_{\hat{\underline{h}}>0}$ and $\mu_{\hat{\underline{g}}>0}$ are the step size parameters.

There is an LMS algorithm for $\hat{\underline{f}}$ as well which can be implemented independent of the above equations.

$$\hat{\underline{f}}(i+1) = \hat{\underline{f}}(i) + \mu_{\hat{\underline{f}}}.\widetilde{\underline{x}}(i+1).e(i+1)$$

However the filter size of $\underline{f}$ is much greater than both $\underline{h}$ and $\underline{g}$ and hence its convergence is slower than the LMS-BF algorithm.The convergence of LMS-BF is well established in literature in terms of the bounds that sep sizes can take and the theoretical misalignment values. Simulation results were observed to match the theoretical values.

2.2.2 NLMS-BF algorithm

Due to the drawbacks similar to NLMS algorithm, LMS-BF can be modified to normalised LMS(NLMS-BF) to achieve better convergence results.The following equations describe the NLMS-BF algorithm.

$$\hat{\underline{h}}(i+1) = \hat{\underline{h}}(i) + \frac{\mu_{\hat{\underline{h}}}.\widetilde{\underline{x}}_{\hat{\underline{g}}}(i+1).e_{\hat{\underline{g}}}(i+1)}{\widetilde{\underline{x}}^T_{\hat{\underline{g}}}(i+1)\widetilde{\underline{x}}_{\hat{\underline{g}}}(i+1) + \delta_{\hat{\underline{h}}}}$$

$$\hat{\underline{g}}(i+1) = \hat{\underline{g}}(i) + \frac{\mu_{\hat{\underline{g}}}.\widetilde{\underline{x}}_{\hat{\underline{h}}}(i+1).e_{\hat{\underline{h}}}(i+1)}{\widetilde{\underline{x}}^T_{\hat{\underline{h}}}(i+1)\widetilde{\underline{x}}_{\hat{\underline{h}}}(i+1) + \delta_{\hat{\underline{g}}}}$$

where $\delta_{\hat{\underline{g}}}$ and $\delta_{\hat{\underline{h}}}$ are the small offset values to avoid 0-division error.
The convergence of NLMS-BF is also well established in literature and is known to perform better than LMS at the cost of computational overhead.

2.3 Sparse Adaptive Algorithms for Bilinear filters

In practical systems,Sparsity in bilinear filters is mostly due to sparsity in temporal filter. It is common in bilinear systems that temporal filter is sparse while the spatial filter is dense. It is also common that the temporal filter is of more length than the spatial filter, which is relatively short. In case of non sparsity, the temporal filter will take longer to converge than the spatial filter, because of its length. If we make use of sparsity information, the convergence of temporal filters can be made more or less

similar to that of spatial filter.

We can modify the updating equation for temporal filter to incorporate sparsity into it, while keeping the spatial filter equation as it is. The proportionate class algorithms for bilinear systems work well, similar to PNLMS and IPNLMS. PNLMS faces same drawbacks: initial fast convergence for sparse systems, but very slow for others. One interesting aspect is that Zero attractors class has not been found in literature for bilinear systems. When attempted to simulate ZA-LMS-BF, the convergence show unexpectedly weird behaviour.

We will now look into proportionate class algorithms for bilinear systems.

2.3.1 PNLMS-BF

Since, the spatial system is established to be non sparse, NLMS-BF can be used for this filter,giving a uniform step size for all coefficients. The temporal filter can be given non-uniform step size based on the coefficient magnitudes. PNLMS-BF algorithm can be described by the following equations:

$$\underline{\hat{\mathrm{h}}}(i+1) = \underline{\hat{\mathrm{h}}}(i) + \frac{\mu_{\underline{\hat{\mathrm{h}}}}.G(i).\underline{\widetilde{\mathrm{x}}}_{\hat{\mathrm{g}}}(i+1).e_{\hat{\mathrm{g}}}(i+1)}{\underline{\widetilde{\mathrm{x}}}^T_{\hat{\mathrm{g}}}(i+1).G(i).\underline{\widetilde{\mathrm{x}}}_{\hat{\mathrm{g}}}(i+1) + \delta_{\underline{\hat{\mathrm{h}}}PNLMS}}$$

$$\hat{\mathrm{g}}(i+1) = \hat{\mathrm{g}}(i) + \frac{\mu_{\hat{\mathrm{g}}}.\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1).e_{\underline{\hat{\mathrm{h}}}}(i+1)}{\underline{\widetilde{\mathrm{x}}}^T_{\underline{\hat{\mathrm{h}}}}(i+1)\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1) + \delta_{\hat{\mathrm{g}}PNLMS}}$$

where G(i)=diag{ $g_1(i), g_2(i), ..., g_L(i)$ } and

$$g_l(i) = \frac{\gamma_l(i)}{\sum_{j=1}^{L}\gamma_j(i)} \quad 1 \le l \le L$$

$$\gamma_l(i) = max\{\ \rho.max[\ \delta_p, |\hat{h}_1(i)|, |\hat{h}_1(i)|, ..., |\hat{h}_L(i)|\ ], |\hat{h}_l(i)|\ \} \quad 1 \le l \le L$$

where $\underline{\hat{\mathrm{h}}}(i) = [\ \hat{h}_1(i), \hat{h}_2(i), ..., \hat{h}_L(i)\ ]$ and δ_p avoids stalling of the progress when the adaptive filter coefficients are initially taken $0.\rho$ avoids stalling of progress of individual coefficients when they are much smaller than the highest coefficient.

PNLMS-BF has same drawbacks as in PNLMS algorithm. It shows initial fast convergence but eventually slows down. It works well for sparse systems, although it does not effect misalignment. However for non sparse system, the convergence is too slow. IPNLMS-BF was proposed to rectify these drawbacks.

2.3.2 IPNLMS-BF Algorithm

The proportionate idea is better used in Improved PNLMS-BF than in PNLMS-BF. where it naively takes the magnitude directly for step size. The below equations describe the IPNLMS-BF algorithm:

$$\underline{\hat{\mathrm{h}}}(i+1) = \underline{\hat{\mathrm{h}}}(i) + \frac{\mu_{\underline{\hat{\mathrm{h}}}}.G(i).\underline{\widetilde{\mathrm{x}}}_{\hat{\mathrm{g}}}(i+1).e_{\hat{\mathrm{g}}}(i+1)}{\underline{\widetilde{\mathrm{x}}}^T_{\hat{\mathrm{g}}}(i+1).G(i).\underline{\widetilde{\mathrm{x}}}_{\hat{\mathrm{g}}}(i+1) + \delta_{\underline{\hat{\mathrm{h}}}IPNLMS}}$$

$$\hat{\mathrm{g}}(i+1) = \hat{\mathrm{g}}(i) + \frac{\mu_{\hat{\mathrm{g}}}.\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1).e_{\underline{\hat{\mathrm{h}}}}(i+1)}{\underline{\widetilde{\mathrm{x}}}^T_{\underline{\hat{\mathrm{h}}}}(i+1)\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1) + \delta_{\hat{\mathrm{g}}IPNLMS}}$$

where G(i)=diag{ $g_1(i), g_2(i), ..., g_L(i)$ } and

$$g_l(i) = \frac{\gamma_l(i)}{\sum_{j=1}^{L}\gamma_j(i)} \quad 1 \le l \le L$$

$$\gamma_l(i) = \frac{1-\alpha}{2L} + (1+\alpha).\frac{|\hat{h}_l(i)|}{2||\underline{\hat{\mathrm{h}}}(i)||_1 + \epsilon} \quad 1 \le l \le L$$

where ϵ is a small bias term added to avoid 0-division error and α is a value in between -1 and 1. It can be seen that if $\alpha = 1$, then IPNLMS-BF behaves same like PNLMS-BF. If $\alpha = -1$, then IPNLMS-BF behaves same like NLMS. A good choice is to take middle ground where α=0 or -0.5. In that it behaves on par with PNLMS for sparse systems and NLMS for dispersive(non-sparse) systems. It behaves better than both for semi-sparse systems.

As already stated, zero-attractive class is not yet studied in literature. When simulations were attempted to apply ZA-LMS for bilinear systems, the convergence results showed unexpected behaviours. Convex combination of ZA-LMS and LMS algorithm is not yet studied in literature and they are likely to perform well for bilinear systems as well.

Chapter 3

Simulations

3.1 Adaptive Algorithms for linear filters

We will now look at simulations, that demonstrate certain theoretical results.The explained Adaptive algorithms for linear filters are simulated and the normalized misalignment is plotted.Though Mean square deviation can also be considered as a performance measurement(in fact it is practically viable to measure than the misalignment), taking a single experiment produces results with high variance. Hence many experiments(of order of 1000 for neat results) have to be simulated and the average mean square error has to be taken.A better measurement is normalized misalignment Although we do not give the theoretical values for normalized misalignment expected to be reached, the values reached in the simulations have been matched to the theoretical values very closely.

We will look into different aspects of the algorithms, how they vary with the parameters of the algorithm(especially the step size) and consider the trade off between speed of convergence and normalized misalignment.

Throughout the simulations, the input data generated is zero mean white gaussian with variance 1, until and unless otherwise stated. A zero mean white gaussian noise corrupts the data output and affects the misalignment even though the other parameters remain same

It can be seen from figure 3.1 that if the noise power is more, then LMS algorithm will not perform its best. It should be noted that the ill effect is because of the innate limit which does not allow any algorithm to exceed this limit. From now on, the noise

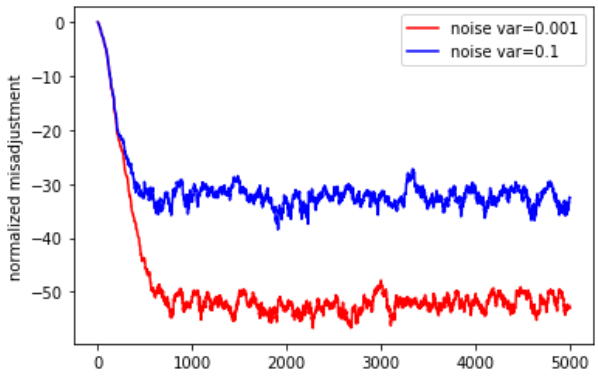


Figure 3.1: The effect of noise on LMS algorithm's performance.μ=0.01 for both the filters.Filter size is 16

variance is fixed to be 0.001, until and otherwise stated.

One of the key parameter that characterizes the LMS algorithm and its derivates is step size μ.From figure 3.2,it is evident that even though higher value of step size tend to converge faster, it's misalignment gets affected.

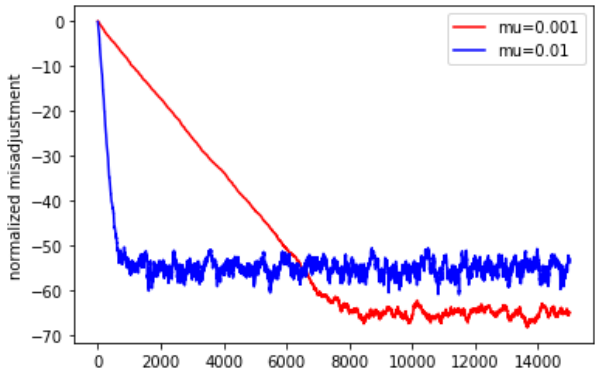


Figure 3.2: The effect of step size on LMS algorithm.The filter length L is 16 in both cases

From now on, μ is fixed to be 0.01 until and otherwise stated.

The convergence of the algorithm also depends on the filter length. When the other parameters of the algorithm are fixed, higher length filters tend to converge slower than the smaller filters. This effect is demonstrated in figure 3.3.

It is already noted that there are theoretical limits for convergence of the algorithm.

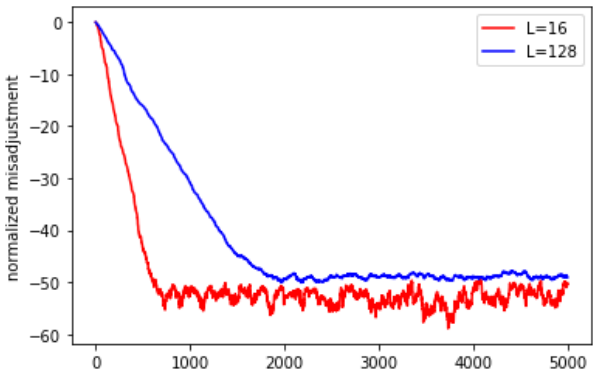


Figure 3.3: The effect of filter length on LMS algorithm.μ=0.01 in both cases

The approximate bounds are $0 < \mu < \frac{2}{Tr(R)}$. Since, data is white guassian with variance 1, $R = I_L$. With L=16,Tr(R)=16. Therefore, for convergence, $0 < \mu < 0.125$.It can be seen from figure3.4,that divergence is attained in case the step size has exceeded the bounds. Hence, it must be taken care that the step size should stay within the limits.

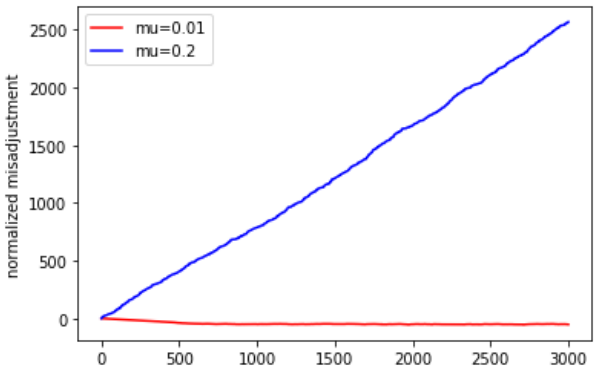


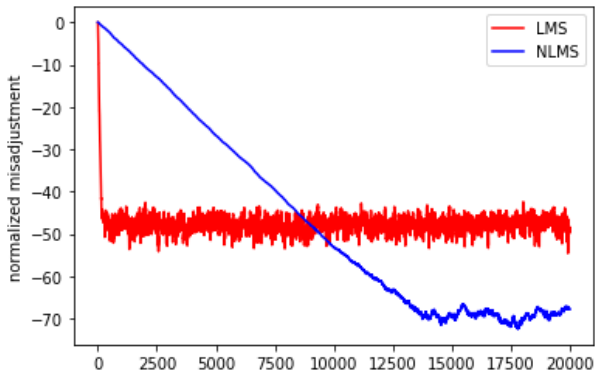
Figure 3.4: Divergence in case the step size exceeded its theoretical bounds.L=16

. The above general statements on the effect of convergence are true for any filter derived from LMS algorithm. It should be taken care that the parameters are chosen reasonably for proper convergence and based on the application.

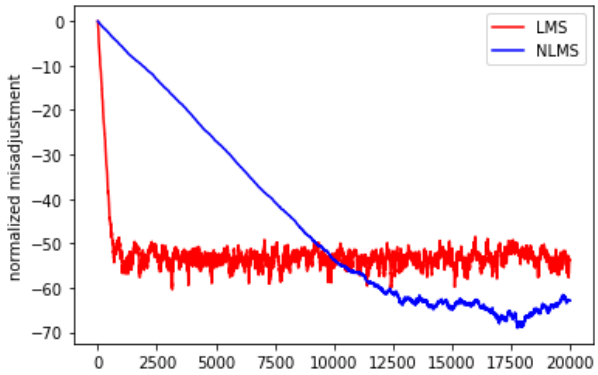
we will now look into other algorithms and how they compare among themselves.

It could be seen from figure 3.5b that LMS algorithm converges much faster than NLMS,but has moremisalignment as well. It could also be seen from 3.5a, that when

.5



(a) NLMS vs LMS when input variance=5,L=16,$\mu = 0.01$



(b) NLMS vs LMS when input variance=1,L=16,$\mu = 0.01$,$\delta_{NLMS} = 0.01$

Figure 3.5: Effect of data scaling on LMS and NLMS

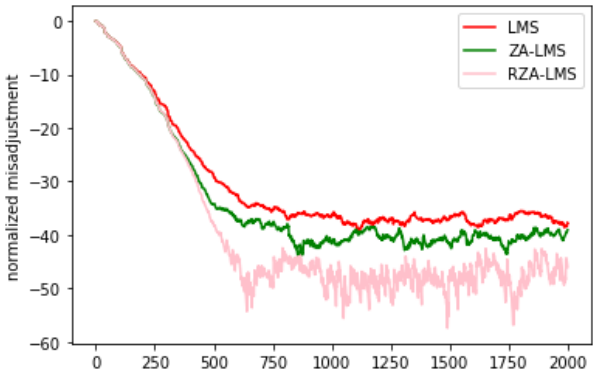
data variance is more(scaled), then LMS tends to increaszzze the misalignment, while NLMS is insensitive to data scaling.

From now on, the data variance = 1, noise variance=0.001, μ=0.01 is fixed until and otherwise stated.

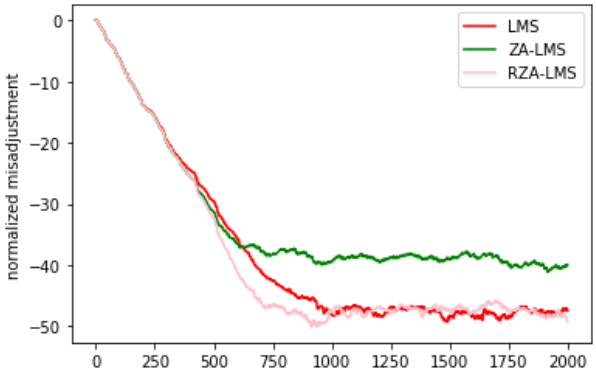
We will now look into how the zero attractor class improve the misalignment on sparse systems.The parameter of the algorithms used are : L=64,μ=0.01,δ_{NLMS} = 0.001,ρ=0.0001,ϵ=5.

It could be seen from figure 3.6, that for sparse systems, ZA-LMS and RZA-LMS tend to achieve lower misalignment. For semi-sparse system, RZA-LMS achieves better misalignment and for dispersive systems, LMS shows better misalignment while RZA-LMS is close to it.

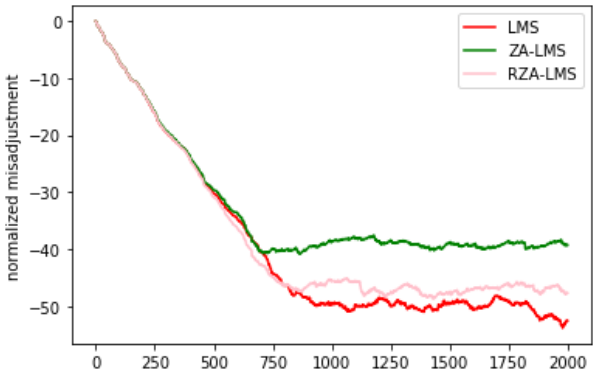
r.5



(a) sparse system,only 2 out of 64 taps are non zero



(b) semi-sparse system,32 out of 64 taps are non zero



(c) dispersive system, none of 64 taps are 0

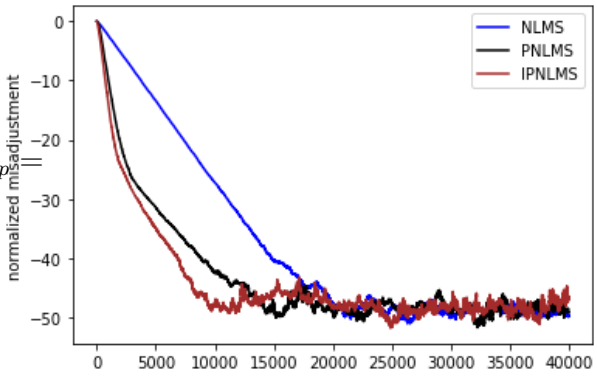
Figure 3.6: Performance of LMS,ZA-LMS and RZA-LMS on sparse,semi-sparse and dispersive systems

We can compare proportionate class algorithms with NLMS for the three kinds of system.

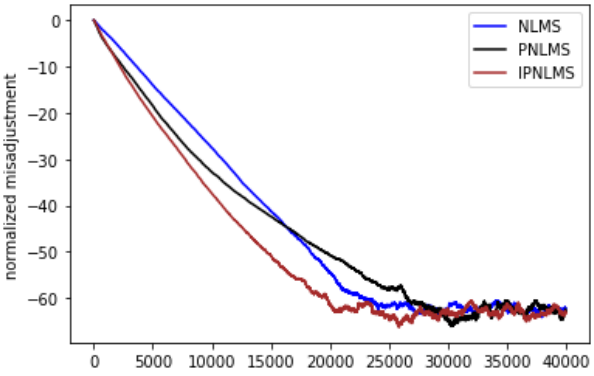
The parameters for the proportionate class algorithms are as follows :L=32,μ=0.01,δ_{NLMS} = 0.001,α=-0.5(for IPNLMS),ϵ = 0.001(for IPNLMS),δ_p= 0.01(for PNLMS).It can be seen that PNLMS and IPNLMS converge faster for sparse systems. However, for semi-sparse and dense systems, PNLMS converges slowly than NLMs. IPNLMS seems to be converging very fast at all times.However, all the three will reach same msialignment eventually.These can be observed visually form 3.7

We can compare the overall performance for all algorithms. It can be seen that LMS class algorithms(LMS,ZA-LMS,RZA-LMS) tend to converge faster than NLMS class(NLMS,PNLMS,IPNLMS),while the latter achieves better misalignment under similar conditions.

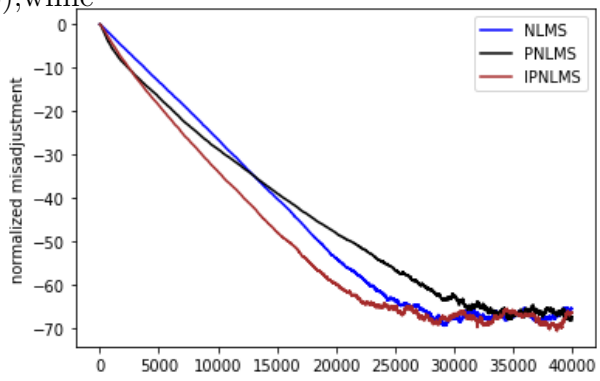
The above simulations are all done for the case of zero white gaussian input. The trends observed may not be similar to all inputs. The nature of the input(colored or white) also infuences the convergence rate. In real applications, most of the data observed is colored. Hence one should keep in mind, the type



(a) sparse system,only 2 out of 64 taps are non zero



(b) semi-sparse system,32 out of 64 taps are non zero



(c) dispersive system, none of 64 taps are 0

Figure 3.7: Performance of proportionate class on sparse,semi-sparse and dense systems

of data, while observing the convergence
rate.

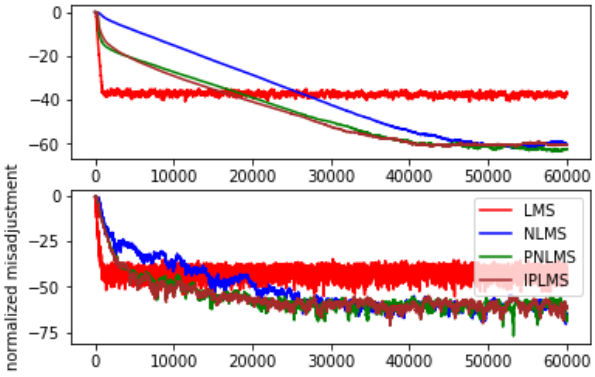
3.2 Adaptive algorithms for Bilinear systems

Bilinear systems present their own set of challenges for adaptive signal processing. However most of the trends observed in linear Adaptive filters holds true for bilinear systems as well.

We compare LMS,NLMS,PNLMS and IPNLMS for sparse, semi-sparse and dispersive systems. The type of system is characterised by whether the temporal filter is sparse,semi-sparse or dispersive.Since, Zero attractor class is not yet studied, we are not interested in its behavior yet.

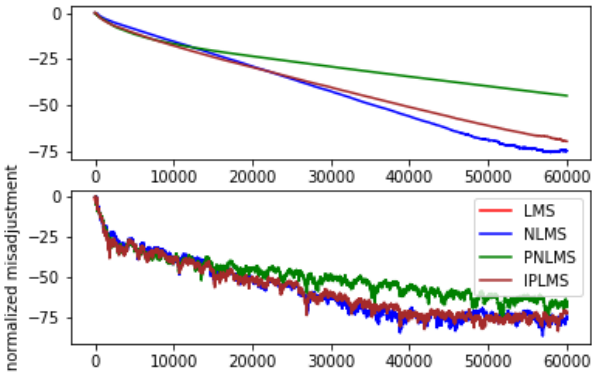
It is also observed that spatial filter converges faster than the temporal filter. It is expected because the temporal filter has larger length(L=64 for this simulation shown) usually than the spatial filter(M=8).

The first half of figure3.8a,3.8b,3.8c corresponds to the temporal filter convergence, while the second half corresponds to spatial filters

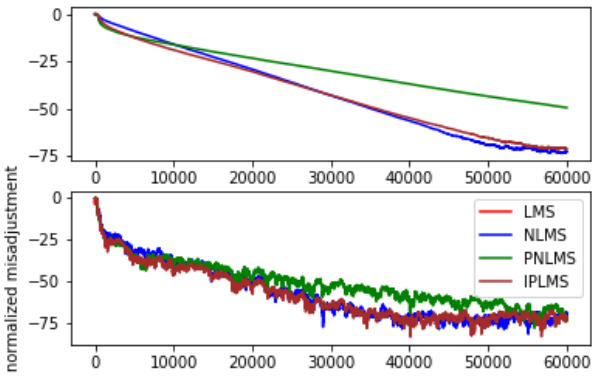


(a) sparse system,only 2 out of 64 taps of temporal filter are non zero

r.5



(b) semi-sparse system,32 out of 64 of temporal filter are non zero



(c) dispersive system, none of 64 taps of temporal are 0

Figure 3.8: Performance of proportionate class on sparse,semi-sparse and dense systems

Bibliography

[1] Benesty, J., and L.Gay, S. An improved pnlms algorithm. *Bell Laboratories,Lucent Technologies* (2002).

[2] Das, R. L., and Chakraborty, M. Improving the performance of the pnlms algorithm using l1 norm regularization. *IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING 24*, 7 (2016).

[3] Das, R. L., and Chakraborty, M. On convergence of proportionate-type normalized least mean square algorithms. *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS 62*, 5 (2016).

[4] Deng, H., and Doroslovacki, M. Improving convergence of the pnlms algorithm for sparse impulse response identification. *IEEE SIGNAL PROCESSING LETTERS 12*, 3 (2005).

[5] Paleologu, C., Benesty, J., Elisei-Iliescu, C., Stanciu, C., Anghel, C., and Ciochin˘, S. A proportionate affine projection algorithm for the identification of sparse bilinear forms. *University of Quebec*.

[6] Paleologu, C., Benesty, J., Elisei-Iliescu, C., Stanciu, C., Anghel, C., and Ciochin˘, S. A proportionate nlms algorithm for the identification of sparse bilinear forms. *University of Quebec*.

[7] Yilun Chen, Yuantao Gu, A. O. H. Sparse lms for system identification.
(7) (1) (4) (2) (6) (5) (3) (**?**)